Vilius Stakėnas

Kodai ir šifrai

Pratarmė

Kas dažniausiai rašoma knygų pratarmėse? Patarimai, kaip skaityti.

Skaitykite, kaip norite. Galiu tik papasakoti, kaip aš skaitau. Pirmą kartą skaitau knygą, norėdamas nuspręsti, ar verta ją skaityti antrą kartą. Skaitydamas antrą kartą, noriu įsitikinti, ar teisingai pasielgiau nusprendęs perskaityti ją vėl. Trečią kartą skaitau norėdamas gerai suprasti, kas knygoje parašyta. Ketvirtą kartą – norėdamas suvokti, kas parašyta nelabai gerai arba neparašyta iš viso. O penktą ir kitus kartus skaitau galvodamas, kaip būtų galima ją perrašyti.

Tikriausiai apie tai galvoti verta skaitant jau pirmąjį kartą. Nes tikras knygos skaitymas ir yra jos „perrašymas“. Net jeigu tai vyksta vien mintyse. Kitoks skaitymas tėra laiko gaišimas. O kokias knygas verta „perrašyti“, arba pergalvoti, ar pergyventi, jeigu norite? Tokias, kurios pasakoja istorijas. Deja, matematinės knygos retai pasakoja istorijas. Dažniausiai jos panašios į laiptus, kuriais tenka kopti nuo vieno apibrėžimo prie kito, nuo vienos teoremos prie kitos. O dažnai – net ir be turėklų!

O ką reiškia pasakoti istoriją? Tai reiškia neskubant atskleisti, kas yra už šito, o paskui – už ano kampo. Kokia kliūtis ir koks netikėtas būdas ją įveikti.

Sritis, kuriai skirta ši knyga – tikrai gera erdvė istorijoms skleistis. Joje lyg kokioje senovės prekybos kelių sankryžoje susitinka iš skirtingų pusių atkeliavusios sąvokos ir idėjos. Ir pati ji skirta nuolatinio judesio, kelionės rūpesčiams – skaitmeninės informacijos perdavimo uždaviniams.

Kelionės – visada sunkumai ir nuotykiai. Klausimai, kas įmanoma, o kas ne. Kaip pasirengti, ką pasiimti, kaip apsisaugoti?

Trumpai tariant, knygoje pasakojama apie patikimo skaitmeninės informacijos perdavimo ir apsaugos uždavinius bei jų sprendimo būdus. Tačiau tai ne receptų rinkinys, o tam tikrų matematinių idėjų sklaida. Galbūt kas nors pamanys, kad neverta vargti bandant jas suprasti.

O aš manau, kad kaip tik tokie žygiai, kurie reikalauja ištvermės bei ryžto, ir yra šio to verti.

Knygos skaitymui reikalingos matematinės žinios. Kiek buvo įmanoma, stengiausi jas priminti, išdėstyti. Tačiau yra skyrių, kuriuose vartojamos matematinės sąvokos ir teiginiai knygoje išsamiau neaptariamos. Pavyzdžiui, prieš skaitant informacijos teorijai skirtą dalį pravartu prisiminti keletą pagrindinių tikimybių teorijos sąvokų ir teiginių: sąlyginę tikimybę, pilnosios tikimybės formulę, atsitiktinio dydžio vidurkį...

Nenuslėpsiu vieno knygos trūkumo – joje nėra uždavinių. Tačiau galima tai pavadinti ir privalumu: knyga plonesnė, o uždavinių ir kitų papildymų rasite šios knygos tinklalapyje ....

NAUJOJI VILNIA 2006

Pastaba: šis tekstas nuo spaudai įteiktojo skiriasi redakcinio pobūdžio (ir rinkimo klaidų) taisymais.

Kodai ir šifrai

INFORMACIJOS TEORIJA

KODAVIMO TEORIJA

Informacijos teorija

Informacija, informacijos kiekis – šias sąvokas dabar vartoja visi. Kai sąvoką vartoja visi, ji neišvengiamai tampa daugiareikšme. Todėl žmonėms ir yra sunku susitarti dėl to, ką kiekvienas gerai išmano: žodžiai tie patys, bet jų reikšmės skiriasi. Vieniems informacija – tai žinios, kurios vertinamos prasmės ir reikšmės kriterijais, kitiems – bet kokie duomenys apie reiškinį, kurie tiriami specialiais metodais... Matematinis požiūris į informaciją iš pirmo žvilgsnio gali pasirodyti keistas. Kalbant apie informaciją, neminima nei reikšmė, nei prasmė, taigi subjektyvaus suvokimo nėra nei šešėlio, tačiau, kita vertus, subjektyvus suvokėjo įspūdis dalyvauja šioje teorijoje daugiau nei šešėlis. Informacijos kiekio sąvoka apibrėžta taip, kad didesnė suvokėjo nuostaba atitinka didesnį gautos informacijos kiekį.

1 Informacijos šaltiniai

1.1. Įvykiai ir įspūdžiai

Įvykio sukeltas įspūdis tuo didesnis, kuo labiau netikėtas yra tas įvykis. Sugalvokime matą netikėtumo dydžiui išreikšti.

Įsivaizduokime kokį nors bandymą, kurio baigtys daugiau ar mažiau priklauso nuo atsitiktinumų, pavyzdžiui, egzaminą. Jeigu gerai pasiruošęs egzaminui studentas bus ir gerai įvertintas – nenustebsime. Tačiau jeigu gerai įvertintas bus tas studentas, kuris nelabai stengėsi pasiruošti – nuostabos bus kur kas daugiau.

Abu įvykiai nėra vienodai tikėtini, todėl ir mūsų reakcija į juos skiriasi.

Ar galima sugalvoti būdą įvykio sukeltai nuostabai išmatuoti? Pabandykime pasvarstyti, kokias savybes privalėtų turėti toks įvykio sukeltos „nuostabos matas“.

1. Įvykio A „nuostabos matas“ turi būti tolydi įvykio tikimybės $p = P(A)$ funkcija $f(p)$, įgyjanti neneigiamas reikšmes.

2. Kadangi mažiau tikėtini įvykiai stebina labiau, tai funkcija $f(p)$ turi būti nedidėjanti intervale $(0; 1]$.

3. Jeigu tuo pačiu metu įvyksta du nepriklausomi įvykiai, tai jų sukelta nuostaba turi būti lygi abiejų įvykių skyrium sukeltų nuostabų sumai, t. y. turi būti

$$f(p \cdot q) = f(p) + f(q), \quad p, q \in (0, 1].$$

4. Įvykiai, kurie visada įvyksta, mūsų nestebina, taigi $f(1) = 0$.

Galima įrodyti, kad visas išvardytas sąlygas tenkina tik vienos šeimos funkcijos.

1 teorema. *Jeigu funkcija $f(p)$, apibrėžta intervale $(0; 1]$, tenkina 1)-4) sąlygas, tai egzistuoja toks $b > 1$, kad*

$$f(p) = \log_b \frac{1}{p}. \tag{1}$$

Kita vertus, kiekvienam $b > 1$ funkcija $f(p)$, apibrėžta (1), tenkina 1)-4) sąlygas.

Įrodymas. Kad (1) funkcijos tenkina visas išvardytas sąlygas, įsitikinti nesunku prisiminus logaritminės funkcijos savybes.

Įrodysime, kad kitokių funkcijų, tenkinančių 1)-4) sąlygas, nėra.

Tarkime, funkcija f tenkina šias sąlygas. Apibrėžkime

$$h(u) = f(\mathrm{e}^{-u}), \quad u \geq 0.$$

Jeigu surastume funkciją $h(u)$, tai, pasinaudoję ja, gautume:

$$f(p) = h\Big(\ln \frac{1}{p}\Big), \quad 0 < p \leq 1. \tag{2}$$

Įrodysime, kad funkcija h yra tiesinė, t. y. $h(u_1 + u_2) = h(u_1) + h(u_2)$. Tai išplaukia iš funkcijos f ketvirtosios savybės:

$$h(u_1 + u_2) = f(\mathrm{e}^{-u_1} \cdot \mathrm{e}^{-u_1}) = f(\mathrm{e}^{-u_1}) + f(\mathrm{e}^{-u_1}) = h(u_1) + h(u_2).$$

Pasinaudoję matematine indukcija, gauname, kad lygybė

$$h(u_1 + u_2 + \ldots + u_k) = h(u_1) + h(u_2) + \ldots + h(u_k)$$

teisinga bet kokiam neneigiamų skaičių rinkiniui $u_1, u_2, \ldots, u_k$.

Įrodysime, kad visiems teigiamiems racionaliesiems skaičiams $u = \frac{m}{n}$ teisinga lygybė $h(u) = h(1)u$. Pasinaudoję funkcijos tiesiškumu, gauname

$$\begin{aligned} h\Big(\frac{m}{n}\Big) &= h\Big(\frac{1}{n} + \frac{1}{n} + \ldots + \frac{1}{n}\Big) = m \cdot h\Big(\frac{1}{n}\Big), \\ h(1) &= h\Big(\frac{1}{n} + \frac{1}{n} + \ldots + \frac{1}{n}\Big) = n \cdot h\Big(\frac{1}{n}\Big), \\ h\Big(\frac{1}{n}\Big) &= h(1) \cdot \frac{1}{n}, \quad h\Big(\frac{m}{n}\Big) = h(1) \cdot \frac{m}{n}. \end{aligned}$$

Jeigu $u > 0$ yra bet koks skaičius, o u_n ir v_n dvi į u konverguojančios racionaliųjų skaičių sekos, $u_n < u < v_n$, tai

$$h(u_n) < h(u) < h(v_n), \quad h(1) \cdot u_n < h(u) < h(1) \cdot v_n.$$

Imdami ribines sekų reikšmes, gauname $h(u) = h(1) \cdot u$. Kadangi $h(1) > 0$, tai galime parinkti $b > 1$, kad būtų

$$h(1) = \frac{1}{\ln b}.$$

Tada iš (2) gausime, kad f yra tokia kaip (1) .

Nuostabos dydžio matavimo funkciją galime pasirinkti iš begalinės šeimos. Pasirinkime $b = 2$, tada

$$f\left(\frac{1}{2}\right) = \log_2 \frac{1}{2} = 1.$$

Taigi nuostabos, kurią patiriame, kai simetriška moneta atvirsta herbu į viršų, matas lygus vienetui.

O kokio didumo nuostabą patiriame sužinoję, kad įvyko įvykis, kurio tikimybė lygi nuliui? Labai didelę? Nebūtinai. Jeigu mestas akmuo ant žemės nenukris, tačiau pakibs ore, tikrai labai nustebsime. Tačiau galima pateikti ir kitokių pavyzdžių.

Tarkime, bandymas – intervalo $[0; 1]$ skaičiaus parinkimas. Jeigu bus parinktas, pavyzdžiui, skaičius $2/3$, niekas ypatingai nenustebs. Tačiau juk tokio įvykio tikimybė lygi nuliui! Taigi tokiems atvejams reiktų sukurti kitokį „nuostabos matavimo“ matematinį modelį.

1.2. Informacijos šaltiniai ir kiekiai

Kaip išmatuoti informacijos kiekį, kurį mums perduoda šaltinis? Tikrai ne pavartotų ženklų skaičiumi, nes, pavyzdžiui, septynių ženklų formulė $a \cdot a \cdot a \cdot a$ nepasako mums daugiau nei a^4.

Puikiausiai žinome: ne knygos puslapių skaičius lemia knygos vertę. Pavartotų ženklų skaičius ir jais perduotas informacijos kiekis toli gražu ne tas pats. Kada manome, kad gavome daug informacijos? Kai sužinome daug naujo, daug tokių dalykų, kurių iki šiol nežinojome, kitaip tariant – kai įgyjame daug netikėtų žinių.

Įvykio „netikėtumo matą“ jau turime, belieka jį susieti su informacijos šaltinio sąvoka.

Žinias reiškiame ženklais, t. y. abėcėlės simboliais. Tegu

$$\mathcal{A} = \{a_1, a_2, \ldots\}$$

kokia nors abėcėlė, daugtaškis reiškia, kad abėcėlė gali būti ir begalinė. Tačiau dažniausiai mums pakaks dvejetainės abėcėlės $\mathcal{B} = \{0, 1\}$.

Informacijos šaltinis – tam tikras procesas, kuris parenka mums šios abėcėlės simbolį. Koks tai procesas – nelabai ką galime pasakyti, tiesą sakant, matematiniam tyrinėjimui tai nėra svarbu. Todėl verčiau jo iš viso

jo neminėkime. Tiesiog tapatinkime informacijos šaltinį su diskrečiuoju atsitiktiniu dydžiu, įgyjančiu reikšmes iš aibės (abėcėlės) $\mathcal{A} = \{a_1, a_2, \ldots\}$. Tai – diskretusis informacijos šaltinis. Jeigu vietoj skaičios abėcėlės $\mathcal{A}$ imtume kokią nors begalinę neskaičią aibę, pavyzdžiui, realiųjų skaičių aibę ar intervalą, galėtume apibrėžti nediskretųjį informacijos šaltinį.

Gautos iš dydžio X informacijos kiekis priklauso nuo jo įgytos reikšmės tikimybės. Viso šaltinio informatyvumą galime matuoti reikšmių suteikiamų informacijos kiekių vidurkiu.

1 apibrėžimas. *Diskrečiojo atsitiktinio dydžio X, įgyjančio reikšmes iš baigtinės abėcėlės, entropija vadinsime skaičių*

$$H(X) = \sum_{x, P(X=x)>0} \log_2 \frac{1}{P(X=x)} \cdot P(X=x).$$

Žinoma, šitaip galime apibrėžti ir dydžio, įgyjančio reikšmes iš begalinės, tačiau skaičios abėcėlės, entropiją, bet ji ne visada būtų baigtinė. Kitas klausimas – kaip reiktų apibrėžti atsitiktinio dydžio, įgyjančio reikšmes iš, pavyzdžiui, realiųjų skaičių aibės, entropiją? Tokio atvejo taip pat neketiname tyrinėti, tačiau jei įdomu – žvilgtelkite į knygas, nurodytas literatūros sąraše.

Taigi jei informacijos šaltinį sutapatiname su atsitiktiniu dydžiu, tai dydžio entropija išreiškia šaltinio informatyvumą, matuoja patiriamą jo atžvilgiu „netikėtumą“. Atsitiktinio dydžio entropiją vadinsime tiesiog informacijos šaltinio entropija.

Pavyzdys. Metama moneta nukrenta herbu į viršų su tikimybe p. Ruošiamasi mesti ją n kartų, mums bus pranešti visų metimų rezultatai. Koks tokio informacijos šaltinio informatyvumas, t. y. entropija?

Rezultatai, kurie bus mums pranešti – ženklų H, S (herbas, skaičius) n ilgio sekos. Tai ir yra mūsų informacijos šaltinio abėcėlės simboliai. Jeigu sekoje $x = x_1 x_2 \ldots x_n$ herbo ženklas pasitaiko m kartų, tai tokios sekos tikimybė

$$P(X=x) = p^m q^{n-m}, \quad q = 1-p.$$

Taigi šaltinio entropija

$$\begin{aligned} H(X) &= \sum_{m=0}^{n} C_n^m p^m q^{n-m} \log_2 \frac{1}{p^m q^{n-m}} = \log_2 \frac{1}{p} \sum_{m=0}^{n} m C_n^m p^m q^{n-m} + \\ &\log_2 \frac{1}{q} \sum_{m=0}^{n} (n-m) C_n^m p^m q^{n-m} = n\left(\log_2 \frac{1}{p} + \log_2 \frac{1}{q} \right). \end{aligned}$$

Jeigu mums būtų pranešta tiktai, kiek kartų iškrito herbas, turėtume kitą informacijos šaltinį Y, kurio reikšmės – skaičiai $\{0, 1, \ldots, n\}$. Tokio šaltinio informatyvumas, t. y. entropija, būtų mažesnis. Nesunku suskaičiuoti, kad

$$H(Y) = H(X) - \sum_{m=0}^{n} C_n^m p^m q^{n-m} \log_2 C_n^m.$$

Tarkime, dviejų šaltinių abėcėlės yra vienodo dydžio, o simbolių tikimybės atitinkamai

$$p_1, p_2, ..., p_n \quad \text{ir} \quad q_1, q_2, ..., q_n.$$

Pirmojo šaltinio simbolių suteikiami informacijos kiekiai yra $\log_2 \frac{1}{p_i}$, antrojo – $\log_2 \frac{1}{q_i}$. Suskaičiavę pirmojo šaltinio entropiją, gautume dydį

$$H = \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i}.$$

O kokį dydį gautume, jeigu, skaičiuodami pirmojo šaltinio entropiją, pasinaudotume antrojo šaltinio simbolių suteikiamais informacijos kiekiais? Atsakymas, pasirodo, toks – visada ne mažiau už H. Taip būna ir gyvenime: žinios apie paprastą įvykį, praneštos jo nemačiusių žmonių, gali virsti tikra sensacija.

2 teorema. *Tegu*

$$p_1, p_2, ..., p_n \quad \textit{ir} \quad q_1, q_2, ..., q_n$$

yra du tikimybiniai skirstiniai, t. y. teigiamų skaičių, kurių suma lygi vienetui, sekos. Tada

$$\sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i} \le \sum_{i=1}^{n} p_i \log_2 \frac{1}{q_i}. \tag{3}$$

Įrodymas. Pastebėkime, kad pakanka įrodyti nelygybę, gautą iš (3), pakeitus joje dvejetainius logaritmus natūriniais. Jeigu tokios nelygybės abi puses padauginsime iš $1/\ln 2$, gausime (3) nelygybę.

Nesunku įsitikinti, kad funkcija $f(x) = x - 1 - \ln x$, apibrėžta intervale $(0; +\infty)$, įgyja mažiausią reikšmę, kai $x = 1$, taigi

$$f(1) \le f(x), \quad \text{arba} \quad \ln x \le x - 1,$$

ir lygybė teisinga tik tada, kai $x = 1$.

Įrodymui užbaigti reikia tik kelių eilučių:

$$\ln \frac{q_i}{p_i} \le \frac{q_i}{p_i} - 1, \quad (i = 1, 2, \ldots, n),$$

$$\sum_{i=1}^{n} p_i \ln \frac{q_i}{p_i} \le \sum_{i=1}^{n} p_i \Big(\frac{q_i}{p_i} - 1 \Big) = 0,$$

$$\sum_{i=1}^{n} p_i \ln \frac{1}{p_i} \le \sum_{i=1}^{n} p_i \ln \frac{1}{q_i}.$$

Pastaba. Įsižiūrėję į įrodymą, greitai pastebėsime: kad nelygybė būtų teisinga, netgi nebūtina, kad skaičių q_i suma būtų lygi vienetui, pakanka, kad galiotų nelygybė

$$q_1 + q_2 + \ldots + q_n \le 1.$$

Jeigu įstatysime į (3) $q_i = 1/n$, gausime

$$\sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i} \leq \log_2 n.$$

Jeigu ir $p_i = 1/n$, tai ši nelygybė virsta tikslia lygybe. Taigi teisingas toks teiginys.

3 teorema. *Jeigu atsitiktinis dydis X įgyja n reikšmių, tai*

$$H(X) \leq \log_2 n.$$

Lygybė $H(X) = \log_2 n$ teisinga tada ir tik tada, kai visos reikšmės įgyjamos su vienodomis tikimybėmis.

1.3. Šaltiniai be atminties ir su ja

Matematikai netyrinėja tikrovės reiškinių! Jie tyrinėja reiškinių modelius. Taigi reikia nutarti, kokį informacijos šaltinio modelį pasirinksime.

Šaltinį tapatinome su atsitiktiniu dydžiu, įgyjančiu reikšmes iš abėcėlės. Šaltinį, kuris išsenka pateikęs vos vieną ženklą, nagrinėti nelabai įdomu. Toliau nagrinėsime tokius šaltinius, kurie niekada „nepavargsta“, t. y. gali generuoti simbolius be paliovos.

2 apibrėžimas. *Informacijos šaltiniu vadinsime atsitiktinių dydžių, įgyjančių reikšmes iš tos pačios abėcėlės $\mathcal{A}$, seką $U_1, U_2, \ldots$*

Iš pirmųjų n atsitiktinių dydžių galime sudaryti vektorių

$$U^{(n)} = \langle U_1, U_2, \ldots, U_n \rangle.$$

Šio vektoriaus reikšmės – n ilgio abėcėlės $\mathcal{A}$ žodžiai, juos žymėsime tiesiog $u = u_1 u_2 \ldots u_n$.

Dydžius U_m gali sieti įvairiausi ryšiai: generuodamas m-ąjį simbolį, šaltinis gali „atsiminti“, kokius simbolius pateikė mums anksčiau. Paprasčiausias atvejis – kai šaltinis visai neturi atminties.

3 apibrėžimas. *Jeigu atsitiktiniai dydžiai U_m yra nepriklausomi, tai sakysime, kad šaltinis neturi atminties. Jeigu neturinčio atminties šaltinio dydžiai U_m yra vienodai pasiskirstę, jį vadinsime Bernulio šaltiniu.*

Štai paprasčiausias Bernulio šaltinio pavyzdys. Mėtykime tą pačią monetą ir užsirašykime rezultatus: 0 – jei atvirto herbas, 1 – jei skaičius. Gautieji dvejetainės abėcėlės žodžiai – tokio šaltinio perduoti duomenys. Jeigu mėtysime šešiasienį kauliuką ir rašysime rezultatus, gausime Bernulio šaltinio su šešių simbolių abėcėle generuotus žodžius...

Neturėti atminties galima vienu būdu, o ją turėti – daugeliu. Apibrėšime paprasčiausias šaltinių, turinčių atmintį, rūšis. Šaltinio atminties savybėms nusakyti naudosime sąlygines tikimybes.

4 apibrėžimas. *Jeigu su visomis $n > 1$ ir $a_{i_j} \in \mathcal{A}$ reikšmėmis atsitiktiniai dydžiai tenkina sąlygą*

$$P(U_n = a_{i_n} | U_{n-1} = a_{i_{n-1}}, \ldots, U_1 = a_{i_1}) = P(U_n = a_{i_n} | U_{n-1} = a_{i_{n-1}}),$$

tai šaltinį vadinsime Markovo šaltiniu.

Taigi Markovo šaltinio atmintis labai „trumpa“: generuojamas simbolis priklauso tik nuo vieno prieš jį generuoto simbolio. Ta atmintis gali „silpti“ arba „stiprėti“, paprasčiausiu atveju – ji nesikeičia, t. y. „perėjimo“ tikimybės

$$p_{ij} = P(U_n = a_i | U_{n-1} = a_j)$$

nepriklauso nuo n. Tokį Markovo šaltinį vadinsime stacionariuoju.

Apibrėžkime Markovo šaltinį, kurio atminties gylis yra m.

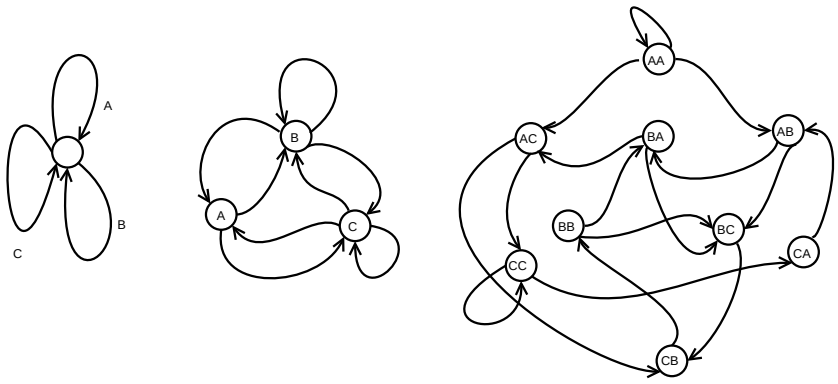
5 apibrėžimas. *Tegu $m \geq 1$ yra fiksuotas natūrinis skaičius, o atsitiktiniai dydžiai U_j su visomis $n > m$ ir $a_{i_j} \in \mathcal{A}$ reikšmėmis tenkina sąlygą*

$$\begin{aligned} &P(U_n = a_{i_n} | U_{n-1} = a_{i_{n-1}}, \ldots, U_1 = a_{i_1}) = \\ &P(U_n = a_{i_n} | U_{n-1} = a_{i_{n-1}}, U_{n-2} = a_{i_{n-2}}, \ldots, U_{n-m} = a_{i_{n-m}}). \end{aligned}$$

Tokį šaltinį vadinsime m-osios eilės Markovo šaltiniu.

Taigi m-os eilės Markovo šaltinis „prisimena“ m jau generuotų simbolių. Jeigu tos atminties „pobūdis“, t. y. atitinkamos n-ojo simbolio sąlyginės tikimybės, nesikeičia, tai m-osios eilės Markovo šaltinis yra stacionarus.

Informacijos teorijos pagrindus sukūrė C. Shannonas. Pagrindinės sąvokos ir teiginiai išdėstyti jo darbe „Mathematical theory of Communication“. Jame pateiktos ir Bernulio bei Markovo šaltinių sąvokos. C. Shannonas parodė, kaip šie šaltiniai gali būti vaizduojami būsenų diagramomis. Šaltinio būseną nusako jo „atmintis“, taigi šaltinio būsenų kiekis lygus jo „atsimenamų“ žodžių skaičiui. Perdavus simbolį, pasikeičia atminties turinys, t. y. šaltinio būsena, žr. brėžinį.



Diagramomis pavaizduoti trys šaltiniai, perduodantys abėcėlės $\mathcal{A} = \{A, B, C\}$ simbolius. Pirmasis šaltinis – Bernulio, kiti du – Markovo.

Pirmojoje diagramoje pavaizduotas Bernulio šaltinis. Jis neturi atminties, todėl gali būti vienintelės „tuščios galvos“ būsenos. Prie linijų, rodančių grįžimą į tą pačią būseną, parašyti šaltinio perduodami simboliai. Norėdami išsamiai aprašyti šaltinį tokia diagrama, turėtume prie linijų prirašyti šių simbolių perdavimo tikimybes. Kitos dvi diagramos vaizduoja pirmosios ir antrosios eilės Markovo šaltinius. Pirmosios eilės Markovo šaltinis gali būti trijų būsenų. Būseną nusako tie simboliai, kuriuos šaltinis atsimena. Prie perėjimo linijų nėra parašyti perduodami simboliai, nes jie sutampa su naują būseną reiškiančiu simboliu. Antrosios eilės Markovo šaltinio diagramoje taip pat nebūtina rašyti perduodamus simbolius: jie sutampa su naujos būsenos paskutiniuoju simboliu. Tačiau prie perėjimo linijų reiktų parašyti atitinkamas perėjimo tikimybių reikšmes.

Kaip nusakyti mūsų „niekada neišsenkančio“ šaltinio informatyvumą, t. y. entropiją?

Pirmųjų n generuotų simbolių žodis yra atsitiktinio vektoriaus $U^{(n)}$ reikšmė. Galime apskaičiuoti entropiją $H(U^{(n)}) = H(U_1, U_2, \ldots, U_n)$. Jeigu šaltinis nepavargs ir nepradės „kartotis“, tai tikriausiai entropija $H(U^{(n)})$ neaprėžtai didės. Paprasčiausia idėja bandant surasti visos begalinės sekos informatyvumą – nagrinėti, kaip keičiasi santykis $H(U^{(n)})/n$.

6 apibrėžimas. *Jeigu egzistuoja baigtinė riba*

$$H = \lim_{n\to\infty} \frac{H(U_1, U_2, \ldots, U_n)}{n},$$

tai skaičių H vadinsime šaltinio entropija.

Tačiau kaipgi apskaičiuoti tą entropiją?

1.4. Sąlyginės entropijos

Nustatysime kelias labai svarbias informacijos kiekio savybes. Vieną jų, bene pačią svarbiausią, visi gerai žinome. Kaip nusakyti žinias, kurias mums suteikė laikraščių $X_1, X_2, \ldots, X_n$ šūsnis? Žinios, kurias suradome X_1, žinios iš X_2, kurių nebuvo X_1, žinios iš X_3, kurių neradome X_1, X_2, ir taip toliau...

Įsivaizduokime, kad laukiame, kokią reikšmę įgis diskretusis atsitiktinis dydis X, pavyzdžiui, kokią vietą užims Lietuvos krepšinio komanda pasaulio čempionate. Neapibrėžtumo, netikrumo matu galime laikyti atsitiktinio dydžio entropiją $H(X)$. Tarkime, sužinojome kito atsitiktinio dydžio Y reikšmę y, pavyzdžiui, sužinojome, kad Lietuva laimėjo rungtynes prieš JAV komandą 10 taškų skirtumu. Aišku, kad apie galutinę komandos vietą lentelėje galvosime jau kitaip: galbūt klaustukų sumažės, o gal priešingai – tik padaugės.

Naują būseną galime apibūdinti sąlygine dydžio X entropija su sąlyga, kad dydis Y įgijo reikšmę y.

7 apibrėžimas. *Sąlygine diskrečiojo atsitiktinio dydžio X entropija su sąlyga, kad Y įgijo reikšmę y, vadinsime skaičių*

$$H(X|Y=y)=\sum_x P(X=x|Y=y)\log_2\frac{1}{P(X=x|Y=y)}.$$

Sąlygine atsitiktinio dydžio X entropija atsitiktinio dydžio Y atžvilgiu vadinsime skaičių

$$H(X|Y)=\sum_y H(X|Y=y)P(Y=y).$$

Sąlyginę entropiją galime apibrėžti ne tik vieno dydžio, bet ir kelių dydžių, t. y. atsitiktinio vektoriaus, atžvilgiu.

Įdomu, kad sužinota dydžio Y reikšmė gali padidinti neapibrėžtumą X atžvilgiu, t. y. gali būti teisinga nelygybė $H(X|Y=y)>H(X)$. Sužinoję apie vieną didelį laimėjimą, galime pagalvoti, kad loterija yra geresnis būdas pinigams išleisti, negu iki šiol manėme, tačiau išsamesni duomenys apie laimėjimus galbūt paskatins susidaryti dar skeptiškesnę nuomonę, t. y. visada teisinga nelygybė $H(X|Y)\le H(X)$.

Štai tikriausiai pats paprasčiausias pavyzdys. Tegu yra dvi vienodos urnos, vienoje – du juodi rutuliai, kitoje – vienas baltas, kitas juodas. Pirmoji urna pažymėta skaičiumi 1, antroji - skaičiumi 2. Atsitiktinai pasirenkama viena iš urnų, iš jos ištraukiamas rutulys. Dydžio X reikšmė – rutulio spalva, dydžio Y – urnos numeris. Nesunku įsitikinti, kad

$$P(X=\text{balta})=\frac{1}{4},\quad P(X=\text{juoda})=\frac{3}{4}.$$

Taigi $H(X)=\frac{1}{2}+\frac{3}{4}\log_2\frac{4}{3}\approx 0.8113$. Tačiau

$$H(X|Y=1)=0,\quad H(X|Y=2)=1,\quad H(X|Y)=0.5.$$

4 teorema. *Visiems diskretiesiems atsitiktiniams dydžiams teisingos nelygybės*

$$H(X|Y)\le H(X),\quad H(X|Y,Z)\le H(X|Y).$$

Pirmoji nelygybė virsta tikslia lygybe tada ir tik tada, kai X ir Y yra nepriklausomi.

Įrodymas. Įrodysime tik pirmąją nelygybę, antrosios įrodymas analogiškas.

Pagal sąlyginės entropijos apibrėžimą

$$H(X|Y)=\sum_y H(X|Y=y)P(Y=y), \tag{4}$$

$$H(X|Y=y) \;=\; \sum_x P(X=x|Y=y)\log_2 \frac{1}{P(X=x|Y=y)}. \qquad (5)$$

Pritaikę nelygybę (3) su $p_i = P(X=x|Y=y)$ ir $q_i = P(X=x)$ sąlyginei entropijai (5), gausime

$$H(X|Y=y) \le \sum_x P(X=x|Y=y)\log_2 \frac{1}{P(X=x)}. \qquad (6)$$

Ši nelygybė virsta tikslia lygybe tada ir tik tada, kai

$$P(X=x|Y=y) = P(X=x).$$

Tai teisinga tada ir tik tada, kai atsitiktiniai dydžiai X, Y yra nepriklausomi. Įrodymui užbaigti pakanka pasinaudoti įverčiu (6) sąlyginės entropijos išraiškoje (4):

$$\begin{aligned} H(X|Y) &\le \sum_y P(Y=y)\sum_x P(X=x|Y=y)\log_2 \frac{1}{P(X=x)} \\ &= \sum_x \log_2 \frac{1}{P(X=x)} \sum_y P(Y=y)P(X=x|Y=y) \\ &= \sum_x \log_2 \frac{1}{P(X=x)} P(X=x) = H(X). \end{aligned}$$

Diskretųjį atsitiktinį dydį X, įgyjantį reikšmes iš aibės $\mathcal{A}^m$, vadinsime diskrečiuoju m-mačiu vektoriumi. Galime įsivaizduoti, kad atsitiktinis vektorius X – tai atsitiktinių dydžių rinkinys

$$X = \langle X_1, X_2, \ldots, X_m\rangle, \quad X_i \text{ įgyja reikšmes iš } \mathcal{A}. \qquad (7)$$

Diskrečiojo atsitiktinio vektoriaus entropija apibrėžiama taip pat kaip ir atsitiktinio dydžio, juk atsitiktinį vektorių galime interpretuoti kaip atsitiktinį dydį, įgyjantį reikšmes iš atitinkamo ilgio žodžių aibės. Atsitiktinio vektoriaus $X = \langle X_1, X_2, \ldots, X_n\rangle$ entropiją žymėsime $H(X) = H(X_1, X_2, \ldots, X_n)$, atsitiktinio dydžio (arba vektoriaus) Y sąlyginę entropiją atsitiktinio vektoriaus X atžvilgiu $H(Y|X) = H(Y|X_1, X_2, \ldots, X_n)$.

5 teorema. *Bet kokiems diskretiesiems atsitiktiniams dydžiams teisinga lygybė*

$$H(X_1, \ldots, X_n) = H(X_1) + H(X_2|X_1) + \ldots + H(X_n|X_1, \ldots, X_{n-1}). \qquad (8)$$

Lygybę (8) vadinsime entropijų grandinės taisykle. Jos prasmė turėtų būti gana aiški. Žinias, kurias įgyjame perskaitę laikraščius $X_1, X_2, \ldots, X_n$,

galime išdėstyti į grandinę taip:

žinios iš laikraščioX_1(entropija$H(X_1)$) +
žinios iš laikraščioX_2, kurių nebuvo laikraštyje X_1, ir t. t.

Įrodymas. Iš pradžių įrodykime lygybę dviejų diskrečiųjų dydžių (vektorių) atveju:

$$H(X,Y) = H(X) + H(Y|X). \tag{9}$$

Patogumo dėlei žymėkime:

$$\begin{aligned} p(x,y) &= P(X=x, Y=y),\ p(x) = P(X=x),\ p(y) = P(Y=y), \\ p(y|x) &= P(Y=y|X=x). \end{aligned}$$

Kadangi $p(x,y) = p(x)p(y|x)$, tai

$$\log_2 \frac{1}{p(x,y)} = \log_2 \frac{1}{p(x)} + \log_2 \frac{1}{p(y|x)}.$$

Tada

$$H(X,Y) = \sum_{x,y} p(x,y) \log_2 \frac{1}{p(x,y)} = \sum_{x,y} p(x,y) \log_2 \frac{1}{p(x)} + \sum_{x,y} p(x,y) \log_2 \frac{1}{p(y|x)}.$$

Tačiau

$$\sum_{x,y} p(x,y) \log_2 \frac{1}{p(x)} = \sum_x \log_2 \frac{1}{p(x)} \sum_y p(x,y) = \sum_x \log_2 \frac{1}{p(x)} p(x) = H(X),$$

o

$$\begin{aligned} \sum_{x,y} p(x,y) \log_2 \frac{1}{p(y|x)} &= \sum_x p(x) \sum_y p(y|x) \log_2 \frac{1}{p(y|x)} = \\ \sum_x p(x) H(Y|X=x) &= H(Y|X). \end{aligned}$$

Taigi (9) lygybė įrodyta.

Bendrąjį grandinės taisyklės atvejį dabar nebesunku įrodyti matematine indukcija. Kadangi atveju $n = 2$ lygybę įrodėme, tai pakanka įsitikinti, kad iš (8) lygybės su $n = m$ išplaukia lygybė su $n = m + 1$.

Jeigu pritaikysime (9) su $X = \langle X_1, X_2, \ldots, X_m \rangle$ ir $Y = X_{m+1}$, gausime

$$H(X_1, X_2, \ldots, X_m, X_{m+1}) = H(X_1, X_2, \ldots, X_m) + H(X_{m+1}|X_1, X_2, \ldots, X_m).$$

Pasinaudoję indukcijos prielaida, gausime (8) su $n = m + 1$.

Entropiją $H(X,Y)$ galime išreikšti dvejopai:

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y). \tag{10}$$

Iš šių lygybių gauname:

$$H(X) - H(X|Y) = H(Y) - H(Y|X).$$

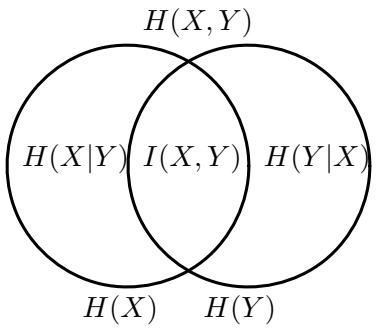
Pasinaudoję 4 teorema, gauname, kad $H(X) - H(X|Y) \geq 0$. Šį dydį galime paprastai interpretuoti: dydis $H(X)$ yra žinių, kurias suteikia X, matas, dydis $H(X|Y)$ – žinių, kurias suteikia X, bet ne Y, matas. Vadinasi, skirtumas $H(X) - H(X|Y)$ yra matas tų žinių, kurias suteikia tiek X, tiek Y.

8 apibrėžimas. *Tegu X ir Y yra diskretieji dydžiai (vektoriai). Abipusės informacijos kiekiu vadinsime dydį*

$$I(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X). \tag{11}$$

Iš (10) ir (11) gauname:

$$H(X,Y) = H(X) + H(Y) - I(X,Y).$$



Visus svarbiausius dviejų dydžių entropijų sąryšius paprasta prisiminti pasinaudojus šia diagrama. Tiesiog įsivaizduokite, kad dydžiai reiškia atitinkamų sričių plotus, rašykite jų sąryšius ir nesuklysite.

1.5. Informacijos šaltinių entropijos

Apibrėšime informacijos šaltinio informatyvumo matą ir jį apskaičiuosime, kai šaltinis pats paprasčiausias – Bernulio.

Kaip ir anksčiau atsitiktinių dydžių, įgyjančių reikšmes baigtinėje abėcėlėje, seką $\mathcal{U} = \langle U_1, U_2, \ldots \rangle$ interpretuokime kaip informacijos šaltinį. Apibrėžkime jo entropiją (informatyvumą) taip:

9 apibrėžimas. *Jeigu egzistuoja riba*

$$\lim_{n\to\infty} \frac{H(U_1, U_2, \ldots, U_n)}{n},$$

tai jos reikšmę vadinsime šaltinio entropija ir žymėsime $H(\mathcal{U})$.

Prisiminkime grandinės taisyklę:

$$H(U_1, U_2, \dots U_n) = H(U_1) + H(U_2|U_1) + \dots + H(U_n|U_1, U_2, \dots U_{n-1}). \quad (12)$$

Kai šaltinis neturi atminties, tai

$$H(U_1, U_2, \dots U_n) = H(U_1) + H(U_2) + H(U_3) + \dots + H(U_n).$$

Jeigu $\mathcal{U}$ yra Bernulio šaltinis, tai visi atsitiktiniai dydžiai U_i yra vienodai pasiskirstę, todėl ir jų entropijos lygios:

$$H(U_1, U_2, \dots U_n) = nH(U_1).$$

Taigi gavome paprasčiausio šaltinio entropijos reikšmę:

6 teorema. *Bernulio šaltinio* $\mathcal{U} = \langle U_1, U_2, \dots \rangle$ *entropija yra*

$$H(\mathcal{U}) = H(U_1).$$

Kad atsitiktinių dydžių entropija yra labai svarbi sąvoka įvairiuose sąryšiuose, dar ne kartą pamatysime. Panagrinėkime, kaip ja galima išreikšti vieną svarbią Bernulio šaltinių savybę.

Tegu $\mathcal{A} = \{a_1, a_2, \dots, a_r\}$ yra šaltinio abėcėlė, o

$$p_1 = P(U_1 = a_1),\ p_2 = P(U_1 = a_2),\ \dots,\ p_r = P(U_1 = a_r)$$

simbolių pasirodymo tikimybės. Tarkime, $\mathbf{x} = x_1 x_2 \dots x_n$ yra šaltinio perduotas simbolių srautas, t. y. atsitiktinių dydžių $U_1, U_2, \dots, U_n$ reikšmių seka. Suskaičiavę, kiek kartų šiame sraute pasitaiko abėcėlės simboliai $a_1, a_2, \dots, a_r$, gausime sveikųjų neneigiamų skaičių (simbolių dažnių) seką

$$n_1, n_2, \dots, n_r, \quad n_1 + n_2 + \dots + n_r = n.$$

Jeigu daug kartų mestume simetrišką monetą ir rašytume rezultatus, t. y. fiksuotume, ar moneta atvirto herbu, ar skaičiumi į viršų, dažniausiai gautume tokias sekas, kuriose herbo ir skaičiaus pasirodymų dažniai būtų apylygiai skaičiai. Tai tipinės simetriniškos monetos n metimų rezultatų sekos. Kitokios sekos pasitaikytų labai retai.

Analogiškai tipinės mūsų Bernulio šaltinio generuotos simbolių sekos būtų tokios, kurioms

$$\frac{n_1}{n} \approx p_1,\ \frac{n_2}{n} \approx p_2,\ \dots,\ \frac{n_r}{n} \approx p_r.$$

Netipinės sekos (kai n didelis skaičius) pasirodytų labai retai.

Nagrinėkime kokią nors tipinę Bernulio šaltinio seką $\mathbf{x} = x_1 x_2 \dots x_n$. Įvertinsime tokios sekos pasirodymo tikimybę:

$$\begin{aligned} P(U^{(n)} = \mathbf{x}) &= P(U_1 = x_1) P(U_2 = x_2) \cdots P(U_n = x_n) \\ &= p_1^{n_1} p_2^{n_2} \cdots p_r^{n_r} = \left(p_1^{\frac{n_1}{n}} p_2^{\frac{n_2}{n}} \cdots p_r^{\frac{n_r}{n}}\right)^n. \end{aligned}$$

Kadangi seka yra tipinė, tai

$$P(U^{(n)} = \mathbf{x}) \approx (p_1^{p_1} p_2^{p_2} \cdots p_r^{p_r})^n = 2^{-n\sum_{i=1}^{r} p_i \log_2 \frac{1}{p_i}} = 2^{-nH(U_1)}.$$

Nustatėme įdomią Bernulio šaltinio savybę: visų jo generuotų tipinių n ilgio sekų tikimybės yra beveik vienodos ir yra išreiškiamos entropija. Kiek yra tokių tipinių sekų? Iš gautos tikimybių išraiškos nesunku padaryti išvadą, kad jų yra maždaug $2^{nH(U_1)}$.

Tiesa, šios mūsų išvados grindžiamos nepakankamai griežtais matematiniais samprotavimais. Tačiau galima suformuluoti tikslų matematinį teiginį ir jį įrodyti.

7 teorema. *Tegu $\mathcal{U}$ yra Bernulio šaltinis, $\mathcal{A} = \{a_1, a_2, \ldots, a_r\}$ jo abėcėlė, $\epsilon > 0$ – bet koks pasirinktas skaičius. Kai n yra pakankamai didelis skaičius, vektoriaus $U^{(n)}$ reikšmių aibę $\mathcal{A}^n$ galima išreikšti tipinių reikšmių aibės*

$$\mathcal{A}_\epsilon^n = \{\mathbf{x} \in \mathcal{A}^n, : 2^{-n(H(U_1)+\epsilon)} \le P(U^{(n)} = \mathbf{x}) \le 2^{-n(H(U_1)-\epsilon)}\}$$

ir netipinių reikšmių aibės $\mathcal{A}^n \backslash \mathcal{A}_\epsilon^n$ sąjunga, kad būtų teisingos nelygybės:

$$P(U^{(n)} \in \mathcal{A}_\epsilon^n) \ge 1 - \epsilon, \quad |\mathcal{A}_\epsilon^n| \ge (1-\epsilon)2^{n(H(U_1)-\epsilon)}.$$

Ši Bernulio šaltinio savybė informacijos teorijoje vadinama asimptotiškai tolygaus pasiskirstymo savybe (AEP – asymptotic equipartition property, angl.). Ją turi ir kiti šaltiniai, pavyzdžiui, „geri“ Markovo šaltiniai. Taigi jeigu šaltinio entropija yra H ir jis turi asimptotiškai tolygaus pasiskirstymo savybę, tai galima įsivaizduoti, kad kai n didelis skaičius, tai šaltinis su maždaug vienodomis tikimybėmis generuoja $\approx 2^{nH}$ n ilgio sekų, o kitos sekos beveik nepasitaiko. Neretai ši savybė padeda geriau suprasti sudėtingus informacijos ir kodavimo teorijos dėsnius.

Asimptotiškai tolygaus pasiskirstymo savybė yra didžiųjų skaičių dėsnio išvada. Pirmiausia suformuluokime šį dėsnį.

8 teorema. *Jeigu $V_1, V_2, \ldots$ yra nepriklausomų vienodai pasiskirsčiusių atsitiktinių dydžių, turinčių vidurkį $\mathbf{E}V_j = a$, seka, tai bet kokiam $\epsilon > 0$ atsiras toks $n_0 = n_0(\epsilon)$, kad su visais $n \ge n_0$ teisingas įvertis*

$$P\Big(\Big|\frac{V_1 + V_2 + \ldots + V_n}{n} - a\Big| \le \epsilon\Big) \ge 1 - \epsilon. \tag{13}$$

7 teoremos įrodymas. Žymėsime $p_k = P(U_j = a_k),\ k = 1, 2, \ldots, r$. Apibrėšime atsitiktinius dydžius V_j taip:

$$\text{jei } U_j = a_k, \text{ tai } V_j = \log_2 \frac{1}{p_k}.$$

Atsitiktiniai dydžiai, kurių reikšmės reiškiamos tikimybėmis, gana neįprasti. Galbūt juos bus lengviau suvokti pasitelkus paprastesnį pavyzdį. Tarkime,

atsitiktinio dydžio reikšmės – skaičiai, užrašyti ant atvirtusios paridento nesimetriško kauliuko sienelės. Jeigu ant sienelių užrašysime tų sienelių atvirtimo tikimybes (arba kokios nors šių tikimybių funkcijos reikšmes), gausime naują atsitiktinį dydį, kurio reikšmės reiškiamos atitinkamų baigčių tikimybėmis.

Atsitiktiniai dydžiai V_j yra nepriklausomi ir vienodai pasiskirstę, jų vidurkis lygus

$$\mathbf{E}V_j = \sum_{k=1}^{r} p_k \log_2 \frac{1}{p_k} = H(U_1).$$

Taigi jiems galime taikyti didžiųjų skaičių dėsnį (13). Pastebėkime, kad jei $U^{(n)} = \mathbf{x}$, tai

$$\frac{V_1 + V_2 + \ldots + V_n}{n} = \frac{1}{n} \log_2 \frac{1}{P(U^{(n)} = \mathbf{x})}.$$

Taigi didžiųjų skaičių dėsnis teigia: jei $n \geq n_0(\epsilon)$, tai

$$P\Big(U^{(n)} = \mathbf{x} : \Big|\frac{1}{n} \log_2 \frac{1}{P(U^{(n)} = \mathbf{x})} - H(U_1)\Big| \leq \epsilon\Big) \geq 1 - \epsilon.$$

Tačiau ši nelygybė yra ekvivalenti nelygybei $P(U^{(n)} \in \mathcal{A}_\epsilon^n) \geq 1 - \epsilon$. Antroji teoremos nelygybė gaunama taip:

$$\begin{aligned} 1 - \epsilon &\leq P(U^{(n)} \in \mathcal{A}_\epsilon^n) = \sum_{\mathbf{x} \in \mathcal{A}_\epsilon^n} P(U^{(n)} = \mathbf{x}) \\ &\leq 2^{-n(H(U_1)-\epsilon)} \sum_{\mathbf{x} \in \mathcal{A}_\epsilon^n} 1 = 2^{-n(H(U_1)-\epsilon)} |\mathcal{A}_\epsilon^n|. \end{aligned}$$

Po akivaizdžių pertvarkių gauname antrąją teoremos nelygybę.

1.6. Markovo šaltinių entropija

Sudėtingų šaltinių entropijas apskaičiuoti sunku. Kartais pavyksta.

Kai $\mathcal{U}$ yra m-osios eilės Markovo šaltinis, tai

$$H(U_t|U_1, U_2, \ldots, U_{t-1}) = H(U_t|U_{t-m}, U_{t-m+1}, \ldots, U_{t-1}).$$

Nagrinėkime paprasčiausią Markovo šaltinio atvejį, t. y. atvejį, kai atminties gylis $m = 1$. Tada grandinės taisyklę galime užrašyti taip:

$$H(U_1, U_2, \ldots U_n) = H(U_1) + H(U_2|U_1) + \ldots + H(U_n|U_{n-1}). \tag{14}$$

Panagrinėkime, nuo ko priklauso sąlyginės entropijos

$$H(U_m|U_{m-1}) = \sum_a H(U_m|U_{m-1} = a) P(U_{m-1} = a).$$

Jeigu šaltinis yra stacionarus, tai $H(U_m|U_{m-1}=a)$ priklauso tik nuo perėjimo tikimybių, bet ne nuo m. Vadinasi, tokio šaltinio atveju priklausomybės nuo m priežastis – tikimybės $P(U_{m-1}=a)$. Jeigu šios tikimybės, didėjant m „stabilizuojasi“, t. y. artėja prie ribinių reikšmių, tai prie ribinės reikšmės artėja ir $H(U_n|U_{n-1})$. Ši riba ir bus šaltinio entropija. Toks yra teoremos apie stacionaraus Markovo šaltinio entropiją įrodymo kelias. O štai ir pati teorema.

9 teorema. *Tegu $\mathcal{U}=\langle U_1,U_2,\ldots\rangle$ yra stacionarus pirmos eilės Markovo šaltinis,*

$$p(b|a)=P(U_m=b|U_{m-1}=a)$$

šaltinio perėjimo tikimybės, be to, egzistuoja ribos

$$\pi(a)=\lim_{n\to\infty}P(U_n=a). \tag{15}$$

Tada šaltinio entropija lygi

$$H(\mathcal{U})=\sum_{a\in\mathcal{A}}\pi(a)H_a,\quad H_a=\sum_{b\in\mathcal{A}}p(b|a)\log_2\frac{1}{p(b|a)}.$$

Tačiau kada gi (15) ribos egzistuoja? Atsakymą į šį klausimą galėtume surasti Markovo grandinių teorijoje. Kone kiekviename solidesniame tikimybių teorijos vadovėlyje surasite skyrių, skirtą Markovo grandinėms.[1]. O mes apsiribokime keliais paprastais, tačiau įdomiais pavyzdžiais.

Gali būti, kad Markovo šaltinio dydžių tikimybės $P(U_n=a)$ iš viso nepriklauso nuo indekso n. Tada tikimybių rinkinį

$$\pi(a)=P(U_n=a)$$

vadiname stacionariuoju skirstiniu. Kadangi

$$P(U_{n+1}=x)=\sum_y p(x|y)P(U_n=y),$$

tai stacionariojo skirstinio tikimybės turi tenkinti lygybę

$$\pi(x)=\sum_y p(x|y)\pi(y),\quad x\in\mathcal{A}=\{a_1,\ldots,a_r\}. \tag{16}$$

Jeigu iš stacionariojo skirstinio tikimybių sudarysime vektorių, o iš perėjimo tikimybių – matricą $\mathbf{P}$, tai visas (16) lygybes galėsime užrašyti viena eilute taip:

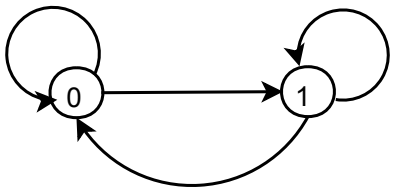
$$(\pi(a_1),\ldots,\pi(a_m))=(\pi(a_1),\ldots,\pi(a_r))\mathbf{P},\quad \mathbf{P}=(p_{ij}),p_{ij}=p(a_j|a_i).$$

[1] Žr., pvz., J. Kubilius. Tikimybių teorija ir matematinė statistika. Vilnius 1980. Tikimybių ribų egzistavimo tema dėstoma skyrelyje „Markovo grandinių ergodinės teoremos“).

Pavyzdys. Tegu pirmosios eilės stacionaraus Markovo šaltinio $\mathcal{U} = \langle U_1, U_2, \ldots\rangle$ dydžiai įgyja reikšmes iš dvejetainės abėcėlės $\mathcal{B} = \{0, 1\}$, o perėjimo tikimybės

$$\mathbf{P} = \begin{pmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{pmatrix}.$$

Šaltinį su šia perėjimo tikimybių matrica galime pavaizduoti tokia diagrama:



Šaltinio tikimybės: $p(0|0) = 1-\alpha, p(0|1) = \beta, p(1|0) = \alpha, p(1|1) = 1-\beta$.

Surasime stacionariojo skirstinio tikimybes $\pi_0 = \pi(0), \pi_1 = \pi(1)$. Iš lygybės

$$\langle \pi_0, \pi_1\rangle = \langle \pi_0, \pi_1\rangle \mathbf{P}$$

gauname tokią lygčių sistemą:

$$\begin{aligned} (1-\alpha)\pi_0 + \beta\pi_1 &= \pi_0, \\ \alpha\pi_0 + (1-\beta)\pi_1 &= \pi_1. \end{aligned}$$

Yra be galo daug skaičių porų $\langle \pi_0, \pi_1\rangle$, tenkinančių šią sistemą, tačiau tik viena pora tenkina sąlygą $\pi_0 + \pi_1 = 1$. Ši pora

$$\pi_0 = \frac{\beta}{\alpha+\beta}, \quad \pi_1 = \frac{\alpha}{\alpha+\beta}$$

ir yra ieškomas stacionarusis skirstinys. Šaltinio entropija

$$H(\mathcal{U}) = \frac{h(\alpha)\beta + h(\beta)\alpha}{\alpha+\beta}, \quad h(u) = u\log_2\frac{1}{u} + (1-u)\log_2\frac{1}{1-u},\ 0 < u < 1.$$

Pavyzdys. Tegu G yra jungusis neorientuotas grafas, t. y. bet kurias dvi jo viršūnes jungia iš briaunų sudarytas kelias. Galime įsivaizduoti, kad grafo viršūnės – tai miestai, kuriuos lanko keliautojas. Nuvykęs į miestą, keliautojas mums pasiunčia atviruką. Taigi galime tarti, kad mūsų informacijos šaltinio $\mathcal{U} = \langle U_1, U_2, \ldots\rangle$ dydžių reikšmės – aplankytų viršūnių numeriai (miestų pavadinimai). Jeigu, būdamas viršūnėje i, kitam kelionės etapui jis su vienodomis tikimybėmis renkasi bet kurią iš viršūnės išeinančią briauną, tai $\mathcal{U}$ yra stacionarus Markovo šaltinis. Tegu grafas turi iš viso m viršūnių, o i-osios viršūnės laipsnis (kelių, vedančių iš šios viršūnės, skaičius) lygus W_i. Apibrėžkime dydžius:

$$\epsilon_{ij} = \begin{cases} 1, & \text{jei viršūnės } i, j \text{ sujungtos briauna,} \\ 0, & \text{jeigu nesujungtos.} \end{cases}$$

Tada perėjimo tikimybių matricą galime užrašyti taip:

$$\mathbf{P} = \begin{pmatrix} 0 & \frac{\epsilon_{12}}{W_1} & \frac{\epsilon_{13}}{W_1} & \dots & \frac{\epsilon_{1m}}{W_1} \\ \frac{\epsilon_{21}}{W_2} & 0 & \frac{\epsilon_{23}}{W_2} & \dots & \frac{\epsilon_{2m}}{W_2} \\ \dots & \dots & \vdots & \dots & \dots \\ \frac{\epsilon_{m1}}{W_m} & \frac{\epsilon_{m2}}{W_m} & \frac{\epsilon_{m3}}{W_m} & \dots & 0 \end{pmatrix}.$$

Stacionarusis skirstinys egzistuoja ir yra labai paprastas. Štai jis:

$$\pi(i) = \frac{W_i}{W_1 + W_2 + \ldots + W_m} \quad i = 1, 2, \ldots, m.$$

Jeigu abejojate, ar skirstinys tikrai stacionarusis – įstatykite į (16) ir įsitikinkite. Taigi apskaičiuoti mūsų „atvirukų šaltinio“ entropiją nesunku.

Šaltinio $\mathcal{U}$ entropiją apibrėžėme kaip ribą

$$H = \lim_{n\to\infty} \frac{H(U_1, U_2, \ldots, U_n)}{n}.$$

Ją dažnai galima interpretuoti ir kitaip.

10 apibrėžimas. *Šaltinį $\mathcal{U} = \langle U_1, U_2, \ldots\rangle$ vadinsime stacionariu, jeigu su bet kokiu $k \geq 0$ atsitiktinio vektoriaus*

$$V_m = \langle U_m, U_{m+1}, \ldots, U_{m+k}\rangle$$

reikšmių tikimybės nepriklauso nuo m.

Jeigu šaltinis stacionarus, tai entropijos

$$H(U_m|U_{m-k}, U_{m-k+1}, \ldots, U_{m-1})$$

nepriklauso nuo m, bet tik nuo k. Įsitikinkime, kad stacionaraus šaltinio atveju entropijų seka

$$H_n = H(U_n|U_1, U_2, \ldots, U_{n-1})$$

mažėja, kai n auga. Pasinaudoję entropijos savybe $H(X|Y, Z) \leq H(X|Y)$ ir šaltinio stacionarumu, gauname

$$H(U_{n+1}|U_1, U_2, \ldots, U_n) \leq H(U_{n+1}|U_2, \ldots, U_n) = H(U_n|U_1, \ldots, U_{n-1}).$$

Taigi seka H_n monotoniškai mažėja.

10 teorema. *Jeigu šaltinis yra stacionarus, tai jo entropija*

$$H = \lim_{n\to\infty} H(U_n|U_1, U_2, \ldots, U_{n-1}). \tag{17}$$

Įrodymas. Pažymėkime monotoniškai mažėjančios sekos

$$H_n = H(U_n|U_1, U_2, \ldots, U_{n-1})$$

ribą H^*. Tegu $\epsilon > 0$ – pasirinktas mažas skaičius. Tada kai $n > n_0(\epsilon)$, tai $H^* \leq H_n \leq (1+\epsilon)H^*$. Pasinaudoję grandinės taisykle, gauname

$$\begin{aligned} H(U_1, U_2, \ldots, U_n) &= H(U_1) + H(U_2|U_1) + \ldots + H(U_n|U_1, U_2, \ldots, U_{n-1}) \\ &= H_1 + H_2 + \ldots + H_n. \end{aligned}$$

Taigi kai $n > n_0(\epsilon)$,

$$nH^* \leq H(U_1, U_2, \ldots, U_n) \leq \sum_{i<n_0} H_i + (1+\epsilon)(n-n_0)H^*.$$

Dabar jau nesunku, pasinaudojus entropijos apibrėžimu

$$H = \lim_{n\to\infty} \frac{H(U_1, U_2, \ldots, U_n)}{n}$$

įrodyti (17) lygybę.

Sąryšį (17) galime interpretuoti taip: stacionaraus šaltinio entropija – tai jo neapibrėžtumo (arba informatyvumo, arba įdomumo) kiekis, kuris dar liko po to, kai sužinojome labai ilgą šio šaltinio generuotą simbolių srautą.

1.7. Informacijos šaltinis – kalba

Kalbėtojo žodžiai yra tarpusavyje susiję, tačiau šiek tiek ir atsitiktiniai. Matematinio modelio, tiksliai nusakančio tokio šaltinio savybes, niekada nesukursime. Tačiau galime panagrinėti „apytikslius", t. y. išreiškiančius tik nedaugelį tų šaltinių savybių modelius.

Koks matematinis modelis tinka mūsų kalbai aprašyti?

Kokios taisyklės valdo mūsų kalbos srautus, nežinome, užtat puikiai mokame jomis naudotis. Žinių dažniausiai įgyjama gretinant ir lyginant. Galima, pavyzdžiui, generuoti abėcėlės simbolių srautus pagal tam tikras taisykles ir lyginti juos su natūralios kalbos tekstais arba paprasčiausiai – vadovaujantis subjektyviu, tačiau patikimu kalbos jausmu, spręsti, ar generuotas srautas nors kiek panašus į tikrovišką.

Lietuvių kalbos abėcėlėje yra 32 raidės; norėdami atskirti žodžius, galime į abėcėlę įtraukti ir tarpą, tada bus 33 simboliai.

Paprasčiausias būdas generuoti abėcėlės simbolių srautą – rinkti simbolius nepriklausomai vienas nuo kito su vienodomis, lygiomis 1/33 tikimybėmis. Tokiu būdu bandytume modeliuoti lietuvių kalbos tekstą, naudodamiesi Bernulio šaltiniu, kuriam visos raidės yra lygiavertės.

Štai šitaip generuoto srauto pavyzdėlis:

HURŽVGLKŲHERKC ŠZHAN ČSSZ DKŠGSČ
AZĖHJJNYVNŠFĮMIĖČJMĘKŲUFUČĮHNEPŪPĮAJGCNČYZPYVHKŠNYTĄMKDČZ
EIPNLMPSZ

Tekste yra 200 simbolių, įskaitant tarpus. Raidės A, Ą, B, C, D, ... jame pasikartoja atitinkamai $6, 10, 4, 3, 9, \ldots$ kartus. Į tikrą kalbą nepanašu nė kiek. Tokį kalbos modelį pavadinkime nulinės eilės artiniu.

Šio šaltinio entropija lygi $H_0 = \log_2 33 \approx 5.0444$.

Galime patobulinti modelį, ištyrę daug kalbos tekstų ir nustatę simbolių pasitaikymo dažnius

$$p(1) = \frac{n_1}{n},\ p(2) = \frac{n_2}{n},\quad \ldots,\quad p(33) = \frac{n_{33}}{n},$$

čia n – bendras simbolių skaičius, o n_i – i-osios raidės pasikartojimų skaičius. Dabar galime naudoti šiuos dažnius kaip Bernulio šaltinio generuojamų simbolių tikimybes. Šitaip gauname šiek tiek geresnį modelį – pirmosios eilės artinį. Štai tokio šaltinio generuoto teksto pavyzdys:

TEAEASTIIO UGKLŠ R TAMU IAMGO ŲIOMTRČMSAJIOĮ ŽMIIIIENŠČAKSMIII M NUYIIVAK DMTPIĖVNLOOTPOKSOOILJAISN

Jeigu skaičiuosime entropiją, gausime $H_1 \approx 4,34$.[2]

Kitas žingsnis, tikslinant kalbos modelį – keisti Bernulio šaltinius Markovo šaltiniais. Ištyrę itin daug tekstų, galėtume nustatyti raidžių porų (digramų) tikimybes ir sąlygines tikimybes (kad po raidės i sutiksime raidę j):

$$p(i,j) = \frac{n_{i,j}}{n},\quad p(j|i) = \frac{p(i,j)}{p(i)},$$

čia $n_{i,j}$ yra raidžių poros $\langle i,j\rangle$ pasitaikymų skaičius, n – bendras porų skaičius. Toks šaltinis – Markovo grandinė su vienetiniu atminties gyliu – antrosios eilės artinys. Suskaičiavę šaltinio entropiją, gautume $H_2 \approx 3,59$.

Tačiau ieškoti tikimybių ir generuoti tokio šaltinio srautą yra daug sudėtingiau. Žinoma, kai turime kompiuterius ir mokame programuoti – anoks čia ir sudėtingumas. C. Shannonas sugalvojo, kaip generuoti tokio šaltinio srautą kitaip. Štai jo idėja.

Pasiimkite kokią nors storą knygą, atsitiktinai atsiverskite kokį nors puslapį, atsitiktinai pasirinkite bet kokią raidę ir ją užrašykite. Po to vėl atsitiktinai atverskite puslapį ir, pradėdami nuo viršaus, ieškokite užrašytosios raidės. Radę ją, užrašykite raidę, kuri seka po jos. Tada vėl atsitiktinai atverskite puslapį... Kokia elegantiška idėja! Ją paprasta įgyvendinti programa. Štai tokiu būdu generuoto teksto pavyzdys:

MERIKAIGUOTOSUGANDADE VIENIA JŲ GIAMSTO PĖS NOS AIS THBŪTGIS CIU TAIŲ RTERĖSNEISIAIHINEDAUDANI

O toliau – galime pasitelkti antrosios, trečiosios ... eilės Markovo grandines. Tokių šaltinių generuotus srautus irgi galime kurti Shannono metodu.

Teksto fragmentas, generuotas naudojant antros eilės Markovo grandinių modelį:

---

[2] Entropijų reikšmės gautos šiuose, lietuvių kalbos tekstų statistinėms savybėms skirtuose darbuose:

R. Merkytė. Apie lietuvių kalbos informatyvumą. Lietuvos matematikos rinkinys, t. XVIII, nr.3, p. 109–116.

J. Damskienė. Lingvistinių elementų statistinės charakteristikos. VU MIF, Matematinės statistikos katedra. Magistro darbas. 2001, p. 1–48.

SUTIKIMINTYTŲ DVĖLIAUJŲ PAS BUVO NUOTI MISE ARGINIJOIS TROGIA PASAUDAMASTAS MATRIP IŠMO DAVIMŲ PO LAN

Ir trečiosios eilės:

A TY PADĖTIEK NETŲ UŽBAS BUVO PATYTI MAISŽADŽIUI ŽASTI IR TIKALE PASAS ŠIO TO ŠTAIS TEKSANDOMA APIE ĮSI

Dar vienas požiūris į kalbos entropiją. Tarkime, kad kalbos, kurios žodžiai parašyti naudojantis abėcėle $\mathcal{A}, |\mathcal{A}| = N$, šaltinis turi asimptotiškai tolygaus pasiskirstymo savybę. Tai reiškia, kad tipinių (prasmingų) n ilgio sakinių skaičius T_n yra maždaug

$$T_n \approx 2^{nH},$$

čia H yra kalbos entropija. Tarkime, visus šiuos tipinius sakinius norime koduoti aibės $\mathcal{A}^m$ žodžiais. Koks m turi būti? Visi sakiniai turi būti koduojami skirtingai, todėl m ilgio žodžių turi užtekti:

$$2^{nH} \le N^m = 2^{m \log_2 N}, \quad \text{taigi } m \approx \frac{nH}{\log_2 N}.$$

Tačiau sakiniai užrašomi vartojant n simbolių, taigi perteklius yra

$$n - m \approx nR, \quad R = 1 - \frac{H}{\log_2 N}.$$

Dydis R vadinamas kalbos pertekliškumu (redundancy, angl.). Perskaitę žodžius „infrmcjs mtas“, be abejo, supratote jų prasmę. Dažnai tą pačią prasmę galima perteikti ir trumpiau, tačiau... žmonėms bendraujant, pertekliškumas tiesiog būtinas! Jeigu netikite – atlikite bandymą. Prieš kreipdamiesi į žmogų kokiu nors reikalu, sugalvokite tokį sakinį, kad jame būtų vien tik būtini žodžiai, nei vieno, kurio galima atsisakyti. Rezultatas bus tikriausiai toks: jūsų paprašys paaiškinti dar kartą!

2 Šaltinio kodavimas

Senovės egiptiečių, graikų raštai – papiruso ar pergamento ritinėliai, senovės romėnai rašė ant vašku padengtų medinių lentelių. Taigi romėnai – knygų puslapių išradėjai. Lotyniškas žodis „codex“, reiškia medžio kamieną, iš kurio tie puslapiai buvo daromi. Tokia yra žodžio „kodas“ kilmė.

2.1. Kodai ir kodavimas

> *„Čia kažkoks kodas“, – sako žmonės, pamatę ženklų, kurių prasmės iš karto negalima įžvelgti, seką. Turima galvoje, kad ženklai surašyti pagal tam tikras taisykles. Taigi kodas – tam tikras būdas žinioms užrašyti. Tačiau matematiniam tyrinėjimui reikalingas tikslesnis kodo apibrėžimas.*

Kad galėtume sudaryti kodą, reikia ženklų rinkinio (abėcėlės) ir naudojimosi tais ženklais taisyklių.

Prisiminkime keletą jau ne kartą vartotų sąvokų.

11 apibrėžimas. *Baigtinę abėcėlės $\mathcal{A}$ simbolių seką $x_1 \dots x_m$ vadinsime m ilgio žodžiu. Jei $\mathbf{x}$ yra žodis, tai $|\mathbf{x}|$ žymėsime jo ilgį. Jei $\mathbf{x}, \mathbf{y}$ yra du tos pačios abėcėlės žodžiai, tai $\mathbf{xy}$ žymėsime sudurtinį žodį, kuris gaunamas tiesiog sujungiant $\mathbf{x}$ ir $\mathbf{y}$. Žodį $\mathbf{x}$ vadinsime šio sudurtinio žodžio priešdėliu, o $\mathbf{y}$ – priesaga.*

Abėcėlės $\mathcal{A}$ ilgio m žodžių aibę žymime $\mathcal{A}^m$, o visų žodžių aibę – $\mathcal{A}^*$.

Tegu $\mathcal{A}$ ir $\mathcal{B}$ yra dvi abėcėlės (gali būti, kad $\mathcal{A} = \mathcal{B}$). Koduoti abėcėlės $\mathcal{A}$ žodžius reiškia keisti juos abėcėlės $\mathcal{B}$ žodžiais. Taigi kodavimo taisyklė yra funkcija

$$c^* : \mathcal{A}^* \to \mathcal{B}^*.$$

Vertimas iš vienos kalbos į kitą taip pat yra kodavimas: vienos kalbos sakiniai keičiami į kitos kalbos sakinius. Tik kodavimo taisyklė nėra griežtai apibrėžta.

Jeigu abėcėlių $\mathcal{A} = \{0, 1, \dots, r-1\}$ ir $\mathcal{B} = \{0, 1, 2, \dots, t-1\}$ elementus interpretuosime kaip skaičiavimo sistemų su pagrindais r ir t skaitmenis, tai šių abėcėlių žodžius galėsime suvokti kaip natūrinius skaičius. Kiekvieną funkciją $f : \mathbb{N} \to \mathbb{N}$, apibrėžtą natūrinių skaičių aibėje ir toje pat aibėje įgyjančią reikšmes, galime suvokti kaip kodavimo taisyklę $c_f : \mathcal{A}^* \to \mathcal{B}^*$, jeigu funkcijos f argumentą užrašysime skaičiavimo sistemoje su pagrindu r, o jos reikšmę – sistemoje su pagrindu t. Pavyzdžiui, funkciją $f(n) = n$ atitinka tokia kodavimo (skaičiavimo sistemos keitimo) taisyklė:

$$\text{jei } a_1 r^0 + a_2 r^1 + \dots + a_k r^{k-1} = b_1 t^0 + b_2 t^1 + \dots + b_l r^{l-1},\ a_i \in \mathcal{A},\ b_j \in \mathcal{B},$$
$$\text{tai } c_f(a_1 a_2 \cdots a_k) = b_1 b_2 \cdots b_l.$$

Kodavimo taisyklių yra be galo daug, jų yra „tiek pat“ kiek realiųjų skaičių, t. y. kodavimo taisyklių aibė yra kontinuumo galios. Kad galėtume

naudotis kodavimo taisykle, turime ją apibrėžti. Apibrėžimas – vėl tam tikros baigtinės abėcėlės žodis. Galbūt ši abėcėlė yra didžiulė, sudaryta iš kelių mažesnių abėcėlių raidžių ir matematinių ženklų, tačiau vis tiek baigtinė. Vadinasi, jos žodžių (o taip pat ir kodavimo taisyklių apibrėžimų) aibė yra skaiti. Taigi yra be galo daug kodavimo taisyklių, kurių iš viso negalima mums priimtinu būdu apibrėžti!

Tačiau užuot abstrakčiai svarstę apie kodavimo taisykles, verčiau pasielkime itin praktiškai: nagrinėkime tik tokias kodavimo taisykles, kurias galima apibrėžti nurodžius, kaip koduojami pavieniai abėcėlės $\mathcal{A}$ simboliai.

12 apibrėžimas. *Tegu $\mathcal{A}$ ir $\mathcal{B}$ yra dvi baigtinės abėcėlės, o $c : \mathcal{A} \to \mathcal{B}^*$ injektyvus atvaizdis. Kodavimo taisyklę $c^* : \mathcal{A}^* \to \mathcal{B}^*$,*

$$c^*(x_1x_2\cdots x_n) = c(x_1)c(x_2)\cdots c(x_n), \quad x_i \in \mathcal{A},$$

vadinsime atvaizdžio c tęsiniu. Abėcėlės $\mathcal{B}$ žodį $c^*(x_1x_2\cdots x_n)$ vadinsime žodžio $x_1x_2\cdots x_n$ kodu.

Taigi turime itin paprastą būdą kodavimo taisyklėms konstruoti: reikia apibrėžti atvaizdį $c : \mathcal{A} \to \mathcal{B}^*$ ir pratęsti jį iki funkcijos $\mathcal{A}^* \to \mathcal{B}^*$.

Kadangi svarbiausias šios konstrukcijos elementas yra atvaizdis $c : \mathcal{A} \to \mathcal{B}^*$, tai dažnai būtent jį ir vadinsime kodu.

Tegu abėcėlėje $\mathcal{A}$ yra nustatyta tvarka, t. y. jos raidės yra sunumeruotos: $\mathcal{A} = \{a_1, a_2, \ldots, a_r\}$. Tada kodą $c : \mathcal{A} \to \mathcal{B}^*$ galime apibrėžti tiesiog surašydami žodžius $c(a_i)$ į eilę (sudarydami gretinį):

$$\langle \mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_r \rangle, \quad \mathbf{c}_i = c(a_i).$$

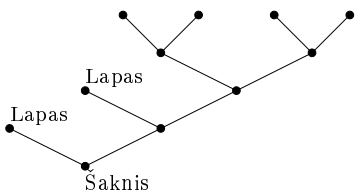
Taigi abėcėlės $\mathcal{A}$ galima ir neminėti; dažnai taip ir darysime – kodu vadinsime tiesiog atitinkamos abėcėlės žodžių gretinį, arba tiesiog žodžių poaibį

$$\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_r\} \subset \mathcal{B}^*.$$

Kartais, nagrinėjant kodus, patogu juos vaizduoti grafais.

Grafą įsivaizduosime kaip geometrinį darinį plokštumoje. Jo viršūnes vaizduoja taškai, o briaunas – orientuotos arba neorientuotos tuos taškus jungiančios atkarpos.

Svarbus mums bus specialus grafas, kurį vadinsime **medžiu**, jo briaunas – šakomis (žr. brėž.). Vieną jo viršūnę vadinsime **šaknimi**; kiekvienai kitai viršūnei egzistuoja vienintelis kelias, jungiantis viršūnę su šaknimi. Šį kelią sudarančių briaunų skaičių vadinsime viršūnės atstumu iki šaknies. Sakysime, kad dvi viršūnės yra tame pat lygyje, jei jų atstumai iki šaknies yra tie patys. Viršūnes, turinčias tik vieną šaką, vadinsime **lapais**.

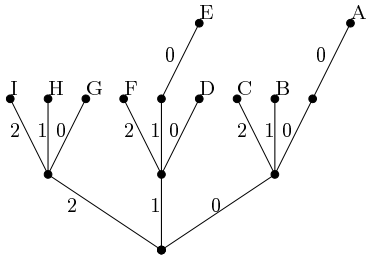


Medis

Medį vadinsime **q-nariu**, jei iš kiekvienos viršūnės išeina ne daugiau kaip q šakų; jeigu iš kiekvienos viršūnės, kuri nėra lapas, išeina lygiai q šakų, tai medį vadinsime **reguliariuoju q-nariu**. Jei visų reguliariojo q-nario medžio lapų atstumai iki šaknies yra vienodi, tai medį vadinsime **pilnuoju q-nariu** medžiu.

Tegu $\mathcal{B}$ yra abėcėlė, turinti r simbolių. Kiekvienai pilnojo r-nario grafo šakai priskirkime „svorį“ – vieną abėcėlės simbolį taip, kad iš tos pačios viršūnės išeinančios šakos turėtų skirtingus simbolius. Jeigu V yra n-ojo lygio viršūnė, tai kelią, kuris veda iš šaknies į V, atitinka n ilgio abėcėlės $\mathcal{B}$ žodis. Keliai, vedantys į n-ojo lygio viršūnes, abipus vienareikšmiškai vaizduojami aibės $\mathcal{B}^n$ žodžiais. Jei kelias į viršūnę V atitinka žodį **a**, tai visus žodžius, kurie turi priešdėlį **a**, atitinka keliai, einantys per viršūnę V.

Tegu $c : \mathcal{A} \to \mathcal{B}^*$ yra koks nors kodas. Į medžio, kurį susiejome su abėcėlės $\mathcal{B}$ žodžiais, viršūnę V „įkelkime“ abėcėlės $\mathcal{A}$ simbolį a, jei kelias, jungiantis šaknį su viršūne V, vaizduoja žodį $c(a)$. Nutrinkime šakas, kurios neįeina į kelius, vedančius į abėcėlės $\mathcal{A}$ simboliais pažymėtas viršūnes. Gautąjį medį vadinsime kodo c medžiu.



Kodo $c : \mathcal{A} \to \mathcal{B}^*$ *medis,* $\mathcal{A} = \{A, B, C, D, E, F, G, H, I\}$, $\mathcal{B} = \{0, 1, 2\}$.

Taigi kiekvieną kodą galime pavaizduoti medžiu, kurio šakoms priskirti kodo abėcėlės simboliai. Keliai, vedantys iš šaknies į lapus, atitinka kodo žodžius. Kitaip sužymėję medžio briaunas, gautume kitą kodą. Tačiau po kitu „apdaru“ slėptųsi ta pati struktūra. Kodų medžiai – puikus instrumentas paslėptai struktūrai atskleisti ir tyrinėti.

2.2. Kodų pavyzdžiai

Kokių tik kodų žmonės nėra naudoję! Egiptiečių, majų, kinų rašmenys – irgi kodai. Tai taisyklės, kurios tikrovės daiktams ir reiškiniams priskiria nedidelius piešinius. Jų kūrėjai vadovavosi savo menine nuovoka... Panagrinėkime keletą senų ir dabartinių kodų, kurių sandara pagrįsta formaliomis taisyklėmis.

Antikinės Graikijos istorikas Polibijus (apie 208 m. pr. Kr.) rašo, kad, perduodami karines žinias, graikai naudojosi ugnies kodu. Abėcėlę $\mathcal{A}$ sudaro 24 graikiškos raidės, $\mathcal{B} = \{\wr, \wr\wr, \wr\wr\wr, \wr\wr\wr\wr, \wr\wr\wr\wr\wr\}$; čia $\wr$ žymi degantį fakelą. Žinoma, galime fakelus keisti skaičiais ir naudoti abėcėlę $\mathcal{B} = \{1, 2, 3, 4, 5\}$.

α	11	ι	24	ρ	42
β	12	κ	25	σ	43
γ	13	λ	31	τ	44
δ	14	μ	32	υ	45
ϵ	15	ν	33	ϕ	51
ζ	21	ξ	34	χ	52
η	22	o	35	ψ	53
θ	23	π	41	ω	54

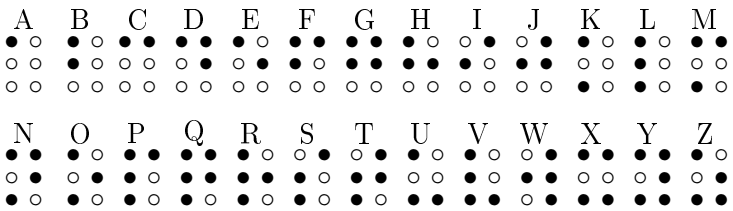
Polibijaus aprašytas ugnies kodas. Norėdami perduoti, pavyzdžiui, raidę β, ryšininkai vienoje kalvos vietoje turėjo iškelti vieną degantį fakelą, o kitoje – du.

Renesanso laikotarpiu ir vėliau kodų apibrėžimai sudarydavo ištisas knygas. Tose knygose nurodoma, kuo rašant keisti žodžius ir ištisus sakinius. Kodai būdavo naudojami perduodamų tekstų prasmei paslėpti.

S. Morse (1791–1872) išrastas telegrafas – pirmasis moderniųjų laikų ryšių įrenginys. Jam pritaikytas kodas – tiesiog techninė būtinybė. S. Morse kodas tai lotyniškos abėcėlės $\mathcal{A}$ ir žodžių, sudarytų iš taškų ir brūkšnelių atitiktis. Tiesa, tie žodžiai atskiriami pauze. Taigi iš tiesų abėcėlę $\mathcal{B}$ sudaro trys simboliai.

a	$\cdot-$	j	$\cdot---$	s	$\cdots$
b	$-\cdot\cdot\cdot$	k	$-\cdot-$	t	$-$
c	$-\cdot-\cdot$	l	$\cdot-\cdot\cdot$	u	$\cdot\cdot-$
d	$-\cdot\cdot$	m	$--$	v	$\cdot\cdot\cdot-$
e	$\cdot$	n	$-\cdot$	w	$\cdot--$
f	$\cdot\cdot-\cdot$	o	$---$	x	$-\cdot\cdot-$
g	$--\cdot$	p	$\cdot--\cdot$	y	$-\cdot--$
h	$\cdot\cdot\cdot\cdot$	q	$--\cdot-$	z	$--\cdot\cdot$
i	$\cdot\cdot$	r	$\cdot-\cdot$		

Morse kodas pritaikytas telegrafui.



Apie 1820 metus Louis Braille sukurtas kodas akliesiems. Iš 3×2 *matricos tašky galima sudaryti* 64 *ženklus; nepavartotais abėcėlės ženklais galima koduoti dažnai pasitaikančius žodžius.*

Kitą telegrafui pritaikytą kodą apie 1880 metus sukūrė J. M. Baudot'as. Kiekvienas simbolis koduojamas penkių dvejetainių ženklų žodžiu. Tačiau kiekvienas toks žodis priskiriamas dviem ženklams: raidei ir simboliui. Raidžių srities pradžią ženklina žodis 00001, o simbolių srities – 11000. Taigi šiuo kodu galima koduoti iš viso $2 \times 32 - 2 = 62$ skirtingus ženklus. Baudot'o kodas naudotas ne tik telegrafui, bet ir pirmųjų kartų kompiuteriams.

Pirmieji kompiuteriai duomenims koduoti dvejetainės abėcėlės žodžiais naudojo Baudot'o kodą. Kol kompiuteriai buvo naudojami tik kaip skaičiuokliai, to kodo užteko. Tačiau ėmus kompiuterius taikyti plačiau, ženklų, kuriuos galima koduoti Baudot'o kodu, nebeužteko. 1963 metais buvo paskelbtas naujas kodavimo standartas ASCII (American Standard Code for Information Interchange). Kiekvienam ženklui koduoti naudojami septyni bitai, mažosios ir didžiosios lotyniškos abėcėlės raidės koduojamos skirtingai. Numatyti ir ekrane nevaizduojamų tarnybinių simbolių kodai. Kodas pritaikytas visų pirma amerikiečių vartotojams, tačiau ir Amerikoje jis ne iš karto prigijo. Pavyzdžiui, IBM net iki 1980 metų naudojo savo kodą EBCDIC (Extended Binary Coded Decimal Interchange Code). Kai prireikė koduoti ir kitų abėcėlių ženklus, buvo pridėtas dar vienas bitas ir atsirado įvairūs išplėstinio ASCII kodo variantai.

Raidė	Kodas	Simbolis	Raidė	Kodas	Simbolis
A	10000	1	*Q*	10111	/
B	00110	8	*R*	00111	–
C	10110	9	*S*	00101	*Tarpas*
D	11110	0	*T*	10101	∅
E	01000	2	*U*	10100	4
F	01110	∅	*V*	11101	′
G	01010	7	*W*	01101	?
H	11010	+	*X*	01001	,
I	01100	∅	*Y*	00100	3
J	10010	6	*Z*	11001	:
K	10011	(	*RP*	00001	*RP*
L	11011	=	*SP*	11000	*SP*
M	01011	)	*IV*	11000	*IV*
N	01111	∅	*EP*	10001	*EP*
O	11100	5	*ER*	00011	*ER*
P	11111	%	∅	00000	∅

Baudot'o kodas. Tas pats dvejetainis žodis priskiriamas dviem ženklams – raidei ir simboliui. RP reiškia raidžių srities pradžią, SP – simbolių srities; IV reiškia įvesties simbolį, EP – eilutės pabaigą, ER – klaidos ženklą. Ženklu ∅ pažymėti žodžiai, kuriems simboliai nepriskirti.

Simbolis	*Kodas*	*Simbolis*	*Kodas*	*Simbolis*	*Kodas*	*Simbolis*	*Kodas*
NUL	0000000		0100000	@	1000000	‘	1100000
SOH	0000001	!	0100001	*A*	1000001	*a*	1100001
STX	0000010	“	0100010	*B*	1000010	*b*	1100010
ETX	0000011	#	0100011	*C*	1000011	*c*	1100011
EOT	0000100	$	0100100	*D*	1000100	*d*	1100100
ENQ	0000101	%	0100101	*E*	1000101	*e*	1100101
ACK	0000110	&	0100110	*F*	1000110	*f*	1100110
BEL	0000111	′	0100111	*G*	1000111	*g*	1100111
BS	0001000	(	0101000	*H*	1001000	*h*	1101000
TAB	0001001	)	0101001	*I*	1001001	*i*	1101001
LF	0001010	∗	0101010	*J*	1001010	*j*	1101010
VT	0001011	+	0101011	*K*	1001011	*k*	1101011
FF	0001100	,	0101100	*L*	1001100	*l*	1101100
CR	0001101	−	0101101	*M*	1001101	*m*	1101101
SO	0001110	.	0101110	*N*	1001110	*n*	1101110
SI	0001111	/	0101111	*O*	1001111	*o*	1101111
DLE	0010000	0	0110000	*P*	1010000	*p*	1110000
*DC*1	0010001	1	0110001	*Q*	1010001	*q*	1110001
*DC*2	0010010	2	0110010	*R*	1010010	*r*	1110010
*DC*3	0010011	3	0110011	*S*	1010011	*s*	1110011
*DC*4	0010100	4	0110100	*T*	1010100	*t*	1110100
NAK	0010101	5	0110101	*U*	1010101	*u*	1110101
SYN	0010110	6	0110110	*V*	1010110	*v*	1110110
ETB	0010111	7	0110111	*W*	1010111	*w*	1110111
CAN	0011000	8	0111000	*X*	1011000	*x*	1111000
EM	0011001	9	0111001	*Y*	1011001	*y*	1111001
SUB	0011010	:	0111010	*Z*	1011010	*z*	1111010
ESC	0011011	;	0111011	[	1011011	{	1111011
FS	0011100	<	0111100	\	1011100	|	1111100
GS	0011101	=	0111101	]	1011101	}	1111101
RS	0011110	>	0111110	^	1011110	∼	1111110
US	0011111	?	0111111	_	1011111	*α*	1111111

ASCII kodas. Tarnybiniai simboliai pažymėti santrumpomis, pavyzdžiui, STX reiškia „Start of text“ ir t. t.

Paskutiniajame XX amžiaus dešimtmetyje pradėtas kurti ir diegti naujas kodavimo standartas – Unicode. Kiekvienam simboliui priskiriamas šešiolikos bitų (dviejų baitų) dvejetainės abėcėlės žodis, taigi ženklų, kuriuos galima koduoti, yra daugiau nei 65 tūkstančiai. Tačiau šitaip koduojant atminties prireikia dvigubai daugiau negu koduojant ASCII. Sugalvota ir gerų išeičių. Viena jų – UTF-8 kodas, kurio struktūra susijusi su Unicode, bet koduojama skirtingo ilgio (nuo vieno iki keturių baitų) dvejetainiais žodžiais. Vieno baito žodžiais koduojamos lotyniškos abėcėlės raidės, skaitmenys

ir skyrybos ženklai, dviejų baitų – išplėstinės lotyniškos abėcėlės simboliai (taip pat ir lietuviškos raidės), trijų baitų – Azijos šalių abėcėlių ženklai...

2.3. Momentiniai arba p-kodai

Jeigu į savo kompiuterį siunčiate suspaustos informacijos archyvą, teks palaukti ryšio pabaigos, kad galėtumėte pradėti skaityti. Jeigu siunčiate, pavyzdžiui, tinklalapį, o ryšys lėtas, tai turinys ryškėja palengva. Tačiau kas parašyta teksto pradžioje, galite sužinoti nesulaukę ryšio pabaigos. Taigi galime sakyti, kad tinklalapio informacijai koduoti panaudotas momentinis kodas.

Jei kodavimas yra rašymas, tai dekodavimas – skaitymas. Ne kiekvieną raštą galima vienareikšmiškai perskaityti, ne kiekvieno kodo žodžius galima vienareikšmiškai dekoduoti.

13 apibrėžimas. *Kodą $c : \mathcal{A} \to \mathcal{B}^*$ vadinsime dekoduojamu, jeigu jo tęsinys $c^* : \mathcal{A}^* \to \mathcal{B}^*$ yra injektyvus atvaizdis.*

Dekoduojamo kodo sąvoką apibrėžkime neminėdami šaltinio abėcėlės $\mathcal{A}$.

14 apibrėžimas. *Kodą $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_r\} \subset \mathcal{B}^*$ vadinsime dekoduojamu, jeigu lygybė*

$$\mathbf{x}_1\mathbf{x}_2 \ldots \mathbf{x}_k = \mathbf{y}_1\mathbf{y}_2 \ldots \mathbf{y}_l, \quad \mathbf{x}_i, \mathbf{y}_j \in \mathbf{C},$$

galima tada ir tik tada, kai $k = l$ ir $\mathbf{x}_i = \mathbf{y}_i,\ i = 1, 2, \ldots, k$.

Taigi kodas yra dekoduojamas, jeigu dviems skirtingiems šaltinio žodžiams iš $\mathcal{A}^*$ visada priskiriami skirtingi žodžiai iš $\mathcal{B}^*$.

Pavyzdys. Tegu $\mathcal{A} = \{A, B, C\}$, o kodas $c : \mathcal{A} \to \{0, 1\}^*$ apibrėžiamas taip:

$$c(A) = 0, \quad c(B) = 10, \quad c(C) = 110.$$

Žinoma, kad toks kodas yra dekoduojamas. Kai tik gauname paskutinį šaltinio simboliui priskirto žodžio ženklą, galime nustatyti ir patį šaltinio simbolį.

Pavyzdys. O dabar su tomis pat abėcėlėmis kaip ankstesniame pavyzdyje apibrėžkime kitą kodą:

$$c'(A) = 0, \quad c'(B) = 01, \quad c'(C) = 011.$$

Įsivaizduokime, kad siuntėjas kodavo savo pranešimą šiuo kodu ir siunčia gautą srautą simbolis po simbolio. Taigi gavėjas mato vis ilgėjančią nulių-vienetų eilutę. Jeigu pirmasis simbolis yra 0, gavėjas dar negali nuspręsti, koks buvo pirmasis šaltinio simbolis: jis galėjo būti bet kuris iš trijų. Jeigu antrasis gavėjo simbolis bus 0, gavėjas galės nuspręsti, kad pirmasis simbolis buvo A, tačiau jeigu antrasis gautas simbolis bus 1, gavėjas negalės nieko nuspręsti, kol nesulauks trečiojo simbolio.

Ir vis dėlto – toks kodas yra dekoduojamas. Paprasčiausias dekodavimo būdas: sulaukti, kol perdavimas baigsis, ir dekoduoti gautą srautą nuo galo į priekį!

Akivaizdus šių dviejų dekoduojamų kodų skirtumas: pirmuoju atveju šaltinio simbolį galėjome nustatyti vien tik iš jam priskirto žodžio, antruoju atveju – ne, tekdavo šiek tiek luktelėti.

Natūralu tokius kodus kaip pirmasis pavadinti momentiniais. Juos galima labai paprastai apibūdinti visiškai neminint nei gavėjo, nei jo rūpesčių.

15 apibrėžimas. *Kodą $c : \mathcal{A} \to \mathcal{B}^*$ vadinsime p-kodu, jeigu nei vienas žodis $c(a)$, $a \in \mathcal{A}$, nėra jokio kito žodžio $c(a')$, $a' \in \mathcal{A}, a \neq a'$, priešdėlis.*

Kiek pagalvoję, tikrai įsitikinsite, kad

momentiniai kodai ir p-kodai yra vienas ir tas pats!

Kodo naudojimo aplinkybės, suprantama, diktuoja ir kodų vertinimo kriterijus. Tačiau galime suformuluoti kelis reikalavimus, kurie pageidautini beveik bet kokiu atveju. Kodas turi būti dekoduojamas; kuo jo žodžiai trumpesni, kuo jų daugiau – tuo geriau. Tačiau bet kokiems pageidavimams yra ribos: parinkę daug trumpų žodžių, negalėsime būti tikri, kad kodas bus dekoduojamas.

Panagrinėkime šį uždavinį: kiek ir kokio ilgio žodžių gali turėti kodas, kad jis būtų dekoduojamas? Du paprasti pavyzdžiai padės suvokti esmę.

Pavyzdys. Tegu kodo abėcėlė dvejetainė: $\mathcal{B} = \{0, 1\}$. Ar galime iš šios abėcėlės žodžių sudaryti kodą, kad jų ilgiai būtų $2; 2; 2; 3; 3$? Atsakymas labai paprastas: vos pradėję tokį kodą konstruoti, greitai ir užbaigsime:

$$\mathbf{c}_1 = 00, \quad \mathbf{c}_2 = 01, \quad \mathbf{c}_3 = 10, \quad \mathbf{c}_4 = 111, \quad \mathbf{c}_5 = 110.$$

Sudarėme p-kodą; taigi jis yra dekoduojamas. O dabar pabandykime sukonstruoti dekoduojamą kodą, kurio vienas žodis būtų trumpesnis negu ankstesniame pavyzdyje.

Pavyzdys. Ar galima iš dvejetainės abėcėlės žodžių sudaryti dekoduojamą kodą, kad žodžių ilgiai būtų $2; 2; 2; 2; 3$? Bandykime kaip anksčiau:

$$\mathbf{c}_1 = 00, \quad \mathbf{c}_2 = 01, \quad \mathbf{c}_3 = 10, \quad \mathbf{c}_4 = 11, \quad \mathbf{c}_5 = ?$$

Akivaizdu, kad p-kodo sudaryti negalėsime. Tačiau galbūt galima sudaryti nors ir ne momentinį, tačiau dekoduojamą kodą? Atsakymą duoda tokia teorema.

11 teorema. *Tegu $\mathcal{B}$ yra abėcėlė, $b = |\mathcal{B}|$, $\mathbf{C} \subset \mathcal{B}^*$ yra dekoduojamas kodas iš n žodžių, jų ilgiai yra $s_1, s_2, \ldots, s_n$. Tada teisinga nelygybė*

$$\sum_{i=1}^{n} b^{-s_i} \leq 1. \tag{18}$$

Įrodymas. Pažymėkime $s = \max s_i$, pasirinkime natūralųjį r ir panagrinėkime lygybę

$$(b^{-s_1} + \ldots + b^{-s_n})^r = \sum_{l=1}^{rs} N_l b^{-l}. \tag{19}$$

Kokia gi koeficientų N_l prasmė?

Koeficientas N_l lygus skaičiui l ilgiožodžių, kuriuos galima sudaryti iš r kodo $\mathbf{C}$ žodžių juos jungiant. Kodas $\mathbf{C}$ yra dekoduojamas, todėl visi šie sudurtiniai žodžiai (jeigu jų iš viso yra!) skirtingi. Kadangi iš viso l ilgio žodžių yra b^l, tai

$$N_l \le |\mathcal{B}^l| = b^l, \quad N_l b^{-l} \le 1.$$

Dabar iš (19) nelygybės gauname

$$(b^{-s_1} + \ldots + b^{-s_n})^r \le rs, \quad \text{arba} \quad b^{-s_1} + \ldots + b^{-s_n} \le r^{1/r} s^{1/r}.$$

Ši nelygybė teisinga su visomis r reikšmėmis; be to, kairė nelygybės pusė nepriklauso nuo r. Neaprėžtai didindami r, gausime

$$r^{1/r} s^{1/r} = \mathrm{e}^{\frac{\ln r}{r} + \frac{\ln s}{r}} \to 1.$$

Taigi (18) nelygybė įrodyta.

Dabar jau galime teigti, kad dekoduojamo kodo iš dvinarės abėcėlės žodžių su ilgiais $2; 2; 2; 2; 3$ sukonstruoti negalima, nes žodžių ilgiai netenkina (18) sąlygos.

Įsitikinkime, kad (18) nelygybė garantuoja, kad kodas su iš anksto nustatytais žodžių ilgiais $s_1, s_2, \ldots, s_n$ gali būti sukonstruotas.

12 teorema. *Tegu $\mathcal{B}$ yra abėcėlė, $b = |\mathcal{B}|$, o $s_1, s_2, \ldots, s_n$ yra natūralieji skaičiai, tenkinantys (18) nelygybę. Tada egzistuoja p-kodas $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_n\}$, kad $|c_i| = s_i$, $i = 1, 2, \ldots, n$.*

Įrodymas. Tegu $s = \max s_i$, o n_m $(m = 1, 2, \ldots, s)$ yra kiekis skaičių s_i, kurie lygūs m. Taigi $n_i \ge 0$ ir $n_1 + n_2 + \ldots + n_s = n$. Dabar (18) nelygybę galime užrašyti taip

$$n_1 b^{-1} + n_2 b^{-2} + \ldots + n_s b^{-s} \le 1. \tag{20}$$

Dauginkime abi (20) nelygybės puses iš $b, b^2, \ldots, b^s$ ir pertvarkykime gautas nelygybes, palikdami kairėje pusėje tik $n_1, n_2, \ldots, n_s$:

$$\begin{aligned}
&n_1 \le b - (n_2 b^{-1} + n_2 b^{-2} + \ldots + n_s b^{-s+1}),\\
&n_2 \le b^2 - n_1 b - (n_3 b^{-1} + n_2 b^{-2} + \ldots + n_s b^{-s+2}),\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&n_t \le b^t - n_1 b^{t-1} - n_2 b^{t-2} - \ldots - n_{t-1} b - (n_{t+1} b^{-1} + n_2 b^{-2} + \cdots + n_s b^{-s+t}),\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&n_s \le b^s - n_1 b^{s-1} - n_2 b^{s-2} - \ldots - n_{s-1} b.
\end{aligned}$$

Kadangi $n_i \geq 0$, tai dešinėse nelygybių pusėse apskliausti reiškiniai yra neneigiami, todėl, nubraukę juos, nelygybių „nesugadinsime“:

$$\begin{aligned}
&n_1 \leq b,\\
&n_2 \leq b^2 - n_1 b,\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&n_t \leq b^t - n_1 b^{t-1} - n_2 b^{t-2} - \cdots - n_{t-1} b,\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&n_s \leq b^s - n_1 b^{s-1} - n_2 b^{s-2} - \cdots - n_{s-1} b.
\end{aligned}$$

Dabar galime imtis kodo medžio konstravimo. Iš šaknies išveskime visas b šakas. Lapais paverskime n_1 viršūnių. Iš nepanaudotų viršūnių išveskime po b šakų. Antrajame lygyje turėsime $b^2 - n_1 b$ viršūnių, n_2 iš jų paverskime lapais. Antroji nelygybė garantuoja, kad tai mums pavyks. Iš nepanaudotų antrojo lygio viršūnių veskime po b šakų į trečiąjį lygį. Sukurkime n_3 lapų trečiajame lygyje. Trečioji nelygybė vėl garantuoja mums sėkmę. Tęskime konstrukciją; s-ajame lygyje fiksuokime n_s lapų ir nutrinkime kelius, vedančius iš šaknies į nepanaudotas s-ojo lygio viršūnes. Gausime kodo medį. Belieka vienu iš daugelio būdų priskirti šakoms abėcėlės simbolius.

(18) nelygybė vadinama Krafto-McMillano nelygybe. L. G. Kraftas 1949 m. ją įrodė p-kodams, B. McMillanas 1956 m. – bet kokiems dekoduojamiems kodams.

Kodo su parinktais žodžių ilgiais egzistavimo įrodymas yra konstruktyvus: jame nurodytas algoritmas tokiam kodui sudaryti. Dar viena išvada, kuri išplaukia iš šios teoremos, yra tokia: galime nagrinėti vien tik p-kodus. Iš tiesų, bet kokį dekoduojamą kodą galima pakeisti atitinkamu p-kodu, turinčiu tiek pat to paties ilgio žodžių.

2.4. Optimalūs kodai

Optimalūs maršrutai taupo mūsų laiką ir išteklius. Tą patį daro ir optimalūs kodai.

Kaip konstruoti dekoduojamus kodus, žinome. Dabar nagrinėsime, kaip juos pritaikyti informacijos šaltiniams. Mūsų naudojamas modelis toks: informacijos šaltinis – tai atsitiktinių dydžių, įgyjančių reikšmes iš abėcėlės $\mathcal{A}$, seka

$$\mathcal{U} = \langle U_1, U_2, \ldots \rangle.$$

Žinoma, tikrovėje begaliniai simbolių srautai niekada nepasitaikys. Jeigu apsiribosime tik šaltinio perduotu n ilgio simbolių srautu, tai sakysime, kad nagrinėjame dalinį šaltinį

$$\mathcal{U}_n = \langle U_1, U_2, \ldots, U_n \rangle.$$

Šaltinį $\mathcal{U}_1$ galime tiesiog sutapatinti su atsitiktiniu dydžiu U_1. Nuo šio šaltinio ir pradėkime, t. y. kol kas nagrinėkime vieno abėcėlės $\mathcal{A} = \{a_1, a_2, \ldots, a_r\}$ simbolio kodavimą.

Pažymėkime simbolių perdavimo tikimybes:

$$p_i = P(U_1 = a_i), \quad i = 1, 2, \ldots, r.$$

Tegu $\mathcal{B} = \{b_1, b_2, \ldots, b_s\}$ yra kita abėcėlė. Nagrinėsime dekoduojamus kodus

$$c : \mathcal{A} \to \mathcal{B}^*.$$

Žinome, kad pakanka nagrinėti p-kodus. Tokio kodo žodžiai gali būti įvairaus ilgio. Šaltinio generuotą srautą koduojame jungdami to srauto simboliams priskirtus kodo žodžius. Būtų geriau, jeigu dažniau pasitaikantiems simboliams būtų priskirti trumpesni, o rečiau – ilgesni žodžiai. Kodo tinkamumą šaltiniui galime vertinti vartodami vidutinio žodžių ilgio sąvoką.

16 apibrėžimas. *Vidutiniu p-kodo $c : \mathcal{A} \to \mathcal{B}^*$ žodžių ilgiu vadinsime skaičių*

$$\lambda(c) = \sum_{i=1}^{r} |c(a_i)| \cdot p_i.$$

p-kodą $\hat{c} : \mathcal{A} \to \mathcal{B}^*$ vadinsime optimaliu šaltinio $\mathcal{U}_1$ kodu, jeigu

$$\lambda(\hat{c}) = \min_c \lambda(c),$$

čia minimumas imamas pagal visus p-kodus $c : \mathcal{A} \to \mathcal{B}^*$.

Tegu $b = |\mathcal{B}|$ yra kodo abėcėlės simbolių skaičius, o t – natūralusis skaičius, tenkinantis nelygybes

$$b^{t-1} < r \leq b^t.$$

Tada žodžių aibėje $\mathcal{B}^t$ yra ne mažiau žodžių kaip abėcėlėje $\mathcal{A}$ ir bet kuris injektyvus atvaizdis

$$c' : \mathcal{A} \to \mathcal{B}^t$$

yra p-kodas. Tokio kodo vidutinis žodžio ilgis $\lambda(c') = t$; taigi optimalaus kodo reikia ieškoti kodų, tenkinančių nelygybę

$$\lambda(c) = \sum_{i=1}^{r} p_i |c(a_i)| \leq t, \tag{21}$$

aibėje. Ar optimalus kodas visada egzistuoja? Juk kartais reikšmės, su kuria funkcija įgyja minimalią ar maksimalią reikšmę apibrėžimo srityje, gali ir nebūti: pavyzdžiui, intervale $(0; 1)$ negalime rasti nei paties mažiausio, nei didžiausio skaičiaus.

Tegu p_* yra mažiausia iš tikimybių $p_i, i = 1, 2, \ldots, r$. Tada iš (21) gauname

$$p_* \sum_{i=1}^{r} |c(a_i)| \leq t; \quad \sum_{i=1}^{r} |c(a_i)| \leq t/p_*.$$

Taigi optimalaus kodo reikia ieškoti kodų, tenkinančių nelygybę

$$\sum_{i=1}^{r} |c(a_i)| \le t/p_*,$$

aibėje. Kadangi ši aibė yra baigtinė, tai ir optimalų kodą visada galima rasti.

13 teorema. *Kiekvienam dekoduojamam šaltinio $\mathcal{U}$ kodui $c : \mathcal{A} \to \mathcal{B}^*$ teisinga nelygybė*

$$\lambda(c) \ge \frac{H(U_1)}{\log_2 b}.$$

Tarkime, abėcėlė $\mathcal{B}$ yra dvejetainė. Tada $\lambda(c)$ reiškia vidutinį skaičių dvejetainių simbolių (bitų), naudojamų šaltinio simboliams koduoti. Entropiją $H(U_1)$ galime suvokti kaip informacijos vienetų (bitų) kiekį, kurį perduoda šaltinis. Tada teoremos nelygybę $\lambda(c) \ge H(U_1)$ galime paaiškinti taip: vidutinis kodo žodžių ilgis bitais turi būti ne mažesnis kaip šaltinio perduodamų bitų kiekis.

Įrodymas. Pažymėkime $s_i = |c(a_i)|$ kodo žodžių ilgius, $p_i = P(U_1 = a_i)$ – šaltinio simbolių tikimybes. Vertinsime skirtumą

$$\begin{aligned}\frac{H(U_1)}{\log_2 b} - \lambda(c) &= \frac{1}{\log_2 b} \sum_{i=1}^{n} p_i \left(\log_2 \frac{1}{p_i} - s_i \log_2 b \right) \\ &= \frac{1}{\log_2 b} \sum_{i=1}^{n} p_i \log_2 \frac{b^{-s_i}}{p_i} = \frac{1}{\log_2 b \cdot \ln 2} \sum_{i=1}^{n} p_i \ln \frac{b^{-s_i}}{p_i}.\end{aligned}$$

Pasinaudoję nelygybe $\ln x \le x - 1$ $(x > 0)$, gausime

$$\sum_{i=1}^{n} p_i \ln \frac{b^{-s_i}}{p_i} \le \sum_{i=1}^{n} p_i \left(\frac{b^{-s_i}}{p_i} - 1 \right) = \sum_{i=1}^{n} b^{-s_i} - \sum_{i=1}^{n} p_i \le 0.$$

Paskutinę nelygybę gavome pasinaudoję tuo, kad dekoduojamam kodui teisinga Krafto-McMillano nelygybė, t. y.

$$\sum_{i=1}^{n} b^{-s_i} \le 1,$$

o tikimybių suma lygi vienetui. Taigi gavome

$$\frac{H(U_1)}{\log_2 b} - \lambda(c) \le 0.$$

Teorema įrodyta.

Iš teoremos įrodymo matyti, kad atskiru atveju, kai šaltinio tikimybės yra skaičiaus b laipsniai, t. y. $p_i = b^{-s_i}$, $i = 1, 2, \ldots, n$, teoremos nelygybė virsta

tikslia lygybe: $\lambda(c) = H(U_1)$. Aišku, kad toks šaltinio tikimybių ir kodo abėcėlės dydžio ryšys yra išskirtinis atvejis. O ką galime pasiekti bendruoju atveju?

14 teorema. *Egzistuoja šaltinio $\mathcal{U}$ kodas $\hat{c} : \mathcal{A} \to \mathcal{B}^*$, kuriam teisinga nelygybė*

$$\lambda(\hat{c}) < \frac{H(U_1)}{\log_2 b} + 1. \tag{22}$$

Įrodymas. Tarkime, šaltinio tikimybės išrikiuotos didėjimo tvarka:

$$p_1 \le p_2 \le \ldots \le p_r.$$

Parinkime natūraliuosius skaičius $1 \le s_1 \le s_2 \le \ldots \le s_r$, kad jie tenkintų nelygybes

$$b^{-s_i} \le p_i < b^{-s_i+1}, \quad i = 1, 2, \ldots, r.$$

Akivaizdu, kad juos galime parinkti vieninteliu būdu. Kadangi

$$\sum_{i=1}^{r} b^{-s_i} \le \sum_{i=1}^{r} p_i = 1,$$

tai galima sudaryti p-kodą $\hat{c} : \mathcal{A} \to \mathcal{B}^*$, kad $|\hat{c}(a_i)| = s_i$. Įsitikinkime, kad šiam kodui teisinga (22) nelygybė.

Iš tikrųjų, iš nelygybės $p_i < b^{-s_i+1}$ gauname

$$s_i - 1 < \log_b \frac{1}{p_i}, \quad s_i < \frac{1}{\log_2 b} \cdot \log_2 \frac{1}{p_i} + 1.$$

Taigi

$$\lambda(\hat{c}) = \sum_{i=1}^{r} p_i s_i < \frac{1}{\log_2 b} \sum_{i=1}^{r} p_i \log_2 \frac{1}{p_i} + \sum_{i=1}^{r} p_i = \frac{H(U_1)}{\log_2 b} + 1.$$

Teorema įrodyta.

Iš abiejų teoremų galime padaryti tokią išvadą:

15 teorema. *Optimalus šaltinio $\mathcal{U}_1$ kodas tenkina nelygybę*

$$\frac{H(U_1)}{\log_2 b} \le \lambda(\hat{c}) < \frac{H(U_1)}{\log_2 b} + 1. \tag{23}$$

O dabar aptarkime „didžiojo“ šaltinio $\mathcal{U}$ generuoto simbolių srauto kodavimą.

Tarkime, kad reikia koduoti dalinio šaltinio $\mathcal{U}_n = \langle U_1, U_2, \ldots, U_n \rangle$ generuotus žodžius, t. y. žodžius iš aibės $\mathcal{A}^n$. Šiems žodžiams koduoti galime sugalvoti kokią nors taisyklę

$$c_n : \mathcal{A}^n \to \mathcal{B}^*.$$

Vidutinis kodo žodžių ilgis

$$\lambda(c_n) = \sum_{\mathbf{a}\in\mathcal{A}^n} |c_n(\mathbf{a})| P(\mathcal{U}_n = \mathbf{a}).$$

Jeigu ilginsime koduojamą srautą, t. y. n didės, tai didės ir $\lambda(c_n)$. Apibrėžkime dydį, kuris nusako, kiek vidutiniškai $\mathcal{B}$ abėcėlės simbolių kodas panaudoja vienam šaltinio simboliui koduoti.

17 apibrėžimas. *Tegu $c_n : \mathcal{A}^n \to \mathcal{B}^*$ yra šaltinio $\mathcal{U}_n = \langle U_1, U_2, \ldots, U_n\rangle$ kodas. Kodo vienetinės sąnaudos koeficientu vadinsime skaičių*

$$\bar{\lambda}(c_n) = \frac{\lambda(c_n)}{n}.$$

Žinome, kad šaltiniai būna įvairūs. Nagrinėkime paprasčiausią atvejį – Bernulio šaltinį. Jeigu $\mathcal{U}$ yra Bernulio šaltinis, tai atsitiktiniai dydžiai U_i yra vienodai pasiskirstę ir nepriklausomi. Taigi optimalus kodas ir pirmajam, ir antrajam, ir kitiems srauto simboliams koduoti yra tas pats. Sudarę optimalų kodą $c : \mathcal{A} \to \mathcal{B}^*$ vienam simboliui koduoti, galėtume apibrėžti šaltinio $\mathcal{U}_n = \langle U_1, U_2, \ldots, U_n\rangle$ kodą $c_n : \mathcal{A}^n \to \mathcal{B}^*$ kaip c tęsinį:

$$c_n(x_1x_2\ldots x_n) = c(x_1)c(x_2)\ldots c(x_n), \quad x_j \in \mathcal{A}. \tag{24}$$

Nesunku įsitikinti, kad

$$\lambda(c_n) = n\lambda(c), \quad \bar{\lambda}(c_n) = \lambda(c),$$

taigi vienetinės kodo c_n sąnaudos būtų tokios pačios kaip ir kodo c. Jeigu c būtų optimalus kodas vienam simboliui koduoti, tai iš įrodytų teoremų gautume

$$\frac{H(U_1)}{\log_2 b} \le \bar{\lambda}(c_n) < \frac{H(U_1)}{\log_2 b} + 1. \tag{25}$$

Dar kartą pasinaudokime tomis pačiomis teoremomis, tačiau kitaip. Galime suvokti $\mathcal{U}_n = \langle U_1, U_2, \ldots, U_n\rangle$ kaip vieną abėcėlės $\mathcal{A}^n$ simbolį perduodantį šaltinį. Pritaikę 13 ir 14 teoremas, gausime, kad egzistuoja kodas $c_n : \mathcal{A}^n \to \mathcal{B}^*$, tenkinantis sąlygą

$$\frac{H(U_1, U_2, \ldots, U_n)}{\log_2 b} \le \lambda(c_n) < \frac{H(U_1, U_2, \ldots, U_n)}{\log_2 b} + 1,$$

arba

$$\frac{H(U_1, U_2, \ldots, U_n)}{n\log_2 b} \le \bar{\lambda}(c_n) < \frac{H(U_1, U_2, \ldots, U_n)}{n\log_2 b} + \frac{1}{n}. \tag{26}$$

Šis teiginys teisingas bet kokiems šaltiniams. Jeigu nagrinėjame Bernulio šaltinį, tai $H(U_1, U_2, \ldots, U_n) = nH(U_1) = nH(\mathcal{U})$. Taigi iš (26) gauname tokį teiginį:

16 teorema. *Jeigu $\mathcal{U} = \langle U_1, U_2, \ldots\rangle$ yra Bernulio šaltinis, tai kiekvienam n egzistuoja dalinio šaltinio $\mathcal{U}_n = \langle U_1, U_2, \ldots, U_n\rangle$ kodas $c_n : \mathcal{A}^n \to \mathcal{B}^*$, tenkinantis sąlygą*

$$\frac{H(\mathcal{U})}{\log_2 b} \le \bar{\lambda}(c_n) < \frac{H(\mathcal{U})}{\log_2 b} + \frac{1}{n},$$

čia $H(\mathcal{U}) = H(U_1)$ yra šaltinio entropija.

Nedaug pasikeistų teiginys, jeigu atsisakytume sąlygos, kad šaltinis būtų Bernulio!

17 teorema. *Tegu $\mathcal{U} = \langle U_1, U_2, \ldots\rangle$ yra informacijos šaltinis, kurio entropija yra $H(\mathcal{U})$. Tada bet kokiam skaičiui $\epsilon > 0$ egzistuoja toks $n(\epsilon)$, kad kiekvienam $n > n(\epsilon)$ egzistuoja dalinio šaltinio $\mathcal{U}_n = \langle U_1, U_2, \ldots, U_n\rangle$ kodas $c_n : \mathcal{A}^n \to \mathcal{B}^*$, tenkinantis sąlygą*

$$\frac{H(\mathcal{U})}{\log_2 b} - \epsilon \le \bar{\lambda}(c_n) < \frac{H(\mathcal{U})}{\log_2 b} + \epsilon.$$

Teiginį galime įrodyti samprotaudami visiškai taip pat kaip Bernulio šaltinio atveju ir pasiremdami bendruoju šaltinio entropijos apibrėžimu.

2.5. Shannono ir Fano kodai

Du praktiški būdai kodams konstruoti. Deja, garantijos, kad gausime patį geriausią kodą, nėra.

Įrodydami vieną ankstesnio skyrelio teoremą, panaudojome kodą, kurio žodžių ilgius apibrėžia nelygybės

$$b^{-s_i} \le p_i < b^{-s_i+1}, \quad i = 1, 2, \ldots, r.$$

Įrodymui pakako, kad toks kodas tikrai egzistuoja. Sudarę kodo medį, galėtume sukonstruoti daug kodų, turinčių $s_1 \le s_2 \le \ldots \le s_r$ ilgio žodžius. Šiuos kodus vadinsime Shannono kodais. Vieną iš jų pasinaudoję paties Shannono metodu, galime sukonstruoti ir nebraižydami kodo medžio.

Tegu $b > 1$ yra natūralusis skaičius (mūsų atveju - abėcėlės simbolių skaičius). Kiekvieną intervalo $(0;1)$ skaičių α galime užrašyti b-aine trupmena:

$$\alpha = \frac{d_1}{b} + \frac{d_2}{b^2} + \frac{d_3}{b^3} + \ldots, \quad d_i \in \{0, 1, \ldots, b-1\}.$$

Elementus b_i vadinsime b-ainiais skaitmenimis. Visi skaičiai užrašomi b-ainėmis trupmenomis vieninteliu būdu, išskyrus paprastąsias nesuprastinamas trupmenas

$$\frac{m}{b^k}, \quad k \ge 1.$$

Pavyzdžiui, su $b = 3$

$$\frac{5}{27} = \frac{0}{3} + \frac{1}{3^2} + \frac{2}{3^3} + \frac{0}{3^4} + \ldots = \frac{0}{3} + \frac{1}{3^2} + \frac{1}{3^3} + \frac{2}{3^4} + \frac{2}{3^5} + \ldots$$

Susitarkime, kad tokioms trupmenoms bus naudojama trupmena su „nulių uodega“, t. y. iš tiesų baigtinė trupmena.

Dabar Shannono kodo konstrukciją galima aprašyti taip. Tegu p_i yra šaltinio simbolių tikimybės, o s_i – Shannono kodo žodžių ilgiai:

$$\begin{aligned} &p_1 \geq p_2 \geq \ldots \geq p_n, \\ &b^{-s_i} \leq p_i < b^{-s_i+1}, \quad i = 1, 2, \ldots, r, \\ &s_1 \leq s_2 \leq \ldots \leq s_n. \end{aligned}$$

Sudarykime tikimybių sumas:

$$F_1 = 0,\ F_2 = p_1,\ F_3 = p_1 + p_2,\ \ldots,\ F_r = p_1 + p_2 + \ldots + p_{r-1}.$$

Dabar Shannono kodą $\hat{c} : \mathcal{A} \to \mathcal{B}^*, \mathcal{A} = \{a_1, \ldots, a_r\},\ \mathcal{B} = \{1, 2, \ldots, b\}$ apibrėžkime taip:

$\hat{c}(a_i)$ = žodis iš pirmųjų s_i skaičiaus F_i skleidinio b-aine trupmena skaitmenų.

Pavyzdžiui, $\hat{c}(a_1) = 00\ldots0$, iš viso žodyje yra s_1 nulių.

Jeigu toks kodas yra p-kodas, tai jis tikrai yra Shannono kodas. Įsitikinkime, kad joks žodis $w_k = \hat{c}(a_k)$ negali būti kito žodžio $w_{k+m} = \hat{c}(a_{k+m}),\ m \geq 1$, priešdėlis. Jeigu taip būtų, tai skaičių F_k skleidinio b-aine trupmena pirmieji s_k skaitmenys sutaptų su F_{k+m} skleidinio pirmaisiais s_k skaitmenimis. Tada būtų

$$F_{k+m} - F_k < \frac{b-1}{b^{s_k+1}} + \frac{b-1}{b^{s_k+2}} + \ldots = \frac{1}{b^{s_k}}.$$

Tačiau $F_{k+m} - F_k \geq p_k \geq b^{-s_k}$. Taigi tarę, kad $\hat{c}$ nėra p-kodas, gavome prieštarą. Sukonstruotas kodas yra tikrai p-kodas.

Pavyzdys. Panagrinėkime pavyzdį su tikimybėmis $p_1 = p_2 = 0,3;\ p_3 = 0,2;\ p_4 = p_5 = 0,1$ ir $b = 2,3$. Shannono kodų žodžių ilgiai bus tokie:

	s_1	s_2	s_3	s_4	s_5
$b = 2$	2	2	3	4	4
$b = 3$	2	2	2	3	3

Taigi dvejetainės abėcėlės atveju pakaks trijų skleidinio skaitmenų, o trejetainės – dviejų.

	F_0	F_1	F_2	F_3	F_4
$b = 10$	0	0,3	0,6	0,8	0,9
$b = 2$	0,00...	0,01...	0,100...	0,1100...	0,1110...
$b = 3$	0,00...	0,02.....	0,12...	0,210...	0,220...

Skaičiai F_i dešimtainėje, dvejetainėje ir trejetainėje sistemose. Užrašyti skaitmenys po kablelio sudaro Shannono kodo žodžius. Apskaičiavę gauname, kad dvejetainio Shannono kodo vidutinis žodžių ilgis lygus $2,6$, *o trejetainio* – $2,2$.

Dar vieną kodų sudarymo būdą sugalvojo R. M. Fano. Jį C. Shannonas mini savo darbe „Mathematical Theory of Communication“, pabrėždamas, kad abiem algoritmais gaunami kodai beveik tokie patys. Todėl kodavimo literatūroje šiais metodais sukonstruoti kodai dažnai vadinami tiesiog Shannono-Fano kodais.

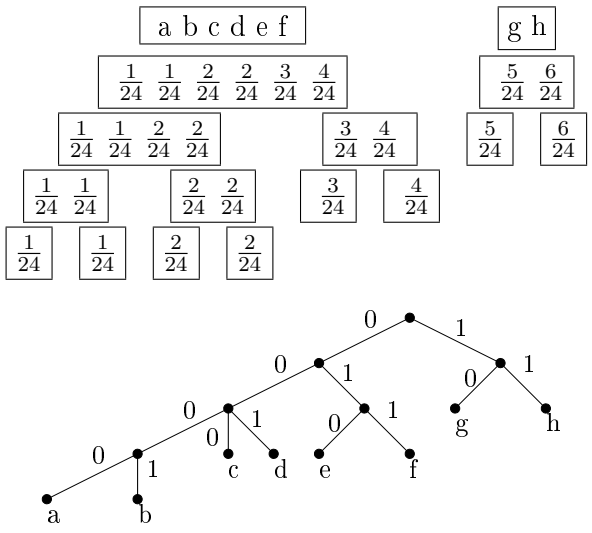
Geriausia Fano kodo konstrukciją nagrinėti pasitelkus pavyzdį. Tarkime, šaltinis generuoja abėcėlės

$$\mathcal{A} = \{a, b, c, d, e, f, g, h\}$$

simbolius, šių simbolių tikimybės pateiktos lentelėje:

a	b	c	d	e	f	g	h
$\frac{1}{24}$	$\frac{1}{24}$	$\frac{2}{24}$	$\frac{2}{24}$	$\frac{3}{24}$	$\frac{4}{24}$	$\frac{5}{24}$	$\frac{6}{24}$

Šaltiniui koduoti norime sudaryti dvejetainės abėcėlės kodą. Kodo sudarymas prasideda nuo abėcėlės dalijimo į dvi dalis siekiant, kad abiejų dalių simbolių tikimybių sumos skirtųsi kiek įmanoma mažiau. Antrajame žingsnyje į dvi dalis dalijamos pirmajame žingsnyje sudarytos abėcėlės simbolių aibės ir taip toliau, kol nebėra ko dalinti. Baigus dalyti galima nubraižyti kodo medį ir, priskyrus jo šakoms simbolius, sukonstruoti kodą. Tokios konstrukcijos pavyzdys pateiktas brėžinyje.



Kiekviename žingsnyje simbolių grupės dalijamos į dvi dalis taip, kad abiejų dalių simbolių tikimybių sumos skirtųsi kiek įmanoma mažiau. Akylas skaitytojas pastebės, kad priešpaskutiniame žingsnyje vieną tokį dalijimą galima buvo atlikti ir geriau. Tačiau toks geresnis dalijimas nepakeistų vidutinio kodo ilgio. Pagal padalytą abėcėlę galime nubraižyti kodo medį.

Šiuo metodu galime konstruoti kodus ne vien tik iš dvejetainės abėcėlės. Jeigu kodo abėcėlėje yra s simbolių, tai pirmajame žingsnyje abėcėlę reikia dalyti į s dalių, siekiant, kad visų dalių simbolių tikimybių sumos būtų apylygės.

Abi kodo sudarymo taisyklės – Shannono ir Fano – negarantuoja, kad, naudodamiesi jomis, gausime optimalų kodą. Kaipgi sudaryti optimalius kodus?

2.6. Huffmano kodai

Jeigu būtų pasiūlyta rinktis: atlikti projektą ar laikyti egzaminą, daugelis studentų rinktųsi pirmąjį variantą. Taip 1951 metais nusprendė ir D. A. Huffmanas, pasirinkęs atsiskaitymui profesoriaus R. M. Fano pasiūlytą uždavinį – sukurti efektyvų dvejetainį kodą. Nors uždavinys neatrodė sunkus, tačiau darbas nesisekė. Iki egzamino liko vos savaitė, ir D. A. Huffmanas jau ketino atsisakyti projekto ir pradėti ruoštis egzaminui. Ir tada švystelėjo idėja...

Tegu $\mathcal{A} = \{a_1, a_2, \ldots, a_r\}$ yra šaltinio abėcėlė, $p(a_i)$, $i = 1, 2, \ldots, r$, – abėcėlės simbolių perdavimo tikimybės. Šį tikimybių rinkinį – diskretųjį tikimybinį skirstinį – žymėsime

$$P(\mathcal{A}) = \langle p(a_1), p(a_2), \ldots, p(a_r) \rangle.$$

Pasirinkime abėcėlę $\mathcal{B}$ kodo žodžiams sudaryti. Mūsų tikslas – išsiaiškinti, kaip, turint porą $\langle \mathcal{A}, P(\mathcal{A}) \rangle$, galima sudaryti optimalų kodą

$$c : \mathcal{A} \to \mathcal{B}^*.$$

Optimalaus kodo sudarymo algoritmą 1952 m. paskelbė D. A. Huffmanas, tad algoritmas ir vadinamas jo vardu. Juo naudojantis sudarytus kodus vadinsime **Huffmano kodais** (r-nariais Huffmano kodais, kai norėsime pabrėžti, kiek simbolių naudojama koduojant).

Iš pradžių panagrinėsime kodavimo dvinarės abėcėlės žodžiais atvejį, t. y. atvejį, kai $\mathcal{B} = \{0, 1\}$. Huffmano algoritmą galima suskaidyti į dvi procedūras. Pirmoji – abėcėlių redukcija:

$$\langle \mathcal{A}_1, P(\mathcal{A}_1) \rangle \to \langle \mathcal{A}_2, P(\mathcal{A}_2) \rangle \to \ldots \to \langle \mathcal{A}_{r-1}, P(\mathcal{A}_{r-1}) \rangle;$$

čia $\mathcal{A}_k$ yra abėcėlės, $\mathcal{A}_1 = \mathcal{A}, |\mathcal{A}_k| = r - k + 1$, o $P(\mathcal{A}_k)$ – šias abėcėles atitinkantys diskretieji skirstiniai, t. y. simbolių tikimybių rinkiniai.

Antroji procedūra – Huffmano kodų sudarymas:

$$c_{r-1} \to c_{r-2} \to \ldots \to c_1; \tag{27}$$

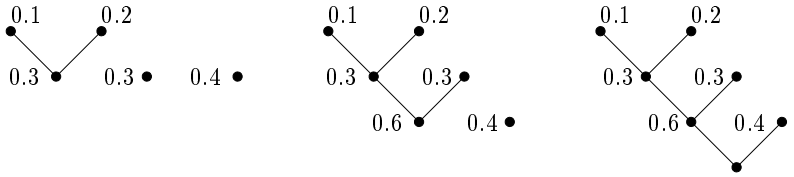
čia c_k – porą $\langle \mathcal{A}_k, P(\mathcal{A}_k) \rangle$ atitinkantis Huffmano kodas.

Abėcėlių redukcijos procese (27) žingsniai atliekami pagal tokią taisyklę:

jei $\mathcal{A}_k = \{a_1, \ldots, a_{r-k+1}\}, P(\mathcal{A}_k) = \langle p(a_1), \ldots, p(a_{r-k+1})\rangle$ *ir tikimybės* $p(a_i), p(a_j)$ *yra mažiausios, tai*

$$\begin{aligned} \mathcal{A}_{k+1} &= \big(\mathcal{A}_k \setminus \{a_i, a_j\}\big) \cup \{\langle a_i, a_j\rangle\}, \\ P(\mathcal{A}_{k+1}) &= \big(P(\mathcal{A}_k) \setminus \{p(a_i), p(a_j)\}\big) \cup \{p(\langle a_i, a_j\rangle)\}, \\ p(\langle a_i, a_j\rangle) &= p(a_i) + p(a_j). \end{aligned}$$

Taigi pereinant prie naujos abėcėlės, du mažiausias tikimybes turintys simboliai keičiami nauju (mūsų schemoje šis naujas simbolis vaizduojamas tiesiog simbolių pora), o jam priskiriama tikimybė lygi mažiausiųjų tikimybių sumai. Nesunku pastebėti, kad abėcėlė $\mathcal{A}_{k+1}$ turi vienu simboliu mažiau negu $\mathcal{A}_k$, tad redukcijos procesas baigiasi, kai gaunama tik du simbolius turinti abėcėlė. Abėcėlių redukcijos procesą atitinka kodo medžio braižymas „nuo viršaus“. Vaizduokime abėcėlės elementus grafo viršūnėmis, kurioms priskirti „svoriai“ – tuos simbolius atitinkančios tikimybės. Tada dviejų simbolių „suliejimas“ į vieną reiškia, jog iš dviejų viršūnių vedamos šakos į naują – vidinę.



Huffmano kodo sudarymas tikimybių skirstiniui $P(\mathcal{A}) = \langle 0,1; 0,2; 0,3; 0,4\rangle$.
Vienas iš kodų, sudarytas pagal kodo medį, toks: $\mathbf{C} = \{111, 110, 10, 0\}$.
Jo vidutinis ilgis lygus $1,9$.

Patogu vaizduoti abėcėlių redukciją ir lentele. 1 lentelė sudaryta tam pačiam skirstiniui kaip ir diagramoje pavaizduotas medis. Stulpelyje $\mathcal{A}_k$ surašytos abėcėlės $\mathcal{A}_k$ simbolius atitinkančios tikimybės, žvaigždutės dešinėje žymi, kurie simboliai kitu žingsniu bus jungiami į vieną, pabraukimas reiškia, kad tikimybė (simbolis) ankstesniu žingsniu yra gauta iš dviejų. Tušti stulpeliai skirti kodams, kuriuos gausime atlikę (27) perėjimus, užrašyti.

$\mathcal{A}_1$	c_1	$\mathcal{A}_2$	c_2	$\mathcal{A}_3$	c_3
0.4		0.4		<u>0.6</u>	
0.3		0.3*		0.4	
0.2*		<u>0.3*</u>			
0.1*					

1 lentelė. Huffmano kodo sudarymas

Dabar išsamiau aptarkime (27) procesą, t. y. kodo sudarymą. Jeigu abėcėlių redukciją interpretuojate kaip kodo medžio braižymą, tai kodo sudarymas pagal gautą medį – jau žinomas dalykas. Vis dėlto (27) perėjimus aprašykime formaliai.

Kadangi abėcėlę $\mathcal{A}_{r-1}$ sudaro tik du simboliai, tai kad ir koks būtų ją atitinkantis diskretusis skirstinys, optimalus kodas yra akivaizdus: vieną simbolį reikia koduoti 0, kitą – 1.

1. *Abėcėlės $\mathcal{A}_{r-1}$ simbolius koduojame 0 ir 1.*

2. *Jei abėcėlei $\mathcal{A}_k, k \geq r-1$, kodas c_k sudarytas ir simbolis $a \in \mathcal{A}_{k-1} \cap \mathcal{A}_k$, tai $c_{k-1}(a) = c_k(a)$. Jei $\langle a, a' \rangle \in \mathcal{A}_k$, tai $c_{k-1}(a) = c_k(\langle a, a' \rangle)0$, $c_{k-1}(a') = c_k(\langle a, a' \rangle)1$.*

Dabar galime užpildyti tuščius 1 lentelės stulpelius.

$\mathcal{A}_1$	c_1	$\mathcal{A}_2$	c_2	$\mathcal{A}_3$	c_3
0.4	0	0.4	0	$\underline{0.6}$	1
0.3	10	0.3*	10	0.4	0
0.2*	110	$\underline{0.3^*}$	11		
0.1*	111				

2 lentelė. Huffmano kodo sudarymas

Skaitytojas, be abejo, pastebės, jog algoritmą nesunku modifikuoti, kad jis tiktų kodui iš s-narės abėcėlės žodžių sudaryti. Redukuojant abėcėles, vienu simboliu reikia keisti ne porą, bet s simbolių rinkinį. Tiesa, gali atsitikti, kad po paskutinės redukcijos gausime abėcėlę iš mažiau nei s simbolių. Siekdami, kad jų liktų lygiai s, nustatykime, kiek simbolių reikia sujungti pirmuoju redukcijos žingsniu.

Jei $|\mathcal{A}| = r, |\mathcal{B}| = s$ ir abėcėlių redukcijos procese atlikome t žingsnių, pirmajame sujungdami $u \leq r$ simbolių, o visuose kituose po s, tai $r = (u-1) + (t-1)(s-1) + s$, arba

$$u \equiv r \pmod{(s-1)}, \quad 2 \leq u \leq s.$$

Ši sąlyga vienareikšmiškai apibrėžia pirmuoju žingsniu sujungiamų simbolių skaičių.

3 lentelėje aprašytas trinario Huffmano kodo sudarymas. Kadangi koduojamų simbolių skaičius $r = 6$, $s = 3$, tai pirmuoju žingsniu reikia sujungti 2 simbolius.

$\mathcal{A}_1$	c_1	$\mathcal{A}_2$	c_2	$\mathcal{A}_3$	c_3
0.4	1	0.4	1	$\underline{0.4}$	0
0.2	2	0.2	2	0.4	1
0.2	00	0.2*	00	0.2	2
0.1	01	0.1*	01		
0.05*	020	$\underline{0.1^*}$	02		
0.05*	021				

3 lentelė. Huffmano kodo sudarymas iš trinarės abėcėlės žodžių

Įrodysime, jog Huffmano kodai yra optimalūs. Apsiribosime teoremos įrodymu dvinarės abėcėlės Huffmano kodams. Įrodymo idėjos lieka tos pačios ir bendruoju atveju, bet, panaudojus jas, sukurti įrodymą nėra itin paprasta.

Jei $a \in \mathcal{A}$, tai šį simbolį atitinkančią tikimybę žymėsime $p(a)$. Iš pradžių įrodysime tokį teiginį.

18 teorema. *Jei $a_1, a_2 \in \mathcal{A}$ ir $p(a_1), p(a_2)$ yra mažiausios nenulinės tikimybės, tai egzistuoja optimalus kodas d, kad*

$$|d(a_1)| = |d(a_2)| = \max_a |d(a)|,$$

o žodžiai $d(a_1), d(a_2)$ skiriasi tik paskutiniu simboliu.

Įrodymas. Priminsime, kad nagrinėjame tik p-kodus. Jei kodas d yra optimalus, tai dydžio

$$\lambda(d) = \sum_{a \in \mathcal{A}} p(a)|d(a)|$$

reikšmė yra mažiausia.

Tarkime, egzistuoja tik vienas optimalaus kodo d žodis $d(a)$, turintis maksimalų ilgį. Sutrumpinę žodį $d(a)$ paskutiniu simboliu, gautume naują p-kodą d', kuriam $\lambda(d') = \lambda(d) - p(a) < l(d)$. Tai prieštarautų kodo d optimalumui. Vadinasi, egzistuoja keli maksimalaus ilgio žodžiai. Sakykime, bet kuri tokių žodžių pora skiriasi kuriuo nors ne paskutiniu simboliu. Sutrumpinę visus maksimalaus ilgio žodžius paskutiniais simboliais, vėl gautume geresnį už d kodą. Taigi egzistuoja du maksimalaus ilgio žodžiai $d(a_1'), d(a_2')$, kurie skiriasi tik paskutiniu simboliu.

Jeigu $\{a_1, a_2\} = \{a_1', a_2'\}$, tai kodas d tenkina teoremos sąlygas. Jeigu $\{a_1, a_2\} \neq \{a_1', a_2'\}$, tai, turėdami galvoje, kad žodžių a_1, a_2 tikimybės mažiausios, tarkime

$$p(a_1) \leq p(a_1'), \quad p(a_2) \leq p(a_2').$$

Sukeitę kodo d žodžius $d(a_1)$ ir $d(a_1')$, $d(a_2)$ ir $d(a_2')$, gausime naują kodą d', kuris simbolius a_1, a_2 koduoja maksimalaus ilgio žodžiais $d'(a_1) = d(a_1)$, $d'(a_2) = d(a_2)$, besiskiriančiais tik paskutiniu simboliu.

Kadangi d yra optimalus kodas, tai $\lambda(d') \geq \lambda(d)$. Kita vertus,

$$\lambda(d') = \lambda(d) - (|d(a_1')| - |d(a_1)|)(p(a_1') - p(a_1)) -$$

$$-(|d(a_2')| - |d(a_2)|)(p(a_2') - p(a_2)) \leq \lambda(d).$$

Taigi kodas d' irgi yra optimalus. Jis tenkina teoremos sąlygas.

19 teorema. *Huffmano kodai yra optimalūs.*

Įrodymas. Visi kodai (27) kodų grandinėje yra Huffmano kodai. Įrodysime, kad visi jie yra optimalūs. Kodas c_{r-1} yra, žinoma, optimalus. Tarkime, kodas c_k, $1 < k \leq r-1$, yra optimalus, ir nagrinėkime kodą c_{k-1}.

Tarkime, $a, a' \in \mathcal{A}_{k-1}$ yra mažiausias pasirodymo tikimybes $p(a)$ ir $p(a')$ turintys simboliai, kuriuos sujungę gauname abėcėlę $\mathcal{A}_k$.

Jei kodas c_{k-1} nėra optimalus, tai iš 18 teoremos išplaukia, jog egzistuoja optimalus abėcėlės $\mathcal{A}_{k-1}$ kodas d, atitinkantis diskretųjį skirstinį $P(\mathcal{A}_{k-1})$, kad žodžių $d(a)$, $d(a')$ ilgiai yra maksimalūs, o žodžiai skiriasi tik paskutiniu simboliu. Taigi

$$\lambda(d) = \sum_{a \in \mathcal{A}_{k-1}} p(a)|d(a)| < \sum_{a \in \mathcal{A}_{k-1}} p(a)|c_{k-1}(a)| = \lambda(c_{k-1}). \qquad (28)$$

Iš kodų c_{k-1}, d gausime abėcėlės $\mathcal{A}_k$ kodus c_k ir d', jei simbolį $\langle a, a' \rangle$ koduosime žodžiais, kuriuos gauname sutrumpinę $c_{k-1}(a)$ ir $d(a)$ paskutiniu simboliu. Tada

$$\begin{aligned} \lambda(c_k) &= \lambda(c_{k-1}) - |c_{k-1}(a)|p(a) - |c_{k-1}(a')|p(a') \\ &+ (|c_{k-1}(a)| - 1)(p(a) + p(a')), \\ \lambda(c_k) &= \lambda(c_{k-1}) - (p(a) + p(a')). \end{aligned}$$

Analogiškai

$$\lambda(d') = \lambda(d) - (p(a) + p(a')).$$

Kadangi pagal prielaidą kodas c_k yra optimalus, tai $\lambda(c_k) \leq \lambda(d')$. Tačiau iš gautų lygybių išplaukia, jog tada taip pat $\lambda(c_{k-1}) \leq \lambda(d)$. Tai prieštarauja (28) nelygybei, vadinasi, prielaida, kad kodas c_{k-1} nėra optimalus, yra klaidinga.

Teorema įrodyta.

3 Duomenų spūda

Bernulio šaltinių informacijai koduoti sukonstravome optimalų Huffmano kodą. Tačiau Bernulio šaltinių savo gyvenime beveik nesutinkame! Niekas

nedrįs teigti, kad šios dienos dienraščio straipsnis, skaitmeniniu fotoaparatu padaryta nuotrauka ar muzikinis įrašas yra informacijos srautai, kuriuos sukūrė Bernulio šaltiniai! Vis dėlto, jei šaltinis ir nėra Bernulio, tai nereiškia, kad jis nepanašus į jį. Natūralioje kalboje simboliai turi savo pasitaikymo dažnius. Taigi manydami, kad kalbos srautus generuoja Bernulio šaltinis su tikimybėmis, lygiomis simbolių pasitaikymo dažniams, ir taikydami kalbai koduoti Huffmano kodą, elgsimės teisingai. Koduoti duomenys užims mažiau vietos, taigi juos suspausime, tačiau tikriausiai ne tiek, kiek įmanoma. Juk Huffmano kodu koduojame neatsižvelgdami į koduojamų duomenų struktūrą – vien tik į šaltinio tikimybes. Darbuodamiesi visai nekreipiame dėmesio, ar koduojama seka yra būdinga šaltiniui, ar ne. Taigi optimalūs kodai nebūtinai visada optimalūs.

Sąlyginai galime suskirstyti įvairius duomenų spaudimo metodus į dvi grupes – statistinius ir kitokius. Statistiniais vadiname tuos metodus, kurie remiasi žiniomis apie simbolių dažnius. Taigi duomenų spaudimas koduojant juos Huffmano kodu – statistinis metodas.

3.1. Aritmetinis kodavimas

Matematinėje analizėje garbingą vietą užima „susitraukiančių intervalų“ lema: jeigu sukonstruosite uždarų nykstamai mažėjančio ilgio intervalų „teleskopą“ $[a_1; b_1] \supset [a_2; b_2] \supset \ldots$ egzistuos skaičius, priklausantis visiems intervalams. Aritmetinis kodavimas yra panašus metodas: simbolių blokui konstruojamas panašus teleskopas, kol jo gale randamas skaičius – simbolių bloko kodas.

Bernulio šaltinio generuotų duomenų aritmetinis kodavimas yra statistinis metodas: ir koduojant, ir dekoduojant reikia žinoti abėcėlės $\mathcal{A} = \{a_1, a_2, \ldots, a_r\}$ simbolių pasirodymo dažnius $p_1, p_2, \ldots, p_r$. Šiuo metodu sudarytą kodą vadinsime aritmetiniu.

Tarkime, koduojami n ilgio žodžiai, visų galimų žodžių aibė yra $\mathcal{A}^n$. Kiekvienam šios aibės žodžiui $\mathbf{x} = a_{i_1} a_{i_2} \ldots a_{i_n}$ intervale $I = [0; 1]$ tam tikru būdu priskirkime jo „teritoriją“ – intervalą $I(\mathbf{x}) = I(i_1, i_2, \ldots, i_n)$. Pasirūpinę, kad skirtingiems žodžiams $\mathbf{x}, \mathbf{y}$ jų intervalai $I(\mathbf{x}), I(\mathbf{y})$ nesikirstų, žodžio $\mathbf{x}$ kodu galėtume laikyti bet kurį skaičių $\rho \in I(\mathbf{x})$. Užrašę šį skaičių dvejetaine išraiška galėtume pasiųsti dvejetainių simbolių žodį gavėjui, tikėdamiesi, kad pagal skaičių jis susiras intervalą $I(\mathbf{x})$ ir patį žodį $\mathbf{x}$. Kaip ši idėja gali būti įgyvendinta?

Žodį $\mathbf{x} = a_{i_1} a_{i_2} \ldots a_{i_n}$ atitinkančio intervalo ieškosime konstruodami „teleskopą“:

$$I \supset I(i_1) \supset I(i_1, i_2) \supset \ldots \supset I(i_1, i_2, \ldots, i_{n-1}) \supset I(i_1, i_2, \ldots, i_{n-1}, i_n) = I(\mathbf{x}).$$

Pirmiausia paskirsime intervalus pavienėms raidėms:

$$\begin{aligned} I(1) &= [0; p_1),\ I(2) = [p_1; p_1 + p_2), \\ I(3) &= [p_1 + p_2; p_1 + p_2 + p_3),\ \ldots, I(r) = [p_1 + p_2 + \ldots + p_{r-1}; 1). \end{aligned}$$

Toliau intervalų priskyrimą galime aprašyti rekurentiškai. Jeigu

$$I(i_1, i_2, \ldots, i_{m-1}) = [\alpha; \alpha + \beta),$$

tai

$$\begin{aligned} I(i_1, i_2, \ldots, i_{m-1}, 1) &= [\alpha; \alpha + p_1\beta), \\ I(i_1, i_2, \ldots, i_{m-1}, 2) &= [\alpha + p_1\beta; \alpha + (p_1 + p_2)\beta), \\ &\cdots\cdots\cdots \\ I(i_1, i_2, \ldots, i_{m-1}, r) &= [\alpha + (p_1 + \ldots + p_{r-1})\beta; 1). \end{aligned}$$

Štai ir visa esmė. Tačiau tikriausiai pravartu ją pabrėžti ir skaitiniu pavyzdžiu.

Pavyzdys. Tegu abėcėlės $\mathcal{A} = \{1, 2, 3, 4, 5\}$ simbolių tikimybės yra

$$p_1 = 0,2;\ p_2 = 0,25;\ p_3 = 0,15;\ p_4 = 0,1;\ p_5 = 0,3.$$

Tarkime, reikia koduoti žodį $\mathbf{x} = 213552$. Konstruojame intervalus:

$$\begin{aligned} I(2) &= [p_1; p_1 + p_2] = [0,2; 0,45) \\ I(21) &= [0,2; 0,25) \\ I(213) &= [0,2225; 0,23) \\ I(2135) &= [0,222775; 0,23) \\ I(21355) &= [0,229325; 0,23) \\ I(213552) &= [0,22946; 0,22962875) \end{aligned}$$

Taigi kaip žodžio $\mathbf{x} = 213552$ kodą galėtume siųsti, pavyzdžiui, dešimtainę trupmeną $\rho = 0,2295$. Tačiau nulio prieš kablelį siųsti nėra jokios prasmės, taigi galime siųsti tiesiog žodį $\mathbf{c} = 2295$. Vietoj šešių simbolių tereikia siųsti 4, savo duomenis suspaudėme net trečdaliu!

Žinoma, dažniausiai tenka koduoti dvejetainės abėcėlės žodžiais, tačiau tokią sąlygą nesunku patenkinti – tiesiog reikia parinkti intervale $I(\mathbf{x})$ skaičių, kurio dvejetainė išraiška būtų pati trumpiausia ir tiek.

Tačiau yra dar dvi aplinkybės, kurios verčia susimąstyti, ar toks kodavimo būdas tinkamas. Viena vertus, kad pasiųstume kodą, turime palaukti, kol baigsis kodavimo procesas ir rasime skaičių ρ. Kitas dalykas: kai koduosime ilgus žodžius – teks skaičiuoti su labai jau mažais skaičiais!

Atsakymas į pirmą pastabą paprastas: laukti iki galo nebūtina! Pažiūrėkime į pavyzdį. Juk sukonstravę intervalą $I(21) = [0,2; 0,25)$ matome, kad toliau visų intervalų rėžiai prasidės $0,2$, taigi skaitmenį 2 galime pasiųsti jau po antrojo žingsnio. Po trečiojo žingsnio taip pat galėtume nuspręsti, kad

antrasis simbolis vėl bus 2, taigi ir jį jau galima pasiųsti. Tačiau iš intervalo $I(2135) = [0,222775; 0,23)$ rėžių dar negalime spręsti, koks bus trečiasis simbolis, taigi reiktų palaukti. O štai po šeštojo žingsnio, jeigu jis dar nebūtų paskutinis, vis tiek žinotume, kad trečiasis simbolis tikrai bus 9. Taigi duomenų kodavimas ir kodo siuntimas gali vykti lygiagrečiai.

Tarkime, sukonstravę intervalą $I(21) = [0,2; 0,25)$, pasiuntėme pirmąjį kodo simbolį 2. Tačiau šis skaitmuo kaskart rašomas kitų intervalų išraiškose. Galime to išvengti štai taip: padauginę intervalo $I(21)$ rėžius iš 10 (kad išsiųstas skaitmuo atsidurtų prieš kablelį), sudarykime naują intervalą $I^*(21) = [0; 0,5$ vien tik iš trupmeninių dalių ir naudokime jį sekančiam intervalui konstruoti vietoj $I(21)$. Šitaip ne tik išvengsime jau išsiųsto simbolio kartojimo, bet ir skaičiavimų su labai mažais skaičiais. Toks kodavimas vadinamas aritmetiniu kodavimu su mastelio keitimu.

Panagrinėkime pavyzdį, kiek pailginę jau koduotą žodį. Tegu dabar $\mathbf{x} =$ 21355241. Mastelio keitimą atliksime tik tada, kai ir apatinis ir viršutinis intervalo rėžiai prasidės tuo pačiu skaitmeniu (arba keliais vienodais).

Simbolis	I	I^*	Kodas
2	[0,2;0,45)		
1	[0,2;0,25)	[0;0,5)	2
3	[0,425; 0,5)		
5	[0,4775; 0,5)		
5	[0,49325; 0,5)		
2	[0,4946; 0,4962875)	[0,46; 0,62875)	49
4	[0,56125; 0,578125)	[0,6125; 0,78125)	5
1	[0,6125; 0,64625]	[0,125; 0,4625)	62

Paskutiniame žingsnyje parinkome skaičių $0,62$ iš sukonstruotojo intervalo. Pakeisto mastelio intervalas lieka nepanaudotas. Taigi aštuonių simbolių žodžiui koduoti pakako šešių. Duomenų apimtį sumažinome 25%.

Dekoduotojas turi žinoti, kokio ilgio žodis buvo suspaustas ir ar naudotas mastelio keitimo veiksmas.

Tokia yra aritmetinio kodavimo esmė. Žinoma, taikydami tokį kodavimą tikroviškais atvejais, susidurtume su įvairiais realizavimo sunkumais... Tektų pavartyti solidžius duomenų spaudimą aprašančius veikalus.

3.2. Kintamų kodų metodas

Skyrelis apie kodų džiazą: kodai tarsi džiazo motyvai kuriami ekspromtu, atsižvelgiant į koduojamų duomenų srautą.

Tiek Huffmano, tiek aritmetiniai kodai naudoja šaltinio simbolių tikimybes.

Jeigu šaltinio savybės pasikeistų arba šaltinis generuotų kokį nors „netipišką“ srautą, Huffmano kodas nebesuspaustų mūsų duomenų tiek, kiek įmanoma.

Vienas iš būdų atsižvelgti į duomenų, kuriuos reikia suspausti, savybes – prieš spaudžiant nustatyti kaip dažnai pasitaiko jame abėcėlės simboliai.

Gavę ilgą abėcėlės $\mathcal{A} = \{a_1, a_2, \ldots, a_r\}$ simbolių srautą $x_1 x_2 \ldots x_n$, galime suskaičiuoti, kiek kartų pasikartoja abėcėlės simboliai, t. y. suskaičiuoti dažnius

$$n_m = |\{x_i : i \leq n, x_i = a_m\}|, \quad m = 1, 2, \ldots, r,$$

ir manyti, kad šį srautą generavo Bernulio šaltinis su simbolių tikimybėmis

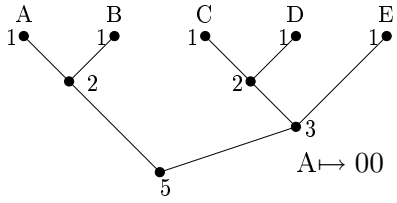
$$p_1 = \frac{n_1}{n},\ p_2 = \frac{n_2}{n},\ \ldots,\ p_r = \frac{n_r}{n}.$$

Tada, naudodamiesi šiomis tikimybėmis, galėtume koduoti duomenis Huffmano arba aritmetiniu kodu, tikėdamiesi, kad šitaip jie bus gerai suspausti. Darbo koduojant šiuo būdu padaugėja – juk reikia iš pradžių sudaryti simbolių dažnių lentelę! Kita aplinkybė – kartu su suspaustais duomenimis gavėjui turime perduoti ir Huffmano kodo medį arba simbolių dažnių lentelę.

Aptarsime dar vieną būdą naudotis Huffmano kodais kartu atsižvelgiant ir į koduojamų duomenų savybes. Šis metodas vadinamas adaptyviu Huffmano kodu, t. y. prie koduojamų duomenų „prisitaikančiu“ kodu. Jo esmė tokia: koduojant tikslinami simbolių pasitaikymo duomenų sraute dažniai, kartu keičiant ir kodavimui naudojamą Huffmano kodą. Taigi Huffmano kodas kuriamas kodavimo metu; dekoduojant kartojamas tas pats kodo kūrimo procesas.

Kodavimą Huffmano kodu, sudarytu pagal šaltinio tikimybes, galime įsivaizduoti kaip „leidimąsi“ nuo lapų prie šaknies: vienintelis kelias nuo simbolio lapo prie šaknies nurodo tam simboliui priskirtą žodį. Dekodavimas savo ruožtu – „kopimas“ nuo šaknies link lapų. Susitarsime naudotis Huffmano kodo medžiais, kurių viršūnėms yra priskirti skaičiai: lapams – atitinkamų simbolių dažniai, o vidinėms viršūnėms – su jomis sujungtų aukštesniojo lygio viršūnių skaičių sumos. Kadangi pagal simbolių tikimybes (dažnius) galima sukonstruoti ne vieną Huffmano kodą, reikia susitarti, kokį kodą rinksimės. Susitarimas toks:

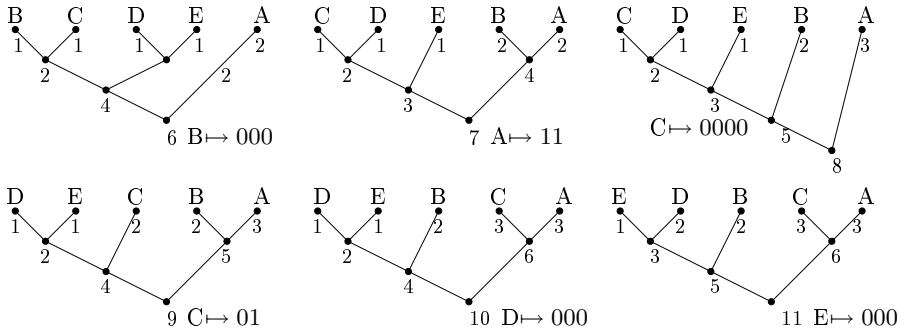
- abėcėlės simbolius visada išdėstysime dažnių didėjimo tvarka iš kairės į dešinę;
- konstruojant kodo medį, jungiant mažiausius dažnius turinčių viršūnių porą, visada bus jungiami patys „kairiausieji“ elementai; sujungus viršūnių porą, gautoji viršūnė bus vaizduojama vienu lygiu žemiau už žemesniojo lygio viršūnę.
- į kairę vedančioms briaunoms visada priskirsime 0, į dešinę – 1.



Brėžinyje parodytas Huffmano kodo medis tekstui, kuriame raidės A,B, C, D ir E pasirodė atitinkamai po vieną kartą.

Tarkime, koduojamo duomenų srauto abėcėlė yra $\mathcal{A} = \{a_1, a_2, \ldots, a_r\}$, o kodo abėcėlė dvejetainė. Kadangi šaltinio tikimybių neturime, o paties duomenų srauto analizuoti nenorime (arba ketiname pradėti koduoti dar neturėdami viso srauto), tai reikia susitarti dėl pradinio kodo. Viena iš tokio pradinio susitarimo galimybių – naudoti Huffmano kodą, kuris atitinka pagal mūsų taisykles sukonstruotą medį su visų simbolių dažniais, lygiais 1. Gavę pirmąjį simbolį, jį koduojame, padidiname šio simbolio dažnį 1, perrikiuojame simbolius, perkeldami užkoduotąjį simbolį (jei reikia) į dešinę per mažiausią reikalingą pozicijų kiekį, kad simbolių dažniai vėl sudarytų nemažėjančią seką. Tada perbraižome kodo medį (taigi sudarome naują Huffmano kodą) ir laukiame antrojo simbolio. Užkodavę antrąjį simbolį, vėl atliekame analogiškus medžio pertvarkymo veiksmus.

Panagrinėkime pavyzdį su $\mathcal{A} = \{A, B, C, D, E\}$, taigi pradinis Huffmano kodas atitinka brėžinyje pavaizduotą medį. Tegu srautas, kurį reikia koduoti, yra toks: ABACCDE. Brėžinyje pavaizduota, kaip keitėsi Huffmano kodo medis.

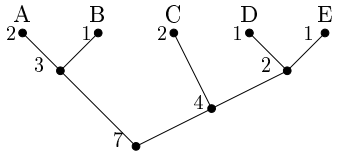


Brėžinyje parodyta, kaip keitėsi kodo medis, koduojant žodį ABACCDE. Pirmajai raidei koduoti buvo panaudotas medis, atitinkantis vienodus abėcėlės simbolių pasirodymo dažnius.

A	B	A	C	C	D	E
00	000	11	0000	01	000	000

Žodžio ABACCDE kodavimas naudojant kintamų Huffmano kodų metodą

Taigi tuo pačiu nulių trejetu kodavome net tris skirtingus abėcėlės simbolius. Palyginkime gautą kodą su tuo, kurį būtume sudarę pagal simbolių dažnius visame bloke. Koduojamame žodyje simboliai pasitaikė atitinkamai po 2,1,2,1,1 kartą. Kodo medis pavaizduotas brėžinyje.



Brėžinyje parodytas Huffmano kodo medis, sudarytas pasinaudojant simbolių dažniais žodyje ABACCDE.

A	B	A	C	C	D	E
00	01	00	10	10	110	111

Žodžio ABACCDE kodavimas naudojant simbolių dažnius

Pagal šį medį sudarytas žodžio ABACCDE kodas turi trimis dvejetainiais simboliais mažiau. Tačiau juk reiktų siųsti ir visų simbolių dažnius! Taigi šiuo paprastu atveju adaptyvus Huffmano kodas yra gerokai taupesnis (koduotojui ir dekoduotojui „užkraunamo“ darbo sąskaita!).

Taigi prie duomenų srauto prisitaikančio kodo idėja yra paprasta. Ją galima taikyti ir aritmetiniam kodavimui. Žinoma, geros idėjos diegimas visada susijęs su įvairiais techniniais sunkumais. Pavyzdžiui, kad ir simbolių dažnių kaupimas: skaičiai greitai gali labai padidėti; kita vertus – kaip kuo efektyviau pertvarkyti kodų medžius? Pavartykite adaptyviam kodavimui skirtas publikacijas, jeigu įdomu žinoti.

3.3. Blokų ilgių kodavimas

Dvejetainės abėcėlės simbolių srautą galime koduoti natūrinių skaičių seka. Ar galime ką nors laimėti, jeigu tuos skaičius vėl koduosime dvejetainės abėcėlės žodžiais?

Dvejetainė abėcėlė $\mathcal{B} = \{0, 1\}$ yra pati svarbiausia. Nulių ir vienetų srautą visada galime interpretuoti kaip vienodų simbolių blokų srautą ir keisti jį natūrinių skaičių seka, o pastaruosius vėl koduoti dvejetainės abėcėlės žodžiais. Pavyzdžiui,

$$00111011111000011111 \to 2; 3; 1; 6; 4; 5 \to 10; 11; 1; 110; 100; 101.$$

Dvidešimt septyniems pradinio srauto bitams koduoti panaudojome septyniolika bitų. Tai, žinoma, laimėjimas, tačiau toks kodavimo būdas netinkamas praktikai. Juk dar reikia nurodyti, kokių simbolių – nulių ar vienetų – bloku prasideda srautas ir kaip atskirti dvejetainius natūrinių skaičių kodus. Srauto pradžios klausimą galime išspręsti labai paprastai. Visada galima manyti, kad srautas prasideda nuliais. Jeigu taip nėra, galima tiesiog pridėti vieną nulį. Skaičių kodams atskirti galima įvesti specialų atskyrimo simbolį arba skaičiams koduoti galime naudoti kokį nors p-kodą. Jeigu žinoma vienodų simbolių blokų ilgių statistika, galima ja pasinaudoti ir sukonstruoti geresnį kodą, t. y. leidžiantį sumažinti kodavimo sąnaudas.

Taigi norint pasinaudoti blokų ilgių kodavimo idėja, reikia nuspręsti, kaip koduoti natūrinių skaičių sekas. Panagrinėsime vieną iš daugelio galimų būdų – Golombo kodus.

Parinkę natūralųjį skaičių $m \geq 1$, sukonstruosime Golombo kodą G_m, t. y. injektyvų atvaizdį

$$\{1, 2, 3, \ldots\} \to \{0, 1\}^*.$$

Tegu n yra natūralusis skaičius, kurio kodą reikia apibrėžti. Padaliję n iš m su liekana, gausime

$$n = q \cdot m + r, \quad 0 \leq r < m.$$

Skaičiaus n kodą sudarysime sujungę dalmens q ir liekanos r kodus. Dalmenį koduosime pagal tokią – pačią paprasčiausią – taisyklę:

$$0 \mapsto 0,\ 1 \mapsto 10,\ 2 \mapsto 110,\ 3 \mapsto 1110, \ldots$$

Tiesiog pirmykštis skaičių rašymo būdas – kiek skaičių, tiek brūkšnelių! Liekanos r kodavimas priklauso nuo parinktojo m. Tarkime, $m = 2^s$. Liekanai r užrašyti naudosime jos dvejetainę išraišką, sudarydami ją iš s simbolių. Bet kokiai liekanai užrašyti tokio ilgio dvejetainės abėcėlės žodžio užteks ir visų jų prireiks.

$n =$	1	2	3	4	5	6	7	8	9	...
Kodas	0001	0010	0011	0100	0101	0110	0111	10000	10001	...

Golombo kodas G_8

Kaip dekoduoti Golombo kodą su $m = 2^s$, visiškai aišku: suskaičiavę, kiek yra vienetų iki pirmojo nulio, rasime dalmenį q, o atmetę nulį, iš sekančių s bitų, surasime liekaną.

O dabar tegu m bet koks. Apibrėžkime tokius skaičius:

$$s = [\log_2 m] + 1, \quad m_* = 2^s - m,$$

čia $[t]$ žymi skaičiaus t sveikąją dalį. Visoms liekanoms užrašyti pakaktų s ilgio dvejetainių žodžių, tačiau ne visi jie būtų panaudoti. Todėl galime kiek

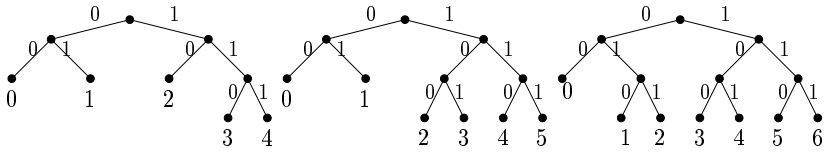
pagerinti liekanų užrašymo būdą: mažoms liekanoms, t. y. tokioms, kurios tenkina nelygybę $r < m_*$, koduoti naudosime $s-1$ ilgio dvejetainius žodžius, o likusioms – s ilgio žodžius:

$$m-1 \mapsto 111\ldots11,\ m-2 \mapsto 111\ldots10,\ m-3 \mapsto 111\ldots00,\ldots$$

Kodėl mažoms liekanoms koduoti užtenka $s-1$ ilgio žodžių? Įsitikinkime:

$$\begin{aligned} s &= \log_2 m + \theta,\ 0 \le \theta < 1,\ m = 2^{s-\theta}, \\ m^* &= 2^s - m = 2^s - 2^{s-\theta} = 2^{s-1}(2 - 2^{1-\theta} < 2^{s-1}. \end{aligned}$$

Taigi visų liekanų $r < m^*$ dvejetainėms išraiškoms $s-1$ simbolių užtenka. Įrodysime, kad šitaip gaunamas p kodas. Bet iš pradžių panagrinėkime atvejus su $m = 5, 6, 7$.



Liekanų kodavimas Golombo koduose G_5, G_6, G_7. Visais atvejais $s = 3$, o m_* reikšmės atitinkamai lygios $3, 2, 1$. Liekanos kodas yra dvejetainis žodis, kurį pradedame skaityti nuo medžio šaknies.

Taigi liekanos, kurios yra mažesnės už $2^s - m$, koduojamos $s-1$ ilgio žodžiais, o visos kitos – s ilgio. Golombo kodai turi tokią dekodavimui svarbią savybę: jeigu liekanos s bitų ilgio žodyje atmesime paskutinįjį bitą ir likusį $s-1$ bitų žodį interpretuosime kaip skaičiaus dvejetainę išraišką, tai tas skaičius bus ne mažesnis už $2^s - m$. Šia savybe dekoduojant galime pasinaudoti tokiu būdu: nustatę dalmenį, perskaitome $s-1$ bito žodį kaip natūralųjį skaičių. Jeigu šis skaičius yra mažesnis už $2^s - m$, tai jis ir yra liekana. Jeigu didesnis – liekanos kodui gauti reikia prišlieti dar vieną bitą. Įsitikinsime, kad šią savybę turi visi Golombo kodai G_m, kai m nėra dvejeto laipsnis.

Didžiausia liekana kode G_m, kuri koduojama s ilgio žodžiu, yra $m-1$. Jos kodą sudaro žodis iš s vienetų: $11\ldots1$. Jeigu šį žodį perskaitysime kaip natūralųjį skaičių, tas skaičius bus lygus $2^s - 1$. Iš viso s bitų žodžiais koduojama $m-1-(2^s-m)+1 = 2m-2^s$ liekanų. Išdėstę jas mažėjimo tvarka ir perskaitę kodus kaip natūraliuosius skaičius, gautume seką: $2^s-1, 2^s-2, \ldots$ Pats mažiausias šioje sekoje būtų $2^s - (2m - 2^s) = 2(2^s - m)$. Bet kurį šios sekos skaičių galime užrašyti taip: $2x + i$, čia $i = 0$ arba $i = 1$, o x yra natūrinis skaičius, kurį gautume iš pirmųjų $s-1$ dvejetainės išraiškos bitų. Kadangi $2x + i \ge 2(2^s - m)$, tai $x \ge 2^s - m - i/2$ ir $x \ge 2^s - m$, nes x yra natūrinis skaičius.

Golombo kodų yra be galo daug – yra iš ko pasirinkti. Su kokiu m Golombo kodas yra geriausias? Tai, žinoma, priklauso nuo srauto savybių.

Jeigu parinksime mažą m, bet dažnai teks koduoti didelius skaičius, tai daug simbolių bus panaudojama dalmeniui koduoti. Jeigu parinksime didelį m, bet dažnai teks koduoti mažus skaičius, liekanų kodavimas bus neefektyvus.

Yra nustatyta, koks Golombo kodas yra geriausias vienu specialiu atveju. Tarkime, pradinį bitų srautą sukūrė Bernulio šaltinis, kuris perduoda nulį su tikimybe p. Tada blokų ilgiams koduoti geriausias tas Golombo kodas G_m, kurio parametras m tenkina sąlygą

$$p^m + p^{m+1} \leq 1 < p^m + p^{m-1}.$$

Iš šių nelygybių gauname, kad su geriausia m reikšme teisinga apytikslė lygybė

$$p^m \approx \frac{1}{2}.$$

3.4. Žodyno metodas

Jonas sako: „Dvyliktas“. Visi užstalės draugai nusijuokia, patyli. Petras sako: „Dvidešimt antras“. Vėl panaši draugų reakcija. Ką gi veikia šie žmonės? Jie pasakoja anekdotus, suspaustus naudojant žodyno metodą!

Koduoti duomenų srautą žodyno metodu reiškia keisti žodžius ir net frazes jų eilės numeriais žodyne. Tačiau tai tik labai „žalia“ idėja... Panagrinėkime du Lempelio ir Zivo pasiūlytus duomenų spaudimo algoritmus, kuriuos jie išdėstė 1977 ir 1978 metais. Bendras abiejų metodų bruožas – prieš pradedant koduoti, žodyno nėra, jis kuriamas palaipsniui, naudojantis koduojamais duomenimis.

Pirmasis metodas paprastai vadinamas LZ77 metodu. Jame naudojami slenkantys langai. Langą sudaro dvi fiksuoto ilgio dalys: kairioji dalis, kurioje telpa, tarkime, m simbolių, ir dešinioji, kurioje telpa n simbolių. Kairiąją langa pusę vadinsime žodyno langu, o dešiniąją – teksto langu. Šis langas slenka išilgai koduojamo simbolio srauto.

Tegu, pavyzdžiui, $m = 8$ ir $n = 4$. Panagrinėkime brėžinyje pavaizduotą padėtį.

VASARIS VASARA PAVASARIS

Teksto lange atsidūrė žodžio fragmentas VASA, o žodyno lange – VASARIS ir žodžius skiriantis tarpas. Dabar žodyno lange ieškome ilgiausio fragmento, kuris būtų teksto lango priešdėlis, randame – VASA. Savo radinį nurodome dviem skaičiais: rasto priešdėlio pradžios atstumu nuo teksto lango ir priešdėlio ilgiu: šie skaičiai yra 8 ir 4. Juos papildome pirmuoju dar nekoduotu simboliu, t. y. raide R. Taigi sudaromas srauto kodas papildomas trejetu $\langle 8; 4; R\rangle$, o langas pastumiamas taip, kad dešiniojo lango pradžioje atsidurtų pirmas dar nekoduotas simbolis:

VASARIS VASARA PAVASARIS

Visa teksto VASARIS VASARA PAVASARIS eiga parodyta lentelėje

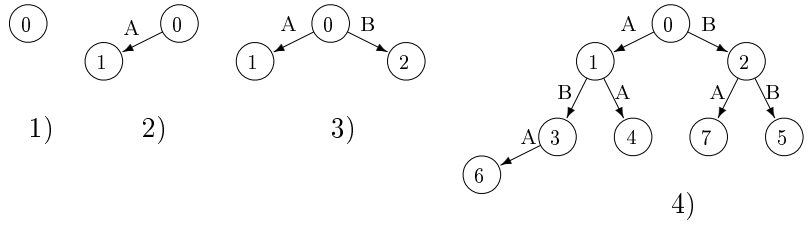
	Kodas
VASA RIS VASARA PAVASARIS	0;0;V;
V ASAR IS VASARA PAVASARIS	0;0;A;
VA SARI S VASARA PAVASARIS	0;0;S;
VAS ARIS VASARA PAVASARIS	2;1;R;
VASAR IS V ASARA PAVASARIS	0;0;I;
VASARI S VA SARA PAVASARIS	4;1; :
VASARIS VASA RA PAVASARIS	8;4;R;
VASAR IS VASAR A PA VASARIS	2;1; ;
VASARIS VASARA PAVA SARIS	0;0;P;
VASARIS VASARA P AVAS ARIS	3;1;V;
VASARIS VA SARA PAV ASAR IS	2;1;S;
VASARIS VASA RA PAVAS ARIS	2;1;R;
VASARIS VASARA PAVASAR IS	0;0;I;
VASARIS VASARA PAVASARI S	4;1;

Šis pavyzdys parodo, kad spaudžiant duomenis, kartais duomenų apimtis netgi padidėja. Tačiau kartu rodo ir priežastį – mūsų žodyno langas pernelyg mažas. Padidinę jį, gautume geresnį rezultatą. Kita priežastis – žodynas visą laiką kinta. Dėl šios priežasties negalėjome, pavyzdžiui, pasinaudoti tuo, kad fragmentas VASAR pasikartoja duomenų sraute net tris kartus.

Kitas Lempelio ir Zivo pasiūlytas algoritmas LZ78 naudoja žodyną, kuris nuolat didėja, nes yra pildomas vis naujais tekste pasitaikiusiais žodžiais. Panagrinėsime kodavimo idėją, konstruodami simbolių srauto

ABABAABBABABAABAB

kodą. Koduojant simboliai skaitomi nuosekliai vienas po kito, kartu sudarant žodyno medį. Iš pradžių žodynas tuščias, jį atitinka šaknies viršūnė, kuriai suteiktas numeris lygus nuliui, žr. 1 brėžinį. Perskaitę simbolį A, „atverčiame“ žodyną, tikriname, ar jame yra frazių, prasidedančių šia raide. Nėra. Tada į sudaromo kodo seką užrašome porą $\langle 0; A\rangle$, o žodyną papildome žodžiu „A“, kuriam suteikiame numerį 1, žr. 2 brėžinį. Perskaitę antrąjį simbolį, vėl į kodo seką užrašome $\langle 0; B\rangle$, o žodyną papildome žodžiu „B“, jo numeris 2. Trečioji koduojamo žodžio raidė yra „A“, ji jau yra žodyne, taigi į kodo seką užrašome jos numerį su sekančia raide, o žodyną papildome žodžiu „AB“, kuriam suteikiame numerį 3. Taigi auga kodo simbolių eilė, o kartu ir žodynas.



Žodyno, sudaryto koduojant seką ABABAABBABABAABAB LZ78 kodu, medis

Surašykime kodavimo žingsnius lentele, nurodydami, kaip papildoma kodo seka ir žodynas.

Kodavimo žingsnis	Kodo papildymas	Žodyno papildymas
1	$0; A$	A
2	$0; B$	B
3	$1; B$	AB
4	$1; A$	AA
5	$2; B$	BB
6	$3; A$	ABA
7	$2; A$	BA
8	$6; B$	$ABAB$

Kodavimo LZ78 eiga. Į žodyną naujai įtraukiamo žodžio numeris sutampa su žingsnio numeriu.

Duomenų spaudimą nagrinėjančioje literatūroje galima surasti įvairių šių dviejų kodų modifikacijų. Pavyzdžiui, galima pradėti ne nuo tuščio žodyno, jame jau gali būti visi iš vieno simbolio sudaryti žodžiai (visos baitų reikšmės). Kita vertus, taikant LZ78 kodą praktiškai, reikia nuspręsti, ką daryti, kai žodyno apimtis labai padidėja.

Dekoduojant LZ78 kodą, atkuriamas tas pats žodynas, koks buvo panaudotas koduojant.

3.5. „Kelk į priekį“ ir kitos geros idėjos

Du teksto kodavimo metodai: išradingi, paprasti, efektyvūs...

Simbolius keisti skaičiais – informatikų ir matematikų įprotis. Tarkime, šaltinio abėcėlė yra $\mathcal{A} = \{a, b, c, d\}$. Šios abėcėlės simbolių srautą galime

pakeisti natūrinių skaičių seka – tiesiog keisdami raides jų eilės numeriais abėcėlėje. Pavyzdžiui:

$$bcdddaa\ldots \mapsto 1;2;3;3;3;0;0\ldots$$

Tačiau šitaip duomenų nesuspausime, nebent skaičiams koduoti dvejetainiais simboliais turėtume geresnį būdą negu raidėms. Bet nei vienos minties neverta atmesti neįsitikinus, kad jos negalima patobulinti. Pagerinti raidžių keitimo skaičiais idėją galime šitaip: užrašę raidės kodą (eilės numerį) pakeiskime abėcėlę, perkeldami užkoduotą abėcėlės raidę į priekį. Šį kodavimo būdą pavadinkime „kelk į priekį“ metodu.

Abėcėlė	Raidė	Kodas
a, b, c, d	b	1
b, a, c, d	c	2
c, b, a, d	d	3
d, c, b, a	d	0
d, c, b, a	d	0
d, c, b, a	a	3
a, d, c, b	a	0

Kodavimas „kelk į priekį“ metodu. Palyginę su anksčiau gauta skaičių seka, matome, kad nedaug telaimėjome – trejetų skaičius sumažėjo vienetu, o nulių – vienetu padidėjo. Taigi laimėjimas vis dėlto yra!

Kokios naudos galima tikėtis koduojant šiuo metodu? Dažniau pasitaikantys simboliai bus keičiami mažesniais skaičiais. Pavyzdžiui, jeigu vienoje srauto dalyje dažnai pasitaiko aibės X raidės, tai jos bus keičiamos nedideliais skaičiais; jeigu kitoje vietoje dažnesnės aibės Y raidės, tai būtent jos bus keičiamos mažesniais skaičiais. Taigi galima tikėtis, kad gautoje skaičių sekoje vyraus mažesni skaičiai. Mažesniems skaičiams koduoti reikia mažiau bitų – prisiminkime, pavyzdžiui, Golombo kodus.

Panagrinėsime dar vieną kodavimo būdą, kurį galima naudoti kartu su „kelk į priekį“ idėja – Burrowso ir Wheelerio metodą. Lengviausia jį suprasti nagrinėjant pavyzdį. Tarkime, reikia koduoti žodį „pasaka“. Pirmiausia, atlikime visus ciklinius šio žodžio postūmius, o paskui visus šešis gautus žodžius sutvarkykime leksikografine tvarka:

Žodžiai po postūmių	Sutvarkyti žodžiai
pasaka	**a** *kapa* **s**
asakap	**a** *pasa* **k**
sakapa	**a** *saka* **p**
akapas	**k** *apas* **a**
kapasa	**p** *asak* **a**
apasak	**s** *akap* **a**

Žodžio „pasaka“ kodą sudarysime užrašę paskutinį sutvarkytos lentelės stulpelį ir nurodę, kurioje jos eilutėje yra pradinis žodis, taigi

$$pasaka \mapsto skpaaa; 5.$$

Kyla teisėta abejonė: kokia tokio kodavimo nauda? Juk kodas netgi ilgesnis, nes nurodomas papildomas simbolis – skaičius. Kai koduojami ilgesni žodžiai, to pailgėjimo galime ir nelaikyti svarbiu – jeigu nauda atsiranda kitur.

Galbūt ir nelabai įtikinantis argumentas, kad šį bei tą laimime, toks: palyginkime žodžių „pasaka“ ir „skpaaa“ kodus, gautus „kelk į priekį“ metodu, kai abėcėlė yra $\mathcal{A} = \{a, k, p, s\}$. Štai tie kodai:

$$pasaka \mapsto 2; 1; 3; 1; 3; 1 \quad skpaaa \mapsto 3; 1; 2; 3; 0; 0.$$

Esmė tokia: Burrowso-Wheelerio metodu atitinkamai perstatant raides iš pradinio žodžio sudaromas naujas, o šiame žodyje vienodos raidės „linkusios“ grupuotis; šis polinkis ir suteikia galimybę koduoti efektyviau.

Tačiau ar iš kodo vienareikšmiškai galime atkurti koduotą srautą? Čia ir yra visas šios idėjos žavesys. Galima. Panagrinėkime kaip.

Pažvelkime į sutvarkytos lentelės pirmąjį ir paskutinįjį (pajuodintus) stulpelius. Jie sudaryti iš tų pačių raidžių, tik pirmasis sutvarkytas didėjimo tvarka. Todėl žinodami paskutinįjį, galime jį sutvarkyti ir gauti pirmąjį. Taigi iš Burrowso- Wheelerio metodu gauto kodo (paskutiniojo sutvarkytos lentelės stulpelio) galime atkurti pirmąjį. O ką reiškia, pavyzdžiui, kad antrosios eilutės paskutiniajame stulpelyje yra „k“, o pirmajame – „a“? Tai reiškia, kad pradiniame žodyje po „k“ seka „a“! Tačiau juk žinome, nuo kurios eilutės pradėti – nuo penktosios. Taigi paskutinė žodžio raidė yra „a“, o pirmoji – „p“. Kas seka po „p“? Ieškome paskutiniame stulpelyje „p“ ir randame, kad po jos eina „a“. Ieškome paskutiniajame stulpelyje „a“ ... Bet jų čia kelios! Ką gi daryti? Sugrįžkime prie raidės „a“, kurią radome pirmajame stulpelyje – ji yra iš trijų vienodų raidžių trečioji. Todėl ir paskutiniajame stulpelyje imkime trečiąją iš eilės raidę „a“ – ji yra šeštojoje eilutėje. Nesijaudinkite,

jei neaišku, kodėl taip reikia daryti – netrukus pabandysiu paaiškinti. Taigi randame, kad prieš „a“ turi būti „s“ ir taip toliau...

Viskas turėtų būti suprantama, išskyrus atvejį, kai paskutiniajame stulpelyje reikia rinktis vieną iš kelių vienodų raidžių. Panagrinėsime, kaip išsidėsto vienodos raidės pirmajame ir paskutiniajame stulpeliuose imdami tą patį pavyzdį. Kad būtų galima sekti, kur atsiduria vienodos raidės „a“, žymėsime jas indeksais. Taigi kiek pakeiskime savo lentelę:

Žodžiai po postūmių	Sutvarkyti žodžiai
$pa_1sa_2ka_3$	$\mathbf{a}_2\ ka_3pa_1\ \mathbf{s}$
$a_1sa_2ka_3p$	$\mathbf{a}_3\ pa_1sa_2\ \mathbf{k}$
$sa_2ka_3pa_1$	$\mathbf{a}_1\ sa_2ka_3\ \mathbf{p}$
$a_2ka_3pa_1s$	$\mathbf{k}\ a_3pa_1s\ \mathbf{a}_2$
$ka_3pa_1sa_2$	$\mathbf{p}\ a_1sa_2k\ \mathbf{a}_3$
$a_3pa_1sa_2k$	$\mathbf{s}\ a_2ka_3p\ \mathbf{a}_1$

Dabar jau gerai matyti – pirmajame ir paskutiniajame stulpeliuose vienodos raidės išsidėsto ta pačia tvarka. Kodėl taip atsitinka? Tai nesunku pačiam suvokti, bet sunku kitam paaiškinti. Prisipažinsiu, nesupratau nei vieno kitose knygose išdėstyto aiškinimo, bet užsirašiau šį pavyzdį, pagalvojau apie sutvarkymą ir iš karto viską kuo geriausiai supratau. Siūlau ir jums tą patį būdą!

4 Kanalai ir kodai

Ar su netobulais įrankiais galima sukurti tobulą kūrinį? Jei aplankysite Rokiškyje medžio drožėjo Liongino Šepkos darbų parodą, pamatysite ir nuostabias jo skulptūras, ir varganus peiliukus, kuriais naudojosi meistras. Netobuli įrankiai ne kliūtis meistrui, bet iššūkis.

Šio skyriaus paskirtis – įtikinti skaitytoją, kad nepatikimais informacijos perdavimo kanalais yra įmanomas patikimas ryšys!

4.1. Perdavimo kanalai

Jeigu gavėjas gauna ne visada tai, ką siuntė siuntėjas, vadinasi, perdavimo kanalas yra nepatikimas. Nepatikimumo, kaip ir melo, rūšių yra daug. Kad galėtume matematiškai nagrinėti informacijos perdavimą nepatikimais kanalais, sukurkime matematinį tokio kanalo modelį.

Mokslo apie informacijos perdavimą srityje vienas iš svarbiausių faktų yra Shannono įrodytas teiginys: nors ryšio kanalai nėra patikimi, jais patikimas ryšys yra įmanomas.

Kaip išdėstyti šį teiginį matematiškai?

Pradėkime nuo perdavimo kanalų. Jeigu jau yra perdavimo kanalas, tai turi būti ir siuntėjai, ir gavėjai. Paprasčiausiu atveju ir siuntėjas vienas, ir gavėjas vienas. Siuntėjas perduoda tam tikrus duomenis, gavėjas gauna. Jeigu perduodamas tolydžiai kintantis signalas, kurį galime įsivaizduoti laiko funkcija, sakome, kad kanalas yra analoginis. Gavėjas irgi gauna funkciją, galbūt jau šiek tiek kitokią.

Jeigu siuntėjas siunčia kokios nors abėcėlės žodį, t. y. keletą simbolių, kurie seka vienas po kito, sakome, kad kanalas yra diskretus. Gavėjas taip pat gauna simbolių seką, galbūt jau nebe tokios pat kaip siuntėjo abėcėlės.

Taigi kanalą galime suvokti kaip dviejų informacijos šaltinių: siuntėjo ir gavėjo, porą. Siuntėjo abėcėlę žymėsime $\mathcal{A}$, o atsitiktinius dydžius, kurių reikšmės - siuntėjo perduodami simboliai, $U_1, U_2, \ldots$ Atitinkamai gavėjo abėcėlę žymėsime $\mathcal{B}$, o atsitiktinius dydžius, kurių reikšmės – gavėjo gauti simboliai, $V_1, V_2, \ldots$ Jeigu gavėjo šaltinis yra tiksli siuntėjo šaltinio kopija, vadinasi, turime idealų perdavimo kanalą ir jokia teorija mums nereikalinga. Tačiau tikrovėje idealių kanalų nėra. Taigi gavėjo gaunami duomenys gali skirtis nuo siųstųjų. Tokiu atveju sakome, kad perduota iškraipyta informacija, o visas aplinkybes, dėl kurių tai įvyko, vadinsime vienu vieninteliu žodžiu – triukšmas.

Taigi informacijos iškraipymų perduodant kanalu priežastis – triukšmas. Priemonės, kurių galime griebtis norėdami padidinti perdavimo patikimumą, priklauso, žinoma, nuo triukšmo pobūdžio. Apskritai triukšmas tėra „patogus“ žodis, kuris konkrečiais atvejais gali reikšti skirtingus dalykus. Pavyzdžiui, siunčiant žodį kanalu, triukšmas gali įterpti papildomus simbolius arba juos pašalinti... Tada gavėjas gaus nebe tokio paties ilgio žodį, kokį siuntė siuntėjas.

Triukšmas gali pakeisti vienus siunčiamus simbolius kitais, tada sakome, kad kanalas iškreipia informaciją. Jeigu kanalo triukšmas gali pašalinti simbolius, tačiau gavėjas žino, kurie žodžio simboliai buvo pašalinti, sakome, kad kanalas trina informaciją.

Sunku aprėpti visą įvairovę iš karto. Todėl apsiribosime tik tokiais kanalais, kurie kraipo simbolius arba juos trina. Tokių kanalų atveju viena aplinkybė yra garantuota: jeigu siuntėjas pasiuntė n ilgio žodį, tai tokio pat ilgio žodį gaus ir gavėjas (ištrintus simbolius jis gali pakeisti, pavyzdžiui, klaustukais).

Taigi tegu $\mathcal{A} = \{a_1, a_2, \ldots, a_q\}$ yra siuntėjo, o $\mathcal{B} = \{b_1, b_2, \ldots, b_s\}$ – gavėjo abėcėlė. Siuntėjo siunčiami žodžiai yra atsitiktinio vektoriaus $U^{(n)}$ reikšmės, o gavėjo gaunami žodžiai – atsitiktinio vektoriaus $V^{(n)}$ reikšmės.

Statistines kanalo triukšmo savybes nusako tikimybės

$$P(V^{(n)} = \mathbf{v} | U^{(n)} = \mathbf{u}), \quad \mathbf{u} \in \mathcal{A}^n, \ \mathbf{v} \in \mathcal{B}^n.$$

t. y. tikimybės gauti $\mathbf{v}$, jei buvo pasiųsta $\mathbf{u}$. Kaip ir informacijos šaltiniai, perdavimo kanalai taip pat gali turėti atmintį: m-asis gavėjo simbolis gali priklausyti ne tik nuo m-ojo siųsto simbolio, bet ir nuo to, kas buvo siųsta ir gauta anksčiau. Paprasčiausia kanalų rūšis – praeities neatsimenantys kanalai. Galime įsivaizduoti, kad, siųsdami simbolį a tokiu kanalu, atliekame bandymą, kurio baigtys – abėcėlės $\mathcal{B}$ simboliai, o jų tikimybės priklauso tik nuo a ir nuo kanalo triukšmo savybių. Tikimybę, kad, siunčiant simbolį a, bus gautas simbolis b žymėsime $p(b|a)$. Tarsime, kad šios tikimybės nepriklauso nuo to, kuris iš eilės simbolis yra siunčiamas ir gaunamas, t. y. nagrinėsime stacionarius kanalus. Stacionarių kanalų „triukšmo savybės“ nesikeičia laikui bėgant.

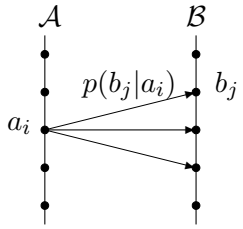
18 apibrėžimas. *Perdavimo kanalą vadinsime kanalu be atminties, jeigu visiems žodžiams* $\mathbf{u} = u_1 u_2 \ldots u_n \in \mathcal{A}^n, \mathbf{v} = v_1 v_2 \ldots v_n \in \mathcal{B}^n$ *teisinga lygybė*

$$P(V^{(n)} = \mathbf{v} | U^{(n)} = \mathbf{u}) = p(v_1|u_1) p(v_2|u_2) \ldots p(v_n|u_n).$$

Kanalo triukšmo savybes, t. y. polinkį kraipyti siunčiamus simbolius, galime nusakyti kanalo tikimybių $p(b_j|a_i), a_i \in \mathcal{A}, b_j \in \mathcal{B}$, matrica

$$\mathbf{P} = \begin{pmatrix} p(b_1|a_1) & p(b_2|a_1) & \ldots & p(b_s|a_1) \\ p(b_1|a_2) & p(b_2|a_2) & \ldots & p(b_s|a_2) \\ \ldots & \ldots & \vdots & \ldots \\ p(b_1|a_t) & p(b_2|a_t) & \ldots & p(b_s|a_q) \end{pmatrix}.$$

Diskrečiuosius perdavimo kanalus patogu vaizduoti diagramomis.



Perdavimo kanalo diagrama. Strėlės nukreiptos į tuos simbolius, kurie, siunčiant simbolį, gali būti gauti iš kanalo. Prie strėlių nurodytos sąlyginės tokio perdavimo tikimybės.

Kanalo tikimybių i-ojoje eilutėje surašytos tikimybės, kad, pasiuntus i-ąjį siuntėjo abėcėlės simbolį, bus gautas atitinkamai pirmasis, antrasis, ... gavėjo abėcėlės simbolis. Taigi kanalo tikimybių matricos kiekvienos eilutės elementų suma lygi vienetui.

4.2. Perdavimo kanalų įvairovė

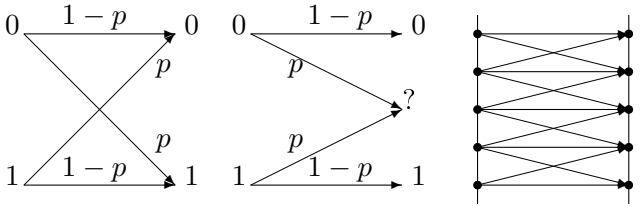
Tarptautinis arba vietinis paštas, telefonas, elektroninis paštas, kompaktinė plokštelė, popieriaus lapas, kurį galime prirašyti, ... , tuščias butelis, į kurį galima įdėti laišką ir paleisti upe pasroviui... Perdavimo kanalų yra įvairių, visų jų neaprėpsime. Panagrinėkime, keletą paprastų kanalų matematinių modelių.

Nagrinėsime tik neturinčius atminties kanalus. Pradėkime nuo atvejo, kai siuntėjas ir gavėjas naudoja dvejetainę abėcėlę.

19 apibrėžimas. *Perdavimo kanalą su dvejetaine siuntėjo ir gavėjo abėcėle $\mathcal{A} = \mathcal{B} = \{0, 1\}$ vadinsime dvejetainiu simetriniu kanalu, jeigu perdavimo tikimybių matrica yra*

$$\mathbf{P} = \begin{pmatrix} p(0|0) & p(1|0) \\ p(0|1) & p(1|1) \end{pmatrix} = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix},$$

čia p, $0 \le p \le 1$, reiškia vieno simbolio iškraipymo tikimybę.



Trys perdavimo kanalų modeliai. Dvejetainio simetrinio kanalo „požiūris“ į siunčiamus simbolius vienodas: jis visus juos vienodai linkęs iškreipti ir vienodai – perduoti teisingai. Trinantis kanalas su su visais simboliais irgi elgiasi vienodai: trina juos arba perduoda teisingai. Kai

renkame tekstą klaviatūra, dažnai atsitinka, kad vietoj reikalingo klavišo paspaudžiame vieną iš jam gretimų. „Klaviatūros kanalas“ pavaizduotas 3 brėžinyje.

20 apibrėžimas. *Perdavimo kanalą su dvejetaine siuntėjo abėcėle $\mathcal{A}$ ir gavėjo abėcėle $\mathcal{B} = \mathcal{A} \cup \{?\}$ vadinsime trinančiu kanalu, jei simbolio perdavimo tikimybės tokios:*

$$p(?|a) = p, \quad p(a|a) = 1 - p, \quad p(b|a) = 0, \quad a, b \in \mathcal{A}, a \neq b,$$

čia $0 \leq p \leq 1$ yra simbolio trynimo tikimybė.

Simetrinio kanalo sąvoką galima suformuluoti ir tam atvejui, kai siuntėjo ir gavėjo abėcėlės nėra dvejetainės.

21 apibrėžimas. *Perdavimo kanalą su siuntėjo abėcėle $\mathcal{A} = \{a_1, a_2, \ldots, a_q\}$ ir gavėjo abėcėle $\mathcal{B} = \{b_1, b_2, \ldots, b_s\}$ vadinsime simetriniu, jeigu kanalo tikimybių matricos*

$$\mathbf{P} = \begin{pmatrix} p(b_1|a_1) & p(b_2|a_1) & \ldots & p(b_s|a_1) \\ p(b_1|a_2) & p(b_2|a_2) & \ldots & p(b_s|a_2) \\ \ldots & \ldots & \vdots & \ldots \\ p(b_1|a_t) & p(b_2|a_t) & \ldots & p(b_s|a_q) \end{pmatrix}$$

eilutės yra gaunamos iš vienos eilutės, perstatant jos elementus, o kiekvieno stulpelio tikimybių suma yra ta pati.

Visos simetrinio kanalo tikimybių matricos eilutės sudarytos iš tų pačių skaičių. Galima įsivaizduoti, kad, „spręsdamas“, kaip perduoti siunčiamą simbolį, kanalas meta kauliuką, ant kurio sienelių užrašytos gavėjo abėcėlės raidės. Kokia raidė atvirto, tokia ir perduodama. Kauliukų yra tiek, kiek siuntėjo abėcėlėje raidžių: po vieną kiekvienai raidei. Fizinės kauliukų savybės vienodos, tik raidės ant sienelių užrašytos skirtingai. Sakote, kauliukų su trimis sienomis nėra? Tada ridenkite trisienę prizmę!

Kadangi tikimybių matricos kiekvienos eilutės elementų suma lygi vienetui, visų elementų suma bus lygi q. Tada simetrinio kanalo tikimybių matricos kiekvieno stulpelio tikimybių suma lygi $\frac{q}{s}$. Šią sąlygą galime interpretuoti, pavyzdžiui, taip: jeigu visi siuntėjo simboliai būtų perduodami su vienodomis tikimybėmis, tai ir visi gavėjo abėcėlės simboliai pasirodytų kitame kanalo gale su vienodomis tikimybėmis. Taigi simetrinis kanalas garantuoja, kad „lygiavos principas“, galiojantis siuntėjo pusėje, išliks galioti ir gavėjo pusėje.

Reikia simetrinių kanalų pavyzdžių? Jeigu kanalas, perduodamas kiekvieną simbolį „mestų“ tą patį simetrišką kauliuką, gautume paprasčiausią visiškai niekam netinkamą simetrinį kanalą.

Štai kiek įdomesnis pavyzdys. Tegu $\mathcal{A} = \{a, b, c\}$, o perdavimo kanalas su tikimybe p simbolį perduoda teisingai, su tikimybe x paverčia kitu simboliu, o su tikimybe $y = 1 - p - 2x$ ištrina. Taigi tikimybių matrica yra tokia:

$$P = \begin{pmatrix} p(a|a) & p(b|a) & p(?|a) \\ p(a|b) & p(b|b) & p(?|b) \\ p(a|c) & p(b|c) & p(?|c) \end{pmatrix} = \begin{pmatrix} p & x & x & y \\ x & p & x & y \\ x & x & p & y \end{pmatrix}.$$

Kada toks kanalas yra simetrinis? Nesunku įsitikinti, kad būtina ir pakankama sąlyga, kad kanalas trintų maždaug kas ketvirtą perduodamą simbolį, t. y. $y = 0,25$.

4.3. Kanalo talpa

Kiek vandens telpa į kibirą, tokia jo ir talpa. O jeigu kibiras kiauras? Kokia tada jo talpa?

Jeigu informacijos kanalas patikimas, kokį informacijos kiekį pasiųsime, tokį gavėjas ir gaus. O jeigu nepatikimas? Kiek informacijos galima perduoti tokiu kanalu?

Jeigu kibiras kiauras, tai pripylę jį sklidiną, nunešime jau nebe pilną. Tą vandens kiekį, kurį pavyko nunešti, galime pavadinti kiauro kibiro talpa.

Perdavimo kanalas iškraipo simbolius. Toks kanalas yra tarsi kiauras kibiras – dalis informacijos, siunčiant kanalu, prarandama. Kiekgi jos perduodama kanalu? Kaip apibrėžti ir apskaičiuoti kanalo talpą?

Prisiminkime abipusės informacijos sąvoką. Jeigu X ir Y yra du diskretieji atsitiktiniai dydžiai, tai abipuse informacija pavadinome skaičių

$$I(X, Y) = H(X) - H(X|Y).$$

Abipusė informacija – puikus įrankis kanalo talpai matuoti.

Siuntėją sutapatinkime su atsitiktiniu dydžiu U, o gavėją – su V. Dydžio U reikšmės – siuntėjo abėcėlės simboliai, o V reikšmės – gavėjo gauti simboliai. Tada gavėjo siunčiamos informacijos kiekis yra $H(U)$, $H(U|V)$ – kiekis, pradingęs kanale, o $I(U, V)$ – perduotas informacijos kiekis.

Žinoma, perduodamos informacijos kiekis priklauso nuo to, kiek jos siunčiama. Be to, kanalas – tai ne kiauras kibiras, jeigu siųsime daugiau ($H(U)$ reikšmė bus didesnė), nebūtinai daugiau ir perduosime. Perduodamos informacijos kiekis gali priklausyti nuo šaltinio savybių (siuntėjo abėcėlės tikimybių). Taigi norėdami nustatyti maksimalų informacijos kiekį, kurį galima perduoti kanalu, turėtume išbandyti visus šaltinius su ta pačia abėcėle.

Žymėkime $P_U = \langle p(a_1), p(a_2), \ldots, p(a_q) \rangle$ šaltinio tikimybių, t. y. tikimybių $p(a) = P(U = a)$, rinkinį. Tada kanalo talpą apibrėšime taip:

22 apibrėžimas. *Kanalo talpa vadinsime skaičių*

$$C = \max\{I(U,V) : P_U\}.$$

Kanalo talpa – tai maksimali perduodamos informacijos kiekio $I(U,V)$ reikšmė; šios funkcijos apibrėžimo sritis yra skirstinių aibė

$$\{\langle p_1, p_2, \ldots, p_q\rangle : p_i \geq 0, p_1 + p_2 + \ldots + p_q = 1\}.$$

Suskaičiuoti kanalo talpą dažnai nelengva. Tačiau kartais – visai nesunku. Pavyzdžiui, nesunku apskaičiuoti simetrinio kanalo talpą.

20 teorema. *Tegu siuntėjo abėcėlė yra $\mathcal{A} = \{a_1, a_2, \ldots, a_q\}$, gavėjo – $\mathcal{B} = \{b_1, b_2, \ldots, b_s\}$, kanalas yra simetrinis, o jo tikimybės yra $p(b_j|a_i)$. Tada kanalo talpa lygi*

$$C = \log_2 s - H, \quad H = \sum_{j=1}^{s} p(b_j|a_i) \log_2 \frac{1}{p(b_j|a_i)}.$$

Skaičiuodami dydį H, apibrėžtą teoremos formuluotėje, naudojame kanalo tikimybių matricos vienos eilutės elementus. Kadangi kanalas yra simetrinis, tai kiekvienoje eilutėje yra surašyti tie patys skaičiai, taigi H reikšmė nepriklauso nuo to, kurią eilutę pasirinksime.

Įrodymas. Pasinaudokime abipusės informacijos išraiška:

$$\begin{aligned}
I(U,V) &= H(V) - H(V|U), \\
H(V) &= \sum_{j=1}^{s} p(b_j) \log_2 \frac{1}{p(b_j)}, \\
H(V|U) &= \sum_{i=1}^{q} p(a_i) H(V|U = a_i), \\
H(V|U = a_i) &= \sum_{j=1}^{s} p(b_j|a_i) \log_2 \frac{1}{p(b_j|a_i)},
\end{aligned}$$

čia žymime

$$p(b_j) = P(V = b_j), \quad p(a_i) = P(U = a_i), \quad p(b_j|a_i) = P(V = b_j|U = a_i).$$

Dabar įsižiūrėkime ir pasvarstykime. Kadangi kanalas simetrinis, tai visos sąlyginės entropijos $H(V|U = a_i)$ lygios, t. y. $H(V|U = a_i) = H$. Tačiau tada

$$H(V|U) = H \sum_{i=1}^{q} p(a_i) = H$$

nepriklausomai nuo šaltinio tikimybių $p(a_i)$. Taigi

$$I(U,V) = H(V) - H,$$

ir didžiausią reikšmę dydis $I(U,V)$ įgis tada, kai $H(V)$ reikšmė bus didžiausia. Kadangi dydis V gali įgyti s skirtingų reikšmių, tai jo entropija negali būti didesnė už $\log_2 s$. Lygybė $H(V) = \log_2 s$ būtų teisinga tuomet, jei visos V reikšmės būtų vienodai galimos. Tačiau kanalas yra simetrinis, t. y. išlaiko „lygiavos principą“: jeigu visi siuntėjo simboliai bus siunčiami su vienodomis tikimybėmis, tai ir visi gavėjo simboliai pasirodys su vienodomis tikimybėmis. Taigi kanalo talpa

$$C = \max\{I(U,V) : P_U\} = \log_2 s - H. \tag{29}$$

Užrašykime kanalo talpos išraišką, kai kanalas dvinaris ir simetrinis, t. y. ir siuntėjo, ir gavėjo abėcėlės turi po du simbolius, o kanalo tikimybių matrica yra

$$\mathbf{P} = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}.$$

Išvada. *Dvinario simetrinio kanalo talpa yra*

$$C = 1 - p\log_2 p - (1-p)\log_2(1-p).$$

Tačiau jei dvinaris kanalas nėra simetrinis, kanalo talpą apskaičiuoti nelengva. Netikite – pabandykite. Užtat kanalo su trynimu talpą apskaičiuoti visai nesudėtinga.

Taigi siuntėjo abėcėlė yra $\mathcal{B} = \{0,1\}$, gavėjo – $\mathcal{B}^* = \{0,1,?\}$, o kanalo tikimybių matrica

$$\mathbf{P} = \begin{pmatrix} 1-p & 0 & p \\ 0 & 1-p & p \end{pmatrix}.$$

Jeigu $p = 1/3$, tai kanalas simetrinis, taigi jo talpą galime apskaičiuoti pasinaudoję (29):

$$C = \log_2 3 - \frac{1}{3}\log_2 3 - \frac{2}{3}\log_2\frac{3}{2} = \frac{2}{3}.$$

Jeigu kanalas ištrina simbolį su tikimybe $\frac{1}{3}$, t. y. ištrina maždaug trečdalį siunčiamų bitų, tai galime įsivaizduoti, kad siunčiant vieną bitą informacijos, mažiausiai trečdalis jo „nutrinama“. Galima spėti, kad jeigu kanalas trina simbolį su tikimybe p, tai iš vieno bito daugiausiai gali likti $1-p$ bito. Ir tai tiesa.

21 teorema. *Jeigu kanalas kiekvieną siuntėjo abėcėlės $\mathcal{A} = \{a_1, a_2, \ldots, a_q\}$ simbolį ištrina su tikimybe p ir su tikimybe $1-p$ perduoda teisingai, tai jo talpa*

$$C = (1-p)\log_2 q.$$

Įrodymas. Įrodysime teoremą dvejetainei abėcėlei, t. y. kai $q = 2$. Bendruoju atveju įrodymo idėjos yra tos pačios, tik ženklų ir indeksų daugiau.

$$H(V|U = 0) = H(V|U = 1) = p \log_2 \frac{1}{p} + (1 - p) \log_2 \frac{1}{1 - p}.$$

Taigi ir

$$H(V|U) = p \log_2 \frac{1}{p} + (1 - p) \log_2 \frac{1}{(1 - p)}.$$

Jeigu $P(U = 0) = x, P(U = 1) = 1 - x$, tai

$$P(V = 0) = x(1 - p), \quad P(V = 1) = (1 - x)(1 - p), \quad P(V =?) = p.$$

Šiek tiek paskaičiavę, gautume:

$$I(U, V) = H(V) - H(V|U) = (1-p)\Big(x \log_2 \frac{1}{x} + (1-x) \log_2 \frac{1}{1 - x}\Big) = (1-p)H(U).$$

Kadangi didžiausia galima $H(U)$ reikšmė yra 1, tai didžiausia galima $I(U, V)$ reikšmė, t. y. kanalo talpa, lygi $1 - p$.

4.4. Patikimo perdavimo kaina

Tarkime, kiauru kibiru reikia atnešti, pavyzdžiui, penkis litrus vandens. Kiekvienas žinome sprendimą: reikia iš pradžių įpilti daugiau. Tačiau perdavimo kanalo lyginimas su kiauru kibiru – nelabai vykęs. Iš tiesų, norėdami, kad gavėją mūsų informacija pasiektų patikimiau, mes turime ne padidinti siunčiamos informacijos kiekį, bet prieš ją siųsdami atitinkamai „įpakuoti“, t. y. parengti arba koduoti.

Galvojame: kaip patikimiau perduoti informaciją? Pirmoji mintis – kartokime perduodamus simbolius. Šitaip perduodamos informacijos kiekio nepadidinsime, tačiau galbūt saugiau „įpakuosime“.

Jeigu siunčiamą simbolį pakartosime, pavyzdžiui, tris kartus ir nurodysime, kad tris gautus simbolius reikia pakeisti tuo, kuris toje trijulėje pasitaiko dažniausiai, tai simbolio teisingo perdavimo tikimybę nuo $P_t = 1-p$ padidinsime iki

$$P_t^* = (1 - p)^3 + 3p(1 - p)^2,$$

čia p žymi tikimybę, kad siunčiamas simbolis kanale bus iškreiptas. Kai $p = 0,1$, tai $P_t = 0,9$, o $P_t^* = 0,972$. Nekartojant siunčiamų simbolių, gavėjas iš 1000 pasiųstų simbolių teisingai atpažins maždaug 900, o kartojant – maždaug 970. Tačiau patikimumo padidėjimo kaina – tris kartus sulėtėjęs ryšys! Tikimybė, kad visi 1000 simbolių bus perduoti teisingai, būtų vis tiek labai maža.

Jeigu norėtume, kad ji būtų didelė, ar perdavimo greičio neturėtume sumažinti kone iki nulio? Kiek kartų reiktų kartoti simbolį, kad klaidingo perdavimo tikimybė būtų mažesnė, pavyzdžiui, už 0,001?

Taigi tarkime, kad norime patikimai perduoti vieną bitą dvinariu simetriniu kanalu, kuris iškreipia simbolį su tikimybe $p < 1/2$. Nesugalvodami nieko geriau, nusprendėme pakartoti siunčiamą bitą n kartų, patarę gavėjui kiekvieną n bitų bloką keisti tuo simboliu, kuris šiame bloke pasitaiko dažniausiai.

Kiek kartų reikia kartoti siunčiamą simbolį, kad tikimybė, jog gavėjas, dekoduodamas jį, suklys, būtų mažesnė už ϵ?

Siunčiamas simbolis bus dekoduotas neteisingai, jeigu siunčiant n vienodų simbolių, iškraipyta bus ne mažiau kaip $n/2$. Tokio įvykio tikimybė lygi

$$P(\text{bus dekoduota klaidingai}|n) = \sum_{m>n/2} C_n^m p^m (1-p)^{n-m}.$$

Norėdami sužinoti, kiek kartų reiktų kartoti simbolį, kad klaidingo dekodavimo tikimybė būtų mažesnė už ϵ, turėtume spręsti nelygybę

$$P(\text{bus dekoduota klaidingai}|n) \leq \epsilon.$$

Šiek tiek paskaičiuokime. Suraskime vieno bito klaidingo dekodavimo tikimybes, kai perduodant jis kartojamas atitinkamai $3, 5, 7$ ir 9 kartus.

$n=$	3	5	7	9
$p = 0,1$	$0,0028$	$0,0856$	$0,002728$	$0,0009$
$p = 0,2$	$0,104$	$0,05792$	$0,03334$	$0,01958$

Lentelėje pateiktos vieno simbolio klaidingo dekodavimo tikimybės. Kai vieno simbolio iškraipymo tikimybė lygi $0,1$, *o perduodant kiekvienas simbolis kartojamas devynis kartus, vis tiek maždaug vienas iš tūkstančio simbolių bus perduotas klaidingai. Jeigu teksto simbolis koduojamas aštuoniais bitais, tai klaidingai perskaitytas bus maždaug vienas simbolis iš* 125. *Vidutinio dydžio puslapyje* 125 *simboliai užpildo maždaug dvi eilutes. Taigi klaidą rastume beveik kas antroje eilutėje!*

Nesugalvojus nieko geriau kaip kartoti perduodamą simbolį, didinti perdavimo patikimumą įmanoma tik lėtinant perdavimo greitį. „Sveikas protas“ tikriausiai su tuo sutiktų.

C. Shannonas įrodė, kad tokia „sveiko proto“ nuomonė čia visiškai klaidinga. Iš tikrųjų įmanoma, pavyzdžiui, dvinariu simetriniu kanalu su simbolio iškreipimo tikimybe $p = 0,1$ koduotą informaciją perduoti taip, kad klaidingo perdavimo tikimybė būtų kiek norime maža, o pats perdavimas nesulėtėtų net du kartus! Vienintelė papildoma sąlyga – turi būti siunčiama daug informacijos iš karto.

Didmeninės siuntos teikia daugiau privalumų!

4.5. Kodai ir dekodavimo taisyklės

Kaip sudaryti kodavimo, kaip dekodavimo taisykles? Pradėkime nuo pastarųjų...

Tegu šaltinio abėcėlė yra $\mathcal{A} = \{a_1, a_2, \ldots, a_q\}$, o kanalas stacionarus ir neturi atminties. Aptarėme patį paprasčiausią būdą padidinti perdavimo patikimumą – siunčiamo simbolio kartojimą. Tai labai paprastas kodavimas: vienas simbolis keičiamas vienodų simbolių bloku. O jeigu m simbolių ilgio žodžius keistume n ($n \geq m$) simbolių žodžiais?

Tarkime, pradinis šaltinio simbolių srautas bus skaidomas į m ilgio žodžius, o šie žodžiai koduojami (keičiami) tos pačios abėcėlės n ilgio žodžiais, parenkamais pagal tam tikrą taisyklę iš aibės $\mathbf{C} = \{\mathbf{c_1}, \ldots, \mathbf{c_N}\}$. Parinkti šią aibę taip, kad jos žodžių struktūra suteiktų galimybių padidinti perdavimo patikimumą, ir yra kodavimo esmė. Parinktų žodžių aibę vadinsime tiesiog kodu.

23 apibrėžimas. *Abėcėlės $\mathcal{A}$ n ilgio skirtingų žodžių rinkinį, turintį N elementų, vadinsime (n,N) kodu.*

Parametrą n vadinsime kodo ilgiu, N – kodo dydžiu.

Parinkę kodą, turime apibrėžti kodavimo taisyklę, t. y. injektyvų atvaizdį, kuriuo naudojantis m ilgio žodžiai keičiami kodo žodžiais. Injektyvumas reiškia, kad skirtingi žodžiai bus koduojami skirtingai. Kad tokį atvaizdį būtų galima apibrėžti, kodo žodžių turi būti ne mažiau kaip visų galimų m ilgio žodžių, t.y.

$$q^m \leq N.$$

Jeigu (n, N) kodo elementais koduojami m ilgio pradinio srauto žodžiai, tai perdavimo greičio sumažėjimą galime apibūdinti koeficientu

$$R = \frac{m}{n}.$$

Būtų patogu kodavimo įtaką perdavimo sąnaudoms (greičiui) nusakyti paties kodo parametrais.

24 apibrėžimas. *Tegu $\mathbf{C}$ yra (n, N) kodas, sudarytas iš q elementų turinčios abėcėlės žodžių. Dydį*

$$R(\mathbf{C}) = \frac{\log_q N}{n}$$

vadinsime kodo $\mathbf{C}$ koeficientu.

Koks šio koeficiento ryšys su perdavimo greičiu?

Tegu šaltinis mums pateikia abėcėlės $\mathcal{A}$, $|\mathcal{A}| = q$, simbolių srautą. Jį mes ketiname skaidyti m ilgio žodžiais, pastaruosius koduoti iš anksto parinkto (n, N) kodo $\mathbf{C}$ žodžiais ir siųsti juos kanalu. Kokį m pasirinkti, kad perdavimo greitis būtų kiek galima didesnis?

Kad kodo žodžių užtektų, turi būti patenkinta nelygybė

$$q^m \le N, \quad \text{arba} \quad m \le \log_q N.$$

Taigi didžiausia galima m reikšmė yra $m = [\log_q N]$, čia laužtiniai skliaustai žymi skaičiaus sveikąją dalį. Pasirinkę ją, gautume didžiausią perdavimo greitį, jis būtų lygus perdavimo be kodavimo greičiui, padaugintam iš dydžio

$$R = \frac{m}{n} = \frac{[\log_q N]}{n}.$$

Kodo koeficientas $R(\mathbf{C})$ yra beveik lygus šiam greičio sulėtėjimo koeficientui; atsisakę tikslios lygybės, gauname dydį, kurį patogiau naudoti įvairiuose analiziniuose reiškiniuose.

Taigi kanalu perduodami iš abėcėlės $\mathcal{A}$ simbolių sudaryti kodo $\mathbf{C}$ žodžiai, o gavėjas gauna to paties ilgio abėcėlės $\mathcal{B}$ žodžius. Pirmuosius žymėsime $\mathbf{c}, \mathbf{c_i}$, antruosius – $\mathbf{d}, \mathbf{d_j}$. Pažymėkime žodžių, kurie gali būti gauti, aibę $\mathbf{D}$. Apskritai galime manyti, kad $\mathbf{D} = \mathcal{B}^n$.

Dabar šaltinį galime tapatinti su atsitiktiniu vektoriumi, įgyjančiu reikšmes iš $\mathbf{C}$, gavėją – su vektoriumi, įgyjančiu reikšmes iš $\mathbf{D}$. Pažymėsime tikimybes:

$$\begin{aligned}
p(\mathbf{c_i}) &= P(\text{bus siųstas žodis } \mathbf{c_i}),\ \mathbf{c_i} \in \mathbf{C};\\
p(\mathbf{d_j}) &= P(\text{bus gautas žodis } \mathbf{d_j}),\ \mathbf{d_j} \in \mathbf{D};\\
p(\mathbf{c_i}, \mathbf{d_j}) &= P(\text{bus siųstas žodis } \mathbf{c_i},\ \text{o gautas } \mathbf{d_j}),\ \mathbf{c_i} \in \mathbf{C},\ \mathbf{d_j} \in \mathbf{D};\\
p(\mathbf{d_j}|\mathbf{c_i}) &= P(\text{jei bus siųstas žodis } \mathbf{c_i}, \text{tai gautas } \mathbf{d_j}),\ \mathbf{c_i} \in \mathbf{C},\ \mathbf{d_j} \in \mathbf{D};\\
p(\mathbf{c_i}|\mathbf{d_j}) &= P(\text{jei gautas žodis } \mathbf{d_j}, \text{tai buvo siųstas } \mathbf{c_i}),\ \mathbf{c_i} \in \mathbf{C},\ \mathbf{d_j} \in \mathbf{D}.
\end{aligned}$$

Primenu, kad sąlyginės tikimybės reiškiamos per besąlygines taip:

$$p(\mathbf{d_j}|\mathbf{c_i}) = \frac{p(\mathbf{c_i}, \mathbf{d_j})}{p(\mathbf{c_i})}, \quad p(\mathbf{c_i}|\mathbf{d_j}) = \frac{p(\mathbf{c_i}, \mathbf{d_j})}{p(\mathbf{d_j})}.$$

Pastebėkime, kad tikimybes $p(\mathbf{d_j}|\mathbf{c_i})$ galime reikšti kanalo tikimybėmis: jei $\mathbf{c_i} = a_1 a_2 \dots a_n$, $\mathbf{d_j} = b_1 b_2 \dots b_n$; čia $a_u \in \mathcal{A}$, $b_v \in \mathcal{B}$, tai

$$p(\mathbf{d_j}|\mathbf{c_i}) = p(b_1|a_1)p(b_2|a_2) \cdot \dots \cdot p(b_n|a_n).$$

Tikimybės $p(\mathbf{c_i})$ priklauso nuo šaltinio savybių, o $p(\mathbf{d_j})$ ir $p(\mathbf{c_i}|\mathbf{d_j})$ gali būti išreikštos šaltinio ir kanalo tikimybėmis.

Dabar panagrinėkime, kas gi vyksta aname kanalo gale. Gavėjas, priėmęs kurį nors aibės $\mathbf{D}$ žodį $\mathbf{d}$, turi stengtis atpažinti, koks kodo $\mathbf{C}$ žodis buvo jam siųstas. Taigi jis turi vadovautis tam tikra dekodavimo taisykle.

25 apibrėžimas. *Tegu* $\mathbf{C}$ *yra kodas, o* $\mathbf{D}$ *priimamų žodžių aibė. Dekodavimo taisykle vadinsime funkciją*

$$f : \mathbf{D} \to \mathbf{C}.$$

Dekoduojant, suprantama, galimos klaidos. Kaip skaičiuoti jų tikimybes? Jeigu pasiuntėme kodo žodį $\mathbf{c}$ ir dekoduojame naudodami taisyklę f, tai klaidos tikimybė

$$P(klaida|\mathbf{c}) = \sum_{\mathbf{d}:\mathbf{d}\notin f^{-1}(\mathbf{c})} p(\mathbf{d}|\mathbf{c}).$$

Taip pat galime užrašyti klaidos tikimybę, jeigu priimtas žodis $\mathbf{d}$:

$$P(klaida|\mathbf{d}) = \sum_{\mathbf{c}\neq f(\mathbf{d})} p(\mathbf{c}|\mathbf{d}) = 1 - p(f(\mathbf{d})|\mathbf{d}). \tag{30}$$

Tada galime apibrėžti vidutinę klaidos tikimybę:

$$p_{vid} = \sum_{\mathbf{c}} P(klaida|\mathbf{c})p(\mathbf{c}) = \sum_{\mathbf{d}} P(klaida|\mathbf{d})p(\mathbf{d}).$$

Pasiremdami (30), gausime

$$p_{vid} = 1 - \sum_{\mathbf{d}} p(f(\mathbf{d})|\mathbf{d})p(\mathbf{d}). \tag{31}$$

Ši vidutinės klaidos išraiška rodo, kaip sudaryti dekodavimo taisyklę, kuri minimizuotų vidutinę klaidą. Iš tikrųjų tam reikia pasiekti, kad sumos reikšmė (31) lygybėje būtų didžiausia. Taip bus, jei žodžiui $\mathbf{d}$ parinksime $f(\mathbf{d})$ taip, kad tikimybė $p(f(\mathbf{d})|\mathbf{d})$ būtų maksimali.

26 apibrėžimas. *Tegu* $\mathbf{C}$ *yra kodas, o* $\mathbf{D}$ *– priimamų žodžių aibė. Dekodavimo taisyklę* $f : \mathbf{D} \to \mathbf{C}$ *vadinsime idealaus stebėtojo taisykle, jei kiekvienam* $\mathbf{d} \in \mathbf{D}$ f *tenkina sąlygą*

$$p(f(\mathbf{d})|\mathbf{d}) = \max_{\mathbf{c}} p(\mathbf{c}|\mathbf{d}). \tag{32}$$

Pavadinimą galima paaiškinti šitaip. Pasinaudodami sąlyginės tikimybės apibrėžimu, gausime

$$p(\mathbf{c}|\mathbf{d}) = \frac{p(\mathbf{c},\mathbf{d})}{p(\mathbf{d})} = \frac{p(\mathbf{d}|\mathbf{c})p(\mathbf{c})}{p(\mathbf{d})}.$$

Iš šios išraiškos matome, kad, norėdami žodžiui $\mathbf{d}$ parinkti $\mathbf{c}$, maksimizuojantį tikimybę $p(\mathbf{c}|\mathbf{d})$, turime žinoti ne tik kanalo, bet ir šaltinio tikimybes, t. y. turime būti idealūs stebėtojai.

Susumuokime atliktos analizės rezultatus.

22 teorema. *Idealaus stebėtojo taisyklė minimizuoja vidutinę klaidos tikimybę* p_{vid}.

Sukeitę (32) lygybėje argumentus vietomis, gausime kitos taisyklės apibrėžimą.

27 apibrėžimas. *Tegu* $\mathbf{C}$ *yra kodas, o* $\mathbf{D}$ *– priimamų žodžių aibė. Dekodavimo taisyklę* $f : \mathbf{D} \to \mathbf{C}$ *vadinsime didžiausio tikėtinumo taisykle, jei kiekvienam* $\mathbf{d} \in \mathbf{D}$ f *tenkina sąlygą*

$$p(\mathbf{d}|f(\mathbf{d})) = \max_{\mathbf{c}} p(\mathbf{d}|\mathbf{c}). \tag{33}$$

Nesunku įsitikinti, kad didžiausio tikėtinumo taisyklei sudaryti pakanka žinoti tik kanalo tikimybes. Taigi dekoduodami pagal maksimalaus tikėtinumo taisyklę, atsižvelgiame tik į perdavimo kanalo savybes. Apibrėžkime vidutinę klaidos tikimybę kiek kitaip:

$$p^*_{vid} = \frac{1}{N} \sum_{\mathbf{c}} P(klaida|\mathbf{c});$$

čia N yra kodo žodžių skaičius. Ši tikimybė sutampa su p_{vid}, kai visi kodo žodžiai perduodami su vienodomis tikimybėmis.

23 teorema. *Jei šaltinis perduoda visus kodo žodžius su vienodomis tikimybėmis, tai idealaus stebėtojo ir didžiausio tikėtinumo taisyklių apibrėžimai yra ekvivalentūs.*

Įrodymas. Tegu $\mathbf{d}$ yra informacijos gavėjo priimtas žodis. Pakaks įrodyti, kad tas pats kodo žodis $\mathbf{c}$ maksimizuoja tiek $p(\mathbf{c}|\mathbf{d})$, tiek $p(\mathbf{d}|\mathbf{c})$. Pasirėmę sąlyginės tikimybės apibrėžimu ir tuo, kad $p(\mathbf{c}) = 1/N$, gausime

$$p(\mathbf{c}|\mathbf{d}) = \frac{p(\mathbf{d}|\mathbf{c})p(\mathbf{c})}{p(\mathbf{d})} = \frac{p(\mathbf{d}|\mathbf{c})}{Np(\mathbf{d})}.$$

Įrodymo pabaiga akivaizdi.

Taigi dekodavimo taisyklių galime ieškoti spręsdami tikimybių p_{vid}, p^*_{vid} minimizavimo uždavinius. Kartais patogu vietoj šių tikimybių naudoti maksimalią klaidos tikimybę

$$p_{max} = \max_{\mathbf{c}} P(klaida|\mathbf{c}).$$

Kai šaltinio bei gavėjo abėcėlės sutampa, apibrėšime dar vieną dekodavimo taisyklę.

28 apibrėžimas. *Tegu* $\mathbf{x} = x_1 \dots x_n$, $\mathbf{y} = y_1 \dots y_n$ *yra du abėcėlės* $\mathcal{A}$ *žodžiai. Hammingo atstumu tarp jų vadinsime skaičių*

$$h(\mathbf{x}, \mathbf{y}) = \sum_{\substack{i=1,\dots,n \\ x_i \neq y_i}} 1.$$

Hammingo atstumas tarp dviejų to paties ilgio žodžių lygus nesutampančių komponenčių skaičiui.

29 apibrėžimas. *Dekodavimo taisyklę $f : \mathbf{D} \to \mathbf{C}$ vadinsime minimalaus atstumo taisykle, jei kiekvienam* $\mathbf{d}$

$$h(f(\mathbf{d}), \mathbf{d}) = \min_{\mathbf{c}} h(\mathbf{c}, \mathbf{d}).$$

Taikyti minimalaus atstumo taisyklę labai paprasta – tereikia turėti kodo žodžių lentelę ir mokėti suskaičiuoti, keliomis komponentėmis skiriasi du nagrinėjami žodžiai.

24 teorema. *Jeigu siuntėjo ir gavėjo abėcėlės sutampa, o kanalo tikimybės yra*

$$p(a|a) = 1 - p, \quad p(b|a) = p, \quad a \neq b, \quad p < 0,5,$$

tai maksimalaus tikėtinumo ir minimalaus atstumo dekodavimo taisyklių apibrėžimai ekvivalentūs.

Įrodymas. Tegu $\mathbf{d}$ yra gautas žodis, $\mathbf{c}$ – bet koks kodo žodis ir $h = h(\mathbf{c}, \mathbf{d})$. Tada

$$p(\mathbf{c}|\mathbf{d}) = p^h(1-p)^{n-h} = (1-p)^n \Big(\frac{p}{1-p}\Big)^h.$$

Kadangi $p/(1-p) < 1$, tai $p(\mathbf{c}|\mathbf{d})$ įgyja didžiausią reikšmę, kai h reikšmė yra mažiausia.

4.6. Shannono teorema dvinariam kanalui

Bent jau du šio skyrelio puslapius reiktų įveikti. Juose dėstomas vienas svarbiausių informacijos ir kodavimo teorijos teiginių.

Tarkime, nusprendėme sudaryti kodą $\mathbf{C}$ iš abėcėlės $\mathcal{A}$, $|\mathcal{A}| = q$, žodžių. Jei parinksime N žodžių iš aibės $\mathcal{A}^n$, t. y. sudarysime (n, N) kodą $\mathbf{C}$, tai informacijos perdavimo sulėtėjimą, naudojant šį kodą, nusakys koeficientas

$$R(\mathbf{C}) = \frac{\log_q N}{n}.$$

Siekdami sparčiau perduoti informaciją, turėtume didinti $N \leq q^n$. Tačiau kai žodžių daug, sunku sudaryti patikimą dekodavimo taisyklę. Juk nesunku patikimai skirti kelias pagrindines spalvas, bet lengva suklysti, kai tenka skirti šimtus atspalvių!

Pusiausvyrą, kurią šioje padėtyje galima pasiekti, apibūdina teorema, kurią 1948 m. įrodė C. Shannonas. Jo straipsnis[3], kurį minėjome jau anksčiau, laikomas pirmuoju kodavimo teorijos, kaip atskiros tyrimų srities, darbu.

[3] „Mathematical theory of communication'", 'The Bell System Technical Journal, 1948.

Paprastumo dėlei nagrinėsime dvinarį simetrinį kanalą su vieno simbolio iškraipymo tikimybe $p \neq 0,5$. Sąlyga $p \neq 0,5$ reiškia, kad atsisakome nagrinėti bevertį kanalą. Iš tiesų, jeigu simbolis iškreipiamas su tikimybe $p = 0,5$, tada gavėjui iš kanalo gauta informacija bus tiek pat vertinga kiek simetriškos monetos metimų serijos rezultatų seka.

Taigi C. Shannono teorema.

25 teorema. *Tegu dvinario simetrinio kanalo talpa lygi $\mathcal{C}$, o vieno simbolio iškraipymo tikimybė $p \neq 0,5$. Tada bet kokiam skaičiui R, $0 < R < \mathcal{C}$, egzistuoja kodų $\mathbf{C}_m$ ir juos atitinkančių dekodavimo taisyklių seka, kad*

$$R(\mathbf{C}_m) \geq R, \quad p_{max}(\mathbf{C}_m) \to 0, \quad m \to \infty.$$

Priminsime, kad kodui $\mathbf{C}$ apibrėžėme

$$p_{max}(\mathbf{C}) = \max_{\mathbf{c} \in \mathbf{C}} P(klaida|\mathbf{c}).$$

Išvada. *Tegu dvinario simetrinio kanalo talpa lygi $\mathcal{C}$, o vieno simbolio iškraipymo tikimybė $p \neq 0,5$. Tada egzistuoja kodų $\mathbf{C}_m$ bei juos atitinkančių dekodavimo taisyklių seka, kad*

$$R(\mathbf{C}_m) \to \mathcal{C}, \quad p_{max}(\mathbf{C}_m) \to 0, \quad m \to \infty.$$

Jei, pavyzdžiui, $p = 0,1$, tai kanalo talpa

$$C = 1 - p \log_2 \frac{1}{p} - (1-p) \log_2 \frac{1}{1-p} \approx 0.584.$$

Taigi galime sumažinti žodžio klaidingo perdavimo tikimybę kiek tik norime, nesulėtindami perdavimo netgi du kartus. Tiesiog puiki perspektyva! Tačiau gražūs matematiniai rezultatai nebūtinai tobulai išsprendžia praktines problemas. Viena vertus, iš teoremos išplaukia, kad itin patikimas perdavimas galimas tik tuomet, kai koduojami ir siunčiami dideli informacijos kiekiai (n turi būti didelis!). Nors žodžio klaidingo dekodavimo tikimybė ir nedidelė, įvykus klaidai, būtų prarastas didelis informacijos blokas. Tačiau dar svarbiau, kad teoremos įrodymas visiškai nerodo, kaip tie geri kodai gali būti sukonstruoti. Net jei ir sukonstruotume tokį kodą, jo naudojimas gali pareikalauti didelių išteklių – juk reikia atlikti kodavimo ir dekodavimo operacijas!

Taigi ši nuostabi Shannono teorema yra tarsi koks puikus filosofijos veikalas – ji nurodo perspektyvą, paskatina susimąstyti, tačiau... Kai iškyla koks nors praktinis gyvenimo klausimas, atsakymo dažniausiai ieškome ne filosofų raštuose, bet kokioje nors „Namų ūkio enciklopedijoje“.

Ir vis dėlto – gaila tų, kuriems filosofija visiškai neįdomi! Taigi sugaiškime kiek laiko ir panagrinėkime Shannono teoremą nuodugniau.

Pirmiausia parodysime, kad pakanka įrodyti Shannono teoremą pakeitus joje tikimybę p_{max} vidutine klaidos tikimybe p^*_{vid}. Ši tikimybė kodui $\mathbf{C}$ apibrėžiama taip:

$$p^*_{vid}(\mathbf{C}) = \frac{1}{|\mathbf{C}|}\sum_{\mathbf{c}\in\mathbf{C}} P(klaida|\mathbf{c}).$$

26 teorema. *Tegu dvinario diskretaus kanalo be atminties talpa lygi $\mathcal{C}$. Tada šie teiginiai yra ekvivalentūs:*

1. *bet kokiam skaičiui R, $0 < R < \mathcal{C}$, egzistuoja kodų $\mathbf{C}_m$ ir juos atitinkančių dekodavimo taisyklių seka, kad*

$$R(\mathbf{C}_m) \geq R, \quad p_{max}(\mathbf{C}_m) \to 0, \quad m \to \infty; \tag{34}$$

2. *bet kokiam skaičiui R, $0 < R < \mathcal{C}$, egzistuoja kodų $\mathbf{C}_m$ ir juos atitinkančių dekodavimo taisyklių seka, kad*

$$R(\mathbf{C}_m) \geq R, \quad p^*_{vid}(\mathbf{C}_m) \to 0, \quad m \to \infty. \tag{35}$$

Įrodymas. Kadangi

$$p^*_{vid}(\mathbf{C}) \leq p_{max}(\mathbf{C}),$$

tai kodai, tenkinantys (34), tenkins ir (35). Tad iš 1) išplaukia 2).

Tegu dabar 2) teiginys teisingas. Imkime skaičius $0 < R < \mathcal{C}, \epsilon > 0$, ir parinkime R', kad būtų $R < R' < \mathcal{C}$. Kadangi 2) teiginys teisingas, tai egzistuoja (n_m, N_m) kodų seka $\mathbf{C}_m$ su juos atitinkančiomis dekodavimo taisyklėmis, kad $R(\mathbf{C}_m) \geq R'$, $p^*_{vid}(\mathbf{C}_m) \leq 0,5\epsilon$. Imkime kurį nors kodą $\mathbf{C}_m$, kurio ilgis $n = n_m$ yra toks didelis, jog $R + 1/n \leq R'$. Tada iš sąlygos $R(\mathbf{C}_m) \geq R' \geq R + 1/n$ gauname

$$N_m = 2^{nR(\mathbf{C}_m)} \geq 2^{nR+1}. \tag{36}$$

Kadangi

$$p^*_{vid}(\mathbf{C}_m) = \frac{1}{N_m}\sum_{\mathbf{c}\in\mathbf{C_m}} P(klaida|\mathbf{c}) \leq 0.5\epsilon,$$

tai bent pusei kodo žodžių turi galioti

$$P(klaida|\mathbf{c}) \leq \epsilon.$$

Sudarykime tik iš šių žodžių naują (n, N'_m) kodą $\mathbf{C}'_m$; jo žodžių skaičius N'_m ne mažesnis už $N_m/2$. Iš (36) gauname $N'_m \geq 2^{nR+1}/2 = 2^{nR}$. Tada mūsų naujajam kodui galioja nelygybė

$$R(\mathbf{C}'_m) = \frac{\log_2 N'_m}{n} \geq R.$$

Iš (4.6.) gauname $p_{max}(\mathbf{C'}_m) \leq \epsilon$. Skaičių ϵ galime parinkti kiek norima mažą. Teorema įrodyta.

O dabar aptarkime Shannono teoremos įrodymo idėjas. Priminsime, kad šaltinio ir gavėjo abėcėlės sutampa: $\mathcal{A} = \mathcal{B} = \{0, 1\}$, o kanalo talpa

$$\mathcal{C} = \mathcal{C}(p) = 1 + p \log_2 p + (1 - p) \log_2 (1 - p);$$

čia p yra simbolio iškraipymo tikimybė.

Pakanka nagrinėti atvejį $0 < p < 1/2$. Iš tikrųjų, jeigu kanalas yra toks, kad jis labiau linkęs iškreipti simbolį nei perduoti jį teisingai, tai gavėjas prieš dekoduodamas gali kiekviename priimtame žodyje nulius keisti vienetais, vienetus – nuliais ir manyti, kad toks srautas gautas iš kanalo, kuris iškraipo simbolį su tikimybe $1 - p < 1/2$.

Pradėkime ne nuo kodų, bet nuo dekodavimo taisyklių, kurias apibrėšime pasinaudoję Hammingo atstumu tarp žodžių $\mathbf{x}, \mathbf{y} \in \mathcal{A}^n$.

Priminsime, kad Hammingo atstumu tarp $\mathbf{x} = x_1 x_2 \dots x_n$, $\mathbf{y} = y_1 y_2 \dots y_n$ vadiname dydį

$$h(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} |x_i - y_i|.$$

Fiksavę $\mathbf{d}$ ir $r \geq 1$, apibrėžkime r spindulio rutulį $B(\mathbf{d}, r)$:

$$B(\mathbf{d}, r) = \{\mathbf{x} : \mathbf{x} \in \mathcal{A}^n, h(\mathbf{x}, \mathbf{d}) \leq r\}.$$

Skaičiui r apibrėšime dekodavimo taisyklę $f_r = f, f : \mathcal{A}^n \to \mathbf{C}$.

Tegu kanalu perduodant vieną kodo $\mathbf{C}$ žodį, buvo gautas žodis $\mathbf{d}$. Peržiūrėkime aibę $B(\mathbf{d}, r)$. Jeigu joje yra tik vienas kodo $\mathbf{C}$ žodis $\mathbf{c}$, juo ir dekoduokime: $f(\mathbf{d}) = \mathbf{c}$. Jei nėra nė vieno ar yra bent du, dekoduokime bet kaip, pavyzdžiui, pirmuoju kodo žodžiu: $f(\mathbf{d}) = \mathbf{c_1}$.

Vidutinę klaidos tikimybę, kuri gaunama naudojant kodą $\mathbf{C}$ ir ką tik apibrėžtą dekodavimo taisyklę, žymėsime $p^*_{vid}(\mathbf{C}|r)$.

Dekodavimo taisykles jau sugalvojome, kaipgi sudaryti pačius kodus? Užuot vargę ieškodami gero kodo, nagrinėsime visų įmanomų (n, N) kodų aibę, kurią galima sutapatinti su

$$\mathbf{C}(n, N) = \underbrace{\mathcal{A}^n \times \ldots \times \mathcal{A}^n}_{N}.$$

Šios aibės elementai – n ilgio žodžių rinkiniai po N žodžių – visi įmanomi kodai.

Mes kuriam laikui pavadiname kodais net pačius netinkamiausius žodžių rinkinius, pavyzdžiui, sudarytus iš vieno N kartų pakartoto elemento. Susitarkime, kad žodis, kode pakartotas s kartų ir įeinantis į $B(\mathbf{d}, r)$, duoda lygiai s bendrų rutulio ir kodo elementų. Tada į kanalą siunčiamas žodis, kuris kode kartojasi bent du kartus, bus visada dekoduojamas klaidingai.

Aibėje $\mathbf{C}(n,N)$ yra iš viso 2^{nN} elementų. Sudarykime aritmetinį klaidos tikimybės vidurkį:

$$\pi = \frac{1}{2^{nN}} \sum_{\mathbf{C}\in\mathbf{C}(n,N)} p^*_{vid}(\mathbf{C}|r).$$

Fiksuokime skaičius $0 < R < \mathcal{C},\ \epsilon > 0$.

Įrodymo esmę sudaro samprotavimai, jog galima parinkti n, N ir r, kad būtų

$$\pi \le \epsilon, \quad N \ge 2^{nR}.$$

Jeigu aritmetinis dydžių vidurkis yra ne didesnis už ϵ, tai turi būti bent vienas dydis, ne didesnis už ϵ! Taigi turi būti nors vienas $\mathbf{C}_n \in \mathbf{C}(n,N)$, kad $p^*_{vid}(\mathbf{C}|r) \le \epsilon$. Šis kodas ir yra tas „gerasis“ kodas, kurio egzistavimą siekiama nustatyti.

Gali būti, kad skaitytojui kils abejonė, ar tas „gerasis kodas“, kurio egzistavimą nustatėme, iš tikrųjų yra „tikras“ kodas, t. y. ar visi jo žodžiai yra skirtingi. Iš tikrųjų, taip gali ir nebūti, tačiau gautąjį kodą galima pataisyti. Tegu žodžių, kurie kartojasi parinktame kode $\mathbf{C}_n$, (kartu su pasikartojimais) yra s. Kadangi siunčiant kiekvieną iš jų su tikimybe 1 dekoduojama klaidingai, tai šių žodžių įnašas į tikimybę $p^*_{vid}(\mathbf{C}_n|r)$ yra $s/N_n \le \epsilon$. Išmetę visus šiuos žodžius, gausime "tikrą" kodą $\mathbf{C'}_n$, kuriam klaidos tikimybė yra mažesnė, o žodžių skaičius $N'_n \ge (1-\epsilon)N_n$. Taigi

$$R(\mathbf{C'}_n) \ge R(\mathbf{C}_n) + \log_2(1-\epsilon)/n.$$

Todėl kodų, kurių egzistavimą skelbia Shannono teorema, seką galima sudaryti iš „tikrų“ kodų.

4.7. Tik matematikams: Shannono teoremos įrodymas

Shannono teoremos įrodymo logika paprasta – rinkime žodžius į kodą atsitiktinai, skaičiuokime klaidingo dekodavimo tikimybes, jų vidurkį... Vidurkis mažas, vadinasi, egzistuoja bent vienas geras kodas! Tačiau „įvilkti“ šiuos samprotavimus į tikslią matematinę kalbą nėra taip paprasta.

Kodams $\mathbf{C} \in \mathbf{C}(n,N)$ dekoduoti taikysime anksčiau apibrėžtas dekodavimo taisykles $f_r : \mathcal{A}^n \to \mathbf{C}$. Jų taikymą galime nusakyti taip: gavęs iš kanalo žodį $\mathbf{d}$, gavėjas turi peržiūrėti rutulyje

$$B(\mathbf{d},r) = \{\mathbf{x} : \mathbf{x} \in \mathcal{A}^n, h(\mathbf{x},\mathbf{d}) \le r\}$$

esančius žodžius. Jeigu rutulyje yra tik vienas žodis $\mathbf{c} \in \mathbf{C}$, tai juo ir dekoduojama: $f_r(\mathbf{d}) = \mathbf{c}$. Jeigu tokių žodžių iš viso nėra arba yra daugiau nei vienas – tenka susitaikyti su dekodavimo klaida; jeigu būtinai reikia dekoduoti – galima parinkti bet kurį $\mathbf{C}$ žodį.

Kiekgi iš viso yra žodžių rutulyje $B(\mathbf{d}, r)$? Nesunku suvokti, kad žodžių skaičius nepriklauso nuo rutulio centro $\mathbf{d}$.

27 teorema. *Rutulio $B(\mathbf{d}, r)$ žodžių skaičius*

$$V_n(r) = |B(\mathbf{d}, r)| = \sum_{0 \le k \le r} C_n^k. \tag{37}$$

Jeigu $r \le n/2$, tai

$$V_n(r) \le 2^{nH(r/n)};$$

čia

$$H(\lambda) = \lambda \log_2 \frac{1}{\lambda} + (1-\lambda) \log_2 \frac{1}{1-\lambda}.$$

Įrodymas. Rutuliui priklauso jo centras, jį atitinka dėmuo C_n^0; žodžius, kurie skiriasi nuo centro viena komponente „suskaičiuoja“ dėmuo C_n^1 ir t. t. Taigi (37) lygybė teisinga.

Pažymėkime $\lambda = r/n$; jei $r < n/2$, tai $\lambda \le 1/2$, $\lambda/(1-\lambda) < 1$. Tada gausime

$$\begin{aligned} 1 &= (\lambda + 1 - \lambda)^n \ge \sum_{0 \le k \le r} C_n^k \lambda^k (1-\lambda)^{n-k} \\ &\ge (1-\lambda)^n \sum_{0 \le k \le r} C_n^k \Big(\frac{\lambda}{1-\lambda}\Big)^{\lambda n} = \lambda^{\lambda n} (1-\lambda)^{n(1-\lambda)} V_n(r). \end{aligned}$$

Taigi

$$V_n(r) \le 2^{n(\lambda \log_2 \frac{1}{\lambda} + (1-\lambda)\log_2 \frac{1}{1-\lambda})} = 2^{nH(\lambda)}.$$

Teiginys įrodytas.

Shannono teoremai įrodyti bus reikalingas dar vienas įrankis – Čebyšovo nelygybė.

28 teorema. *Tegu X yra atsitiktinis dydis, įgyjantis reikšmes iš realiųjų skaičių aibės, turintis vidurkį $\mathbf{E}[X]$ ir dispersiją $\mathbf{D}[X]$. Tada bet kokiam skaičiui $\epsilon > 0$ teisinga nelygybė*

$$P(|X - \mathbf{E}[X]| > \epsilon) < \frac{\mathbf{D}[X]}{\epsilon^2}. \tag{38}$$

Be to, dar pasinaudosime šiomis vidurkio ir dispersijos savybėmis:

1. jei X, Y yra nepriklausomi atsitiktiniai dydžiai, turintys vidurkius, tai ir jų sandauga turi vidurkį: $\mathbf{E}[X \cdot Y] = \mathbf{E}[X] \cdot \mathbf{E}[Y]$;
2. jei dydis X turi dispersiją, o c yra skaičius, tai $\mathbf{D}[cX] = c^2 \mathbf{D}[X]$;

3. jei $X_1, X_2, \ldots, X_n$ yra nepriklausomi atsitiktiniai dydžiai, turintys dispersijas, tai ir jų suma turi dispersiją:

$$\mathbf{D}[X_1 + X_2 + \ldots + X_n] = \mathbf{D}[X_1] + \mathbf{D}[X_2] + \ldots + \mathbf{D}[X_n].$$

Dekodavimo taisyklę turime, o pačių kodų – ne. Kadangi nežinome, kaip sudaryti kodą, tenkinantį teoremos sąlygas, tai kodo žodžius iš $\mathcal{A}^n$ vieną po kito rinksime atsitiktinai ir nepriklausomai. Tada į kodą $\mathbf{C}$ galėsime žvelgti kaip į atsitiktinį vektorių $\mathbf{C} = \langle \omega_1, \ldots, \omega_N \rangle$, kurio komponentės ω_i yra nepriklausomi atsitiktiniai dydžiai, su vienodomis tikimybėmis įgyjantys reikšmes iš aibės $\mathcal{A}^n$. Vidutinės klaidos tikimybė $p^*_{vid}(\mathbf{C}|r)$ irgi virsta atsitiktiniu dydžiu, o jo vidurkis lygus

$$\pi = \frac{1}{2^{nN}} \sum_{\mathbf{C} \in \mathbf{C}(n,N)} p^*_{vid}(\mathbf{C}|r).$$

Taigi mūsų tikslas – parodyti, kad, fiksavus $0 < R < \mathcal{C}, \epsilon > 0$, įmanoma taip parinkti n, N ir r, kad būtų

$$N \geq 2^{nR}, \quad \pi = \pi(n, N, r) = \mathbf{E}[p^*_{vid}(\mathbf{C}|r)] \leq \epsilon.$$

Tirkime dydį

$$p^*_{vid}(\mathbf{C}|r) = \frac{1}{N} \sum_{\omega_i \in \mathbf{C}} P(klaida|\omega_i).$$

Kada gi, pasiuntus ω_i ir dekoduojant pagal taisyklę f_r, įvyksta klaida? Galimi du atvejai:

1. priimtas žodis $\mathbf{d}$, bet $\omega_i \notin B(\mathbf{d}, r)$;
2. priimtas žodis $\mathbf{d}$ ir egzistuoja $j,\ i \neq j,\ \omega_j \in B(\mathbf{d}, r)$.

Sakysime, kad pirmuoju atveju įvyko pirmos, antruoju – antros rūšies klaida. Taigi

$$P(\text{klaida}|\omega_i) \leq P(\text{I rūšies klaida}|\omega_i) + P(\text{II rūšies klaida}|\omega_i).$$

Kada įvyks pirmos rūšies klaida? Kai kanalas iškraipys daugiau negu r siunčiamo žodžio ω_i simbolių.

Apibrėžę n nepriklausomų atsitiktinių dydžių

$$X_k = \begin{cases} 1, & \text{jei } k\text{-asis } \omega_i \text{ simbolis iškraipytas,} \\ 0, & \text{jei } k\text{-asis } \omega_i \text{ simbolis neiškraipytas,} \end{cases}$$

gausime

$$P(\text{I rūšies klaida}|\omega_i) = P(X_1 + \ldots + X_n > r).$$

Susiesime dekodavimo taisyklės parametrą r su kodo žodžių ilgiu. Imsime $r = (p+\sigma)n$; čia $0 < \sigma < 0,5-p$ yra skaičius, kurio parinkimą patikslinsime vėliau. Kadangi $P(X_k = 1) = p,\ P(X_k = 0) = 1-p$, tai $\mathbf{E}[X_k] = p$ ir

$$\begin{aligned} P(\text{I rūšies klaida}|\omega_i) &= P(\sum_{k=1}^{n} X_k > r) \\ &\le P(|\sum_{k=1}^{n} X_k - \sum_{k=1}^{n} \mathbf{E}[X_k]| > r - np). \end{aligned}$$

Pirmos rūšies klaidos įverčiui gauti pasinaudosime Čebyšovo nelygybe atsitiktiniam dydžiui $Y = X_1 + \ldots + X_n$:

$$P(\text{I rūšies klaida}|\omega_i) \le \frac{\mathbf{D}[X_1 + X_2 + \ldots + X_n]}{(r-np)^2}.$$

Tačiau $\mathbf{D}[X_j] = p(1-p)$, o atsitiktiniai dydžiai yra nepriklausomi, todėl

$$P(\text{I rūšies klaida}|\omega_i) \le \frac{np(1-p)}{\sigma^2 n} = \delta(n,r). \tag{39}$$

Taigi

$$p^*_{vid}(\mathbf{C}|r) \le \delta(n,r) + \frac{1}{N}\sum_{\omega_i \in \mathbf{C}} P(\text{II rūšies klaida}|\omega_i). \tag{40}$$

Antros rūšies klaida įvyksta, kai priimamas toks žodis $\mathbf{d}$, kad rutuliui $B(\mathbf{d}, r)$ priklauso dar bent vienas kodo žodis, skirtingas nuo siųstojo. Tad tikimybę $P(\text{II rūšies klaida}|\omega_i)$ gausime sumuodami tikimybes $p(\mathbf{d}|\omega_i)$; čia $\mathbf{d}$ tenkina ką tik aptartą sąlygą. Šią tikimybę patogu reikšti panaudojus indikatorius:

$$\chi(\mathbf{d}, \omega) = \begin{cases} 1, & \text{jei } \omega \in B(\mathbf{d}, r), \\ 0, & \text{jei } \omega \notin B(\mathbf{d}, r). \end{cases}$$

Dabar

$$P(\text{II rūšies klaida}|\omega_i) = \sum_{\substack{\mathbf{d}\in\mathcal{A}^n \\ \omega_i \in B(\mathbf{d},r) \\ |B(\mathbf{d},r)\cap\mathbf{C}|\ge 2}} p(\mathbf{d}|\omega_i) \le \sum_{\mathbf{d}\in\mathcal{A}^n} p(\mathbf{d}|\omega_i) \sum_{j\ne i} \chi(\mathbf{d}, \omega_j).$$

Pasinaudoję šiuo įverčiu nelygybėje (40), gausime

$$p^*_{vid}(\mathbf{C}|r) \le \delta(n,r) + \frac{1}{N}\sum_{\mathbf{d}\in\mathcal{A}^n} \sum_{\substack{\omega_i,\omega_j \in \mathbf{C} \\ j\ne i}} p(\mathbf{d}|\omega_i)\chi(\mathbf{d}, \omega_j).$$

Verta pasvarstyti, ką gavome. Tikimybė $p^*_{vid}(\mathbf{C}|r)$ priklauso nuo kodo $\mathbf{C}$, kurį parenkame atsitiktinai. Taigi ši tikimybė yra atsitiktinis dydis. Jį

įvertinome tam tikru reiškiniu; svarbiausias šio reiškinio elementas – suma, kurioje yra tiek dėmenų, kiek yra žodžių $\mathbf{d} \in \mathcal{A}^n$. Kiekvienas sumos dėmuo $p(\mathbf{d}|\omega_i)\chi(\mathbf{d},\omega_j)$ yra dviejų nepriklausomų atsitiktinių dydžių $p(\mathbf{d}|\omega_i)$ ir $\chi(\mathbf{d},\omega_j)$ sandauga. Kadangi nepriklausomų atsitiktinių dydžių sandaugos vidurkis lygus vidurkių sandaugai, tai

$$\mathbf{E}[p(\mathbf{d}|\omega_i)\chi(\mathbf{d},\omega_j)] = \mathbf{E}[p(\mathbf{d}|\omega_i)] \cdot \mathbf{E}[\chi(\mathbf{d},\omega_j)].$$

Dabar jau galime vertinti atsitiktinio dydžio $p^*_{vid}(\mathbf{C}|r)$ vidurkį:

$$\mathbf{E}[p^*_{vid}(\mathbf{C}|r)] \le \delta(n,r) + \frac{1}{N} \sum_{\mathbf{d}\in\mathcal{A}^n} \sum_{\substack{\omega_i,\omega_j\in\mathbf{C}\\ j\neq i}} \mathbf{E}[p(\mathbf{d}|\omega_i)]\mathbf{E}[\chi(\mathbf{d},\omega_j)].$$

Prisiminkime, kad, sudarant kodą, ω_i galima suvokti kaip atsitiktinius dydžius, su vienodomis tikimybėmis 2^{-n} įgyjančius visas reikšmes iš $\mathcal{A}^n$. Taigi fiksuotam $\mathbf{d}$

$$\mathbf{E}[p(\mathbf{d}|\omega_i)] = 2^{-n} \sum_{\omega\in\mathcal{A}^n} p(\mathbf{d}|\omega),$$

$$\mathbf{E}[\chi(\mathbf{d},\omega_j)] = \sum_{\omega\in B(\mathbf{d},r)} 2^{-n} = 2^{-n}|B(\mathbf{d},r)|.$$

Elementų skaičius rutulyje $B(\mathbf{d},r)$ nepriklauso nuo žodžio $\mathbf{d}$, šį skaičių pažymėjome $V_n(r)$. Taigi

$$\mathbf{E}[p^*_{vid}(\mathbf{C}|r)] \le \delta(n,r) + \frac{V_n(r)}{4^n N} \sum_{\substack{1\le i,j\le N\\ j\neq i}} \sum_{\mathbf{d},\omega\in\mathcal{A}^n} p(\mathbf{d}|\omega).$$

Kadangi

$$\sum_{\mathbf{d},\omega\in\mathcal{A}^n} p(\mathbf{d}|\omega) = \sum_{\omega\in\mathcal{A}^n} \sum_{\mathbf{d}\in\mathcal{A}^n} p(\mathbf{d}|\omega) = \sum_{\omega\in\mathcal{A}^n} 1 = 2^n$$

ir yra $N(N-1)$ porų $\langle i,j\rangle, j \neq i$, tai

$$\mathbf{E}[p^*_{vid}(\mathbf{C}|r)] \le \delta(n,r) + \frac{V_n(r)(N-1)}{2^n}. \tag{41}$$

Kadangi $r = (p+\sigma)n < n/2$, tai, pasirėmę 27 teoremoje pateiktu $V_n(r)$ įverčiu, iš (41) gausime

$$\mathbf{E}[p^*_{vid}(\mathbf{C}|r)] \le \delta(n,r) + 2^{n(H(p+\sigma)-1)}(N-1). \tag{42}$$

Fiksuokime $0 < R < \mathcal{C}(p)$ ir $\epsilon > 0$, čia $\mathcal{C}(p) = 1 - H(p)$ yra kanalo talpa. Jeigu parinksime $N = \left[2^{nR}\right] + 1$, tai bet kurio (n,N) kodo $\mathbf{C}$ koeficientas

$$R(\mathbf{C}) = \frac{\log_2 N}{n} \ge R.$$

Kadangi $N-1<2^{nR}$, tai (42) galime perrašyti taip:

$$\mathbf{E}[p^*_{vid}(\mathbf{C}|r)]\leq\delta(n,r)+2^{-n(1-R-H(p+\sigma))}.$$

Pasinaudoję $\mathcal{C}(p)$ ir $\delta(n,r)$ išraiška iš (39), gausime

$$\mathbf{E}[p^*_{vid}(\mathbf{C}|r)]\leq\frac{p(1-p)}{\sigma^2 n}+2^{-n(\mathcal{C}(p)-R+H(p)-H(p+\sigma))}.$$

Kadangi $H(\lambda)$ yra tolydi funkcija, tai dydį σ galime parinkti tokį mažą, kad būtų

$$\mathcal{C}(p)-R+H(p)-H(p+\sigma)>0.$$

Tada imant pakankamai didelį n, galios

$$\mathbf{E}[p^*_{vid}(\mathbf{C}|r)]\leq\epsilon.$$

Pagaliau tikslas pasiektas: įrodėme, kad klaidos vidurkis neviršija iš anksto parinkto skaičiaus ϵ. Tai šios nuostabios, bet sudėtingos teoremos įrodymo pabaiga.

4.8. Kai nepaisoma saugaus greičio

Jeigu vairuotojas viršijo leistiną greitį, tikimybė, kad jis laimingai pasieks kelionės tikslą, nėra lygi nuliui. O jeigu viršijame tą duomenų perdavimo greičio ribą, kurią nustato kanalo talpa? Nėra jokių išlygų: tikimybė, kad duomenys bus perduoti klaidingai, bus didelė.

Shannono teorema tvirtina: jei nesistengsime informacijos perduoti pernelyg greitai, t. y. jei perdavimo koeficientas neviršys kanalo talpos, tai galime perduoti informaciją labai patikimai. Tačiau ar neįmanomas ir greitas, ir patikimas perdavimas? C. Shannonas atsakė ir į šį klausimą: jeigu perdavimas yra patikimas, tai naudojamo kodo koeficientas negali būti didesnis už kanalo talpą.

Pabandykime suvokti, kodėl taip yra, nesirūpindami, kad samprotavimai būtų matematikos požiūriu itin tikslūs.

Tarkime, naudojant kodą $\mathbf{C}\subset\{0,1\}^n$ ir tam tikrą dekodavimo taisyklę, perdavimas yra patikimas, t. y. klaidingo dekodavimo tikimybė yra maža. Tegu p yra simbolio iškraipymo tikimybė, o $N=|\mathbf{C}|$ yra kodo žodžių skaičius. Patikimas perdavimas reiškia, kad dažniausiai pasitaikančiais atvejais įvykusios klaidos yra ištaisomos. Kadangi žodį sudaro n bitų, tai dažniausiai pasitaikantis klaidų skaičius yra $k\approx np$. Taigi esant patikimam perdavimui, toks klaidų skaičius visada ištaisomas. Jeigu siunčiamas kodo žodis $\mathbf{c}$ ir jame įvyksta k klaidų, tai gavėjas gali gauti bet kokį žodį iš C_n^k žodžių turinčios aibės. Visi šios aibės žodžiai dekoduojami tuo pačiu kodo žodžiu $\mathbf{c}$. Taigi

$$N\leq\frac{2^n}{C_n^k}.$$

Tada kodo koeficientui teisingas įvertis

$$R = \frac{\log_2 N}{n} = 1 - \frac{\log_2 C_n^k}{n}.$$

Dydžio

$$C_n^k = \frac{n!}{k!(n-k)!}$$

reikšmę vertinsime naudodamiesi Stirlingo formule

$$x! \approx x^x \mathrm{e}^{-x}\sqrt{2\pi x},$$

kuri tuo tikslesnė, kuo didesnis x. Kiek paskaičiavę, gautume

$$\ln C_n^k = \ln n! - \ln k! - \ln (n-k)! \approx (n-k)\ln\frac{n}{n-k} + k\ln\frac{n}{k}.$$

Tačiau $k \approx np$, todėl

$$\ln C_n^k \approx n\Big((1-p)\ln\frac{1}{1-p} + p\ln\frac{1}{p}\Big), \quad \frac{\log_2 C_n^k}{n} \approx (1-p)\ln\frac{1}{1-p} + p\ln\frac{1}{p}$$

ir $R < \mathcal{C}$, čia $\mathcal{C}$ yra kanalo talpa.

O dabar žvilgtelkime, kaip atrodo tikslūs teiginiai apie informacijos perdavimą, kai greitis viršija kanalo talpą. Priminsime, kad vidutinę klaidingo dekodavimo tikimybę, kai naudojamas kodas ir atitinkama dekodavimo taisyklė, apibrėžėme taip:

$$p^*_{vid}(\mathbf{C}) = \frac{1}{|\mathbf{C}|}\sum_{\mathbf{c}\in\mathbf{C}} P(klaida|\mathbf{c}).$$

29 teorema. *Tegu diskretaus be atminties su dvinare šaltinio abėcėle kanalo talpa yra $\mathcal{C}$, o $\mathbf{C}$ – koks nors (n, N) kodas, kuriam turime sudarę dekodavimo taisyklę.*

Kiekvienam $R > \mathcal{C}$ egzistuoja $\delta(R) > 0$, kad iš $R(\mathbf{C}) > R$ išplaukia

$$p^*_{vid}(\mathbf{C}) > \delta(R).$$

Taigi teorema teigia, kad nors kiek viršijus „saugų“ perdavimo greitį, atsiranda perdavimo „kokybės“ riba, kurios nepavyks sumažinti.

Įrodinėdami Shannono teoremą, matėme, kad kodų, kurių koeficientai neviršija kanalo talpos, aibėje patikimesni tie, kurių ilgiai yra dideli. Kodams, kuriais perduodame greičiau, nei leidžia kanalo talpa, yra priešingai – ilgesni kodai yra mažiau patikimi. Tokia šios teoremos tvirtinimo esmė.

30 teorema. *Tegu diskretaus be atminties su dvinare šaltinio abėcėle kanalo talpa yra $\mathcal{C}$, o $\mathbf{C}_n$ yra (n, N) kodų, kuriems turime sudarę dekodavimo taisykles, seka.*

Jeigu $R > \mathcal{C}$ *ir* $R(\mathbf{C}_n) > R$, *tai*

$$p^*_{vid}(\mathbf{C}_n) \to 1,\ n \to \infty.$$

Pirmoji teorema kartais vadinama silpnuoju atvirkštinės Shannono teoremos variantu, antroji teorema – stipriuoju. Pastebėkime, kad nė viena iš jų neišplaukia iš kitos.

5 Pastabos ir nuorodos

„Studijuokite pirmuosius šaltinius“ – šis patarimas nepraranda vertės. Iš tikrųjų visas svarbiausias įžvalgas ir idėjas apie informacijos kiekį ir entropiją rasite klasikiniame C. Shannono darbe [68]. Jame C. Shannonas nagrinėja taip pat ir tolydaus informacijos šaltinio (kai abėcėlė nėra diskreti) entropijos sąvoką ir savybes. Šį klasikinį tekstą paieškoję tikrai surasite Interneto elektroninių tekstų saugyklose.

Informacijos kiekio sąvokai skirtus skyrius rasite kone kiekvienoje knygoje, nagrinėjančioje informacijos kodavimą, pavyzdžiui, [4], [34]. R. M. Gray knyga [34] tinkama norintiems įsigilinti į sudėtingesnius tikimybinius informacijos šaltinių modelius, joje daug vietos skirta Markovo, stacionariesiems, ergodiniams šaltiniams. Šaltinio perduotos informacijos kiekį siejome su netikėtumu, neapibrėžtumu. Informacijos kiekio matą, nagrinėjamą pirmajame skyrelyje, pradėjo naudoti Hartley [41]. Šaltinio entropija – Shannono apibrėžta sąvoka. Tai informacijos teorijos klasika. Tolesnei šių temų raidai skirta G. Kliro knyga „Neapibrėžtumas ir informacija“ [43].

Jeigu susidomėjote informacijos teorija – būtinai pastudijuokite D. MacKay knygą [51]! Tuo labiau, kad iš autoriaus svetainės galite atsisiųsti jos skaitmeninę kopiją. Teorija šioje knygoje dėstoma tikrai įdomiai, pateikiama daug pavyzdžių ir interpretacijų. Rasite ir humoro – jo niekada nebus per daug matematikos ir informatikos knygose.

Ir dar vieną knygą tikrai būtina paminėti [17]. Joje klasikinė informacijos teorija ir taikymai dėstomi išsamiai, matematiškai griežtai, kartu iliustruojant paprastais ir įdomiais pavyzdžiais.

Entropijos sąvoka svarbi ne vien tik informacijos teorijoje; įvairiems šios sąvokos taikymams skirtas elektroninis žurnalas „Entropija“ [24].

Teoremą apie šaltinio kodavimą įrodė C. Shannonas, ji išdėstyta jo darbe [68]. D. Huffmanas savo algoritmą išdėstė vos trijų puslapių straipsnyje [39]. Darbų, kurie vėliau parašyti jį tyrinėjant, gausą sunku ir apžvelgti. Viena svarbiausių šio metodo tobulinimo idėjų – adaptyvumas, t. y. kodo pritaikymas koduojamo srauto struktūrai. Adaptyvių Huffmano kodų metodą išdėstė N. Falleris, R. Gallagheris ir D. Knuthas ([27], [28], [44]).

Aritmetinių kodų idėjos autorius – P. Eliasas, pirmą kartą šių kodų konstrukcija išdėstyta N. Abramsono knygoje [1].

Kodavimo su žodynu metodą sukūrė J. Zivas ir A. Lempelis, pirmoji jų publikacija [80] pasirodė 1977 m. Burrowso-Wheelerio metodas paskelbtas 1994 [13].

Šioje knygoje apžvelgtos tik bendros šaltinio duomenų srautų efektyvaus kodavimo (spaudimo) idėjos. Praktikoje visada atsižvelgiama į koduojamo srauto savybes. Vieni metodai naudojami spaudžiant tekstinius duomenis, kiti – grafikos, audio- ir videofailus. Taigi duomenų spaudimo teorija – labai plati sritis. Norint įsigilinti į bendrų kodavimo idėjų praktinio taikymo ypatybes, teks pastudijuoti šiai teorijai skirtas knygas. Viena išsamiausių – D. Salomono knyga [67], tai tikras duomenų spaudimo algoritmų sąvadas. Visi svarbiausi metodai ir praktinio jų taikymo detalės dėstomos ir [37].

Internete galima rasti duomenų spaudimo teorijai skirtų tinklalapių, pavyzdžiui, [23].

Pagrindinės šio skyriaus teoremos paskelbtos jau minėtame Shannono darbe [68]. C. Shannono darbų reikšmė moderniųjų ryšių teorijai tolygi Niutono darbų reikšmei mechanikai. Apie Shannono idėjų vaidmenį žr. R. Gallagherio apžvalginiame straipsnyje [29], taip pat – informacijos teorijos ir kodavimo teorijos penkiasdešimtmečiui skirtose apžvalgose [78], [14].

Įvairius Shannono teoremų variantus galime rasti kone kiekvienoje išsamesniam informacijos teorijos dėstymui skirtoje knygoje, pavyzdžiui, [50], [65], [17].

Kodavimo teorija

6 Hammingo geometrija

6.1. Sėkmė aplanko pasiruošusius

Šį L. Pastero posakį mėgo Richardas Hammingas (1915 – 1998), matematikas, kurio idėjos padarė didelę įtaką moderniųjų telekomunikacijų raidai. Jis parašė devynias matematikos bei informatikos knygas, padarė daug atradimų, tačiau tikriausiai labiausiai žinomas iš jų visų – Hammingo kodas.

Net trisdešimt metų (1946–1976) R. Hammingas dirbo įžymiajame mokslinių tyrimų ir jų rezultatų taikymo ryšių praktikoje centre – Bello laboratorijoje.[4] Šioje laboratorijoje dirbo ir C. Shannonas. Jau žinome, kad jis matematiškai įrodė, kad net ir nepatikimais kanalais įmanomas patikimas ryšys. Tačiau recepto, kaip to pasiekti, nedavė.

O R. Hammingui pavyko surasti labai praktišką sprendimą. Jis pasiūlė kodo, taisančio vieną klaidą, konstrukciją ir šitaip padėjo kertinį šiuolaikinės algebrinės kodavimo teorijos akmenį.

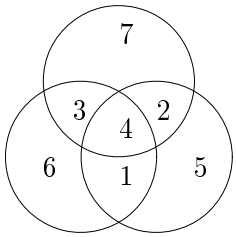
Savo straipsnyje „Error Detecting and Error Correcting Codes“, išspausdintame „The Bell System Technical Journal“ 1950 metų antrajame numeryje jis rašo:

Imtis šiame straipsnyje dėstomų uždavinių autorių paskatino ilgalaikis skaičiavimo mašinų, kurios privalo atlikti daugybę operacijų ir nepadaryti nei vienos klaidos, darbo stebėjimas.

O tikrovė buvo tokia: skaičiavimo mašinos galėjo skaityti duomenis tik iš perforacinių juostų, kurias pateikdavo operatorius; dėl vienintelio neteisingai perskaityto ar perforuoto simbolio skaičiavimo mašina tiesiog atmesdavo programą... Atėjęs pirmadienį į darbą, Hammingas, matyt, neretai vietoj norimų rezultatų gaudavo tik pranešimą apie įvedimo klaidas. Kodėl mašina, sugebanti aptikti klaidas, negalėtų jų ir išsitaisyti? R. Hammingas dažnai pabrėždavo, kad atradimus padaro tik tie, kurie turi tam drąsos.

[4] A. G. Bellas (1847-1922) – telefono išradėjas. „Mr. Watson. Come here! I want you¡' – šie A. G. Bello žodžiai asistentui – pirmieji žodžiai žmonijos istorijoje, perduoti telefonu. Jo vardu pavadintos laboratorijos įsteigtos 1925 metais. Ilgą laiką šios laboratorijos buvo pirmaujantis ryšio technikos mokslinių tyrimų centras. Net šeši šio centro darbuotojai tapo Nobelio premijos laureatais.

Kiekvieną ketvertą perduodamų bitų jis papildė trimis papildomais. Vėliau R. J. McEliece pastebėjo, kad Hammingo idėją paprasta paaiškinti naudojantis Veno diagramomis. Pasinaudokime šiomis diagramomis ir mes. Tarkime, kanalu reikia perduoti keturis dvejetainius simbolius (bitus).



Žodis, sudarytas iš bitų $1-4$, *papildomas dar trimis taip, kad bitų, kurių numeriai užrašyti tame pačiame skritulyje, suma būtų lyginė.*

Srityse 1, 2, 3, 4 užrašykime keturis bitus (simbolius 0 arba 1), kuriuos reikia perduoti. Likusiose srityse užrašykime simbolius taip, kad, sumuodami bitus, kurių numeriai užrašyti tame pačiame skritulyje, gautume lyginį skaičių. Dabar pasiųskime į kanalą šį septynių bitų rinkinį. Kitame kanalo gale gavėją pasieks galbūt kitas rinkinys, kai kurie simboliai gali būti iškreipti. Gavėjas gali patikrinti, ar sumos pagal visus tris skritulius yra lyginės. Jeigu nors viena nelyginė, tai įvyko iškraipymų. Nesunku įsitikinti, kad jeigu iškreiptas tik vienas simbolis, tai gavėjas gali atkurti jį nesuklysdamas. Jeigu iškraipymų daugiau, atkurdami simbolius, galime ir suklysti.

Tegu perdavimo kanalas yra dvejetainis ir simetrinis, o simbolio iškraipymo tikimybė lygi p. Palyginkime tikimybes, kad keturi simboliai bus perduoti teisingai, kai informacijos nekoduojame ir kai ją koduojame Hammingo kodu. Tarkime, kanalas kiekvieną simbolį iškreipia su tikimybe p $(0 < p < 1)$, be to, visi simboliai iškraipomi nepriklausomai. Tada tikimybės, kad keturi simboliai bus perduoti teisingai, kai informacija nekoduojama ir koduojama Hammingo kodu lygios:

$$P_0 = q^4 \quad \text{ir} \quad P_1 = q^7 + 7q^6p, \quad q = 1 - p.$$

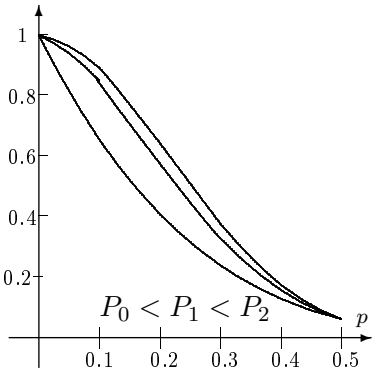
Kadangi

$$\frac{P_1}{P_0} = q^2(1 + 6p) = 1 + p(6p^2 - 11p + 4),$$

tai $P_1 > P_0$, kai $6p^2 - 11p + 4 > 0$. Nesunku įsitikinti, kad ši nelygybė teisinga, kai $0 < p < 2/3$.

O jeigu tiems keturiems simboliams perduoti naudotume pakartojimo kodą, pavyzdžiui, perduodamą simbolį pakartotume tris kartus? Tada vietoj keturių simbolių tektų siųsti dvylika – Hammingo kodas yra „greitesnis“. Tikimybė, kad, naudojant kartojimo kodą, visi keturi simboliai bus perduoti teisingai, lygi

$$P_2 = (q^3 + 3pq^2)^4.$$



Brėžinyje pavaizduoti tikimybių P_0, P_1 ir P_2 grafikai. Iš brėžinio matyti, $P_0 < P_1 < P_2$. Tris kartus kartojant simbolį, keturi bitai perduodami teisingai su didesne tikimybe nei koduojant Hammingo kodu. Tačiau perdavimo sąnaudos išauga trigubai!

Koduojant Hammingo kodu, keturi perduodamos informacijos bitai papildomi dar trimis, kurių paskirtis – suteikti galimybę ištaisyti vieno simbolio perdavimo klaidą. Jeigu būtų neteisingai perduoti, pavyzdžiui, du simboliai, gavėjas pastebėtų, jog įvyko klaida, ir bandytų pagal nurodytą būdą ištaisyti tą klaidą. Tačiau manydamas, kad ją taiso, iš tikrųjų neištaisytų!

Hammingas pastebėjo, kad šį kodą galima šiek tiek pagerinti. Jeigu prie septynių kodo žodžių bitų pridėsime dar vieną – aštuntąjį taip, kad visų simbolių suma būtų lyginė, tai galėsime atpažinti, kada įvyko viena klaida, o kada – dvi. Pirmuoju atveju klaidą galima ištaisyti, o antruoju žinosime, kad taisyti neverta, t. y. bent jau neapgaudinėsime savęs.

Panagrinėkime pavyzdį. Tarkime, reikia pasiųsti žodį $\mathbf{x} = 1111$; naudodamiesi Hammingo kodu, papildytume jį trimis simboliais ir perduotume $\mathbf{c} = 1111111$. Jeigu būtų iškreipti pirmieji du bitai, tai gavėjas taisytų žodį taip:

$$0011111 \to 0001111,$$

taigi įveltų dar vieną klaidą. Jeigu naudotume pailgintą Hammingo kodą, tai į kanalą siųstume 11111111. Gavėjas gautų 00111111 ir, nustatęs, kad simbolių suma žodyje lyginė, padarytų išvadą: arba iškraipymų nebuvo, arba jų skaičius buvo lyginis. Tačiau, naudodamiesi tais pačiais skrituliais, galime įsitikinti, kad klaidų būta. Tektų pripažinti, kad jos neištaisomos.

R. Hammingo kodas – labai paprasta, praktiška ir efektyvi konstrukcija. Tačiau su ja mes beveik nepriartėjame prie tos patikimo ryšio galimybės, kurias mums žada Shannono teorema. Kodavimo teorijos tikslas – kurti naujus patogius naudoti kodus, kurie vis geriau realizuotų Shannono pažadą. Jis atrodo kone kaip nepasiekiamas horizontas. Nors... Visai neseniai sugalvoti nauji kodai, beveik pasiekiantys Shannono ribą... Bet kol kas mums dar toli iki jų.

6.2. Hammingo atstumas ir kodai

Jeigu yra būdas skaičiuoti atstumą, yra ir geometrija. Šiame skyrelyje geometrijos terminais nusakomos kodų savybės.

Prisiminkime žymėjimus: $\mathcal{A}_q$ žymi abėcėlę, turinčią q simbolių, $|\mathcal{A}| = q$. Kol kas mums nesvarbu nei kaip užrašomi tie simboliai, nei kokie vidiniai ryšiai sieja tuos simbolius. Abėcėlė – aibė ir tiek.

Pati svarbiausia – dvinarė abėcėlė $\mathcal{A}_2$. Apibrėžtumo dėlei tarkime, kad šios abėcėlės simboliai – nulis ir vienas, t. y. $\mathcal{A}_2 = \{0, 1\}$.

Nagrinėsime $\mathcal{A}_q$ abėcėlės n simbolių ilgio žodžių aibę

$$\mathcal{A}_q^n = \mathcal{A}_q \times \mathcal{A}_q \times \ldots \times \mathcal{A}_q.$$

Apibrėžę atstumą tarp dviejų šios aibės žodžių „sukursime" joje tam tikrą geometriją.

30 apibrėžimas. *Tegu* $\mathbf{x} = x_1 \ldots x_n$, $\mathbf{y} = y_1 \ldots y_n$ *yra du aibės* $\mathcal{A}_q^n$ *žodžiai. Hammingo atstumu tarp* $\mathbf{x}$, $\mathbf{y}$ *vadinsime dydį*

$$h(\mathbf{x}, \mathbf{y}) = \sum_{\substack{i=1,\ldots,n \\ x_i \neq y_i}} 1.$$

Hammingo atstumas turi visas pagrindines atstumo savybes, kitaip tariant – Hammingo atstumas yra metrika.

31 teorema. *Hamingo atstumas aibėje* $\mathcal{A}_q^n$ *turi šias savybes:*

- $h(\mathbf{x}, \mathbf{x}) = 0, \quad \mathbf{x} \in \mathcal{A}_q^n$;
- $h(\mathbf{x}, \mathbf{y}) = h(\mathbf{y}, \mathbf{x}), \quad \mathbf{x}, \mathbf{y} \in \mathcal{A}_q^n$;
- $h(\mathbf{x}, \mathbf{y}) \leq h(\mathbf{x}, \mathbf{z}) + h(\mathbf{y}, \mathbf{z}), \quad \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathcal{A}_q^n$.

Dar kartą prisiminkime kodo apibrėžimą.

31 apibrėžimas. *(n, N) kodu iš abėcėlės $\mathcal{A}_q$ žodžių vadinamas bet koks poaibis $\mathbf{C} \subset \mathcal{A}_q^n$, čia $|\mathbf{C}| = N$.*

Kodavimo taisyklė turi nusakyti, kaip koduojamos informacijos žodžiui priskirti naudojamo kodo žodį. Dekodavimą apskritai sudaro du žingsniai: pirmiausia pagal gautą (galbūt iškraipytą) žodį reikia nustatyti, koks kodo žodis buvo siųstas, paskui pagal kodo žodį atkurti pradinį šį žodį atitinkantį simbolių rinkinį. Kol kas mums rūpės tik pirmasis žingsnis, t. y. dekodavimo taisyklė $f : \mathcal{A}_q^n \to \mathbf{C}$.

32 apibrėžimas. *Dekodavimo taisyklę* $f : \mathcal{A}_q^n \to \mathbf{C}$ *vadinsime minimalaus atstumo taisykle, jei su kiekvienu* $\mathbf{d} \in \mathcal{A}_q^n$

$$h(\mathbf{d}, f(\mathbf{d})) = \min_{\mathbf{c} \in \mathbf{C}} h(\mathbf{d}, \mathbf{c}). \tag{43}$$

Taigi minimalaus atstumo dekodavimo taisyklė remiasi labai paprasta idėja – jeigu gautas ne kodo žodis, tai manoma, kad buvo siųstas arčiausiai gautojo esantis kodo žodis. Jeigu kuriam nors $\mathbf{d}$ minimumas (43) lygybėje pasiekiamas, kai $\mathbf{c} = \mathbf{c}_1$ ir $\mathbf{c} = \mathbf{c}_2, \mathbf{c}_1 \neq \mathbf{c}_2$, tai minimalaus atstumo taisyklė apibrėžiama nevienareikšmiškai. Jei minimumas pasiekiamas su dviem ar daugiau skirtingų žodžių, tokį atvejį vadinsime **ryšiu**. Ryšio atveju galime tiesiog fiksuoti klaidą (*soft decision*) arba dekoduoti vienu iš galimų būdų (*hard decision*), pavyzdžiui, vieną iš arčiausių žodžių pasirinkę atsitiktinai.

Kuo atokiau vienas nuo kito yra išsidėstę žodžių aibėje kodo žodžiai, tuo patikimesnė yra minimalaus atstumo dekodavimo taisyklė. Kodo žodžių „išsidėstymui“ nusakyti vartosime minimalaus kodo atstumo sąvoką.

33 apibrėžimas. *Kodo* $\mathbf{C}$ *minimaliu atstumu vadinsime dydį*

$$d(\mathbf{C}) = \min_{\substack{\mathbf{c},\mathbf{d}\in\mathbf{C} \\ \mathbf{c}\neq\mathbf{d}}} h(\mathbf{d},\mathbf{c}).$$

Jei (n, N) *kodo* $\mathbf{C}$ *minimalus atstumas yra* d*, tai kodą vadinsime* (n, N, d) *kodu.*

Šis minimalaus kodo atstumo apibrėžimas netinka tuo atveju, kai kodą sudaro vienas žodis. Apibrėžtumo dėlei tarkime, kad tokiu atveju $d = n+1$.

Kaip gavėjas gali nustatyti, kad, perduodant kodo žodį, įvyko klaidos? Kadangi jis žino, kokie žodžiai galėjo būti siunčiami, bet kuris iš kanalo gautas, bet kodui nepriklausantis žodis yra kartu signalas apie įvykusius iškraipymus.

34 apibrėžimas. *Kodą* $\mathbf{C}$ *vadinsime* t *klaidų randančiu kodu, jei, bet kuriame kodo žodyje įvykus* $m, m \leq t$*, iškraipymų, gautas rezultatas* $\mathbf{d}$ *jau nebėra kodo žodis, t. y.* $\mathbf{d} \notin \mathbf{C}$.

t *klaidų randantį kodą vadinsime tiksliai* t *klaidų randančiu kodu, jei jis nėra* $t+1$ *klaidų randantis kodas.*

32 teorema. (n, N, d) *kodas* $\mathbf{C}$ *yra tiksliai* t *klaidų randantis kodas tada ir tik tada, kai* $d = t + 1$.

Įsitikinti, kad šis teiginys teisingas – paprasta. Iš tiesų, jeigu bet kokiame kodo žodyje $\mathbf{x} \in \mathbf{C}$ pakeisime $d-1$ simbolį, naujasis žodis nepriklausys kodui. Vadinasi, kodas aptinka $d-1$ klaidą. Kita vertus, egzistuoja du žodžiai $\mathbf{x}, \mathbf{y} \in \mathbf{C}$, kad $h(\mathbf{x}, \mathbf{y}) = d$. Taigi žodyje $\mathbf{x}$ galime pakeisti lygiai d simbolių taip, kad gautume žodį $\mathbf{y}$. Jeigu tai „atliks“ perdavimo kanalas, klaida liks nepastebėta. Taigi kodas jau nebe visada suranda d įvykusių klaidų.

Analogiškai apibrėšime klaidas taisančius kodus. Patys kodai, aišku, nei randa klaidas, nei jas taiso. Klaidas taisome mes, naudodamiesi minimalaus atstumo dekodavimo taisykle.

35 apibrėžimas. *Kodą* $\mathbf{C}$ *vadinsime* t *klaidų taisančiu kodu, jei, siunčiamame žodyje įvykus* $m, m \leq t$*, iškraipymų ir dekoduojant pagal minimalaus atstumo taisyklę, visada bus dekoduojama teisingai. Tokį kodą*

vadinsime tiksliai t klaidų taisančiu kodu, jeigu jis ne visada taiso $t+1$ klaidų.

Kiek klaidų taiso kodas $\mathbf{C}$, lemia minimalus jo atstumas.

33 teorema. *Kodas $\mathbf{C}$ yra tiksliai t klaidų taisantis kodas tada ir tik tada, kai $d(\mathbf{C}) = 2t+1$ arba $d(\mathbf{C}) = 2t+2$.*

Įrodymas. Tegu $d(\mathbf{C}) = 2t+1$, $\mathbf{c}$ yra siųstas žodis, $\mathbf{d}$ – gautas, be to, $h(\mathbf{c},\mathbf{d}) \leq t$. Jeigu būtų kitas kodo žodis $\mathbf{c}'$, kad $h(\mathbf{c}',\mathbf{d}) \leq t$, tuomet, panaudoję trikampio nelygybę, gautume

$$h(\mathbf{c},\mathbf{c}') \leq h(\mathbf{c},\mathbf{d}) + h(\mathbf{c}',\mathbf{d}) \leq 2t.$$

Tačiau šitaip negali būti, nes

$$h(\mathbf{c},\mathbf{c}') \geq d = 2t+1.$$

Taigi kodas, kurio minimalus atstumas $d(\mathbf{C}) = 2t+1$, taiso t klaidų. Nesunku įsitikinti, kad $t+1$-ą klaidų jis ištaiso ne visuomet.

Jei $d(\mathbf{C}) = 2t+2$, tai toks kodas irgi taiso t klaidų. Yra du kodo žodžiai, kuriems

$$h(\mathbf{c},\mathbf{c}') = 2(t+1).$$

Jeigu žodyje $\mathbf{c}$ pusę tų simbolių, kuriais jis skiriasi nuo $\mathbf{c}'$, pakeisime prilygindami juos atitinkamiems $\mathbf{c}'$ simboliams, gausime naują žodį $\mathbf{d}$, kad

$$h(\mathbf{c},\mathbf{d}) = h(\mathbf{d},\mathbf{c}') = t+1.$$

Jeigu tokį „pakeitimą“ atliks kanalas, gavėjas turės dvi lygiavertes dekodavimo galimybes, t. y. nebūtinai dekoduos teisingai. Taigi kodas su minimaliu atstumu $d(\mathbf{C}) = 2t+2$ taiso tiksliai t klaidų.

Tarkime dabar, kad kodas $\mathbf{C}$ taiso tiksliai t klaidų. Jeigu būtų $d(\mathbf{C}) \leq 2t$, tai, įvykus t klaidų, jos ne visada būtų ištaisomos. Taigi $d(\mathbf{C}) > 2t$. Jeigu būtų $d(\mathbf{C}) = 2(t+1)+1 = 2t+3$, tai, pasirėmę pirmos dalies samprotavimais, gautume, kad kodas taiso $t+1$ klaidų. Taigi lieka atvejai $d(\mathbf{C}) = 2t+1$ arba $d(\mathbf{C}) = 2t+2$.

Išvada. *Bet koks (n, N, d) kodas taiso tiksliai*

$$\left[\frac{d-1}{2}\right]$$

klaidų.

Pavyzdys. Prisiminkime Hammingo kodą iš dvinarės abėcėlės žodžių, kurį apibrėžėme pasinaudoję diagramomis. Jo žodžių ilgis $n = 7$, žodžių skaičius $N = 16$. Šis kodas visada ištaiso vieną klaidą, taigi $d \geq 3$. Tačiau nesunku surasti du kodo žodžius, pavyzdžiui, $\mathbf{x} = 1000110$ ir $\mathbf{y} = 0100101$, kad $h(\mathbf{x},\mathbf{y}) = 3$. Taigi Hammingo kodo parametrai yra $(7, 16, 3)$.

6.3. Tobulieji kodai

Tobulumas – nebūtinai praktiška savybė. Tobulieji kodai taip pat nėra patys geriausi klaidų taisymo požiūriu. Tiesiog jų žodžiai itin taisyklingai išsidėstę žodžių aibėje.

Jeigu jau yra atstumas, geometrija, turi būti ir rutuliai... Dvinarės abėcėlės žodžių aibės rutuliais jau naudojomės nagrinėdami Shannono teoremą apie kodus.

36 apibrėžimas. *Žodžių aibės $\mathcal{A}_q^n$ spindulio $r \geq 1$ rutuliu su centru $\mathbf{x} \in \mathcal{A}_q^n$ vadinsime aibę*

$$B_q(\mathbf{x}, r) = \{\mathbf{y} \in \mathcal{A}_q^n : h(\mathbf{x}, \mathbf{y}) \leq r\}.$$

Šios aibės elementų skaičių vadinsime rutulio tūriu.

34 teorema. *Teisinga lygybė*

$$|B_q(\mathbf{x}, r)| = \sum_{0 \leq k \leq r} C_n^k (q-1)^k.$$

Įrodymas. Rutuliui su centru $\mathbf{x}$ ir spinduliu r priklauso tie žodžiai, kurie skiriasi nuo $\mathbf{x}$ ne daugiau kaip r komponentėmis. Visus žodžius, kurie skiriasi nuo $\mathbf{x}$ lygiai k komponentėmis, galime gauti taip: parenkame k indeksų ir pakeičiame žodžio $\mathbf{x}$ komponentes su šiais indeksais kitomis. Tai galime padaryti $C_n^k(q-1)^k$ skirtingais būdais. Susumavę šiuos dydžius, gauname rutulio tūrio formulę.

Kadangi rutulio tūris nepriklauso nuo jo centro, žymėsime

$$V_q(n, r) = |B_q(\mathbf{x}, r)|.$$

37 apibrėžimas. *Tegu $\mathbf{C}$ yra koks nors (n, N) kodas. Didžiausią sveikąjį skaičių t, kuriam*

$$B_q(\mathbf{c}_1, t) \cap B_q(\mathbf{c}_2, t) = \emptyset, \quad \text{jei} \quad \mathbf{c}_1, \mathbf{c}_2 \in \mathbf{C}, \quad \mathbf{c}_1 \neq \mathbf{c}_2,$$

vadinsime kodo $\mathbf{C}$ pakavimo spinduliu. Jį žymėsime $r_p = r_p(\mathbf{C})$.

Taigi rutuliai $B_q(\mathbf{c}, t), \mathbf{c} \in \mathbf{C}, t \leq r_p(\mathbf{C})$, sudaro nesikertančių aibių šeimą; jei $t \geq r_p(\mathbf{C}) + 1$, bent du rutuliai kertasi.

38 apibrėžimas. *Mažiausią sveikąjį skaičių s, tenkinantį sąlygą*

$$\mathcal{A}_q^n \subset \bigcup_{\mathbf{c} \in \mathbf{C}} B_q(\mathbf{c}, s),$$

vadinsime kodo dengimo spinduliu ir žymėsime $r_d = r_d(\mathbf{C})$.

Akivaizdu, kad teisinga nelygybė $r_p(\mathbf{C}) \leq r_d(\mathbf{C})$. Pakavimo spindulio reikšmę lemia tie kodo žodžiai $\mathbf{x}, \mathbf{y} \in \mathbf{C}$, kuriems atstumas $h(\mathbf{x}, \mathbf{y})$ yra minimalus, t. y. $h(\mathbf{x}, \mathbf{y}) = d(\mathbf{C})$. Nesunku įsitikinti, kad

$$r_p(\mathbf{C}) = \left[\frac{d-1}{2}\right];$$

čia $[\cdot]$ žymime sveikąją skaičiaus dalį. Tad pakavimo spindulys lygus maksimaliam klaidų, kurias gali ištaisyti kodas, skaičiui.

Skirtumas $r_d(\mathbf{C}) - r_p(\mathbf{C})$ gali būti didelis. Iš tikrųjų, kodas, turintis nedaug žodžių, kurie yra arti vienas kito, turės mažą pakavimo ir didelį dengimo spindulį.

Tegu, pavyzdžiui, $\mathbf{C} = B_q(\mathbf{c}_0, r) \subset \mathcal{A}_q^n$, čia $r \geq 1$ yra sveikas skaičius. Tokio kodo žodžiams „ankšta“: $r_p(\mathbf{C}) = 0$; jeigu į kodą įtrauktume tik dalį (du ar daugiau) rutulio žodžių, t. y. imtume $\mathbf{C} \subset B_q(\mathbf{c}_0, r)$, tai būtų $r_p(\mathbf{C}) \leq r$. Įvertinkime dengimo spindulį. Imkime kokį nors žodį $\mathbf{x} \in \mathcal{A}_q^n$, kad $h(\mathbf{x}, \mathbf{c}_0) = n$. Jeigu $\mathbf{c}$ yra kitas kodo žodis, tai, panaudoję trikampio taisyklę, gauname

$$h(\mathbf{x}, \mathbf{c}_0) \leq h(\mathbf{x}, \mathbf{c}) + h(\mathbf{c}, \mathbf{c}_0),$$

arba

$$h(\mathbf{x}, \mathbf{c}) \geq h(\mathbf{x}, \mathbf{c}_0) - h(\mathbf{c}, \mathbf{c}_0) \geq n - r,$$

t. y. $r_d(\mathbf{C}) \geq n - r$.

Paprastas pavyzdys: $(n, 2)$ kodui

$$\mathbf{C} = \{000\ldots00, 000\ldots11\}$$

gauname: $r_p = 0, r_d = n - 1$.

39 apibrėžimas. *Kodą* $\mathbf{C}$ *vadinsime tobulu, jei*

$$r_p(\mathbf{C}) = r_d(\mathbf{C}).$$

35 teorema. (n, N, d) *kodas* $\mathbf{C}$ *yra tobulas tada ir tik tada, kai* $d = 2t + 1$ *ir galioja lygybė*

$$NV_q(n, t) = q^n.$$

Įrodymas. Tobulo (n, N, d) kodo $\mathbf{C}$ minimalus atstumas negali būti lyginis; taigi turi būti $d(\mathbf{C}) = 2t + 1, r_p(\mathbf{C}) = t$. Rutuliai $B(\mathbf{c}, t), \mathbf{c} \in \mathbf{C}$, nesikerta bet kokio kodo atveju. Kodas yra tobulas tada ir tik tada, kai

$$\bigcup_{\mathbf{c}\in\mathbf{C}} B(\mathbf{c}, t) = \mathcal{A}_q^n,$$

Tačiau šis sąryšis ekvivalentus lygybei

$$\sum_{\mathbf{c}\in\mathbf{C}} |B(\mathbf{c}, t)| = |\mathcal{A}_q^n| \quad \text{arba} \quad NV_q(n, t) = q^n.$$

Žinome, kad Hammingo kodo parametrai yra tokie: $(n, N, d) = (7, 16, 3)$. Pakavimo spindulys $r_p = 1$, rutulio tūris $V_2(7, 1) = 8$,

$$N \cdot V_2(n, t) = 16 \cdot 8 = 2^7 = 2^n.$$

Taigi Hammingo kodas yra tobulas!

Kita vertus, jeigu natūraliesiems skaičiams q, n, N, t teisinga lygybė

$$NV_q(n, t) = q^n,$$

tai nereiškia, kad kodas su parametrais $(n, N, 2t + 1)$ egzistuoja. Iš tikrųjų tobulųjų kodų yra nedaug.

Panagrinėkime atvejį, kai abėcėlės simbolių skaičius q yra pirminio skaičiaus laipsnis. Šiuo atveju žinomi visi egzistuojančių tobulų kodų parametrų (n, N, d) rinkiniai.

Visų pirma – „trivialūs“ tobulieji kodai:

- kodas, sudarytas iš vieno aibės $\mathcal{A}_q^n$ žodžio, yra tobulas (pakavimo ir dengimo spinduliai yra vienodi);
- kodas sudarytas iš visų aibės $\mathcal{A}_q^n$ žodžių;
- dvejetainis pakartojimo kodas su nelyginio ilgio žodžiais: $00\ldots0, 11\ldots1$.

Tačiau yra ir netrivialių:

- Dvejetainis kodas ($q = 2$) su parametrais $(23, 2^{12}, 7)$ būtų tobulas. Tokie kodai tikrai egzistuoja. Vienas jų – dvejetainis Golay kodas; jį nagrinėsime vėliau.
- Trejetainis kodas ($q = 3$) su parametrais $(11, 3^6, 5)$ irgi būtų tobulas. Šiuos parametrus turi, pavyzdžiui, trejetainis Golay kodas.
- Kodas su parametrais

  $$\left(\frac{q^m - 1}{q - 1}, \frac{q^m - 1}{q - 1} - m, 3\right), \quad m \geq 2,$$

  būtų tobulas. Tokie kodai tikrai egzistuoja – tai Hammingo kodų šeima. Vieną šios šeimos atstovų jau sutikome – dvejetainį Hammingo kodą ($q = 2, m = 3$).

6.4. Bendrieji kodų konstravimo uždaviniai

Būti maksimalistu – teisinga pozicija. Siekdamas daugiau, daugiau ir pasieksi. O ko reiktų siekti konstruojant klaidas taisantį kodą? Kad būtų paprasta juo naudotis: koduoti ir dekoduoti, kad informacijos perdavimas sulėtėtų kiek galima mažiau, kad galėtume ištaisyti daugiau klaidų. Visų norų nesuderinsi. Todėl šiame skyrelyje panagrinėsime tik tokį uždavinį: kokio dydžio galima sudaryti kodą, jeigu iš anksto nurodyta, kiek klaidų jis turi taisyti?

Tarkime, iš abėcėlės $\mathcal{A} = \mathcal{A}_q$ n ilgio žodžių norime sudaryti kodą, kuris taisytų t klaidų. Kadangi kodo taisomų klaidų skaičius yra glaudžiai susijęs su minimaliu kodo atstumu, tarsime, kad ir šis parametras apibrėžtas. Taigi norėtume, kad minimalus kodo atstumas būtų d.

Natūralu ieškoti paties geriausio (n, N, d) kodo; čia n ir d reikšmės iš anksto pasirinktos. Geriausias (n, N, d) kodas bus tas, kuris turės daugiausia žodžių, t. y. didžiausią parametrą N. Žinoma, epitetas „geriausias“ yra sąlyginis. Šis kodas gali būti visai netikęs naudojimosi juo paprastumo požiūriu. Žymėkime:

$$A_q(n, d) = \max\{N : \text{egzistuoja } (n, N, d) \text{ kodas } \mathbf{C} \subset \mathcal{A}_q^n\}.$$

Kodus su parametrais $(n, A_q(n, d), d)$ vadinsime **maksimaliais**. Jie turi tokią savybę: nė vieno n ilgio žodžio negalima pridėti prie tokio kodo, nesumažinant minimalaus kodo atstumo.

Kiek žodžių turi maksimalus kodas? Kitaip tariant, jei dydžių q, n, d reikšmės fiksuotos, kam lygi dydžio $A_q(n, d)$ reikšmė?

Kaip dažnai būna, uždavinį lengva išspręsti su ribinėmis parametrų reikšmėmis.

36 teorema. *Bet kokiems* $q \geq 1, n \geq 1$,

$$A_q(n, 1) = q^n, \quad A_q(n, n) = q.$$

Palyginkime $n - 1$ ir n ilgio žodžių maksimalių kodų dydžius.

37 teorema. *Bet kokiems* $q \geq 1, n \geq 1, d \geq 1$ *teisingos nelygybės*

$$A_q(n-1, d) \leq A_q(n, d) \leq qA_q(n-1, d).$$

Įrodymas. Nelygybė $A_q(n-1, d) \leq A_q(n, d)$, žinoma, akivaizdi.

Tegu $M = A_q(n, d)$, o $\mathbf{C}$ koks nors (n, M, d) kodas. Suskaidykime kodą $\mathbf{C}$ į q nesikertančių klasių

$$\mathbf{C}_a = \{\mathbf{c} : \mathbf{c} \in \mathbf{C}, \mathbf{c} = aa_2 \dots a_n\}, \ a \in \mathcal{A}_q.$$

Suprantama, jog egzistuoja nors vienas a, kad $|\mathbf{C}_a| \geq M/q$. Iš šios klasės "sutrumpintų" žodžių sudarykime naują kodą

$$\mathbf{C}' = \{\mathbf{c}' : \mathbf{c}' \in \mathcal{A}_q^{n-1}, a\mathbf{c}' \in \mathbf{C}_a\}.$$

Kadangi bet kokiems $\mathbf{c}', \mathbf{c}'' \in \mathbf{C}$ Hammingo atstumas

$$h(\mathbf{c}', \mathbf{c}'') = h(a\mathbf{c}', a\mathbf{c}'') \geq d,$$

tai $\mathbf{C}'$ yra $(n-1, |\mathbf{C}'|, d')$ kodas, $d' \geq d$. Tada

$$A_q(n-1, d) \geq |\mathbf{C}'| \geq \frac{M}{q}.$$

Teorema įrodyta.

Derindami abiejų teiginių nelygybes, galime gauti dydžių $A_q(n,d)$ įverčius, kad ir nelabai tikslius. Pavyzdžiui:

$$\begin{aligned} A_q(d,d) &= q, \quad A_q(d+1,d) \le qA_q(d,d) = q^2, \\ A_q(d+2,d) &\le q^3, \ \ldots \ , A_q(n,d) \le q^{n-d+1} \ (n \ge d). \end{aligned}$$

Taigi įrodėme tokį teiginį:

38 teorema. *Bet kokiems* $n \ge 1,\ q > 1,\ n \ge d \ge 1$, *teisingas įvertis*

$$A_q(n,d) \le q^{n-d+1}.$$

Šis įvertis kodavimo teorijoje vadinamas Singletono įverčiu. Jį galime gauti samprotaudami ir kitaip. Tegu $\mathbf{C}$ yra (n,N,d) kodas. Nubraukdami šio kodo žodžių paskutinius $d-1$ simbolių ($n \ge d$), gausime skirtingus $n-d+1$ ilgio žodžius; tokių žodžių yra ne daugiau kaip q^{n-d+1}. Taigi $N \le q^{n-d+1}$. Imdami $N = A_q(n,d)$, gauname Singletono įvertį.

Įvertinkime maksimalaus kodo dydį iš viršaus ir apačios.

39 teorema. *Bet kokiems* $n \ge 1,\ q > 1,\ n \ge d \ge 1$, *teisingas įvertis*

$$A_q(n,d) \le q^n \left(\sum_{k=0}^{t} C_n^k (q-1)^k \right)^{-1}, \quad t = \left[\frac{d-1}{2} \right].$$

Įrodymas. Jei r_p yra (n,N,d) kodo $\mathbf{C}$ pakavimo spindulys, tai rutuliai $B_q(\mathbf{c}, r_p), \mathbf{c} \in \mathbf{C}$, nesikerta. Tada

$$\bigcup_{\mathbf{c}\in\mathbf{C}} B_q(\mathbf{c}, r_p) \subset \mathcal{A}_q^n, \quad \Big| \bigcup_{\mathbf{c}\in\mathbf{C}} B_q(\mathbf{c}, r_p) \Big| = \sum_{\mathbf{c}\in\mathbf{C}} |B_q(\mathbf{c}, r_p)| \le |\mathcal{A}_q^n| = q^n.$$

Kadangi $r_p = [(d-1)/2]$, tai, įstatę rutulio tūrio išraišką iš jau įrodytos teoremos, gauname

$$N \sum_{k=0}^{t} C_n^k (q-1)^k \le q^n.$$

Teorema įrodyta. Įrodytus įverčius vadinsime Hammingo įverčiais.

40 teorema. *Bet kokiam* $n \ge 1,\ q > 1,\ d \ge 1$,

$$A_q(n,d) \ge q^n \left(\sum_{k=0}^{d-1} C_n^k (q-1)^k \right)^{-1}. \tag{44}$$

Ši nelygybė vadinama Gilberto-Varšamovo įverčiu.

Įrodymas. Tegu (n,N,d) kodas $\mathbf{C}$ yra maksimalus, t. y. $N = A_q(n,d)$. Tada šio kodo negalima papildyti nei vienu žodžiu, nesumažinant minimalaus

atstumo d. Todėl kiekvienam $\mathbf{x} \in \mathcal{A}_q^n$ egzistuoja $\mathbf{c} \in \mathbf{C}$, kad $h(\mathbf{c}, \mathbf{x}) \leq d-1$. Taigi rutuliai $B_q(\mathbf{c}, d-1)$ padengia visą aibę $\mathcal{A}_q^n$:

$$\bigcup_{\mathbf{c}\in\mathbf{C}} B_q(\mathbf{c}, d-1) \supset \mathcal{A}_q^n.$$

Iš šio sąryšio gauname

$$NV_q(n, d-1) = A_q(n, d)V_q(n, d-1) \geq q^n.$$

Pasinaudoję rutulio tūrio išraiška, gauname (44) nelygybę.

Įrodėme tokius įverčius:

$$\frac{q^n}{V_q(n, d-1} \leq A_q(n, d) \leq \frac{q^n}{V_q(n, t}.$$

Jie gauti samprotaujant labai paprastai. Neverta tikėtis, kad jie bus labai tikslūs. Lentelėje pateiktos įverčių reikšmės, kai $q = 2, d = 3$.

$n =$	3	4	5	6	7	8	9	10	11	12
$2^n =$	8	16	32	64	128	256	512	1024	2048	4096
$\left[2^n/V_2(n,2)\right] + 1 =$	2	2	3	3	5	7	12	19	31	52
$\left[2^n/V_2(n,1)\right] =$	2	3	5	9	16	28	51	93	170	315
$2^{n-2} =$	2	4	8	16	32	64	128	256	512	1024

Lentelėje pateiktos dydžio $A_2(n, 3)$ įverčių iš apačios ir viršaus reikšmės. Paskutinėje eilutėje išrašyti Singletono įverčiai. Matome, kad jie yra itin netikslus. Tačiau Singletono įverčiai praverčia, kai d reikšmės yra didelės.

Panagrinėkime kodo sudarymo uždavinį šiek tiek kitaip. Tarkime, abėcėlės $\mathcal{A}_q$ simbolių srautą ketiname skaidyti k ilgio žodžiais, o pastaruosius keisti (t. y. koduoti) kodo $\mathbf{C} \subset \mathcal{A}_q^n$, $n > k$, žodžiais. Tada galime įsivaizduoti, kad n ilgio kodo žodis „neša“ k informacijos simbolių, o $r = n-k$ papildomų žodžio simbolių yra skirti klaidoms taisyti. Kiek tų papildomų simbolių reikia pridėti, kad galėtume ištaisyti t klaidų? Jeigu naudosime kodą su parametrais (n, N, d), $d = 2t+1$, tai norimą klaidų kiekį tikrai ištaisysime. Tarkime, pasirinkome maksimalų kodą, t. y. $N = A_q(n, d)$. Kodo žodžių turi užtekti visiems k ilgio blokams koduoti, t. y. turi būti $q^k \leq A_q(n, d)$; pasinaudoję Singletono įverčiu, gautume

$$q^k \leq A_q(n, d) \leq q^{n-d+1}, \quad k \leq n-d+1, \quad r = n-k \geq 2t.$$

Taigi Singletono įvertis garantuoja, kad norimą kodą galime sukonstruoti pridėję prie k informacinių simbolių $2t$ papildomų simbolių. Tada perdavimo sulėtėjimo koeficientui $\frac{k}{n}$ galime užrašyti tokį įvertį:

$$\frac{k}{n} \leq 1 - \frac{2t}{n}.$$

Tarkime, kanalas iškreipia simbolį su tikimybe $p, p < \frac{1}{2}$. Perduodant n ilgio kodo žodį, jame vidutiniškai įvyksta np klaidų. Jeigu norėtume, kad kodas ištaisytų „tipinį“ klaidų skaičių $t = [np]$, turėtume pridėti maždaug $2np$ papildomų, klaidoms taisyti skirtų simbolių, o perdavimas sulėtėtų ne daugiau kaip dydžiu

$$1 - \frac{2[np]}{n} < 1 - 2p + \frac{2}{n}.$$

6.5. Ekvivalentūs kodai

Perdažytas dviratis rieda taip pat kaip anksčiau, perdėjus pinigus į kitą kišenę, turtas nepadidėja. Kokios operacijos nepakeičia klaidas taisančių kodų savybių?

Yra dvi tokių operacijų rūšys. Panagrinėkime paprastą abėcėlės $\mathcal{A}_3 = \{0, 1, 2\}$ žodžių kodą $\mathbf{C} = \{0000, 1011, 2220\}$. Jis taiso vieną klaidą. Tokie kodai

$$\mathbf{C}_1 = \{0000, 1110, 2022\}, \quad \mathbf{C}_2 = \{0000, 0111, 2202\}$$

irgi taiso vieną klaidą. Abu jie gauti iš pirmojo, kiekvieno žodžio simboliams pritaikius tą pačią perstatą.

O dabar pakeiskime kodo $\mathbf{C}$ žodžių pirmuosius simbolius: $0 \mapsto 2, 1 \mapsto 0, 2 \mapsto 1$. Gausime kodą

$$\mathbf{C}_3 = \{2000, 0011, 1220\}.$$

Visi keturi kodai skirtingi, bet jų klaidų taisymo savybės tos pačios. Klaidų taisymo požiūriu jie yra ekvivalentūs. Galime simbolių perstatų ir keitimų veiksmus taikyti ne vieną kartą. Gausime vis naujus, tačiau turinčius tas pačias klaidų taisymo savybes kodus. Čia visa esmė. O dabar matematinis, kiek painokas ekvivalenčių kodų apibrėžimas.

Tegu $\mathbf{C}$ yra (n, N) kodas, o σ – n elementų perstata, t. y. injektyvus atvaizdis

$$\sigma \ : \ \{1, 2, \ldots, n\} \ \to \ \{1, 2, \ldots, n\}.$$

Juo apibrėšime injektyvų atvaizdį $\mathcal{A}_q^n \to \mathcal{A}_q^n$, kurį taip pat žymėsime σ. Jei $\mathbf{x} = x_1 \ldots x_n$, tai

$$\sigma(\mathbf{x}) = x_{\sigma(1)} \ldots x_{\sigma(n)}.$$

Iš kodo $\mathbf{C}$ šiuo atvaizdžiu gauname naują kodą

$$\sigma(\mathbf{C}) = \{\sigma(\mathbf{c}) : \mathbf{c} \in \mathbf{C}\}.$$

Natūralu kodus $\mathbf{C}$ ir $\sigma(\mathbf{C})$ laikyti ekvivalenčiais. Atskiru atveju gali būti $\mathbf{C} = \sigma(\mathbf{C})$. Šią savybę turintys atvaizdžiai σ ypatingu būdu susiję su kodo struktūra. Tokių atvaizdžių $\sigma : \mathcal{A}_q^n \to \mathcal{A}_q^n$ aibę

$$Aut(\mathbf{C}) = \{\sigma : \mathbf{C} = \sigma(\mathbf{C})\}$$

vadinsime kodo $\mathbf{C}$ automorfizmų grupe. Ji iš tikrųjų yra grupė algebroje priimta prasme.

Be ką tik aptarto būdo, yra ir kita galimybė gauti formaliai naujus kodus, naudojant perstatas. Tegu $\pi : \{1, 2, \ldots, q\} \to \{1, 2, \ldots, q\}$ yra kokia nors perstata, o abėcėlės $\mathcal{A}$ simboliai sunumeruoti: $\mathcal{A} = \{a_1, \ldots, a_q\}$. Galime π nagrinėti kaip atvaizdį $\mathcal{A} \to \mathcal{A}$, apibrėždami $\pi(a_i) = a_{\pi(i)}$. Pasirinkime $i, i \leq n$, ir apibrėžkime injektyvų atvaizdį $\langle \pi, i \rangle : \mathcal{A}_q^n \to \mathcal{A}_q^n$ šitaip:

$$\langle \pi, i \rangle (x_1 \ldots x_i \ldots x_n) = x_1 \ldots \pi(x_i) \ldots \ x_n.$$

Atvaizdis $\langle \pi, i \rangle$ keičia tik i-ąjį žodžio simbolį. Kodą, kurį gauname iš $\mathbf{C}$, imdami žodžius $\langle \pi, i \rangle (\mathbf{c}), \mathbf{c} \in \mathbf{C}$, žymėsime $\langle \pi, i \rangle (\mathbf{C})$.

40 apibrėžimas. *Du (n, N) kodus $\mathbf{C}, \mathbf{C}'$ vadinsime ekvivalenčiais, jei egzistuoja n elementų perstata σ ir q elementų perstatos $\pi_1, \ldots, \pi_n$, kad*

$$\mathbf{C}' = \langle \pi_1, 1 \rangle (\ldots (\langle \pi_n, n \rangle (\sigma(\mathbf{C}))) \ldots).$$

Ekvivalenčių kodų savybės, susijusios su klaidų taisymu yra tos pačios. Tai išplaukia iš tokio teiginio, kurio teisingumu įsitikinsite šiek tiek pagalvoję:

41 teorema. *Jei kodai $\mathbf{C}, \mathbf{C}'$ ekvivalentūs, tai $d(\mathbf{C}) = d(\mathbf{C}')$.*

Perstatyti pagal tą pačią taisyklę visų kodo žodžių simbolius arba pakeisti visuose žodžiuose tos pačios komponentės simbolius ir šitaip gauti naujus, ekvivalenčius pradiniam kodui žodžius nesunku. Tačiau įsivaizduokime tokį uždavinį: duoti tos pačios abėcėlės žodžių kodai, žodžių kiekiai vienodi, ar tie kodai ekvivalentūs? Kokių tikrinimų, perrankų reiktų imtis? Jeigu nuspręsite, kad tai labai sunkus uždavinys, nesuklysite.

7 Hadamardo kodai

Tiek tyrinėjus bendruosius kodų konstravimo klausimus, laikas sukurti ką nors praktiško. Tačiau sukonstruoti kodą su dideliu minimaliu atstumu – anaiptol nelengvas uždavinys. Pasvarstykime: pirmąjį žodį galime parinkti bet kokį, tačiau antrąjį turime rinkti taip, kad jo Hammingo atstumas iki pirmojo žodžio būtų didelis, rinkdami trečiąjį jau turime stengtis, kad jis būtų pakankamai toli nuo pirmųjų dviejų ir t. t. Tačiau matematikai nėra tokie darbštūs, kad jiems patiktų perrinkimo procedūros. Geriems kodams sudaryti yra ir kitų būdų.

Šiame skyriuje nagrinėjame vieną kodų šeimą. Tiems kodams sudaryti reikia visai nedaug papildomų žinių.

7.1. Matricos

Veiksmų su matricomis priminimas, daugiau nieko...

Primename, kad $n \times m$ matavimų matrica vadiname elementų (dažniausiai skaičių) lentelę, turinčią n eilučių ir m stulpelių. Jei $n = m$, tai matrica vadinama kvadratine. Matricas žymėsime didžiosiomis raidėmis, o jos elementus – mažosiomis su indeksais. Pavyzdžiui,

$$A = (a_{ij}),\ i = 1, \dots n, j = 1, \dots m,$$

žymi matricą, sudarytą iš elementų a_{ij}. Indeksai žymi elemento vietą matricoje, pavyzdžiui, a_{23} reiškia elementą, esantį antrojoje eilutėje ir trečiajame stulpelyje. Kad būtų išvengta neaiškumų, kartais tenka indeksus atskirti kableliu.

Tų pačių matavimų matricas galime sudėti; rezultatas – tų pačių matavimų matrica. Matricą, kurios matavimai yra $n \times m$, galima dauginti iš matricos, kurios matavimai yra $m \times k$. Daugybos rezultatas – $n \times k$ matavimų matrica. Bet kokią matricą taip pat galime dauginti iš skaičiaus. Šių veiksmų mums prireiks nuolat. Be abejonės, žinote, kaip šie veiksmai atliekami – štai taip:

$$\begin{aligned}
(a_{ij}) + (b_{ij}) &= (c_{ij}), \quad & c_{ij} &= a_{ij} + b_{ij};\\
(a_{ij}) \cdot (b_{jl}) &= (d_{il}), \quad & d_{il} &= a_{i1}b_{1l} + a_{i2}b_{2l} + \ldots + a_{im}b_{ml};\\
\alpha \cdot (a_{ij}) &= (f_{ij}), \quad & f_{ij} &= \alpha a_{ij};
\end{aligned}$$

šiose lygybėse indeksai įgyja reikšmes $i = 1, \dots, n; j = 1, \dots, m; l = 1, \dots, k$.

Tų pačių matavimų matricų sudėtis turi savo „nulį“. Matricą, kurios matavimai yra $n \times m$, o visi elementai lygūs nuliui, žymėsime $O_{n,m}$, taigi bet kokiai $n \times m$ matavimų matricai A teisinga lygybė

$$A + O_{n,m} = O_{n,m} + A = A.$$

Kvadratines, t. y. $n \times n$ matavimų matricas galima ir sudėti, ir dauginti. Kaip ir skaičių aibėje, yra ir „nulis“ ir „vienetas“, t. y. kvadratinės matricos O_n ir I_n, su visomis $n \times n$ matricomis A tenkinančios lygybes

$$A + O_n = O_n + A = A, \quad A \cdot I_n = I_n \cdot A = A.$$

Šias matricas taip ir vadinsime: nuline ir vienetine:

$$O_n = \begin{pmatrix} 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 0 \end{pmatrix}, \quad I_n = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{pmatrix}.$$

Dar viena operacija su matricomis, kurią nuolat naudosime – transponavimas. Tegu A yra $n \times m$ matrica. Sudarykime naują matricą, pirmojo A stulpelio elementus surašę į pirmąją eilutę, antrojo – į antrąją ir t. t. Gautąją $m \times n$ matavimų matricą vadinsime transponuota ir žymėsime A^T. Akivaizdu, pavyzdžiui, kad

$$(A^T)^T = A.$$

Gerai dar būtų žinoti, kas yra matricos rangas, kvadratinės matricos determinantas ir kaip šiuos dydžius apskaičiuoti...

7.2. Hadamardo matricos

Tai paprastos, iš skaičių $-1, +1$ sudarytos kvadratinės matricos. Tačiau išsamių žinių apie jas kol kas niekas pasaulyje neturi.

Matricas, kurias dabar nagrinėsime, sugalvojo J. Sylvesteris (1867), tačiau išsamiau tyrinėjo J. Hadamardas (1893). Matricoms prigijo jo vardas.

41 apibrėžimas. *n-osios eilės kvadratinė matrica $H_n = (h_{ij})$ vadinama Hadamardo matrica, jei*

$$h_{ij} = \pm 1, \quad H_n \cdot H_n^T = nI_n.$$

Apibrėžimo sąlygą galime užrašyti štai taip:

$$h_{i1}h_{j1} + h_{i2}h_{j2} + \ldots + h_{in}h_{jn} = \begin{cases} n, & \text{jei } i = j, \\ 0, & \text{jei } i \neq j. \end{cases}$$

Kartais sakysime, kad Hadamardo matricos skirtingos eilutės yra ortogonalios, t. y. jų skaliarinė sandauga lygi nuliui.

O štai ir Hadamardo matricų pavyzdžiai:

$$H_1 = (1), \quad H_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad H_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}.$$

Kaip konstruoti didesnių matavimų matricas? Iš pradžių aptarkime, kokių Hadamardo matricų apskritai gali būti.

42 teorema. *Sukeitus Hadamardo matricos dvi eilutes (stulpelius) vietomis, gautoji matrica vėl yra Hadamardo matrica. Padauginus Hadamardo matricos eilutę (stulpelį) iš* -1, *gaunama Hadamardo matrica.*

Jeigu gerai supratote Hadamardo matricos apibrėžimą, tai šis teiginys turėtų būti visiškai aiškus.

43 teorema. *n-osios eilės Hadamardo matrica $H_n = (h_{ij})$ vadinama normaline, jei visiems j teisinga lygybė $h_{1j} = h_{j1} = 1$.*

Taigi normalinės Hadamardo matricos pirmasis stulpelis ir pirmoji eilutė sudaryti vien tik iš vienetų. Pasirėmę ankstesniu teiginiu galime įrodyti, kad kiekvieną Hadamardo matricą galima suvesti į normalinį pavidalą.

44 teorema. *Atitinkamas eilutes ir stulpelius dauginant iš* -1, *kiekvieną Hadamardo matricą galima suvesti į normalinę matricą.*

Naudojantis šia teorema, nesunku įrodyti, kad trečiosios eilės Hadamardo matricos nėra. Iš tiesų, jeigu egzistuotų trečiosios eilės Hadamardo matrica, tai suteikę jai normalinį pavidalą, gautume Hadamardo matricą

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & a & b \\ 1 & c & d \end{pmatrix},$$

čia a, b yra arba 1, arba -1. Pirmosios ir antrosios eilutės skaliarinė sandauga turi būti lygi nuliui, todėl $1 + a + b = 0$. Aišku, kad tokia lygybė negali būti patenkinta, todėl trečios eilės Hadamardo matricų nėra.

O kokių gi Hadamardo matricų gali būti?

45 teorema. *Jei H_n yra Hadamardo matrica, tai $n = 1, 2$ arba n dalijasi iš* 4.

Įrodymas. Pakanka nagrinėti normalines n-osios eilės Hadamardo matricas. Tada kiekvienoje tokios matricos eilutėje pradedant antrąja turi būti po tiek pat skaičių 1 ir -1. Taigi n turi būti lyginis $n = 2m$. Sukeisdami matricos stulpelius vietomis, galime ją pertvarkyti taip, kad pirmieji m antrosios eilutės elementai būtų lygūs vienetui. Tegu tarp pirmųjų m trečiosios eilutės

elementų yra j vienetų ir $m-j$ minus vienetų. Tada tarp paskutiniųjų m trečiosios eilutės elementų vienetų yra $m-j$, o minus vienetų – j. Antroji ir trečioji eilutės yra ortogonalios, t. y. jų skaliarinė sandauga lygi 0:

$$j-(m-j)-(m-j)+j=0.$$

Iš šios lygybės gauname, kad $m=2j$. Taigi skaičius $n=2m=4j$ dalijasi iš keturių.

Hadamardo matricos eilė būtinai yra ketverto kartotinis. Ar kiekvienas toks skaičius gali būti Hadamardo matricos eilė? Linkstama manyti, kad taip ir yra, tačiau kol kas tai – neįrodyta hipotezė. Ji patikrinta visiems $4m \leq 668$.

7.3. Dvi konstrukcijos

Galima sukonstruoti be galo daug Hadamardo matricų. Panagrinėsime du būdus. Abu jie paprasti. Kodėl „veikia“ pirmas būdas suprasti nesudėtinga. Kodėl antrasis – sunkiau.

Jau sudarėme antrosios ir ketvirtosios eilės Hadamardo matricas H_2, H_4. Naudodamiesi jomis galime sukonstruoti aukštesnės eilės Hadamardo matricas, ir dar aukštesnės, ir dar...

Bet iš pradžių – dar vienos operacijos su matricomis apibrėžimas.

42 apibrėžimas. *Tegu $A=(a_{ij})$ yra $n\times n$, $B=(b_{ij})$, $m\times m$ matricos. Jų Kroneckerio sandauga vadinama $nm\times nm$ matrica*

$$A\otimes B=\begin{pmatrix} a_{11}B & a_{12}B & \dots & a_{1m}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{11}B & a_{12}B & \dots & a_{1m}B \end{pmatrix}.$$

Šis veiksmas iš Hadamardo matricų visada sukuria naujas Hadamardo matricas.

46 teorema. *Jei A, B yra Hadamardo matricos, tai $A\otimes B$ irgi Hadamardo matrica.*

Šios teoremos įrodymo beveik nereikia – kad ji teisinga, bus aišku kiekvienam, kas gerai suvoks Hadamardo matricos ir Kroneckerio sandaugos apibrėžimus.

Taigi naudodamiesi Kroneckerio sandauga, galime konstruoti $2^m\times 2^m$ matavimų Hadamardo matricas:

$$H_4=H_2\otimes H_2,\quad H_8=H_4\otimes H_2,\quad H_{16}=H_4\otimes H_4,\quad \dots$$

Ne kažin kas! Kadangi nelygybė $2^m<100$ turi tik septynis natūraliuosius sprendinius, tai šitaip galime sukonstruoti tik septynias mažesnės už 100 eilės Hadamardo matricas.

Tačiau yra ir kitų būdų. Vieną jų panagrinėkime.

47 teorema. *Jei $p = 4m + 3$ yra pirminis skaičius, tai $p + 1$ eilės Hadamardo matrica egzistuoja.*

Šios teoremos įrodymas yra konstruktyvus, t. y. pateikia nurodytos eilės Hadamardo matricų konstravimo būdą. Šį būdą sugalvojo anglų matematikas R. Paley.

Yra 14 pirminių skaičių, mažesnių už 100 ir tenkinančių teoremos sąlygą. Pasinaudoję jais, galime sukonstruoti Hadamardo matricas

$$H_n, \quad n = 4, 8, 12, 20, 24, 32, 44, 48, 60, 68, 72, 80, 84.$$

Tris šio rinkinio matricas galime sudaryti naudodamiesi Kroneckerio sandauga. Taigi naudodamiesi šiais dviem būdais galime sudaryti iš viso 17 Hadamardo matricų H_n, $1 \leq n < 100$.

Panagrinėsime Paley konstrukciją, sudarydami Hadamardo matricą, atitinkančią skaičių $p = 7$. Pirmiausia surašykime skaičių dalybos iš 7 liekanų seką:

$$0,\ 1,\ 2,\ 3,\ 4,\ 5,\ 6. \tag{45}$$

Dabar kelkime šios sekos skaičius kvadratu ir surašykime šių kvadratų dalybos iš 7 liekanų seką:

$$0,\ 1,\ 4,\ 2,\ 2,\ 4,\ 1. \tag{46}$$

Dabar sudarykime eilutę iš -1 ir $+1$: jeigu (45) sekos skaičius yra įrašytas sekoje (46), keiskime jį $+1$, jei nėra -1. Gausime tokią eilutę:

$$+1, +1, +1, -1, +1, -1, -1.$$

Sudarykime $p \times p$, t. y. 7×7 matavimų matricą, įrašydami į pirmąją eilutę gautąją vienetų ir minus vienetų seką, o į kitas eilutes – ciklišk ai per $1, 2, \ldots, 6$ pozicijas į dešinę perstumtą pirmąją eilutę:

$$\begin{pmatrix} +1 & +1 & +1 & -1 & +1 & -1 & -1 \\ -1 & +1 & +1 & +1 & -1 & +1 & -1 \\ -1 & -1 & +1 & +1 & +1 & -1 & +1 \\ +1 & -1 & -1 & +1 & +1 & +1 & -1 \\ -1 & +1 & -1 & -1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 & -1 & +1 & +1 \\ +1 & +1 & -1 & +1 & -1 & -1 & +1 \end{pmatrix}.$$

Kad gautume Hadamardo matricą H_8, turime atlikti paskutinį veiksmą – papildyti sudarytąją matricą iki 8×8 matavimų matricos, pridėdami dar

vieną stulpelį – vien iš -1, o tada eilutę, sudarytą iš $+1$:

$$H_8 = \begin{pmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ -1 & +1 & +1 & +1 & -1 & +1 & -1 & -1 \\ -1 & -1 & +1 & +1 & +1 & -1 & +1 & -1 \\ -1 & -1 & -1 & +1 & +1 & +1 & -1 & +1 \\ -1 & +1 & -1 & -1 & +1 & +1 & +1 & -1 \\ -1 & -1 & +1 & -1 & -1 & +1 & +1 & +1 \\ -1 & +1 & -1 & +1 & -1 & -1 & +1 & +1 \\ -1 & +1 & +1 & -1 & +1 & -1 & -1 & +1 \end{pmatrix}.$$

Šis algoritmas tinka bet kokiam pirminiam $p = 4m + 3$ skaičiui. Kodėl jis veikia? Priežastys glūdi tam tikruose skaičių teorijos sąryšiuose.

7.4. Hadamardo kodai

Pasinaudoję Hadamardo matricomis, sudarysime kodus. Vienas jų turi vietelę kosmonautikos istorijoje...

Tarkime, H_n yra n-osios eilės normalinė Hadamardo matrica. Pirmiausia pertvarkykime ją taip, kad nebeliktų elementų -1.

43 apibrėžimas. *Jei H_n yra Hadamardo matrica, tai matrica M_n, gauta iš H_n, pakeitus -1 į 0, vadinama dvinare Hadamardo matrica.*

Matricos M_n eilutes galime interpretuoti kaip dvejetainės abėcėlės $\mathcal{A} = \{0, 1\}$ ženklų n ilgio žodžius. Pastebėkime, kad Hammingo atstumas tarp žodžių a_i, a_j, gautų iš skirtingų M_n eilučių, lygus $n/2$.

Dabar sudarysime Hadamardo kodus.

44 apibrėžimas. *Tegu M_n yra n-tosios eilės dvinarė Hadamardo matrica, gauta iš normalizuotos Hadamardo matricos, M'_n – matrica, gauta iš H_n, pakeitus 1 į 0 ir -1 į 1. Kodas $\mathbf{A}_n$ sudarytas iš M_n eilučių, sutrumpintų pirmuoju simboliu, $\mathbf{B}_n$ – iš M_n ir M'_n eilučių, o $\mathbf{C}_n$ – iš $\mathbf{B}_n$ eilučių, sutrumpintų pirmuoju simboliu.*

Kodus $\mathbf{A}_n, \mathbf{B}_n, \mathbf{C}_n$ vadinsime Hadamardo kodais.

Įrodysime teiginį apie Hadamardo kodų parametrus.

48 teorema. *Kodų $\mathbf{A}_n, \mathbf{B}_n, \mathbf{C}_n$ parametrai yra atitinkamai*

$$(n-1, n, n/2), \quad (n, 2n, n/2), \quad (n-1, 2n, d),\ d \geq n/2 - 1.$$

Įrodymas. Imkime žodžius $\mathbf{a}_i, \mathbf{a}_j$, gautus iš skirtingų M_n eilučių; Hammingo atstumas tarp jų:

$$h(\mathbf{a}_i, \mathbf{a}_j) = \frac{n}{2}.$$

Sutrumpinę juos pirmuoju simboliu, gauname du kodo $\mathbf{A}_n$ žodžius. Tačiau abu nubrauktieji simboliai buvo lygūs vienetui, taigi juos nubraukę, atstumo tarp žodžių nesumažinome. Įsitikinome, kad ir minimalus kodo $\mathbf{A}_n$ atstumas, ir apskritai – atstumas tarp bet kurių dviejų skirtingų šio kodo žodžių lygus $n/2$.

Nagrinėkime du kodo $\mathbf{B}_n$ žodžius. Jeigu jie abu gauti iš matricos M_n (arba matricos M'_n) eilučių, tai Hammingo atstumas tarp jų taip pat lygus $n/2$. Lieka išnagrinėti atvejį, kai vienas žodis gautas iš vienos, kitas iš kitos matricos. Tarkime, žodis $\mathbf{c}$ gautas iš i-osios M_n eilutės, o žodis $\mathbf{d}$ – iš j-osios M'_n eilutės. Jeigu $i = j$, tai $h(\mathbf{c}, \mathbf{d}) = n$. Jeigu $i \neq j$, tai $h(\mathbf{c}, \mathbf{d}) = n/2$.

Ir šiuo atveju nustatėme šiek tiek daugiau, negu užrašyta teoremos tvirtinime: yra lygiai n kodo žodžių porų su Hammingo atstumu, lygiu n, visos kitos žodžių poros duoda perpus mažesnį atstumą.

Teiginys apie kodo $\mathbf{C}_n$ minimalų atstumą beveik akivaizdus – sutrumpinę kodo žodžius vienu simboliu, minimalų atstumą galime sumažinti daugiausiai vienetu.

Hadamardo kodas $\mathbf{B}_{32}$ su parametrais $(32, 64, 16)$ 1969–1972 metais buvo naudotas siunčiant į Žemę Marso nuotraukas iš kosminio laivo „Mariner“. Nuotrauka buvo dalijama taškais, taškams buvo priskiriamas vienas iš 64 skirtingų pilkumo lygių, kiekvieną iš šių lygių atitiko kodo žodis. Šie žodžiai ir buvo siunčiami į Žemę. Taigi Hadamardo kodas $\mathbf{B}_{32}$ – pirmasis kodas, taikytas žvalgant kosmosą. Jis gali ištaisyti 7 įvykusias klaidas.

Kaip šias klaidas taisyti? Skaičiuoti gautojo žodžio Hammingo atstumą iki visų žodžių? Yra kiek geresnis būdas. Aptarkime $\mathbf{B}_n$ kodo, gauto iš matricų M_n ir M'_n eilučių, dekodavimą (šios matricos gautos iš H_n).

Tarkime, siunčiant kodo $\mathbf{B}_n$ žodį $\mathbf{c}$, iš kanalo gautas žodis $\mathbf{x}$. Jeigu pradinis žodis sudarytas iš matricos M_n eilutės, tai egzistuoja matricos H_n eilutė $\mathbf{h}$, kad, pakeitę joje -1 į 0, iškart gauname $\mathbf{c}$. Sakysime, kad žodis $\mathbf{c}$ gautas iš $\mathbf{h}$ pirmuoju būdu. Jeigu $\mathbf{c}$ sutampa su viena iš M'_n eilučių, tai taip pat yra vienintelė H_n eilutė $\mathbf{h}$, kurioje pakeitę $-1 \mapsto 1, 1 \mapsto 0$, gautume $\mathbf{c}$. Sakysime, kad žodis $\mathbf{c}$ gautas iš $\mathbf{h}$ antruoju būdu.

Taigi dekoduoti gautą žodį $\mathbf{x}$ reiškia surasti Hadamardo matricos eilutę $\mathbf{h}$ ir nustatyti, kuriuo iš dviejų būdų ją pakeisti, kad gautume kodo žodį.

Dviejų dvejetainės abėcėlės žodžių (eilučių) $\mathbf{x} = x_1x_2\ldots x_n, \mathbf{y} = y_1y_2\ldots y_n$ skaliarinę sandaugą žymėsime taip:

$$\mathbf{x} \cdot \mathbf{y} = x_1y_1 + \ldots + x_ny_n.$$

Žodį, gautą iš dvejetainės abėcėlės žodžio $\mathbf{x}$, pakeitus jame $0 \mapsto -1$, žymėsime $\hat{\mathbf{x}}$.

49 teorema. *Tegu $\mathbf{c} \in \mathbf{B}_n$, $\mathbf{h}$ – matricos H_n eilutė, iš kurios šis žodis gautas, o $\mathbf{x}$ – pasiuntus $\mathbf{c}$, iš kanalo gautas žodis. Jeigu iškraipytų simbolių skaičius t ne didesnis už $\left[\frac{n}{4} - \frac{1}{2}\right]$ ir $\mathbf{c}$ gautas iš $\mathbf{h}$ pirmuoju būdu, tai $\hat{\mathbf{x}} \cdot \mathbf{h} > \frac{n}{2}$; jei antruoju – $\hat{\mathbf{x}} \cdot \mathbf{h} < -\frac{n}{2}$. Skaliarinės sandaugos su kitomis*

matricos H_n eilutėmis $\mathbf{h}'$ *tenkina nelygybę*

$$|\hat{\mathbf{x}} \cdot \mathbf{h}| < \frac{n}{2}.$$

Taigi naudodamiesi šiuo teiginiu, galime dekoduoti taip: gavę iš kanalo žodį $\mathbf{x}$, skaičiuojame skaliarines sandaugas $\hat{\mathbf{x}} \cdot \mathbf{h} > \frac{n}{2}$ su H_n eilutėmis, kol gausime reikšmę didesnę už $\frac{n}{2}$ arba mažesnę už $-\frac{n}{2}$. Šitaip išvengsime visų kodo žodžių perrankos.

Įrodymas. Jeigu, siunčiant pirmuoju būdu iš $\mathbf{h}$ gautą žodį $\mathbf{c}$, iškraipymų nebuvo, tai gautas žodis tenkins lygybes:

$$\hat{\mathbf{x}} \cdot \mathbf{h} = n, \quad \hat{\mathbf{x}} \cdot \mathbf{h}' = 0, \text{ jei } \mathbf{h}' \neq \mathbf{h}.$$

Antruoju būdu gautajam žodžiui turėtume atitinkamai $\hat{\mathbf{x}} \cdot \mathbf{h} = -n$.

Tarkime, siunčiant $\mathbf{c}$, vienas simbolis buvo iškreiptas. Tada sandaugų $\hat{\mathbf{x}} \cdot \mathbf{h}$ ir $\hat{\mathbf{x}} \cdot \mathbf{h}'$ išraiškose pasikeis vienas dėmuo: $+1$ į -1 arba -1 į $+1$. Taigi sandaugos padidės arba sumažės dvejetu. Todėl turėsime

$$\hat{\mathbf{x}} \cdot \mathbf{h} \geq n - 2, \quad |\mathbf{x} \cdot \mathbf{h}'| \leq 2$$

arba

$$\hat{\mathbf{x}} \cdot \mathbf{h} \leq -n + 2, \quad |\mathbf{x} \cdot \mathbf{h}'| \leq 2.$$

Jeigu įvyks t iškraipymų, pirmuoju atveju bus teisingos nelygybės

$$\hat{\mathbf{x}} \cdot \mathbf{h} \geq n - 2t > \frac{n}{2}, \quad |\mathbf{x} \cdot \mathbf{h}'| \leq 2t < \frac{n}{2}.$$

Panašios nelygybės bus teisingos antruoju atveju. Teorema įrodyta.

8 Baigtiniai kūnai ir tiesinės erdvės

Iki šiol iš abėcėlės – tos pradinės žaliavos kodų žodžiams sudaryti – nereikalavome jokių savybių. Jos elementai buvo ženklai ir tiek. Nuo šiol abėcėlė bus ne šiaip aibė, bet aibė, turinti tam tikrą algebrinę struktūrą. Šis skyrius skirtas „algebros inventoriui“ – baigtiniams kūnams, tiesinėms erdvėms ir daugianariams bei jų savybėms aptarti. Jeigu šios sąvokos yra gerai žinomos (taip turėtų būti, jeigu sėkmingai įveikėte pradinį universitetuose dėstomą algebros kursą), galite šį skyrių tiesiog praleisti.

Veiksmų su žodžiais prisireiks labai greitai, o daugianariais pasinaudosime vėliau. Tad galima peržvelgus pirmuosius skyrelius pereiti prie kodų, o prie daugianarių sugrįžti, kai skaitysite kitus skyrius.

8.1. Dalybos liekanų kūnas

Skaičiavimo dalybos iš n liekanų aibėje savybių apžvalga.

Teiginys apie sveikųjų skaičių dalybą su liekana – paprastas, bet labai svarbus.

50 teorema. *Bet kokiems sveikiesiems skaičiams m, n $(n > 0)$ yra vienintelė sveikųjų skaičių pora q, r, kad*

$$m = qn + r, \quad 0 \leq r < b.$$

Skaičių r vadiname m dalybos iš n liekana, o q – dalmeniu.

Teiginio įrodymas irgi paprastas. Nagrinėkime skaičiaus n kartotinius, t. y. sveikųjų skaičių seką

$$\ldots,\ -3 \cdot n,\ -2 \cdot n,\ -1 \cdot n,\ 0 \cdot n,\ 1 \cdot n,\ 2 \cdot n,\ 3 \cdot n,\ \ldots$$

Tik vienas šios sekos narys $q \cdot n$ pateks į intervalą $(m-n, m]$, t. y. tenkins nelygybę

$$m - n < qn \leq m, \quad \text{arba} \quad 0 \leq m - qn < n.$$

Taigi

$$m = qn + r, \quad r = m - qn, \quad 0 \leq r < n.$$

Padarę prielaidą, kad egzistuoja ir kita pora, gautume išvadą, kad intervale $(m - n, m]$ yra bent du skirtingi skaičiaus n kartotiniai. Taigi teiginys įrodytas.

Jeigu du sveikieji skaičiai a ir b, dalijant juos iš n, duoda tą pačią liekaną, žymėsime

$$a \equiv b \pmod{n}.$$

Taip užrašytą sąryšį vadinsime lyginiu.

Taigi šis žymuo reiškia dviejų skaičių giminystę pagal dalybos iš n liekanas. Nesunku įrodyti šio sąryšio savybes.

51 teorema. *Teisingi tokie teiginiai:*

- *jei $a \equiv b \pmod{n}$ ir $c \equiv d \pmod{n}$, tai $a + c \equiv b + d \pmod{n}$;*
- *jei $a \equiv b \pmod{n}$ ir c yra sveikasis skaičius, tai $ca \equiv cb \pmod{n}$;*
- *jeigu $ca \equiv cb \pmod{n}$ ir skaičiai c, n neturi bendrųjų daliklių, išskyrus vienetą, tai $a \equiv b \pmod{n}$.*

Pažymėkime visų galimų dalybos iš n liekanų aibę

$$\mathbb{Z}_n = \{0, 1, 2, \ldots, n-1\}.$$

Šios aibės skaičių sumos ar sandaugos rezultatas, žinoma, nebūtinai priklauso $\mathbb{Z}_n$. Apibrėžkime naujus liekanų sudėties ir sandaugos veiksmus, kurių rezultatai visada priklauso $\mathbb{Z}_n$.

45 apibrėžimas. *Elementų $a, b \in \mathbb{Z}_n$ suma moduliu n $(n > 1)$ vadinsime natūrinio skaičiaus $a + b$ dalybos iš n liekaną, o sandauga – skaičiaus $a \cdot b$ dalybos iš n liekaną. Apibrėžtųjų veiksmų rezultatus žymėsime*

$$a +_n b, \quad a \times_n b.$$

Svarbu, kad dauguma apibrėžtųjų veiksmų savybių – tos pačios kaip ir įprastinių veiksmų su skaičiais.

52 teorema. *Suma ir sandauga moduliu n turi šias savybes*

- $(a +_n b) +_n c = a +_n (b +_n c)$;
- $a +_n b = b +_n a$;
- $a +_n 0 = a$;
- *bet kokiam a lygtis $a +_n x = 0$ turi vienintelį sprendinį;*
- $a \times_n b = b \times_n a$;
- $1 \times_n a = a$;
- $(a \times_n b) \times_n c = a \times_n (b \times_n c)$;
- $a \times_n (b +_n c) = a \times_n b +_n a \times_n c$.

Aibė, su kurios elementais apibrėžtos dvi operacijos (paprastai vadinamos sudėtimi ir daugyba), tenkinančios teoremoje išvardytas sąlygas, vadinama žiedu. Taigi dalybos liekanų aibė $\mathbb{Z}_n$ su apibrėžtomis operacijomis yra žiedas. Šiame žiede yra nulinis elementas (nulis), vienetinis elementas (vienetas), kiekvienas elementas turi jam priešingą, t. y. tokį, su kuriuo sudėjus gaunamas nulis. Tai įprastos veiksmų su skaičiais savybės.

Tačiau yra viena ypatybė. Dviejų nelygių nuliui skaičių sandauga niekada nėra lygi nuliui. Daugybė matematinių samprotavimų ir įrodymų remiasi šiuo faktu. Tačiau liekanų žiede tai nebėra visada teisinga. Pavyzdžiui,

$$4 \times_6 3 = 0.$$

Ir dar vienas skirtumas. Lygtis

$$a \times_n x = b \ (a \neq 0) \quad \text{atskiru atveju} \quad a \times_n x = 1$$

ne su visais a turi sprendinį, t. y. ne kiekvienas nenulinis elementas turi atvirkštinį.

Tačiau ir šią gerą savybę galime išgelbėti, jeigu pasirinksime „gerus“ skaičius n.

53 teorema. *Jeigu n yra pirminis skaičius, tai kiekvienam a, $a \neq 0$, lygtis*

$$a \times_n x = 1$$

turi vienintelį sprendinį.

Taigi kai n yra pirminis, visi nenuliniai $\mathbb{Z}_n$ elementai turi atvirkštinius.

Įrodyti šį teiginį nesunku. Tegu $a \in \mathbb{Z}_n, a \neq 0$. Nagrinėkime seką iš $n-1$ aibės $\mathbb{Z}_n$ elemento:

$$1 \times_n a,\ 2 \times_n a,\ \ldots,\ (n-1) \times_n a.$$

Visi šie elementai nenuliniai ir skirtingi, todėl lygiai vienas lygus 1. Taigi elementas a turi atvirkštinį.

Galbūt abejojate, kodėl elementai nenuliniai? Jeigu būtų $b \times_n a = 0$, tai natūrinių skaičių sandauga ba dalytųsi iš pirminio n. Bet tai neįmanoma, nes a ir b yra mažesni už n.

Kodėl elementai skirtingi? Jeigu būtų $b \times_n a = c \times_n a$ $(0 < b < c < n)$, tai būtų teisingas lyginys

$$ba \equiv ca \pmod{n}.$$

Šį lyginį galime suprastinti. Padarę tai, gautume $b \equiv c \pmod{n}$. Kadangi $0 < b, c < n$, tai iš to gauname, kad $b = c$.

Taigi, kai n yra pirminis skaičius, tai veiksmų su dalybos liekanomis savybių sąrašą galime papildyti dar viena savybe. Aibė, kurioje apibrėžti du veiksmai (sudėtis ir daugyba), turintys abiejose teoremose išvardytas savybes, vadinama kūnu. Veiksmų su kūno elementais savybės yra visiškai tos pačios kaip veiksmų, pavyzdžiui, su racionaliaisiais skaičiais.

O dabar susitarkime dėl žymėjimų. Norėdami pabrėžti, kad nagrinėjamas skaičius yra pirminis, žymėsime jį p. Kūną $\mathbb{Z}_p$ žymėsime, kaip įprasta algebroje, $\mathbb{F}_p$. Vietoj ženklų $+_p, \times_p$ rašysime įprastinius sudėties ir sandaugos ženklus, žodžiu nurodydami, kokiu moduliu turi būti atliekami skaičiavimai. Nenulinio elemento $\alpha \in \mathbb{F}_p$ atvirkštinį žymėsime α^{-1}.

Taigi turime be galo daug kūnų, kuriuos galime naudoti kaip kodų abėcėles:

$$\mathbb{F}_2,\ \mathbb{F}_3,\ \mathbb{F}_5,\ \ldots$$

Kuris iš jų yra pats svarbiausias kodavimo teorijoje? Be abejonės, pats mažiausias kūnas $\mathbb{F}_2 = \{0, 1\}$ yra ir pats svarbiausias.

8.2. Tiesinės žodžių erdvės ir poerdviai

Nuo simbolių – prie žodžių! Mažai naudos iš turto, jeigu jį tik turi; kitas dalykas, jei gali su juo ką nors veikti! Pasinaudodami veiksmais su abėcėlės simboliais, apibrėšime veiksmus su iš tų simbolių sudarytais žodžiais ir apžvelgsime svarbiausias kodavimo teorijai jų savybes.

Žodžiams sudaryti naudosime abėcėlę $\mathbb{F}_p = \{0, 1, 2, \ldots, p-1\}$, čia $p > 1$ yra pirminis skaičius. Naudodamiesi veiksmais su abėcėlės ženklais, apibrėšime veiksmus su to paties ilgio žodžiais, t. y. aibės $V_n = \mathbb{F}_p^n$ elementais. Pirmiausia apibrėžkime sudėtį:

46 apibrėžimas. *Žodžių* $\mathbf{x}, \mathbf{y} \in V_n, \mathbf{x} = x_1x_2\ldots x_n, \mathbf{y} = y_1y_2\ldots y_n$, *suma vadinsime žodį* $\mathbf{z} = z_1z_2\ldots z_n$, *čia*

$$z_1 = x_1 + y_1,\ z_2 = x_2 + y_2,\ \ldots, z_n = x_n + y_n.$$

Žodžių sudėties savybės – tokios pat kaip skaičių.

54 teorema. *Tegu* $\mathbf{x}, \mathbf{y}, \mathbf{z}$ *yra bet kokie aibės* V_n *elementai. Teisingi šie teiginiai:*

- $(\mathbf{x} + \mathbf{y}) + \mathbf{z} = \mathbf{x} + (\mathbf{y} + \mathbf{z})$;
- $\mathbf{x} + \mathbf{y} = \mathbf{y} + \mathbf{x}$;
- *egzistuoja* $\mathbf{0} \in V_n$, *kad* $\mathbf{x} + \mathbf{0} = \mathbf{x}$;
- *egzistuoja* $\bar{\mathbf{x}}$, *kad* $\mathbf{x} + \bar{\mathbf{x}} = \mathbf{0}$.

Aibė su kiekvienai elementų porai apibrėžtu veiksmu (dažniausiai vadinamu sudėtimi), turinčiu teoremoje išvardytas savybes, vadinama Abelio grupe. Taigi pasinaudoję abėcėlės ženklų sudėtimi, pavertėme žodžių aibę grupe.

Elementas $\mathbf{0} = 00\ldots0$ atlieka nulio vaidmenį; jį taip ir vadinsime – nuliniu žodžiu arba elementu. Žodį $\bar{\mathbf{x}}$ vadinsime priešingu žodžiui $\mathbf{x}$ ir žymėsime $-\mathbf{x}$.

Vienas veiksmas su kūno $\mathbb{F}_p$ elementais dar nepanaudotas – daugyba.

47 apibrėžimas. *Elemento* $\alpha \in \mathbb{F}_p$ *ir žodžio* $\mathbf{x} = x_1x_2\ldots x_n \in \mathbb{F}_p^n$ *sandauga vadinsime žodį* $\mathbf{y} = y_1y_2\ldots y_n$, *kad*

$$y_1 = \alpha x_1,\ y_2 = \alpha x_2,\ \ldots,\ y_n = \alpha x_n.$$

Taigi su žodžiais galime atlikti du veiksmus: žodžius galime sudėti, žodžius galime dauginti iš kūno elementų. Nesudėtinga įrodyti pastarojo veiksmo savybes.

55 teorema. *Su bet kokiais* $\alpha, \beta \in \mathbb{F}_p$ *ir* $\mathbf{x}, \mathbf{y} \in \mathbb{F}_p$ *teisingi teiginiai*

- $\alpha(\mathbf{x} + \mathbf{y}) = \alpha\mathbf{x} + \alpha\mathbf{y}$;
- $(\alpha + \beta)\mathbf{x} = \alpha\mathbf{x} + \beta\mathbf{x}$;
- $(\alpha\beta)\mathbf{x} = \alpha(\beta\mathbf{x})$;
- $1\mathbf{x} = \mathbf{x}$.

Aibė su elementų sudėties ir daugybos iš kūno elementų operacijomis, tenkinančiomis abiejose teoremose išvardytas savybes, vadinama **tiesine erdve**. Taigi žodžių aibė V_n yra tiesinė erdvė. Visos tos savybės panašios į įprastines veiksmų su skaičiais savybes. Taigi beveik nereikia įsiminti nieko naujo!

Visų veiksmų esmė – iš to, ką turi, sukurti ką nors nauja. Tegu $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ yra aibės V_n žodžiai. Tiesine šių žodžių kombinacija su koeficientais $\alpha_1, \ldots, \alpha_m \in \mathbb{F}_p$ vadinsime žodį

$$\mathbf{y} = \alpha_1\mathbf{x}_1 + \alpha_2\mathbf{x}_2 + \ldots + \alpha_m\mathbf{x}_m.$$

Pasirinkę kelis aibės V_n žodžius, galime sudaryti pasirinktųjų žodžių visų tiesinių kombinacijų aibę.

48 apibrėžimas. *Elementų $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m \in \mathbb{F}_p$ tiesiniu apvalku vadinsime aibę*

$$\mathcal{L}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m) = \{\alpha_1\mathbf{x}_1 + \alpha_2\mathbf{x}_2 + \ldots + \alpha_m\mathbf{x}_m : \alpha_i \in \mathbb{F}_p\}.$$

Kiekvienoje didelėje erdvėje galime įžvelgti mažesnes.

49 apibrėžimas. *Tiesinės erdvės V_n poaibį L vadinsime tiesiniu poerdviu, jeigu*

- *bet kokiems $\mathbf{x}, \mathbf{y} \in L$ jų suma $\mathbf{x} + \mathbf{y} \in L$;*
- *bet kokiems $\alpha \in \mathbb{F}_p$, $\mathbf{x} \in L$ sandauga $\alpha\mathbf{x} \in L$.*

Nesunku įsitikinti, kad tiesinis žodžių apvalkas yra tiesinis poerdvis. Ar kiekvienas tiesinis poerdvis yra kokios nors žodžių sistemos tiesinis apvalkas? Tai teisinga, pavyzdžiui, visai žodžių erdvei.

Visa erdvė V_n yra žodžių

$$\mathbf{e}_1 = 100\ldots00,\ \mathbf{e}_2 = 010\ldots00,\ \ \ldots,\ \mathbf{e}_n = 000\ldots01 \tag{47}$$

tiesinis apvalkas, t. y. $V_n = \mathcal{L}(\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_m)$.

Jeigu sudarydami tiesinę žodžių kombinaciją, visus koeficientus imsime lygius nuliui, tai gausime vien iš nulių sudarytą žodį. Tačiau gali būti, kad nulinį žodį gausime ir su kitokiu koeficientų rinkiniu.

50 apibrėžimas. *Sakysime, kad žodžiai $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ yra tiesiškai nepriklausomi, jeigu lygybė*

$$\alpha_1\mathbf{x}_1 + \alpha_2\mathbf{x}_2 + \ldots + \alpha_m\mathbf{x}_m = \mathbf{0}, \quad \mathbf{0} = 00\ldots0,$$

teisinga tik tuomet, kai $\alpha_1 = \alpha_2 = \ldots = \alpha_m = 0$. Priešingu atveju sakysime, kad žodžiai yra tiesiškai priklausomi.

Jeigu žodžiai $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ yra tiesiškai priklausomi, tai galime sudaryti nulinę jų kombinaciją

$$\alpha_1\mathbf{x}_1 + \alpha_2\mathbf{x}_2 + \ldots + \alpha_m\mathbf{x}_m = \mathbf{0},$$

panaudoję ne vien tik nulinius koeficientus. Tarkime, $\alpha_m \neq 0$. Žodį $\mathbf{x}_m$ galime išreikšti per kitus:

$$\mathbf{x}_m = -\alpha_m^{-1}\alpha_1\mathbf{x}_1 - \alpha_m^{-1}\alpha_2\mathbf{x}_2 - \ldots - \alpha_m^{-1}\alpha_{m-1}\mathbf{x}_{m-1}.$$

Tada

$$\mathcal{L}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m) = \mathcal{L}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{m-1}).$$

Taigi sudarant tiesinius apvalkus, galima „praretinti“ turimų žodžių sistemą, kad likusieji būtų tiesiškai nepriklausomi.

51 apibrėžimas. *Tegu $L \subset \mathbb{F}_p^n$ yra tiesinis poerdvis. Tiesiškai nepriklausomų žodžių sistemą $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ vadinsime poerdvio baze, jei*

$$L = \mathcal{L}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m).$$

Visa žodžių erdvė turi bazę; tai (47) žodžių sistema. Ar kiekvienas poerdvis turi bazę ir savo ruožtu yra tiesinis bazės žodžių apvalkas? Ar visos to paties poerdvio bazės turi po tą patį žodžių skaičių? Į visus šiuos klausimus atsako lakoniškas, bet labai svarbus teiginys.

56 teorema. *Bet kuris poerdvis $\mathbf{L} \subset \mathbb{F}_p^n$ turi bazę. Visos poerdvio bazės turi tą patį žodžių skaičių.*

Šis teiginys yra labai svarbus. Jis išaiškina tiesinių poerdvių struktūrą. Svarbūs teiginiai retai įrodomi bemaž be pastangų. Mums pakaks, kad galėsime juo naudotis. Kam rūpi įrodymas – atsiverskite kokią nors tiesinės algebros knygą.

57 teorema. *Tiesinio poerdvio $\mathbf{L}$ bazės žodžių skaičių vadinsime jo dimensija ir žymėsime $dim(\mathbf{L})$.*

Taigi norėdami nusakyti tiesinį poerdvį, galime tiesiog surašyti visus jo bazės žodžius. Tačiau yra ir kitas labai naudingas būdas tiesiniams poerdviams apibrėžti. Iš pradžių – dar viena operacija su žodžiais.

52 apibrėžimas. *Tegu $\mathbf{x} = x_1x_2\ldots x_n$, $\mathbf{y} = y_1y_2\ldots y_n$ yra du erdvės V_n žodžiai. Jų vidine sandauga vadinsime $\mathbb{F}_p$ elementą, apibrėžiamą lygybe*

$$\mathbf{x}\cdot\mathbf{y} = x_1y_1 + \ldots + x_ny_n.$$

Pasinaudosime šia sąvoka ir kiekvienam tiesiniam poerdviui $\mathbf{L}$ parinksime į porą kitą poerdvį.

58 teorema. *Tegu $\mathbf{L}$ yra tiesinis poerdvis. Žodžių aibė*

$$\mathbf{L}^\perp = \{\mathbf{y} \in V_n : \mathbf{x}\cdot\mathbf{y} = 0 \text{ su visais } \mathbf{x} \in \mathbf{L}\}$$

taip pat yra tiesinis poerdvis. Poerdvių dimensijos susijusios lygybe

$$dim\,(\mathbf{L}) + dim\left(\mathbf{L}^\perp\right) = n.$$

Tiesinį poerdvį $\mathbf{L}^\perp$ vadinsime dualiu poerdviui $\mathbf{L}$. Beveik akivaizdu, kad

$$\left(\mathbf{L}^\perp\right)^\perp = \mathbf{L}.$$

Tiesinį poerdvį $\mathbf{L}$ galime nusakyti naudodami jo bazę arba – jam dualaus poerdvio bazę.

59 teorema. *Tegu* $\mathbf{L}$ *yra tiesinis poerdvis,* $dim(\mathbf{L}) = k$, *o* $\mathbf{h}_1, \ldots, \mathbf{h}_{n-k}$ *yra dualaus poerdvio* $\mathbf{L}^\perp$ *bazė. Tada*

$$\mathbf{L} = \{\mathbf{x} \in V_n : \mathbf{x} \cdot \mathbf{h}_1 = \mathbf{x} \cdot \mathbf{h}_2 = \ldots = \mathbf{x} \cdot \mathbf{h}_{n-k} = 0\}. \tag{48}$$

Sąlygas (48) galime užrašyti tiesinėmis lygtimis. Iš tikrųjų, jeigu dualaus poerdvio bazės žodžiai yra

$$\mathbf{h}_1 = h_{11}h_{12}\ldots h_{1n}, \mathbf{h}_2 = h_{21}h_{22}\ldots h_{2n}, \ \ldots \ \mathbf{h}_{n-k} = h_{n-k,1}h_{n-k,2}\ldots h_{n-k,n},$$

tai (48) sąlygos reiškia, kad poerdvis $\mathbf{L}$ yra sudarytas iš tų žodžių $\mathbf{x}$, kurių komponentės $x_1x_2\ldots x_n$ tenkina tokias lygybes:

$$\begin{aligned}
&h_{11}x_1 + h_{12}x_2 + \ldots + h_{1n}x_n = 0,\\
&h_{21}x_1 + h_{22}x_2 + \ldots + h_{2n}x_n = 0,\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&h_{n-k,1}x_1 + h_{n-k,2}x_2 + \ldots + h_{n-k,n}x_n = 0.
\end{aligned}$$

8.3. Daugianariai

Nagrinėsime daugianarius su koeficientais iš baigtinio kūno. Kas gi yra tie daugianariai? Kažkas panašaus į aitvarų uodegas!

Fiksuokime pirminį skaičių p ir nagrinėkime daugianarius su koeficientais iš kūno $\mathbb{F}_p$:

$$f(x) = a_0 + a_1x + \ldots + a_nx^n, \quad a_i \in \mathbb{F}_p, \quad a_n \neq 0.$$

Daugianaris sudarytas sudėjus vienanarius. Didžiausio laipsnio vienanarį tokioje sumoje (t. y. turintį nenulinį koeficientą), vadinsime vyriausiuoju nariu, jo laipsnį – daugianario laipsniu. Daugianario laipsnį žymėsime

$$n = \deg(f(x)) \quad \text{arba } n = \deg(f).$$

Kartais žmonės klausia: kas gi yra tas daugianaris, kas gi yra tas x? Suprasti, kas yra daugianaris, jiems trukdo mokykloje įdiegta nuomonė, kad x yra nežinomas skaičius, kurį reikia rasti. Jeigu ir jums sunku įsivaizduoti, kas yra tas daugianaris, paklausykite tokio patarimo. Įsivaizduokite, kad skaičiai $a_0, a_1, \ldots, a_n$ užrašyti ant skrituliukų su skylutėmis per vidurį (kaip daniškos monetos!), įsivaizduokite, kad x – tai kaspinas su viena „ausele“, x^2

– kaspinas su dviem auselėm, x^3 – su trimis... Pagaliau įsivaizduokite, kad imate virvelę, prie kurios galo pririšate skritulėlį su a_0, kiek toliau – a_1 su „kaspinu“ x šalia, dar toliau – a_2 ir x^2... Baigę darbą, turėsite daugianarį – kažką panašaus į aitvaro uodegą. Gal ir nelabai tinka tvirtinti per egzaminą, kad daugianaris – tai aitvaro uodega, bet šitaip sau įsivaizduoti daugianarį teisinga visiškai!

Pažymėkime visų daugianarių ir ne didesnio kaip n-ojo laipsnio daugianarių aibes:

$$\begin{aligned}\mathbb{F}_p[x] &= \{a_0 + a_1x + \ldots + a_mx^m : a_i \in \mathbb{F}_p, a_m \neq 0, m \geq 0\},\\ \mathbb{F}_{p,n}[x] &= \{a_0 + a_1x + \ldots + a_mx^m : a_i \in \mathbb{F}_p, a_m \neq 0, m < n\}.\end{aligned}$$

Akivaizdu, kad

$$\mathbb{F}_p = \mathbb{F}_{p,0}[x] \subset \mathbb{F}_{p,1}[x] \subset \ldots \subset \mathbb{F}_{p,n}[x] \subset \ldots \subset \mathbb{F}_p[x].$$

Daugianarį galime dauginti iš kūno elemento $\alpha \in \mathbb{F}_p$, du daugianarius galima sudėti; abiejų veiksmų rezultatai – nauji daugianariai:

$$\begin{aligned}&f(x) = a_0 + a_1x + \ldots + a_nx^n,\\ &g(x) = b_0 + b_1x + \ldots + b_mx^m,\\ &(\alpha \cdot f)(x) = (\alpha a_0) + (\alpha a_1)x + \ldots + (\alpha a_n)x^n,\\ &(f+g)(x) = (a_0+b_0) + (a_1+b_1)x + \ldots + (a_k+b_k)x^k,\ k = \max(n,m);\end{aligned}$$

jei $n > m$, tai paskutinėje lygybėje imame $b_l = 0$, kai $l > m$; jei $n < m$, tai $a_l = 0$, kai $l > n$.

Atliekant veiksmus su daugianariais, laipsniai keičiasi taip:

$$\deg(\alpha \cdot f) = \deg(f)\ (\alpha \neq 0), \quad \deg(f+g) = \max(\deg(f), \deg(g)).$$

Šie du veiksmai paverčia daugianarių aibes geromis algebrinėmis struktūromis.

60 teorema. *Daugianarių aibės $\mathbb{F}_{p,n}[x], \mathbb{F}_p[x]$ yra tiesinės erdvės virš kūno $\mathbb{F}_p$.*

Tačiau daugianarius galime ir dauginti: tiesiog naudokimės įprastinėmis sudėties, daugybos asociatyvumo, komutatyvumo ir distributyvumo savybėmis ir laipsnių daugybos taisyklėmis $x^n \cdot x^m = x^{n+m}$.

Nesunku suvokti, kad teisinga tokia laipsnių kitimo taisyklė:

$$\deg(f \cdot g) = \deg(f) + \deg(g), \quad f, g \in \mathbb{F}_p[x].$$

Jeigu galima dauginti, tai galima bandyti ir dalyti. Ir čia mūsų laukia maloni staigmena: teisinga dalybos su liekana teorema.

61 teorema. *Tegu $f, g \in \mathbb{F}_p[x]$ yra du daugianariai, $\deg(g) < \deg(f)$. Tada egzistuoja vienintelė daugianarių pora $q, r \in \mathbb{F}_p[x]$, kad*

$$f(x) = q(x)g(x) + r(x), \quad 0 \leq \deg(r) < \deg(g). \tag{49}$$

Kaip ir sveikųjų skaičių atveju, daugianarį $q(x)$ vadinsime f dalybos iš g dalmeniu, o r – liekana.

Įrodymas. Tegu $\deg(f) = n, \deg(g) = m$ $(m \le n)$ ir

$$g(x) = b_0 + b_1 x + \ldots + b_m x^m, \quad b_m \neq 0.$$

Nagrinėkime baigtinę daugianarių aibę

$$\mathbf{D} = \{f(x) - h(x)g(x) : h(x) \in \mathbb{F}_p[x], \deg(h) \le n - m\}.$$

Šioje aibėje suraskime mažiausiojo laipsnio daugianarį $r(x) = f(x) - q(x)g(x)$. Įsitikinkime, kad $\deg(r) < m$. Iš tikrųjų, jeigu būtų

$$r(x) = f(x) - q(x)g(x) = c_0 + c_1 x + \ldots + c_k x^k, \quad k \ge m, c_k \neq 0,$$

tai aibei $\mathbf{D}$ priklausantis daugianaris

$$r_1(x) = f(x) - q(x)g(x) - (b_m^{-1} c_k) x^{k-m} g(x) = f(x) - q_1(x)g(x)$$

būtų mažesnio nei $r(x)$ laipsnio. Tai prieštarautų $r(x)$ parinkimui. Taigi bent viena pora, tenkinanti (49), egzistuoja. Tarkime, egzistuoja dvi skirtingos tokios poros, t. y.

$$f(x) = q(x)g(x) + r(x) = q_1(x)g(x) + r_1(x), \quad 0 \le \deg(r), \deg(r_1) < \deg(g).$$

Tada gautume

$$\begin{aligned} &(q(x) - q_1(x))g(x) = r(x) - r_1(x), \\ &\deg((q(x) - q_1(x))g(x)) \ge m, \quad \deg(r(x) - r_1(x)) < m. \end{aligned}$$

O tai jau prieštaravimas!

Jeigu dalydami daugianarį $f(x)$ iš $g(x)$, gauname nulinę liekaną, t. y. $r(x) = 0$, tai sakome, kad f dalijasi iš g, arba – g yra f daliklis. Tada

$$f(x) = q(x)g(x).$$

Akivaizdu, kad kiekvienas daugianaris dalijasi iš pats savęs ir iš nenulinių $\mathbb{F}_p$ elementų – nulinio laipsnio daugianarių:

$$f(x) = \alpha q(x), \quad q(x) = \alpha^{-1} f(x), \quad \alpha \neq 0.$$

Tokius $f(x)$ daliklius vadinsime trivialiaisiais.

Ar kiekvienas daugianaris turi netrivialiųjų daliklių? Anaiptol.

62 teorema. *Kiekvienam $n \ge 1$ yra n-ojo laipsnio daugianarių, neturinčių netrivialiųjų daliklių.*

Tokie daugianariai daugeliu savybių (bet ne visomis) panašūs į pirminius skaičius.

53 apibrėžimas. *Daugianarį $f \in \mathbb{F}_p[x]$, neturintį netrivialiųjų daliklių vadinsime neskaidžiu.*

Pavyzdžiui, visi pirmojo laipsnio daugianariai yra neskaidūs. Bet jie nėra patys įdomiausi! Yra bet kokio laipsnio neskaidžių daugianarių.

Jeigu du daugianariai f_1, f_2 dalijasi iš to paties daugianario

$$d(x) = a_k x^k + \ldots + a_1 x + a_0, \ (a_k \neq 0)$$

t. y. $f_1(x) = g_1(x)d(x), f_2(x) = g_2(x)d(x)$, tai jie dalijasi ir iš tam tikro daugianario su vyriausiuoju koeficientu, lygiu 1. Iš tikrųjų,

$$f_1(x) = (a_k g_1(x))(a_k^{-1} d(x)), \quad f_2(x) = (a_k g_2(x))(a_k^{-1} d(x))$$

ir daugianario $a_k^{-1}d(x)$ laipsnis lygus k, o vyriausiasis koeficientas lygus 1.

54 apibrėžimas. *Tegu $f_1, f_2 \in \mathbb{F}_p[x]$ yra du daugianariai. Jų didžiausiuoju bendruoju dalikliu vadiname didžiausiojo laipsnio daugianarį su vienetiniu vyriausiuoju koeficientu, iš kurio dalijasi ir f_1, ir f_2. Bendrąjį didžiausiąjį daliklį žymėsime (f_1, f_2).*

Jeigu f_1 dalijasi iš f_2, tai bendrasis didžiausiasis jų daliklis yra beveik f_2 – tereikia padauginti jį iš kūno elemento, kad vyriausiasis koeficientas taptų lygus 1.

Rasti bendrąjį didžiausiąjį daliklį – svarbus uždavinys, kurio sprendimu pasinaudosime ne kartą. O sprendimas gaunamas pritaikius Euklido algoritmą.

Tegu reikia rasti daugianarių $f_1, f_2 \in \mathbb{F}_p[x]$, $\deg(f_2) \leq \deg(f_1)$, bendrąjį didžiausiąjį daliklį. Padalykime f_1 iš f_2 su liekana:

$$f_1(x) = g_1(x)f_2(x) + r_1(x), \quad 0 \leq \deg(r_1) < \deg(f_2).$$

Euklido algoritmas remiasi sąryšiu, kurį nesudėtinga įrodyti:

$$(f_1, f_2) = (f_2, r_1).$$

Dalybos su liekana veiksmus galime tęsti, tačiau ne be galo:

$$\begin{aligned}
f_2(x) &= g_2(x)r_1(x) + r_2(x), \quad 0 < \deg(r_2) < \deg(r_1) \\
r_1(x) &= g_3(x)r_2(x) + r_3(x), \quad 0 < \deg(r_3) < \deg(r_2) \\
\ldots &\quad \ldots \\
r_{m-2}(x) &= g_m(x)r_{m-1}(x) + r_m(x), \quad 0 < \deg(r_m) < \deg(r_{m-1}) \\
r_{m-1} &= g_{m+1}(x)r_m(x) + 0.
\end{aligned}$$

Tada

$$(f_1, f_2) = (f_2, r_1) = (r_1, r_2) = \ldots = (r_{m-1}, r_m) = \alpha r_m.$$

Taigi daugianariai yra nauji objektai – aitvarų uodegos. Tačiau kiekvieną daugianarį

$$f(x) = a_0 + a_1 x + \ldots + a_n x^n$$

galime interpretuoti kaip funkciją $f : \mathbb{F}_p \to \mathbb{F}_p$, priskiriančią argumentams reikšmes taip:

$$\alpha \mapsto a_0 + a_1\alpha + \ldots + a_n\alpha^n.$$

Argumentui α priskiriamą reikšmę žymėsime $f(\alpha)$. Kartais toks žymėjimas gali sukelti dvejonių: ką žymi, pavyzdžiui, $f(y)$ – kūno elementą ar daugianarį? Pabandysime išvengti panašių dviprasmybių, daugianarius visada žymėdami $f(x)$.

Jei $\alpha \in \mathbb{F}_p, f(x) \in \mathbb{F}_p[x]$ ir $f(\alpha) = 0$, tai elementą α vadinsime daugianario $f(x)$ šaknimi. Nesudėtinga įsitikinti, kad tada $f(x)$ dalijasi iš $x - \alpha$, t. y.

$$f(x) = (x - \alpha)g(x), \quad g(x) \in \mathbb{F}_p[x].$$

Jeigu

$$f(x) = (x - \alpha)^k h(x), \quad k \geq 1, \quad h(\alpha) \neq 0,$$

tai sakysime, kad šaknies α kartotinumas yra lygus k arba kitaip – α yra k kartotinumo daugianario $f(x)$ šaknis. Dar galima sakyti, kad toks daugianaris turi k vienodų (lygių α) šaknų.

Svarbus teiginys apie daugianario šaknų skaičių.

63 teorema. *Daugianario $f \in \mathbb{F}_p[x]$ šaknų skaičius yra ne didesnis už jo laipsnį.*

Paprastai ši teorema įrodinėjama matematinės indukcijos metodu pagal daugianarių laipsnius.

8.4. Kūnų plėtiniai

Dalybos iš pirminio skaičiaus liekanų kūnai – ne vieninteliai baigtiniai kūnai. O kokių dar yra? Jei nežinote – perskaitykite šį skyrelį.

Pasirinkime kokį nors n-ojo laipsnio daugianarį

$$f(x) = a_0 + a_1 x + \ldots + a_n x^n, \quad a_i \in \mathbb{F}_p.$$

Dalydami kitus daugianarius $g \in \mathbb{F}_p[x]$ iš f, gausime liekanas. Šių liekanų aibė sutampa su ne didesnio kaip $n-1$-ojo laipsnio daugianarių aibe $\mathbb{F}_{p,n}[x]$.

Jeigu sudėsime dvi dalybos iš f liekanas – vėl gausime liekaną, t. y. $\mathbb{F}_{p,n}[x]$ elementą. Jeigu liekanas dauginsime – sandauga nebūtinai bus liekana. Tačiau, kaip ir dalybos iš skaičių liekanų atveju, daugianarių daugybos apibrėžimą galime pakeisti:

$$\text{jei } g, h \in \mathbb{F}_{p,n}[x], \text{ tai } g \times_f h \;=\; g \cdot h \text{ dalybos iš } f \text{ liekana.}$$

Naujosios liekanų daugybos rezultatas – vėl tos pačios aibės elementas, t. y. dalybos iš f liekana. Tikras malonumas naujoje aplinkoje pastebėti jau žinomą dėsnį:

64 teorema. *Tegu f yra n-ojo laipsnio daugianaris. Daugianarių aibė $\mathbb{F}_{p,n}[x]$ su sudėties ir daugybos veiksmais $+, \times_f$ sudaro žiedą.*

Dar maloniau rasti ne tik naujų žiedų, bet ir kūnų.

65 teorema. *Jeigu $f \in \mathbb{F}_p[x]$ yra neskaidus n-ojo laipsnio daugianaris, tai $\mathbb{F}_{p,n}[x]$ su sudėties ir daugybos veiksmais $+, \times_f$ sudaro kūną.*

Įrodymas. Kad daugianarių aibė su apibrėžtomis operacijomis sudaro žiedą, įrodyti nesunku. Pakanka nustatyti, kad sudėties ir daugybos veiksmai tenkina žiedo apibrėžimo savybes. Norėdami įrodyti, kad neskaidaus daugianario f atveju šis žiedas yra ir kūnas, turime įrodyti, kad kiekvienas nenulinis daugianaris g turi atvirkštinį, t. y. kiekvienam $g \in \mathbb{F}_{p,n}[x], g(x) \neq 0$, egzistuoja $h \in \mathbb{F}_{p,n}[x]$, kad

$$g(x) \times_f h(x) = 1.$$

Pasirinkime $g \in \mathbb{F}_{p,n}[x]$ ir nagrinėkime daugianarių aibę

$$\langle g \rangle = \{g(x) \times_f a(x) : a \in \mathbb{F}_p[x]\}.$$

Suraskime joje mažiausio laipsnio nenulinį daugianarį $r(x)$ (tokių yra ne vienas – galime pasirinkti). Tada galime užrašyti tokią daugianarių lygybę

$$g(x)a(x) = b(x)f(x) + r(x),\ a, b \in \mathbb{F}_{p,n}[x],\ \deg(r) < \deg(f) = n. \qquad (50)$$

Įrodysime, kad $r(x)$ būtinai yra nulinio laipsnio daugianaris, t. y. nenulinis kūno $\mathbb{F}_p$ elementas. Tarkime priešingai: $\deg(r) > 0$. Tada, padaliję $f(x)$ iš $r(x)$ su liekana, gautume

$$f(x) = q(x)r(x) + r_1(x), \quad 0 \leq \deg(r_1) < \deg(r).$$

Kadangi $f(x)$ yra neskaidus daugianaris, tai $r_1(x) \neq 0$. Padauginkime (50) iš $q(x)$ ir pasinaudokime f dalybos iš r išraiška:

$$\begin{aligned}
&g(x)a(x)q(x) = b(x)q(x)f(x) + r(x)q(x),\\
&g(x)a(x)q(x) = b(x)q(x)f(x) + f(x) - r_1(x),\\
&g(x)c(x) = d(x)f(x) - r_1(x),\ c(x) = a(x)q(x), d(x) = b(x)q(x) + 1,\\
&g(x) \times_f c(x) = -r_1(x),\ \deg(-r_1) < \deg(r).
\end{aligned}$$

Iš paskutinės lygybės išplaukia, kad $-r_1 \in \langle g \rangle$, t. y. šioje aibėje yra mažesnio laipsnio daugianaris negu r. Tai prieštarauja mūsų pasirinkimui, todėl $r(x)$ turi būti tiesiog kuris nors nenulinis kūno $\mathbb{F}_p$ elementas α. Tada iš (50) gauname:

$$\begin{aligned}
&g(x)a(x) = b(x)f(x) + \alpha,\\
&g(x)a(x)\alpha^{-1} = b(x)\alpha^{-1}f(x) + 1,\\
&g(x) \times_f h(x) = 1,\ h(x) = a(x)\alpha^{-1}.
\end{aligned}$$

Taigi $\mathbb{F}_{p,n}[x]$ su veiksmais $+, \times_f$ tikrai sudaro kūną.

Sudarytojo kūno elementus užrašome daugianariais. Jeigu jau yra kūnas, tai galima nagrinėti ir daugianarius su koeficientais iš to kūno. Kartais tenka svarstyti, kaipgi suvokti tekste nurodytą žymenį $f(x)$: kaip kūno elementą ar daugianarį su koeficientais iš to kūno? Bandydami išvengti tokių dviprasmybių, sukonstruoto kūno elementams žymėti naudosime reiškinius, gautus iš daugianarių, pakeitus simbolį x į α, o kartais tiesiog surašydami $\mathbb{F}_p$ elementų rinkinį: jei $g \in \mathbb{F}_{p,n}$, tai

$$g(x) = a_0 + a_1 x + \ldots + a_k x^k \mapsto a_0 + a_1\alpha + \ldots + a_k\alpha^k \mapsto (a_0, a_1, \ldots, a_{n-1});$$

čia $a_l = 0$, kai $l > k$.

Sukonstruotame naudojant neskaidų n-osios eilės daugianarį su koeficientais iš $\mathbb{F}_p$ kūne yra p^n elementų. Pats kūnas $\mathbb{F}_p$ yra naujojo kūno dalis, taigi teisinga šį kūną pavadinti kūno $\mathbb{F}_p$ plėtiniu. Tačiau yra ne vienas neskaidus n-ojo laipsnio daugianaris. Koks ryšys tarp kūnų, gaunamų naudojant skirtingus to paties laipsnio daugianarius? Jį paaiškinti galime pasinaudoję kūnų izomorfizmo sąvoka.

55 apibrėžimas. *Tegu K_1 ir K_2 yra du kūnai, kurių sudėties ir daugybos operacijas žymėsime $+_1, \times_1$ ir $+_2, \times_2$. Kūnus vadinsime izomorfiškais, jeigu egzistuoja abipus vienareikšmė atitiktis $\lambda : K_1 \to K_2$, su visais $a, b \in K_1$ tenkinanti sąlygas*

$$\lambda(a +_1 b) = \lambda(a) +_2 \lambda(b); \quad \lambda(a \times_1 b) = \lambda(a) \times_2 \lambda(b).$$

Izomorfiškus kūnus galime įsivaizduoti kaip objektus, sukonstruotus pagal tą patį „projektą“, t. y. vieno prototipo kopijas.

Štai teiginys, nustatantis baigtinių kūnų visumos savybes:

66 teorema. *Jeigu F yra baigtinis kūnas, tai jo elementų skaičius yra pirminio skaičiaus laipsnis, t. y.*

$$|F| = p^n, \quad p \text{ yra pirminis skaičius, } n \geq 1.$$

Visi tiek pat elementų turintys kūnai yra izomorfiški.

Taigi nesvarbu, kokį neskaidų n-ojo laipsnio daugianarį pasirinksime, sukonstruotas kūnas iš esmės bus tas pats. Todėl jį žymėsime tiesiog $\mathbb{F}_{p^n}$, arba $\mathbb{F}_q$, pabrėždami, kad q yra pirminio skaičiaus laipsnis. Kartais baigtiniai kūnai žymimi ir taip – $GF(q)$, šitaip primenant prancūzų matematiką E. Galois'ą, kurio darbai padarė labai didelę įtaką šiuolaikinės algebros raidai (GF - Galois fields, angl.).

Visi kūnai $\mathbb{F}_{p^n}$ yra pagrindinio kūno $\mathbb{F}_p$ plėtiniai, o koks jų tarpusavio ryšys?

67 teorema. *Jeigu natūrinis skaičius d dalija n, tai egzistuoja kūno $\mathbb{F}_{p^n}$ pokūnis (poaibis, sudarantis kūną), izomorfiškas $\mathbb{F}_{p^d}$.*

Taigi, pavyzdžiui, būtų teisinga rašyti

$$\mathbb{F}_3 \subset \mathbb{F}_{3^2} \subset \mathbb{F}_{3^4} \subset \mathbb{F}_{3^{12}},$$

tačiau $\mathbb{F}_{3^4} \subset \mathbb{F}_{3^6}$ – neteisinga.

Kūne $\mathbb{F}_p$ sudėję p vienetų, gauname nulį: $1+1+\ldots+1=0$; kadangi $\mathbb{F}_p \subset \mathbb{F}_{p^m}$ kiekvienam m, tai ši lygybė teisinga visuose $\mathbb{F}_p$ plėtiniuose. Jeigu sudėtume p kitų elementų, taip pat gautume nulį:

$$a+a+\ldots+a=(1+1+\ldots+1)\cdot a=0\cdot a=0, \quad a\in\mathbb{F}_{p^m}.$$

Šia savybe kartais pasinaudosime. Pavyzdžiui, iš jos išplaukia viena lygybė, kurią taikyti realiesiems skaičiams būtų tiesiog grubi klaida.

68 teorema. *Su bet kokiais elementais $\alpha,\beta\in\mathbb{F}_{p^m}$ teisinga lygybė*

$$(\alpha+\beta)^p=\alpha^p+\beta^p.$$

Iš tikrųjų, iš Niutono binomo formulės gautume

$$(\alpha+\beta)^p=\alpha^p+\sum_{i=1}^{n-1}C_p^i\alpha^i\beta^{p-1}+\beta^p.$$

Tačiau visi binominiai koeficientai C_p^i dalijasi iš p, todėl jie atitinkamus dėmenis paverčia nuliais.

Sukonstravę kūno $\mathbb{F}_p$ plėtinį $\mathbb{F}_q$ ($q=p^m$), galime vėl nagrinėti daugianarių su koeficientais iš naujojo kūno erdvę $\mathbb{F}_q[x]$. Visi veiksmai ir savybės, kurios tinka daugianariams su koeficientais iš $\mathbb{F}_p$, tinka ir erdvės $\mathbb{F}_q[x]$ daugianariams.

8.5. Generuojantys elementai

Dar vienas požiūris į baigtinį kūną: jį sudaro nulis ir vieno nenulinio elemento laipsnių aibė!

Pažymėkime visų nenulinių kūno $\mathbb{F}_q$ ($q=p^m$) elementų aibę $\mathbb{F}_q^*$. Šios aibės elementai daugybos veiksmo atžvilgiu sudaro grupę, šios grupės eilė lygi $|\mathbb{F}_q^*|=q-1=p^m-1$. Jeigu pasirinktume elementą $\alpha\in\mathbb{F}_q^*$ ir sudarytume jo laipsnius

$$\alpha,\ \alpha^2,\ \alpha^3,\ldots \tag{51}$$

gautume periodinę elementų seką. Šios sekos periodas – mažiausias natūralusis skaičius m su kuriu $\alpha^m=1$.

56 apibrėžimas. *Elemento $\alpha\in\mathbb{F}_q^*$ eile vadinamas mažiausias natūralusis skaičius m, su kuriuo $\alpha^m=1$. Elemento eilę žymėsime $o(\alpha)$.*

Algebroje įrodoma, kad bet kokio baigtinės grupės elemento eilė dalija grupės eilę. Taigi bet kokiam nenuliniam elementui α skaičius $q-1$ dalijasi iš $o(\alpha)$. Ne kiekviena baigtinė grupė turi elementą, kurio eilė yra pati didžiausia, t. y. lygi grupės eilei. Tačiau baigtiniams kūnams pasisekė.

69 teorema. *Aibėje $\mathbb{F}_q^*$ yra elementų, kurių eilės lygios $q-1$.*

Šie maksimalią eilę turintys elementai labai svarbūs baigtinių kūnų tyrimuose ir taikymuose.

57 apibrėžimas. *Aibės $\mathbb{F}_q^*$ elementą γ, kurio eilė lygi $q-1$, vadinsime generuojančiu (arba primityviuoju) kūno $\mathbb{F}_q$ elementu.*

Generuojančių elementų reikšmė yra akivaizdi: jeigu $\alpha=\gamma$ yra generuojantis elementas, tai (51) seka prasideda visų $\mathbb{F}_q^*$ elementų eile, kuri toliau kartojasi. Kitaip sakant, bet kuris nenulinis kūno elementas α gali būti užrašytas kaip generuojančio elemento laipsnis:

$$\alpha=\gamma^j, \quad 0\le j<q-1.$$

Kaip rasti generuojančius elementus? Iš teiginio apie grupės elementų skaičiaus dalumą iš elementų eilių iš karto išplaukia toks kriterijus:

70 teorema. *Jeigu kiekvienam $q-1$ dalikliui d $(d<q-1)$ elementas $\gamma\in\mathbb{F}_q^*$ tenkina sąlygą*

$$\gamma^d\neq 1,$$

tai γ – primityvusis elementas.

Pavyzdžiui, patikrinkime, ar $\gamma=2$ yra primityvusis elementas kūne $\mathbb{F}_{23}$. Kadangi $23-1=2\cdot 11$, tai pakaks apskaičiuoti tik du laipsnius:

$$2^2=4, \quad 2^{11}=2(2^5)^2=2\cdot 9^2=8,$$

taigi $\gamma=2$ yra primityvusis elementas.

Panagrinėkime sudėtingesnį pavyzdį. Pasirinkę neskaidų virš $\mathbb{F}_2$ daugianarį $f(x)=x^4+x+1$, sukonstruokime kūno $\mathbb{F}_{2^4}$ kopiją

0000	0	1000	α^3
0001	1	1001	α^3+1
0010	α	1010	$\alpha^3+\alpha$
0011	$\alpha+1$	1011	$\alpha^3+\alpha+1$
0100	α^2	1100	$\alpha^3+\alpha^2$
0101	α^2+1	1101	$\alpha^3+\alpha^2+1$
0110	$\alpha^2+\alpha$	1110	$\alpha^3+\alpha^2+\alpha$
0111	$\alpha^2+\alpha+1$	1111	$\alpha^3+\alpha^2+\alpha+1$

Primename, kad kūno elementų sudėtį galime atlikti visai paprastai – kaip dvejetainių žodžių sudėtį (arba daugianarių sudėtį sutvarkant gautąjį reiškinį), o daugybos veiksmo rezultatą galime gauti sudauginę reiškinius įprastu būdu ir suradę gautos sandaugos dalybos iš $\alpha^4+\alpha+1$ liekaną. Galima skaičiuojant naudotis lygybe $\alpha^4+\alpha+1=0$, arba jai ekvivalenčia lygybe $\alpha^4=\alpha+1$.

Taigi pabandykime surasti primityvųjį elementą. Išbandykime laimę su elementu α. Kadangi $q-1=15=3\cdot 5$, tai pakanka įsitikinti, kad laipsniai α^3 ir α^5 nelygūs 1. Dėl pirmojo laipsnio tai akivaizdu, o antrasis lygus

$$\alpha^5=\alpha^4\cdot\alpha=(\alpha+1)\cdot\alpha=\alpha^2+\alpha\neq 1.$$

Mums pasisekė. Galbūt taip yra visada – elementas α yra primityvusis elementas? Ne, taip būna ne visada.

58 apibrėžimas. *Jeigu $f(x)$ yra neskaidus virš kūno $\mathbb{F}_p$ daugianaris, o naudojantis juo sukonstruotame kūne $\mathbb{F}_{p^r}$ ($r=\deg(f)>1$) elementas α yra primityvusis, tai daugianaris $f(x)$ pats vadinamas primityviuoju.*

Taigi kūnuose, sukonstruotuose su primityviaisiais daugianariais, ilgai vargti ieškant primityviojo elemento netenka. Tačiau tenka pasidarbuoti ieškant paties primityviojo daugianario.

71 teorema. *Daugianarių erdvėje $\mathbb{F}_p[x]$ egzistuoja bet kokio laipsnio $r>1$ primityvieji daugianariai.*

r-ojo laipsnio daugianaris $f(x)$ yra primityvus tada ir tik tada, kai jis nėra jokio daugianario x^m-1 su $m<p^r-1$ daliklis.

Todėl visada galime sukonstruoti baigtinio kūno $\mathbb{F}_q$ kopiją, kad jo nenuliniai elementai būtų reiškiami laipsniais α^j, $j=0,1,\ldots,q-2$.

8.6. Minimalieji daugianariai

Kaip turint šaknį $\beta\in\mathbb{F}_{p^r}$, rasti daugianarį $f(x)\in\mathbb{F}_p[x]$, kad $f(\beta)=0$? Šio uždavinio sprendimas pravers konstruojant kodus, bet dar negreitai...

Jeigu sukonstravome kūno $\mathbb{F}_p$ plėtinį $\mathbb{F}_q$ ($q=p^m$), tai galime nagrinėti daugianarius su koeficientais iš šio kūno. Pažymėkime jų aibę $\mathbb{F}_q[x]$. Akivaizdu, kad $\mathbb{F}_p[x]\subset\mathbb{F}_q[x]$.

Kiekvieną šios aibės daugianarį $f=a_0+a_1x+\ldots+a_rx^r$ galime suvokti kaip funkciją $f:\mathbb{F}_q\to\mathbb{F}_q$:

$$\beta\mapsto a_0+a_1\beta+\ldots+a_r\beta^r=f(\beta),\quad \beta\in\mathbb{F}_q.$$

Jei $f(\beta)=0$, elementą β vadiname daugianario šaknimi. Tada daugianaris $f(x)$ dalijasi iš $x-\beta$. Jeigu jis dalijasi iš $(x-\beta)^m$, bet nesidalija iš $(x-\beta)^{m+1}$, sakome, kad daugianaris turi m vienodų (lygių β) šaknų, arba – šaknies β kartotinumas lygus m.

Visiems baigtiniams kūnams teisingas teiginys, kurį suformulavome kūnui $\mathbb{F}_p$.

72 teorema. *Daugianario $f \in \mathbb{F}_q[x]$ šaknų skaičius ne didesnis už daugianario laipsnį.*

Kadangi bet kurio nenulinio $f \in \mathbb{F}_q$ elemento β eilė dalija $q-1$, tai šis elementas yra daugianario $x^{q-1}-1$ šaknis. O apskritai visi, neišskiriant nei nulio, kūno elementai yra daugianario

$$g(x) = x^q - x$$

šaknys. Šio daugianario koeficientai yra 1 ir -1 (arba 1 ir $p-1$, jei norite), taigi $g(x) \in \mathbb{F}_p[x]$. Tačiau kas tinka visiems, nebūtinai geriausiai tinka kiekvienam. Kūno elementai gali būti ir mažesnio laipsnio daugianarių iš $\mathbb{F}_p[x]$ šaknys.

59 apibrėžimas. *Minimaliuoju elemento $\beta \in \mathbb{F}_q$ daugianariu vadinsime mažiausiojo laipsnio daugianarį $m_\beta(x)$ iš $\mathbb{F}_p[x]$ su vienetiniu vyriausiuoju koeficientu, kad $m_\beta(\beta) = 0$.*

Kadangi β yra daugianario $m_\beta(x)$ šaknis, tai šis daugianaris turi dalytis iš $x-\beta$, dalmuo bus daugianaris iš $\mathbb{F}_q[x]$:

$$m_\beta(x) = (x-\beta)g(x), \quad g(x) \in \mathbb{F}_q[x].$$

Jeigu $\beta \in \mathbb{F}_p$, tai $m_\beta(x) = x-\beta$; jeigu β yra primityvusis kūno $\mathbb{F}_q$ elementas, tai $m_\beta(x) = x^{q-1}-1$; o pats įdomumas – kitais atvejais.

Taigi norime ištirti, kaip surasti elemento $\beta \in \mathbb{F}_q$ minimalųjį daugianarį.

Galime galvoti visai paprastai. Jeigu kuriam laikui pamiršime elementų sandaugą, tai kūną $\mathbb{F}_{p^r}$ galime sutapatinti su tiesine erdve $\mathbb{F}_{p,r}[x]$ (ją sudaro ne didesnio kaip r-ojo laipsnio daugianariai). Ši aibė yra r-matė tiesinė erdvė virš $\mathbb{F}_p$, vadinasi, bet kuris $r+1$-o elemento rinkinys yra tiesiškai priklausomas. Elementus $1, \beta, \beta^2, \ldots, \beta^r$ galime interpretuoti kaip šios erdvės elementus. Vadinasi, turėtų būti $\mathbb{F}_p$ elementai $a_0, a_1, \ldots, a_r$, kad

$$a_0 + a_1\beta + a_2\beta^2 + \ldots + a_r\beta^r = 0.$$

Suradę tokių elementų rinkinį, kad paskutiniojo nenulinio koeficiento a_j numeris j būtų kuo mažiausias, gautume minimalųjį daugianarį

$$m_\beta(x) = a_0 + a_1x + a_2x + \ldots + a_jx^j.$$

Išbandykime šį skaičiavimų būdą su kūnu $\mathbb{F}_{2^4}$ ir $\beta = \alpha^2 + 1$; žr. kūno elementų lentelę ankstesniame skyrelyje. Tarkime, kūnas buvo sudarytas su primityviuoju daugianariu $f(x) = x^4 + x + 1$, taigi α yra primityvusis elementas, galime naudotis lygybe $\alpha^4 = \alpha + 1$.

$1 = 1$	$1 = 0001$
$\beta = \alpha^2 + \alpha$	$\beta = 0110$
$\beta^2 = \alpha^4 + \alpha^2 = \alpha^2 + \alpha + 1$	$\beta^2 = 0111$
$\beta^3 = (\alpha^2 + \alpha + 1)(\alpha^2 + \alpha) = 1$	$\beta^3 = 0001$
$\beta^4 = \beta$	$\beta^4 = 0110$

Dydžių β^3, β^4, matyt, neprireiks. Ieškokime koeficientų $a_0, a_1, a_2 \in \mathbb{F}_p$, kad būtų

$$a_0 \cdot 0001 + a_1 \cdot 0110 + a_2 \cdot 0111 = 0000.$$

Tikrai netruksime juos surasti: $a_0 = a_1 = a_2 = 1$. Taigi minimalus daugianaris yra $m_\beta(x) = 1 + x + x^2$.

Dabar minimaliuosius daugianarius panagrinėkime kitu požiūriu. Tegu elemento β minimalus daugianaris yra

$$m_\beta(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_j x^j, \text{ t. y. } m_\beta(\beta) = a_0 + a_1\beta + a_2\beta^2 + \ldots + a_j\beta^j = 0.$$

Pakėlę pastarąją lygybę laipsniu p ir pasinaudoję tuo, kad $(u+v)^p = u^p + v^p$ ir $w^p = w$, kai $w \in \mathbb{F}_p$, gausime

$$m_\beta(\beta^p) = a_0 + a_1\beta^p + a_2(\beta^p)^2 + \ldots + a_j(\beta^p)^j = 0.$$

Taigi elemento β minimalaus daugianario $m_\beta(x)$ šaknis yra ir β^p, ir visi sekos

$$\beta,\ \beta^p, (\beta^p)^p = \beta^{p^2},\ \beta^{p^3},\ \beta^{p^4}, \ldots \tag{52}$$

skaičiai. Tačiau ne visi jie skirtingi. Kiekgi jų yra?

Pažymėkime $b = o(\beta)$, taigi m yra mažiausias natūrinis skaičius, su kuriuo $\beta^b = 1$. Tada (52) bus tiek skirtingų narių, kiek skirtingų dalybos iš b liekanų duoda skaičiai $1, p, p^2, \ldots$ Šis skirtingų liekanų skaičius savo ruožtu lygus mažiausiam natūraliajam skaičiui m, su kuriuo

$$p^m \equiv 1 \pmod{b}.$$

Tada visi skaičiai $\beta, \beta^p, \beta^{p^2}, \ldots, \beta^{p^{m-1}}$ bus skirtingi, o daugianaris $m_\beta(x)$ dalysis iš sandaugos

$$(x - \beta)(x - \beta^p) \cdots (x - \beta^{p^{m-1}}).$$

Tiesa yra ta, kad minimalus daugianaris tiesiog lygus šiai sandaugai.

73 teorema. *Tegu $\beta \in \mathbb{F}_{p^r}$, b yra šio elemento eilė, o m yra mažiausias natūralusis skaičius, su kuriuo $p^m \equiv 1 \pmod{b}$. Tada minimalus elemento β daugianaris yra*

$$m_\beta(x) = (x - \beta)(x - \beta^p) \cdots (x - \beta^{p^{m-1}}).$$

Išbandykime naująjį minimalaus elemento skaičiavimo būdą, vėl imdami kūną $\mathbb{F}_{2^4}$ ir $\beta = \alpha^2 + 1$. Skaičiuodami jau įsitikinome, kad $b = 3$. Dabar iš $2^1 \equiv 2 \pmod 3$ ir $2^2 \equiv 1 \pmod 3$ gauname, kad $m = 2$. Taigi

$$m_\beta(x) = (x - \beta)(x - \beta^2) = (x - (\alpha^2 + \alpha))(x - (\alpha^2 + \alpha + 1)) = x^2 + x + 1.$$

Skaičiuoti reikia pasinaudojant lygybe $\alpha^4 = \alpha + 1$.

9 Tiesiniai kodai

Abėcėle kodų žodžiams sudaryti pasirinksime baigtinį kūną $\mathbb{F}_q$, $q = p^m$ yra koks nors pirminio skaičiaus laipsnis, dažniausiai – tiesiog pirminis skaičius, pavyzdžiui, $p = 2$. Tada žodžių aibė $\mathbb{F}_q^n$ yra tiesinė erdvė, joje galime nagrinėti įvairius tiesinius poerdvius.

Štai pagrindinis šio didelio skyriaus apibrėžimas:

60 apibrėžimas. *Tiesinį erdvės $\mathbb{F}_q^n$ žodžių poerdvį $\mathbf{L} \subset \mathbb{F}_q^n$ vadinsime tiesiniu kodu. Jeigu šio kodo dimensija yra k, o minimalus atstumas – d, sakysime, kad tai $[n, k, d]$ kodas.*

Taigi bet kokio kodo parametrus žymime (n, N, d), o $[n, k]$ ir $[n, k, d]$ – tik tiesinių kodų parametrus. Jei $\mathbf{L}$ yra abėcėlės $\mathbb{F}_q$ žodžių $[n, k, d]$ kodas, tai $|\mathbf{L}| = q^k$, o jo koeficientas

$$R(\mathbf{L}) = \frac{\log_q |\mathbf{L}|}{n} = \frac{k}{n}.$$

Prisiminkime, kad kiekvienas tiesinis poerdvis turi sau dualų, todėl tiesiniai kodai pasirodo poromis.

61 apibrėžimas. *Tiesiniam kodui $\mathbf{L} \subset \mathbb{F}_q^n$ dualiuoju kodu vadinsime tiesinį poerdvį $\mathbf{L}^\perp \subset \mathbb{F}_q^n$. Jeigu $\mathbf{L} = \mathbf{L}^\perp$, kodą $\mathbf{L}$ vadinsime savidualiu.*

Tiesinio poerdvio ir jam dualaus poerdvio dimensijos yra susijusios lygybe:

$$\dim(\mathbf{L}) + \dim(\mathbf{L}^\perp) = n.$$

Savidualaus kodo atveju $2\dim(\mathbf{L}) = n$, todėl savidualių kodų galime rasti tik lyginio ilgio žodžių aibėse.

Greitai pamatysime, kaip tiesinių kodų struktūra galima pasinaudoti koduojant duomenis ir taisant įvykusias klaidas. Būtų gerai, jeigu „gerųjų kodų“ sekoje, kurios egzistavimą garantuoja Shannono teorema, atsirastų ir tiesinių kodų... Tokie norai yra visiškai įgyvendinami. Galima įrodyti, kad Shannono kodų seką įmanoma sudaryti vien tik iš tiesinių kodų!

9.1. Kodavimas tiesiniais kodais

Tiek dėmesio skyrę baigtiniams kūnams ir tiesinėms erdvėms, pradedame naudotis jų teikiamais patogumais. Pirmiausia aptarsime kodavimą...

Tegu $\mathbf{L}$ yra tiesinis $[n, k]$ kodas. Kad jį apibrėžtume, nebūtina surašyti visus jo žodžius, kurių yra q^k. Pakanka nurodyti tuos žodžius $\mathbf{a}_1, \ldots, \mathbf{a}_k$, kurie sudaro $\mathbf{L}$ kaip tiesinio $\mathbb{F}_q^n$ poerdvio bazę.

62 apibrėžimas. *Tegu $\mathbf{L} \subset \mathbb{F}_q^n$ yra tiesinis $[n, k]$ kodas. Kūno $\mathbb{F}_q$ elementų $k \times n$ matricą G vadinsime generuojančia kodo $\mathbf{L}$ matrica, jei n ilgio žodžiai, gauti surašant matricos G eilučių elementus, sudaro kodo $\mathbf{L}$ bazę.*

Jeigu žinome $[n, k]$ kodo $\mathbf{L}$ generuojančią matricą G, tai visus kodo $\mathbf{L}$ žodžius ir tik juos gauname imdami generuojančios matricos eilučių kombinacijas:

$$\mathbf{L} = \{\mathbf{x}G : \mathbf{x} \in \mathbb{F}_q^k\}; \tag{53}$$

čia $\mathbf{x}G$ reiškia žodžio, kaip vektoriaus-eilutės, ir matricos sandaugą.

Panagrinėkime (53) sąryšį. Atvaizdis

$$\mathbf{x} \to \mathbf{x}G$$

apibrėžia abipusiškai vienareikšmę erdvės $\mathbb{F}_q^k$ ir kodo $\mathbf{L}$ žodžių atitiktį. Tad šį priskyrimą galime interpretuoti kaip šaltinio informacijos, pateikiamos erdvės $\mathbb{F}_q^k$ žodžiais, kodavimą kodo $\mathbf{L}$ žodžiais.

Tik šitokį kodavimo būdą ir naudosime šiame skyriuje. Tačiau kodo generuojanti matrica nėra vienintelė. Skirtingas generuojančias matricas atitinka skirtingi kodavimo būdai. Informacijos gavėjui turi būti pranešta, kokią generuojančią matricą naudoja siuntėjas. Tada pagal kodo žodį $\mathbf{y}$ gavėjas, pasinaudojęs sąryšiu $\mathbf{y} = \mathbf{x}G$, galės surasti šaltinio siųstą žodį $\mathbf{x}$.

Pavyzdys. Kodavimui naudojamas dvinaris tiesinis $[4, 3]$ kodas su generuojančia matrica

$$G = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}.$$

Tegu šaltinis perdavė tokį iš nulių ir vienetų sudarytą informacijos srautą:

$$110\ 010\ 001\ 111\ 101\ 010\ldots$$

Tada kanalu siųsime tokią simbolių seką:

$$1011\ 0111\ 1010\ 0001\ 0110\ 0111\ldots$$

Taigi kodavimas tiesiniais kodais – dalykas nesudėtingas. Tačiau, atitinkamai parinkus generuojančią matricą G, jį dar labiau galima suprastinti.

63 apibrėžimas. *Tegu G yra kūno $\mathbb{F}_q$ elementų $k \times n$ matavimų matrica. Elementariaisiais matricos G pertvarkiais vadinsime šiuos veiksmus:*

- *dviejų eilučių (arba stulpelių) keitimą vietomis;*
- *eilutės daugybą iš $f \in \mathbb{F}_q,\ f \neq 0$;*
- *eilutės keitimą jos bei kitos eilutės suma;*
- *stulpelio daugybą iš $f \in \mathbb{F}_q,\ f \neq 0$.*

Jei matrica G yra tiesinio $[n,k]$ kodo $\mathbf{L}$ generuojanti matrica, o matrica G' gaunama iš G, atlikus elementariųjų pertvarkių seką, tai G' yra taip pat tam tikro tiesinio $[n,k]$ kodo $\mathbf{L}'$ generuojanti matrica.

74 teorema. *Tegu G yra tiesinio kodo $\mathbf{L}$ generuojanti matrica, o G' yra matrica, gauta iš G, atlikus elementariųjų jos pertvarkių seką. Tada G' yra kodo, ekvivalentaus $\mathbf{L}$, generuojanti matrica.*

Įsitikinti, kad šis teiginys teisingas, galima prisiminus ekvivalenčiųjų kodų apibrėžimą ir supratus, kaip keičiasi kodas, kai atliekame vieną jo generuojančios matricos pertvarkį.

Dabar prisiminkime tiesinės algebros patirtį. Jei G yra $[n,k]$ kodo generuojanti matrica, tai atitinkamais elementariaisiais pertvarkiais iš jos galima gauti tokio pavidalo matricą:

$$G' = \begin{pmatrix} 1 & 0 & \dots & 0 & a_{1,1} & \dots & a_{1,n-k} \\ 0 & 1 & \dots & 0 & a_{2,1} & \dots & a_{2,n-k} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & a_{k,1} & \dots & a_{k,n-k} \end{pmatrix} = (I_k, A); \qquad (54)$$

čia: I_k yra vienetinė $k \times k$ matrica, A – kūno $\mathbb{F}_q$ elementų $k \times (n-k)$ matrica. Susitarsime sakyti, jog (54) matrica yra **standartinio pavidalo**.

Iš teoremos apie generuojančios matricos pertvarkius galime daryti tokią išvadą:

75 teorema. *Kiekvienas $[n,k]$ kodas yra ekvivalentus $[n,k]$ kodui, turinčiam standartinio pavidalo generuojančią matricą.*

Bet kokį tiesinį kodą galime pakeisti jam ekvivalenčiu kodu, turinčiu standartinio pavidalo generuojančią matricą. Todėl pakanka nagrinėti tik tokius kodus. Pastebėkime, jog kodavimo algoritmą, kai naudojama standartinio pavidalo generuojanti matrica $G = (I_k, A)$, galima užrašyti šitaip:

$$\mathbf{x} \to \mathbf{xy}, \quad \mathbf{y} = \mathbf{x}A,$$

taigi koduojami žodžiai tiesiog pailginami pridedant $n-k$ kontrolinių klaidoms taisyti skirtų simbolių. Toks kodavimas, kai koduojamų simbolių žodis tiesiog pailginamas pridedant papildomus klaidoms taisyti reikalingus simbolius, vadinamas **sisteminiu**. Sisteminio kodavimo atveju, mažiau vargo turi ir informacijos gavėjas. Kad nustatytų, kokie simboliai reiškia tikrąją informaciją, jam pakanka sutrumpinti kodo žodį, nubraukiant pridėtus simbolius.

Taigi veiksmai su žodžiais suteikia galimybę naudoti efektyvesnius kodavimo algoritmus. Tai tik vienas iš daugelio tiesinių kodų privalumų. Štai dar vienas – efektyviau negu bendruoju atveju galima skaičiuoti tiesinio kodo minimalų atstumą.

64 apibrėžimas. *Žodžio* $\mathbf{x} \in \mathbb{F}_q^n,\ \mathbf{x} = x_1 \dots x_n$, *svoriu vadinsime skaičių*

$$w(\mathbf{x}) = \sum_{x_i \neq 0} 1.$$

Žodžio svoris – tiesiog nenulinių jo komponenčių skaičius. Tik vienas žodis „nieko nesveria“, t. y. jo svoris lygus nuliui: $w(00\dots0) = 0$; kitų žodžių svoriai ne mažesni už 1.

76 teorema. *Tegu* d *yra tiesinio kodo* $\mathbf{L}$ *minimalus atstumas. Tada*

$$d = \min\{w(\mathbf{x}) : \mathbf{x} \in \mathbf{L}, \mathbf{x} \neq 00\dots0\}.$$

Įrodymas. Tegu $\mathbf{0} = 00\dots0$ – nulinis žodis; h kaip ir anksčiau žymime Hammingo atstumą. Tada bet kokiam $\mathbf{x} \in \mathbf{L},\ \mathbf{x} \neq 00\dots0$,

$$d \leq h(\mathbf{0}, \mathbf{x}) = w(\mathbf{x}), \quad \text{todėl} \quad d \leq \min\{w(\mathbf{x}) : \mathbf{x} \in \mathbf{L}, \mathbf{x} \neq 00\dots0\}.$$

Tegu dabar $d = h(\mathbf{x}, \mathbf{y}),\ \mathbf{x}, \mathbf{y} \in \mathbf{L}$. Kadangi $\mathbf{x} - \mathbf{y} \in \mathbf{L}$, tai

$$\begin{aligned} d &= h(\mathbf{x}, \mathbf{y}) = h(\mathbf{0}, \mathbf{x} - \mathbf{y}) = w(\mathbf{x} - \mathbf{y}),\ \mathbf{x} - \mathbf{y} \neq \mathbf{0}, \\ d &\geq \min\{w(\mathbf{z}) : \mathbf{z} \in \mathbf{L}, \mathbf{z} \neq 00\dots0\}. \end{aligned}$$

Teorema įrodyta.

Jei kodas turi N žodžių, tai, ieškant jo minimalaus atstumo tiesioginės perrankos būdu, gali tekti peržiūrėti $N(N-1)/2$ žodžių porų. Ši teorema teigia, kad tiesinių kodų atveju pakanka „pasverti“ $N-1$ žodžių.

9.2. Kontrolinė tiesinio kodo matrica

Kartais aibę apibrėžiame išvardydami elementus, kurie jai priklauso, kartais – formuluodami reikalavimus, kuriuos turi tenkinti elementai, kad juos priimtume į aibę. Tai tinka ir tiesiniams kodams. Abu būdai turi savų privalumų...

Prisiminkime, kad tiesiniai poerdviai – taigi ir kodai – pasirodo poromis. Tegu $\mathbf{L} \subset \mathbb{F}_p^n$ yra tiesinis $[n, k]$ kodas; tada jam dualaus kodo $\mathbf{L}^\perp$ parametrai yra $[n, n-k]$.

Tegu $\mathbf{g}_1, \mathbf{g}_2, \dots, \mathbf{g}_k$ yra kodo $\mathbf{L}$ bazė, o $\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_{n-k}$ yra kodo $\mathbf{L}^\perp$ bazė. Tada $k \times n$ matrica G, gauta surašius žodžių $\mathbf{g}_i$ elementus į eilutes, bus generuojanti kodo G matrica, o $(n-k) \times n$ matrica H, gauta iš $\mathbf{h}_i$ eilučių – generuojanti H matrica.

Kodą $\mathbf{L}$ sudaro jo bazės žodžių tiesinės kombinacijos:

$$\mathbf{L} = \{\mathbf{x}G : \mathbf{x} \in \mathbb{F}_p^k\},$$

tačiau jį galime apibrėžti ir naudodami dualaus kodo bazę:

$$\mathbf{L} = \{\mathbf{x} : \mathbf{x} \in \mathbb{F}_p^n, \mathbf{x} \cdot \mathbf{h}_j = 0, j = 1, 2, \ldots, n-k\}. \tag{55}$$

Pastebėkime, kad visas (55) lygybes, kurias turi tenkinti kodo žodis $\mathbf{x}$, galime užrašyti taip:

$$\mathbf{x}H^T = O_{1,n-k},$$

čia $O_{1,n-k}$ yra eilutė, sudaryta iš $n-k$ nulių; ją galime interpretuoti kaip nulinį erdvės $\mathbb{F}_p^{n-k}$ žodį.

65 apibrėžimas. *Tegu* $\mathbf{L}$ *yra tiesinis* $[n, k]$ *kodas.* $(n-k) \times n$ *matricą* H*, kuri tenkina sąlygą*

$$\mathbf{L} = \{\mathbf{x} : \mathbf{x}H^T = O_{1,n-k}\},$$

vadinsime kodo $\mathbf{L}$ *kontroline matrica.*

Kontrolinės tiesinio kodo $\mathbf{L}$ matricos – tai dualaus kodo $\mathbf{L}^\perp$ generuojančios matricos; generuojančios kodo $\mathbf{L}$ matricos yra kontrolinės $\mathbf{L}^\perp$ matricos.

Tiesinis kodas yra visiškai apibrėžtas, jeigu nurodyta arba jo generuojanti arba kontrolinė matrica.

Pavyzdys. Sudarykime matricą iš vienos eilutės:

$$H = (h_1\ h_2 \ldots h_n), \quad h_j \in \mathbb{F}_p, \quad h_j \neq 0.$$

Ši matrica yra kontrolinė $[n, n-1]$ kodo $\mathbf{K}$ matrica. Kodui priklauso tie žodžiai $\mathbf{x} = x_1, x_2, \ldots, x_n$, kurie tenkina lygybę

$$h_1x_1 + h_2x_2 + \ldots + h_nx_n = 0. \tag{56}$$

Kodavimas šiuo kodu – tai informacinių simbolių eilutės papildymas dar vienu simboliu (kontroliniu):

$$x_1x_2 \ldots x_{n-1} \mapsto x_1x_2 \ldots x_{n-1}x_n, \quad x_n = -(h_1h_n^{-1}x_1 + \ldots + h_{n-1}h_n^{-1}x_{n-1}).$$

Nesunku nustatyti, kad minimalus kodo $\mathbf{K}$ atstumas yra $d = 2$, taigi jis negali ištaisyti nei vienos klaidos. Tačiau visada gali vieną klaidą aptikti! Tokį kodą pavadinkime **kodu su kontroliniu simboliu**.

Pavyzdžiai. Brūkšniniai kodai – tai kodai su kontroliniu simboliu. Juodų ir baltų juostų seka užrašomi atitinkamos abėcėlės simboliai. Pavyzdžiui, knygų žymėjimo sistema ISBN – tai kodas su abėcėle

$$\mathcal{A} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, X\},$$

čia X žymi skaičių 10. Informacija apie knygą užrašoma devyniais šios abėcėlės simboliais, suskirstytais į tris grupes: pirmoji simbolių grupė žymi šalį,

antroji – leidyklą, trečioji – knygą. Devynių simbolių žodis papildomas dešimtuoju – kontroliniu. Kontrolinė lygybė tokia:

$$X \cdot x_1 + 9 \cdot x_2 + 8 \cdot x_3 + 7 \cdot x_4 + 6 \cdot x_5 + \ldots + 2 \cdot x_9 + 1 \cdot x_1 \equiv 0 \pmod{11}.$$

Pavyzdžiui, ISBN - 9986-16-180-0 yra knygos, išleistos Lietuvoje (9986 – Lietuvos kodas), „Tyto alba“ leidykloje (leidyklos kodas – 16).

Nuo 2007 metų išleidžiamos knygos bus žymimos abėcėlės

$$\mathcal{A} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

trylikos skaitmenų kodu. Taip sistemoje EAN (European Article Numeration) ženklinamos prekės. Kontrolinė lygybė

$$1 \cdot x_1 + 3 \cdot x_2 + \ldots + 3 \cdot x_{12} + 1 \cdot x_{13} \equiv 0 \pmod{10}.$$

Kodo, dualaus kodui su kontroliniu simboliu parametrus taip pat nesunku nustatyti. Jie tokie – $[n, 1, n]$.

Kaip, žinant generuojančią $[n, k]$ kodo matricą G, surasti kontrolinę matricą H? Pastebėkime, kad iš kontrolinės matricos apibrėžimo išplaukia toks generuojančios ir kontrolinės matricų ryšys:

$$G \cdot H^T = O_{k,n-k}. \tag{57}$$

Taigi jei $k \times n$ matrica G yra kodo generuojanti matrica, tai bet kuri $(n-k) \times n$ matrica, sudaryta iš tiesiškai nepriklausomų eilučių ir tenkinanti (57), bus šio kodo kontrolinė matrica.

Kontrolinę matricą paprasta surasti, jeigu kodas turi standartinio pavidalo generuojančią matricą.

77 teorema. *Tegu $G = (I_k, A)$ yra tiesinio $[n, k]$ kodo $\mathbf{L} \subset \mathbb{F}_p^n$ generuojanti matrica. Tada matrica*

$$H = (-A^T, I_{n-k})$$

yra kontrolinė šio kodo matrica.

Norint įrodyti šį teiginį, reikia įsitikinti, kad teisinga lygybė

$$G \cdot H^T = (I_k, A) \cdot \begin{pmatrix} -A \\ I_{n-k} \end{pmatrix} = O_{k,n-k}.$$

Patarimas toks: užsirašykite matricas, pavyzdžiui, kai $k = 3, n = 5$ ir patikrinkite lygybę. Kaipmat suprasite, kur „šuo pakastas“!

Minimalų tiesinio kodo atstumą galima surasti peržiūrėjus kodo žodžių svorius. Tačiau yra dar paprastesnis būdas!

78 teorema. *Tegu H yra tiesinio kodo $\mathbf{L}$ kontrolinė matrica. Jeigu egzistuoja d tiesiškai priklausomų H stulpelių, o bet kuri $d-1$ šios matricos*

stulpelių sistema yra tiesiškai nepriklausoma, tai minimalus kodo atstumas lygus d.

Įrodymas. Tegu $\mathbf{x}$ yra kodo $\mathbf{L}$ žodis, $w(\mathbf{x}) = d$, čia d yra minimalus kodo atstumas. Toks žodis tikrai egzistuoja. Kadangi H yra kontrolinė kodo matrica, tai

$$\mathbf{x}H^T = \mathbf{0}.$$

Iš šios lygybės išplaukia, jog matrica H^T turi d tiesiškai priklausomų eilučių, o H – d tiesiškai priklausomų stulpelių.

Ir atvirkščiai: jei H turi w tiesiškai priklausomų stulpelių, tada egzistuoja žodis $\mathbf{x} \in \mathbb{F}_q^n$, jog $w(\mathbf{x}) = w$ ir $\mathbf{x}H^T = \mathbf{0}$. Taigi $\mathbf{x} \in \mathbf{L}$. Vadinasi, minimalus teigiamas kodo žodžių svoris (kartu ir minimalus kodo atstumas) sutampa su mažiausiu tiesiškai priklausomų kontrolinės matricos stulpelių skaičiumi.

9.3. Sindromai, lyderiai ir dekodavimas

Gautų iš kanalo žodžių dekodavimas – tarsi ligonio gydymas. Pirmiausia nustatomi ligos požymiai (sindromai), tada paskiriamas vaistas. Panagrinėkime, kaip tai daroma.

Geriau nežinoti, nei sužinoti netiesą. Trynimo klaidas lengviau taisyti nei iškraipymus. Tarkime, kodavimui naudojamas $[n, k]$ tiesinis kodas $\mathbf{C} \subset \mathbb{F}_q^n$ su kontroline $(n-k) \times n$ matavimų matrica H, kurios elementus žymėsime h_{ij}. Jeigu $\mathbf{c} = c_1c_2 \ldots c_n$ yra kodo žodis, tai $\mathbf{c}H^T = \mathbf{0}$. Šią matricinę lygybę galime užrašyti kaip lygybių sistemą

$$\begin{aligned} h_{11}c_1 + h_{12}c_2 + \ldots + h_{1n}c_n &= 0, \\ h_{21}c_1 + h_{22}c_2 + \ldots + h_{2n}c_n &= 0, \\ &\ldots\ldots\ldots \\ h_{n-k,1}c_1 + h_{n-k,2}c_2 + \ldots + h_{n-k,n}c_n &= 0. \end{aligned}$$

Tarkime, kad perdavimo kanalas simbolių neiškraipo, tačiau gali ištrinti. Tada vietoj žodžio $\mathbf{c} = c_1c_2c_3 \ldots c_n$ gausime, pavyzdžiui, $\mathbf{d} = ?c_2? \ldots c_n$. Dekodavimo, t. y. ištrintų simbolių ieškojimo, metodas kone akivaizdus – perduotas simbolių reikšmes reikia įstatyti į sistemos lygybes, o ištrintų simbolių vietoje – nežinomuosius; pavyzdyje – $x_1, x_3, \ldots$ Šitaip gausime tiesinių lygčių sistemą. Jeigu ji turi vienintelį sprendinį, trynimo klaidas ištaisysime.

Iškraipymo klaidas taisyti sunkiau. Aptarkime tiesinių kodų dekodavimą, kai kanalas iškraipo simbolius. Dekoduodami taikysime minimalaus atstumo taisyklę.

Tegu $\mathbf{L} \subset \mathbb{F}_q^n$ yra tiesinis $[n, k]$ kodas. Suskaidysime žodžių erdvę $\mathbb{F}_q^n$ sluoksniais

$$\mathbf{L}_\mathbf{x} = \mathbf{x} + \mathbf{L} = \{\mathbf{x} + \mathbf{c} : \mathbf{c} \in \mathbf{L}\}, \quad \mathbf{x} \in \mathbb{F}_q^n.$$

Aibės $\mathbf{L_x}, \mathbf{L_y}$ iš tikrųjų yra „sluoksniai“, t. y. skirtingiems $\mathbf{x}, \mathbf{y} \in \mathbb{F}_q^n$ jos arba nesikerta, arba sutampa. Pažymėkime sluoksnių aibę

$$\mathbb{F}_q^n/\mathbf{L} = \{\mathbf{L_x} : \mathbf{x} \in \mathbb{F}_q^n\}.$$

Pastebėkime, kad tų sluoksnių yra lygiai tiek: $|\mathbb{F}_q^n/\mathbf{L}| = q^{n-k}$.

Tarkime, informacija koduojama kodu $\mathbf{L}$. Tegu siunčiant žodį $\mathbf{c}$, kitame kanalo gale buvo gautas žodis $\mathbf{x}$, kuris galbūt skiriasi nuo siųstojo. Taikydami minimalaus atstumo taisyklę, šį žodį dekoduosime kodo žodžiu $\mathbf{c}$, kuris tenkina sąlygą

$$h(\mathbf{c}, \mathbf{x}) = w(\mathbf{x} - \mathbf{c}) = \min_{\mathbf{c}' \in \mathbf{L}} w(\mathbf{x} - \mathbf{c}').$$

Tačiau žodis $\mathbf{a} = \mathbf{x} - \mathbf{c}$ yra kuriame nors sluoksnyje $\mathbf{L_b}$ – tame pačiame kaip $\mathbf{x}$. Vadinasi, dekoduojant reikia peržiūrėti sluoksnį $\mathbf{L_b}$, kuriame atsidūrė gautas žodis $\mathbf{x}$, rasti jame mažiausią svorį turintį elementą $\mathbf{a}$ ir dekoduoti taip:

$$\mathbf{x} \to f(\mathbf{x}) = \mathbf{x} - \mathbf{a}.$$

Dekodavimui reikia pasirengti. Sunumeruokime kodo $\mathbf{L}$ žodžius taip, kad žodis $\mathbf{0} = 00\ldots0$ būtų pirmas: $\mathbf{L} = \{\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_N\}, \mathbf{c}_0 = \mathbf{0}, N = q^k - 1$. Visus $\mathbb{F}_q^n$ elementus surašysime tokioje matricoje-lentelėje:

$$\begin{pmatrix} \mathbf{a}_0 & \mathbf{c}_1 & \mathbf{c}_2 & \ldots & \mathbf{c}_N \\ \mathbf{a}_1 & \mathbf{a}_1 + \mathbf{c}_1 & \mathbf{a}_1 + \mathbf{c}_2 & \ldots & \mathbf{a}_1 + \mathbf{c}_N \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{a}_s & \mathbf{a}_s + \mathbf{c}_1 & \mathbf{a}_s + \mathbf{c}_2 & \ldots & \mathbf{a}_s + \mathbf{c}_N \end{pmatrix}. \tag{58}$$

Pirmojo stulpelio žodžius $\mathbf{a}_i$ parenkame taip, kad būtų patenkintos sąlygos:

$$\mathbf{a}_0 = \mathbf{0},\ w(\mathbf{a}_i) = \min \left\{ w(\mathbf{a}) : \mathbf{a} \in \mathbb{F}_q^n, \mathbf{a} \notin \bigcup_{j<i} \mathbf{L}_{\mathbf{a}_j} \right\},\ j \geq 1.$$

Šis (58) matricos sudarymo būdas garantuoja, jog kiekvienoje eilutėje bus surašyti atitinkamos klasės $\mathbf{L_a}$ elementai, o pirmasis iš jų turės mažiausią svorį. Matricą (58) vadinsime **standartine** kodo $\mathbf{L}$ **lentele**, o žodžius $\mathbf{a}_i$ – atitinkamų klasių **lyderiais**.

Turint standartinę kodo lentelę, dekodavimo algoritmą galima aprašyti taip:

- randame, kurioje standartinės lentelės eilutėje yra gautasis žodis $\mathbf{x}$;
- randame šios eilutės lyderį $\mathbf{a}$ ir dekoduojame $\mathbf{x}$ žodžiu $f(\mathbf{x}) = \mathbf{x} - \mathbf{a}$.

Ar, naudojantis tokiu algoritmu, bus visada dekoduojama teisingai? Ar visi sluoksniai turi tik po vieną lyderį? Nebūtinai. Tačiau jeigu kodo minimalus atstumas yra d ir perduodant įvykusių klaidų skaičius ne didesnis už $t = \left[\frac{d-1}{2}\right]$, tai iš kanalo gautas žodis būtinai pateks į tokį sluoksnį, kuriame lyderis tik vienas ir dekoduojama bus teisingai. Galime sumažinti standartinę kodo lentelę, išbraukydami tuos sluoksnius, kurie turi ne po vieną lyderį. Tada kiekvienam gautam iš kanalo žodžiui turėtume dvi galimybes: arba jis priklausytų vienam iš sluoksnių, arba nepriklausytų nei vienam. Antruoju atveju žinotume, kad minimalaus atstumo taisyklė neduoda vienareikšmio atsakymo.

Galime įsivaizduoti, kad iškraipymai kanale įvyksta taip:

$$\mathbf{c} \mapsto \mathbf{c} + \mathbf{e}, \quad \mathbf{c} \in \mathbf{L}, \quad \mathbf{e} \in \mathbb{F}_q^n,$$

čia $\mathbf{e}$ yra klaidų žodis.

Jeigu standartinėje kodo lentelėje palikome tik tuos sluoksnius, kurie turi po vieną lyderį, tai pirmajame jos stulpelyje bus surašyti visi klaidų žodžiai $\mathbf{e}$, kuriems įvykus dekoduojama teisingai. Taigi lyderiai yra tas pats kaip ištaisomų klaidų žodžiai. Visi kiti lentelės stulpeliai sudaryti iš žodžių, kurie dekoduojant keičiami kodo žodžiu, užrašytu stulpelio viršuje. Aibės, sudarytos iš vieno stulpelio žodžių, vadinamos dekodavimo sritimis. Galima įsivaizduoti, kad dekodavimo sritis yra tarsi atitinkamo kodo žodžio aplinka ar traukos sritis...

Dekodavimą, naudojantis standartine lentele, galima dar labiau suprastinti. Išties, jeigu rastume būdą nustatyti, kurioje lentelės eilutėje yra gautas žodis, pakaktų pasinaudoti tik pirmuoju standartinės lentelės stulpeliu.

Tegu H yra kontrolinė kodo $\mathbf{L}$ matrica. Jei $\mathbf{c}$ yra kodo $\mathbf{L}$ žodis, tai $\mathbf{c}H^T = \mathbf{0}$. Jei $\mathbf{x} = \mathbf{a}_j + \mathbf{c}$, tai $\mathbf{x}H^T = \mathbf{a}_j H^T$. Taigi sandaugos $\mathbf{x}H^T, \mathbf{x} \in \mathbb{F}_q^n$, reikšmė priklauso tik nuo to, kuriai aibės $\mathbb{F}_q^n/\mathbf{L}$ klasei priklauso $\mathbf{x}$.

66 apibrėžimas. *Tegu H yra $[n,k]$ kodo $\mathbf{L}$ kontrolinė matrica, $\mathbf{x} \in \mathbb{F}_q^n$. Žodžio $\mathbf{x}$ sindromu vadinsime $\mathbb{F}_q^{n-k}$ elementą $s(\mathbf{x}) = \mathbf{x}H^T$.*

Skirtingiems aibės $\mathbb{F}_q^n/\mathbf{L}$ sluoksniams priklausančių žodžių sindromai yra skirtingi. Taigi graikiškos kilmės žodis „sindromas“, kuris gydytojams reiškia ligos požymius, kodavimo teorijoje turi panašią prasmę – jis „nusako“ iškraipymų pobūdį.

Kadangi skirtingiems sluoksniams priklausančius žodžius atitinka skirtingi sindromai, tad dekoduojant pagal minimalaus atstumo taisyklę, pakanka turėti prieš akis tokią lentelę:

Sindromai	$\mathbf{s}_1$	$\mathbf{s}_2$	...	$\mathbf{s}_m$
Lyderiai	$\mathbf{a}_1$	$\mathbf{a}_2$	...	$\mathbf{a}_m$.

Dabar pirmąją užrašyto dekodavimo algoritmo dalį galime formuluoti taip:

- *randame gautojo žodžio sindromą.* Sindromui rasti pakanka mokėti padauginti vektorių iš kontrolinės matricos.

Jeigu minimalus kodo atstumas yra d, tai visi klaidų žodžiai, priklausantys rutuliui $B(\mathbf{0}, t)$, čia $\mathbf{0} = 00\ldots0, t = [(d-1)/2]$, yra ištaisomi. Tačiau gali būti ištaisomi ir kai kurie kiti klaidų žodžiai, t. y. dekodavimo srityse gali būti daugiau žodžių negu rutulyje $B(\mathbf{0}, t)$.

Pavyzdys. Nagrinėkime tiesinį kodą iš dvejetainės abėcėlės žodžių, kurio generuojanti matrica yra

$$G = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}.$$

Kodo kontrolinė matrica yra

$$H = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Iš kontrolinės matricos matyti, kad minimalus kodo atstumas yra $d = 2$ ir kodas negarantuoja, kad bus ištaisyta ir viena klaida. Tačiau klaidų žodžių, kurie teisingai ištaisomi, yra. Štai jie:

$$100000,\ 001000,\ 000100,\ 000001.$$

Taigi šio kodo dekodavimo sritis sudaro žodžių penketai.

Jeigu, siunčiant žodį kanalu, jame įvykusių klaidų skaičius yra didesnis negu garantuotai taisomų klaidų kiekis, tai galimi trys atvejai:

- klaidos liks nepastebėtos;
- klaidos bus pastebėtos ir ištaisytos;
- klaidos bus pastebėtos, bet neištaisytos.

Tegu žodžiai siunčiami kanalu, neturinčiu atminties, kuris kiekvieną simbolį iškreipia su tikimybe p. Kokia tikimybė, kad, siunčiant tiesinio kodo $\mathbf{L} \subset \mathbb{F}_q^n$ žodį (pavyzdžiui, nulinį žodį $\mathbf{0}$), įvykusios klaidos bus nepastebėtos?

Gavę žodį, klaidos nepastebėsime, jeigu jis bus kodo žodis:

$$\mathbf{0} \mapsto \mathbf{0} + \mathbf{e} = \mathbf{e}, \quad \mathbf{e} \in \mathbf{L}.$$

Todėl kanalas turi tiek galimybių nepastebimai iškreipti siunčiamą žodį, kiek yra nenulinių kodo žodžių. Tarkime, perdavimo kanalas siunčiamą simbolį

iškreipia į kitą su ta pačia tikimybe, t. y. kanalo tikimybės $p(b|a) = p,\ a \neq b$. Tada tikimybė, kad siunčiamas simbolis bus perduotas teisingai, lygi $1-(q-1)p$. Pažymėkime:

$$A_i = |\{\mathbf{c} \in \mathbf{L} : w(\mathbf{c}) = i\}|, \quad i = 0, 1, \ldots, n.$$

Taigi A_i yra kodo žodžių, kurių svoris lygus i, skaičius.

Tikimybė, kad įvykusios klaidos bus nepastebėtos, lygi:

$$P(\mathbf{L} \text{ žodžio iškraipymai bus nepastebėti}) = \sum_{i=1}^{n} A_i p^i (1-(q-1)p)^{n-i}.$$

Pažymėkime standartinės kodo lentelės, iš kurios išbraukti sluoksniai su ne vieninteliais lyderiais, lyderius: $\mathbf{a}_0, \mathbf{a}_2, \ldots, \mathbf{a}_m$, čia $\mathbf{a}_0 = 00\ldots0$. Tegu

$$\alpha_i = |\{\mathbf{a}_j : w(\mathbf{a}_j) = i\}|, \quad i = 0, \ldots, n.$$

Tada tikimybė, kad dekodavimo algoritmas duos teisingą atsakymą, lygi:

$$P(\mathbf{L} \text{ žodis bus dekoduotas teisingai}) = \sum_{i=0}^{n} \alpha_i p^i (1-(q-1)p)^{n-i}.$$

9.4. Hammingo kodų šeima

Grįžtame prie to, nuo ko pradėjome – prie Hammingo kodo. Ten, kur anksčiau matėme tik vieną gerą kodą, dabar rasime visą begalinę kodų šeimą!

Dvejetainės abėcėlės kodas, kurį nagrinėjome pačioje pradžioje, taiso lygiai vieną klaidą. Jo minimalus atstumas $d = 3$. Ieškosime daugiau tokių kodų. Fiksuosime dar vieną ieškomų kodų parametrą – dualaus kodo dimensiją.

Taigi ieškosime daugiausiai abėcėlės $\mathbb{F}_q$ žodžių turinčio tiesinio kodo, jeigu iš anksto duota jo dualaus kodo dimensija r ir minimalus atstumas $d = 3$. Tokio kodo kontrolinė matrica H turi turėti r eilučių, o bet kurie du jos stulpeliai turi būti tiesiškai nepriklausomi. Tai reiškia, kad nei vienas stulpelis negali būti gaunamas iš kito, padauginus pastarąjį iš $\alpha \in \mathbb{F}_q$. Sudarysime H, imdami tiek stulpelių, kiek tik yra įmanoma.

Pasirinkime iš aibės $V_1 = \mathbb{F}_q^r$ nenulinį žodį $\mathbf{s}_1$ ir sudarykime iš jo elementų pirmąjį H stulpelį. Apibrėžkime aibę

$$V_2 = V_1 \backslash \{\alpha \mathbf{s}_1 : \alpha \in \mathbb{F}_q\}.$$

Antrąjį H stulpelį sudarykime iš pasirinkto aibės V_2 žodžio elementų. Bendra stulpelių pasirinkimo taisyklė tokia:

•*jeigu m-asis matricos H stulpelis sudarytas iš žodžio $\mathbf{s}_m \in V_m$ komponenčių, sudarykime aibę*

$$V_{m+1} = V_m \backslash \{\alpha \mathbf{s}_m : \alpha \in \mathbb{F}_q\}$$

ir, jeigu ši aibė nėra tuščia, pasirinkime iš jos žodį $\mathbf{s}_{m+1}$. Jeigu $V_{m+1} = \emptyset$, matricos H sudarymą užbaikime.

Visų pirma pastebėkime, jog gautos matricos H eilutės yra tiesiškai nepriklausomos, t. y. matricos rangas lygus r. Išties tegu $\mathbf{f}_1, \ldots, \mathbf{f}_r$ yra tiesiškai nepriklausomi aibės $\mathbb{F}_q^r$ žodžiai. Tada egzistuoja $\alpha_i \in \mathbb{F}_q, \alpha_i \neq 0$, kad žodžiai $\alpha_1 \mathbf{f}_1, \ldots, \alpha_r \mathbf{f}_r$ buvo atitinkamuose žingsniuose pasirinkti. Taigi matrica H turi r tiesiškai nepriklausomų stulpelių, todėl ir eilutės yra tiesiškai nepriklausomos.

Kiek stulpelių parenkama tokiu būdu? Kadangi

$$|V_1| = q^r, \quad |V_m| = q^r - 1 - (m-1)(q-1), \quad m \geq 2,$$

tai iš viso galima parinkti $n = (q^r - 1)/(q-1)$ stulpelių. Taigi matrica H yra kontrolinė tiesinio $[n, n-r, 3]$ kodo matrica.

67 apibrėžimas. *Tegu $r \geq 1$, $n = (q^r - 1)/(q-1)$. Tiesinius $[n, n-r, 3]$ kodus iš $\mathbb{F}_q$ abėcėlės žodžių vadinsime Hammingo kodais ir žymėsime $\mathbf{H}_q(r)$.*

Jau minėjome, kad kodai su tokiais parametrais yra tobuli.

79 teorema. *Hammingo kodai yra tobuli.*

Įrodymas. Prisiminkime būtiną ir pakankamą (n, N, d) kodo tobulumo sąlygą:

$$N \cdot V_q(n,t) = q^n, \quad t = \left[\frac{d-1}{2}\right], \quad V_q(n,t) = \sum_{i=0}^{t} C_n^t (q-1)^t.$$

Hammingo kodui su parametrais $[n, n-r, 3]$ ši sąlyga užrašoma taip:

$$q^{n-r} \cdot (1 + n(q-1)) = q^n, \quad q^{n-r} \cdot \left(1 + \frac{q^r - 1}{q-1} \cdot (q-1)\right) = q^n.$$

Taigi Hammingo kodai yra tobuli.

Kodai $\mathbf{H}_2(r)$ yra geriausiai žinomi Hammingo kodai, $\mathbf{H}_2(3)$ – tai klasikinis mūsų nagrinėtas Hammingo kodas. Kodų $\mathbf{H}_2(r)$ kontrolines matricas labai paprasta sudaryti. Surašykime pirmųjų $2^r - 1$ natūraliųjų skaičių skleidinių dvejetainėje sistemoje elementus į matricos stulpelius. Gautoji $r \times (2^r - 1)$ matrica yra kontrolinė kodo $\mathbf{H}_2(r)$ matrica.

Pavyzdžiui, kontrolinė kodo $\mathbf{H}_2(3)$ matrica yra

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Dvinarių Hammingo kodų dekodavimas taip pat labai paprastas. Imkime jau minėtu būdu sudarytą $\mathbf{H}_2(r)$ kodo kontrolinę matricą H. Jeigu siųstas kodo žodis $\mathbf{c}$ siuntimo metu buvo i-oje pozicijoje iškraipytas, tai gautasis žodis yra $\mathbf{x} = \mathbf{c} + \mathbf{e}_i$; čia $\mathbf{e}_i$ yra žodis, kurio visos komponentės, išskyrus i-ąją, lygios nuliui. Raskime $\mathbf{x}$ sindromą:

$$\mathbf{x}H^T = \mathbf{e}_i H^\top.$$

Sindromas $\mathbf{e}_i H^T$ sudarytas iš tų pačių simbolių kaip ir matricos H i-asis stulpelis. Perskaitę $\mathbf{e}_i H^T$ kaip natūralųjį skaičių, užrašytą dvejetainėje sistemoje, gauname reikšmę i, t. y. neteisingai perduotos žodžio komponentės numerį. Tokiu būdu gavėjas gali nustatyti tą kodo žodį, kuris buvo siųstas. Kodai, dualūs Hammingo kodams vadinami, simplekso kodais. Pažymėkime $\mathbf{S}_q(r) = \mathbf{H}_2(r)^\perp$.

80 teorema. *Simplekso kodų $\mathbf{S}_q(r)$ parametrai yra*

$$\left[\frac{q^r-1}{q-1}, r, q^{r-1}\right].$$

Atstumas tarp bet kurių kodo $\mathbf{S}_q(r)$ žodžių lygus q^{r-1}.

Jeigu atstumas tarp bet kurių dviejų tam tikros aibės taškų yra pastovus, tai tokia aibė geometrijoje vadinama simpleksu. Suformuluotoji teorema paaiškina, kodėl kodai, dualūs Hammingo kodams, vadinami simplekso kodais.

Įrodymas. Kodo $\mathbf{S}_q(r)$ žodžiai – tai matricos H eilučių tiesinės kombinacijos. Pakanka įrodyti, kad visų kodo $\mathbf{S}_q(r)$ žodžių (išskyrus nulinį) svoriai lygūs q^{r-1}. Kitaip tariant – kiekviena Hammingo kodo kontrolinės matricos H eilučių tiesinė kombinacija turi lygiai q^{r-1} nenulinių elementų.

Paprasčiausia tuo įsitikinti, kai $q = 2$. Tada į H stulpelius yra surašyti visi nenuliniai r ilgio dvejetainiai žodžiai. Jeigu pridėtume dar ir nulinį stulpelį, tai gautume visus aibės $\mathbb{F}_2^r$ žodžius. Jeigu pasirinktume i-ąją H eilutę, tai būtų tas pats, kas peržiūrėti visus $\mathbb{F}_2^r$ žodžius ir užsirašyti i-ąsias komponentes. Gautume 2^{r-1} vienetų ir tiek pat nulių. Taigi matricos H kiekvienos eilutės svoris lygus 2^{r-1}. Lieka įsitikinti, kad bet kokios šių eilučių tiesinės kombinacijos svoris irgi toks pat.

Jeigu, pavyzdžiui, prie pirmosios H eilutės pridėsime kelias kitas matricos eilutes, gausime naują matricą, kuri irgi bus generuojanti kodo $\mathbf{S}_q(r)$ (ir kontrolinė $\mathbf{H}_2(r)$) matrica. Tarkime, pirmojoje šios matricos eilutėje yra daugiau kaip $2^{r-1}-1$ nulių (tada iš šios eilutės elementų sudarytas kodo žodis sveria mažiau kaip 2^{r-1}. Nagrinėkime tuos H stulpelius, kurių elementai pirmojoje eilutėje lygūs 0. Kadangi tokių stulpelių yra bent 2^{r-1} ir visi jie nenuliniai, tai atsiras du vienodi stulpeliai. Tačiau tada matrica negali būti kontrolinė kodo $\mathbf{H}_2(r)$ matrica, nes šio kodo minimalus atstumas lygus 3. Taigi kodo žodis negali sverti mažiau kaip 2^{r-1}. Panašiai galime įsitikinti, kad daugiau sverti kodo žodžiai irgi negali.

Kiek pakeitus šiuos samprotavimus, galima įrodyti teiginį ir atveju $q > 2$.

Kadangi simplekso kodo nenulinių žodžių svoriai vienodi, tikimybės, kad siųsto žodžio iškraipymai bus nepastebėti, išraiška labai paprasta. Pavyzdžiui, pasinaudoję (9.3.)

$$P(\mathbf{S}_2(r) \text{ žodžio iškraipymai bus nepastebėti}) = (2^r - 1)p^{2^{r-1}}(1-p)^{2^{r-1}-1},$$

čia p – vieno simbolio iškraipymo tikimybė simetriniame be atminties kanale.

Savo ruožtu paprasta užrašyti kodo $\mathbf{H}_2(r)$ žodžio teisingo dekodavimo tikimybę:

$$P(\mathbf{H}_2(r) \text{ žodis bus dekoduotas teisingai}) = (1-p)^{2^r-1} + (2^r-1)p(1-p)^{2^r-2}.$$

Norėdami užrašyti klaidos nepastebėjimo tikimybę kodui $\mathbf{H}_2(r)$, turėtume nustatyti, kokio svorio žodžių jis turi, t. y. rasti dydžius

$$A_i = |\{\mathbf{c} \in \mathbf{H}_2(r) : w(\mathbf{c}) = i\}|, \quad i = 0, 1, \ldots, n.$$

Žinome, kad $A_0 = 1, A_1 = A_2 = 0$. Kitus dydžius galima skaičiuoti pasinaudojant rekurentiniu sąryšiu.

81 teorema. *Hammingo kodo* $\mathbf{H} = \mathbf{H}_2(r)$ *dydžiai* A_i *tenkina rekurenčiąją lygybę*

$$iA_i = C_n^{i-1} - A_{i-1} - (n-i+2)A_{i-2} \quad (i \geq 2, n = 2^r - 1).$$

Įrodymas. Kiekvieną $\mathbf{H}$ žodį $\mathbf{x}$ su svoriu $w(x) = m$ atitinka kontrolinės matricos H m stulpelių tiesinė kombinacija, kuri lygi nuliniam stulpeliui. Ir atvirkščiai: jei, sudėję m kontrolinės matricos stulpelių, gauname nulinį stulpelį, tai šią sumą atitinka vienintelis kodo žodis su svoriu m. Parinkime $i-1$ matricos H stulpelių, tai galima padaryti C_n^{i-1} būdais. Galimi trys atvejai:

1. stulpelių suma yra nulinis stulpelis;
2. stulpelių suma lygi vienam iš pasirinktų stulpelių;
3. stulpelių suma lygi vienam iš nepasirinktų stulpelių.

Skirtingų pasirinkimų, atitinkančių 1) atvejį, yra A_{i-1}, atitinkančių 2) – $(n-i+2)A_{i-2}$ ir atitinkančių 3) atvejį – iA_i. Taigi

$$C_n^{i-1} = A_{i-1} + (n-i+2)A_{i-2} + iA_i.$$

Pavyzdžiui, kodui $\mathbf{H}_2(3)$ turime

$$(A_0, A_1, A_2, A_3, A_4, A_4, A_6, A_7) = (1, 0, 0, 7, 7, 0, 0, 1).$$

9.5. Maksimalaus atstumo kodai

Kaip sukonstruoti maksimalų kodą – sudėtingas klausimas. Vis dėlto tai kartais pavyksta. Ir gana lengvai! Sužinosite, kaip tą padaryti, jeigu perskaitysite skyrelį iki galo.

Prisiminkime Singletono įvertį: jeigu $\mathbf{C} \subset \mathcal{A}_q^n$ yra (n, N, d) kodas, tai

$$N \leq q^{n-d+1}.$$

Tegu $\mathcal{A}_q = \mathbb{F}_q$, o $\mathbf{L}$ yra tiesinis $[n, k, d]$ kodas. Tada jis turi q^k žodžių ir iš Singletono įverčio gauname

$$q^k \leq q^{n-d+1}, \quad d \leq n - k + 1.$$

Jeigu tiesiniam kodui teisinga lygybė $d = n-k+1$, tai kodas yra maksimalus; t. y. daugiau žodžių neturi joks kitas kodas, turintis tą patį minimalų atstumą.

68 apibrėžimas. *Tiesinį $[n, k, d]$ kodą vadinsime maksimalaus atstumo kodu, jeigu teisinga lygybė*

$$d = n - k + 1.$$

Ar maksimalaus atstumo kodai egzistuoja? Štai paprastas teikiantis vilčių atsakymas: bet kokio kūno $\mathbb{F}_q$ atveju egzistuoja tiesiniai $[n, n, 1]$, $[n, 1, n]$ ir $[n, n-1, 2]$ kodai. Akivaizdu, kad kodai su tokiais parametrais yra maksimalaus atstumo kodai. Galbūt nėra visai aišku, kaip sudaryti kodą su parametrais $[n, n-1, 2]$. Štai užuomina. Imkime visus erdvės $\mathbb{F}_q^{n-1}$ žodžius ir pailginkime juos vienu simboliu:

$$x_1 x_2 \ldots x_{n-1} \mapsto x_1 x_2 \ldots x_{n-1} x_n, \quad x_1 + x_2 + \ldots + x_{n-1} + x_n = 0.$$

Gautoji žodžių aibė yra $[n, n-1, 2]$ kodas!

Tačiau iš tokių kodų mažai naudos. Vadinsime juos trivialiais maksimalaus atstumo kodais. Paieškokime kitokių.

82 teorema. *Tegu $\mathbf{L}$ yra tiesinis $[n, k]$ kodas, o H – jo kontrolinė matrica. Tada $\mathbf{L}$ yra maksimalaus atstumo kodas tuo ir tik tuo atveju, kai bet kurie $n-k$ matricos H stulpeliai yra tiesiškai nepriklausomi.*

Įrodymas. Minimalus kodo $\mathbf{L}$ atstumas lygus d tada ir tik tada, kai egzistuoja d tiesiškai priklausomų kontrolinės matricos stulpelių, tačiau bet kurie $d-1$ jos stulpelių yra tiesiškai nepriklausomi. Todėl sąlyga $d = n-k+1$ yra ekvivalenti reikalavimui, kad bet kurie $d-1 = n-k$ stulpelių sudarytų tiesiškai nepriklausomą sistemą ir egzistuotų $n-k+1$ tiesiškai priklausomų stulpelių.

Kontrolinės matricos H matavimai yra $(n-k) \times n$, o jos rangas lygus $n-k$. Bet kurie $n-k+1$ matricos stulpeliai sudaro tiesiškai priklausomą sistemą. Teorema įrodyta.

Kaip netrukus įsitikinsime, maksimalaus atstumo kodai egzistuoja poromis.

83 teorema. *Tegu* $\mathbf{L}$ *yra maksimalaus atstumo kodas. Tada* $\mathbf{L}^\perp$ *yra taip pat maksimalaus atstumo kodas.*

Įrodymas. Tegu G, H yra atitinkamai $[n,k]$ kodo $\mathbf{L}$ generuojanti bei kontrolinė matricos. Bet kurie skirtingi $n-k$ matricos H stulpeliai sudaro tiesiškai nepriklausomą sistemą. Tada bet kuris šios matricos $n-k$ eilės minoras nelygus nuliui. Iš matricos H eilučių elementų sudarykime žodžius $\mathbf{a}_1, \ldots, \mathbf{a}_m, m = n-k$. Bet kuris kitas kodo $\mathbf{L}^\perp$ žodis $\mathbf{x}$ yra šių žodžių tiesinė kombinacija:

$$\mathbf{x} = \alpha_1\mathbf{a}_1 + \ldots + \alpha_m\mathbf{a}_m, \quad \alpha_i \in \mathbf{F}_q.$$

Pagal šią lygybę atlikime matricos H pertvarkius; dėl apibrėžtumo tarkime, $\alpha_1 \neq 0$. Padauginkime pirmąją eilutę iš α_1, tada dauginkime i-ąją eilutę iš α_i $(i \geq 2)$ ir pridėkime prie pirmosios. Atlikę šiuos veiksmus, pirmojoje eilutėje gausime žodžio $\mathbf{x}$ simbolius. Jokio H matricos $n-k$ minoro reikšmė netapo lygi nuliui, nes pakito tik daugikliu $\alpha_1 \neq 0$. Taigi žodyje $\mathbf{x}$ negali būti $n-k$ komponenčių, lygių nuliui. Tai reiškia, jog kiekvieno žodžio $\mathbf{x} \in \mathbf{L}^\perp$ svoris $w(\mathbf{x}) \geq n-(n-k-1) = k+1$. Todėl $d \geq k+1$, tačiau Singletono įvertis kodui $\mathbf{L}^\perp$ tvirtina, kad $d \leq n-(n-k)+1 = k+1$. Taigi $d = k+1 = n-(n-k)+1$, o tai reiškia, jog kodas $\mathbf{L}^\perp$ yra maksimalaus atstumo kodas.

Derindami abiejų teoremų tvirtinimus, nesunkiai gausime tokią išvadą:

84 teorema. *Jei* $[n,k]$ *kodo* $\mathbf{L}$ *generuojanti matrica yra* G*, tai* $\mathbf{L}$ *yra maksimalaus atstumo kodas tada ir tik tada, kai bet kurie* k *matricos* G *stulpeliai yra tiesiškai nepriklausomi.*

Apsiginklavę šiais teiginiais bandykime konstruoti netrivialius maksimalaus atstumo kodus.

69 apibrėžimas. *Tegu* $\mathbb{F}_q = \{\alpha_1, \alpha_2, \ldots, \alpha_q\}$ *(apibrėžtumo dėlei tarkime,* $\alpha_q = 0$*),* o $1 \leq k \leq q$ *yra natūralusis skaičius. Tiesinį kodą, kurio generuojanti matrica yra*

$$G_k = \begin{pmatrix} 1 & 1 & \ldots & 1 \\ \alpha_1 & \alpha_2 & \ldots & \alpha_q \\ \alpha_1^2 & \alpha_2^2 & \ldots & \alpha_q^2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{k-1} & \alpha_2^{k-1} & \ldots & \alpha_q^{k-1} \end{pmatrix},$$

vadinsime Reedo-Solomono kodu ir žymėsime $\mathbf{RS}_{q,k}$.

Kodo $\mathbf{RS}_{q,k}$ dimensija lygi k. Kad tuo įsitikintume, pakanka įrodyti, kad bet kurie k matricos G_k stulpeliai sudaro tiesiškai nepriklausomų stulpelių sistemą (paskutinio stulpelio pirmasis elementas lygus vienetui, o kiti – nuliui). Tam pakanka parodyti, kad iš bet kurių k stulpelių sudarytas determinantas nelygus nuliui. Tai galima padaryti pasinaudojus algebroje gerai

žinomu Vandermondo determinantu. Tegu $\alpha_1, \ldots, \alpha_s$ yra skirtingi ir nelygūs nuliui kūno $\mathbb{F}_q$ elementai,

$$V(\alpha_1, \alpha_2, \ldots, \alpha_s) = \det \begin{pmatrix} 1 & 1 & \ldots & 1 \\ \alpha_1 & \alpha_2 & \ldots & \alpha_s \\ \alpha_1^2 & \alpha_2^2 & \ldots & \alpha_s^2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{s-1} & \alpha_2^{s-1} & \ldots & \alpha_s^{s-1} \end{pmatrix}.$$

Tada

$$V(\alpha_1, \alpha_2, \ldots, \alpha_s) = \prod_{1 \le i < j \le s} (\alpha_j - \alpha_i).$$

O dabar keletas gerų naujienų apie Reedo-Solomono kodus.

85 teorema. *Visi $\mathbf{RS}_{q,k}$ kodai yra maksimalaus atstumo kodai.*
Jei $1 \le k_1 < k_2 \le q$, tai $\mathbf{RS}_{q,k_1} \subset \mathbf{RS}_{q,k_2}$.
Su visomis k reikšmėmis teisinga lygybė $\mathbf{RS}_{q,k}^{\perp} = \mathbf{RS}_{q,q-k}$.

Įrodymas. Pasinaudoję Vandermondo determinantu, galime įrodyti, kad bet kokie k matricos G_k stulpeliai sudaro tiesiškai nepriklausomų stulpelių sistemą. Iš anksčiau įrodytos teoremos išplaukia, kad $\mathbf{RS}_{q,k}$ yra maksimalaus atstumo kodas.

Sąryšis $\mathbf{RS}_{q,k_1} \subset \mathbf{RS}_{q,k_2}$ yra akivaizdus.

Kadangi $\dim(\mathbf{RS}_{q,k}) + \dim(\mathbf{RS}_{q,q-k}) = q$, tai trečiajam teiginiui įrodyti pakanka įsitikinti, kad matricų G_k ir G_{n-k} eilutės yra poromis ortogonalios. Parinkime iš pirmosios matricos eilutę $\mathbf{u}$ iš antrosios – $\mathbf{v}$:

$$\mathbf{u} = \begin{pmatrix} \alpha_1^i & \alpha_2^i & \ldots & \alpha_q^i \end{pmatrix}, \quad \mathbf{v} = \begin{pmatrix} \alpha_1^j & \alpha_2^j & \ldots & \alpha_q^j \end{pmatrix}.$$

Jeigu $i = j = 0$, tai

$$\mathbf{u} \cdot \mathbf{v} = 1 + 1 \ldots + 1 = q \cdot 1 = 0.$$

Kitais atvejais

$$S = \mathbf{u} \cdot \mathbf{v} = \alpha_1^l + \alpha_2^l + \ldots + \alpha_{q-1}^l, \quad l \equiv i + j \pmod{q-1}, \quad 1 \le l \le q-2. \quad (59)$$

Egzistuoja bent vienas nenulinis elementas $\alpha \in \mathbb{F}_q$, kad $\alpha^l \ne 1$. Iš tiesų, jeigu tokio elemento nebūtų, tai visi $q-1$ elementai būtų daugianario $x^l - 1$ $l < q-1$, šaknys.

Padauginę (59) iš α^l, gausime

$$\alpha^l S = (\alpha\alpha_1)^l + (\alpha\alpha_2)^l + \ldots + (\alpha\alpha_{q-1})^l = S, \quad (\alpha^l - 1)S = 0.$$

Taigi gauname, kad $S = 0$ ir eilutės $\mathbf{u}, \mathbf{v}$ yra ortogonalios. Teorema įrodyta.

Įsižiūrėkime į sąryšį $\mathbf{RS}_{q,k}^{\perp} = \mathbf{RS}_{q,q-k}$. Jeigu $q = 2^m, k = q^{m-1}$, tai šis sąryšis pavirsta lygybe

$$\mathbf{RS}_{2^m,2^{m-1}}^{\perp} = \mathbf{RS}_{2^m,2^{m-1}}.$$

Taigi suradome be galo daug savidualių kodų!

Patyrinėkime dar. Generuojančią kodo $\mathbf{RS}_{q,k}$ matricą G_k (paskutinį jos stulpelį sudaro elementai $1, 0, 0, \ldots, 0$) papildykime dar vienu stulpeliu:

$$G_k^* = \begin{pmatrix} 1 & 1 & \ldots & 1 & 0 \\ \alpha_1 & \alpha_2 & \ldots & 0 & 0 \\ \alpha_1^2 & \alpha_2^2 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \alpha_1^{k-1} & \alpha_2^{k-1} & \ldots & 0 & 1 \end{pmatrix}.$$

Vėl pasinaudoję Vandermondo determinantu, galime įsitikinti, kad bet kurie k stulpeliai yra tiesiškai nepriklausomi. Taigi G_k^* taip pat yra maksimalaus atstumo kodo generuojanti matrica. Šio kodo parametrai yra $[q+1, k, q-k+2]$. Pažymėkime šį kodą $\mathbf{RS}_{q+1,k}^*$. Jeigu q yra nelyginis pirminis arba jo laipsnis, o $k = \frac{q+1}{2}$, tai, samprotaujant panašiai kaip Reedo-Solomono kodų atveju, galima įrodyti, kad

$$\mathbf{RS}_{q+1,k}^{*\perp} = \mathbf{RS}_{q+1,k}^*.$$

Taigi radome dar vieną savidualių kodų šeimą.

Reedo-Solomono kodai yra netrivialūs, kai $q > 2$. Ar yra netrivialių maksimalaus atstumo kodų, sudarytų iš dvejetainės abėcėlės žodžių? Atsakymas toks: iš dvejetainės abėcėlės žodžių galime sudaryti tik trivialiuosius maksimalaus atstumo kodus.

9.6. Marcelio Golay kodai

1948 metais Richardas Hammingas paskelbė kodą, kuris visada ištaiso vieną klaidą. Šveicarų matematikas Marcelis Golay 1949 metais sugalvojo net tris klaidas taisantį kodą. Šis kodas yra didelis ir geras – tiesiog tobulas.

Parinkime tokį dvejetainės abėcėlės 23 bitų ilgio žodį

$$\mathbf{c}_1 = 11000111010100000000000.$$

Cikliškai perstūmę žodžio simbolius, sudarykime dar vienuolika žodžių:

$$\begin{aligned}
\mathbf{c}_2 &= 01100011101010000000000,\\
\mathbf{c}_3 &= 00110001110101000000000,\\
&\ldots\ldots\ldots\ldots\ldots\ldots,\\
\mathbf{c}_{12} &= 00000000000110001110101.
\end{aligned}$$

Visi šie žodžiai yra tiesiškai nepriklausomi. Taigi galime šią žodžių sistemą panaudoti kaip tiesinio kodo bazę.

70 apibrėžimas. *Kodą, kurį generuoja žodžiai* $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{12}$*, vadinsime dvejetainiu Golay kodu* $\mathbf{G}_{23}$.

O dabar pailginkime visus žodžius $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{12}$, prirašydami kiekvieno žodžio gale simbolį 1. Gausime erdvės $\mathbb{F}_2^{24}$ žodžius; juos žymėkime $\mathbf{c}_1^*, \mathbf{c}_2^*, \ldots, \mathbf{c}_{12}^*$.

71 apibrėžimas. *Kodą, kurį generuoja žodžiai* $\mathbf{c}_1^*, \mathbf{c}_2^*, \ldots, \mathbf{c}_{12}^*$*, vadinsime dvejetainiu Golay kodu* $\mathbf{G}_{24}$.

Pastebėkime, kad kiekvieno žodžio $\mathbf{c}_i^*$ visų simbolių suma moduliu 2 lygi 0. Šią savybę turės ir visi kiti kodo $\mathbf{G}_{24}$ žodžiai. Vadinasi, galime įsivaizduoti, kad kodas $\mathbf{G}_{24}$ gaunamas iš kodo $\mathbf{G}_{23}$, pailginus šio kodo žodžius kontroliniu simboliu, o $\mathbf{G}_{23}$ – iš $\mathbf{G}_{24}$, nubraukiant paskutinį šio kodo žodžių simbolį. Abiejų kodų dimensijos lygios 12.

Surašę žodžių $\mathbf{c}_i^*$ simbolius į eilutes, gautume kodo $\mathbf{G}_{24}$ generuojančią matricą. Patikrinę įsitikintume, kad visos šios matricos eilutės poromis yra ortogonalios (galima pasinaudoti žodžių $\mathbf{c}_i$ ciklišku mu ir išvengti tikrinimo visoms poroms!). Šitaip nustatytume svarbią Golay kodo savybę.

86 teorema. *Kodas* $\mathbf{G}_{24}$ *yra savidualus.*

Žinome, kad kodo standartinio pavidalo generuojanti matrica kodavimui yra pati geriausia. Ne visi kodai ją turi (bet visada galima nurodyti ekvivalentų kodą su standartinio pavidalo generuojančia matrica). Tačiau kodas $\mathbf{G}_{24}$ tokią matricą turi.

87 teorema. *Matrica* $G = (I_{12}, A)$,

$$A = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

yra kodo $\mathbf{G}_{24}$ *generuojanti matrica.*

Turėdami standartinio pavidalo generuojančią matricą, galime sudaryti kontrolinę matricą:

$$H = (-A^T, I_{12}) = (A, I_{12}), \quad \text{nes } -A^T = A^T = A.$$

Tačiau kodas $\mathbf{G}_{24}$ yra savidualus, todėl abi matricos

$$G = (I_{12}, A),\ H = (A, I_{12})$$

yra ir generuojančios, ir kontrolinės.

Laikas užrašyti abiejų kodų parametrus.

88 teorema. *Kodo* $\mathbf{G}_{24}$ *parametrai yra* $[24, 12, 8]$ *o kodo* $\mathbf{G}_{23}$ – $[23, 12, 7]$.

Įrodymas. Pakanka nustatyti, kad kodo $\mathbf{G}_{24}$ minimalus atstumas yra 8. Kodo $\mathbf{G}_{23}$ minimalus atstumas gali būti mažesnis ne daugiau kaip vienetu. Tačiau šis kodas turi daug žodžių, kurių svoriai lygūs 7, pavyzdžiui, žodžiai $\mathbf{c}_i$. Taigi jei kodo $\mathbf{G}_{24}$ minimalus atstumas yra 8, tai kodo $\mathbf{G}_{23}$ – 7.

Pirmiausia įsitikinsime, kad visų kodo $\mathbf{G}_{24}$ žodžių svoriai dalijasi iš 4. Tegu $\mathbf{x} = x_1x_2 \ldots x_{24}, \mathbf{y} = y_1y_2 \ldots y_{24}$ yra du kodo žodžiai. Nesunku įsitikinti, jog svoriams galioja tokia lygybė:

$$w(\mathbf{x} + \mathbf{y}) = w(\mathbf{x}) + w(\mathbf{y}) - 2S, \quad S = x_1y_1 + x_2y_2 + \ldots + x_{24}y_{24}.$$

Tačiau kodas $\mathbf{G}_{24}$ yra savidualus, todėl skaičius S yra lyginis, taigi

$$w(\mathbf{x}+\mathbf{y}) \equiv w(\mathbf{x}) + w(\mathbf{y}) \pmod 4. \tag{60}$$

Jeigu $\mathbf{x}$ ir $\mathbf{y}$ yra žodžiai, sudaryti iš dviejų matricos G eilučių elementų (du kodo bazės žodžiai), tai $w(\mathbf{x})$ ir $w(\mathbf{y})$ dalijasi iš 4. Iš (60) gauname, kad žodžio $\mathbf{x}+\mathbf{y}$ svoris dalijasi iš 4. Taigi bet kurio žodžio, sudaryto iš dviejų bazės žodžių, svoris dalijasi iš 4. Tegu $\mathbf{x}$ yra toks žodis, o $\mathbf{y}$ – bazės žodis (matricos G eilutė). Pasinaudoję (60), gauname, kad kodo žodžių, kurie yra trijų bazės žodžių tiesinės kombinacijos, svoriai taip pat dalijasi iš 4. Kaip samprotauti toliau, tikriausiai aišku. Galime manyti, kad teiginys įrodytas.

Dabar liko įrodyti, kad nėra nei vieno kodo $\mathbf{G}_{24}$ žodžio, kurio svoris būtų lygus 4. Samprotaudami naudosimės tuo, kad matricos G ir H yra generuojančios kodo $\mathbf{G}_{24}$ matricos.

Tarkime, egzistuoja kodo $\mathbf{G}_{24}$ žodis $\mathbf{x}$, kad $w(\mathbf{x}) = 4$. Kadangi abi matricos G, H generuoja tą patį kodą $\mathbf{G}_{24}$, tai $\mathbf{x}$ galime nagrinėti kaip matricos G arba H eilučių tiesinę kombinaciją. Pažymėję $\mathbf{l}$ ir $\mathbf{r}$ atitinkamai kairiąją ir dešiniąją žodžio $\mathbf{x}$ puses (sudarytas iš 12 simbolių), gausime

$$w(\mathbf{x}) = w(\mathbf{l}) + w(\mathbf{r}).$$

Pastebėsime, jog atvejis $w(\mathbf{l}) = 0$ arba $w(\mathbf{r}) = 0$ yra negalimas. Jei $w(\mathbf{l}) = 1$, tai $w(\mathbf{r}) = 3$; $\mathbf{x}$ turi būti viena iš matricos G eilučių, bet eilutės su dešinės pusės svoriu, lygiu 3, nėra. Analogiškai gauname, jog atvejis $w(\mathbf{l}) = 3$, $w(\mathbf{r}) = 1$ taip pat neįmanomas. Lieka atvejis $w(\mathbf{l}) = w(\mathbf{r}) = 2$. Bet tokiu atveju $\mathbf{x}$ yra dviejų skirtingų matricos G eilučių suma. Tiesiogiai tikrindami galime nustatyti, jog $w(\mathbf{u}+\mathbf{v}) \neq 4$ jokioms dviem skirtingoms G eilutėms $\mathbf{u}$,$\mathbf{v}$. Teiginys įrodytas. Įrodyta ir teorema.

Patikrinę įsitikintume, kad kodo $\mathbf{G}_{23}$ parametrai tenkina tobulumo sąlygas.

Išvada. *Kodas* $\mathbf{G}_{23}$ *yra tobulas.*

Aptarsime kodo $\mathbf{G}_{24}$ dekodavimą. Žinoma, galima taikyti bendrą lyderių-sindromų metodą, tinkantį visiems tiesiniams kodams. Tačiau Golay kodas turi tiek gerų savybių! Kodėl gi jomis nepasinaudojus?

Naudodamiesi šiuo kodu, galime ištaisyti ne daugiau kaip 3 klaidas. Jeigu, siunčiant kodo žodį $\mathbf{c}$, įvyko iškraipymas $\mathbf{e}$ (šį žodį sudaro nuliai tose pozicijose, kurios perduotos teisingai, ir vienetai ten, kur įvyko perdavimo klaidos), tai gautasis žodis yra $\mathbf{x} = \mathbf{c} + \mathbf{e}$. Nurodysime metodą, kuris leidžia teisingai nustatyti nežinomą $\mathbf{e}$, kai $w(\mathbf{e}) \leq 3$. Radę $\mathbf{e}$, gautą žodį $\mathbf{x}$ dekoduosime kodo žodžiu $\mathbf{c} = \mathbf{x} - \mathbf{e}$.

Žinoma, dekoduodami iš anksto nežinome, ar $w(\mathbf{e}) \leq 3$. Jeigu metodas „neveikia“, tai ši sąlyga nėra patenkinta ir įvykusių klaidų skaičius yra didesnis, nei kodas gali ištaisyti!

Kairiąją iškraipymo $\mathbf{e}$ dalį, sudarytą iš 12 pirmųjų simbolių, pažymėję $\mathbf{e}_1$, o dešiniąją – $\mathbf{e}_2$, gauname

$$\mathbf{e} = \mathbf{e}_1\mathbf{e}_2, \quad w(\mathbf{e}) = w(\mathbf{e}_1) + w(\mathbf{e}_2) \leq 3.$$

Galimi atvejai:

$$1)\ w(\mathbf{e}_1)=0;\quad 2)\ w(\mathbf{e}_2)=0;\quad 3)\ w(\mathbf{e}_1)>0,\ w(\mathbf{e}_2)>0.$$

Iš pradžių ištirsime, kaip pagal gautą žodį nustatyti, kuris iš 1), 2), 3) atvejų įvyko. Kadangi abi generuojančios matricos $G=(I_{12},A), H=(A,I_{12})$ yra kartu ir kontrolinės, tai galime nagrinėti du gauto žodžio $\mathbf{x}$ sindromus:

$$\begin{aligned}\mathbf{s}_1 &= (\mathbf{c}+\mathbf{e})G^T=\mathbf{e}G^T=\mathbf{e}_1\mathbf{e}_2\begin{pmatrix}I_{12}\\A\end{pmatrix}=\mathbf{e}_1+\mathbf{e}_2A,\\ \mathbf{s}_2 &= (\mathbf{c}+\mathbf{e})H^T=\mathbf{e}H^T=\mathbf{e}_1\mathbf{e}_2\begin{pmatrix}A\\I_{12}\end{pmatrix}=\mathbf{e}_1A+\mathbf{e}_2.\end{aligned}$$

Jeigu $w(\mathbf{e}_1)=0$, tai iš sindromų išraiškų gauname $\mathbf{s}_1=\mathbf{e}_2A, \mathbf{s}_2=\mathbf{e}_2$. Taigi $w(\mathbf{e})=w(\mathbf{e}_2)=w(\mathbf{s}_2)\le 3$. Kita vertus, $\mathbf{s}_1$ yra dešinioji kodo žodžio $\mathbf{c}=\mathbf{e}_2(I_{12},A)$ pusė. Kadangi $w(\mathbf{c})\ge 8$, o kairioji pusė sveria ne daugiau kaip 3, tai $w(\mathbf{s}_1)=w(\mathbf{c})-w(\mathbf{e}_2)\ge 8-3=5$.

Jeigu $w(\mathbf{e}_2)=0$, tai analogiškai gauname

$$w(\mathbf{e})=w(\mathbf{e}_1)=w(\mathbf{s}_1)\le 3,\quad w(\mathbf{s}_2)\ge 5.$$

Jei $w(\mathbf{e}_1)>0,\ w(\mathbf{e}_2)>0$, tai $w(\mathbf{s}_1)\ge 5,\ w(\mathbf{s}_2)\ge 5$. Pavyzdžiui, jei $w(\mathbf{e}_1)=1, w(\mathbf{e}_1)=2$, tai

$$w(\mathbf{s}_1)=w(\mathbf{e}_1+\mathbf{e}_2A)\ge w(\mathbf{e}_2A)-w(\mathbf{e}_1)\ge(8-2)-1=5.$$

Taigi pagal sindromų svorius $w(\mathbf{s}_1), w(\mathbf{s}_2)$ galima nustatyti, kuris iš 1), 2), 3) atvejų įvyko.

Jei $w(\mathbf{s}_1)\le 3$, tai $w(\mathbf{e}_2)=0$ ir $\mathbf{e}=\mathbf{e}_1\mathbf{0}=\mathbf{s}_1\mathbf{0}$; čia $\mathbf{0}$ yra žodis, sudarytas iš 12 nulių. Analogiškai jei $w(\mathbf{s}_2)\le 3$, tai $\mathbf{e}=\mathbf{0}\mathbf{s}_2$.

Tad 1), 2) atvejais dekodavimas nesukelia didelių sunkumų. Lieka ištirti dekodavimą tuo atveju, kai $w(\mathbf{s}_1)\ge 5,\ w(\mathbf{s}_2)\ge 5$, t. y. 3) atveju. Jį suskaidysime į dvi galimybes:

$$1)\ w(\mathbf{e}_1)=1,\ w(\mathbf{e}_2)=1 \text{ arba } 2;\quad 2)\ w(\mathbf{e}_1)=2,\ w(\mathbf{e}_2)=1.$$

Pastebėsime, jog 1) atveju pakanka rasti $\mathbf{e}_1$. Išties, suradę šią dalį, galime manyti, jog gautasis žodis yra $\mathbf{x}'=\mathbf{x}-\mathbf{e}_1\mathbf{0}$, ir dekoduoti jau aptartu būdu. Analogiška pastaba teisinga ir 2) atvejui. Pakaks išnagrinėti 1) atvejį. Tegu iškraipymas įvyko j-ojoje pozicijoje, $1\le j\le 12$. Tada $\mathbf{e}_1=\varepsilon_j$; čia ε_j žymime žodį iš 12 simbolių, kurių tik j-asis yra vienetas, kiti – nuliai. Sudarykime 12 naujų žodžių

$$\mathbf{x}+\varepsilon_1\mathbf{0},\dots,\mathbf{x}+\varepsilon_{12}\mathbf{0}$$

ir 12 juos atitinkančių sindromų

$$\begin{aligned}\mathbf{s}_i &= (\mathbf{x}+\varepsilon_i\mathbf{0})\begin{pmatrix}A\\I_{12}\end{pmatrix}=(\mathbf{e}+\varepsilon_i\mathbf{0})\begin{pmatrix}A\\I_{12}\end{pmatrix}\\ &= (\varepsilon_j\mathbf{e}_2+\varepsilon_i\mathbf{0})\begin{pmatrix}A\\I_{12}\end{pmatrix}=\varepsilon_jA+\mathbf{e}_2+\varepsilon_iA.\end{aligned}$$

Jeigu $i = j$, tai $\mathbf{s}_i = \mathbf{e}_2$, taigi $w(\mathbf{s}_i) \le 2$. Jeigu $i \ne j$, tai

$$w(\mathbf{s}_i) \ge w(\varepsilon_j A + \varepsilon_i A) - w(\mathbf{e}_2).$$

Žodis $\varepsilon_j A + \varepsilon_i A = (\varepsilon_j + \varepsilon_i)A$ yra kodo žodžio $(\varepsilon_j + \varepsilon_i)(I_{12}, A)$ dešinioji pusė; kadangi visas žodis sveria ne mažiau kaip 8, kairioji – lygiai 2, tai

$$w((\varepsilon_j + \varepsilon_i)A) \ge 8 - 2 = 6, \quad w(\mathbf{s}_i) \ge 6 - 2 = 4.$$

Taigi peržiūrėję sindromų svorius, pamatysime, kad visi jie, išskyrus vieną, yra ne mažesni už 4. Imdami tą j, kuriam $w(\mathbf{s}_j) \le 2$, rasime $\mathbf{e}_1 = \varepsilon_j$. Jeigu svorių seka kitokia, tai susidūrėme su 2) atveju. Reikės ieškoti $\mathbf{e}_2 = \varepsilon_j$. Dabar teks peržiūrėti sindromus

$$\mathbf{s}_i' = (\mathbf{x} + \mathbf{0}\varepsilon_i)\begin{pmatrix} I_{12} \\ A \end{pmatrix} = \mathbf{e}_1 + \varepsilon_j A + \varepsilon_i A.$$

Radę j, kuriam $w(\mathbf{s}_j') \le 2$, gausime $\mathbf{e}_2 = \varepsilon_j$. Jeigu vėl nepavyks – žodis yra pernelyg iškraipytas, kad galėtume jį atkurti.

9.7. Reedo-Mullerio kodai

Ši kodų šeima taip pat kaip ir Hadamardo kodų įėjo į kosmonautikos istoriją. Jais buvo naudojamasi 1969–1977 metais palaikant ryšį su kosminėmis stotimis. Ir šiaip tai labai įdomūs kūriniai.

Galima sukonstruoti Reedo-Mullerio kodus iš bet kokios abėcėlės $\mathbb{F}_p$ žodžių. Tačiau panagrinėkime paprasčiausią ir svarbiausią dvejetainės abėcėlės atvejį.

Tarkime, tiesinės erdvės $\mathbb{F}_2^m$ žodžiai kokiu nors būdu sunumeruoti:

$$\mathbb{F}_2^m = \{\mathbf{a}_1, \dots, \mathbf{a}_n\}, \quad n = 2^m.$$

Galime, pavyzdžiui, žvelgti į žodį $\mathbf{a} \in \mathbb{F}_2^m$ kaip į sveikojo skaičiaus $0 \le l < 2^m$ išraišką dvejetainėje skaičiavimo sistemoje ir sutvarkyti žodžius atitinkamų skaičių didėjimo tvarka.

Žymėkime $x_i(\mathbf{a})$ i-ąją žodžio $\mathbf{a}$ komponentę. Tada aibės

$$H_i = \{\mathbf{a} : \mathbf{a} \in \mathbb{F}_2^m, x_i(\mathbf{a}) = 0\}, \quad i = 1, \dots, m,$$

yra tiesinės erdvės $\mathbb{F}_2^m$ poerdviai.

Apibrėšime abipusiškai vienareikšmę $\mathbb{F}_2^m$ poaibių ir erdvės $\mathbf{F}_2^n$, $n = 2^m$, žodžių atitiktį. Jei $D \subset \mathbb{F}_2^m$, tai priskirsime: $D \to \mathbf{v}(D)$; čia $\mathbf{v}(D) \in \mathbb{F}_2^n$ komponentės apibrėžiamos taip:

$$x_i(\mathbf{v}(D)) = \begin{cases} 0, & \text{jei } \mathbf{a}_i \notin D, \\ 1, & \text{jei } \mathbf{a}_i \in D. \end{cases}$$

Pavyzdžiui, jei $m = 3, D = \{\mathbf{a}_1, \mathbf{a}_3, \mathbf{a}_5\}$, tai $\mathbf{v}(D) = 10101000$. Taigi žodyje $\mathbf{v}(D)$ vienetukai žymi, kurie žodžiai įeina į D; $\mathbf{v}(D)$ yra tarsi poaibio D „inventorizacijos“ dokumentas. Visą erdvę $\mathbb{F}_2^m$ atitinka žodis $\mathbf{v}_0 = 11\ldots1$.

Toliau šiame skyrelyje naudosime dvi operacijas su žodžiais $\mathbf{a}, \mathbf{b} \in \mathbb{F}_2^n$: žodžių daugybą ir skaliarinę sandaugą. Jei $\mathbf{x} = x_1x_2\ldots x_n$ ir $\mathbf{y} = y_1y_2\ldots y_n$, žymėsime

$$\begin{aligned} \mathbf{a}\cdot\mathbf{b} &= z_1z_2\ldots z_n, \quad z_j = x_iy_i, \quad i = 1, 2, \ldots, n; \\ (\mathbf{a}, \mathbf{b}) &= x_1y_1 + x_2y_2 + \ldots + x_ny_n. \end{aligned}$$

Skaičiuojant skaliarinę sandaugą, dėmenys sumuojami moduliu 2, taigi $(\mathbf{a}, \mathbf{b})$ reikšmė yra 0 arba 1. Akivaizdu, kad žodžių daugybai teisingas distributyvumo sudėties atžvilgiu dėsnis:

$$(\mathbf{a} + \mathbf{b})\cdot\mathbf{c} = \mathbf{a}\cdot\mathbf{c} + \mathbf{b}\cdot\mathbf{c}.$$

Žodžių daugybą naudosime ne tik žodžių poroms, bet ir didesniems rinkiniams, pavyzdžiui,

$$\mathbf{a}\cdot\mathbf{b}\cdot\mathbf{c} = (\mathbf{a}\cdot\mathbf{b})\cdot\mathbf{c}.$$

Pastebėsime, jog su visais poaibiais $E_1, E_2, \ldots, E_s \subset \mathbb{F}_2^n$ teisinga lygybė:

$$\mathbf{v}(E_1)\cdot\mathbf{v}(E_2)\cdot\ldots\cdot\mathbf{v}(E_s) = \mathbf{v}(E_1 \cap E_2 \ldots \cap E_s). \tag{61}$$

72 apibrėžimas. *Tegu* $1 \le m,\ r \le m,\ n = 2^m$. *Reedo–Mullerio* $\mathbf{RM}(m, r)$ *kodu vadinsime tiesinį* $\mathbb{F}_2^n$ *poerdvį, kurį generuoja žodžiai*

$$\mathbf{v}_0,\ \mathbf{v}_{i_1}\cdots\mathbf{v}_{i_s}; \tag{62}$$

čia $\mathbf{v}_0 = 11\ldots1,\ \mathbf{v}_i = \mathbf{v}(H_i),\ 1 \le i_1 < \ldots < i_s \le m,\ s \le r,\ i = 1, \ldots, m.$

Vektorių, kurie užrašyti (62), skaičius lygus

$$k(m, r) = 1 + C_m^1 + C_m^2 + \ldots + C_m^r.$$

Kaip netrukus pamatysime, visi šie vektoriai yra tiesiškai nepriklausomi. Taigi $\mathbf{RM}(m, r)$ kodo generuojančią matricą gausime surašę šių vektorių elementus į matricos eilutes.

Iš pradžių pastebėkime tokią žodžių $\mathbf{v}_i$ savybę. Imkime poerdvių H_i pildinius

$$H_i^c = \{\mathbf{a} : \mathbf{a} \in \mathbb{F}_2^m, x_i(\mathbf{a}) = 1\}, \quad i = 1, \ldots, m.$$

Nesunku pastebėti, jog

$$\mathbf{v}(H_i^c) = \mathbf{v}_0 + \mathbf{v}_i. \tag{63}$$

89 teorema. $\mathbf{RM}(m, r)$ *kodo dimensija lygi* $k(m, r)$, *o (62) vektoriai sudaro jo bazę.*

Įrodymas. Imkime (62) vektorių sistemoje $r = m$. Tada gausime lygiai $n = 2^m$ vektorių; tokia yra ir visos erdvės $\mathbb{F}_2^n$ dimensija. Jeigu parodysime, jog kiekvieną $\mathbb{F}_2^n$ vektorių galima išreikšti šiais n vektoriais, tai galėsime teigti, kad jie sudaro tiesiškai nepriklausomą sistemą. Tada bet kokiam r (62) vektoriai irgi bus tiesiškai nepriklausomi, nes sudarys tiesiškai nepriklausomos sistemos posistemę.

Pakanka parodyti, kad kiekvieną $\mathbb{F}_2^n$ standartinės bazės vektorių galima išreikšti (62) vektorių tiesine kombinacija. Pavyzdžiui, imkime standartinės bazės vektorių $\mathbf{e}_i$, jo i-oji komponentė lygi vienetui, o visos kitos lygios nuliui. Prisiminę erdvės $\mathbb{F}_2^m$ poaibių ir $\mathbb{F}_2^n$ žodžių atitiktį, galime rašyti: $\mathbf{e}_i = \mathbf{v}(D)$; čia $D = \{\mathbf{a}_i\}$. Tegu $\mathbf{a}_i = a_1 \dots a_m$. Žymėdami $H_i(0) = H_i$ ir $H_i(1) = H_i^c$, gausime

$$D = \{\mathbf{a}_i\} = H_1(a_1) \cap \dots \cap H_m(a_m).$$

Pasirėmę (63), gausime

$$\mathbf{e}_i = \mathbf{v}(D) = \mathbf{v}(H_1(a_1)) \cdots \mathbf{v}(H_m(a_m)).$$

Tačiau $\mathbf{v}(H_i(a_i))$ lygus arba $\mathbf{v}_i$, arba $\mathbf{v}_0 + \mathbf{v}_i$. Pasinaudoję žodžių daugybos distributyvumo savybe, įsitikinsime, kad $\mathbf{e}_i$ yra (62) sistemos vektorių tiesinė kombinacija.

Pavyzdys. Užrašysime $\mathbf{RM}(3, 2)$ kodo bazę, o kartu ir kodo generuojančią matricą. Iš pradžių surašykime erdvės $\mathbb{F}_2^3$ žodžius, o pagal juos konstruokime ir bazės vektorius.

	$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	$\mathbf{a}_6$	$\mathbf{a}_7$	$\mathbf{a}_8$
	000	001	010	011	100	101	110	111
$\mathbf{v}_0$	1	1	1	1	1	1	1	1
$\mathbf{v}_1$	1	1	1	1	0	0	0	0
$\mathbf{v}_2$	1	1	0	0	1	1	0	0
$\mathbf{v}_3$	1	0	1	0	1	0	1	0
$\mathbf{v}_1 \cdot \mathbf{v}_2$	1	1	0	0	0	0	0	0
$\mathbf{v}_1 \cdot \mathbf{v}_3$	1	0	1	0	0	0	0	0
$\mathbf{v}_2 \cdot \mathbf{v}_3$	1	0	0	0	1	0	0	0

Fiksuotam m gavome ištisą Reedo-Mullerio kodų koloniją, kurios nariai yra $\mathbf{RM}(m, r)$, $r = 0, \dots, m$. Akivaizdu, kad $\mathbf{RM}(m, 0) = \{00 \dots 0, 11 \dots 1\}$, $\mathbf{RM}(m, m) = \mathbb{F}_2^n$.

Pastebėkime, kad visų Reedo-Mullerio kodo $\mathbf{RM}(m, r)$, kai $r < m$, žodžių svoriai yra lyginiai. Iš tikrųjų:

$$w(\mathbf{v}_0) = n = 2^m, \quad w(\mathbf{v}_{i_1} \cdots \mathbf{v}_{i_s}) = 2^{m-s}.$$

Iš lygybės

$$w(\mathbf{x} + \mathbf{y}) = w(\mathbf{x}) + w(\mathbf{y}) - 2w(\mathbf{x} \cdot \mathbf{y}), \quad \mathbf{x}, \mathbf{y} \in \mathbb{F}_2^n,$$

gauname, kad dviejų lyginio svorio žodžių sumos svoris irgi lyginis. Nebesunku įsitikinti, kad bet kokios (62) sistemos žodžių tiesinės kombinacijos svoris irgi lyginis.

Užrašykime tokią akivaizdžią lygybę:

$$1 + C_m^1 + C_m^2 + \ldots + C_m^r + C_m^{r+1} + \ldots + C_m^{m-1} + 1 = 2^m = n.$$

Pasinaudoję binominių koeficientų sąryšiu $C_m^k = C_m^{m-k}$ ir Reedo-Mullerio kodų dimensijų išraiškomis, galime šią lygybę užrašyti taip:

$$k(m,r) + k(m, m-r-1) = \dim(\mathbf{RM}(m,r)) + \dim(\mathbf{RM}(m, m-r-1)) = n.$$

Ar tik nebus Reedo-Mullerio kodų šeima (taip pat kaip ir Reedo-Solomono) „uždara“, t. y. šios šeimos nario dualus irgi priklauso tai pačiai šeimai? Tikra tiesa!

90 teorema. *Su visomis m, r $(m < r)$ reikšmėmis teisingas sąryšis* $\mathbf{RM}(m,r)^\perp = \mathbf{RM}(m, m-r-1)$.

Įrodymas. Kad teiginys būtų teisingas, reikia, kad būtų patenkintos dvi sąlygos: kodų $\mathbf{RM}(m,r), \mathbf{RM}(m, m-r-1)$ bazių žodžių skaliarinės sandaugos turi būti lygios nuliui moduliu 2, o dimensijų suma lygi n. Tačiau pastarąją lygybę jau nustatėme!

Tegu $\mathbf{a} = \mathbf{v}_{i_1} \cdots \mathbf{v}_{i_s}$ ir $\mathbf{b} = \mathbf{v}_{j_1} \cdots \mathbf{v}_{j_t}$ yra šių kodų bazių žodžiai, $s + t \leq r + m - r - 1 < m$. Sudauginę juos, gausime žodį

$$\mathbf{c} = \mathbf{v}_{l_1} \cdots \mathbf{v}_{l_z}, \quad z < m.$$

Tačiau

$$w(\mathbf{c}) \equiv (\mathbf{a}, \mathbf{b}) \pmod 2, \quad w(\mathbf{c}) \equiv 0 \pmod 2.$$

Taigi ir pirmoji dualumo sąlyga yra teisinga.

Galima į Reedo-Mullerio kodus pažvelgti visai kitaip. Pirmiausia kitaip pažvelkime į žodžių erdvės

$$\mathbb{F}_2^m = \{\mathbf{a}_1, \ldots, \mathbf{a}_n\}, \quad n = 2^m,$$

ir $\mathbb{F}_2^n$ sąryšį. Į kiekvieną žodį $\mathbf{f} \in \mathbb{F}_2^n$ galime žvelgti kaip į tam tikros funkcijos $f : \mathbb{F}_2^m \to \mathbb{F}_2$ reikšmių lentelę:

$$\text{jei } \mathbf{f} = y_1 y_2 \ldots y_n, \text{ tai } f(\mathbf{a}_1) = y_1, \ldots, f(\mathbf{a}_n) = y_n.$$

Kadangi šios funkcijos įgyja tik dvi reikšmes, galime jas interpretuoti kaip logines funkcijas. Pažymėję $\mathbf{a} = x_1 x_2 \ldots x_m$, $\mathbf{a} \in \mathbb{F}_2^m$, galime suvokti $\mathbb{F}_2^n$ žodžius kaip logines funkcijas $f(x_1, x_2, \ldots, x_m)$. Šios funkcijos sudaro tiesinę erdvę. Kokios funkcijos atitinka Reedo-Mullerio kodo bazės žodžius? Nesunku suprasti, kad

$$\begin{aligned} \mathbf{v}_0 &\mapsto 1, \ \mathbf{v}_i \mapsto 1 + x_i, \ i = 1, 2, \ldots, m, \\ \mathbf{v}_{i_1} \cdot \mathbf{v}_{i_2} \cdot \ldots \cdot \mathbf{v}_{i_s} &\mapsto (1 + x_{i_1})(1 + x_{i_2}) \cdots (1 + x_{i_s}) \\ &= 1 + x_{i_1} + \ldots + x_{i_1} x_{i_2} \cdots x_{i_s}. \end{aligned}$$

Taigi kodą $\mathbf{RM}(m,r)$ galime suvokti kaip ne didesnio nei r laipsnio „loginių daugianarių“ erdvę! Naudojantis šiuo požiūriu, galima toliau tirti Reedo-Mullerio kodų šeimos savybes. Pavyzdžiui, nustatyti minimalų kodų atstumą.

91 teorema. *Minimalus kodo* $\mathbf{RM}(m,r)$ *atstumas lygus* 2^{m-r}.

Kad minimalus kodo atstumas negali būti didesnis už 2^{m-r}, rodo kodo bazės žodžio svoris:

$$w(\mathbf{v}_1 \cdot \mathbf{v}_2 \cdots \mathbf{v}_r) = 2^{m-r}.$$

Norėdami įrodyti, kad ne mažesnis, turėtume kiek pasidarbuoti... Šį darbą atidėkime kitam skyreliui.

Dabar aptarkime Reedo-Mullerio kodų dekodavimą. Naudosime loginės daugumos (majority logic decoding, angl.) metodą – savotišką sprendimų priėmimą „balsavimo“ būdu.

Tarkime, informacija koduojama Reedo-Mullerio $\mathbf{RM}(m,r)$ kodu: kanalu siunčiami žodžiai $\mathbf{c} = c_1 \dots c_n,\ n = 2^m$,

$$\mathbf{c} = a(0)\mathbf{v}_0 + \sum_{\substack{1 \le i_1 < \dots < i_s \le m \\ s \le r}} a(i_1, \dots, i_s)\mathbf{v}_{i_1} \cdots \mathbf{v}_{i_s}; \tag{64}$$

čia $a(0), a(i_1, \dots, i_s) = 0$ arba 1. Kanalas galbūt iškraipė siunčiamus simbolius; pažymėkime gautąjį žodį $\mathbf{d} = d_1 \dots d_n, n = 2^m$. Naudodamiesi šiuo žodžiu, rasime teisingas koeficientų $a(0), a(i_1, \dots, i_s)$ reikšmes, taigi atstatysime siųstąjį žodį $\mathbf{c}$, jeigu įvykusių iškraipymų nėra daugiau kaip $(2^{m-r} - 1)/2$.

Dekoduoti pasirengsime taip: kiekvienam koeficientui $a(i_1, \dots, i_r)$ sudarysime lygiai 2^{m-r} išraiškų

$$a(i_1, \dots, i_r) = \sum_{i \in I_j} c_i, \quad j = 1, \dots, 2^{m-r}, \tag{65}$$

tokių, kad $|I_j| = 2^r$, $I_i \cap I_j = \emptyset$, jei $i \neq j$. Šios sąlygos reiškia, kad kievienoje iš (65) lygybių yra lygiai po 2^r dėmenų ir kiekvienas siunčiamo žodžio simbolis c_i pasirodo tik vienoje lygybėje. Jeigu iškraipyta ne daugiau kaip $(2^{m-r} - 1)/2$ žodžio $\mathbf{c}$ simbolių, tai ne daugiau kaip $(2^{m-r}-1)/2$ (65) lygybių nebebus teisingos. Tačiau ne mažiau kaip $(2^{m-r} + 1)/2$, t. y. daugiau kaip pusė, liks galioti. Tikrąją koeficiento reikšmę rasime suskaičiavę, kokių simbolių – vienetų ar nulių – yra daugiau sekoje

$$\sum_{i \in I_j} d_i, \quad j = 1, \dots, 2^{m-r}.$$

Daugiau kartų pasikartojęs simbolis ir bus koeficiento $a(i_1, \dots, i_r)$ reikšmė. Šitokiu būdu suradę visus koeficientus $a(i_1, \dots, i_r)$, galime iš gautojo žodžio $\mathbf{d}$ atimti atitinkamus r-osios eilės narius ir manyti, kad gautasis skirtumas

yra žodis, gautas siunčiant kodo $\mathbf{RM}(m, r-1)$ žodį. Tada analogiškai galime ieškoti koeficientų $a(i_1, \ldots, i_{r-1})$. Pagaliau suradę visus koeficientus $a(i_1, \ldots, i_s)$ ir iš gautojo žodžio atėmę atitinkamus dėmenis, gausime žodį $\mathbf{d}_0$, kurį galėsime interpretuoti kaip gautą siunčiant kanalu $a(0)\mathbf{v}_0$. Ar $a(0) = 0$, ar $a(0) = 1$, galėsime nuspręsti, suskaičiavę, ko daugiau – vienetų ar nulių yra žodyje $\mathbf{d}_0$.

Pastebėkime, kad koeficientai $a(i_1, \ldots, i_s)$ yra tie informaciniai bitai, kuriuos „neša“ kodo žodis. Taigi šiuo dekodavimo algoritmu iškart nustatome koduotos informacijos srauto dalį.

Tačiau kaip sudaryti (65) lygybes? Imkimės šio darbo.

Dekodavimui reikalingoms lygybėms sudaryti pasinaudosime skaliarine žodžių $\mathbf{a} = a_1 \ldots a_n, \mathbf{b} = b_1 \ldots b_n$ daugyba:

$$(\mathbf{a}, \mathbf{b}) = a_1 b_1 + \ldots + a_n b_n,$$

čia sudėtis imama kūne $\mathbb{F}_2$. Pastebėsime, jog

$$(\mathbf{v}(D), \mathbf{v}(E)) \equiv |D \cap E| \pmod 2. \tag{66}$$

Kaip ir anksčiau, vartosime žymenis

$$H_i(0) = H_i = \{\mathbf{a} : \mathbf{a} \in \mathbb{F}_2^m, x_i(\mathbf{a}) = 0\},\ H_i(1) = H_i^c,\ i = 1, \ldots, m.$$

Fiksuokime rinkinį $1 \le i_1 < \ldots < i_r \le m$. Tegu

$$\{l_1, \ldots, l_{m-r}\} = \{1, \ldots, m\} \backslash \{i_1, \ldots, i_r\};\ \text{čia } l_1 < \ldots < l_{m-r}.$$

Kiekvienam nulių ir vienetų rinkiniui $t = t_1 \ldots t_{m-r}$ apibrėžkime

$$\mathbf{w}_t = \mathbf{v}(H_{l_1}(t_1) \cap \ldots \cap H_{l_{m-r}}(t_{m-r})).$$

Iš viso turime 2^{m-r} žodžių $\mathbf{w}_t$. Svarbi įžvalga: kiekviename žodyje $\mathbf{w}_t$ yra lygiai 2^r vienetų ir nėra vieneto, kuris būtų toje pat pozicijoje skirtingiems $\mathbf{w}_t, \mathbf{w}_{t'}$.

Tegu $\mathbf{u} = \mathbf{v}_{j_1} \cdots \mathbf{v}_{j_s} = \mathbf{v}(H_{j_1}(0) \cap \ldots \cap H_{j_s}(0))$. Tada bet kokiam t

$$(\mathbf{v}, \mathbf{w}_t) \equiv |H_{j_1}(0) \cap \ldots \cap H_{j_s}(0) \cap H_{l_1}(t_1) \cap \ldots \cap H_{l_{m-r}}(t_{m-r})| (\mathrm{mod}\ 2).$$

Jei $\{j_1, \ldots, j_s\} = \{i_1, \ldots, i_r\}$, tai $\{j_1, \ldots, j_s\} \cup \{l_1, \ldots, l_{m-r}\} = \{1, 2, \ldots, n\}$. Tada $(\mathbf{v}, \mathbf{w}_t)$ išraiškos aibių sankirtai priklauso tik vienas žodis $\mathbf{a}$, kurį nusako lygybės

$$x_i(\mathbf{a}) = 0,\ \text{jei } i \in \{i_1, \ldots, i_r\};\ x_i(\mathbf{a}) = t_j,\ \text{jei } i = l_j.$$

Jeigu $\{j_1, \ldots, j_s\} \ne \{i_1, \ldots, i_r\}$, tai aibių sankirtai priklausančių žodžių skaičius yra lyginis. Taigi

$$(\mathbf{v}, \mathbf{w}_t) = \begin{cases} 0, & \text{jei } \{j_1, \ldots, j_s\} \ne \{i_1, \ldots, i_r\}, \\ 1, & \text{jei } \{j_1, \ldots, j_s\} = \{i_1, \ldots, i_r\}. \end{cases}$$

Padauginę (64) skaliariškai iš $\mathbf{w}_t$ ir atsižvelgdami į nustatytas lygybes, gausime

$$(\mathbf{c}, \mathbf{w}_t) = a(i_1, \ldots, i_r).$$

Tačiau tai ir yra loginės daugumos metodui reikalingos lygybės!

Pavyzdys. Panagrinėsime $\mathbf{RM}(3,1)$ dekodavimą. Prisiminkime kodo sudarymo lentelę:

	$\mathbf{a}_1$	$\mathbf{a}_2$	$\mathbf{a}_3$	$\mathbf{a}_4$	$\mathbf{a}_5$	$\mathbf{a}_6$	$\mathbf{a}_7$	$\mathbf{a}_8$
	000	001	010	011	100	101	110	111
$\mathbf{v}_0$	1	1	1	1	1	1	1	1
$\mathbf{v}_1$	1	1	1	1	0	0	0	0
$\mathbf{v}_2$	1	1	0	0	1	1	0	0
$\mathbf{v}_3$	1	0	1	0	1	0	1	0

Šio kodo žodžiai užrašomi taip:

$$\mathbf{c} = a(0)\mathbf{v}_0 + a(1)\mathbf{v}_1 + a(2)\mathbf{v}_2 + a(3)\mathbf{v}_3.$$

Lygybių sistemą sudarysime $a(3)$ koeficientui. Rinkinius $t = 00, 01, 10, 11$ atitinka žodžiai

$$\mathbf{w}_t = 11000000, 00110000, 00001100, 00000011.$$

Todėl dekodavimo lygybės atrodys taip:

$$\begin{aligned} a(3) &= c_1 + c_2, \\ a(3) &= c_3 + c_4, \\ a(3) &= c_5 + c_6, \\ a(3) &= c_7 + c_8. \end{aligned}$$

Tarkime, buvo siųstas žodis $\mathbf{c} = \mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_3 = 10010110$, tačiau vienas simbolis buvo iškraipytas ir gautasis žodis yra $\mathbf{d} = 11010110$. Pagal gautąjį žodį naudodamiesi dekodavimo lygybėmis, gauname reikšmes $0, 1, 1, 1$. Taigi $a(3) = 1$.

9.8. Veiksmai su kodais

Keletas visai paprastų ir kiek sudėtingesnių būdų konstruoti naujus kodus iš jau sudarytų.

Iš jau sudarytų kodų galima konstruoti naujus. Jeigu kodas tiesinis, tai galime sudaryti jam dualų. Tiesa, kartais būna, kad kodas yra dualus pats sau. Taigi tiesiniai kodai egzistuoja poromis, tačiau pasitaiko ir vienišių.

Štai dar vienas būdas sukurti naują kodą.

73 apibrėžimas. *Tegu $\mathbf{L} \subset \mathbb{F}_q^n$ yra tiesinis kodas. Jo plėtiniu (extended code) vadinsime kodą $\mathbf{L}^* \subset \mathbb{F}_q^{n+1}$,*

$$\mathbf{L}^* = \{c_1c_2 \dots c_nc_{n+1} : c_1c_2 \dots c_n \in \mathbf{L}, c_1 + c_2 + \dots + c_n + c_{n+1} = 0\}.$$

Jeigu tiesinio kodo $\mathbf{L}$ kontrolinė matrica yra

$$H = \begin{pmatrix} h_{11} & h_{12} & h_{13} & \dots & h_{1n} \\ h_{21} & h_{22} & h_{23} & \dots & h_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{r1} & h_{r2} & h_{r3} & \dots & h_{rn} \end{pmatrix},$$

tai jo plėtinio $\mathbf{L}^*$ kontrolinė matrica yra

$$H^* = \begin{pmatrix} h_{11} & h_{12} & h_{13} & \dots h_{1n} & 0 \\ h_{21} & h_{22} & h_{23} & \dots h_{2n} & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{r1} & h_{r2} & h_{r3} & \dots h_{rn} & 0 \\ 1 & 1 & 1 & \dots 1 & 1 \end{pmatrix}.$$

Plėtinys yra taip pat tiesinis kodas, jo dimensija yra ta pati kaip ir pradinio kodo. Jei $\mathbf{L}$ yra iš abėcėlės $\mathbb{F}_2$ žodžių sudarytas kodas, o jo minimalus atstumas d yra nelyginis, tai plėtinio minimalus atstumas yra $d + 1$.

Prisiminkime klasikinį Hammingo kodą $\mathbf{H}_2(3)$. Jo plėtinys suteikia galimybę atskirti, kada įvyko viena, o kada dvi klaidos, ir išvengti saviapgaulės taisant dvi klaidas kaip vieną. Tai teisinga ir bet kokiam Hammingo kodui $\mathbf{H}_2(r)$.

Dar dvi operacijos: kodo sutrumpinimas (*puncturing*) ir sumažinimas (*shortening*).

74 apibrėžimas. *Kodą, kuris gaunamas iš kodo $\mathbf{C}$, išbraukiant visų jo žodžių i-ąjį simbolį, vadinsime sutrumpintu $\mathbf{C}$ kodu.*

92 teorema. *Jei kodas* $\mathbf{C} \subset \mathbb{F}_q^n$ *yra tiesinis, tai ir sutrumpintas kodas* $\mathbf{C}^* \subset \mathbb{F}_q^{n-1}$ *yra tiesinis.*

Akivaizdu, kad tuo atveju, kai nubraukiamas simbolis visuose kodo žodžiuose lygus nuliui, sutrumpinto kodo dimensija ir minimalus atstumas yra tokie patys kaip ir pradinio kodo.

Jei pradinis kodas sutampa su visa žodžių erdve $\mathbb{F}_q^n$, tai sutrumpintas kodas – žodžių erdvė $\mathbb{F}_q^{n-1}$. Taigi abiejų kodų minimalus atstumas – tas pats vienetas, o sutrumpinto kodo dimensija vienetu mažesnė. Panagrinėkime dar vieną atvejį.

93 teorema. *Tegu* $\mathbf{C}$ *yra tiesinis kodas,* $\mathbf{C}^*$ *– sutrumpintas kodas. Jeigu* $d(\mathbf{C}) > 1$, *tai* $\dim(\mathbf{C}) = \dim(\mathbf{C}^*)$.

Įrodymas. Tegu $k = \dim(\mathbf{C})$ ir $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_k$ yra šio kodo bazės žodžiai. Pažymėkime $\mathbf{f}_1^*, \mathbf{f}_2^*, \ldots, \mathbf{f}_k^*$ žodžius, gautus iš bazės žodžių juos sutrumpinus. Tegu komponentės, kurios buvo nubrauktos, yra $u_1, \ldots, u_k$. Jeigu visos jos lygios nuliui, tai sutrumpinto kodo dimensija ta pati kaip pradinio kodo. Taigi tarsime, kad ne visos nubrauktosios komponentės lygios nuliui.

Visi sutrumpinto kodo žodžiai yra tiesinės žodžių $\mathbf{f}_1^*, \mathbf{f}_2^*, \ldots, \mathbf{f}_k^*$ kombinacijos. Įrodysime, kad žodžiai $\mathbf{f}_j^*$ yra tiesiškai nepriklausomi. Tarkime priešingai – su tam tikrais elementais $\alpha_i \in \mathbb{F}_q$, kurie ne visi lygūs nuliui, teisinga lygybė

$$\alpha_1 \mathbf{f}_1^* + \alpha_2 \mathbf{f}_2^* + \ldots + \alpha_k \mathbf{f}_k^* = \mathbf{0}.$$

Nagrinėkime reiškinį, sudarytą panaudojant $\alpha_i \in \mathbb{F}_q$ ir išbrauktąsias komponentes:

$$\beta = \alpha_1 u_1 + \alpha_2 u_2 + \ldots + \alpha_k u_k.$$

Jeigu būtų $\beta = 0$, gautume, kad $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_k$ yra tiesiškai priklausomi. Jeigu būtų $\beta \neq 0$, gautume, kad $d(\mathbf{C}) = 1$. Abi išvados prieštarauja teoremos prielaidoms, taigi sutrumpinto kodo dimensija ta pati kaip ir pradinio kodo.

75 apibrėžimas. *Kodą, kuris gaunamas iš* $\mathbf{C}$, *surinkus jo žodžius, turinčius tą patį* i*-ąjį simbolį, ir jį išbraukus, vadinsime sumažintu* $\mathbf{C}$ *kodu.*

94 teorema. *Tegu* $\mathbf{C} \subset \mathbb{F}_q^n$ *yra tiesinis* $[n, k, d]$ *kodas ir ne visų jo žodžių* i*-oji komponentė yra nulinė. Jei* $\mathbf{C}^*$ *yra sumažintas kodas, gautas surinkus visus kodo žodžius, turinčius nulinę* i*-ąją komponentę, ir ją išbraukus, tai* $\mathbf{C}^*$ *yra tiesinis kodas su parametrais* $[n-1, k-1, d^*]$, *čia* $d^* \geq d$.

Įrodymas. Kad sumažintas kodas yra tiesinis, akivaizdu. Taip pat akivaizdu, kad $d^* \geq d$.

Apibrėžtumo dėlei tarkime, $i = n$. Tegu G yra kodo $\mathbf{C}$ generuojanti, H – kontrolinė matrica, o H^* – matrica H be paskutinio stulpelio. Tada

$$\mathbf{C} = \{\mathbf{x} \in \mathbb{F}_q^n : \mathbf{x}H^T = \mathbf{0}\}, \quad \mathbf{C}^* = \{\mathbf{x} \in \mathbb{F}_q^{n-1} : \mathbf{x}H^{*T} = \mathbf{0}\}.$$

Taigi H^* yra kontrolinė kodo $\mathbf{C}^*$ matrica. Jeigu įrodytume, kad matricos H^* rangas toks pat kaip H, tai gautume, kad $\dim(\mathbf{C}^*) = n-1-\dim(\mathbf{C}^\perp) = k-1$.

Taigi reikia įrodyti, kad, nubraukus n-ąjį matricos H stulpelį, sutrumpėjusios matricos eilutės netampa tiesiškai priklausomos. Jeigu taip būtų, tai iš H^* eilučių galėtume sudaryti nulinę eilutę; tada matricos H eilučių su tais pačiais koeficientais tiesinė kombinacija būtų eilutė, turinti tik n-ąjį nenulinį elementą, t. y. egzistuotų kodo $\mathbf{C}^\perp$ žodis $\mathbf{a} = 00\ldots0u, u \neq 0$. Tada turėtų būti

$$\mathbf{a}G^T = \mathbf{0}.$$

Taip gali būti tik tada, kai n-asis matricos G stulpelis sudarytas vien iš nulių. Tačiau taip nėra, nes ne visų kodo $\mathbf{C}$ žodžių n-oji komponentė yra nulinė. Teorema įrodyta.

Turėdami du tiesinius kodus, kurių ilgiai nebūtinai vienodi, galime sudaryti naują tiesinį kodą, tiesiog prijungdami prie pirmojo kodo žodžių antrojo kodo žodžius. Nesunku įsitikinti, kad tokio kodo minimalus atstumas būtų lygus kodų, iš kurių jis sudarytas, minimalių atstumų minimumui, o dimensija – kodų dimensijų sumai. Taigi kodas garantuotai taisytų tiek pat klaidų kiek „blogesnysis“ iš abiejų kodų, tačiau – ilgesniame žodyje.

Yra ir geresnių būdų naujam kodui gauti sujungiant dviejų turimų kodų žodžius.

76 apibrėžimas. *Tegu* $\mathbf{L}_1, \mathbf{L}_2 \subset \mathbb{F}_q^n$ *yra du tiesiniai kodai. Kodu* $\mathbf{L}_1|\mathbf{L}_2$ *vadinsime tiesinį kodą*

$$\mathbf{L}_1|\mathbf{L}_2 = \{\mathbf{x}|\mathbf{x}+\mathbf{y} : \mathbf{x} \in \mathbf{L}_1, \mathbf{y} \in \mathbf{L}_2\}.$$

Šis naujų kodų sudarymo būdas kodavimo teorijoje vadinamas $u|u+v$ konstrukcija.

95 teorema. *Jei* $\mathbf{L}_1, \mathbf{L}_2$ *yra du tiesiniai kodai iš aibės* $\mathbf{F}_q^n$ *žodžių,* d_1, d_2 *– jų minimalūs atstumai, tai kodo* $\mathbf{L}_1|\mathbf{L}_2$ *dimensija lygi kodų* $\mathbf{L}_1, \mathbf{L}_2$ *dimensijų sumai, o minimalus atstumas yra* $d = \min(2d_1, d_2)$.

Įrodymas. Jeigu $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k$ yra pirmojo, o $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_l$ – antrojo kodo bazė, tai nesunku įsitikinti, kad $\mathbf{e}_1|\mathbf{e}_1, \mathbf{e}_2|\mathbf{e}_2, \ldots, \mathbf{e}_k|\mathbf{e}_k, \mathbf{0}|\mathbf{f}_1, \mathbf{0}|\mathbf{f}_2, \ldots, \mathbf{0}|\mathbf{f}_l$ – kodo $\mathbf{L}_1|\mathbf{L}_2$ bazė, čia $\mathbf{0} = 00\ldots0$.

Taip pat beveik akivaizdu, kad yra kodo $\mathbf{L}_1|\mathbf{L}_2$ žodžių, kurių svoriai lygūs $2d_1$ ir d_2. Pakanka įsitikinti, kad bet kuriam kitam šio kodo žodžiui $\mathbf{x}$ teisinga arba nelygybė $w(\mathbf{x}) \geq 2d_1$, arba $w(\mathbf{x}) \geq d_2$, čia $w(\mathbf{x})$ kaip visada žymime žodžio $\mathbf{x}$ svorį.

Jei $\mathbf{u} \in \mathbf{L}_1, \mathbf{v} \in \mathbf{L}_2$, tai $\mathbf{u}|\mathbf{u}+\mathbf{v} \in \mathbf{L}_1|\mathbf{L}_2$ ir $w(\mathbf{u}|\mathbf{u}+\mathbf{v}) = w(\mathbf{u})+w(\mathbf{u}+\mathbf{v})$. Jei žodis $\mathbf{v}$ sudarytas vien tik iš nulinių elementų, tai

$$w(\mathbf{u}|\mathbf{u}+\mathbf{v}) = w(\mathbf{u}) + w(\mathbf{u}) \geq 2d_1.$$

Jei $\mathbf{v}$ ne vien tik iš nulių sudarytas žodis, tai pasinaudokime nelygybe

$$w(\mathbf{u}) + w(\mathbf{u}+\mathbf{v}) \geq w(\mathbf{v}) \geq d_2.$$

Kodėl nelygybė $w(\mathbf{u}) + w(\mathbf{u}+\mathbf{v}) \geq w(\mathbf{v})$ teisinga? Svarstykime taip: jeigu, pridėjus prie $\mathbf{v}$ žodį $\mathbf{u}$, kurios nors $\mathbf{v}$ komponentės pavirto nuliais, tai žodžio $\mathbf{u}$ komponentės su tais pačiais numeriais nelygios nuliui. Taigi svorių suma $w(\mathbf{u}) + w(\mathbf{u}+\mathbf{v})$ ne mažesnė už svorį $w(\mathbf{v})$.

Panagrinėkime vieną įdomų šios konstrukcijos taikymo pavyzdį. Jeigu įsigilinote į ankstesniame skyrelyje išdėstytą Reedo-Mullerio kodų $\mathbf{RM}(m,r)$ konstrukciją, tai žinote, kad šio kodo žodžius galima gauti iš ne didesnio kaip r-ojo laipsnio loginių daugianarių

$$f(x_1, x_2, \ldots, x_m) = a(0) + \sum_{1 \leq i_1 < i_2 < \ldots < i_s \leq m} a(i_1, i_2, \ldots, i_s) x_{i_1} x_{i_2} \cdots x_{i_s},$$

čia $a(0), a(i_1, i_2, \ldots, i_s) \in \mathbb{F}_2, s \leq r$. Žodžiai gaunami surašant funkcijos reikšmes, kai $x_1 x_2 \ldots x_m$ perbėga atitinkama tvarka visus $\mathbb{F}_2^m$ žodžius. Pavyzdžiui, funkcija

$$f(x_1, x_2) = 1 + x_1 + x_2$$

apibrėžia tokį erdvės $\mathbb{F}_2^2$ žodį:

$$\mathbf{v} = f(0,0)f(0,1)f(1,0)f(1,1) = 1001.$$

Tačiau šią funkciją galime panaudoti ir erdvės $\mathbb{F}_2^3$ žodžiui gauti. Jis bus toks:

$$10011001 = \mathbf{v}|\mathbf{v}.$$

Nagrinėkime kodą $\mathbf{RM}(m+1, r)$. Visus jo žodžius galime gauti iš ne didesnio kaip r-ojo laipsnio loginių daugianarių $f(x_1, x_2, \ldots, x_m, x_{m+1})$. Kiekvieną iš šių daugianarių galime užrašyti taip:

$$f(x_1, x_2, \ldots, x_m, x_{m+1}) = f_1(x_1, x_2, \ldots, x_m) + x_{m+1} f_2(x_1, x_2, \ldots, x_m),$$

čia f_1 yra ne didesnio kaip r-ojo laipsnio, o f_2 – ne didesnio kaip $r-1$-ojo laipsnio loginiai daugianariai.

Daugianaris f_1 apibrėžia erdvės $\mathbb{F}_2^m$ žodį $\mathbf{u}$, kuris priklauso kodui $\mathbf{RM}(m,r)$. Savo ruožtu šis daugianaris apibrėžia erdvės $\mathbb{F}_2^{m+1}$ žodį $\mathbf{u}|\mathbf{u}$. Daugianaris f_2 apibrėžia erdvės $\mathbb{F}_2^m$ žodį $\mathbf{v}$, kuris priklauso kodui $\mathbf{RM}(m, r-1)$, o daugianaris $x_{m+1} f_2(x_1, x_2, \ldots, x_m)$ – erdvės $\mathbb{F}_2^{m+1}$ žodį $00\ldots0|\mathbf{v}$. Taigi daugianaris f apibrėžia kodo $\mathbf{RM}(m+1, r)$ žodį

$$\mathbf{u}|\mathbf{u}+\mathbf{v}, \quad \mathbf{u} \in \mathbf{RM}(m,r),\ \mathbf{v} \in \mathbf{RM}(m, r-1).$$

Suformuluokime ką tik įrodytą teiginį.

96 teorema. *Reedo-Mullerio kodams teisingi sąryšiai*

$$\mathbf{RM}(m+1, r) = \mathbf{RM}(m,r)|\mathbf{RM}(m, r-1).$$

Pasinaudoję šiuo sąryšiu, galime įrodyti ankstesniame skyrelyje minėtą teiginį apie minimalų Reedo-Mullerio kodo atstumą.

97 teorema. *Minimalus kodo* $\mathbf{RM}(m,r), 0 \le r \le m$, *atstumas lygus* 2^{m-r}.

Įrodymas. Lygybes $d(\mathbf{RM}(m,r)) = 2^{m-r},\ 0 \le r \le m$ įrodysime pasinaudoję matematine indukcija. Iš konstrukcijos aišku, kad

$$\mathbf{RM}(m,0) = \{00\ldots00, 11\ldots11\}, \quad \mathbf{RM}(m,m) = \mathbb{F}_2^n$$

ir minimalaus atstumo išraiškos, kai $r = 0$ ir $r = m$, teisingos. Atskiru atveju, kai $m = 1$, lygybės teisingos su visomis (abiem) r reikšmėmis. Tarkime, teisingos lygybės

$$d(\mathbf{RM}(k,r)) = 2^{k-r},\ 0 \le r \le k.$$

Kadangi $\mathbf{RM}(k+1,r) = \mathbf{RM}(k,r)|\mathbf{RM}(k,r-1)$, tai

$$d(\mathbf{RM}(k+1,r)) = \min(2d(\mathbf{RM}(k,r)), d(\mathbf{RM}(k,r-1)) = 2^{k+1-r}.$$

Iš lygybių su $m = k$ gavome lygybes su $m = k+1$. Teorema įrodyta.

Išnagrinėjome savotišką kodų sudėties veiksmą. Yra ir kodų daugyba.

Tegu $\mathbf{C_1} \subset \mathbb{F}_q^{n_1}, \mathbf{C_2} \subset \mathbb{F}_q^{n_2}$ yra du tiesiniai kodai, kurių parametrai yra $[n_1,k_1,d_1]$ ir $[n_2,k_2,d_2]$. Naudodamiesi šiais kodais, sudarysime kodą $\mathbf{C} \subset \mathbb{F}_q^n$, kurio parametrai bus $[n,k,d], n = n_1n_2, k = k_1k_2, d = d_1d_2$. Šį kodą vadinsime kodų $\mathbf{C}_1$ ir $\mathbf{C}_2$ sandauga ir žymėsime $\mathbf{C} = \mathbf{C}_1 \times \mathbf{C}_2$.

Iš pradžių panagrinėkime, kaip šiuo kodu gali būti koduojama. Taigi reikia apibrėžti kodavimo taisyklę

$$\mathbf{x} \in \mathbb{F}_q^k,\ \mathbf{x} \mapsto \mathbf{c},\ \mathbf{c} \in \mathbb{F}_q^n,\ k = k_1k_2, n = n_1n_2.$$

Pirmiausia žodį $\mathbf{x} \in \mathbb{F}_q^k$ padalykime į k_2 žodžių iš $\mathbb{F}_q^{k_1}$:

$$\mathbf{x} = \mathbf{x}_1\mathbf{x}_2\ldots\mathbf{x}_{k_2}.$$

Dabar žodžius $\mathbf{x}_i$ koduokime atitinkamais kodo $\mathbf{C_1}$ žodžiais $\mathbf{y}_i$ ir surašykime šiuos žodžius į matricos Y eilutes. Matricos Y matavimai – $k_2 \times n_1$. Iš šios matricos stulpelių sudarykime žodžius $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{n_1}$. Šiuos žodžius koduokime atitinkamais kodo $\mathbf{C}_2$ žodžiais $\mathbf{v}_i$. Šiuos žodžius surašę stulpeliais gausime $n_2 \times n_1$ eilės matricą; surašę jos elementus eilutė po eilutės (arba stulpelis po stulpelio), gausime kodo žodį $\mathbf{c} \in \mathbb{F}_q^n$.

Panagrinėkime pavyzdį. Tegu abu kodai yra dvejetainiai, o generuojančios matricos lygios

$$G_1 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix}, \quad G_2 = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

Tegu reikia koduoti šešių bitų žodį

$$\mathbf{x} = 110110 = 11|01|10.$$

Po pirmojo žingsnio – kodavimo kodu $\mathbf{C}_1$ – gausime tokią matricą, sudarytą iš kodo $\mathbf{C}_1$ žodžių:

$$\mathbf{Y} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{pmatrix} G_1 = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

Dabar matricos $\mathbf{Y}$ stulpelius reikia koduoti $\mathbf{C}_2$ kodu ir gautus žodžius surašyti į naujos matricos stulpelius. Tai galime padaryti taip: transponuoti $\mathbf{Y}$, padauginti iš G_2 ir sandaugą vėl transponuoti:

$$\mathbf{V} = (\mathbf{Y}^T G_2)^T = G_2^T \mathbf{Y} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

Dabar panagrinėkime abstrakčiau. Tegu abiejų kodų generuojančios matricos yra standartinio pavidalo: $G_1 = (I_{k_1}, A_1), G_2 = (I_{k_2}, A_2)$. Pažymėkime $\mathbf{X}$ – matricą, sudarytą koduojamą $k_1 k_2$ ilgio informacinių blokų žodį surašius į $k_2 \times k_1$ matavimų matricą. Tada po pirmojo kodavimo žingsnio gausime matricą

$$\mathbf{Y} = \mathbf{X} G_1 = (\mathbf{X}, \mathbf{X} A_1).$$

Po antrojo kodavimo žingsnio gautą matricą galime užrašyti taip:

$$\mathbf{V} = (\mathbf{Y}^T G_2)^T = G_2^T \mathbf{Y} = \begin{pmatrix} \mathbf{X} & \mathbf{X} A_1 \\ A_2^T \mathbf{X} & A_2^T \mathbf{X} A_1 \end{pmatrix}.$$

Galima kodų sandaugą apibrėžti ir visiškai abstrakčiai.

77 apibrėžimas. *Tegu $\mathbf{C_1} \subset \mathbb{F}_q^{n_1}, \mathbf{C_2} \subset \mathbb{F}_q^{n_2}$ yra du tiesiniai kodai, kurių parametrai yra $[n_1, k_1, d_1]$ ir $[n_2, k_2, d_2]$, o generuojančios matricos G_1, G_2. Jų sandauga vadinsime kodą iš $\mathbb{F}_q^{n_1 n_2}$ žodžių, kurie gaunami iš matricų $G_2^T \mathbf{X} G_1$, surašius jų elementus į žodžius.*

98 teorema. *Jei $\mathbf{C_1} \subset \mathbb{F}_q^{n_1}, \mathbf{C_2} \subset \mathbb{F}_q^{n_2}$ yra tiesiniai kodai, kurių parametrai yra $[n_1, k_1, d_1]$ ir $[n_2, k_2, d_2]$, tai jų sandaugos parametrai yra $[n_1 n_2, k_1 k_2, d_1 d_2]$.*

Įrodymas. Dviejų kodų sandaugą apibrėžėme naudodami generuojančias matricas. Pagalvojus reiktų įsitikinti, kad, daugindami pirmiesiems ekvivalenčius kodus, gautume kodą, kuris yra ekvivalentus pirmųjų sandaugai. Todėl pakanka nagrinėti atvejį, kai generuojančios matricos yra standartinio pavidalo. Tada kodų sandaugos žodžiai gaunami iš matricų

$$\mathbf{V} = \begin{pmatrix} \mathbf{X} & \mathbf{X}A_1 \\ A_2^T\mathbf{X} & A_2^T\mathbf{X}A_1 \end{pmatrix}, \tag{67}$$

čia $\mathbf{X}$ yra $k_2 \times k_1$ matavimų matrica, sudaryta iš koduojamų simbolių. Įsitikinti, kad sandaugos dimensija yra lygi k_1k_2, galime nustatę, kad visi kodų sandaugos žodžiai yra tiesiškai nepriklausomų žodžių, gaunamų iš matricų $\mathbf{X}$, kuriose vienas elementas lygus 1, o kiti nuliai, tiesinės kombinacijos. Gerai įsižiūrėkite į (67), turėtų paaiškėti.

Dabar įrodykime teiginį apie minimalų atstumą. Pastebėkime, kad matricos (67) eilutėse surašyti kodo $\mathbf{C}_1$, o stulpeliuose – kodo $\mathbf{C}_2$ žodžiai. Keletas eilučių ir stulpelių gali būti vien tik iš nulių. Tačiau jeigu eilutė nenulinė, tai joje yra ne mažiau kaip d_1 nenulinių elementų, jeigu stulpelis nenulinis – jame ne mažiau kaip d_2 nenulinių elementų.

Nagrinėkime nenulinį kodų sandaugos žodį atitinkančią matricą $\mathbf{V}$. Kadangi ji turi stulpelį, kuriame yra ne mažiau kaip d_2 nenulinių elementų, tai ji turi ne mažiau kaip d_2 nenulinių eilučių. Taigi žodžio svoris ne mažesnis už d_1d_2. Beliko įsitikinti, kad yra lygiai d_1d_2 nenulinių elementų turinčių matricų $\mathbf{V}$. Tokį žodį galime sudaryti taip. Tegu $\mathbf{x} \in \mathbb{F}_q^{k_1}$ yra toks žodis, kad $w(\mathbf{x}G_1) = d_1$ ir $\mathbf{y} = y_1y_2 \dots y_{k_2} \in \mathbb{F}_q^{k_2}$, kad $w(\mathbf{y}G_2) = d_2$. Dabar sudarykime matricą $\mathbf{X}$, surašydami į jos eilutes žodžius $y_j\mathbf{x}$:

$$\mathbf{X} = \begin{pmatrix} y_1\mathbf{x} \\ y_2\mathbf{x} \\ \dots \\ y_{k_2}\mathbf{x} \end{pmatrix}.$$

Jeigu su šia matrica $\mathbf{X}$ sudarysime $\mathbf{V}$, joje bus lygiai d_1 nenulinių stulpelių, visi kiti – nuliniai. Be to, nenuliniai stulpeliai bus kodo $\mathbf{C}_2$ žodžiai, kurių svoris d_2. Taigi matricos $\mathbf{V}$ (kodų sandaugos žodžio) svoris bus tiksliai lygus d_1d_2.

Panagrinėsime dar vieną kodų konstrukciją.

78 apibrėžimas. *Tegu* $\mathbf{L}$ *yra dvinaris* $[n, k, d]$ *kodas, kurio generuojanti matrica yra*

$$G = \left(\begin{array}{cccc|ccc} 1 & 1 & \dots & 1 & 0 & \dots & 0 \\ & & G_1 & & & G_2 & \end{array}\right),$$

čia pirmieji d pirmosios eilutės elementai lygūs vienetui. Tiesinis kodas, kurio generuojanti matrica yra G_2, vadinamas liekanų kodu (residual code).

Pastebėkime, kad kiekvieno kodo generuojančiai matricai galima elementariais pertvarkiais suteikti į apibrėžime naudojamą.

99 teorema. *Tegu* $\mathbf{L}$ *yra dvinaris* $[n,k,d]$ *tiesinis kodas. Tada jo liekanų kodo parametrai yra* $[n-d,k-1,d']$*, o minimalus atstumas* d' *tenkina nelygybę* $d' \geq d/2$.

Įrodymas.

Tegu x yra liekanų kodo žodis, tada x yra tam tikra matricos G_2 eilučių tiesinė kombinacija, sudaryta su ne visais nuliniais koeficientais. Jei y yra analogiška matricos G_1 eilučių kombinacija, tai $y|x \in \mathbf{L}$. Jeigu būtų $x = 00\ldots0$, tai gautume, kad $w(y|x) = w(y) < d$ (reikia kiek pagalvoti, kodėl). Tai prieštaravimas. Taigi G_2 eilutės yra tiesiškai nepriklausomos, o liekanų kodo dimensija lygi $k-1$.

Pridėkime prie žodžio $y|x$ pirmąją matricos G eilutę:

$$y'|x = y|x + 11\ldots1|00...0 \in \mathbf{L}.$$

Taigi $y|x, y'|x \in \mathbf{L}$ ir

$$\begin{aligned} w(y|x) &= w(y) + w(x) \geq d, \\ w(y'|x) &= w(y') + w(x) \geq d. \end{aligned}$$

Tačiau arba $w(y) \leq d/2$, arba $w(y') \leq d/2$. Iš bet kurios šių nelygybių išplaukia, kad kodo $\mathbf{L}'$ žodžiui x teisinga nelygybė $w(x) \geq d/2$. Kadangi ši nelygybė teisinga bet kokiam kodo žodžiui, tai turi būti teisinga ir minimaliam atstumui $d' \geq d/2$.

Liekanų kodo dimensija vienetu mažesnė už pradinio kodo dimensiją. Tarkime, $\mathbf{L}_0$ yra dvejetainės abėcėlės žodžių $[n,k,d_0]$ kodas. Jo liekanų kodo $\mathbf{L}_1$ parametrai bus $[n-d_0, k-1, d_1]$; pastarojo kodo liekanų kodo – $[n-d_0-d_1, k-2, d_2]$. Toliau konstruodami liekanų kodus, turėsime sustoti gavę kodą su parametrais $[n-d_0-\ldots-d_{k-2}, 1, d_{k-1}]$. Šio kodo žodžių ilgis, žinoma, ne mažesnis už minimalų atstumą d_{k-1}, taigi

$$n \geq d_0 + d_1 + \ldots + d_{k-2} + d_{k-1}.$$

Tačiau $d_j \geq d_{j-1}/2$, todėl $d_j \geq d_0/2^j$. Taigi

$$d_0\Big(1 + \frac{1}{2} + \ldots + \frac{1}{2^{k-1}}\Big) \leq n, \quad d_0 \leq n\frac{2^{k-1}}{2^k-1}.$$

Įrodėme dar vieną įvertį tiesinio kodo parametrams. Analogiškas įvertis teisingas ir bendruoju atveju.

100 teorema. *Tiesiniam* $\mathbb{F}_q$ *abėcėlės kodui su parametrais* $[n,k,d]$ *teisinga nelygybė*

$$\sum_{i=0}^{k-1} \frac{d}{q^i} \leq n, \quad d \leq n\frac{(q-1)q^{k-1}}{q^k-1}.$$

Šis nelygybė vadinama Griesmerio įverčiu.

9.9. Šis tas apie žodžių svorius

Norint sužinoti svorį, reikia sverti. Šiame skyrelyje sukursime būdą kodo žodžių svoriams surasti, jų visai nesveriant, netgi visai nežinant, kaip jie atrodo.

Kodo žodžio $\mathbf{x}$ svoriu vadiname nenulinių jo komponenčių skaičių. Svorį žymime $w(\mathbf{x})$. Kartais, nagrinėjant kodo savybes, svarbu žinoti, kiek ir kokio svorio žodžių jis turi. Jau tyrinėjome Hammingo kodų svorius, nustatėme rekurentinius sąryšius, kuriais naudojantis galima nustatyti, kiek ir kokio svorio žodžių yra. Šiame skyrelyje sukursime įrankį, tinkantį bet kokių tiesinių kodų svoriams tyrinėti.

Apibrėšime funkciją, kuri saugo informaciją apie kodo žodžių svorius.

79 apibrėžimas. *Tegu $\mathbf{C}$ yra n ilgio abėcėlės $\mathbb{F}_q$ žodžių kodas, $A_i = |\{\mathbf{c} \in \mathbf{C} : w(\mathbf{c}) = i\}|$. Funkciją*

$$w_{\mathbf{C}}(x, y) = \sum_{i=0}^{n} A_i x^{n-i} y^i$$

vadinsime kodo $\mathbf{C}$ svorių funkcija, o skaičių A_i seką – kodo svorių skirstiniu.

Pastebėsime, kad svorių pasiskirstymo funkciją galima ir taip užrašyti:

$$w_{\mathbf{C}}(x, y) = \sum_{\mathbf{c} \in \mathbf{C}} x^{n-w(\mathbf{c})} y^{w(\mathbf{c})}.$$

Dažnai naudojamas kitas svorių pasiskirstymo funkcijos variantas

$$w_{\mathbf{C}}(z) = w_{\mathbf{C}}(1, z) = \sum_{i=0}^{n} A_i z^i = \sum_{\mathbf{c} \in \mathbf{C}} z^{w(\mathbf{c})}.$$

Vienas pagrindinių rezultatų, gautų nagrinėjant kodo svorių funkciją, yra matematikės F. J. McWilliams įrodyta tapatybė, kurią tuoj suformuluosime.

101 teorema. *Tegu $\mathbf{L}$ yra tiesinis abėcėlės $\mathbb{F}_q$ žodžių kodas, $\mathbf{L}^\perp$ jo dualus kodas. Tada abiejų kodų svorių pasiskirstymo funkcijas sieja tapatybė*

$$w_{\mathbf{L}^\perp}(x, y) = \frac{1}{|\mathbf{L}|} w_{\mathbf{L}}(x + (q-1)y, x - y). \tag{68}$$

Tai „giluminius“ kodų sąryšius atskleidžianti teorema. Nėra ko tikėtis, kad jos įrodymas bus visai paprastas. Pirmiausia patyrinėkime, kaip ją galima taikyti.

Pavyzdys. Tegu $\mathbf{L} \subset \mathbb{F}_2^5$ yra tiesinis kodas, kurį generuoja du žodžiai: $\mathbf{v}_1 = 10101$ ir $\mathbf{v}_2 = 01010$. Kodas turi tik keturis žodžius, visi jie skirtingo svorio. Svorių skirstinys toks: $A_0 = A_2 = A_3 = A_5 = 1$. Dualus kodas irgi

nedidelis, jis turi aštuonis žodžius. Kokie jų svoriai? Galime nustatyti nesukonstravę paties kodo. Pažymėkime dualaus kodo svorių skirstinio elementus $B_0, B_1, \ldots, B_5$. Žinome tik $B_0 = 1$. Užrašykime McWilliams tapatybę, įstatydami į (68) reikšmes $q = 2, x = 1$:

$$B_0 + B_1 y + \ldots + B_5 y^5 = \frac{1}{4}\left((1+y)^5 + (1+y)^4(1-y) + \ldots + (1-y)^5\right).$$

Dešinės pusės daugianarį galime sutvarkyti laipsnių didėjimo tvarka:

$$1 + 4y^2 + 3y^4.$$

Štai kokį dualaus kodo svorių skirstinį gavome: $B_0 = 1, B_2 = 4, B_4 = 3$. Matome, kad dualaus kodo minimalus atstumas lygus 4, taigi kodas taiso vieną klaidą.

Dar vienas pavyzdys. Nagrinėdami maksimalaus atstumo kodus, suradome daug savidualių kodų porų. Pavyzdžiui, nustatėme, kad savidualų kodą galima sudaryti renkant žodžius iš $\mathbb{F}_q^{q+1}$, čia q yra nelyginis pirminis arba pirminio laipsnis. Netgi sukonstravome jį – tai kiek modifikuotas Reedo-Solomono kodas $\mathbf{RS}^*_{q+1,k}, k = \frac{q+1}{2}$. Kiek ir kokio svorio žodžių jis turi?

Panagrinėkime atvejį $\mathbf{L} = \mathbf{RS}^*_{4,2}$, taigi $q = 3$. Kadangi $\mathbf{L} = \mathbf{L}^\perp$, o kodo dimensija yra lygi 2, tai, pažymėję kodo skirstinį A_0, A_1, A_2, A_3, A_4 ($A_0 = 1$) ir įstatę $x = 1, q = 3$, galėsime (68) užrašyti taip:

$$A_0 + A_1 y + A_2 y^2 + A_3 y^3 + A_4 y^4 = \frac{1}{9}\left((1+2y)^4 + A_1(1+2y)^3(1-y)\right.$$
$$\left. + A_2(1+2y)^2(1-y)^2 + A_3(1+2y)(1-y)^3 + A_4(1-y)^4\right).$$

Užrašę dešiniąją pusę kaip daugianarį pagal y laipsnius ir sulyginę atitinkamus kairės ir dešinės pusių daugianarių koeficientus, gauname penkių lygčių sistemą. Ją išsprendę gauname, kad $A_3 = 8$, o kiti svorių skirstinio elementai lygūs nuliui. Taigi savidualaus kodo $\mathbf{L}$ visi nenuliniai žodžiai turi po tris nenulines komponentes. Šiuo atveju MacWilliams tapatybė atrodo taip:

$$1 + 8y^3 = \frac{1}{9}(1+2y)^4 + \frac{8}{9}(1+2y)(1-y)^3.$$

Dabar pabandykime įrodyti McWilliams tapatybę. Įrodinėsime ją tik dvejetainės abėcėlės atveju, taigi $q = 2$. Tada tapatybė virsta tokia:

$$w_{\mathbf{L}^\perp}(x, y) = \frac{1}{|\mathbf{L}|} w_{\mathbf{L}}(x+y, x-y). \tag{69}$$

Įrodymui prireiks kelių pagalbinių teiginių.

Dvejetainės abėcėlės žodžių $\mathbf{x} = x_1 \ldots x_n, \mathbf{y} = y_1 \ldots y_n$ skaliarinę sandaugą žymėsime kaip anksčiau

$$(\mathbf{x}, \mathbf{y}) = x_1 y_1 + x_2 y_2 + \ldots + x_n y_n.$$

Čia sudėtis atliekama moduliu 2.

Toliau $\mathbf{L}$ žymi tiesinį abėcėlės $\mathbb{F}_2$ žodžių kodą.

102 teorema. *Teisinga lygybė*

$$\sum_{\mathbf{u}\in\mathbf{L}}(-1)^{(\mathbf{u},\mathbf{v})}=\begin{cases}|\mathbf{L}|, & \text{jei } \mathbf{v}\in\mathbf{L}^{\perp},\\ 0, & \text{jei } \mathbf{v}\notin\mathbf{L}^{\perp}.\end{cases}$$

Įrodymas. Jeigu $\mathbf{v}\in\mathbf{L}^{\perp}$, tai $(\mathbf{u},\mathbf{u})=0$ su visais $\mathbf{u}\in\mathbf{L}$ ir pirmoji lygybė akivaizdi.

Tegu $\mathbf{v}\notin\mathbf{L}^{\perp}$, tada visus $\mathbf{L}$ žodžius galime padalyti į dvi grupes: $U=\{\mathbf{u}_1,\mathbf{u}_2,\dots,\mathbf{u}_r\}$ ir $W=\{\mathbf{w}_1,\mathbf{w}_2,\dots,\mathbf{w}_s\}$, kad

$$(\mathbf{u}_i,\mathbf{v})=0,\quad (\mathbf{w}_j,\mathbf{v})=1.$$

Reikia įrodyti, kad $r=s$. Pastebėkime, kad

$$\{\mathbf{w}_1+\mathbf{w}_1,\mathbf{w}_1+\mathbf{w}_2,\dots,\mathbf{w}_1+\mathbf{w}_s\}\subset U,\quad \{\mathbf{w}_1+\mathbf{u}_1,\mathbf{w}_1+\mathbf{u}_2,\dots,\mathbf{w}_1+\mathbf{u}_r\}\subset W.$$

Tada $s\le r$ ir $r\le s$, taigi $r=s$.

Prisiminkime, kas yra Abelio grupės: tai algebrinės grupės su sudėties operacija, turinčia savybę $a+b=b+a$.

103 teorema. *Tegu G yra bet kokia Abelio grupė, $f:\mathbb{F}_2^n\to G$,*

$$\widehat{f}(\mathbf{u})=\sum_{\mathbf{v}\in\mathbb{F}_2^n}(-1)^{(\mathbf{u},\mathbf{v})}f(\mathbf{v}).$$

Tada bet kokiam tiesiniam kodui $\mathbf{L}\subset\mathbb{F}_2^n$ teisinga lygybė

$$\sum_{\mathbf{u}\in\mathbf{L}^{\top}}f(\mathbf{u})=\frac{1}{|\mathbf{L}|}\sum_{\mathbf{u}\in\mathbf{L}}\widehat{f}(\mathbf{u}).$$

Įrodymas. Pakeisdami sumavimo tvarką ir pasiremdami anksčiau įrodytu teiginiu, gausime

$$\sum_{\mathbf{u}\in\mathbf{L}}\widehat{f}(\mathbf{u})=\sum_{\mathbf{v}\in\mathbb{F}_2^n}f(\mathbf{v})\sum_{\mathbf{u}\in\mathbf{L}}(-1)^{(\mathbf{u},\mathbf{v})}=$$
$$\sum_{\mathbf{v}\in\mathbf{L}^{\perp}}f(\mathbf{v})\sum_{\mathbf{u}\in\mathbf{L}}(-1)^{(\mathbf{u},\mathbf{v})}+\sum_{\mathbf{v}\notin\mathbf{L}^{\perp}}f(\mathbf{v})\sum_{\mathbf{u}\in\mathbf{L}}(-1)^{(\mathbf{u},\mathbf{v})}=|\mathbf{L}|\sum_{\mathbf{v}\in\mathbf{L}^{\perp}}f(\mathbf{v}).$$

O dabar jau pasirengta įrodyti (69) lygybę.

Įrodymas. Kiekvienam žodžiui $\mathbf{u}\in\mathbb{F}_2^n$ apibrėžkime

$$f(\mathbf{u})=x^{n-w(\mathbf{u})}y^{w(\mathbf{u})}$$

ir pastebėkime, kad bet kokiam kodui $\mathbf{L}$

$$\sum_{\mathbf{u}\in\mathbf{L}} f(\mathbf{u}) = w_{\mathbf{L}}(x,y).$$

Funkcija f įgyja reikšmes daugianarių su dviem simboliais x, y aibėje. Svarbu, kad tokie daugianariai sudaro Abelio grupę, todėl funkcijai galėsime taikyti ką tik įrodytą teiginį. Pasinaudoję juo, gausime

$$w_{\mathbf{L}^\perp}(x,y) = \sum_{\mathbf{u}\in\mathbf{L}^\perp} f(\mathbf{u}) = \frac{1}{|\mathbf{L}|}\sum_{\mathbf{u}\in\mathbf{L}} \widehat{f}(\mathbf{u}). \tag{70}$$

Surasime reikšmes $\widehat{f}(\mathbf{u})$. Tegu $\mathbf{u} = u_1u_2\ldots u_n, \mathbf{v} = v_1v_2\ldots v_n$. Tada

$$\begin{aligned}\widehat{f}(\mathbf{u}) &= \sum_{\mathbf{v}\in\mathbb{F}_2^n} (-1)^{(\mathbf{u},\mathbf{v})} x^{n-w(\mathbf{u})} y^{w(\mathbf{u})} = \sum_{\mathbf{v}\in\mathbb{F}_2^n}\prod_{i=1}^{n} (-1)^{u_iv_i} x^{1-v_i} y^{v_i} \\ &= \prod_{i=1}^{n}\Big(\sum_{w=0}^{1} (-1)^{u_iw} x^{1-w} y^{w}\Big).\end{aligned}$$

Dabar pastebėkime, kad vidinė suma lygi:

$$\sum_{w=0}^{1} (-1)^{u_iw} x^{1-w} y^{w} = \begin{cases} x+y, & \text{jei } u_i = 0, \\ x-y, & \text{jei } u_i = 1. \end{cases}$$

Taigi

$$\widehat{f}(\mathbf{u}) = (x+y)^{n-w(\mathbf{u})}(x-y)^{w(\mathbf{u})}.$$

Įstatę šią reikšmę į (70), gausime McWilliams tapatybę (69).

10 Cikliniai kodai

Matricos, daugianariai – tai svarbiausi tiesinės algebros veikėjai. Tačiau, nagrinėdami tiesinius kodus, naudojomės beveik tik vien matricomis. Dabar pasitelkime daugianarių techniką...

10.1. Daugianarių kodai

Kodavimas – tai daugianarių daugyba, dekodavimas – dalyba. Taip trumpai galima apibūdinti daugianarių kodų naudojimo ypatybes.

Kaip ir anksčiau, $\mathbb{F}_q$ žymėsime kūną iš $q = p^m$ elementų, čia p – pirminis skaičius (dažnai imsime tiesiog $q = p$).

Jau nagrinėjome daugianarių tiesines erdves

$$\begin{aligned}\mathbb{F}_q[x] &= \{a_0 + a_1x + \ldots + a_mx^m : a_i \in \mathbb{F}_q, a_m \neq 0\}, \\ \mathbb{F}_{q,n}[x] &= \{f \in \mathbb{F}_q[x] : \deg(f) < n\}.\end{aligned}$$

Žinome, kad tiesinės erdvės $\mathbb{F}_q^n$ ir $\mathbb{F}_{q,n}[x]$ yra izomorfiškos. Elementų atitiktį nustato labai paprasta taisyklė

$$\mathbf{a} \in \mathbb{F}_q^n,\ \mathbf{a} = a_0 a_1 \dots a_{n-1} \mapsto a_0 + a_1 x + \dots + a_{n-1} x^{n-1} \in \mathbb{F}_{q,n}[x].$$

80 apibrėžimas. *Daugianario $g(x) \in \mathbb{F}_q[x]$ svoriu vadinsime jo nenulinių koeficientų skaičių. Svorį žymėsime $w(g)$.*

Jeigu apsiribojame tik sudėties bei daugybos iš kūno elementų veiksmais, tai visai tas pats, ar juos taikome žodžiams, ar daugianariams. Tačiau daugianarius dar galime ir dauginti!

81 apibrėžimas. *Tegu $g(x) = g_0 + g_1 x + \dots + g_k x^k$ yra daugianaris iš $\mathbb{F}_q[x]$, $0 \le k \le n$, $g_k \neq 0$. Aibę*

$$\mathbf{C}_{g,n} = \{a(x)g(x) : a(x) \in \mathbb{F}_{q,n-k}[x]\} \subset \mathbb{F}_{q,n}[x]$$

vadinsime daugianarių kodu, generuotu $g(x)$.

Trumpai tariant, daugianario $g(x)$ generuotas kodas – šio daugianario kartotinių, kurių laipsnis ne didesnis kaip n, aibė. Žinoma, dydį n galime pasirinkti, taigi su tuo pačiu daugianariu galime generuoti įvairius kodus.

104 teorema. *Daugianarių kodas $\mathbf{C}_{g,n}$, kurį generuoja k-ojo laipsnio daugianaris g, yra tiesinis kodas. Daugianariai*

$$g(x),\ xg(x),\ \dots,\ x^{n-k-1}g(x) \tag{71}$$

sudaro šio kodo bazę.

Įrodymas. Iš tiesų, kad kodas yra tiesinis, beveik akivaizdu. Akivaizdu ir tai, kad bet kuris kodo žodis yra (71) žodžių tiesinė kombinacija. Jeigu šio rinkinio daugianariai būtų tiesiškai priklausomi, tai egzistuotų nenulinis daugianaris $c \in \mathbb{F}_q[x]$, kad

$$c(x)g(x) = 0.$$

Tačiau tokia lygybė su daugianarių žiedo $\mathbb{F}_q[x]$ elementais negalima (matematikai sako – žiede nėra nulio daliklių). Taigi daugianariai (71) tikrai sudaro kodo bazę.

Išvada. *Daugianario g, $\deg(g) = k$, generuoto kodo $\mathbf{C}_{g,n}$ parametrai yra $[n, n-k]$.*

Interpretuodami kodo $\mathbf{C}_{g,n}$ elementus kaip žodžius, sudarytus iš daugianarių koeficientų, galime pagal (71) bazę sudaryti generuojančią kodo matricą:

$$G = \begin{pmatrix} g_0 & g_1 & \dots & g_k & 0 & 0 & \dots & 0 \\ 0 & g_0 & g_1 & \dots & g_k & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & g_0 & \dots & g_k \end{pmatrix}.$$

Prisiminkime, kokios patogios koduoti yra standartinio pavidalo generuojančios matricos: kodavimas yra sisteminis, t. y. prie koduojamo informacijos bloko tiesiog prišliejami papildomi klaidoms taisyti reikalingi kontroliniai simboliai. Jeigu kodavimui daugianarių kodu naudosime ką tik užrašytą generuojančią matricą, tai kodavimas nebus sisteminis. Tačiau galima sudaryti kitą generuojančią matricą ir naudotis sisteminio kodavimo privalumais.

105 teorema. *Tegu g yra k-ojo laipsnio daugianaris, o $r_j(x)$ yra daugianario x^j dalybos iš $g(x)$ liekana. Tada daugianariai*

$$x^{k+1} - r_{k+1}(x),\ x^{k+2} - r_{k+2}(x), \ldots, x^n - r_n(x)$$

sudaro $g(x)$ generuoto kodo $\mathbf{C}_{g,n}$ bazę.

Įrodymas. Iš tikrųjų, bet kuris daugianaris dalijasi iš $g(x)$, todėl priklauso kodui. Jų yra tiek, kokia kodo dimensija. Jeigu jų koeficientus surašytume į $(n-k)\times n$ matricą G', pamatytume, kad paskutinieji $n-k$ stulpeliai sudaro vienetinę matricą:

$$G' = (A, I_{n-k}).$$

Taigi matricos eilutės yra tiesiškai nepriklausomos, jos sudaro kodo bazę. Tačiau koduojant $n-k$ ilgio simbolių bloką su šia matrica, jis tiesiog papildomas iš priekio k simboliais. Kodavimas su šia matrica yra sisteminis, tik kontroliniai simboliai pridedami ne koduojamo kodo pabaigoje, bet priekyje.

O dabar prisiminkime, kaip sudarėme generuojančią Golay kodo $\mathbf{G}_{23}$ matricą: į pirmąją eilutę surašėme žodžio

$$\mathbf{c}_1 = 11000111010100000000000$$

simbolius, o į kitas – žodžius, gautus iš $\mathbf{c}_0$ ciklišk ais postūmiais. Tokią pačią generuojančią matricą turi daugianario

$$g(x) = 1 + x + x^5 + x^6 + x^7 + x^9 + x^{11}$$

generuotas 23 bitų ilgio žodžių kodas. Taigi Golay kodas $\mathbf{G}_{23}$ yra daugianarių kodas. Daugianarių kodai yra tiesiniai kodai, turintys savų ypatybių. O gal bet kuris tiesinis kodas yra kokio nors daugianario generuotas daugianarių kodas? Atsakymas – ne. Štai paprastas pavyzdys. Nagrinėkime tiesinį kodą $\mathbf{L} = \{000, 010, 101, 111\}$. Pakeitę žodžius daugianariais, gautume $\mathbf{L} = \{0, x, 1+x^2, 1+x+x^2\} \subset \mathbb{F}_{2,3}[x]$. Yra tik vienas daugianaris, kuris dalija visus $\mathbf{L}$ daugianarius: $g(x) = 1$. Tačiau $\mathbf{C}_{g,3} \neq \mathbf{L}$.

Kodavimas daugianario kodo žodžiais yra daugianarių daugyba. Iš tiesų:

$$a_0a_1\ldots a_{n-k-1} \mapsto a_0 + a_1x + \ldots + a_{n-k-1}x^{n-k-1} = a(x) \mapsto a(x)g(x) \in \mathbf{C}_{g,n}.$$

O dekodavimas – galbūt daugianarių dalyba? Beveik.

Kaip ir nagrinėtų tiesinių kodų atveju, kiekvienam daugianariui $a(x) \in \mathbb{F}_{q,n}[x]$ galime sudaryti jo „sluoksnį“:

$$\mathbf{L}_a = \{a(x) + c(x) : c(x) \in \mathbf{C}_{g,n}\}.$$

Kiekvienas sluoksnis turi po q^{n-k} elementų. Skirtingus daugianarius $a(x), b(x)$ atitinkantys sluoksniai arba sutampa, arba nesikerta. Taigi visą daugianarių erdvę $\mathbb{F}_{q,n}[x]$ galima išskaidyti į nesikertančių sluoksnių

$$\mathbf{L}_1,\ \mathbf{L}_2,\ \ldots,\ \mathbf{L}_{q^k},\ \mathbf{L}_1 = \mathbf{C}_{g,n},$$

sąjungą. Vieno sluoksnio atstovų dalybos iš $g(x)$ liekanos sutampa. Vadinasi, pagal gautojo iš kanalo iškraipyto žodžio

$$d(x) = c(x) + e(x),\ c(x) \in \mathbf{C}_{g,n},$$

dalybos iš $g(x)$ liekaną galime nustatyti, į kurią klasę žodis pateko. Ši dalybos liekana daugianario kodo atveju atlieka sindromo vaidmenį. Suradę šios klasės lyderį $e(x)$ – mažiausią svorį turintį jos elementą – gautąjį žodį dekoduojame taip:

$$d(x) \mapsto d(x) - e(x).$$

Ištaisomų klaidų skaičius priklauso nuo minimalaus kodo atstumo. Gerai žinome, kad konstruoti daug klaidų taisančius kodus yra nelengva. Ir daugianarių technika čia ne kiek tepadeda. Panagrinėkime tik pavyzdį su dvejetainės abėcėlės kodais.

106 teorema. *Jei daugianaris $g(x) \in \mathbb{F}_2[x]$ su nenuliniu laisvuoju nariu nedalija jokio daugianario $x^k + 1$, kur $k < n$, tai $g(x)$ generuoto kodo iš n ilgio žodžių minimalus atstumas ne mažesnis už 3.*

Įrodymas. Visi $g(x)$ generuoto kodo daugianariai dalijasi iš $g(x)$. Reikia įrodyti, kad visų tokių daugianarių svoriai yra ne mažesni už 3. Kadangi $g(x)$ nėra konstanta (kitaip $g(x)$ dalytų daugianarius $x^k + 1$), tai reikia įrodyti, kad jokie daugianariai

$$x^i, \quad x^i + x^{i+j},\ i + j < n,$$

nesidalija iš $g(x)$. Kadangi $g(x)$ laisvasis narys nelygus nuliui, tai $g(x) \not| x^i$. Jeigu $g(x)$ dalytų $x^i + x^{i+j} = x^i(1 + x^j)$, tai turėtų dalyti ir $1 + x^j,\ j < n$. Tačiau tai prieštarauja teoremos sąlygai.

Pavyzdys. Pasirėmę akivaizdžiomis lygybėmis (jos teisingos tik kūne $\mathbb{F}_2$!):

$$x^4 + 1 = (x + 1)^4, \quad x^5 + 1 = (x + 1)(x^4 + x^3 + x^2 + x + 1),$$

matome, kad $g(x) = 1 + x + x^3$ nedalija nei $x^4 + 1$, nei $x^5 + 1$. Taigi $g(x)$ generuoja $[6, 3]$ kodą, visada ištaisantį vieną klaidą.

10.2. Daugianarių žiedai ir idealai

Visų žodžių kilmės istorija – žmonių siekių, vargų ir džiaugsmų istorija. Jeigu nuspręstumėte sužinoti, kaip algebroje atsirado sąvoka „idealas“, tektų patyrinėti, kaip matematikai galynėjosi su Paskutiniąja Fermat teorema.

Jeigu $f \in \mathbb{F}_q[x]$ yra n-ojo laipsnio daugianaris, tai daugianarių erdvė $\mathbb{F}_{q,n}[x]$ su sudėties ir daugybos operacijomis $+, \times_f$ sudaro žiedą. Šį žiedą žymėsime $\mathbb{F}_q[x]/f$.

Jame galime ieškoti mažesnių žiedų, kitaip sakant, požiedžių.

82 apibrėžimas. *Žiedo R su sudėties ir daugybos operacijomis $+, \cdot$ netuščią poaibį $R' \subset R$ vadiname požiedžiu, jeigu su visais $x, y \in R'$ teisingi sąryšiai $x + y, x \cdot y \in R'$.*

Ieškosime ne bet kokių, bet „gerų“ žiedo $\mathbb{F}_q[x]/f$ požiedžių.

83 apibrėžimas. *Žiedo $\mathbb{F}_q[x]/f$ požiedį I vadinsime idealu, jeigu kiekvienam $b(x) \in I$ teisingas sąryšis $x \times_f b(x) \in I$.*

Iš apibrėžimo beveik iš karto gauname tokią išvadą:

Išvada. *Jeigu I yra žiedo $\mathbb{F}_q[x]/f$ idealas, tai su visais $a(x) \in \mathbb{F}_q[x]/f$, $b(x) \in I$, teisingas sąryšis $a(x) \times_f b(x) \in I$.*

Kitaip sakant – jeigu daugianaris priklauso idealui, tai ir visi jo „kartotiniai“ sandaugos $\times_f$ prasme yra šio idealo elementai.

Ši išvada parodo, kaip idealai gali būti sudaryti.

84 apibrėžimas. *Tegu $g(x) \in \mathbb{F}_q[x]/f, \deg(g) < n$. Idealą*

$$\langle g \rangle = \{a(x) \times_f g(x) : a(x) \in \mathbb{F}_q[x]/f\}$$

vadinsime pagrindiniu idealu. Daugianarį $g(x)$ vadinsime šio idealo generatoriumi.

O dabar naujiena apie lygiavą daugianarių žieduose: visi idealai pagrindiniai!

107 teorema. *Jeigu I yra žiedo $\mathbb{F}_q[x]/f$ idealas, tai $I = \langle g \rangle$, čia g yra mažiausio laipsnio nenulinis daugianaris, priklausantis I.*

Įrodymas. Tarkime, $g(x) \in I$ yra nenulinis mažiausio laipsnio daugianaris iš idealo I. Tada su visais $a(x) \in \mathbb{F}_q[x]/f$ teisingas sąryšis $a(x) \times_f g(x) \in I$. Reikia įrodyti, kad kitokių daugianarių ideale I nėra.

Tegu $b(x) \in I$ yra bet koks idealo daugianaris. Padalykime jį iš $g(x)$ su liekana

$$b(x) = q(x)g(x) + r(x), \quad 0 \leq \deg(r) < \deg(g).$$

Kadangi visų trijų šios lygybės daugianarių laipsniai yra mažesni už n, tai niekas nepasikeis, jeigu daugybos veiksmą pakeisime žiedo daugybos veiksmu:

$$b(x) = q(x) \times_f g(x) + r(x), \quad 0 \leq \deg(r) < \deg(g).$$

Kadangi I yra požiedis, tai $r(x) = b(x) - q(x) \times_f g(x) \in I$. Tačiau tada $r(x) = 0$, nes kitaip $g(x)$ nebūtų mažiausio laipsnio idealo daugianaris. Todėl $b(x) = q(x) \times_f g(x)$.

Taigi bet kuriam žiedo $\mathbb{F}_q[x]/f$ idealui galime surasti jį generuojantį daugianarį. Apskritai tokių daugianarių yra ne vienas. Pavyzdžiui, galima įrodyti, kad jei daugianaris $h(x)$ neturi su $f(x)$ netrivialių bendrųjų daliklių (t. y. jie kartu dalijasi tik iš nulinio laipsnio daugianarių – konstantų), tai $\langle g \rangle = \langle g \times_f h \rangle$. Tačiau mums svarbiausi – mažiausio laipsnio daugianariai generuojantys idealą. Kiek jų yra? Jeigu

$$g(x) = g_0 + g_1 x + \ldots + g_k x^k$$

yra vienas iš jų, tai daugianariai $\alpha g(x)$, $\alpha \in \mathbb{F}_q, \alpha \neq 0$, irgi yra to paties laipsnio ir generuoja tą patį idealą. Dažniausiai patogu pasirinkti tą daugianarį iš šio būrio, kurio vyriausiasis koeficientas lygus 1.

10.3. Daugiau nei paprasti daugianarių kodai

Cikliniai kodai – tai tikras tiesinių kodų elitas. Apibrėžti juos galime labai paprastai, tačiau ryšių jie turi su aukštais matematikos sluoksniais.

Iš pirmo žvilgsnio atlikti veiksmą $a(x) \times_f b(x)$ – nemažas vargas: reikia pirmiausia daugianarius sudauginti įprastu būdu, tada – rasti sandaugos dalybos iš $f(x)$ liekaną. Tačiau kartais tą darbą galima atlikti paprastai ir greitai.

Šiame skyrelyje nagrinėsime tik daugianarių žiedą $\mathbb{F}_q[x]/f$ su

$$f(x) = x^n - 1.$$

Jeigu

$$a(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1} \in \mathbb{F}_q[x]/f,$$

tai

$$\begin{aligned}
&x \times_f a(x) = a_{n-1} + a_0 x + \ldots + a_{n-2} x^{n-1}, \\
&x^2 \times_f a(x) = a_{n-2} + a_{n-1} x + \ldots + a_{n-3} x^{n-1}, \\
&\ldots\ldots\ldots\ldots\ldots\ldots \\
&x^m \times_f a(x) = a_{n-m} + a_{n-1} x + \ldots + a_{n-m-1} x^{n-1},
\end{aligned}$$

taigi dauginant iš vienanarių, pakanka „pastumti“ daugianario koeficientus. Savo ruožtu, kai daugikliai yra bet kokie daugianariai, pakanka atlikti seką postūmio, daugybos iš kūno elementų ir sudėties veiksmų.

85 apibrėžimas. *Tiesinį kodą $\mathbf{C} \subset \mathbb{F}_q^n$ vadinsime cikliniu, jeigu*

$$\textit{kiekvienam } \mathbf{c} = c_0 c_1 \ldots c_{n-1} \in \mathbf{C} \quad c_{n-1} c_0 \ldots c_{n-2} \in \mathbf{C}.$$

Atlikę ciklinio kodo žodžių simbolių postūmį, vėl gauname to paties kodo žodį. Akivaizdu, kad, ciklišKai pastūmę simbolius per bet kiek pozicijų, vėl gausime kodo žodžius.

O dabar interpretuokime kiekvieną ciklinio kodo žodį $\mathbf{c} = c_0c_1 \dots c_{n-1}$ kaip žiedo elementą

$$c(x) = c_0 + c_1x + \ldots + c_{n-1}x^{n-1}.$$

Kodas $\mathbf{C}$ yra $\mathbb{F}_q[x]/f$ poaibis. Kadangi $\mathbf{C}$ yra ciklinis kodas, tai

$$x \times_f c(x) = c_{n-1} + c_0x + c_1x^2 + \ldots + c_{n-2}x^{n-1} \in \mathbf{C}.$$

Nebesudėtinga padaryti išvadą, kad $\mathbf{C}$ yra žiedo $\mathbb{F}_q[x]/f$ idealas! Teisingas ir atvirkštinis teiginys: kiekvienas žiedo $\mathbb{F}_q[x]/f$ idealas yra ciklinis kodas.

108 teorema. *Žiedo $\mathbb{F}_q[x]/f$ idealų ir erdvės $\mathbb{F}_{q,n}[x]$ ciklinių kodų aibės sutampa.*

Dabar galime cikliniams kodams tirti panaudoti visa, ką žinome apie idealus.

109 teorema. *Ciklinis kodas $\mathbf{C} \subset \mathbb{F}_q[x]/f$ yra daugianarių kodas, kurį generuoja mažiausio laipsnio nenulinis $\mathbf{C}$ daugianaris $g(x)$, t. y. $\mathbf{C} = \mathbf{C}_{g,n}$. Šis daugianaris yra $f(x) = x^n - 1$ daliklis. Jei daugianaris $h(x)$ dalija $f(x)$, tai jo generuotas daugianarių kodas $\mathbf{C}_{h,n} \subset \mathbb{F}_q[x]/f$ yra ciklinis.*

Primename, kad daugianarių kodo generatorius – tai daugianaris $g(x)$, su kuriuo sudaroma kodo bazė

$$g(x), xg(x), \ldots, x^{n-k-1}g(x).$$

Taigi visi daugianarių kodo žodžiai dalijasi iš $g(x)$. Ciklinis kodas yra idealas; idealas turi ne vieną generatorių; ne visus juos, bet tik mažiausio laipsnio daugianarius vadiname kodo generatoriais!

Įrodymas. Tegu daugianario $g(x)$ laipsnis lygus k. Tada

$$\begin{aligned} \mathbf{C}_{g,n} &= \{a(x)g(x) : \deg(a) < n-k\}, \\ \mathbf{C} &= \langle g \rangle = \{a(x) \times_f g(x) : a(x) \in \mathbb{F}_{q,n}[x]\}. \end{aligned}$$

Akivaizdu, kad $\mathbf{C}_{g,n} \subset \mathbf{C}$. Tačiau, įrodinėdami 108 teoremą, nustatėme, kad kiekvienas idealo $\langle g \rangle$ žodis $b(x)$ dalijasi iš $g(x)$. Taigi $\mathbf{C} \subset \mathbf{C}_{g,n}$ ir pirmasis teoremos teiginys įrodytas.

Įrodysime, kad $f(x) = x^n - 1$ dalijasi iš $g(x)$. Padalykime $f(x)$ iš $g(x)$ su liekana:

$$f(x) = h(x)g(x) + r(x), \quad (-h(x))g(x) = r(x) - f(x),\ 0 \le \deg(r) < \deg(g).$$

Iš paskutinės lygybės gauname, kad $(-h(x)) \times_f g(x) = r(x)$, taigi $r(x) \in \langle g \rangle$. Tačiau tada turi būti $r(x) = 0$, nes g yra mažiausio laipsnio nenulinis daugianaris, priklausantis $\mathbf{C}$. Taigi

$$x^n - 1 = h(x)g(x)$$

ir antrasis teiginys yra įrodytas.

Tegu dabar $h(x)$ dalija $x^n - 1$. Kodas $\langle h \rangle$ yra ciklinis; kad įrodytume, jog $h(x)$ yra jo generatorius, pakanka įsitikinti, kad joks mažesnio už h laipsnį nenulinis daugianaris nepriklauso $\langle h \rangle$. Tarkime priešingai: ciklinio kodo $\langle h \rangle$ generatorius yra daugianaris $h_1(x)$ ir $\deg(h_1) < \deg(h)$. Tačiau tada $h(x)$ turi dalytis iš $h_1(x)$. Kita vertus, turi atsirasti daugianaris $d(x)$, kad būtų $h_1(x) = d(x) \times_f h(x)$, t. y.

$$d(x)h(x) = k(x)f(x) + h_1(x).$$

Tačiau $f(x)$ dalijasi iš $h(x)$, todėl iš šios lygybės gautume, kad $h_1(x)$ turi dalytis iš $h(x)$. O tai jau prieštaravimas!

Tegu $g(x)$ yra ciklinio kodo generatorius, t. y. daugianaris, dalijantis $f(x) = x^n - 1$. Kadangi ciklinis kodas yra daugianarių kodas, tai gautojo iš kanalo (tikriausiai iškraipyto) žodžio $d(x)$ sindromu galime laikyti dalybos iš $g(x)$ liekaną:

$$d(x) = k(x)g(x) + r(x), \quad \deg(r) < \deg(g). \tag{72}$$

Tegu $x^n - 1 = g(x)h(x)$. Padauginę (72) lygybės puses iš $h(x)$, gausime:

$$\begin{aligned} &d(x)h(x) = k(x)g(x)h(x) + r(x)h(x) = k(x)f(x) + r(x)h(x), \\ &d(x) \times_f h(x) = r(x)h(x). \end{aligned}$$

Jei dviejų žodžių $d_1(x), d_2(x)$, gautų iš kanalo, dalybos iš $g(x)$ liekanos $r_1(x), r_2(x)$ yra skirtingos, tai ir sandaugos $d_1(x) \times_f h(x), d_2(x) \times_f h(x)$ bus skirtingos. Taigi kai kodas ciklinis, tai žodžio $d(x)$ sindromo vaidmenį puikiai atlieka sandauga $d(x) \times_f h(x)$, o daugianariui $h(x)$ tinka kontrolinio daugianario vardas.

Pavyzdžiai. Daugianaris $x^9 - 1$ virš kūno $\mathbb{F}_2$ skaidomas neskaidžiais daugikliais tokiu būdu:

$$x^9 - 1 = (x - 1)(x^2 + x + 1)(x^6 + x^3 + 1).$$

Taigi iš $\mathbb{F}_2^9$ elementų galima sudaryti 8 ciklinius kodus. Pavyzdžiui, generuojantį daugianarį $g(x) = (x-1)(x^6+x^3+1)$ atitinka ciklinis kodas, kurio generuojanti matrica

$$G = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Daugianaris $x^{23}-1$ virš kūno $\mathbb{F}_2$ išskaidomas neskaidžiais daugikliais taip:

$$x^{23}-1=(x+1)(x^{11}+x^9+x^7+x^6+x^5+x+1)\times$$
$$\times(x^{11}+x^{10}+x^6+x^5+x^4+x^2+1).$$

Jau įsitikinome, kad daugianaris

$$g(x)=x^{11}+x^9+x^7+x^6+x^5+x+1$$

generuoja Golay kodą $\mathbf{G}_{23}$. Galima būtų įrodyti, kad ciklinis kodas, kurio generuojantis daugianaris yra

$$g(x)=x^{11}+x^{10}+x^6+x^5+x^4+x^2+1,$$

ekvivalentus $\mathbf{G}_{23}$. Pakaktų atitinkamai pertvarkyti šio ciklinio kodo generuojančią matricą.

Daugianario $x^{11}-1$ skaidinys virš kūno $\mathbb{F}_3$ yra toks:

$$x^{11}-1=(x-1)(x^5+x^4-x^3+x^2-1)(x^5-x^3+x^2-x-1).$$

Ciklinis kodas, kurį generuoja daugianaris

$$g(x)=x^5+x^4-x^3+x^2-1,$$

vadinamas trinarės abėcėlės Golay kodu $\mathbf{G}_{11}$. Šio kodo iš abėcėlės $\mathbb{F}_3$ žodžių parametrai tokie: $[11,6,5]$. Nesudėtinga įsitikinti, kad tai tobulas kodas.

10.4. Reedo-Solomono kodų peržiūra

Reedo-Solomono kodai yra maksimalaus atstumo kodai. Gerokai pagausinę šią šeimą, nustatysime, kad joje yra ir ciklinių kodų.

Prisiminkime, kaip apibrėžėme Reedo-Solomono kodus, naudodami kūną $\mathbb{F}_q$.

86 apibrėžimas. *Tegu $\mathbb{F}_q=\{\alpha_1,\alpha_2,\ldots,\alpha_q\}$ (apibrėžtumo dėlei tarkime, $\alpha_q=0$), o $1\le k\le q$ yra natūralusis skaičius. Tiesinį kodą, kurio generuojanti matrica yra*

$$G_k=\begin{pmatrix}1 & 1 & \ldots & 1\\ \alpha_1 & \alpha_2 & \ldots & \alpha_q\\ \alpha_1^2 & \alpha_2^2 & \ldots & \alpha_q^2\\ \vdots & \vdots & \ddots & \vdots\\ \alpha_1^{k-1} & \alpha_2^{k-1} & \ldots & \alpha_q^{k-1}\end{pmatrix},$$

vadiname Reedo-Solomono kodu ir žymime $\mathbf{RS}_{q,k}$.

Pabandykime pakeisti dvi šios konstrukcijos detales: matricai sudaryti panaudokime nebūtinai visus kūno elementus; matricos stulpelius padauginkime iš nenulinių elementų.

87 apibrėžimas. *Tegu* $\mathbf{v} = (v_1, v_2, \ldots, v_n)$ *yra žodis, sudarytas tik iš nenulinių* $\mathbb{F}_q$ *elementų, o* $\mathbf{a} = (\beta_1, \beta_2, \ldots, \beta_n)$ *– būtinai iš skirtingų šio kūno elementų,* $k \leq n$. *Kodą, kurio generuojanti matrica yra*

$$\begin{pmatrix} v_1 & v_2 & \ldots & v_n \\ v_1\beta_1 & v_2\beta_2 & \ldots & v_n\beta_n \\ v_1\beta_1^2 & v_2\beta_2^2 & \ldots & v_n\beta_n^2 \\ \ldots & \ldots & \ddots & \vdots \\ v_1\beta_1^{k-1} & v_2\beta_2^{k-1} & \ldots & v_n\beta_n^{k-1} \end{pmatrix},$$

vadinsime apibendrintuoju Reedo-Solomono kodu ir žymėsime $\mathbf{RS}_{n,k}(\mathbf{v}, \mathbf{a})$.

Akivaizdu, kad $\mathbf{RS}_{q,k} = \mathbf{RS}_{n,k}(\mathbf{v}, \mathbf{a})$ su $\mathbf{v} = (1, 1, \ldots, 1)$ ir $\mathbf{a} = (\alpha_1, \alpha_2, \ldots, \alpha_q)$. Malonu, kad apibendrintieji kodai išlaiko daugelį gerų Reedo-Solomono kodų savybių. Pavyzdžiui, jeigu $k_1 \leq k_2$, tai $\mathbf{RS}_{n,k_1}(\mathbf{v}, \mathbf{a}) \subset \mathbf{RS}_{n,k_2}(\mathbf{v}, \mathbf{a})$.

110 teorema. *Kodo* $\mathbf{RS}_{n,k}(\mathbf{v}, \mathbf{a})$ $(1 \leq k \leq n)$ *parametrai yra*

$$[n, k, n-k+1],$$

t. y. šis kodas yra maksimalaus atstumo kodas.

Taigi apibendrintieji kodai irgi yra maksimalaus atstumo kodai.

Įrodyti šį teiginį galima nustačius, kad bet kuris generuojančios matricos k stulpelių rinkinys sudaro tiesiškai nepriklausomą sistemą. Tam vėl kaip ir „paprastų“ Reedo-Solomono kodų atveju reikia panaudoti Vandermondo determinantą. Jeigu bet kuri k stulpelių sistema yra tiesiškai nepriklausoma, tai kodas yra maksimalaus atstumo kodas. Taigi teorema tikrai teisinga.

O dabar apibendrintų Reedo-Solomono kodų šeimoje paieškokime ypatingų kodų.

Pasirinkime skaičių n, kuris dalija $q-1$, ir kūne $\mathbb{F}_q$ pasirinkime elementą α, kurio eilė yra n. Tokį elementą visada galime rasti. Iš tiesų, jeigu γ yra primityvusis kūno elementas, tai galime imti $\alpha = \gamma^{\frac{q-1}{n}}$. Sudarykime rinkinį $\mathbf{a}$ taip:

$$\mathbf{a} = (1, \alpha, \alpha^2, \ldots, \alpha^{n-1}),$$

ir imkime

$$\mathbf{v} = \mathbf{a}_0 = (1^0, \alpha^0, (\alpha^2)^0, \ldots, (\alpha^{n-1})^0) = (1, 1, \ldots, 1).$$

Dabar galime sudaryti Reedo-Solomono kodą $\mathbf{RS}_{n,k}(\mathbf{a}_0, \mathbf{a})$. Ar jis kuo nors ypatingas?

111 teorema. *Jeigu $\alpha \in \mathbb{F}_q$ yra n-osios eilės kūno elementas,*

$$\mathbf{a} = (1, \alpha, \alpha^2, \ldots, \alpha^{n-1}), \quad \mathbf{a}_0 = (1, 1, \ldots, 1),$$

tai kodas $\mathbf{RS}_{n,k}(\mathbf{a}_0, \mathbf{a})$ yra ciklinis, o vienas iš jo generuojančių daugianarių yra

$$g(x) = (x - \alpha)(x - \alpha^2) \cdots (x - \alpha^{n-k}). \tag{73}$$

Įrodymas. Sudarykime kodo $\mathbf{RS}_{n,k}(\mathbf{a}_0, \mathbf{a})$ generuojančią matricą:

$$\begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \alpha & \alpha^2 & \ldots & \alpha^{n-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & \ldots & (\alpha^{n-1})^2 \\ 1 & \alpha^3 & (\alpha^2)^3 & \ldots & (\alpha^{n-1})^3 \\ \ldots & \ldots & \ldots & \ddots & \ldots \\ 1 & \alpha^{k-1} & (\alpha^2)^{k-1} & \ldots & (\alpha^{n-1})^{k-1} \end{pmatrix}.$$

Kad įrodytume, jog kodas ciklinis, pakanka įsitikinti, kad bet kurio bazės žodžio ciklinis postūmis taip pat yra kodo žodis.

Imkime, pavyzdžiui, žodį, sudarytą iš $m+1$-osios eilutės elementų:

$$\mathbf{a}_m = (1, \alpha^m, (\alpha^2)^m, \ldots, (\alpha^{n-1})^m).$$

Žodis $\alpha^m \cdot \mathbf{a}_m$ priklauso kodui $\mathbf{RS}_{n,k}(\mathbf{a}_0, \mathbf{a})$, tačiau

$$\alpha^m \cdot \mathbf{a}_m = (\alpha^m, (\alpha^2)^m, \ldots, (\alpha^n)^m) = (\alpha^m, (\alpha^2)^m, \ldots, 1).$$

Taigi $\alpha^m \cdot \mathbf{a}_m$ yra per vieną poziciją ciklišk ai pastumtas bazės žodis $\mathbf{a}_m$. Tada $\alpha^{2m} \cdot \mathbf{a}_m$ bus irgi kodo žodis, savo ruožtu jį galime gauti, ciklišk ai pastūmę $\mathbf{a}_m$ elementus per dvi pozicijas ir t. t. Taigi mūsų sukonstruotas kodas yra ciklinis.

Kadangi kodo dimensija yra k, tai generuojančio kodą daugianario laipsnis yra $n - k$. Daugianario (73) laipsnis irgi yra $n - k$. Pakanka parodyti, kad daugianaris, sudarytas imant koeficientus iš generuojančios matricos bet kurios eilutės, dalijasi iš $g(x)$. Imkime daugianarį, atitinkantį pirmąją eilutę:

$$f_1(x) = 1 + x + x^2 + \ldots + x^{n-1}.$$

Daugianaris $f_1(x)$ dalysis iš $g(x)$, jeigu visos šio daugianario šaknys bus ir $f_1(x)$ šaknys. Taigi reikia įsitikinti, kad $f_1(\alpha^j) = 0$, $j = 1, 2, \ldots, n-k$. Pavyzdžiui,

$$f_1(\alpha^j) = 1 + \alpha^j + (\alpha^j)^2 + \ldots + (\alpha^j)^{n-1} = 1 + \beta + \beta^2 + \ldots + \beta^{n-1}, \quad \beta = \alpha^j.$$

Elementui β irgi teisinga lygybė $\beta^n = 1$, todėl

$$1+\beta+\beta^2+\ldots+\beta^{n-1} = \beta^n+\beta+\beta^2+\ldots+\beta^{n-1} = \beta(1+\beta+\beta^2+\ldots+\beta^{n-1}).$$

Kadangi $\beta \neq 0$ ir $\beta \neq 1$, tai tokia lygybė teisinga tik tada, kai

$$1+\beta+\beta^2+\ldots+\beta^{n-1} = 0, \text{ t. y. } f_1(\alpha^j) = 0.$$

Atvejai su kitomis $g(x)$ šaknimis ir su kitomis generuojančios matricos eilutėmis nagrinėjami analogiškai. Pavyzdžiui, $m+1$-ąją eilutę atitinka daugianaris

$$f_{m+1}(x) = 1 + (\alpha^1)^m \cdot x + (\alpha^2)^m \cdot x^2 + \ldots + (\alpha^{n-1})^m \cdot x^{n-1},$$

todėl

$$f_{m+1}(\alpha^j) = 1+\beta+\beta^2+\ldots+\beta^{n-1}, \quad \beta = \alpha^{m+j}.$$

Teorema įrodyta.

Galima suformuluoti ir kiek bendresnę teoremą. Pavyzdžiui, jei pažymėsime

$$\mathbf{a}_m = (1^m, \alpha^m, (\alpha^2)^m, \ldots, (\alpha^{n-1})^m), \quad \mathbf{a} = (1, \alpha, \alpha^2, \ldots, \alpha^{n-1}),$$

tai $\mathbf{RS}_{n,k}(\mathbf{a}_m, \mathbf{a})$ kodas irgi bus ciklinis, jo generuojanti matrica bus

$$\begin{pmatrix} 1 & \alpha^m & (\alpha^m)^2 & \ldots & (\alpha^m)^{n-1} \\ 1 & \alpha^{m+1} & (\alpha^{m+1})^2 & \ldots & (\alpha^{m+1})^{n-1} \\ 1 & \alpha^{m+2} & (\alpha^{m+2})^2 & \ldots & (\alpha^{m+2})^{n-1} \\ \ldots & \ldots & \ldots & \ddots & \ldots \\ 1 & \alpha^{m+k-1} & (\alpha^{m+k-1})^2 & \ldots & (\alpha^{m+k-1})^{n-1} \end{pmatrix},$$

o jo generatorius – daugianaris

$$g(x) = (x-\alpha^{1-m})(x-\alpha^{2-m})\cdots(x-\alpha^{n-k-m}).$$

10.5. BCH kodai

Santrumpa BCH iš tikrųjų žymi ilgą pavadinimą: Bose, Ray-Chaudhuri ir Hocquenghemo kodas. Šį kodą pirmieji du autoriai sukonstravo 1960 metais, o trečiasis – 1959.

Tarkime, sukonstravome kokį nors tiesinį $[n,k,d]$ kodą $\mathbf{L} \subset \mathbb{F}_q^n$ $(q = p^m)$. Kadangi $\mathbb{F}_p^n \subset \mathbb{F}_q^n$, tai galime sudaryti sankirtą $\mathbf{L}^* = \mathbf{L} \cap \mathbb{F}_p^n$. Ką galima pasakyti apie aibę $\mathbf{L}^*$? Visų pirma, ji netuščia (nulinis žodis jai tikrai priklauso). Antra – $\mathbf{L}^*$ yra tiesinis poerdvis. Taigi $\mathbf{L}^*$ yra tiesinis kodas iš abėcėlės $\mathbb{F}_p$ žodžių. Kokie jo parametrai? Pasakyti galime ne kažin kiek: tai $[n,k^*,d^*]$ kodas, čia $k^* \leq k$, $d^* \geq d$.

Taigi sukūrėme dar vieną būdą naujiems kodams iš turimų konstruoti. Iš karto jį ir išbandykime.

88 apibrėžimas. *Tegu n yra skaičiaus $q-1$ ($q=p^m$) daliklis, $\alpha \in \mathbb{F}_q$ yra n-osios eilės elementas, $1 \le k \le n$, $r = n-k+1$,*

$$\mathbf{a}_0 = (1, 1, \ldots, 1), \quad \mathbf{a}_1 = (1, \alpha, \alpha^2, \ldots, \alpha^{n-1}).$$

Kodą $\mathbf{L} = \mathbb{F}_p^n \cap \mathbf{RS}_{n,k}(\mathbf{a}_0, \mathbf{a}_1)$ vadinsime BCH kodu su numatytuoju minimaliuoju atstumu r.

Taigi BCH kodas – tai tarsi Reedo-Solomono kodo atstovybė žodžių aibėje $\mathbb{F}_p^n$. Jo parametrai yra $[n, k^*, d^*]$, čia $k^* \le k$, o $d^* \ge r$. Paskutinė nelygybė teisinga todėl, kad paties Reedo-Solomono kodo minimalus atstumas tiksliai lygus r (kodas priklauso maksimalaus atstumo kodų šeimai).

Paieškokime būdo, kaip apibūdinti BCH kodą, nesinaudojant Reedo-Solomono kodu. Juk, konstruodami šį kodą praktiškai, vargu ar norėsime surašyti visus Reedo-Solomono kodo žodžius ir juos lyg bulves pavasarį, perrinkinėti.

Tarkime, $\mathbf{L} \subset \mathbb{F}_p^n$ yra BCH kodas, gautas iš kodo $\mathbf{RS}_{n,k}(\mathbf{a}_0, \mathbf{a}_1) \subset \mathbb{F}_q^n$. Tegu α yra Reedo-Solomono kodui konstruoti panaudotas n-osios eilės elementas. Tada Reedo-Solomono kodo generatorius yra

$$g(x) = (x-\alpha)(x-\alpha^2)\ldots(x-\alpha^{r-1}), \quad r = n-k+1.$$

Visi Reedo-Solomono kodo žodžiai, interpretuojami kaip daugianariai, dalijasi iš $g(x)$, kitaip tariant – elementai $\alpha, \alpha^2, \ldots, \alpha^{r-1}$ yra jų šaknys. Kodo $\mathbf{L}$ žodžius irgi galime interpretuoti kaip daugianarius, tačiau su koeficientais iš $\mathbb{F}_p$. Taigi tie patys elementai yra ir jų šaknys. Tačiau jeigu elementas $\beta \in \mathbb{F}_q$ yra daugianario $f(x) \in \mathbb{F}_p[x]$ šaknis, tai $f(x)$ turi dalytis iš elemento β minimalaus daugianario. Todėl bet kuris BCH kodo žodis, suvokiamas kaip daugianaris su koeficientais iš $\mathbb{F}_p$, dalijasi iš minimaliųjų daugianarių

$$m_\alpha(x),\ m_{\alpha^2}(x),\ \ldots,\ m_{\alpha^{r-1}}(x). \tag{74}$$

Iš jų galime sudaryti vieną daugianarį – mažiausią bendrą kartotinį, t. y. mažiausio laipsnio daugianarį $g(x)$, kuris dalijasi iš visų (74) daugianarių. Iš $g(x)$ dalijasi ir visi BCH kodo žodžiai, ir tik jie. Be to, daugianaris x^n-1 dalijasi iš visų (74) daugianarių, todėl ir iš $g(x)$. Gavome naują BCH kodo charakteristiką.

112 teorema. *BCH kodas $\mathbf{L} \subset \mathbb{F}_p^n$ su numatytuoju atstumu r yra ciklinis kodas, kurio generatorius yra mažiausias bendras minimaliųjų daugianarių*

$$m_\beta(x),\ m_{\beta^2}(x),\ \ldots,\ m_{\beta^{r-1}}(x)$$

kartotinis; čia β yra n-osios eilės elementas, kurį galime rasti plėtinyje $\mathbb{F}_{p^k}$, tokiame, kad p^k-1 dalijasi iš n.

Šią teoremą galime naudoti konstruodami BCH kodus: juk tereikia surasti n-osios eilės elementą ir kelis minimaliuosius daugianarius. Pabandykime.

Sukonstruokime, pavyzdžiui, BCH kodą $\mathbf{L} \subset \mathbb{F}_3^{13}$ su numatytuoju atstumu $r = 3$. Taigi $n = 13$; kokiame kūne rasime n-osios eilės elementą? Kadangi iš 13 dalijasi $q-1 = 3^3-1$, tai reikės plėtinio $\mathbb{F}_{3^3}$. Geriausia jo konstrukcijai panaudoti primityvųjį trečiosios eilės daugianarį. Surasti jį nebūtų labai paprasta, tačiau šiuolaikinėms kompiuterinės algebros sistemoms toks uždavinys – vienas juokas. Štai jis:

$$f(x) = x^3 - x + 1.$$

Dabar galime laikyti α yra primityviuoju elementu ir skaičiuoti naudodamiesi sąryšiu $\alpha^3 = \alpha - 1$. Mums reikalingas n-osios eilės elementas bus tiesiog $\beta = \alpha^2$, o elementai, kurių minimaliuosius daugianarius teks skaičiuoti, bus šie:

$$\beta_1 = \beta = \alpha^2, \quad \beta_2 = \beta^2 = \alpha^4 = \alpha^2 - \alpha.$$

Prisiminti, kaip skaičiuojami minimalieji daugianariai. Pradėkime nuo $m_{\beta_1}(x)$. Šio elemento eilė yra 13, todėl reikia surasti mažiausiąjį m, kad $3^m \equiv 1 \pmod{13}$. Aišku, $m = 3$. Jau galime užrašyti pirmojo elemento minimalųjį daugianarį:

$$m_{\beta_1}(x) = (x-\beta_1)(x-\beta_1^{3^1})(x-\beta_1^{3^2}) = (x-\alpha^2)(x-\alpha^6)(x-\alpha^{18}).$$

Elemento β_2 eilė irgi tokia pati, todėl

$$m_{\beta_2}(x) = (x-\beta_2)(x-\beta_2^{3^1})(x-\beta_2^{3^2}) = (x-\alpha^4)(x-\alpha^{12})(x-\alpha^{36}),\ \alpha^{36} = \alpha^{10}.$$

Matome, kad bendrų daugiklių daugianariai neturi, taigi BCH kodo generuojantis daugianaris bus

$$g(x) = m_{\beta_1}(x) m_{\beta_2}(x).$$

Belieka išskleisti minimaliuosius daugianarius ir juos sudauginti. Kiek pasidarbavę (arba pasikvietę į pagalbą kompiuterinės algebros sistemą) gautume:

$$m_{\beta_1}(x) = x^3 + x^2 + x + 2, \quad m_{\beta_2}(x) = x^3 + x^2 + 2.$$

Mūsų sukonstruoto kodo dimensija $k = n - \deg(g) = 7$.

10.6. BCH kodų dekodavimas

BCH kodai yra geri tuo, kad jiems sukurta gerų klaidų taisymo algoritmų. Jų yra ne vienas. Aptarsime, ko gero, vieną pačių elegantiškiausių. Jame pasirodo Euklido algoritmas ir todėl padvelkia tikra klasika.

Tarkime, $\mathbf{L} \subset \mathbb{F}_p^n$ yra BCH kodas su numatytuoju atstumu $r = 2t + 1$. Taigi naudodamiesi šiuo kodu, garantuotai galime ištaisyti t klaidų. Šis kodas

yra Reedo-Solomono kodo iš abėcėlės $\mathbb{F}_q$ ($q = p^m, q-1$ dalijasi iš n) žodžių poaibis. Kodo generatorius yra daugianaris

$$g(x) = (x-\alpha)(x-\alpha^2)\cdots(x-\alpha^{2t}),$$

čia $\alpha \in \mathbb{F}_q$ yra n-osios eilės elementas. Kiekvienam kodo $\mathbf{L}$ žodžiui, interpretuojant jį kaip daugianarį

$$c(x) = c_0 + c_1 x + \ldots + c_{n-1}x^{n-1},$$

teisingos lygybės

$$c(\alpha) = c(\alpha^2) = \ldots = c(\alpha^{2t}) = 0.$$

Tarkime, siunčiant kodo žodį $c(x)$ kanalu, įvyko klaidos ir gautasis žodis yra

$$d(x) = c(x) + e(x), \quad e(x) = e_0 + e_1 x + \ldots + e_{n-1}x^{n-1}.$$

Jeigu surastume klaidų daugianarį $e(x)$, galėtume ištaisyti įvykusias klaidas. Tai įmanoma, jeigu klaidų skaičius ne didesnis už t, taigi – daugianario $e(x)$ svoris turi būti ne didesnis už t.

Kadangi daugianarį $d(x)$ žinome, tai galime suskaičiuoti dydžius

$$s_j = d(\alpha^j) = c(\alpha^j) + e(\alpha^j) = e(\alpha^j), \quad j = 1, 2, \ldots, 2t.$$

Dydžiai s_j – tai visa, ką žinome apie klaidų daugianarį. Tačiau to pakanka, kad jį surastume!

89 apibrėžimas. *Tegu*

$$\begin{aligned} s(x) &= s_1 + s_2 x + \ldots + s_{2t}x^{2t-1}, \\ \lambda(x) &= \prod_{\substack{j=0,\ldots,n-1 \\ e_j \neq 0}} \left(1 - \alpha^j x\right), \\ \omega(x) &= s(x) \times_f \lambda(x) \quad (f(x) = x^{2t}). \end{aligned}$$

Daugianarį $s(x)$ vadinsime gautojo žodžio sindromu, $\lambda(x)$ – klaidų lokatoriumi, $\omega(x)$ – klaidų identifikatoriumi.

Visi trys daugianariai yra $\mathbb{F}_q[x]$ elementai; žinome tik sindromą. Dar žinome, kad lokatoriaus laipsnis ne didesnis už t (nes darome prielaidą, kad įvyko ne daugiau kaip t iškraipymų, taigi ne daugiau kaip t koeficientų e_j nelygūs nuliui). Be to, jo laisvasis narys yra lygus vienetui. Jeigu lokatorių sužinotume, galėtume suskaičiuoti elementus

$$\lambda(\alpha^{-0}), \quad \lambda(\alpha^{-1}), \ldots, \lambda(\alpha^{-m}), \ldots, \lambda(\alpha^{-n+1});$$

nuliai šioje elementų eilėje parodytų klaidų vietą, taigi lokalizuotų klaidas.

Tačiau šio daugianario nežinome, taigi nežinome ir $\omega(x)$ – dviejų daugianarių sandaugos dalybos iš x^{2t} liekanos. Labai svarbu, kad šį daugianarį galime išreikšti naudojant $\lambda(x)$ kitu būdu.

113 teorema. *Teisinga lygybė*

$$\omega(x)=\sum_{i=0}^{n-1}e_i\alpha^i\prod_{\substack{j\neq i\\ e_j\neq 0}}(1-\alpha^j x)=\sum_{i=0}^{n-1}e_i\alpha^i\frac{\lambda(x)}{1-\alpha^i x}.$$

Šios teoremos išvadomis pasinaudosime konstruodami dekodavimo algoritmą. O jeigu norite sužinoti, kodėl jis veikia – įsigilinkite į įrodymą.

Įrodymas. Pirmiausia prisiminkime sindromo apibrėžimą:

$$\begin{aligned} s(x) &= s_1+s_2x+\ldots+s_{2t}x^{2t-1},\\ s_j &= e(\alpha^j),\quad s_j=e_0+e_1\alpha^j+e_2(\alpha^j)^2+\ldots+e_{n-1}(\alpha^j)^{n-1}. \end{aligned}$$

Taigi su $f(x)=x^{2t}$

$$\begin{aligned} \omega(x) &= s(x)\times_f\lambda(x)=\sum_{j=0}^{2t-1}\Big(\sum_{i=0}^{n-1}e_i(\alpha^{j+1})^i\Big)x^j\times_f\prod_{\substack{j=0,\ldots,n-1\\ e_j\neq 0}}(1-\alpha^j x)\\ &= \sum_{i=0}^{n-1}e_i\alpha^i\Big(\sum_{j=0}^{2t-1}(\alpha^i x)^j\Big)\times_f\prod_{\substack{j=0,\ldots,n-1\\ e_j\neq 0}}(1-\alpha^j x)\,. \end{aligned}$$

Dabar įsižiūrėkime į tokią daugianarių lygybę, kuri teisinga su bet kokio kūno elementu β:

$$(1+(\beta x)+(\beta x)^2+\ldots+(\beta x)^{2t-1})(1-\beta x)=1-(\beta x)^{2t}.$$

Jeigu imsime $\beta\in\mathbb{F}_q$ ir $f(x)=x^{2t}$, šią lygybę galime užrašyti

$$(1+(\beta x)+(\beta x)^2+\ldots+(\beta x)^{2t-1})\times_f(1-\beta x)=1. \tag{75}$$

Jeigu $\beta=\alpha^i$, tai iš (75) gauname

$$\Big(\sum_{j=0}^{2t-1}(\alpha^i x)^j\Big)\times_f(1-\alpha^i x)=1.$$

Tada

$$\omega(x)=\sum_{i=0}^{n-1}e_i\alpha^i\Big(\sum_{j=0}^{2t-1}(\alpha^i x)^j\Big)\times_f\prod_{\substack{j=0,\ldots,n-1\\ e_j\neq 0}}(1-\alpha^j x)=\sum_{i=0}^{n-1}e_i\alpha^i\prod_{\substack{j\neq i\\ e_j\neq 0}}(1-\alpha^j x)$$

ir teorema įrodyta.

O dabar – dvi svarbios išvados iš įrodytos teoremos, kuriomis pasinaudosime dekoduodami.

Viena vertus, klaidų identifikatorius yra ne didesnio kaip $t-1$ laipsnio daugianaris.

Jeigu žinotume, kad $e_k \neq 0$ (t. y. $\lambda(\alpha^{-k}) = 0$), ir apskaičiuotume $\omega(\alpha^{-k})$, gautume

$$\omega(\alpha^{-k}) = e_k \alpha^k \prod_{\substack{j \neq k \\ e_j \neq 0}} \left(1 - \alpha^j \alpha^{-k}\right).$$

O šį reiškinį galima užrašyti dar paprasčiau – pasinaudojus formaliomis daugianarių išvestinėmis (jas skaičiuojame pagal tas pačias realiųjų skaičių funkcijoms įrodytas taisykles):

$$\omega(\alpha^{-k}) = -e_k \lambda'(\alpha^{-k}).$$

Taigi klaidų identifikatoriaus vardas pateisintas: naudodamiesi $\omega(x)$, tikrai galime surasti klaidų daugianario koeficientus:

$$\text{jei } \lambda(\alpha^{-k}) = 0, \quad \text{tai} \quad e_k = -\frac{\omega(\alpha^{-k})}{\lambda'(\alpha^{-k})}.$$

Klaidų taisymo metodas jau yra, tik įrankiai – klaidų lokatorius ir identifikatorius – nepagaminti. Imkimės šio darbo.

Panagrinėkime klaidų identifikatoriaus apibrėžimo lygybę

$$\omega(x) = s(x) \times_f \lambda(x) \quad (f(x) = x^{2t}).$$

Ji reiškia, kad egzistuoja daugianaris $m(x) \in \mathbb{F}_q[x]$, kad

$$m(x)x^{2t} + \lambda(x)s(x) = \omega(x), \quad \deg(\omega) < t.$$

Iš šios lygybės matome, kad iš $\omega(x)$ dalijasi bendrasis didžiausiasis žinomų daugianarių x^{2t} ir $s(x)$ daliklis. Galbūt klaidų identifikatorius pasirodo kaip liekana kuriame nors šių daugianarių bendrojo didžiausiojo daliklio ieškojimo Euklido algoritmu žingsnyje? Ir tai beveik tiesa!

Klaidoms taisyti reikalingus daugianarius, naudodamiesi Euklido algoritmu, galime surasti šitaip: pažymėkime $f_0(x) = x^{2t}, f_1(x) = s(x)$ ir atlikime Euklido algoritmo žingsnius, kol gausime pirmąją liekaną, kurios laipsnis mažesnis už t:

$$\begin{aligned}
f_0(x) &= m_1(x)f_1(x) + f_2(x), \quad t \leq \deg(f_2) < \deg(f_1), \\
f_1(x) &= m_2(x)f_2(x) + f_3(x), \quad t \leq \deg(f_3) < \deg(f_2), \\
\dots &\quad \dots\dots\dots\dots\dots \\
f_{k-2}(x) &= m_{k-1}(x)f_{k-1}(x) + f_k(x), \quad t \leq \deg(f_k) < \deg(f_{k-1}), \\
f_{k-1}(x) &= m_k(x)f_k(x) + f_{k+1}(x), \quad 0 \leq \deg(f_{k+1}) < t \leq \deg(f_k).
\end{aligned}$$

Daugianaris $f_{k+1}(x)$ yra jau beveik klaidų identifikatorius. Pradėję nuo paskutinės lygybės ir kopdami į viršų, suraskime išraišką

$$f_{k+1}(x) = m_*(x)x^{2t} + \lambda_*(x)s(x).$$

Prisiminkime, kad klaidų lokatoriaus laisvasis narys turi būti lygus 1. Padauginę gautąją lygybę iš $\beta = \lambda_*(0)^{-1}$, gausime

$$\beta f_{k+1}(x) = (\beta m_*(x))x^{2t} + (\beta\lambda_*(x))s(x), \quad \omega(x) = \beta f_{k+1}(x),\ \lambda(x) = \beta\lambda_*(x).$$

Argi nenuostabu, kai toks prieš porą tūkstančių metų sugalvotas instrumentas išsprendžia tokius šiuolaikiškus uždavinius!

Panagrinėkime skaitinį pavyzdį.

Tarkime, norime sudaryti $n = 5$ simbolių ilgio BCH kodą iš aibės $\mathbb{F}_{11}$ žodžių, kuris taisytų vieną klaidą, t. y. $t = 1$. Tada numatytasis atstumas turi būti $r = 3$. Kokiame plėtinyje yra n-osios eilės elementas? Kadangi $p - 1 = 10$ dalijasi iš n, tai toli ieškoti nereiks, reikiamą elementą rasime jau kūne $\mathbb{F}_{11}$. Taigi šiuo atveju BCH kodas sutaps su Reedo-Solomono kodu, sudarytu iš abėcėlės $\mathbb{F}_{11}$ žodžių. Nesunku patikrinti, kad $\gamma = 2$ yra generuojantis šio kūno elementas, tada $\alpha = \gamma^2 = 4$ bus kodo konstrukcijai reikalingas elementas. Jau galime sukonstruoti ir generuojantį daugianarį:

$$g(x) = (x - \alpha)(x - \alpha^2) = (x - 4)(x - 5) = x^2 + 2x + 9.$$

Taigi mūsų kodo dimensija $k = 3$; sudarykime kokį nors kodo žodį, pavyzdžiui,

$$c(x) = (x^2 + 7)g(x) = x^4 + 2x^3 + 5x^2 + 3x + 8.$$

Tarkime, siunčiant kanalu, šis žodis pavirto į $d(x) = x^4 + 5x^2 + 3x + 8$. Pabandykime vien tik naudodamiesi juo surasti klaidų žodį. Sudarykime sindromą:

$$s_1 = d(\alpha) = d(4) = 4,\ s_2 = d(\alpha^2) = d(5) = 3, \quad s(x) = 4 + 3x.$$

Užtenka vieno Euklido algoritmo žingsnio:

$$x^{2t} = x^2 = (4x + 2)s(x) + 3,\ 3 = x^2 + (7x + 9)s(x),$$
$$4 = 5x^2 + (2x + 1)s(x),\ \lambda(x) = 2x + 1,\ \omega(x) = 4.$$

Skaičiuodami pasinaudojome tuo, kad $9^{-1} \equiv 5 \pmod{11}$. Taigi $\omega(x) = 4$ ir $\lambda(x) = 2x + 1$. Iš karto randame lygties $\lambda(x) = 0$ šaknį:

$$x = -2^{-1} = 5 = \alpha^2 = \alpha^{2-5} = \alpha^{-3}.$$

Taigi klaidų lokatorius rodo, kad neteisingai perduotas koeficientas prie x^3. Raskime atitinkamą klaidos žodžio koeficientą. Skaičiavimai labai paprasti, nes $\lambda'(x) = 2$:

$$e_3 = -\omega(\alpha^{-3})\lambda(\alpha^{-3})^{-1} = -4 \cdot 2^{-1} = 7 \cdot 6 = 9.$$

Taigi $e(x) = 9x^3$ ir $c(x) = d(x) - e(x) = x^4 + 2x^3 + 5x^2 + 3x + 8$. Nuostabu!

10.7. Klaidų pliūpsniai

Duomenų skaitymas iš optinio disko – irgi perdavimas kanalu. Jei diskas įbrėžtas, toks kanalas iškraipys daug simbolių, esančių arti vienas kito. Pliūptelės klaidos, o tada vėl perdavimas vyks be priekaišty. Kaip atsižvelgti į tokių kanalų ypatybes ir efektyviai taisyti klaidas?

Konstravome klaidas taisančius kodus, darydami prielaidą, kad klaidos nėra susijusios, t. y. kanalas iškreipia simbolius nepriklausomai vienas nuo kito. Tačiau esant techninės įrangos trikdžiams (pavyzdžiui, kai duomenų laikmena pažeista), klaidos linkusios sudaryti telkinius, kitaip tariant – įvyksta klaidų pliūpsniai. Tokiais atvejais mūsų kodas gali nesuteikti galimybės ištaisyti, pavyzdžiui, penkių vienas nuo kito atokiai esančių iškraipytų simbolių, tačiau turėtų ištaisyti dešimt klaidų, jeigu jos įvyksta viena po kitos.

90 apibrėžimas. *Jeigu, perdavus žodį* $\mathbf{c} \in \mathbb{F}_q^n$, *gautas žodis yra*

$$\mathbf{d} = \mathbf{c} + \mathbf{e}, \quad \mathbf{e} = 00\ldots0e_ie_{i+1}\ldots e_{i+N-1}0\ldots0, \quad e_i, e_{i+N-1} \neq 0,$$

sakysime, kad įvyko N *dydžio klaidų pliūpsnis.*

Toliau nagrinėsime dvejetainės abėcėlės atvejį, t. y. klaidų pliūpsnius, kurie atsiranda perduodant $\mathbb{F}_2^n$ aibės žodžius. Aptarsime tris metodus: simbolių išbarstymo, kodų su didelėmis abėcėlėmis ir specialių kodų.

Simbolių išbarstymo metodo idėja pati paprasčiausia. Tarkime, dvejetainės abėcėlės simbolių srautui koduoti turime kokį nors pakankamai gerą kodą $\mathbf{C} \subset \mathbb{F}_2^n$. Koduokime šiuo kodu m pradinio srauto fragmentų, gausime m kodo žodžių

$$\mathbf{c}_1 = c_{11}c_{12}\ldots c_{1n},\ \mathbf{c}_2 = c_{21}c_{22}\ldots c_{2n},\ \ldots\ , \mathbf{c}_m = c_{m1}c_{m2}\ldots c_{mn}.$$

Tačiau, sudarę šiuos žodžius, dar nesiųskime į kanalą! Surašykime juos į lentelę eilutė po eilutės:

$$\begin{array}{|c|c|c|c|c|c|} c_{11} & c_{12} & \ldots & c_{1k} & \ldots & c_{1n} \\ c_{21} & c_{22} & \ldots & c_{2k} & \ldots & c_{2n} \\ \cdot & \cdot & \ldots & \cdot & \ldots & \cdot \\ c_{m1} & c_{m2} & \ldots & c_{mk} & \ldots & c_{mn} \end{array}$$

O dabar į kanalą įveskime lentelės simbolius, nuskaitydami juos stulpelis po stulpelio, t. y. iš pradžių pirmąjį stulpelį $c_{11}c_{21}\ldots c_{m1}$, paskui antrąjį ir t. t.

Taigi į kanalą perduodame n žodžių po m simbolių. Tarkime, siunčiant duomenis, įvyko klaidų pliūpsnis ir visi simboliai iš k-ojo stulpelio buvo iškreipti. Kai gavėjas gaus nm simbolių, jis susirašys juos į lentelę stulpelis po stulpelio ir, perskaitęs žodžius iš eilučių, turės m kodo $\mathbf{C}$ žodžių, kuriuose

bus įvykę vos po vieną klaidą. Taigi klaidų pliūpsnio taisymas pavirsta pavienių klaidų taisymu keliuose žodžiuose. Tai galima padaryti naudojantis $\mathbf{C}$ kodo dekodavimo algoritmu.

Aptarkime kitą idėją. Tarkime, turime gerą kodą $\mathbf{C} \subset \mathbb{F}_{2^m}^n$, pavyzdžiui, BCH kodą. Dvejetainės abėcėlės žodžius $\mathbb{F}_2^m$ galime interpretuoti kaip abėcėlės $\mathbb{F}_{2^m}$ simbolius. Pirmasis žingsnis: paversti dvejetainių simbolių srautą abėcėlės $\mathbb{F}_{2^m}$ simbolių srautu:

$$x_1x_2\ldots x_mx_{m+1}x_{m+2}\ldots x_{2m}\ldots \mapsto y_1y_2\ldots, \quad x_i \in \mathbb{F}_2, y_j \in \mathbb{F}_{2^m}.$$

Dabar didelės abėcėlės simbolių srautą galime koduoti kodo $\mathbf{C}$ žodžiais, šių žodžių simbolius keisti $\mathbb{F}_2^m$ žodžiais ir į kanalą perduoti naują dvejetainių simbolių srautą:

$$y_1y_2\ldots \mapsto z_1z_2\ldots \mapsto u_1u_2\ldots, \quad y_j \in \mathbb{F}_{2^m},\ z_i \in \mathbb{F}_2^m,\ u_i \in \mathbb{F}_2.$$

Gavėjas gautąjį dvejetainių simbolių srautą turi paversti kodo $\mathbf{C}$ žodžiais, ištaisyti klaidas ir nustatyti pradinį dvejetainį simbolių srautą. Kokia padėtis susidarys, jeigu kanalas sukels m ilgio klaidų pliūpsnį? Galimi atvejai: įvyks viena arba dvi klaidos viename $\mathbf{C}$ žodyje; įvyks po vieną klaidą dviejuose $\mathbf{C}$ žodžiuose. Abiem atvejais klaidas galime ištaisyti naudodamiesi $\mathbf{C}$ dekodavimo algoritmu.

O dabar aptarkime klaidų pliūpsniams taisyti tinkamų specialių kodų sudarymo uždavinį. Kokias savybes turėtų turėti tiesinis dvejetainės abėcėlės kodas $\mathbf{C}$, kad jis galėtų taisyti klaidų pliūpsnius? Prisiminkime standartinę kodo lentelę, sluoksnius, sindromus ir lyderius. Iš sindromo sužinome, į kokį sluoksnį gautas iš kanalo žodis $\mathbf{d}$ pateko. Jeigu šiame sluoksnyje yra vienas lyderis $\mathbf{e}$, tai žodį dekoduojame taip:

$$\mathbf{d} \mapsto \mathbf{d} - \mathbf{e} = \mathbf{c}, \quad \mathbf{c} \in \mathbf{C}.$$

Taigi teisingai dekoduojama tik tais atvejais, kai klaidų žodis yra vienintelis kurio nors sluoksnio lyderis. Todėl tiesinis kodas galės teisingai dekoduoti klaidų pliūpsnius tik tada, kai tie pliūpsniai yra vieninteliai atitinkamų sluoksnių lyderiai.

Tačiau kaip sudaryti tokius kodus? Vieną tokių kodų šeimą 1959 metais sukonstravo P. Fire.

91 apibrėžimas. *Tegu $p(x) \in \mathbb{F}_2[x]$ yra neskaidus daugianaris, $\deg(p) = m$, s yra mažiausias natūralusis skaičius, kad $x^s + 1$ dalijasi iš $p(x)$, l yra natūralusis skaičius, kad $l \le m$ ir $2l - 1$ nesidalija iš s. Pažymėkime*

$$g(x) = (x^{2l-1} + 1)p(x), \quad n = \text{BMK}(2l - 1, s),$$

čia BMK *reiškia bendrąjį mažiausiąjį kartotinį. Tada daugianaris $g(x)$ dalija $x^n + 1$, o $g(x)$ generuotas ciklinis kodas $\mathbf{C} \subset \mathbb{F}_2^n$ vadinamas Fire kodu.*

Taigi Fire kodo dimensija yra $n - m - 2l + 1$. O kaipgi klaidų pliūpsnių taisymas?

114 teorema. *Fire kodas taiso visus klaidų pliūpsnius, kurių ilgiai neviršija l.*

Kaip įrodyti šį teiginį? Reikia įrodyti, kad bet kuris žodis, atitinkantis klaidų pliūpsnį, kurio ilgis ne didesnis kaip l, yra vienintelis atitinkamo sluoksnio lyderis.

Žodžius, atitinkančius klaidų pliūpsnius, galime užrašyti daugianariais

$$e(x) = x^i a(x), \quad a(x) = 1 + a_1 x + \ldots + a_{k-1} x^{k-1}, \quad \deg(e) < n, k \leq l.$$

Reikia įrodyti, kad jokie du tokie daugianariai negali priklausyti tam pačiam sluoksniui, t. y. būtinai yra savo sluoksnių lyderiai. Tarkime, $e_1(x), e_2(x)$ yra du klaidų pliūpsnius atitinkantys daugianariai. Jie nepriklausys tam pačiam sluoksniui tada ir tik tada, kai jų suma $e_1(x)+e_2(x)$ nebus kodo žodis. Tačiau visi Fire kodo žodžiai dalijasi iš generatoriaus $g(x)$. Taigi reikia įrodyti, kad $e_1(x) + e_2(x)$ negali dalytis iš $g(x)$. Šitaip įrodymas virsta uždaviniu apie specialaus pavidalo daugianarių dalybą. Nuo ko pradėti? Kaip pradedame labai dažnai: tarkime, kad $e_1(x) + e_2(x)$ dalijasi iš $g(x)$...

11 Sąsūkų kodai

11.1. Tiesiniai registrai ir daugianarių daugyba

Nors daugianarių daugyba gana sudėtingas veiksmas, jį galima greitai atlikti naudojant visai paprastą įrenginį.

Koduojant dvejetainės abėcėlės daugianarių kodais, aibės $\mathbb{F}_2[x]$ daugianarius tenka dauginti iš fiksuoto daugianario – kodo generatoriaus. Panagrinėkime, kaip gaunami šių daugianarių sandaugos koeficientai. Tegu generatoriaus vaidmuo tenka daugianariui

$$g(x) = 1 + x + x^2,$$

o kitas daugianaris tebūnie bet koks:

$$u(x) = u_0 + u_1 x + u_2 x^2 + \ldots + u_n x_n, \quad u_i \in \mathbb{F}_2.$$

Tada jų sandauga yra daugianaris

$$v(x) = v_0 + v_1 x + \ldots + v_n x^n + v_{n+1} x^{n+1} + v_{n+2} x^{n+2},$$

kurio koeficientai randami taip:

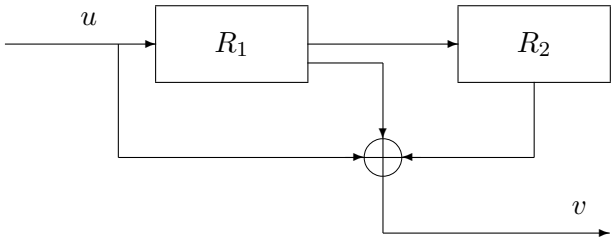
$$v_k = u_k + u_{k-1} + u_{k-2}, \quad k = 2, 3, \ldots, n.$$

Kad ši lygybė tiktų ir $v_0, v_1, v_{n+1}, v_{n+2}$, reikia susitarti, kad

$$u_{-2} = u_{-1} = u_{n+1} = u_{n+2} = 0.$$

Taigi sandaugos $v(x)$ koeficientai gaunami „susukant“ į vieną tris $u(x)$ koeficientus. Jeigu daugianarį $u(x)$ užrašysime jo koeficientų žodžiu, tai kiekvienas $v(x)$ koeficientų žodžio bitas gaunamas iš trijų $u(x)$ žodžio bitų.

Tą bitų sąsūkos veiksmą galime interpretuoti kiek kitaip.



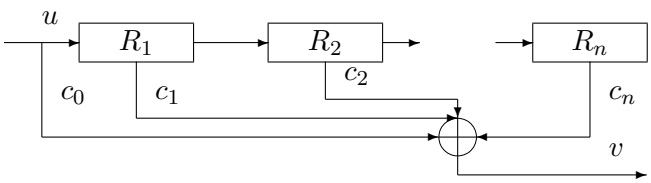
Daugianarių daugybos iš $g(x) = 1 + x + x^2$ tiesinių registrų sistema

Sistemą sudaro du registrai, kurie gali saugoti po vieną bitą. Iš pradžių juose yra nuliai. Į šią sistemą vienas po kito įvedami bitai $u_0, u_1, \ldots, u_n$ (bitas u_0 įvedamas pirmasis). Kai pasirodo u_0 – jis pirmiausia paduodamas į sumavimo įrenginį, į jį paduodami ir registruose saugoti bitai; juos sudėjus gaunamas pirmasis išvesties bitas $v_0 = u_0$. Tada bitas u_0 išstumia pirmojo registro bitą, pirmojo registro bitas – antrojo, o antrojo registro bitas tiesiog dingsta. Kai pasirodo u_1, viskas vyksta analogiškai, tačiau dabar gauname $v_1 = u_1 + u_0$, kai pasirodo u_2 – gauname $v_2 = u_2 + u_1 + u_0$ ir t. t.

Visus daugianario $v(x)$ koeficientus generuos šis paprastas įrenginys, jeigu į jį įvesime $u_0u_1 \ldots u_n00$. Taigi šis paprastas tiesinių registrų įrenginys atlieka daugianarių daugybą! O jeigu norėtume sukonstruoti tiesinių registrų sistemą, kuri daugianarius daugintų iš

$$g(x) = c_0 + c_1x + \ldots + c_nx_n, \quad c_n \neq 0?$$

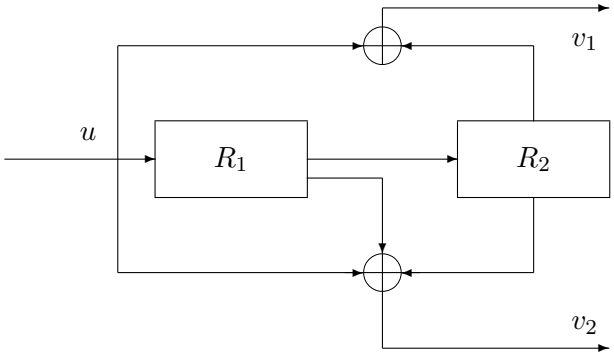
Sprendimas paprastas:



Daugianarių daugybos iš $g(x) = c_0 + c_1x + \ldots + c_nx_n$ tiesinių registrų sistema

Tokia tiesinių registrų sistema iš tikrųjų atlieka bitų srauto kodavimą: kiekvienam įvesties bitui u_i sukuriamas išvesties bitas v_i. Tačiau šitaip koduojant su

$g(x) = 1 + x + x^2$, kad ir koks ilgas būtų pradinis žodis, jis pailginamas tik dviem bitais. Gerų galimybių taisyti klaidas nėra ko tikėtis. Tačiau mes galime su ta pačia registrų sistema kievienam įvesties bitui generuoti ne vieną, bet, pavyzdžiui, du bitus. Štai ta patobulinta registrų sistema:



Įvesties daugianario daugybos iš $g_1(x) = 1 + x^2$ *ir* $g_2(x) = 1 + x + x^2$ *tiesinių registrų sistema*

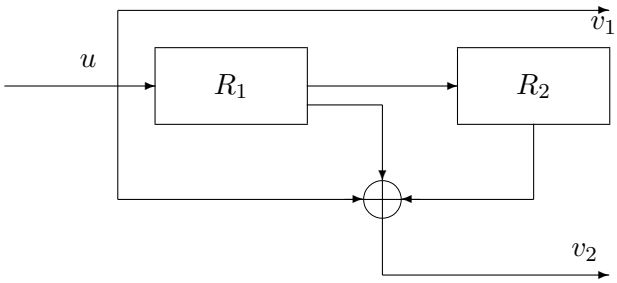
Dabar įvesties bitų žodžiui **u** sistema sukuria du išvesties žodžius $\mathbf{v}_1$ ir $\mathbf{v}_2$, arba, naudojantis daugianarių terminologija, kiekvienam įvesties daugianariui $u(x)$ sukuriami du išvesties daugianariai $v_1(x), v_2(x)$:

$$\begin{aligned} v_1(x) &= u(x)g_1(x), \quad g_1(x) = 1 + x^2, \\ v_2(x) &= u(x)g_2(x), \quad g_2(x) = 1 + x + x^2. \end{aligned}$$

Tai jau panašu į kodavimą, suteikiantį galimybių taisyti klaidas: juk kiekvieną įvesties bitą atitinka du išvesties srauto bitai. Įvesties daugianarį atitinka du išvesties daugianariai $v_1(x)$ ir $v_2(x)$. Galbūt norėtume „supinti“ abu išvesties srautus į vieną, pavyzdžiui, padaryti taip, kad kiekvieną srauto $\mathbf{v}_1$ bitą sektų atitinkamas $\mathbf{v}_2$ bitas? Tokį supintą srautą galima užrašyti daugianariu, kurio koeficientai prie lyginių laipsnių yra skolinti iš $v_1(x)$, o prie nelyginių – iš $v_2(x)$. Tokį daugianarį paprasta sudaryti:

$$v(x) = v_1(x^2) + xv_2(x^2) = u(x^2)(g_1(x^2) + xg_2(x^2) = u(x^2)(1 + x + x^3 + x^4 + x^5).$$

Yra ne vienas būdas sukurti du išvesties srautus iš vieno įvesties srauto. Štai dar vienas:



Sisteminio kodavimo tiesinių registrų sistema

Šios schemos sukurtas srautas $\mathbf{v}_1$ – tiesiog įvesties srauto kopija. Tai primena sisteminį kodavimą tiesiniais kodais su standartinio pavidalo generuojančia matrica. Šitaip koduojant kiekvienas duomenų blokas tiesiog pailginamas simboliais, suteikiančiais galimybę taisyti klaidas.

11.2. Sąsūkų kodai

Kartais koduojame daugindami iš matricos, kartais – iš daugianario.
O šiame skyrelyje – tiesiog keliaudami iš vienos vietos į kitą...

Sugrįžkime prie tiesinių registrų sistemos, kuri atlieka daugybą iš daugianarių $g_1(x) = 1 + x^2$, $g_2(x) = 1 + x + x^2$, (žr. ankstesnio skyrelio brėžinį). Ji apibrėžia bitų srauto $\mathbf{u} = u_0u_1\ldots$ kodavimą dvigubai ilgesniu srautu $\mathbf{v} = v_0^{(1)}v_0^{(2)}v_1^{(1)}v_1^{(2)}\ldots$, čia $v_i^{(1)}v_i^{(2)}$ žymime du bitus, kuriuos sukuria sistema įvesties bitui u_i. Matėme, kad šią sistemą galima nusakyti daugianarių atitiktimis:

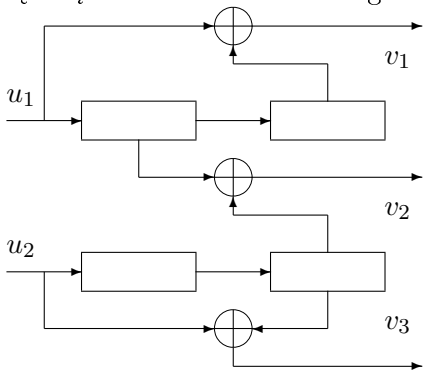
$$u(x) \mapsto \langle v_1(x), v_2(x)\rangle, \quad \text{arba } u(x) \mapsto v(x),$$

čia

$$\begin{aligned} v_1(x) &= u(x)g_1(x), \quad g_1(x) = 1 + x^2, \\ v_2(x) &= u(x)g_2(x), \quad g_2(x) = 1 + x + x^2, \\ v(x) &= u(x^2)g(x), \quad g(x) = 1 + x + x^3 + x^4 + x^5. \end{aligned}$$

Sakysime, kad ši registrų sistema (arba daugianarių atitiktys) apibrėžia sąsūkų kodą su parametrais $(k, m, n) = (1, 2, 2)$. Šie parametrai nusako tokias sistemos savybes: k – viename žingsnyje į sistemą įvedamų bitų skaičių, m – sistemos registrų skaičių, kitaip tariant – įvestų bitų, kuriuos atsimena sistema, skaičių, n – viename žingsnyje išvedamų bitų skaičių. Santykį k/n vadinsime sąsūkų kodo koeficientu. Taigi mūsų sistema apibrėžia sąsūkų kodą su koeficientu $1/2$.

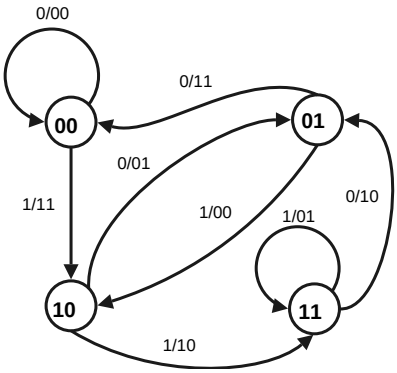
Galime apibrėžti sąsūkų kodus su $k > 1$. Panagrinėkime brėžinį.



Sąsūkų kodo su parametrais $(2, 4, 3)$ *tiesinių registrų sistema*

Į brėžinyje pavaizduotą registrų sistemą kiekviename žingsnyje paduodami du bitai, o išvedami trys. Taigi tokio kodo koeficientas yra 2/3.

Panagrinėsime dar du būdus sąsūkų kodų darbui vaizduoti. Nagrinėsime kodą, kurį atitinka tiesinių registrų sistema su daugianariais $g_1(x), g_2(x)$.. Kiekviename žingsnyje sukuriami du išvesties bitai priklauso nuo registrų turinio ir įvedamo bito. Sakysime, kad registrų turiniai apibrėžia sistemos būseną. Taigi galimos keturios sistemos būsenos, kurias užrašysime bitų poromis $00, 01, 10, 11$. Jeigu, pavyzdžiui, sistemos būsena yra 01, o įvedame 1, tai išvesties bitai bus 00, o sistema pereis į būseną 10. Įvedamus ir išvedamus bitus žymėsime taip: 1/00. Pavaizduosime būsenų aibę grafiškai, nurodydami visus galimus būsenų pasikeitimus.

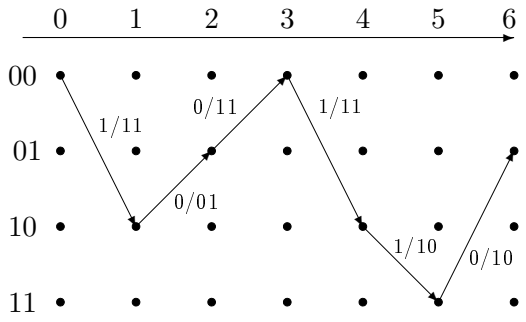


Sąsūkų kodo su parametrais $(1, 2, 2)$ *būsenų diagrama*

Tokią diagramą galime sudaryti bet kokiam sąsūkų kodui. Ja patogu naudotis koduojant srautą. Reikia keliauti tais diagramos keliais, kuriuos nurodo įvesties srauto bitai, ir užrašinėti išvesties srautą. Pavyzdžiui, koduojant pagal pateiktą diagramą,

$$\mathbf{u} = 10011010\ldots \mapsto 1101111110100001\ldots = \mathbf{v}.$$

Koduodami su būsenų diagrama braižome kelią, vedantį nuo vienos būsenos prie kitos. Galime jį „ištiesinti“, kad nebebūtų kilpų ir persidengimų. Tai padarykime taip. Nubrėžkime laiko ašį, pažymėdami joje laiko momentus (žingsnius), o po kiekvieno žingsnio žyma išrikiuokime būsenas žyminčių skrituliukų stulpelį. Dabar kelią nuo būsenos prie būsenos galėsime pavaizduoti laužte.



Šis kelias vaizduoja įvesties žodžio 100110 *kodavimo eigą*

Prie kelio briaunų parašyta, kokius įvesties ir išvesties bitus ta briauna atitinka. Pastebėkime, kad įvesties bitą visada galime atpažinti iš briaunos krypties. Jeigu briauna veda į viršų – įvesties bitas lygus nuliui. Jeigu žemyn – vienetas. Tiesa, yra du atvejai, kai briaunos lygiagrečios laiko ašiai: kai iš 00 grįžtama į 00 ir iš 11 į 11. Pirmuoju atveju įvesties bitas nulis, antruoju – vienetas.

Šią savybę turės bet kokio sąsūkų kodo diagrama, jeigu tik jo būsenas išdėstysime stulpeliu jas žyminčių žodžių $00\ldots0, 00\ldots1, \ldots 11\ldots1$ didėjimo (interpretuojant juos kaip skaičius) tvarka.

Tokia diagrama vadinama sąsūkų kodo grotelių diagrama. Kiekvienas kodavimo eigą atitinkantis kelias prasideda nuo nulinės būsenos. Mūsų pavyzdyje yra iš viso $4 \times 4 \times \ldots \times 4 = 4^m$ kelių, vedančių iš pradinės būsenos į kurią nors m-ojo žingsnio būseną. Tačiau ne visi keliai atitinka kokių nors srautų kodavimo eigą. „Tikrų“ kodavimo kelių yra tik 2^m. Jeigu sistemoje būtų, pavyzdžiui, s atminties registrų, tai iš viso kelių būtų 2^{sm}, o kodavimo kelių – tiek pat, t.y. 2^m.

11.3. Viterbi algoritmas

Tai būdas dekoduoti sąsūkų kodus, pasinaudojant kiekvienam suprantamu dėsniu: jeigu kelias ABC ilgesnis už kelią ADC, tai kelias ABCX irgi ilgesnis už kelią ADCX. Taigi taupant laiką, reikia iš A į X keliauti per D.

Tarkime, kodavimui naudojamas tas sąsūkų kodas, kurį nagrinėjome ankstesniajame skyrelyje. Galima koduoti kokio tik norime ilgio bitų srautus, tačiau susitarkime koduoti taip: bus koduojama po r srauto bitų, tada dar bus įvedami du nuliniai bitai, kad sistema sugrįžtų į būseną 00. Taigi iš tiesų į sistemą bitas po bito bus įvedami žodžiai $u_0u_1\ldots u_{r-1}00$. Po $r+2$ žingsnių sistema sugrįš į pradinę padėtį ir vėl bus koduojama iš naujo. Jeigu sistemoje būtų s atminties registrų, tada prie koduojamo bloko reiktų pridėti iš viso s nulių.

Kiekvieną žodį $u_0u_1\ldots u_{r-1}00$ grotelių diagramoje atitinka kelias iš būsenos 00 į 00. Tokių kodavimo kelių yra 2^r. Į kanalą perduodamas $2(r+2)$ ilgio

žodis

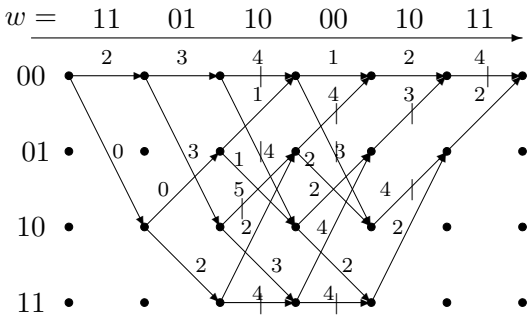
$$\mathbf{v} = v_0^{(1)} v_0^{(2)} v_1^{(1)} v_1^{(2)} \dots v_{r+1}^{(1)} v_{r+1}^{(2)}.$$

Tarkime, kad kanalas yra simetrinis ir be atminties. Tada jis su vienodomis tikimybėmis iškraipo perduodamus simbolius ir gavėjas vietoj srauto $\mathbf{v}$ gaus iškraipytą srautą

$$\mathbf{w} = w_0^{(1)} w_0^{(2)} w_1^{(1)} w_1^{(2)} \dots w_{r+1}^{(1)} w_{r+1}^{(2)}.$$

Viterbi algoritmo esmė – pagal gautąjį srautą nustatyti tą kodavimo kelią grotelių diagramoje, kurį atitinkantis išvesties bitų srautas $\mathbf{v}$ mažiausiai skiriasi nuo gautojo $\mathbf{w}$. Skirtumui matuoti, kaip įprasta kodavimo teorijoje, pasinaudokime Hammingo atstumu. Taigi reikia rasti tą kelią $\mathbf{v}$ iš 2^r galimų kelių aibės, su kuriuo Hammingo atstumo $h(\mathbf{v}, \mathbf{w})$ reikšmė yra mažiausia.

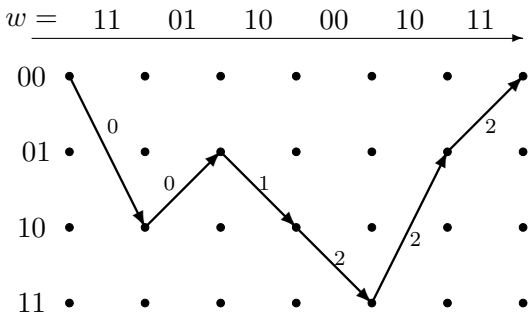
Panagrinėkime, kaip tai atliekama mūsų kodo atveju su $r = 4$. Dekoduotojas žino, kad po 6 žingsnių registrų sistemos būsena vėl tapo 00. Tarkime, iš kanalo gautas toks bitų srautas $w = 11\ 01\ 10\ 00\ 10\ 11$. Tai vieno keturių bitų žodžio, papildyto dviem nuliais, kodavimo rezultatas. Ieškosime, kuris iš galimų srautų mažiausiai skiriasi nuo gautojo.



Šis kelias vaizduoja įvesties žodžio 100110 *kodavimo eigą*

Pradėkime braižyti grotelių diagramą. Pavaizduokime būsenų stulpelius, užrašykime kiekvieną iš šešių žingsnių atitinkančią gauto srauto bitų porą. Žinome, kad kelias grotelių diagramoje prasideda iš nulinės būsenos. Galimi du keliai. Jeigu perduotas bitas 0, tai sistemos būsena liko ta pati, o išvesties bitai – 00. Užrašykime virš atkarpos, atitinkančios šį perėjimą, Hammingo atstumą tarp 00 ir gautų šį žingsnį atitinkančių bitų 11. Jis lygus 2. Tačiau jeigu buvo įvestas bitas 1, tai išvesties bitai tokie patys kaip gautieji, t.y. atstumas 0. Antrajame žingsnyje galime startuoti iš dviejų būsenų 00 ir 10. Iš viso galimi keturi tęsiniai. Pavyzdžiui, jeigu esant būsenai 00, buvo įvestas 1, tai išvesties bitai bus 11. Ši pora skiriasi nuo gautos poros 01, vadinasi, Hammingo atstumas padidėja vienetu. Užrašome sukauptą Hammingo atstumą 3 prie atitinkamos kelio grandies. Šitaip galimus kelius braižome ir toliau, užsirašydami sukauptus atstumus. Po trečiojo žingsnio pasirodo šis tas įdomaus. Į visas būsenas galime patekti dviem keliais, vienas jų „tolimesnis“

(sukauptas Hammingo atstumas didesnis), kitas – „trumpesnis“. Taigi galime atsisakyti ilgesnių kelių, tiesiog perbraukdami juos. Po keturių žingsnių kelius galime braižyti paprasčiau: juk žinome, kad penktasis ir šeštasis bitai lygūs nuliui. Taigi iš kiekvienos būsenos galimas tik vienas kelias. Ir ką gi gauname pabaigoje? Vienintelį kelią! Tai ir yra tas kelias iš visų galimų, kurį atitinkantis srautas mažiausiai skiriasi nuo gautojo.



Šį kelią atitinkantis išvesties srautas mažiausiai skiriasi nuo gautojo

Prisiminę, kad kryptis žemyn reiškia, kad buvo įvestas 1, aukštyn – 0, galime iškart atkurti ir pradinį į sistemą įvestą žodį: $\mathbf{v} = 101100$.

12 Pastabos ir nuorodos

Savo kodą Hammingas išdėstė straipsnyje [36]. Tai vienas pirmųjų kodavimo teorijos darbų. Parametrų įverčiai, kurie dėstomi šiame skyriuje, gauti naudojantis iš esmės vien tik kombinatorika. Atskirais atvejais juos galima patikslinti. Parametrų įverčiams skirtus skyrelius rasite knygose [48], [21]. Knygoje [21] taip pat gana išsamiai išdėstyta Hadamardo kodų teorija. Šiai knygai skirtame tinklalapyje http://www.wiris.com/cc/ rasite elektroninę kopiją, taip pat – skaičiavimams skirtas programas. Skaičiavimus, susijusius su kodų konstravimu ir tyrimu, galima atlikti naudojantis universaliomis kompiuterinėmis sistemomis Maple, Mathematica, MatLab. Verta paminėti dar vieną specialią – GAP (Groups, Algorithms, Programming – a System for Computational Discrete Algebra) [33]. Vienas jos paketas – GUAVA – skirtas kodams tyrinėti. Šis paketas teikia tikrai labai daug galimybių, be to, GAP sistema platinama nemokamai.

Dvinaris Golay kodas paskelbtas 1949 metais autoriaus straipsnyje [31]. Įdomu, kad straipsnis apie trinarį Golay kodą pasirodė 1947 metais suomių žurnale, skirtame futbolui. Kodo savybės jame naudojamos turnyro baigčių aibei tyrinėti. Naudojant kombinatorikos sąvokas, kodų struktūra nagrinėjama P.J. Camerono ir J. H. van Linto knygoje [15].

Sąsūkų kodai pirmą kartą buvo apibrėžti 1955 metais Eliaso darbe [25]. Ciklinius kodus 1957 metais pradėjo nagrinėti Prange [61]. Reedo-Mullerio

kodus autoriai nepriklausomai vienas nuo kito paskelbė 1954 metais [62], [57]. Straipsnį apie BCH kodus 1959 metais paskelbė A. Hocquenghemas [38], o 1960 metais – Bose ir Ray-Chaudhuri [12]. Reedo-Solomono kodai taip pat pasirodė 1960 metais [63]. Greitą BCH kodų dekodavimo algoritmą 1967 metais paskelbė Berlekampas [7]. Panaudoti Euklido algoritmą šių kodų dekodavimui – Sugiyamos idėja [**?**]. Viterbi algoritmą sąsūkų kodams dekoduoti autorius paskelbė 1971 metais [77].

1980 metais Sony ir Philips kompanijos susitarė dėl optinių diskelių (CD) standartizavimo. Kodavimui buvo panaudotas Reedo-Solomono kodas. 1993 atsirado naujo tipo kodai – turbo kodai. Jų autoriai – Berrou, Glavieux, Thitimajshima [8].

Kodavimo teorijai – daugiau kaip penkiasdešimt metų. Tai veržlios raidos metai, nes kodavimo teorija plėtojosi kartu su moderniomis ryšių technologijomis. Daugelis svarbių darbų paskelbti tarptautinės organizacijos IEEE (Institute of Electrical and Electronics Engineers) nuo 1953 metų leidžiamame žurnale „IEEE Transactions on Information Theory“.

Išleidžiama nemažai kodavimo teorijos monografijų ir vadovėlių. Paminėsime kelias knygas, kuriose išsamiai dėstoma klasikinė kodavimo teorija, o taip pat ir naujos kryptys: [50], [48], [10], [65], [40], [75].

Kriptografija

13 Įvadas

Jei kompiuteris yra kasdienės jūsų veiklos įrankis, tikriausiai esate patyrę, kaip muša šaltas prakaitas pastebėjus, kad po jūsų skaitmeninių duomenų – dokumentų ir programų – archyvą pasišvaistyta piktavalio viruso ar neišmanėlio jaunesniojo brolio. O jeigu kas įveiktų banko duomenų apsaugos sistemą ir patvarkytų jūsų sąskaitas?

Taigi informacijos apsaugos reikšmė yra nepaprastai didelė ir vis didėja.

Galima teigti, kad ir kodavimo teorija, nagrinėta antrojoje knygos dalyje sprendžia informacijos apsaugos uždavinius. Iš tiesų, siuntėjas A prieš siųsdamas informaciją gavėjui B gali pasinaudoti kodavimo teorijos idėjomis ir parengti informaciją taip, kad tikimybė, jog B gaus neiškraipytą informaciją, padidėtų. Šitaip sprendžiamas kanalu perduodamos informacijos vientisumo išsaugojimo uždavinys, kai pažeidimų (iškraipymų) priežastis – atsitiktiniai fizinės prigimties trikdžiai.

O dabar įsivaizduokime, kad informacijos perdavimo kanalą gali kontroliuoti protingas moderniausia skaičiavimo technika ir naujausiomis žiniomis apsiginklavęs Z. Šio skyriaus paskirtis – aptarti informacijos apsaugos uždavinius, kurie tuomet iškyla.

13.1. Kriptologijos dėmenys

Kriptologija – mokslas apie matematines duomenų apsaugos priemones. Kas ją sudaro ir kaip keliami jos uždaviniai?

Apžvelkime gerokai supaprastintą padėtį, į kurią patenka šiuolaikinės informacinės visuomenės žmonės: A (Algis) siunčia informaciją B (Birutei), perdavimo kanalą kontroliuoja Z (Zigmas) ir jaučiasi padėties šeimininkas. Jis gali pasyviai stebėti perdavimo kanalą, skaityti siunčiamus pranešimus ir kaupti dosjė; jis gali veikti aktyviai – pakeisti dalį siunčiamos informacijos, o kartais – apsimesti A ir siųsti jo vardu pranešimus B arba apsimesti B. Taigi dėl Zigmo veiksmų gali būti pažeidžiamos šios siunčiamų duomenų savybės:

- slaptumas;
- vientisumas;
- autentiškumas.

Kaip to išvengti? Fizinė kanalo apsauga gali būti pernelyg brangi arba tiesiog neįmanoma. Kokiomis priemonėmis galėtumėte apsaugoti, pavyzdžiui, jūsų elektroninio pašto perdavimo kanalą?

Prisiminkime kodavimo teorijos konstrukcijas. Informacijos, kurią reikia perduoti, blokai keičiami specialaus kodo žodžiais. Taigi – galimų iškraipymų taisymo priemonės „įmontuojamos“ pačioje perduodamų duomenų struktūroje. Ar panašiai negali būti sprendžiami ir informacijos apsaugos uždaviniai, kuriuos kelia Zigmo egzistavimas?

Kriptografija bendriausiu požiūriu ir yra duomenų apsaugos uždavinių sprendimo matematiniais metodais mokslas. Pagrindiniai šios apsaugos tikslai yra trys: užtikrinti informacijos slaptumą, vientisumą ir autentiškumą. Tačiau šiuolaikinė kriptografija tuo toli gražu neapsiriboja. Naudojimasis kompiuteriniais tinklais ir duomenų bazėmis kelia daug naujų įdomių teorinių ir praktinių uždavinių. Tik pagalvokime, pavyzdžiui, kiek reikalavimų reiktų įgyvendinti ir numatyti atvejų, norint tinkamai organizuoti elektroninius rinkimus! Pirmiausia reikia išduoti biuletenius turintiems teisę balsuoti, užkirsti kelią įvairaus pobūdžio klastotėms, garantuoti balsavimo anonimiškumą, suteikti rinkėjui galimybę patikrinti, kad jo balsas tinkamai įskaitytas...

Jeigu A ir B savo ryšiui apsaugoti naudoja tinkamus kriptografijos metodus, tai Zigmui, net ir kontroliuojant perdavimo kanalą, turėtų būti sunku „prisibrauti“ prie siunčiamos informacijos prasmės, pakeisti ką nors ir likti nepastebėtam arba pasiųsti pranešimą A ar B vardu. Sunku, bet nėra neįmanoma! Z kanalu siunčiamiems informacijos srautams analizuoti irgi gali naudoti mokslinius metodus. Jis gali žinoti daugelį naudojamų kriptografinių algoritmų detalių (išskyrus slaptus parametrus – raktus) ir bandyti įveikti kriptografinę duomenų apsaugą. Jo metodų visuma irgi yra mokslas – kriptoanalizė. Apibendrinant galima teigti, kad kriptoanalizė – tai duomenų kriptografinių apsaugos algoritmų patikimumo vertinimo mokslas. Kriptografija siūlo, o kriptoanalizė vertina. Šiuo požiūriu Z nėra A ir B priešas, bet pagalbininkas. Nieko nėra blogiau už apsaugą, kurios patikimumas nėra įvertintas!

O sugretinę kriptografiją ir kriptoanalizę, gauname kriptologiją – mokslą apie duomenų apsaugą naudojant matematikos ir informatikos priemones:

$$\text{kriptologija} = \text{kriptografija} + \text{kriptoanalizė}.$$

Verta paminėti, kad dešinėje lygybės pusėje sau vietos ima reikalauti dar vienas dėmuo – steganografija. Internetu dabar siunčiame pačią įvairiausią informaciją: dokumentus, fotografijas, audio- ir videoinformaciją... Ar negalima siunčiamoje kokio nors obuolio fotografijoje paslėpti svarbų pranešimą, kurį kitaip būtų rizikinga siųsti? Tokių metodų kūrimo ir analizės sritis ir yra steganografija (steganos – slėpti, graphein – rašau (graik.). Šį žodį 1499 metais pirmą kartą pavartojo vokietis Johannes Trithemius, pirmojo spausdinto kriptografijos veikalo autorius. Tačiau jam steganografija reiškė ne tik

mokymą apie šifrus, bet ir apie įvairius okultinius su paslaptimis susijusius dalykus. Kriptografija – taip pat iš graikiškų žodžių sudarytas pavadinimas (krypto – paslėptas graik.) Šį terminą 1661 metais pirmą kartą panvartojo Johnas Williamsas. Kriptografinės ir steganografinės duomenų apsaugos metodai esmingai skiriasi. Kriptografija neslepia duomenų, bet paslepia jų struktūrą, t. y. prasmę. O steganografija paslepia pačius duomenis.

13.2. Šifrai ir parašai

Šifrai ir skaitmeniniai parašai yra duomenų slaptumo ir autentiškumo užtikrinimo priemonės. Patikslinkime šias sąvokas.

Norėdami, kad pašalinis asmuo neįžvelgtų siunčiamų duomenų prasmės, turime pertvarkyti tuos duomenis taip, kad tik teisėtas gavėjas galėtų atkurti pradinę duomenų struktūrą. Duomenų pertvarkymą, kurio tikslas – paslėpti prasmę, vadinsime šifravimu, o pradinės struktūros atkūrimą – dešifravimu. Pradinius duomenis, kuriems taikoma šifravimo operacija vadinsime nešifruotu arba atviru tekstu, o šifravimo operacijos rezultatą – šifru.

Pagrindinis šiuolaikinės kriptografijos reikalavimas – kad šifravimo ir dešifravimo operacijų rezultatai iš esmės priklausytų nuo tam tikrų dydžių, paprastai vadinamų raktais. Tai reiškia, kad net jeigu visos A ir B ryšio slaptumą saugančios sistemos detalės (išskyrus raktus) taptų žinomos Z, įveikti duomenų apsaugos sistemą jam neturėtų pasidaryti lengviau (mažai naudos bus žmogui, norinčiam paskambinti telefonu, jeigu mes jam paaiškinsime, kad reikia vieną po kito paspausti septynis telefono mygtukus su numeriais, bet nepasakysime kokius).

Duomenų apsaugos sistemą, naudojančią šifravimą, vadinsime kriptografine sistema arba tiesiog kriptosistema.

92 apibrėžimas. *Kriptografine sistema vadinsime trejetą* $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$ *čia* $\mathcal{M}$ *– nešifruotų tekstų,* $\mathcal{K}$ *– naudojamų raktų ir* $\mathcal{C}$ *– šifrų aibės, kartu su šifravimo ir dešifravimo operacijomis*

$$e(\cdot|K_e) : \mathcal{M} \to \mathcal{C}, \quad d(\cdot|K_d) : \mathcal{C} \to \mathcal{M}, \quad \langle K_e, K_d \rangle \in \mathcal{K},$$

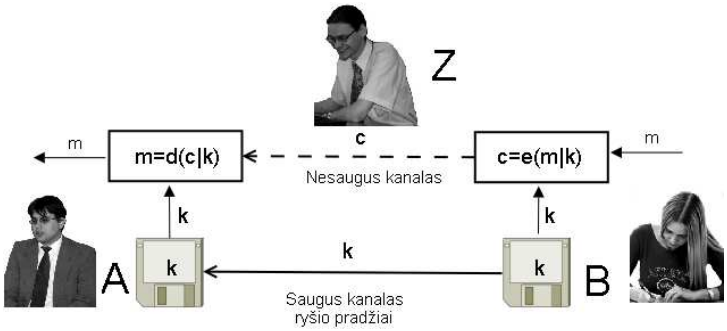
tenkinančiomis sąlygą

$$\text{kiekvienam } M \in \mathcal{M} \quad d(e(M|K_e)|K_d) = M.$$

Galime įsivaizduoti, kad kriptosistemoje naudojamas raktas sudarytas iš dviejų dalių: šifravimui skirto rakto K_e ir dešifravimui skirto rakto K_d.

Gali būti, kad abu raktai sutampa, t. y. $K_e = K_d$, arba, nors formaliai ir būdami skirtingi, lengvai gaunami iš vienas kito. Tokias kriptosistemas vadinsime simetrinėmis. Jeigu Birutė norėtų, kad Algis jai parašytų laišką ir užšifruotų jį simetrine kriptosistema, ji turi slapta nuo Zigmo perduoti Algiui raktą K_e ir jau tuomet laukti laiško. Taigi simetrinės kriptosistemos

veiklos pradžiai būtina nors ir trumpalaikė galimybė pasinaudoti visiškai saugiu kanalu raktui perduoti.



Ryšio apsauga su simetrine kriptosistema

Iki 1976 metų visos kriptosistemos buvo simetrinės. Tais metais du matematikai – Diffie ir Hellmannas[5] paskelbė naujos kartos kriptosistemų principus. Jų esmė tokia: nors dešifravimui skirtas raktas K_d yra susijęs su šifravimui skirtu raktu K_e, tačiau praktiškai, žinant K_e, neturi būti įmanoma per „tikrovišką“ laiką nustatyti K_d. Kaip tai gali būti įmanoma? Galbūt suprasti padės toks „hipotetinis“ pavyzdys.

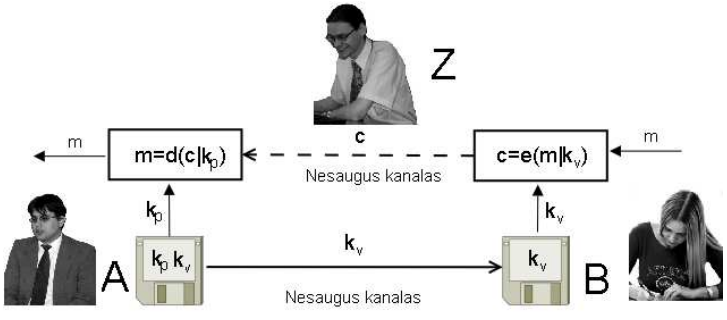
Tarkime, aš sukūriau kriptosistemą ir sudariau raktų porą:

$$K_e = 2, \quad K_d = a_n a_{n+1} \dots a_{n+99}; \qquad n = 10^{10^{10}}, \ \sqrt{K_e} = a_0, a_1 a_2 \dots,$$

čia a_i yra skaičiaus skleidimo begaline dešimtaine trupmena skaitmenys.

Ar, žinodami šifravimui skirtą raktą, greitai galėsite nustatyti, koks raktas naudojamas dešifravimui?

Taigi turėdama tokią kriptosistemą, Birutė gali perduoti šifravimui skirtą raktą K_e ir nesaugiu kanalu (pavyzdžiui, paskelbti savo tinklalapyje). Kadangi šifravimui skirtas raktas gali būti paskelbtas viešai, tokios kriptosistemos pradėtos vadinti viešojo rakto kriptosistemomis. Šifravimui skirtas raktas dažnai vadinamas viešuoju, o dešifravimui – privačiuoju raktu.



[5] W. Diffie, M. E. Hellman. New Directions in Cryptography. IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, p. 644–654.

Ryšio apsauga su viešojo rakto kriptosistema

Viešojo rakto kriptosistemą galime palyginti su rakinama pašto dėžute. Visi gali mesti laiškus pro plyšį, t. y. naudotis viešuoju raktu, tačiau tik dėžutės savininkas gali atsirakinti ją ir išsiimti (dešifruoti) laiškus.

Dabar viešojo rakto kriptosistemų yra sugalvota daug. Pats įdomiausias dalykas – konstruojant šias sistemas, taikomi „gryniausios" matematikos srities – skaičių teorijos – rezultatai. Erdvė, kuri skyrė praktinius uždavinius sprendžiančius informatikus ir „grynuosius" matematikus, dirbančius skaičių teorijos srityje, tapo didelę įtaką šiuolaikinių ryšių raidai padariusio bendradarbiavimo vieta.

Viešojo rakto kriptosistemų atsiradimas išsprendė ir skaitmeninių parašų uždavinį. Mūsų parašas yra tam tikra grafinė informacija, kurią susiejame su dokumentu. Parašo tikrinimo algoritmas labai paprastas – lyginama su dokumentu. O kaip pasirašyti skaitmeninį dokumentą, t. y. nulių-vienetų srautą? Kaip pridėti tą papildomą mus autentifikuojančią informaciją, kad būtų galima patikrinti, kad ją tikrai sukūrėme mes, o ne kažkas už mus?

Sistemą, skirtą skaitmeninių duomenų autentiškumui įrodyti, vadinsime skaitmeninio parašo schema. Ją formaliai galime apibrėžti taip:

93 apibrėžimas. *Skaitmeninių parašų schemą sudaro aibės $\mathcal{M}, \mathcal{P}, \mathcal{K}$, čia $\mathcal{M}$ – tekstų aibė, $\mathcal{P}$ – skaitmeninių parašų aibė, $\mathcal{K}$ – raktų $K = \langle K_v, K_p \rangle$ aibė (K_p – privačioji, parašui sudaryti skirta rakto komponentė, K_v – viešoji, parašui tikrinti skirta rakto komponentė) ir parašų sudarymo bei tikrinimo algoritmų šeimos*

$$\begin{aligned} sig(\cdot|K_p) &: \mathcal{M} \to \mathcal{P}, \\ ver(\cdot|K_v) &: \mathcal{M} \times \mathcal{P} \to \{0,1\}. \end{aligned}$$

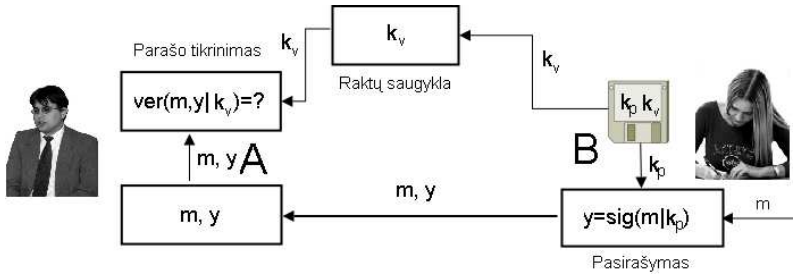
Parašo tikrinimo algoritmai turi savybę:

$$ver(x, sig(x|K_p)|K_v) = 1. \tag{76}$$

Jeigu $y \in \mathcal{P}$ pateikiamas kaip dokumento $x \in \mathcal{M}$ parašas, tai parašas pripažįstamas galiojančiu, jeigu $ver(x, y|K_v) = 1$, ir pripažįstamas negaliojančiu, jei $ver(x, y|K_v) = 0$.

Savybė (76) reiškia, kad tinkamai sudarytas parašas visada bus pripažįstamas. Tai būtina kiekvienos skaitmeninio parašo schemos savybė. Tačiau skaitmeninio parašo schema būtų bevertė, jeigu būtų nesunku klastoti parašus, t. y. jeigu iš viešojo rakto K_v būtų lengva nustatyti parašui sudaryti reikalingą privatųjį raktą K_p arba ir nežinant privataus rakto būtų galima parinkti poras $x \in \mathcal{M}, y \in \mathcal{P}$, tenkinančias lygybę $ver(x, y|K_v) = 1$.

Yra du būdai išvengti klastojimo. Viena vertus, galime tikrinimui skirtą raktą atskleisti tik tada, kai reikia patikrinti parašą ir daugiau jo nebenaudoti. Tokios skaitmeninių parašų schemos vadinamos vienkartinėmis. Arba galima naudoti viešojo rakto kriptosistemų konstrukcijas ir sudaryti skaitmeninio parašo schemą taip, kad, žinant K_v, praktiškai nebūtų įmanoma nustatyti K_p. Tada tuos pačius schemos raktus bus galima naudoti daugeliui dokumentų pasirašyti.



Skaitmeninio parašo schema

Apskritai skaitmeninio parašo schemą galime įsivaizduoti kaip kriptosistemą, kurioje raktai sukeisti vietomis: šifras (parašas) sudaromas naudojant privatųjį raktą, o šifras dešifruojamas (parašas tikrinamas) naudojant viešąjį raktą.

13.3. Algoritmai ir protokolai

> *Jeigu į savo butą įstatysite šarvuotas duris, bet ir toliau išeidami iš namų paliksite raktą po kilimėliu... patys spręskite, ar šarvuotos durys apsaugos jūsų turtą. Jeigu su draugu nutarėte susitikti dideliame svetimos šalies mieste, bet nesusitarėte dėl vietos – net ir detalus miesto žemėlapis nepadės. Panašiai ir su kriptografine apsauga: maža turėti saugius kriptografinius algoritmus, reikia gerai pagalvoti, kaip jie turi būti sujungti į visumą.*

Šifravimas, dešifravimas, parašo sudarymas, tikrinimas... – tai veiksmai, kuriuos gali atlikti vienas asmuo. Jis naudojasi patikimais įrankiais – kriptografiniais algoritmais. Tikrovėje niekas neatlieka šifravimo dėl paties šifravimo ir nekuria skaitmeninių parašų tiesiog šiaip sau (išskyrus galbūt tyrinėtojus, kuriems kriptografijos uždaviniai yra įdomesni ir svarbesni už visus jų rezultatų taikymus). Tikrovėje informacijos slaptumas ir autentiškumas yra svarbūs įvairių praktinių uždavinių sprendimo elementai. Tie tikrovės uždaviniai yra painūs ir sudėtingi, o jų konstruktyvus sprendimas (arba sprendimo sutrukdymas) gali rūpėti daugeliui. Pavyzdžiui, reikia naudojantis kompiuteriniais tinklais pasirašyti daugiašalę sutartį arba organizuoti šifruotą ryšį tarp daugelio dalyvių, neturint galimybės jiems saugiai perduoti

raktus arba organizuoti elektroninius rinkimus arba skaitmeninių pinigų sistemą...

Kad patikimi kriptografiniai įrankiai gali visiškai neatlikti savo paskirties, gerai parodo toks pavyzdys.

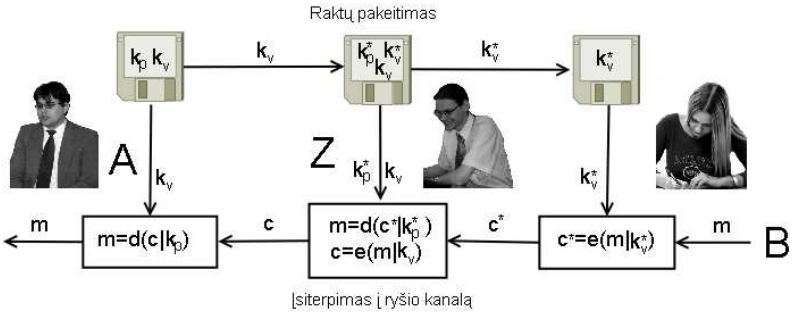
Tarkime, Algis ir Birutė nori palaikyti šifruotą ryšį, naudodamiesi kokia nors saugia viešojo rakto kriptosistema. Atrodo, kas gali būti paprasčiau: tegu A nusiunčia B šifravimui skirtą raktą $K_{e,A}$, B tegu nusiunčia raktą $K_{e,B}$ ir problema išspręsta! Galima saugiai susirašinėti. Tačiau prisiminkime Z, kurio galimybės kontroliuoti ryšio kanalą yra neribotos. Jis taip pat gali sudaryti tos pačios kriptosistemos raktų porą $K = \langle K_{e,Z}, K_{d,Z} \rangle$ ir pasielgti taip: perėmęs A siunčiamą raktą $K_{e,A}$ jis gali jį pakeisti savo raktu $K_{e,Z}$. Taigi užuot gavusi Algio raktą $K_{e,A}$, Birutė gaus $K_{e,Z}$. Galimybės nustatyti, kad raktas pakeistas nėra! Z taip pat gali pakeisti ir B siunčiamą raktą. Kai bus pasikeista raktais, A ir B bandys užmegzti „saugų“ ryšį. Tarkime, B ketina pasiųsti A šifruotą pranešimą M. Ji šifruoja: $C = e(M|K_{e,Z})$, ir siunčia A, bet pirmiausia laiškas patenka Z, kuris jį išsišifruoja ir vėl užšifravęs siunčia A:

$$M = d(C|K_{d,Z}), \quad C' = e(M|K_{e,A}).$$

A irgi išsišifruoja ir skaito laišką:

$$M = d(C'|K_{d,A}),$$

nė nenumanydamas, kad jis nebe pirmasis skaitytojas.



Įsiterpimo (man-in-the middle, angl.) ataka

Šis būdas įveikti kriptografinę apsaugą vadinamas „įsiterpimo“ (man in the middle, angl.) ataka. Zigmas net nebandė atakuoti patikimos kriptosistemos. Jis nelaužė šarvuotų durų, jis įlindo pro atvirą langą! Zigmas atliko ne kriptografinio algoritmo, bet naudojimosi juo taisyklių ataką.

Prieš pradėdami naudotis kriptosistema A ir B turėjo atlikti numatytus veiksmus, kurių tikslas – pasikeisti raktais.

Nustatytų veiksmų seka, kurią atlieka du ar daugiau asmenų, norėdami pasiekti tam tikrą tikslą, vadinama protokolu. Prekės įsigijimas, sąskaitos

atidarymas banke, įsidarbinimas, egzamino laikymas... – visur vieni protokolai.

Protokolas, kuriame, atliekant veiksmus, naudojami kriptografiniai algoritmai, yra kriptografinis protokolas. Kriptografiniai protokolai skiriasi nuo kasdienį gyvenimą reguliuojančių protokolų tuo, kad dažniausiai protokolo dalyviai neturi tiesioginio sąlyčio. Jūs negalite savo partneriui, su kuriuo norite naudodamiesi kompiuteriniais tinklais pasirašyti sutartį, tiesiog pažvelgti į akis ir, pasikliovę nuojauta, nuspręsti, ar jis neketina jūsų apgauti. Apgavystės galimybė turi būti paneigta pačioje protokolo struktūroje. Jis turi būti nedviprasmiškas ir išsamus, kita vertus, jį vykdydami, dalyviai neturėtų gauti vienas iš kito daugiau žinių, nei jų reikia numatytam tikslui pasiekti.

Kurti protokolus ir analizuoti jų patikimumą nėra lengva. Apskritai lengviau nustatyti, kad naudojamas protokolas nėra patikimas, nei įrodyti patikimumą. Juk saugumui paneigti pakanka nurodyti vieną spragą, o teigti, kad trūkumų nėra, nes jų nepastebėta – netikusi logika.

Kuriant protokolą, formuluojamas tikslas, kurį tuo protokolu siekiama pasiekti, nustatomi dalyviai, jų funkcijos ir tarpusavio ryšiai. Beveik visada reikia įvertinti grėsmes, kurios A ir B vykdomam protokolui kyla dėl Z egzistavimo. Kartais į protokolą įtraukiami tretieji pasiekti norimą tikslą padedantys asmenys.

Pavyzdžiui, jeigu A ir B gali pasinaudoti nesuinteresuoto ir patikimo trečiojo asmens J (Justo) pagalba, tai „įsiterpimo“ atakos pasikeičiant viešaisiais raktais galima išvengti. Tarkime, A ir B gali kreiptis į patikimą asmenį, turintį skaitmeninio parašo schemą. Tada A, norėdamas pasiųsti B savo viešąjį raktą ir norėdamas išvengti pakeitimo, gali pasiųsti J laišką su prašymu pasirašyti $K_{e,A}$. Gavęs iš J parašą ir jį patikrinęs

$$y = sig(K_{e,A}|K_{p,Z}), \quad ver(K_{e,A}, y|K_{v,Z}) = 1,$$

A gali siųsti B raktą $K_{e,A}$ kartu su parašu $y = sig(K_{e,A}|K_{d,Z})$ nebebijodamas, kad raktas bus pakeistas ir tai liks nepastebėta.

Tačiau trečiųjų asmenų pagalba gali brangiai kainuoti arba būti tiesiog neįmanoma. Geriausi protokolai yra tie, kurie įvairių apgavysčių galimybę panaikina dėl vidinės numatytų veiksmų struktūros.

13.4. Kriptosistemų saugumas

Vykdyti kriptografinio algoritmo kriptoanalizę reiškia atlikti jo atakas. Kokios tos atakos gali būti? Ką reiškia teiginys, kad kriptosistema yra saugi?

Teisėtas šifro C gavėjas dešifruoja jį, naudodamasis dešifravimui skirtu raktu K_d. Zigmas nežino šio rakto, todėl bando atkurti pradinį tekstą, naudodamasis šifru ir žiniomis apie naudojamą kriptosistemą. Kitaip tariant,

jis atlieka jos kriptoanalizę. Kuo jis gali naudotis? Žiniomis apie šifravimo ir dešifravimo operacijų struktūrą, kriptosistemos darbo duomenimis: šifrais ir juos atitinkančiais tekstais. Viskuo, išskyrus slaptuosius parametrus – raktus. Kriptoanalizės tikslas – naudojantis šifrais, atkurti juos atitinkančius tekstus, o galbūt – netgi nustatyti ir raktus.

Skaitmeninių parašų sistemos kriptoanalizė – tai būdų sudaryti galiojančius parašus nežinant privačiųjų raktų paieška, o taip pat – parašams sudaryti naudojamų raktų nustatymo galimybių analizuojant tekstus ir galiojančius jų parašus tyrimas.

Panagrinėkime kriptosistemų atakų rūšis. Jos skiriasi atakai naudojamų duomenų pobūdžiu. Tačiau visais atvejais daroma prielaida, kad kriptoanalitikas žino bendrąją kriptosistemos struktūrą: kaip veikia šifravimo ir dešifravimo algoritmai, kokia naudojamų raktų aibė.

Jeigu naudojamasi tik šifrais, tai ataka vadinama

- *pavienių šifrų ataka (ciphertext-only attack).*

Tokią ataką galime suformuluoti kaip uždavinį:

Duota: $C_1 = e(M_1|K_e), \ldots, C_i = e(M_i|K_e)$.

Rasti: arba $M_1, \ldots, M_i$ ir K_d, arba algoritmą, kuris bet kokiam $C_{i+1} = e(M_{i+1}|K_e)$ surastų M_{i+1}.

Jeigu pavyko gauti ne tik šifruotų, bet ir juos atitinkančių pradinių tekstų, kriptoanalitikas atlieka

- *teksto-šifro porų ataką (known-plaintext attack).*

Duota: $M_1, C_1 = e(M_1|K_e), \ldots, M_i, C_i = e(M_i|K_e)$.

Rasti: K_d arba algoritmą, kuris bet kokiam $C_{i+1} = e(M_{i+1}|K_e)$ nustatytų M_{i+1}.

Šifravimą mūsų laikais vykdo, žinoma, kompiuteriai. Gali būti, kad Zigmas turi galimybę pateikti šifravimo sistemai kokius nori tekstus ir gauti atitinkamus šifrus. Tada jis gali atlikti rimtą kriptosistemos išbandymą –

- *pasirinktų teksto-šifro porų ataką (chosen-plaintext attack).*

Duota: $M_1, C_1 = e(M_1|K_e), \ldots, M_i, C_i = e(M_i|K_e)$; čia pranešimus $M_1, \ldots, M_i$ pasirinko pats kriptoanalitikas.

Rasti: K_d arba algoritmą, kuris bet kokiam $C_{i+1} = e(M_{i+1}|K_e)$ nustatytų M_{i+1}.

Tačiau Zigmas gali turėti dar daugiau galimybių. Vieną kartą atlikęs pasirinktų teksto-šifro porų ataką, jis gali, atsižvelgdamas į kriptoanalizės rezultatus, pasirinkti naujus pranešimus ir vėl gauti jų šifrus. Ši ataka vadinama

- *adaptyvia pasirinktų teksto-šifro porų ataka (adaptive-chosen-plaintext attack).*

Kartais įmanoma ir tokia ataka: kriptoanalitikas pasirenka šifrus ir gauna juos atitinkančius pranešimus, tai –

- *pasirinktų šifrų ataka (chosen-ciphertext attack).*

Pavyzdžiui, jis gali įvairius pranešimus dešifruoti parašui tikrinti skirtu raktu ir bandyti gauti netiesioginės informacijos apie parašams sudaryti naudojamą raktą.

Kaipgi galima vertinti kriptosistemos saugumą? Yra keli požiūriai, ką laikyti saugia kriptosistema.

Kriptosistema vadinama **besąlygiškai saugia** (*unconditional security*), jei net ir turėdamas beribius skaičiavimo išteklius kriptoanalitikas negali be rakto iš šifro nustatyti, koks pranešimas buvo siųstas. Tai griežčiausias saugios kriptosistemos apibrėžimas.

Kriptosistema vadinama **saugia sudėtingumo teorijos požiūriu** (*complexity-theoretic security*), jei jos negali įveikti Zigmas, kurio skaičiavimo resursai leidžia jam taikyti tik polinominio laiko algoritmus (t. y. kai naudojamas laikas ir atmintis polinomiškai priklauso nuo įvedamų duomenų dydžio).

Sakoma, kad kriptosistemos **saugumas yra įrodomas** (*provable security*), jeigu galima įrodyti, kad sistemos įveikimas yra tolygus matematinio (dažniausiai skaičių teorijos) uždavinio, kuris laikomas sunkiu, sprendimui.

Kriptosistema vadinama **skaičiavimų požiūriu saugia** (*computational security*), jeigu pasiektas skaičiavimų resursų lygis yra pernelyg žemas, kad naudojant geriausias žinomas atakas, sistema būtų įveikta.

Pagaliau **ad hoc saugia**, arba euristiškai saugia, kriptosistema vadinama tokia sistema, kurios saugumą patvirtina tam tikri dažnai euristiniai argumentai. Suprantama, šis terminas tereiškia, kad specialistai atliko tam tikrą sistemos analizę, tačiau įveikti kriptosistemos nepavyko.

Įvertinti kriptosistemos saugumą – sudėtinga užduotis. Jokių bendrų receptų jai spręsti nėra. Kriptografo žinių, patirties ir talento niekas neatstos. Kriptografija yra modernus mokslas, tačiau, kaip ir senaisiais laikais, ir menas.

14 Klasikinė kriptografija

Kaip paslėpti teksto

$$M = m_1 m_2 \dots m_n, \quad \text{čia } m_i \in \mathcal{A} \text{ yra abėcėlės simboliai arba žodžiai,}$$

prasmę, išsaugant galimybę ją atkurti? Galima simbolius perstatyti vietomis, galima juos pakeisti kitais.

Tegu, pavyzdžiui, $\sigma : \{1, 2, \dots, n\} \to \{1, 2, \dots, n\}$ yra elementų perstata (abipus vienareikšmis atvaizdis). Naudodami ją kaip raktą, šifravimo operaciją galime apibrėžti taip:

$$e(M|\sigma) = m_{\sigma(1)} m_{\sigma(2)} \dots m_{\sigma(n)}.$$

Kitas būdas sudaryti šifrą – panaudoti keitinį. Jeigu $\mathcal{B}$ yra kokios nors kitos abėcėlės simbolių arba žodžių aibė ir $\lambda : \mathcal{A} \to \mathcal{B}$ yra injektyvus atvaizdis, tai, naudodamiesi juo kaip raktu, galime šifruoti taip:

$$e(M|\lambda) = \lambda(m_1)\lambda(m_2) \dots \lambda(m_n).$$

Galima teigti, kad kriptografijos istorija – tai tinkamų šifravimui perstatų ir keitinių konstravimo istorija.

Banaloka, žinoma, mintis! Reiškia maždaug tiek pat kaip teiginys, kad kompiuterių raida – tai tobulesnių skaičiavimo priemonių kūrimo istorija.

14.1. Skytalė ir kitos perstatos

Plutarchas savo „Įžymiųjų žmonių gyvenimuose“ rašydamas apie Spartos karaliaus Lisandro gyvenimą, mini, kad spartiečiai, norėdami jį parsikviesti iš karo žygių, pasiuntė skytalę. Skytalė – pirmasis istorijoje žinomas šifravimo įrenginys, kartu ir šifras.

Skytalė graikiškai reiškia lazdelę. Tačiau tuo pačiu žodžiu graikai vadino ir ilgą siaurą odinę juostelę su užrašytu tekstu. Rašydavo ant tokios juostelės taip: užvyniodavo ją ant lazdelės, kad sluoksniai būtų šalia vienas kito, ir rašydavo ant jos eilutė po eilutės vis pasukdami lazdelę. Užrašę visą tekstą, juostelę nuvyniodavo. Ant jos galėjai matyti pabiras raides, išdėstytas tarsi be jokios tvarkos. Jei skaitytum jas iš eilės – tik veltui gaištum laiką. Kad išryškėtų prasmė, juostelę reikia užvynioti ant tokio pat skersmens lazdelės.

Graikų istorikas Herodotas savo raštuose mini spartiečių šifravimo įrenginį – skytalę. Spartiečių skytalė – tai tiesiog lazdelė, ant kurios užvyniojama odos arba papiruso juostelė. Siaurą popieriaus juostelę užvynioję ant pieštuko, galite išbandyti, kaip toks prietaisas veikia.

Taigi skytalė yra šifravimo įrenginys. Siuntėjas ir gavėjas turi turėti po tokio pat skersmens lazdelę – tai šios šifravimo sistemos raktai. Naudodamasis raktu, siuntėjas išbarsto savo pranešimo raides išilgai juostelės – šifruoja pranešimą. Tekstas, kurį gautume nuosekliai skaitydami juostelėje užrašytas raides nuo pradžios iki galo, yra pranešimo šifras. Šifro gavėjas, naudodamasis savo raktu, vėl surenka šifro raides taip, kad išryškėtų siųstasis pranešimas – dešifruoja tekstą.

Ši šifravimo sistema sugalvota daugiau kaip prieš du tūkstančius metų. Tačiau ir dabartines šifravimo sistemas – jas pavadinome kriptosistemomis – sudaro tos pačios dalys: nešifruoti pranešimai, raktai ir šifrai, kurie gaunami, pagal tam tikras taisykles (algoritmus) pertvarkant pranešimus taip, kad nebūtų aiški jų prasmė. Panagrinėkime, kaipgi veikia skytalės kriptosistema. Visai nebūtina ieškoti lazdelės ir popieriaus juostos. Panagrinėję piešinį, greitai suprasime, kaip ji veikia.

M	Ū	S	Ų	Ž	E	M	Ė	J	A	U	N	A	D	A	R
B	U	S	I	R	P	A	S	M	U	S	E	L	D	O	R
A	D	O	A	L	D	O	R	I	J	O	A	D	R	I	J
O	A	D	A		K	A	Z	Y	S	B	I	N	K	I	S

Kai šifruojame naudodami skytalę, tarsi rašome tekstą į lentelę eilutė po eilutės, o šifrą sudarome skaitydami stulpelį po stulpelio: MBAOŪUDA...

Jeigu teksto raides rašysime į stačiakampės lentelės eilutes, o skaitysime stulpelis po stulpelio, tai gausime skytalės šifrą. Dešifruodami turime elgtis priešingai – šifro simbolius rašyti į stulpelius, o skaityti eilutę po eilutės.

Taigi skytalė apibrėžia tam tikrą teksto simbolių išbarstymo taisyklę, kurią galima paaiškinti naudojant lentelę. Tokią taisyklę nesunku apibendrinti. Pavyzdžiui, surašius tekstą į $m \times n$ matavimų lentelę (matricą) eilutė po eilutės, galima susitarti, kad šifras bus gaunamas perskaitant lentelę stulpelis po stulpelio tam tikra tvarka, kurią turi žinoti abu šifro vartotojai – šifruotojas ir dešifruotojas. Kriptografijos istorijoje buvo naudota įvairiausių tokios rūšies taisyklių. Šias operacijas galima matematiškai užrašyti pasinaudojant matricų veiksmais.

Štai pavyzdys. Tarkime, šifruojama taip: tekstas M surašomas į 3×4 matavimų matricą eilutė po eilutės, o tada skaitant stulpelis po stulpelio, surašoma į 4×3 matavimų matricą C, pirma perskaitant antrąjį, tada trečiąjį, paskui – ketvirtąjį ir pirmąjį stulpelius. Taigi tokio šifro raktas – stulpelių numerių seka $2 - 3 - 4 - 1$.

Tegu M yra pradinė teksto lentelė, C – šifras (4×3 matavimų lentelė), o K – 4×4 matavimų matrica, nusakanti M stulpelių skaitymo tvarką (šifro raktas):

$$K = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

Galime įsivaizduoti, kad teksto simboliai pakeisti sveikaisiais skaičiais, pavyzdžiui, simbolių eilės numeriais abėcėlėje. Tada

$$C = e(M|K) = K \cdot M^T,$$

čia M^T yra transponuota matrica; nulinio elemento ir simbolio sandauga laikoma lygia nuliui, o daugyba iš vieneto simbolio nekeičia.

Prisiminę matricų veiksmų savybes, galime ir dešifravimo operaciją užrašyti matematiškai:

$$M = d(C|K) = C^T (K^{-1})^T.$$

Žinoma, praeities laikų kriptografai matricų nedaugino. Pagrindiniai jų įrankiai – pieštukas ir popieriaus lapas, ant kurio rašydami raidžių virtines ieškodavo prasmės. Todėl senieji šifrai kriptografijos literatūroje dažnai ir vadinami „popieriaus ir pieštuko“ šifrais.

Štai dar vienas toks šifras, paremtas perstatomis. Jis nepelnytai vadinamas Fleissnerio kvadratų šifru, nes buvo aprašytas austrų armijos kapitono E. Fleissnerio (1843–1874) knygelėje. Tačiau iš tiesų jis buvo naudotas ir anksčiau.

Kapitono Fleissnerio idėja paprasta. Tarkime, mūsų pranešimą sudaro 16 simbolių. Jam užšifruoti pasigaminkime šešiolikos langelių kvadratą, išpjaukime jame keturis langelius (ne bet kaip!) ir prismeikime smeigtuku per centrą prie popieriaus lapo. Įrašę pirmąsias pranešimo raides į išpjautus lagelius, pasukime kvadratą 90° kampu prieš laikrodžio rodyklę. Jeigu langelius tinkamai išpjovėme, jie atidengs tuščias popieriaus vietas. Į jas vėl įrašykime keturias raides ir vėl pasukime kvadratą 90° kampu prieš laikrodžio rodyklę. Pasukę kvadratą paskutinį kartą, įrašysime likusias keturias pranešimo raides. Nuėmę kvadratą, ant popieriaus pamatysime į kvadratą surašytas pranešimo raides. Tai ir yra šifras. Galime jį pasiųsti nurašę raides eilutė po eilutės. Kad gavėjas galėtų iššifruoti šifrą, jis privalo turėti tokį pat kvadratą, kokiu naudojosi šifruotojas.

			L
A			
	U		
		K	

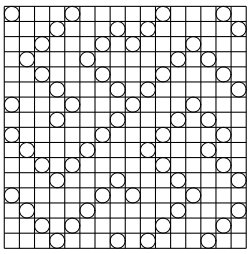
S			L
A			I
	U	M	
	E	K	

S	N		L
A		A	I
	U	M	U
J	E	K	

S	N	I	L
A	E	A	I
N	U	M	U
J	E	K	Ų

Perskaitę „užpildyto“ kvadrato eilutes, gauname tokį šifrą: SNILAEAI...

Žinoma, tokiam šifravimui galime naudoti ir didesnį kvadratą. Kiekviename $2n \times 2n$ langelių kvadrate galime taip išpjauti n langelių, kad, tris kartus $90°$ kampu pasukus kvadratą apie centrą, atsidengtų vis naujos popieriaus lapo vietos.



XVIII amžiuje naudoto šifro šablonas.

Nors šiam šifravimo būdui prigijo Fleissnerio vardas, idėja pernelyg paprasta, kad nebūtų kilusi ir kitiems. Dviem olandų tyrinėtojams pavyko iššifruoti XVIII amžiaus vidurio šifrą, rastą Olandijos valdytojo archyvuose; iššifruoti pavyko sudarius 16×16 dydžio Fleissnerio kvadratą, pavaizduotą brėžinyje. Taigi Fleissnerio sugalvotas šifras jau buvo naudotas šimtmečiu anksčiau![6]

Perstatas naudojantys šifravimo algoritmai nekeičia raidžių dažnių: ir nešifruotame tekste, ir jo šifre tos pačios raidės pasitaiko vienodai dažnai. Taigi sudarius pakankamai ilgo šifro raidžių dažnių lentelę ir lyginant ją su įvairių kalbų raidžių dažnių lentelėmis, galima ne tik nustatyti, kad šifravimui naudota perstata, bet ir sužinoti, kokia kalba parašytas pradinis nešifruotas tekstas.

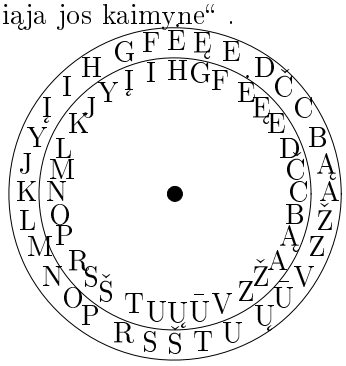
Nors perstatų naudojimas šifravimui labai sena idėja, ji praverčia ir šiandien konstruojant modernius šifrus. Juk kompiuteriai gali perstatų operacijas atlikti labai greitai, be to – taikyti jas milžiniško dydžio duomenų kiekiams.

[6] Karl de Leuw, Hans van der Meer. A Turning Grille from the Ancestral Castle of the Dutch Stadtholders. Cryptologia, XIX, 2, 1995, p. 153–165.

14.2. Keitiniai ir šifrai

Iš romėnų rašytojo Svetonijaus, gyvenusio maždaug 70–150 m., veikalo „Dvylikos cezarių gyvenimas“ žinome, kaip Julijus Cezaris rašė šifruotus laiškus Ciceronui...

Paprastą Julijaus Cezario šifravimo būdą galima nusakyti labai paprastai: „keisk abėcėlės raidę trečiąja jos kaimyne“ .



Cezario šifras su lietuviška abėcėle. Pagal skritulių kraštus viena po kitos išrašytos visos lietuvių kalbos abėcėlės raidės. Po didžiojo skritulio abėcėlės raidėmis antrajame skritulyje surašytos raidės, kurios abėcėlėje stovi trimis pozicijomis toliau – “ trečiosios kaimynės“ . Paskutiniųjų abėcėlės raidžių kaimynės yra pirmosios abėcėlės raidės.

Tarkime, didesnysis skritulys nejuda, o mažesnysis sukiojasi apie bendrą abiejų skritulių ašį. Galime pasukti mažąjį skritulį, kad po didžiojo skritulio raide A atsidurtų kita mažojo skritulio raidė. Gausime naują Cezario šifro variantą. Kiek yra abėcėlės raidžių, tiek yra mažojo skritulio padėčių. Kiekvieną iš jų galime nusakyti skaičiumi padalų, kuriuo prieš laikrodžio rodyklę pasuktas mažasis skritulys. Taigi gauname n šifravimo algoritmų, kurių raktai yra aibės

$$\mathcal{K} = \{0, 1, \ldots, n-1\}$$

skaičiai, čia n yra raidžių skaičius abėcėlėje.

Cezario šifravimo būdą patogu apibrėžti matematiškai, pakeitus abėcėlės raides skaičiais.

Tegu $\mathcal{A} = \mathbb{Z}_n = \{0, 1, \ldots, n-1\}$ yra n raidžių turinti abėcėlė, pranešimų ir šifrų aibės sudarytos iš šios abėcėlės žodžių $\mathcal{M} = \mathcal{C} = \mathcal{A}^*$, o raktų aibė sutampa su abėcėle $\mathcal{K} = \mathcal{A}$. Tada Cezario šifravimo taisyklę galime užrašyti taip:

$$e(x|k) \equiv x + k \pmod{n},\ x \in \mathbb{Z}_n, \quad e(m_1 m_2 \ldots |k) = e(m_1|k)e(m_2|k)\ldots$$

Ši paprasta kriptosistema kriptografijoje vadinama Cezario kriptosistema. Raktų yra tiek pat kiek abėcėlės simbolių. Matematinę šifravimo algoritmo išraišką norisi apibendrinti. Imkime tokią raktų aibę:

$$\mathcal{K} = \{K = \langle k_1, k_2 \rangle : k_i \in \mathbb{Z}_n, (k_1, n) = 1\},$$

ir vieno simbolio šifrą apibrėžkime taip:

$$e(x|K) \equiv k_1 x + k_2 \pmod{n}.$$

Jeigu n yra pirminis, tai tokioje apibendrintoje Cezario kriptosistemoje bus $n \cdot (n-1)$ skirtingų raktų. Jie „valdo“ tik nedidelę dalį visų galimų keitinių $\mathbb{Z}_n \to \mathbb{Z}_n$; iš viso skirtingų keitinių yra $n!$

Cezario kriptosistemos apibendrinimu galime laikyti Lesterio Hillo šifrus, kuriuos jis paskelbė 1929 metais.[7]

Apibrėžkime Hillo kriptosistemos raktų aibę:

$$\mathcal{K}_{n,r} = \{K : K = (k_{ij}) \text{ yra } r\text{-osios eilės kvadratinė matrica } (\det(K), n) = 1\},$$

čia visi skaičia k_{ij} yra sveikieji, o $\det(K)$ žymi matricos determinantą. Dabar r ilgio žodžių šifrus galime apibrėžti taip:

$$e(m_1 m_2 \dots m_r|K) = c_1 c_2 \dots c_r, \quad c_1 c_2 \dots c_r = m_1 m_2 \dots m_r \cdot K,$$

o dešifravimo operaciją

$$d(c_1 c_2 \dots c_r|K) = m_1 m_2 \dots m_r, \quad m_1 m_2 \dots m_r = c_1 c_2 \dots c_r \cdot K^{-1}.$$

Atvirkštinę matricą K^{-1} reikia skaičiuoti, žinoma, moduliu n, o žodis daugi-namas iš matricos naudojantis vektoriaus ir matricos sandaugos apibrėžimu.

Jei $n = p$ yra pirminis skaičius, tai raktų aibėje $\mathcal{K}_{n,r}$ yra tiek matricų, kiek tiesinė erdvė $\mathbb{F}_p^r$ turi skirtingų bazių. Šį dydį nesunku apskaičiuoti.

115 teorema. *Jei p yra pirminis, tai*

$$|\mathcal{K}_{p,r}| = (p^r - 1)(p^r - p)(p^r - p^2) \cdots (p^r - p^{r-1}).$$

Jeigu šifravimas naudojant keitinį iš pradžių apibrėžiamas pavieniams abėcėlės simboliams, o šifruojant žodžius, jis taikomas kiekvienai raidei atskirai, tai pradinio teksto simbolių dažniai lygūs atitinkamiems šifro simbolių dažniams, taip pat – pradinio teksto simbolių porų dažniai lygūs atitinkamiems šifro simbolių porų dažniams ir t. t. Taigi naudojantis įvairių kalbų simbolių, jų porų, trejetų ir t. t. dažnių lentelėmis, iš šifro galima nustatyti ir kalbą, kuria buvo parašytas tekstas, ir patį keitinį.

Jeigu šifruojame (kaip Hillo šifro atveju) ne pavienius simbolius, bet m ilgio žodžius, tai ryšio tarp pavienių simbolių dažnių tekste ir šifre jau nebus, tačiau $m, 2m, \dots$ ilgio simbolių blokų dažniai bus tie patys. Žinoma, kai m yra didelis skaičius, tuos dažnius skaičiuoti ir lyginti yra sudėtinga.

Taigi kai šifruojami ne pavieniai žodžiai, bet m $(m > 1)$ ilgio žodžiai, ryšys tarp simbolių dažnių nešifruotame tekste ir šifre išnyksta. Yra dar viena keitinių, panaikinančių ryšį tarp simbolių dažnių, rūšis.

[7] L. Hill. Cryptography in an Algebraic Alphabet. The American Mathematical Monthly, **36**, 1929, June-July, p. 306–312.

Tegu $\mathcal{A}$ ir $\mathcal{B}$ yra nešifruoto teksto ir šifro abėcėlės. Abėcėlės $\mathcal{B}$ visų poaibių aibę žymėsime $\mathcal{P}(\mathcal{B})$.

Kiekvienam atviro teksto abėcėlės simboliui $a \in \mathcal{A}$ nurodysime homofonų aibę $k(a) \subset \mathcal{B}$, t. y. nurodysime skirtingai užrašomus simbolius, kuriuos šifre reikia „skaityti“ kaip a. Tokios kriptosistemos raktas yra funkcija

$$k : \mathcal{A} \to \mathcal{P}(\mathcal{B}),\ k(x) \cap k(y) = \emptyset,\ \text{jei } x \neq y,\ x, y \in \mathcal{A}.$$

Šifravimui dar reikia turėti būdą, kuriuo naudojantis, iš šifruojamo simbolio homofono būtų atsitiktinai parenkamas vienas jo elementas; visų elementų parinkimo tikimybės turėtų būti vienodos. Žymėdami atsitiktinai parinktą homofono $k(a)$ elementą $k(a, \omega)$, teksto šifrą apibrėšime taip:

$$e(m_1 m_2 \ldots | k) = k(m_1, \omega) k(m_2, \omega) \ldots$$

Naudodami homofoniniais keitiniais grindžiamus šifrus, galime panaikinti ryšį tarp atviro teksto simbolių dažnių lentelės bei šifro simbolių dažnių lentelės. Parinkdami funkciją k taip, kad homofono $k(a)$ elementų skaičius būtų proporcingas simbolio a dažniui atvirame tekste, galime pasiekti, kad šifruotame tekste visų simbolių pasirodymo dažniai būtų beveik vienodi.

Jei kriptosistemoje norime naudoti homofoninį keitinį, turime nuspręsti, kaip apibrėžti raktą, kad jį būtų patogu saugoti bei lengva juo naudotis. Manoma, kad pirmąkart homofoniniai keitiniai šifravimui buvo panaudoti 1401 metais slaptam Mantujos hercogystės ir Simeone'o de Crema susirašinėjimui. Priebalsiams buvo naudoti įprastiniai keitiniai, tačiau balsiams šifruoti – homofoniniai šifrai.

Įdomus homofoninių keitinių pavyzdys – Beale'o šifrai. Thomas Jeffersonas Beale'as buvo nuotykių ieškotojas. Jis paliko tris šifruotus tekstus apie Virdžinijos valstijos apylinkėse užkastą didelį lobį. Tai įvyko apytikriai 1820 metais. Antrąjį Beale'o šifrą, kuriame aprašomas lobis ir nurodoma, kad pirmajame šifre yra detalios instrukcijos, kaip jį surasti, 1880 metais iššifravo Jamesas Wardas. Beale'o raktas – JAV Nepriklausomybės deklaracija. Jis naudojosi šiuo tekstu taip. Įsivaizduokime, kad kokio nors dokumento ar knygos (Beale'o atveju – JAV Nepriklausomybės deklaracijos) žodžius sunumeravome. Žodžių, kurie prasideda raide „a“ numeriai tegu sudaro šios raidės homofonų aibę, žodžių, kurie prasideda raide „b“ numeriai sudaro „b“ homofonų aibę ir t. t. Jeigu norite – galite numeruoti ne žodžius, bet raides.

Puikus metodas!

14.3. Vigenere šifras

Šifras, naudojantis keitinį, keičia pradinio teksto abėcėlės $\mathcal{A}$ raides abėcėlės $\mathcal{B}$ raidėmis. Panaudoti šifrui ne vieną, bet kelias abėcėles $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_d$ – pati geriausia naujosios Europos kriptografijos idėja!

Viduramžių europiečiams kriptografija nerūpėjo. Tačiau prasidėjus naujiesiems laikams, suklestėjus pirmiausia Italijos miestams, kurie ėmė įnirtingai varžytis dėl įtakos, valdžios, naudos ir turto, atėjo metas ir politinei Europos kriptografijai. Italijos miestų valdžia turėjo savo žinioje šifrų specialistus, kurie triūsė šifruodami ir dešifruodami slaptus laiškus, tiek savo miesto, tiek svetimus – pagrobtus, užverbuotų agentų perduotus... Pavyzdžiui, Venecijoje XVI amžiaus viduryje buvo net trys šifrų sekretoriai – visas kriptografų skyrius! Kriptologijos meno buvo mokoma mokyklose, buvo rengiami kriptoanalizės konkursai, už nuopelnus dosniai atlyginama.

1555 metais kriptografo (sekretoriaus šifrų reikalams) pareigybę įsteigė ir popiežius. Apie 1580 metus popiežiams ėmė tarnauti įžymių kriptografų Argenti šeima.

Tuo laiku naudotus pranešimų prasmės slėpimo būdus galima suskirstyti į dvi grupes: kodus ir šifrus. Kodas reiškė išankstinį susitarimą keisti vienus žodžius ar frazes kitais žodžiais, frazėmis arba simboliais. Pavyzdžiui, vienname seniausių Vatikano tekstų, skirtų kriptografijai, nurodoma, kad tuometinėse popiežiaus kovose su Romos imperatoriumi dalyvavusių gelfų ir gibelinų partijas reikia vadinti egiptiečiais ir Izraelio vaikais, minint karalių, rašyti raidę A, popiežių – raidę D ir t. t. Kad gavėjas perskaitytų naudojant kodą parašytą laišką, jam turi būti iš anksto įteiktas pats kodas – naudojamų žodžių ir tikrosios jų prasmės atitikties lentelė.

Kitaip pranešimo prasmė slepiama naudojant šifrus. Iš anksto susitariama, kaip bus keičiamos laiško raidės. Jos gali būti keičiamos kitomis raidėmis, kokiais nors simboliais arba tiesiog skaičiais. Laiško gavėjas turi žinoti dviejų simbolių rinkinių atitikties taisyklę, kuria naudojamasi keičiant raides, t. y. raktą. Tokiam raktui perduoti Argenti pradėjo naudoti prasmingus žodžius (juos lengviau įsiminti!). Argenti idėja parodyta pavyzdžiu.

A	R	G	E	N	T	I	B	C	D	F	H	J
10	11	12	13	14	15	16	17	18	19	20	21	22
K	L	M	O	P	Q	S	U	V	W	X	Y	Z
23	24	25	26	27	28	29	30	31	32	33	33	34

Pirmojoje ir trečiojoje eilutėse užrašyta lotyniška abėcėlė, tačiau ne įprastine tvarka. Ji prasideda rakto žodžiu ARGENTI, o po jo surašytos likusios abėcėlės raidės eilės tvarka. Antrojoje ir ketvirtojoje eilutėse užrašyti skaičiai, kuriais šifruojant keičiamos abėcėlės raidės. Ši idėja buvo naudojama net ir po kelių šimtų metų. Štai Richardo Zorgės – žvalgo, Antrojo pasaulinio karo išvakarėse veikusio Japonijoje, – šifro pavyzdys.

S	U	B	W	A	Y
0	82	87	91	5	97
C	D	E	F	G	H
80	83	3	92	95	98
I	J	K	L	M	N
1	84	88	93	96	7
O	P	Q	R	T	V
2	85	89	4	6	99
X	Z	.	/		
81	86	90	94		

Anglų kalbos abėcėlė surašyta lentelės eilutėse pagal Argenti metodą, pradedant žodžiu SUBWAY. Frazės „a sin to er“ raidės pakeistos skaitmenimis, o likusios – dviženkliais skaičiais, pradedant 80. Taip padaryta norint pagreitinti šifro perdavimą: frazės „a sin to er“ raidės anglų kalbos tekstuose pasitaiko dažniausiai.

Apie 1466 metus Leonas Batista Alberti – žymus italų architektas, architektūros teoretikas, muzikantas... tiesiog tikras Renesanso laikų žmogus – parašė rankraštį apie kriptografiją, kurioje išdėstė itin svarbią tolimesnei europiečių kriptografijos raidai idėją.

Panašius šifravimo skritulius aprašė Leonas Batista Alberti (1404–1472) savo traktate, skirtame šiframs. Mažesnysis skritulys yra sukiojamas, taigi šifro abėcėlė yra kaitaliojama.

Prisiminkime du skritulius su abėcėlės raidėmis, kuriais naudojomės nagrinėdami Cezario šifrus. Tie skrituliai taip pat yra Alberti išradimas, aprašytas jo kriptografijos traktate. Alberti skrituliuose, žinoma, buvo užrašytos lotyniškos abėcėlės raidės; jei surašysime lietuviškos abėcėlės raides – esmė dėl to nepasikeis.

Tarkime, didesnysis skritulys nejuda, o mažesnysis gali sukiotis. Skritulių tarpusavio padėtis apibrėžia laiško raidžių keitimo (šifravimo) taisyklę: laiško raidę suradę didesniajame skritulyje, ją pakeičiame raide, kuri užrašyta po ja mažesniajame skritulyje. Naujiena yra toks Alberti pasiūlymas: „Užšifravę du ar tris laiško žodžius, mažąjį skritulį pasukime per vieną padalą ir kitus du ar tris žodžius užšifruokime naudodamiesi naująja skritulių padėtimi ir vėl pasukime mažąjį skritulį...“ Šifruojant šitokiu būdu, pasikartojančios laiško raidės jau nebėra keičiamos viena ir ta pačia raide. Kokia raide reikia pakeisti, tarkime, raidę A, priklauso nuo skritulių tarpusavio padėties. Taigi šifruojant naudojama nebe viena, bet daugelis abėcėlių. Suprantama, kad laiško siuntėjas ir gavėjas turi turėti vienodus Alberti skritulius ir būti susitarę dėl pradinės skritulių padėties ir mažojo skritulio sukiojimo taisyklės. Kaip nusakyti tokias taisykles, Alberti nenagrinėjo. Šis nedidelis veikalas, galėjęs iš esmės pakeisti visą tuometinę kriptografiją, liko neišspausdintas ir didelės įtakos nepadarė. Matyt, šifravimo su daugeliu abėcėlių idėja buvo pernelyg nauja tiems laikams.

Panaši idėja vėl išniro maždaug po šešiasdešimt metų. Ji kilo ne mažiau talentingam, tačiau visai kito būdo žmogui. Alberti mėgo pasaulietiško gyvenimo spalvas, o štai kitas Europos kriptografijos pradininkas labiau vertino vienuolyno celės ramybę ir tylą. Johannes Trithemius (1462–1516) kriptografijos istorijoje minimas kaip pirmosios išspausdintos knygos apie šifrus autorius. Daugiau kaip penkių šimtų puslapių jo knyga „Poligraphia“ buvo išleista daugelį kartų. Joje buvo aprašyta ir panaši kaip Alberti šifravimo su daugeliu abėcėlių idėja.

Trithemius šifruoti siūlė taip: pirmąją laiško raidę su pirmąja abėcėle, antrąją – su antrąja ir t.t. Taigi ir šiuo būdu šifruojant, pasikartojančios laiško raidės keičiamos įvairiai priklausomai nuo jų vietos laiške.

Būtų teisinga šifravimą naudojant daugelį abėcėlių pavadinti Alberti arba Tritemijaus vardu. Tačiau istorija susiklostė taip, kad tokiam šifrui prigijo visai kito žmogaus vardas. 1586 metais buvo išspausdinta Blezo de Vigenere knyga „Traktatas apie šifrus“ , kuriame taip pat buvo išdėstytas šifravimo su daugeliu abėcėlių būdas. Vigenere vardas kriptografijoje prigijo šifrui, kuriame naudojama keletas abėcėlių, o šifro raktas užrašomas žodžiu.

Pasirinkime kokį nors žodį, pavyzdžiui, MES. Panaudosime jį kaip Vigenere šifro raktą sakiniui „UPELĖ MUSĖ RANGOSI VIKRIAI“ .

Pasinaudosime lotyniškosios abėcėlės lentele, todėl raidę Ė keisime į E.

```
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
B C D E F G H I J K L M N O P Q R S T U V W X Y Z A
C D E F G H I J K L M N O P Q R S T U V W X Y Z A B
D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
E F G H I J K L M N O P Q R S T U V W X Y Z A B C D
F G H I J K L M N O P Q R S T U V W X Y Z A B C D E
G H I J K L M N O P Q R S T U V W X Y Z A B C D E F
H I J K L M N O P Q R S T U V W X Y Z A B C D E F G
I J K L M N O P Q R S T U V W X Y Z A B C D E F G H
J K L M N O P Q R S T U V W X Y Z A B C D E F G H I
K L M N O P Q R S T U V W X Y Z A B C D E F G H I J
L M N O P Q R S T U V W X Y Z A B C D E F G H I J K
M N O P Q R S T U V W X Y Z A B C D E F G H I J K L
N O P Q R S T U V W X Y Z A B C D E F G H I J K L M
O P Q R S T U V W X Y Z A B C D E F G H I J K L M N
P Q R S T U V W X Y Z A B C D E F G H I J K L M N O
Q R S T U V W X Y Z A B C D E F G H I J K L M N O P
R S T U V W X Y Z A B C D E F G H I J K L M N O P Q
S T U V W X Y Z A B C D E F G H I J K L M N O P Q R
T U V W X Y Z A B C D E F G H I J K L M N O P Q R S
U V W X Y Z A B C D E F G H I J K L M N O P Q R S T
V W X Y Z A B C D E F G H I J K L M N O P Q R S T U
W X Y Z A B C D E F G H I J K L M N O P Q R S T U V
X Y Z A B C D E F G H I J K L M N O P Q R S T U V W
Y Z A B C D E F G H I J K L M N O P Q R S T U V W X
Z A B C D E F G H I J K L M N O P Q R S T U V W X Y
```

Lentelę, kurioje surašytos atitinkamai paslinktos lotynų kalbos abėcėlės Tritemijus vadino „tabula recta“ .

Parašykime po šio sakinio raidėmis rakto raides taip:

```
UPELE MUSE RANGOSI VIKRIAI
MESME SMES MESMESM ESMESME
```

Šifruojame pirmąją raidę U: ieškome lentelės stulpelio, kuris prasideda raide U, ir eilutės, kuri prasideda rakto raide M; eilutės ir stulpelio sankirtoje parašyta raidė G, ja ir keičiame raidę U. Šifruodami sakinio raidę P, ieškome stulpelio, prasidedančio raide P, ir eilutės, prasidedančios raide E, jų sankirtoje yra raidė T, šifruodami raidę, naudojame lentelės eilutę, kuri prasideda raide S ir t. t. Taip gauname ir visą sakinio šifrą:

```
UPELE MUSE RANGOSI VIKRIAI
MESME SMES MESMESM ESMESME
GTWXI EGWW DEFSSLU ZAWVAMM
```

Šifravimui panaudojome tik tris „tabula recta“ eilutes – tiek, kiek yra rakto žodžio raidžių. Dešifruojant Vigenere šifrą, taip pat prireiks trijų eilučių (trijų abėcėlių). Šifruodami pirmąją raidę, naudojome taisyklę, kuri nusakoma pirmąja lentelės eilute ir eilute, prasidedančia raide M, t. y. raidė

A yra keičiama raide M. Dešifruodami pirmąją raidę, naudosime taisyklę, kuri raidę M keičia raide A. Ši taisyklė nusakoma pirmąja lentelės eilute ir eilute, kuri prasideda raide O.

Antrajai ir trečiajai šifro raidėms dešifruoti teks naudoti lentelės eilutes, prasidedančias raidėmis W ir I. Taigi dešifruodami galime elgtis panašiai kaip šifruodami, tik vietoj rakto MES turime naudoti raktą OWI:

```
GTWXI EGWW DEFSSLU ZAWVAMM
OWIOW IOWI OWIOWIO WIOWIOW
UPELE MUSE RANGOSI VIKRIAI
```

Vigenere kriptosistemą matematiškai galime apibrėžti visai panašiai kaip Cezario. Tegu $\mathcal{A} = \mathbb{Z}_n = \{0, 1, 2, \ldots, n-1\}$, $\mathcal{M} = \mathcal{C} = \mathcal{A}^*, \mathcal{K} = \mathcal{A}^d$. Šifravimo algoritmas su raktu $k = k_1 k_2 \ldots k_d$ veikia taip:

$$e(m_1 m_2 \ldots m_d m_{d+1} \ldots | k) = c_1 c_2 \ldots c_d c_{d+1} \ldots,$$
$$c_1 \equiv m_1 + k_1 \pmod{n},\ c_2 \equiv m_2 + k_2 \pmod{n},\ \ldots, c_d \equiv m_d + k_d \pmod{n},$$
$$c_{d+1} \equiv m_{d+1} + k_1 \pmod{n},\ \ldots$$

Taigi galime įsivaizduoti, kad Vigenere šifras gaunamas suskaidžius pradinį tekstą į d fragmentų (d – naudojamo rakto ilgis) ir kiekvieną fragmentą užšifravus atskiru Cezario šifru.

14.4. Vigenere šifro analizė

Vigenere šifras ilgai buvo laikomas tiesiog neįveikiamu. Tačiau kai šifro raktas trumpas, visai paprasta idėja padeda jį įveikti.

Šifrai su keletu abėcėlių iš pradžių buvo naudoti retai. Alberti, Tritemijaus ir Vigenere laikais slaptaraščiuose vyravo kodai. Šifrai su daugeliu abėcėlių buvo pernelyg sudėtingi kasdieniam naudojimui. Vartyti kodų knygas ir užrašinėti žodžius to meto slaptų laiškų rašytojams ir skaitytojams atrodė paprasčiau.

Tačiau tie, kas išmanė, kaip Vigenere šifrai veikia, pripažino, kad jie saugūs, netgi neįveikiami. Net XX amžiaus pradžioje buvo manoma, kad jeigu raktas nežinomas, tai nėra būdo jiems įminti. Tai buvo netiesa. Nedidelėje 1863 metais išleistoje knygelėje prūsų armijos majoras Kassiskis išdėstė itin paprastą Vigenere šifro analizės metodą, kuriuo dažnai įmanoma iššifruoti šifrą ir be rakto. Tačiau Kassiskio idėją mažai kas įvertino. Pats Kassiskis irgi nesiekė žymaus kriptologo šlovės. Tarnaudamas Prūsijos armijoje, laisvalaikiu gilinosi į kriptografiją, tačiau, parašęs minėtą 95 puslapių knygelę *„Die Geheimschriften und die Dechiffrir-kunst“*, nustojo ja domėtis ir, išėjęs į atsargą, atsidėjo archeologijai.

Kokį gi Vigenere šifro analizės būdą sugalvojo Kassiskis?

Tarkime, Vigenere raktą sudaro d skirtingų simbolių. Tada pirmąją teksto raidę šifruojame naudodami vieną abėcėlę („tabula recta“ eilutę),

pažymėkime ją A_1, antrąją raidę – naudodami abėcėlę A_2, ... d-ąją – naudodami abėcėlę A_d, $d+1$-ąją – naudodami abėcėlę A_1 ir t. t. Taigi galime įsivaizduoti, kad tekstą $M = m_1 m_2 \ldots$ sudaro d dalių $M_1, M_2, \ldots, M_d$

$$\begin{aligned} M_1 &= m_1 m_{1+d} m_{1+2d} \ldots \\ M_2 &= m_2 m_{2+d} m_{2+2d} \ldots \\ &\ldots\ldots \\ M_d &= m_d m_{2d} m_{3d} \ldots, \end{aligned}$$

o kiekviena dalis šifruojama naudojantis atskira abėcėle.

Kriptoanalitikas nežino nei rakto, nei jį sudarančių simbolių skaičiaus (rakto ilgio), tačiau turi ilgoką šifrą $C = c_1 c_2 \ldots$

Jeigu jis žinotų rakto ilgį d, galėtų turimą šifrą suskaidyti į d fragmentų

$$\begin{aligned} C_1 &= c_1 c_{1+d} c_{1+2d} \ldots \\ C_2 &= c_2 c_{2+d} c_{2+2d} \ldots \\ &\ldots\ldots \\ C_d &= c_d c_{2d} c_{3d} \ldots \end{aligned}$$

ir, naudodamasis raidžių dažnių lentelėmis, dešifruotų kiekvieną paprastu keitinių šifru užšifruotą fragmentą.

Taigi raktas, kuris „atrakina“ šifro raktą, yra jo ilgis! Kaip jo ieškoti? Į šį klausimą Kassiskis atsakė labai paprastai: ieškokite šifre pasikartojančių fragmentų!

Patyrinėkime paprastą šiek tiek dirbtinį pavyzdį. Šifruosime tekstą Vigenere šifru, kurio raktas yra šešių raidžių žodis LAUKAS:

```
PAVARGĘS VASARIS PUSNYNUOSE MIEGA IR VANDENYS SLŪGSO UŽŠALĘ
LAUKASLA UKASLAU KASLAUKASL AUKAS LA UKASLAUK ASLAUK ASLAUK
```

Žvilgtelėkime į pirmąjį ir antrąjį šifruojamo teksto žodžius. Juose yra tas pats skiemuo VA, o po juo abiejuose žodžiuose parašytos tos pačios rakto raidės UK. Vadinasi, ir šifre gausime du vienodus iš dviejų raidžių sudarytus fragmentus. Atstumas tarp jų lygus 6, t. y. rakto ilgiui! Taigi kriptoanalitikas, pastebėjęs šifre šiuos vienodus fragmentus ir suradęs, kiek raidžių juos skiria, sužinotų rakto ilgį! Žinoma, analizuodami tikrą šifrą, tokios lengvos sėkmės negalėtume tikėtis. Rastume daug vienodų šifro fragmentų, atstumai tarp jų irgi būtų įvairūs. Tačiau vis tiek gana dažnai pasitaikytų rakto ilgio kartotiniai!

Panagrinėkime pavyzdį. Užšifravę pačią pirmąją šio skyrelio skiltį Vigenere šifru su raktu VIETA (naudojama lietuviška 32 simbolių abėcėlė), gausime tokį šifrą:

PŠJJA FCAFE YMŽNA ŽOFĄL FĘMLP NIHŠI ŠYARO KIAZO RŠŪŽT VŠEGB
CBŽDT NŠŽŽM FŲENS FBCDG CŽIJE YIMFA FCVGA MDEĖA JMVNS FBELI
KOPDM CLUHI KICĮK LLEDŠ FPŪTI OEHTU ĘMSDU VYYŪĖ YŠĄŲU UAŲŽR
KMSĘG OEHĄT FŽKDK VCHDE KŠEHN VEHĮJ FZADV VBŽĘT FŪUZŲ ĮŽOCA
OŠŪNŽ NIZDN DDMŠO BHMNS RAŠŽT LCSTP RĘSTI PŪĄJA PUŽĮJ VZVDR
OŪEDT HDUĖA JCEMR LLYYA MBEKČ FIA

Suradę vienodas simbolių poras (digramas) šifre, nustatę atstumus tarp jų, o tada suskaičiavę, kiek atstumų dalijasi atitinkamai iš $2, 3, \ldots, 10$, sudarytume tokią lentelę:[8]

Dalikliai	2	3	4	5	6	7	8	9	10
Kiek atstumų dalijasi	27	23	10	30	15	1	6	7	10

Skaičiai iškalbingi – rakto ilgis be abejonės lygus 5.

14.5. Friedmano kappa testas

> *Neretai senųjų amžių matematikai garsėjo ir kaip geri kriptografai: F. Viète, J. Walis... Tačiau senųjų laikų kriptografija buvo veikiau menas, nei mokslas. Bet XX amžiuje jos ryšiai su matematika darėsi vis glaudesni ir glaudesni...*

Kassiskio testą vargu ar pavadintume matematiniu kriptoanalizės metodu. Matematikos jame tiek ir tėra – atstumų tarp fragmentų skaičiavimas ir daliklių nagrinėjimas.

Pirmasis matematinius metodus kriptoanalizei pradėjo taikyti Williamas Friedmanas (1891–1969). Jį kriptografijos istorikai pelnytai vadina vienu didžiausių visų laikų kriptoanalitikų. Jo žmona – Elizebeth Friedman irgi buvo žymi kriptografė. Kriptografija šie žmonės susidomėjo atlikdami gana keistus tyrimus turtingo amerikiečio Fabyano įkurtame Riverbanko tyrimų centre. Jie tyrinėjo, ar didžiojo Šekspyro kūrinių autorius kartais nėra tų laikų žymus filosofas Backonas.

JAV įsitraukus į Pirmąjį pasaulinį karą prireikė kriptoanalitikų pagalbos. Tokių specialistų JAV kariniuose sluoksniuose nebuvo. Užtat jų buvo Riverbanke. Šitaip iš Šekspyro raštų tyrinėtojo W. Friedmanas virto vienu žymiausių JAV kriptografijos specialistų.

Kriptoanalizės metodą, kurį šiame skyrelyje panagrinėsime, W. Friedmanas išdėstė savo veikale „Sutapimų indeksas“ („The Index of Coincidence“). Panagrinėsime, kaip jį galima taikyti Vigenere šifrui. Pats W. Friedmanas jį naudojo ir kitokių šifrų analizei.

Tarkime, teksto ir šifro abėcėlę $\mathcal{A}$ sudaro n simbolių (lietuvių kalbos abėcėlėje $n = 32$):

$$\mathcal{A} = \{a_1, a_2, \ldots, a_n\}.$$

[8] Vigenere šifrų analizei galite pasinaudoti Java įskiepiais šios knygos tinklalapyje.

Jeigu iš šios abėcėlės su grąžinimu rinktume dvi raides, tai tikimybė, kad abi raidės būtų a_1 lygi $\frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$, kad a_2 – tokia pati ir t. t. Taigi dydį

$$\kappa_0 = \frac{1}{n^2} + \frac{1}{n^2} + \ldots + \frac{1}{n^2} = \frac{1}{n}$$

galime suvokti kaip tikimybę gauti dvi vienodas raides, kai jas atsitiktinai su grąžinimu renkame iš abėcėlės. Vadinsime šį dydį atsitiktinio teksto sutapimų indeksu.

Pažymėkime $p_1, p_2, \ldots, p_n$ abėcėlės raidžių pasitaikymo tikimybes kokia nors kalba parašytame tekste.

Jeigu dabar dvi raides atsitiktinai rinktume iš kokio nors tikro teksto, tai tikimybė gauti vienodas būtų

$$\kappa_t = p_1^2 + p_2^2 + \ldots + p_n^2.$$

Šį skaičių vadinsime teksto sutapimų indeksu. Pavyzdžiui, lietuvių kalbos teksto sutapimų indeksas yra $\kappa_t \approx 0,069$.

Jeigu sugretintume surašydami į dvi eilutes du skirtingus, bet to paties ilgio N ir ta pačia kalba parašytus tekstus – kiek stulpelių iš vienodų raidžių gautume? Stulpelių iš vienodų raidžių būtų

$$\approx N \cdot \kappa_t.$$

O jeigu sugretintume dviejų tokių tekstų šifrus, gautus panaudojus vieną keitinį? Vienodų stulpelių skaičius būtų maždaug tas pats. O jeigu abu tekstai būtų užšifruoti tuo pačiu Vigenere šifru? Irgi tas pats! O jeigu skirtingais Vigenere šifrais? Nieko konkretaus negalime pasakyti, tačiau galima tikėtis, kad vienodų raidžių stulpelių būtų

$$\approx N \cdot \kappa_0.$$

Šiais samprotavimais ir remiasi Friedmano kappa testas.

Įsivaizduokime, kad kriptoanalitikui įteiktas Vigenere šifras, kuris buvo sudarytas naudojant, tarkime, raktą ABC, kurio kriptoanalitikas, aišku, nežino. Tarkime, tas šifras yra toks: $C = c_1c_2c_3c_4c_5c_6\ldots$ Galime įsivaizduoti, kad jis gautas naudojant raktą $abcabcabc\ldots$ Jeigu kriptoanalitikas nubrauktų pirmąjį šifro C simbolį, gautų šifrą, kuris sudarytas naudojant raktą $bcabcabc\ldots$ Jeigu jis suskaičiuotų, kiek yra vienodų raidžių stulpelių tokioje lentelėje

$$\begin{matrix} c_1 & c_2 & c_3\ldots \\ c_2 & c_3 & c_4\ldots \end{matrix},$$

matyt, gautų, kad jų yra maždaug κ_0 nuo viso stulpelių skaičiaus. Tą patį rezultatą gautų ir iš lentelės

$$\begin{matrix} c_1 & c_2 & c_3\ldots \\ c_3 & c_4 & c_5\ldots \end{matrix}.$$

O štai skaičiuodamas vienodų raidžių stulpelių dalį lentelėje

$$\begin{matrix} c_1 & c_2 & c_3 \dots \\ c_4 & c_5 & c_6 \dots \end{matrix},$$

jis turėtų gauti reikšmę, artimą skaičiui κ_t. Tada jis galėtų padaryti išvadą, kad rakto ilgis yra tikriausiai lygus 3.

Išbandykime kappa testą. Prisiminkime, kaip sėkmingai pavyko įspėti rakto ilgį, naudojantis Kassiskio testu. Dabar tam pačiam šifrui analizuoti pasinaudokime Friedmano metodu: sugretinkime pradinį šifrą ir šio šifro dalį, gautą atmetus pirmuosius d simbolių ($d = 1, 2, \dots$) ir apskaičiuokime, kokią dalį sudaro vienodų raidžių stulpeliai. Suapvalinę iki tūkstantųjų, gausime tokius skaičius:

Poslinkiai $d =$	1	2	3	4	5	6	7
Sutapimai	0,032	0,029	0,04	0,033	0,055	0,026	0,022

Taigi ir kappa testas rodo, kad rakto ilgis turėtų būti lygus 5.

Naudojantis sutapimų indeksais, netgi galima išvesti apytikslę Vigenere šifro rakto ilgio formulę.

Tarkime, pavyko sugauti N ilgio žinoma kalba parašyto teksto šifrą

$$C = c_1 c_2 \dots c_N.$$

Žinome sutapimų indeksus κ_0, κ_t. Tačiau šifro raidžių dažnių lentelė skiriasi nuo nešifruoto teksto. Kaip surasti šifro sutapimų indeksą κ_c, t. y. tikimybę, kad, atsitiktinai iš šifro rinkdami dvi raides, gausime vienodas? Šį dydį suskaičiuosime dviem būdais: naudodamiesi gautuoju šifru ir tikimybiniais samprotavimais.

Tegu simboliai $a_1, \dots, a_n$ pasikartoja šifre atitinkamai $m_1, \dots, m_n$ kartų. Įsivaizduokime, kad atsitiktinai pasirenkame du gauto šifro simbolius. Tikimybę, kad gausime du vienodus, galime skaičiuoti taip:

$$\kappa_c = \frac{1}{C_N^2} \sum_{i=1}^{n} C_{m_i}^2.$$

Dabar padarykime prielaidą, kad šifravimui panaudotas d ilgio raktas. Suskaidysime šifrą į d fragmentų:

$$\begin{matrix} c_1 & c_{1+d} & c_{1+2d} & \dots \\ c_2 & c_{2+d} & c_{2+2d} & \dots \\ \vdots & \vdots & \vdots & \vdots \\ c_d & c_{2d} & c_{3d} & \dots \end{matrix}.$$

Tarsime, kad N yra pakankamai didelis skaičius, tad kiekviename fragmente yra $\approx N/d$ simbolių. Kiekvienas fragmentas – Cezario šifras, tad galime manyti, kad tikimybė, jog du simboliai, parinkti iš to paties segmento, sutampa, lygi atviro teksto sutapimų indeksui κ_t. Dabar įsivaizduokime, kad atsitiktinai pasirenkame x, y – du gauto šifro simbolius. Skaičiuosime tikimybę, kad jie sutampa. Iš pradžių pažymėkime įvykius:

$$\begin{aligned} A &= \{\text{pasirinkti simboliai } x, y \text{ yra iš to paties segmento}\}, \\ A^c &= \{\text{pasirinkti simboliai } x, y \text{ yra iš skirtingų segmentų}\}. \end{aligned}$$

Pasinaudokime pilnos tikimybės formule

$$P(x = y) = P(x = y|A)P(A) + P(x = y|A^c)P(A^c).$$

Šioje lygybėje imsime:

$$P(x = y|A) = \kappa_t, \quad P(x = y|A^c) = \kappa_0$$

ir suskaičiuosime besąlygines tikimybes:

$$\begin{aligned} P(A) &= \frac{1}{C_N^2} \cdot \frac{N}{2} \cdot \Big(\frac{N}{d} - 1\Big), \\ P(A^c) &= \frac{1}{C_N^2} \cdot \frac{N(N - N/d)}{2}. \end{aligned}$$

Kadangi $P(x = y) = \kappa_c$, tai gausime formulę

$$\kappa_c = \frac{1}{C_N^2} \cdot \frac{N}{2} \cdot \Big(\frac{N}{d} - 1\Big)\kappa_t + \frac{1}{C_N^2} \cdot \frac{N(N - N/d)}{2}\kappa_0.$$

Iš šios lygybės jau galime surasti apytikslę formulę rakto ilgiui:

$$d \approx \frac{N(\kappa_t - \kappa_0)}{\kappa_c(N - 1) + \kappa_t - N\kappa_0}.$$

14.6. Nauji laikai – nauji iššūkiai

Su „juodųjų kambarių“ – Europos absoliutinių monarchijų kriptografų tarnybų laikais baigėsi ir „popieriaus ir pieštuko“ šifrų epocha. Mechanizmai keitė rankų darbą, o kartais ir galvos. Kokie tie pirmieji mechaniniai šifravimo prietaisai? Kokie nauji iššūkiai kriptografijai?

Žinome, kokį kelią nuėjo skaičiavimo technikos kūrėjai: nuo skaitytuvo (abako) iki kompiuterio. Kriptografijos istorijoje irgi galime nubrėžti tokį kelią: nuo pirmojo šifravimui skirto įrenginio (skytalės) iki to paties kompiuterio. Garbingą vietą skaičiavimo technikos istorijoje užsitarnavo mechaninių prietaisų – aritmometrų – kūrėjai: Pascalis, Leibnizas, Babbage'as ir kiti

išradėjai. Kriptografijoje irgi būta mechaninių prietaisų, nors ir gerokai menkiau žinomų. Po spartiečių jų skytalės tikriausiai niekas nenaudojo, Alberti skrituliai taip ir liko išradimu popieriuje...

Visai nebloga idėja, kaip mechanizuoti šifravimą ir dešifravimą XVIII amžiaus pabaigoje kilo Thomui Jeffersonui. Jis buvo trečiasis JAV prezidentas, o pirmojo prezidento – George'o Washingtono metais – valstybės sekretorius, taigi rūpinosi užsienio politika. Galbūt todėl jis ir susimąstė apie šifrus.

Jeffersono prietaisą sudaro trisdešimt šeši siauri vienodo skersmens ritiniai, sumauti ant ašies, apie kurią kiekvienas jų gali sukiotis. Ant kiekvieno ritinio šoninio paviršiaus surašytos abėcėlės raidės – vis kita tvarka. Taigi galime sakyti, kad ant kiekvieno ritinio surašyta vis kita abėcėlė. Ritinius galima numauti nuo ašies ir perstatyti kitaip. Kai visi ritiniai sumauti ant ašies, susidaro 36 raidžių eilutės. Šifruojama taip: pažymėkime vieną eilutę (eilutei fiksuoti galima pritaisyti liniuotę) ir, sukiodami ritinius, surinkime šioje eilutėje 36 (ar mažiau) raidžių tekstą. Visose kitose eilutėse irgi susidarys tekstai – tai šifrai. Galime pasiųsti bet kurį iš jų. Gavėjas, norėdamas iššifruoti šifrą, privalo turėti lygiai tokius pat ritinius, suvertus ant ašies tokia pat tvarka. Surinkęs fiksuotoje eilutėje šifro raides jis peržiūrės kitas eilutes. Vienoje iš jų jis perskaitys mūsų pasiųstą žinią.

Šis šifravimo prietaisas vadinamas Jeffersono ritiniu. Svarbu, kad siuntėjas ir gavėjas naudotųsi juo, sumovę ant ašies siauruosius ritinius su abėcėlėmis ta pačia tvarka. Taigi šios šifravimo sistemos raktas yra ritinių išdėstymo tvarka. Iš viso yra 36! skirtingų išdėstymų. Taigi raktų yra labai daug.

Žinoma, ritinių skaičius gali būti kitas. Galimos ir kitokios šifro sudarymo taisyklės. Pavyzdžiui, pusę šifro galima perskaityti iš vienos, kitą pusę – iš kitos eilutės.

Šis įrenginys buvo ne kartą išrastas pakartotinai. 1891 metais panašų prietaisą sugalvojo įžymus prancūzų kriptografas E. Bazeries, XX amžiaus pradžioje tokį prietaisą naudojo amerikiečiai.

Tiems laikams tai buvo labai saugus šifras. Tačiau juo nesinaudota. Galbūt ir todėl, kad, rūpindamasis sudėtingais to meto politikos reikalais, Jeffersonas primiršo savo išradimą. Panašiais šifravimo įrenginiais naudojosi karinis JAV laivynas ir vyriausybė antrajame XX amžiaus dešimtmetyje. Tačiau ne todėl, kad buvo prisimintas Jeffersono ritinys. Jis tiesiog buvo iš naujo išrastas!

Tačiau tikrasis kriptografijos prietaisų poreikis atsirado tada, kai Samuelis Morse 1844 metais pasiuntė pasauliui pranešimą apie naujos ryšių epochos pradžią. Žinoma, nei jis, nei kiti nemanė, kad prasidėjo kažkas nepaprasta. Samuelis Morse tiesiog išrado telegrafą.

Telegrafas pakeitė visas žmonių tarpusavio bendravimo „per nuotolį“ aplinkybes. Juk svetimo laiško skaitymas greičiau išimtis negu taisyklė, o žinią perduoti telegrafu be pašalinių žmonių pagalbos beveik neįmanoma. Taigi informacijos slaptumo klausimas tapo svarbus ne vien diplomatams ir kariškiams.

Pasikeitusi padėtis kėlė naujus uždavinius kriptografijai.

Vienas iš kriptografijos naujos raidos ženklų – vėl susidomėta šifrais. Tiesa, šifrais visada domėtasi. Tačiau jais domėtasi daugiau teoriškai, o praktikoje vyravo kodai. Sudaryti gerą kodą – specialisto darbas, o išmokyti juo naudotis galima bet kokį eilinį diplomatų atstovybės sekretorių. Tereikia įteikti jam kodų knygą ir paaiškinti, kaip ją vartyti. Kas kita šifrai. Šifruojant ir dešifruojant reikia atidžiai atlikti algoritmo žingsnius, o vienintelė perdavimo arba šifravimo klaida kartais gali visą šifrą paversti neiššifruojamu.

Tačiau telegrafo laikais šifrų pranašumai su kaupu kompensuoja visus nepatogumus. Svarbiausias dalykas, kad šifro raktą pakeisti yra labai lengva, o pakeisti kodą – anaiptol. Juk tektų perrašyti ištisą knygą, o įvykiai juk nelaukia.

Kokių gi šifrų reikia telegrafo epochai, kai didelė dalis svarbių pranešimų keliauja ne vokuose, ne ant brangaus popieriaus su vandens ženklais, bet, pavirtę taškais ir brūkšneliais, įveikia didžiulius atstumus ir pasiekia adresatą mechaninio prietaiso atspausdinti ant siauros popieriaus juostelės?

Pirmasis tai labai aiškiai suvokė ir suformulavo Augustas Kerckhoffas. Jis gimė 1835 metais Olandijoje, buvo senos ir žymios giminės atžala, todėl, matyt, ir visas jo vardas yra itin ilgas: Jean-Guillaume-Hubert-Victor-Francois-Alexandre-Auguste Kerckhoffs von Nieuwenhof. Jis buvo labai išsilavinęs žmogus: Liežo universitete įgijo net du mokslo laipsnius – iš kalbų ir tiksliųjų mokslų, dėstė įvairius dalykus, mokė kalbų ir rašė knygas – daugiausia filologijos.

Kriptografijai svarbus Kerckhoffo veikalas „La cryptographie militaire“ [9] buvo išspausdintas 1883 metais karo mokslams skirtame žurnale, o vėliau išleistas atskira knygele.

Kerckhoffas pabrėžia, kad veikiančioje armijoje atsirado būtinybė palaikyti nuolatinį šifruotą ryšį. Iš to išplaukia nauji reikalavimai naudojamoms šifravimo sistemoms. Kerckhoffas suformuluoja šešias sąlygas.

Pirma, jeigu šifravimo sistema gali būti įveikta, tai tik matematiškai (*le système doit être matériellement, sinon mathématiquement, indéchiffrable*).

Taigi šifruota informacija negali būti atskleista taip kaip iš dėlionės dalelių sudedamas paveikslas, t. y. sistemą galima įveikti tik atskleidus jos matematinius pagrindus.

Antra, sistema turi būti tokia, kad net ją turėdamas priešininkas negalėtų jos įveikti (*il faut qu'il n'exige pas le secret, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi*).

Trečia, sistemos raktas turi būti įsimenamas ir perduodamas jo neužrašius, jis turi būti keičiamas (*la clef doit pouvoir en être communiquée et retenue sans le secours de notes écrites, et être changée et modifiée au gré des corre-*

[9] Auguste Kerckhoffs. La cryptographie militaire. Journal des sciences militaires, vol. IX, Jan. 1883, p. 5—83, Feb. 1883, p. 161—191.

spondants).

Ketvirta, sistema turi būti pritaikyta telegrafo ryšiui (*il faut qu'il soit applicable à la correspondance télégraphique*).

Penkta, šifravimo sistema turi būti nešiojama ir naudojimuisi ja nereiktų daugelio žmonių (*il faut qu'il soit portatif, et que son maniement ou son fonctionnement n'exige pas le concours de plusieurs personnes*).

Šešta, sistema turi būti paprasta naudotis: neturi būti reikalinga nei proto įtampa, nei ilga taisyklių seka (*le systéme doit être d'un usage facile ne demandant ni tension d'esprit, ni la connaissance d'une longue série de règles à observer*).

Jeigu ir reiktų ką pakeisti pritaikant šį „kriptografijos kodeksą" mūsų laikams – tai vietoj telegrafo minėti elektroninį ryšį.

Tačiau kriptografija Kerckhoffo laikais vis dar buvo kūdikystės amžiuje. Pats A. Kerckhoffas savo straipsnyje rašo, kad jį stebina mokyti žmonės, siūlantys šifravimo sistemas, kurias įveikti galima per pusvalandį. Kad kriptografijos reikšmė būtų suvokta, reikėjo sunkių išbandymų. Būtinybę kuo greičiau subręsti atskleidė Pirmasis pasaulinis karas.

14.7. Rotoriai ir Enigma

> *Enigma – šitaip vokiečiai vadino šifravimo įrenginius, kuriuos jie naudojo Antrojo pasaulinio karo mūšiuose. Enigma – tai mūšis, kurį laimėjo britų matematikai, dirbdami tyliuose Bletchley Park kambariuose...*

Telegrafu ir radijo ryšiu Pirmajame pasauliniame kare pradėta naudotis neturint beveik jokios patirties kriptografijos ir kriptoanalizės srityse. Tik prancūzai buvo kiek geriau pasiruošę. Šlovinga britų naujųjų laikų kriptografijos istorija prasidėjo tiesiog prie vieno stalo su vokiečių šifrų šūsnimi, už kurio susėdo keli karo laivininkystės koledžo dėstytojai, tikėdamiesi ką nors iš tos šūsnies išrausti. Kriptoanalitikų darbas niekada nebuvo toks svarbus kaip šio karo metu. Užtenka paminėti vien Zimmermano telegramos atvejį: britams iššifravus vokiečių karo ministro telegramą pasiuntiniui Meksikoje, vis dar delsęs JAV prezidentas W. Wilsonas apsisprendė dėl JAV dalyvavimo kare.

Tačiau karas baigėsi ir dėmesys kriptografijai (o taip pat ir pinigai) sumenko. Talentingas amerikiečių inžinierius Gilbertas Vernamas truputį pavėlavo. 1918 metais Vašingtone jis pateikė paraišką naujo šifravimo-dešifravimo prietaiso patentui gauti. Jo idėja apie telegrafu perduodamos informacijos šifravimą buvo puiki. Tuomet perdavimui buvo naudojamas Baudot'o kodas. Naudojantis šiuo kodu, kiekvienai raidei perduoti telegrafu reikia penkių trumpų laiko intervalų. Per kiekvieną intervalą įrenginys arba pasiunčia elektros impulsą, arba ne. Impulso perdavimą galime pažymėti vienetu, o jo nebuvimą nuliu. Tada kiekviena raidė bus koduojama vienetų ir nulių gretiniu,

kuris sudarytas iš penkių simbolių. Gavėjo įrenginys pagal šiuos perduodamus nulių-vienetų penketus turi atkurti perduodamas abėcėlės raides.

Tarkime, telegrafu reikia perduoti vieną raidę, t. y. tam tikrą nulių-vienetų penketą, pavyzdžiui, 01001. Vernamui kilo mintis: sudarykime elektrinę grandinę, kuri sudėtų jai perduodamas vienodo ilgio nulių-vienetų eilutes panariui, naudodamasi tokia sudėties lentele:

$$0 \oplus 0 = 0; \quad 0 \oplus 1 = 1; \quad 1 \oplus 0 = 1; \quad 1 \oplus 1 = 0.$$

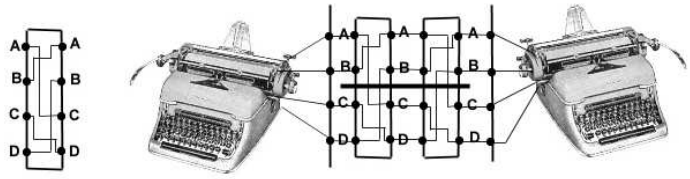
Jeigu tokiai grandinei perduosime eilutes $m = 01001$ ir $k = 11011$, tai grandinė sukurs eilutę $c = 10010$. Jeigu m yra pranešimas, kurį reikia perduoti, tai c yra šifras, gautas panaudojus raktą k. Kaip gavėjas gali iššifruoti gautą šifrą c? Jeigu jo įrenginyje bus tokia pati sudėties grandinė, tai, įvedęs į ją eilutes c ir k, gaus m. Taigi šifruojama ir dešifruojama visiškai vienodai ir šiuos veiksmus gali atlikti pats įrenginys. Tiesa, jeigu reikia perduoti ilgą pranešimą, tai ir rakto reikia ilgo. Taigi G. Vernamas pasiūlė šifravimui naudoti operaciją, kurią matematikai vadina sudėtimi moduliu 2, o informatikai – XOR operacija. Šią operaciją naudojome nagrinėdami klaidas taisančius kodus. Šiuolaikinėse kriptosistemose ji sutinkama kone kiekvianame žingsnyje.

Vernamo įrenginys galėjo padaryti perversmą ryšių technikoje. Tačiau taip neįvyko. Viena vertus, vyriausybės galvojo, kaip baigti kariauti, o ne iš naujo pradėti. Kita vertus, komercinės įstaigos buvo įpratusios naudotis kodų knygomis ir nepirko naujojo įrenginio. Svarbiausia šios puikios idėjos komercinės nesėkmės priežastis ta, kad ji atsirado anksčiau, nei subrendo nauji saugaus ryšio reikalavimai.

Kriptografinių prietaisų raida pasuko ne Vernamo nurodyta kryptimi. Pasirodė kažkas panašaus į Jeffersono ritinius, suvertus ant vienos ašies.

Įsivaizduokime diską, padarytą iš kietos elektrai nelaidžios medžiagos, kurio abiejose pusėse ratu pagal kraštą išdėstyti kontaktai. Kontaktų abiejose pusėse yra tiek, kiek raidžių abėcėlėje (lotyniškoje abėcėlėje 26). Taigi abiejose pusėse kiekviena abėcėlės raidė turi po kontaktą. Šie kontaktai tarpusavyje poromis sujungti laidais. Pavyzdžiui, raidės A kontaktas gali būti sujungtas su raidės C kontaktu, raidės B – su L ir t. t. Šis ritinys ir yra pagrindinė naujųjų kriptografijos prietaisų detalė. Jis vadinamas rotoriumi, jau pats vardas rodo, kad šifravimo sistemoje didelė reikšmė teikiama šio ritinio posūkiams apie pervertą ašį.

Panagrinėkime rotorių panaudojimo šifravimui idėją, kurią kone vienu metu patentavo amerikietis Edwardas Hughas Hebernas, olandas Alexanderis Kochas, švedas Arvidas Gerhardas Dammas ir vokietis Arthuras Scherbiusas.



Šifravimo įrenginio, naudojančio rotorius, veikimo principas. Tikrieji įrenginiai, žinoma, buvo sudėtingesni. Pavyzdžiui, impulsas, praėjęs elektros grandine nuo pirmojo iki paskutiniojo rotoriaus kontaktų, atsispindėjęs grįžtavo jau kita grandine į pirmąjį rotorių.

Įsivaizduokime, kad prie abiejų rotoriaus pusių kontaktų prijungtos dvi elektromechaninės spausdinimo mašinėlės. Jeigu paspausime kairiosios mašinėlės A raidės klavišą, tai į kairiosios rotoriaus pusės A raidės kontaktą bus perduotas srovės impulsas, kuris laidu pateks į dešiniosios rotoriaus pusės raidės C kontaktą, ir dešinioji spausdinimo mašinėlė atspausdins raidę C (jeigu, žinoma, raidės A kontaktas sujungtas su raidės C kontaktu). „Sukonstravome“ paprastą vienu abėcėlių keitiniu paremtą šifravimo sistemą, kuri nėra nei nauja, nei saugi. Jeigu be perstojo spaudysime kairiosios spausdinimo mašinėlės raidės A klavišą, dešinioji mašinėlė spausdins CCCCCC..... Dabar patobulinkime įrenginį.

Mūsų rotorių patalpinkime tarp dviejų skritulio formos nejudamų plokštelių su tokia pat tvarka kaip ant rotoriaus išdėstytais kontaktais: kairiosios plokštelės kontaktai liečiasi su rotoriaus kairiosios pusės kontaktais, dešiniosios plokštelės – su dešiniosios pusės. Spausdinimo mašinėles prijunkime prie atitinkamų plokštelių kontaktų. Šifravimo sistema nepasikeitė, tačiau šitaip mes įgijome galimybę sukioti mūsų rotorių nepriklausomai nuo prijungtų spausdinimo mašinėlių. Padarykime taip, kad, kiekvieną kartą paspaudus kairiosios spausdinimo mašinėlės klavišą, rotorius pasisuktų per 1/26 apskritimo, tarkime, laikrodžio rodyklės kryptimi. Paspaudus raidės A klavišą, antroji spausdinimo mašinėlė atspausdins raidę C, rotorius pasisuks, tačiau nejudamų plokštelių kontaktai vėl liesis su rotoriaus kontaktais, tačiau jau kitais, todėl antrą kartą paspaudus raidės A klavišą, antroji spausdinimo mašinėlė atspausdins nebe C, bet kokią nors kitą raidę! Jeigu vėl be perstojo spaudysime kairiosios spausdinimo mašinėlės raidės A klavišą, tai dešinioji mašinėlė spausdins raidžių seką, kuri ims kartotis tik po 26 raidės. Šis įrenginys realizuoja šifravimo sistemą, panašią į tą, kurią pasiūlė Trithemius. Panagrinėkime matematinę tokios sistemos su vienu rotoriumi struktūrą.

Tarkime, nejudamų plokštelių kontaktams abėcėlės raidės priskirtos eilės tvarka, t. y. jeigu spaudžiame raidės A klavišą, tai impulsas patenka į kairiosios plokštelės pirmąjį kontaktą, jeigu B – į antrąjį ir t. t. Analogiškai jeigu impulsas patenka į dešiniosios plokštelės pirmąjį kontaktą, tai šifrą spausdinanti mašinėlė išspausdina A, jeigu į antrąjį – B ir t. t. Sunumeruokime plokštelių kontaktus eilės tvarka, raidės A kontaktui priskirkime 1, raidės B

– 2 ir t. t. Tegu abėcėlėje yra n raidžių. Tais pačiais skaičiais $1, 2, \ldots, n$ sunumeruokime ir kairiuosius bei dešiniuosius rotoriaus kontaktus, abiejose pusėse numeravimas prasideda nuo tos pačios padėties. Tačiau rotoriaus kontaktai yra poromis sujungti. Sujungimus nurodysime keitiniu

$$\lambda = \begin{pmatrix} 1 & 2 & \ldots & n \\ i_1 & i_2 & \ldots & i_n \end{pmatrix}.$$

Šiuo keitiniu nurodoma, kad pirmasis kairiosios pusės kontaktas sujungtas su i_1 dešiniosios pusės kontaktu ir t. t.

Tarkime, plokštelių ir rotoriaus pradinė padėtis yra tokia, kad kairiosios plokštelės pirmasis kontaktas yra ties pirmuoju kairiosios rotoriaus pusės kontaktu, rotoriaus dešiniosios pusės pirmasis kontaktas yra ties pirmuoju dešiniosios plokštelės kontaktu. Taigi pirmosios plokštelės ir rotoriaus kontaktų bei rotoriaus ir antrosios plokštelės kontaktų atitiktį galime nusakyti tuo pačiu trivialiu keitiniu

$$\epsilon = \begin{pmatrix} 1 & 2 & \ldots & n \\ 1 & 2 & \ldots & n \end{pmatrix}.$$

Kairiosios ir dešiniosios plokštelių kontaktų atitiktį (šifro keitinį) galime užrašyti taip:

$$\sigma_0 = \lambda = \epsilon\lambda\epsilon.$$

Pasukime rotorių per vieną poziciją. Dabar kairiosios plokštelės ir kairiųjų rotoriaus kontaktų bei dešiniųjų rotoriaus ir dešiniosios plokštelės kontaktų atitiktį užrašysime jau kitais keitiniais:

$$\rho_1 = \rho = \begin{pmatrix} 1 & 2 & \ldots & n-1 & n \\ 2 & 3 & \ldots & n & 1 \end{pmatrix}, \quad \rho^{-1} = \begin{pmatrix} 2 & 3 & \ldots & n & 1 \\ 1 & 2 & \ldots & n-1 & n \end{pmatrix}.$$

Savo ruožtu šifravimo keitinys bus

$$\sigma_1 = \rho^{-1}\lambda\rho.$$

Pasukę rotorių per $2, 3, \ldots, m$ pozicijų, gausime plokštelių ir rotoriaus kontaktų atitikties keitinius $\rho^2, \rho^3, \ldots, \rho^m$ ir $\rho^{-2}, \rho^{-3}, \ldots, \rho^{-m}$. Vadinasi pasukę rotorių per m pozicijų, gauname tokį šifro keitinį:

$$\sigma_m = \rho^{-m}\lambda\rho^m, \quad m = 0, 1, 2, \ldots, \rho^n = \epsilon.$$

Taigi visų žingsnių šifravimo keitinius apibrėžia keitinys λ ir pradinė plokštelių ir rotoriaus kontaktų atitiktis.

Pavyzdžiui, jeigu keturių raidžių abėcėlės atveju plokštelių ir rotoriaus kontaktų pradinę padėtį nusako vienetinis keitinys ϵ, tai su rotoriaus kontaktų porų keitiniu λ gausime tokius keturių žingsnių šifravimo keitinius:

$$\lambda = \sigma_0 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 1 & 3 \end{pmatrix}, \ \sigma_1 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 4 & 2 & 3 \end{pmatrix},$$

$$\sigma_2 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 1 & 2 & 4 \end{pmatrix}, \ \sigma_3 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 1 & 3 & 2 \end{pmatrix}.$$

Patobulinkime įrenginį dar kartą. Imkime kitą rotorių (kairiosios ir dešiniosios pusių kontaktai sujungti poromis, tačiau kokia nors kita tvarka) ir užmaukime ant ašies. Tegu abu rotoriai liečiasi savo kontaktais: pirmojo rotoriaus dešiniosios pusės kontaktai liečia antrojo rotoriaus kairiosios pusės kontaktus, taigi pirmojo rotoriaus kairiosios pusės kontaktai liečia kairiosios nejudančios plokštelės kontaktus, o antrojo rotoriaus dešiniosios pusės kontaktai – dešiniosios plokštelės kontaktus. Spausdinimo mašinėlės lieka sujungtos su plokštelių kontaktais. Jeigu dabar nuspausime kairiosios spausdinimo mašinėlės raidės A klavišą, dešinioji atspausdins tikriausiai nebe C, bet kokią nors kitą raidę, pavyzdžiui, V. Padarykime taip, kad kiekvieną kartą, kai nuspaudžiamas kairiosios mašinėlės klavišas, antrasis rotorius pasisuka per 1/26 apskritimo laikrodžio rodyklės kryptimi, o kai šis rotorius po 26 posūkių grįžta į pradinę padėtį, per 1/26 apskritimo laikrodžio rodyklės kryptimi pasisuka pirmasis rotorius. Tada vėl sukiojasi antrasis rotorius, o pirmasis vėl pajuda tik po 26 antrojo rotoriaus posūkių. Panašiai juda elektros, vandens ar dujų apskaitos skaitiklių ritinėliai su skaitmenimis.

Jeigu nuolat spaudysime kairiosios mašinėlės raidės A klavišą, po kelių simbolių dešiniosios mašinėlės spausdinama raidžių seka ims kartotis? Po $26 \times 26 = 676$ simbolių. O jeigu panaudosime ne du, bet tris rotorius? Tada po $676 \times 26 = 17576$ simbolių. Net ir pasitelkus matematinius metodus bei kompiuterius, analizuoti tokį šifrą nėra lengva. Taigi rotorių sistema itin paprastai realizuoja šifrus su daug abėcėlių. Šifras priklauso nuo pradinės rotorių tarpusavio padėties, kurią galima keisti. Šitaip galima pasiekti, kad šifras priklausytų nuo rakto.

Matematiškai šifravimą su rotoriais galime aprašyti taip. Tarkime, turime tris rotorius ir kiekvieno rotoriaus kontaktai sunumeruoti tokia tvarka kaip nagrinėtu atveju, sujungtas kontaktų poras nusako keitiniai $\lambda_1, \lambda_2, \lambda_3$. Be to, tarkime, kad pradinėje padėtyje visų besiliečiančių kontaktų numeriai tie patys, t. y. kairiosios plokštelės pirmąjį kontaktą liečia pirmojo rotoriaus kairiosios pusės pirmasis kontaktas, jo dešiniosios pusės pirmąjį kontaktą liečia antrojo rotoriaus kairiosios pusės pirmasis kontaktas ir t.t. Koks šifravimo keitinys bus m-ajame žingsnyje, jei $m < n^3$? Išreikškime m n-ainėje

skaičiavimo sistemoje:

$$m = m_1 + m_2 n + m_3 n^2, \quad 0 \le m_i < n.$$

Tada kiek pagalvojus galima įsitikinti, kad šifravimo abėcėlės keitinys bus toks:

$$\sigma_m = \rho^{-m_3}\lambda_3\rho^{m_3}\rho^{-m_2}\lambda_2\rho^{m_2}\rho^{-m_1}\lambda_1\rho^{m_1}.$$

Rotorių principas panaudotas įžymiuosiuose vokiečių šifravimo įrenginiuose „Enigma“ . Antrojo pasaulinio karo metu vokiečiai naudojo kelis įrenginių variantus su trimis rotoriais, kuriuos buvo galima parinkti iš penkių rotorių rinkinio. Įrenginiai buvo puikūs, tačiau jie neatlaikė lenkų ir britų kriptoanalizės atakų. Štai tik dvi pavardės iš didingos Antrojo pasaulinio karo kriptoanalizės istorijos: Marijanas Rejewskis ir Alanas Turingas. Kodėl „Enigma“ pasidavė? Ne todėl, kad algoritmas buvo nesaugus, bet todėl, kad kiekvienos apsaugos sistemos saugumo lygis yra toks pat kaip pačios silpniausios jos grandies. Silpnoji „Enigmos“ praktinio naudojimo grandis – būtinybė susitarti dėl raktų (pradinių rotorių padėčių) ir vokiečių įprotis tai daryti.

15 Informacijos teorija ir kriptosistemos

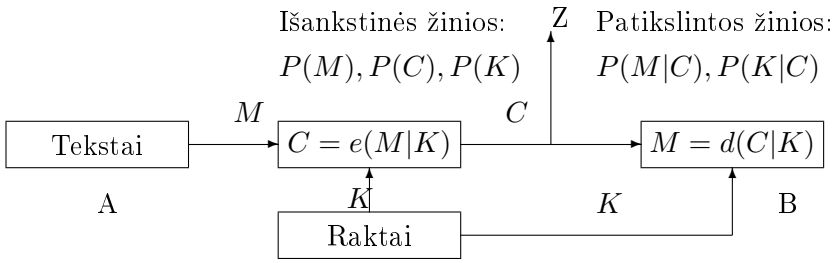
Ištisus šimtmečius kriptosistemų saugumas vertintas pasikliaujant tik subjektyvia šios srities autoritetų nuomone. Žinoma, patyrusių kriptoanalitikų nuomonė ir dabar yra svarbus argumentas. Kriptografijoje ir šiandien ne viską galima objektyviai pasverti ir įvertinti. Tačiau daug ką galima. Kokie gi tie moksliniai kriptosistemų saugumo vertinimo kriterijai?

15.1. Shannono modelis

C. Shannonas – informacijos teorijos pradininkas – yra ir teorinio kriptosistemų vertinimo pagrindų kūrėjas. Panagrinėkime jo sukurtą modelį.

Pirmasis teorinį kriptosistemų vertinimo pagrindą pabandė pateikti C. Shannonas. Jis pasinaudojo savo paties sukurtos informacijos teorijos sąvokomis. Informacijos teorijoje nagrinėjamos patikimo ryšio nepatikimais kanalais galimybės. Kada iš iškraipytos perdavimo kanale informacijos galima atkurti siųstąją? Šifravimą irgi galima interpretuoti kaip pradinės informacijos iškraipymą. Kada iš šifro (iškraipytos informacijos) galima atkurti pradinę, o taip pat – šifravimui naudotą raktą? Kitaip tariant – kaip galima kiekybiškai vertinti kriptosistemos atsparumą pavienių šifrų atakų atžvilgiu?

Panagrinėkime kriptosistema apsaugoto A ir B ryšio modelį, kurį naudojo C. Shannonas. Apsiribosime simetrinėmis kriptosistemomis $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$; tarsime, kad šifravimui ir dešifravimui naudojamas raktas yra tas pats: $K_e = K_d = K$.



C. Shannono šifruoto ryšio modelis. Kriptosistema tuo silpnesnė, kuo labiau gautas šifras patikslina kriptoanalitiko išankstines žinias apie siunčiamą tekstą ir panaudotą raktą.

Tarsime, kad kriptoanalitikui Z visos kriptosistemos komponentės yra gerai žinomos: jis žino, kokiai aibei priklauso šifruojami pranešimai, iš kokios aibės pasirenkami raktai ir kaip veikia šifravimo ir dešifravimo algoritmai. Jis taip pat turi neribotas galimybes stebėti ryšio kanalą bei atlikti skaičiavimus. Įsivaizduokime, kad jis laukia pirmojo siuntinio, kurį jis pasiruošęs įsirašyti į savo laikmenas. Pranešimas, kuris bus užšifruotas, jo požiūriu, yra atsitiktinio dydžio reikšmė. Žymėsime jį taip pat kaip galimų pranešimų aibę, t. y. $\mathcal{M}$. Kriptoanalitikas taip pat turi tam tikrą išankstinę informaciją, kokie pranešimai yra mažiau, kokie daugiau tikėtini, t. y. jis žino tikimybes

$$p(M) = P(\mathcal{M} = M).$$

Analogiškai naudojamą raktą irgi galime interpretuoti kaip atsitiktinį dydį $\mathcal{K}$, įgyjantį reikšmes su tikimybėmis $p(K)$. Natūralu daryti prielaidą, kad dydžiai $\mathcal{M}$ ir $\mathcal{K}$ yra nepriklausomi atsitiktiniai dydžiai. Šifruotą tekstą irgi galime suprasti kaip atsitiktinį dydį $\mathcal{C}$, su tikimybėmis $p(C)$ įgyjantį reikšmes iš galimų šifrų aibės; jis vienareikšmiškai apibrėžiamas su $\mathcal{M}, \mathcal{K}$ sąryšiu

$$e(\mathcal{M}|\mathcal{K}) = \mathcal{C}.$$

Taigi tikimybės $p(M), p(K), p(C)$ yra ta išankstinė informacija, kurią turi kriptoanalitikas Z prieš sugaudamas pirmąjį siunčiamą šifrą. Tarkime, kad to šifro C jis galų gale sulaukė. Tuomet jis gali patikslinti savo turimas žinias apie siunčiamą pranešimą, apskaičiuodamas sąlygines tikimybes

$$P(\mathcal{M} = M|\mathcal{C} = C), \quad M \in \mathcal{M}.$$

Gali būti, kad reikšmės liko tos pačios, tada Z iš gauto šifro neturės jokios naudos!

94 apibrėžimas. *Kriptosistemą $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$ vadinsime besąlygiškai saugia tada ir tik tada, kai su visomis $\mathcal{M}$ ir $\mathcal{C}$ reikšmėmis M, C teisinga lygybė*

$$P(\mathcal{M} = M|\mathcal{C} = C) = P(\mathcal{M} = M).$$

Pastebėkime, kad šis apibrėžimas reiškia, kad atsitiktiniai dydžiai $\mathcal{M}, \mathcal{C}$ yra nepriklausomi. Tiesiog iš šio apibrėžimo ir sąlyginių tikimybių savybių išplaukia toks teiginys:

116 teorema. *Kriptosistema $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$ yra visiškai saugi tada ir tik tada, kai su visomis $\mathcal{M}$ ir $\mathcal{C}$ reikšmėmis M, C teisinga lygybė*

$$P(\mathcal{C} = C | \mathcal{M} = M) = P(\mathcal{C} = C).$$

Ar egzistuoja visiškai saugios sistemos? Prisiminkime amžininkų neįvertintą G. Vernamo idėją.

Tegu $\mathcal{B} = \mathbb{F}_2 = \{0, 1\}$ – dvejetainė abėcėlė, o pranešimų, raktų ir šifrų aibės sudarytos iš šios abėcėlės n ilgio žodžių:

$$\mathcal{M} = \mathcal{K} = \mathcal{C} = \mathcal{B}^n.$$

Tegu šifravimo operacija – tiesiog žodžių sudėtis, kurią naudojome kodavimo teorijoje:

$$C = e(M|K) = M + K, \quad d(C|K) = C + K = M.$$

Jeigu visos rakto reikšmės yra vienodai galimos, t. y.

$$p(K) = \frac{1}{2^n},$$

tai tokia kriptosistema yra visiškai saugi. Iš tikrųjų, jeigu šifruojamas pranešimas yra M, tai šifras C bus gautas tik tuomet, kai bus naudojamas raktas $K = M + C$, taigi

$$P(\mathcal{C} = C | \mathcal{M} = M) = P(\mathcal{K} = M + C) = \frac{1}{2^n}.$$

Kita vertus, pasinaudoję pilnosios tikimybės formule, gauname

$$\begin{aligned} P(\mathcal{C} = C) &= \sum_{M \in \mathcal{M}} P(\mathcal{M} = M) P(\mathcal{C} = C | \mathcal{M} = M) \\ &= \sum_{M \in \mathcal{M}} P(\mathcal{M} = M) P(\mathcal{K} = M + C) = \frac{1}{2^n} \sum_{M \in \mathcal{M}} P(\mathcal{M} = M) = \frac{1}{2^n}. \end{aligned}$$

Taigi tokia kriptosistema yra visiškai saugi. Tikriausiai pastebėjote, kad analogišką visiškai saugią kriptosistemą galime sudaryti imdami vietoj dvejetainės abėcėlės bet kokį baigtinį kūną $\mathbb{F}_q$.

Tokia sistema yra visiškai saugi, jeigu raktas naudojamas tik vienam pranešimui šifruoti. Panagrinėkime atvejį, kai šios sąlygos nesilaikoma. Tarkime, raktas pakeičiamas užšifravus du pranešimus. Tada kriptoanalitikas gali savo analizę pradėti sulaukęs dviejų šifrų C_1, C_2 ir manyti, kad stebi kriptosistemą su pranešimų, šifrų ir raktų aibėmis

$$\mathcal{M} = \mathcal{C} = \mathcal{B}^{2n}, \quad \mathcal{K} = \mathcal{B}^n.$$

Prieš gaudamas šifrus kriptoanalitikas galbūt manė, kad visų 2^{2n} pranešimų tikimybės $P(M)$ yra teigiamos. Kokią informaciją apie siųstą pranešimą $M = M_1M_2$ kriptoanalitikui suteikia šifras $C = C_1C_2$? Sudėję abu šifrus, gauname

$$D = C_1 + C_2 = (M_1 + K) + (M_2 + K) = M_1 + M_2.$$

Taigi iš visų pranešimų, kurie galėjo būti siųsti, aibės kriptoanalitikas gali atmesti visus tuos, kurie neturi šios savybės, t. y. $M_1 + M_2 \neq D$. Tada nelygybė $P(M|C) > 0$ liks galioti tik 2^n visų aibės pranešimų! Gavęs šifrus kriptoanalitikas gerokai patikslino savo išankstines žinias.

Deja, sudėtingoms pranešimų aibėms visiškai saugios sistemos irgi yra neišvengiamai sudėtingos. Štaip galite interpretuoti šią teoremą.

117 teorema. *Jei kriptosistema yra visiškai saugi, tai $|\mathcal{K}| \geq |\mathcal{M}|$, t. y. raktų yra ne mažiau, negu pranešimų.*

Įrodymas. Pažymėkime M_i visus galimus pranešimus, C_j – visus galimus šifrus, kurių tikimybės teigiamos,

$$P(C_j|M_i) = P(\mathcal{C} = C_j|\mathcal{M} = M_j), \quad P(C_j) = P(\mathcal{C} = C_j).$$

Jeigu sistema visiškai saugi, tai su visais i, j

$$P(C_j|M_i) = P(C_j).$$

Tarkime, raktų yra mažiau negu pranešimų. Imkime kokį nors pranešimą M_i ir šifruokime jį visais raktais. Kadangi šifrų yra ne mažiau negu pranešimų, tai šifrai $e(M_i|K)$ nesutaps su visų galimų šifrų aibe, t. y. egzistuos šifras C_j, kad $e(M_i|K) \neq C_j$. Tai reiškia, kad $P(C_j|M_i) = 0$. Tačiau $P(C_j) > 0$. Gavome prieštarą. Kriptosistema negali būti visiškai saugi.

15.2. Kriptosistemos dydžių entropijos

Atsitiktinio dydžio entropija – tai neapibrėžtumo, kurį stebėtojas jaučia laukdamas dydžio reikšmės, matas. Geras įrankis kriptosistemos elementų savybėms reikšti!

Kriptosistemos saugumas priklauso nuo to, kiek netiesioginės informacijos apie atsitiktinį dydį $\mathcal{M}$ (ir $\mathcal{K}$) suteikia atsitikinis dydis $\mathcal{C}$. Informacijos kiekiams reikšti galime pasinaudoti entropijos sąvoka.

Su kriptosistema kriptoanalitikas sieja tris atsitiktinius dydžius $\mathcal{M}, \mathcal{K}, \mathcal{C}$, o neapibrėžtumą jų atžvilgiu prieš gaudamas šifrą ir jį gavęs gali reikšti naudodamas besąlygines ir sąlygines entropijas

$$H(\mathcal{M}),\ H(\mathcal{K}),\ H(\mathcal{C}),\ H(\mathcal{M}|\mathcal{C}),\ H(\mathcal{K}|\mathcal{C}).$$

Jeigu kriptosistema yra visiškai saugi, tai pranešimas ir šifras yra nepriklausomi atsitiktiniai dydžiai. Prisiminę entropijos savybes galime visiško saugumo sąlygą suformuluoti taip:

118 teorema. *Kriptosistema $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$ yra visiškai saugi tada ir tik tada, kai $H(\mathcal{M}|\mathcal{C}) = H(\mathcal{M})$.*

Kriptosistemoje rakto slaptumas yra didesnis negu siunčiamų pranešimų slaptumas. Maždaug tokia yra šios teoremos prasmė.

119 teorema. *Kriptosistemos elementams $\mathcal{M}, \mathcal{K}, \mathcal{C}$ teisinga lygybė*

$$H(\mathcal{K}|\mathcal{C}) = H(\mathcal{M}|\mathcal{C}) + H(\mathcal{K}|\mathcal{M}, \mathcal{C}).$$

Įrodymas. Panaudoję lygybę $H(X|Y) = H(X,Y) - H(Y)$ su $X = \mathcal{M}, Y = \mathcal{C}$, gausime

$$H(\mathcal{M}|\mathcal{C}) = H(\mathcal{M}, \mathcal{C}) - H(\mathcal{C}).$$

Dar kartą pasinaudokime ta pačia lygybe tik dabar su $X = \mathcal{K}, Y = \langle \mathcal{M}, \mathcal{C} \rangle$:

$$\begin{aligned} H(\mathcal{K}|\mathcal{M}, \mathcal{C}) &= H(\mathcal{M}, \mathcal{K}, \mathcal{C}) - H(\mathcal{M}, \mathcal{C}), \\ H(\mathcal{M}, \mathcal{C}) &= H(\mathcal{M}, \mathcal{K}, \mathcal{C}) - H(\mathcal{K}|\mathcal{M}, \mathcal{C}). \end{aligned}$$

Taigi

$$H(\mathcal{M}|\mathcal{C}) = H(\mathcal{M}, \mathcal{K}, \mathcal{C}) - H(\mathcal{K}|\mathcal{M}, \mathcal{C}) - H(\mathcal{C}). \tag{77}$$

Analogiškai gauname

$$H(\mathcal{K}|\mathcal{C}) = H(\mathcal{K}, \mathcal{C}) - H(\mathcal{C}) = H(\mathcal{M}, \mathcal{K}, \mathcal{C}) - H(\mathcal{M}|\mathcal{K}, \mathcal{C}) - H(\mathcal{C}).$$

Pastebėję, kad iš $d(\mathcal{C}|\mathcal{K}) = \mathcal{M}$ išplaukia $H(\mathcal{M}|\mathcal{K}, \mathcal{C}) = 0$, paskutiniąją lygybę galime perrašyti taip:

$$H(\mathcal{K}|\mathcal{C}) = H(\mathcal{M}, \mathcal{K}, \mathcal{C}) - H(\mathcal{C}). \tag{78}$$

Kad gautume teoremos tvirtinimą, belieka pasinaudoti gautuoju sąryšiu (77) lygybėje:

$$H(\mathcal{M}|\mathcal{C}) = H(\mathcal{K}|\mathcal{C}) - H(\mathcal{K}|\mathcal{M}, \mathcal{C}).$$

Skaičiuoti praktikoje naudojamų kriptosistemų dydžių entropijas yra sudėtinga. Panagrinėkime „žaislinį" pavyzdį. Ir toks pavyzdys padeda suvokti, kas vyksta sudėtingų sistemų atvejais.

Pavyzdys. Tarkime urnoje yra baltų, juodų ir raudonų rutulių. Jų yra atitinkamai $b = 5, j = 4, r = 3$. Atsitiktinai ištraukus rutulį, užrašoma pirmoji jo spalvos raidė: B, J arba R. Ta raidė – tai pranešimas. Tačiau tas pranešimas „šifruojamas", t. y. raidė keičiama kita. Šifravimui naudojami

šeši keitiniai $k_1, k_2, \ldots, k_6$, kurie parenkami su vienodomis tikimybėmis:

	B	J	R
k_1	B	J	R
k_2	B	R	J
k_3	J	B	R
k_4	J	R	B
k_5	R	B	J
k_6	R	J	B

Apskaičiuokime entropijas $H(\mathcal{M}), H(\mathcal{K}), H(\mathcal{M}|C), H(\mathcal{K}|C)$. Pranešimų B, J, R tikimybės lygios atitinkamai $\frac{5}{12}$, $\frac{4}{12}$ ir $\frac{3}{12}$, taigi

$$H(\mathcal{M}) = \frac{5}{12}\log_2\frac{12}{5} + \frac{4}{12}\log_2\frac{12}{4} + \frac{3}{12}\log_2\frac{12}{3} \approx 1.5546.$$

Nesunku rasti ir rakto entropiją;

$$H(\mathcal{K}) = \log_2 6 \approx 2.585.$$

Kiek daugiau tenka padirbėti skaičiuojant sąlygines entropijas

$$H(\mathcal{M}|\mathcal{C}) = \sum_C H(\mathcal{M}|\mathcal{C}=C)P(\mathcal{C}=C), \quad H(\mathcal{K}|\mathcal{C}) = \sum_C H(\mathcal{K}|\mathcal{C}=C)P(\mathcal{C}=C).$$

Šifrų tikimybes surasti nesunku, pasinaudojus pilnosios tikimybės formule:

$$P(\mathcal{C}=B) = P(\mathcal{C}=J) = P(\mathcal{C}=R) = \frac{5}{12}\cdot\frac{2}{6} + \frac{4}{12}\cdot\frac{2}{6} + \frac{3}{12}\cdot\frac{2}{6} = \frac{2}{6} = \frac{1}{3}.$$

Entropijai $H(\mathcal{M}|\mathcal{C}=C)$ apskaičiuoti reikia tikimybių $P(\mathcal{M}=M|\mathcal{C}=C)$. Suraskime, pavyzdžiui, $P(\mathcal{M}=B|\mathcal{C}=J)$:

$$\begin{aligned} P(\mathcal{M}=B|\mathcal{C}=J) &= \frac{P(\mathcal{M}=B,\mathcal{C}=J)}{P(\mathcal{C}=J)} = \frac{P(\mathcal{M}=B)P(\mathcal{C}=J|\mathcal{M}=B)}{P(\mathcal{C}=J)} \\ &= \frac{P(\mathcal{M}=B)\cdot\frac{2}{6}}{P(\mathcal{C}=J)} = P(\mathcal{M}=B). \end{aligned}$$

Kitais atvejais taip pat gautume, kad sąlyginės tikimybės lygios besąlyginėms:

$$P(\mathcal{M}=M|\mathcal{C}=C) = P(\mathcal{M}=M).$$

Taigi kriptosistema visiškai saugi ir $H(\mathcal{M}|\mathcal{C}) = H(\mathcal{M})$.

Tačiau sąlyginės raktų tikimybės nelygios besąlyginėms, pavyzdžiui,

$$P(\mathcal{K}=k_1|\mathcal{C}=B)=\frac{P(\mathcal{K}=k_1,\mathcal{C}=B)}{P(\mathcal{C}=B)}=\frac{\frac{1}{6}\cdot\frac{5}{12}}{\frac{1}{3}}=\frac{5}{24}.$$

Apskaičiavę visas tikimybes ir visas entropijas $H(\mathcal{K}|\mathcal{C}=C)$, gautume

$$H(\mathcal{K}|\mathcal{C})=2.5546,\quad H(\mathcal{K}|\mathcal{C})<H(\mathcal{K}).$$

Galima suvokti tokį reiškinį kad ir taip: kriptosistemos rakto entropija buvo „perteklinė", kriptosistema gali būti visiškai saugi ir su mažesne rakto entropija. Štai šiek tiek to pertekliaus ir nubyrėjo...

O dabar tarkime, kad vienas po kito iš urnos traukiami du rutuliai ir pagal jų spalvas sudaromas dviejų raidžių žodis. Žodžiui šifruoti naudojamas vienas iš šešių raktų. Raktai kaip ir anksčiau parenkami su vienodomis tikimybėmis. Dabar iš viso galimi devyni žodžiai, o raktų tik šeši. Taigi sistema visiškai saugi jau nebegali būti.

Kadangi rutuliai traukiami be grąžinimo, tai pranešimų tikimybes skaičiuojame taip:

$$P(\mathcal{M}=BB)=\frac{5\cdot4}{12\cdot11}=\frac{20}{132},\quad P(\mathcal{M}=BR)=\frac{5\cdot3}{12\cdot11}=\frac{15}{132},\ldots$$

Pranešimo besąlyginė entropija lygi

$$H(\mathcal{M})\approx3.0968.$$

Šifrų tikimybės yra dabartiniu atveju tokios:

$$\begin{aligned}P(\mathcal{C}=BB) &= P(\mathcal{C}=JJ)=P(\mathcal{C}=RR)=\frac{19}{198},\\ P(\mathcal{C}=BJ) &= \ldots=P(\mathcal{C}=RR)=\frac{47}{396}.\end{aligned}$$

Apskaičiavę gautume

$$\begin{aligned}&H(\mathcal{M}|\mathcal{C}=BB)=H(\mathcal{M}|\mathcal{C}=JJ)=H(\mathcal{M}|\mathcal{C}=RR)\approx1.433,\\ &H(\mathcal{M}|\mathcal{C}=BJ)=\ldots=H(\mathcal{M}|\mathcal{C}=RJ)\approx2.5533.\end{aligned}$$

Sąlyginė pranešimo entropija dabar yra

$$H(\mathcal{M}|\mathcal{C})=3\cdot H(\mathcal{M}|\mathcal{C}=BB)P(\mathcal{C}=BB)+6\cdot H(\mathcal{M}|\mathcal{C}=BJ)P(\mathcal{C}=BJ)\approx2.2308.$$

Taigi mūsų kriptoanalitikas, sugavęs šifrą, gauna maždaug vieną bitą informacijos apie siunčiamą užšifruotą pranešimą. Kaip jis tą bitą panaudos – ne mūsų reikalas.

15.3. Rakto įminimo taškas

Kuo ilgesnis šifras, tuo daugiau jame netiesioginės informacijos apie šifruotą pranešimą bei naudotą raktą. Kokio ilgio šifro užtenka, kad pavienio šifro ataka būtų įmanoma, t. y. kad, naudojantis juo, būtų galima nustatyti raktą? Įmanoma teoriškai, žinoma, nereiškia, kad praktiškai lengva tai padaryti.

Jeigu raktų aibė lieka ta pati, o pranešimų aibė didėja, tai kriptosistema darosi vis mažiau saugi. Tai pastebėjome ir iš ankstesnių skyrelių pavyzdžių. Panagrinėsime šią mintį detaliau.

Tarkime, siunčiami pranešimai yra parašyti lietuviškos 32 simbolių abėcėlės $\mathcal{A}$ raidėmis, o šifravimui naudojama Cezario kriptosistema, kurios raktai renkami su vienodomis tikimybėmis.

Jeigu šifruojamus pranešimus sudaro pavienės raidės, tai Cezario kriptosistema yra visiškai saugi.

Dabar tarkime, kad pranešimų aibę $\mathcal{M}_2$ sudaro dviejų raidžių lietuviški žodžiai. Gavęs šifrą $c = c_1c_2$, kriptoanalitikas gali bandyti visus raktus vieną po kito. Šitaip jis gautų 32 simbolių poras $m = d(c_1|k_i)d(c_2|k_i)$. Žinodamas, kokia kalba buvo parašytas tekstas, jis gali atmesti beprasmes kombinacijas ir gauti pranešimų bei raktų, kurie galėjo būti panaudoti, aibes.

Jeigu pranešimų aibę $\mathcal{M}_N$ sudaro N simbolių ilgio tekstai, kur N yra didelis skaičius, tai, sugavęs šifrą c_N ir išbandęs visus raktus, kriptoanalitikas ko gero gautų tik vieną galimą prasmingą tekstą $d(c_N|k_i)$. Taigi raktas vienareikšmiškai atkuriamas pagal šifrą: $\mathcal{K} = f(\mathcal{C}_N)$. Tačiau tada $H(\mathcal{K}|\mathcal{C}_N) = 0$, t. y. kriptosistemos slaptumas išnyksta, ji įmenama.

Šiame pavyzdyje padarėme prielaidą, kad kriptoanalitikas gali vertinti tekstus kaip prasmingus ir neprasmingus. Tačiau iš pavyzdžių matėme, kad kriptosistema silpnėja ir tada, kai prasmės faktoriaus nėra.

Nagrinėkime kokią nors kriptosistemą $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$, tegu šifrus sudaro abėcėlės $\mathcal{A}$ pavieniai simboliai. Interpretuodami $\mathcal{M}, \mathcal{K}, \mathcal{C}$ kaip atsitiktinius dydžius, galime apibrėžti jų entropijas $H(\mathcal{M})$, $H(\mathcal{K})$, $H(\mathcal{C})$. Kaip visada, tarsime, kad $\mathcal{M}, \mathcal{K}$ yra nepriklausomi, o $\mathcal{C} = e(\mathcal{M}|\mathcal{K})$. Taigi

$$H(\mathcal{M}, \mathcal{K}) = H(\mathcal{M}) + H(\mathcal{K}), \quad H(\mathcal{C}|\mathcal{M}, \mathcal{K}) = 0.$$

Tada

$$H(\mathcal{M}, \mathcal{K}, \mathcal{C}) = H(\mathcal{M}, \mathcal{K}) + H(\mathcal{C}|\mathcal{M}, \mathcal{K}) = H(\mathcal{M}) + H(\mathcal{K}).$$

Panaudoję šį sąryšį lygybėje (78), gausime

$$H(\mathcal{K}|\mathcal{C}) = H(\mathcal{M}) + H(\mathcal{K}) - H(\mathcal{C}). \tag{79}$$

Dabar apibrėžkime naujas kriptosistemas $\langle \mathcal{M}_N, \mathcal{K}, \mathcal{C}_N \rangle$, kur pranešimų ir šifrų aibės yra $\mathcal{M}_N = \mathcal{M}^N, \mathcal{C}_N = \mathcal{C}^N$, raktai renkami su tomis pačiomis tikimybėmis, o šifravimo procedūros apibrėžiamos taip:

$$e(m_1m_2 \dots m_N|K) = e(m_1|K)e(m_2|K) \dots e(m_N|K).$$

Tada šioms kriptosistemoms (79) lygybė irgi teisinga:

$$H(\mathcal{K}|\mathcal{C}_N) = H(\mathcal{M}_N) + H(\mathcal{K}) - H(\mathcal{C}_N).$$

Jau matėme, kad entropija $H(\mathcal{K}|\mathcal{C}_N)$ gali mažėti augant N.

95 apibrėžimas. *Jei N yra mažiausias skaičius, tenkinantis lygybę*

$$H(\mathcal{M}_N) + H(\mathcal{K}) - H(\mathcal{C}_N) = 0, \tag{80}$$

tai jį vadinsime rakto įminimo tašku.

Čia apibrėžtas rakto įminimo taškas angliškoje literatūroje vadinamas *unicity point*. Kartais vietoj lygybės reikalaujama, kad reiškinys būtų ne didesnis už 1. Jeigu $H(\mathcal{K}|\mathcal{C}_N) \leq 1$, tai iš esmės reiškia, kad raktą galima nustatyti naudojantis vien šifru.

Taigi rakto įminimo taškas – tai trumpiausio šifro, iš kurio galima nustatyti raktą, ilgis. Šį skaičių turėtume rasti iš (80) lygybės. Tačiau kaip tai padaryti? Pasielkime lyg fizikai: negalėdami išspręsti bendros lygties, suformuluokime sveikam protui priimtinas sąlygas, kurios palengvina rakto įminimo taško lygties sprendimą.

Visų pirma tarkime, kad mūsų pranešimus $\mathcal{M}_N$ generuoja šaltinis, turintis entropiją H. Tada galime manyti, kad dideliems skaičiams N

$$H(\mathcal{M}_N) = NH(\mathcal{M}) = NH.$$

Jei, pavyzdžiui, pranešimai yra kokios nors kalbos tekstai, tai H yra kalbos entropija, kurią galima suskaičiuoti naudojantis simbolių pasirodymo tikimybių lentelėmis. Toliau, tarkime, kad visi aibės $\mathcal{A}^N$ elementai turi vienodas galimybes pasirodyti kaip šifrai. Tokią savybę matėme nagrinėtuose paprastuose pavyzdžiuose. Tada

$$H(\mathcal{C}_N) = N \log_2 |\mathcal{A}|.$$

Pagaliau jei visi raktai renkami su vienodomis tikimybėmis, tai

$$H(\mathcal{K}) = \log_2 |\mathcal{K}|.$$

Su šiomis prielaidomis rakto įminimo taško lygtį (80) galima išspręsti:

$$N = \frac{\log_2 |\mathcal{K}|}{\log_2 |\mathcal{A}| - H}.$$

Gavome gerokai supaprastintos rakto įminimo lygties sprendinį. Jis pateikia praktikai reikalingą įvertį, rodantį, kada kriptosistemos naudojimas jau tampa nebesaugus. Lygties su nurodytomis sąlygomis sprendimas duoda tik apytikslę įminimo taško reikšmę[10], nei tikslus sprendimas, bet tai juk nėra didelis trūkumas.

[10] Tikslios lygtys gali iš viso neturėti sprendinių.

Rakto įminimo taškas nurodo, kokio ilgio šifrą reikia turėti, kad būtų efektyvi pavienio šifro ataka. Tačiau ši ataka atliekama vykdant perranką! Taigi praktiškai ją įvykdyti gali būti labai sudėtinga.

Panagrinėkime konkretų kriptosistemos pavyzdį. Imkime kokią nors skaičių aibės perstatą (bijekciją)

$$k : \{1, 2, , \ldots, d\} \to \{1, 2, , \ldots, d\}$$

ir apibrėžkime tokią pranešimo M, parašyto abėcėlės $\mathcal{A}$ simboliais, šifravimo taisyklę

$$\text{jei } M = m_1 m_2 \ldots m_d m_{d+1} m_{d+2} \ldots m_{2d}, \ldots,$$

$$\text{tai } e(M, k) = m_{k(1)} m_{k(2)} \ldots m_{k(d)} m_{d+k(1)} m_{d+k(2)} \ldots m_{d+k(d)} \ldots$$

Taigi pranešimas skaidomas d ilgio žodžiais ir jie vienas po kito šifruojami. Todėl galime manyti, kad teksto abėcėlė iš tikrųjų yra $\mathcal{A}^d$. Tada rakto įminimo taško reikšmė nurodys, kiek d ilgio šifro žodžių reikia raktui įminti. Kadangi raktų skaičius tokioje sistemoje yra lygus $d!$, tai iš rakto įminimo lygties gauname:

$$N = \frac{\log_2 d!}{\log_2 |\mathcal{A}^d| - H_d} = \frac{\log_2 d!}{d(\log_2 |\mathcal{A}| - H)},$$

čia H_d yra teksto, kurį interpretuojame kaip sudarytą iš d ilgio žodžių, entropija, o H – teksto, sudaryto iš pavienių simbolių, entropija, $H_d \approx dH$. Lietuviškoje abėcėlėje yra 32 raidės, todėl $\log_2 |\mathcal{A}| = 5$; lietuviškų tekstų entropija $H \approx 3$. Taigi įminimo taško reikšmė

$$N \approx \frac{\log_2 d!}{2d}.$$

Paskaičiavę su nedidelėmis d reikšmėmis (N reikšme imame mažiausią sveikąjį skaičių, didesnį už reiškinio reikšmę), gautume $N = 1$, kai $d \leq 8$, ir $N = 2$, kai $9 < d \leq 20$.

16 Blokiniai šifrai

Blokiniais šifrais vadinsime kriptosistemas, kurių šifravimo algoritmai transformuoja fiksuoto ilgio teksto žodžius (blokus) į tokio pat ilgio šifro žodžius. Raktas, valdantis šifravimo operaciją, irgi paprastai renkamas iš fiksuoto ilgio žodžių aibės.

Dvejetainė abėcėlė $\mathcal{B} = \{0, 1\}$ yra pati svarbiausia kriptografijoje, todėl tik ją ir naudosime.

96 apibrėžimas. *Kriptosistemą, kurios tekstų, šifrų ir raktų aibės yra sudarytos iš fiksuoto ilgio žodžių:* $\mathcal{M} = \mathcal{C} = \mathcal{B}^n, \mathcal{K} = \mathcal{B}^k$, *vadinsime blokine kriptosistema arba tiesiog – blokiniu šifru.*

Taigi blokinės kriptosistemos šifravimo ir dešifravimo taisyklės yra funkcijos, kurių argumentai – dvejetainių žodžių poros:

$$e(\cdot|\cdot), d(\cdot|\cdot) : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n.$$

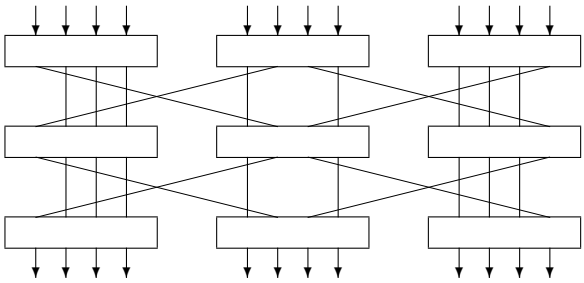
Turint blokinę kriptosistemą, reikia nuspręsti, kaip ja bus šifruojami ilgi duomenų srautai. Pasirinkti galima įvairiai. Tie būdai kriptografijoje vadinami blokinių šifrų naudojimo režimais. Svarbiausius iš jų aptarsime kiek vėliau.

Apskritai blokinės kriptosistemos gali būti tiek simetrinės, tiek viešojo rakto. Tačiau paprastai taip vadinamos simetrinės kriptosistemos.

16.1. Dvi schemos

Du svarbūs šiuolaikinių blokinių kriptosistemų projektavimo principai.

Teksto žodį galime pakeisti kitu (atlikti keitinį), galima sumaišyti jo simbolius (atlikti perstatą). Tokios pavienės transformacijos, žinoma, nesukurs saugaus šifro. Viena iš C. Shannono idėjų, kuria remiasi ir mūsų laikų kriptosistemų kūrėjai, yra labai paprasta: teksto blokui reikia taikyti tokią perstatų ir keitinių seką, kad kiekvienas gautojo šifro bitas priklausytų nuo kiekvieno teksto ir rakto bito, t. y. pakeitus net vienintelį teksto ar rakto bitą, šifro žodis labai pasikeistų. Šitaip sukonstruota daug blokinių šifrų: kol gaunamas šifras, teksto žodis praeina keletą ciklų (arba iteracijų), kur jam taikomos perstatos ir keitiniai. Kiekvieno ciklo transformacijas valdo daliniai raktai, gaunami iš kriptosistemos rakto pagal tam tikrą taisyklę. Tokia schema kriptografijoje vadinama keitinių-perstatų tinklu (Substitution-Permutation Network, (PSN) angl.).



Keitinių-perstatų tinklo schema. Dažnai cikle apdorojamas simbolių blokas skaidomas į mažesnius ir keitiniai taikomi pastariesiems. Struktūriniai schemos elementai, kurie skirti keitiniams atlikti, kriptografijoje vadinami S-dėžėmis.

Daugelio gerų blokinių šifrų konstrukcijoje taikoma Horsto Feistelio struktūrinė schema, kurią jis, dirbdamas IBM, panaudojo LUCIFER kriptosistemai. Feistelio struktūros blokiniuose šifruose transformacijos atliekamos su lyginio ilgio duomenų blokais.

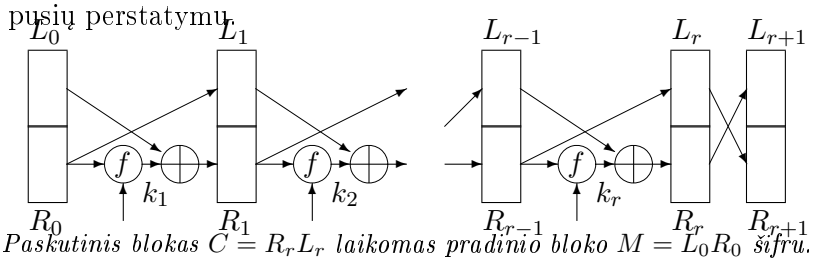
Tegu $M_0 = m_1 m_2 \dots m_n m_{n+1} \dots m_{2n}$ yra pradinis tekstas. Jo šifras gaunamas po r Feistelio iteracijų, kurias valdo daliniai raktai $k_1, k_2, \dots k_r$:

$$\begin{array}{ccccccccccc} & & \downarrow\ k_1 & & \downarrow\ k_2 & & & & \downarrow\ k_r & & \\ M = M_0 & \longrightarrow & M_1 & \longrightarrow & M_2 & \longrightarrow \dots \longrightarrow & M_{r-1} & \longrightarrow & M_r = C. \end{array}$$

Atliekant vieną iteraciją, blokas M_j dalijamas į dvi vienodo ilgio dalis – kairiąją ir dešiniąją – ir pertvarkomas taip:

$$M_i = L_i R_i \longrightarrow M_{i+1} = L_{i+1} R_{i+1}, \quad L_{i+1} = R_i,\ R_{i+1} = L_i \oplus f(R_i, k_{i+1}),$$

čia $\oplus$ žymi blokų sudėtį moduliu 2, o f yra funkcija, iš dvejetainės abėcėlės n ilgio žodžio ir rakto sukurianti naują n ilgio žodį. Naudinga r iteracijų grandinę papildyti dar vienu paprastu pertvarkiu: kairiosios ir dešiniosios blokų pusių perstatymu.



Paskutinis blokas $C = R_r L_r$ laikomas pradinio bloko $M = L_0 R_0$ šifru.

Taigi visa blokų pertvarkymo grandinė atrodo taip:

$$\begin{aligned} L_1 &= R_0,\ R_1 = L_0 \oplus f(R_0, k_1) \\ L_2 &= R_1,\ R_2 = L_1 \oplus f(R_1, k_2) \\ &\dots\dots\dots\dots\dots\dots\dots\dots \\ L_r &= R_{r-1}, R_r = L_{r-1} \oplus f(R_{r-1}, k_r) \\ L_{r+1} &= R_r, R_{r+1} = L_r. \end{aligned}$$

Kam gi reikalingas tas paskutinis žingsnis? Tuoj įsitikinsime, kad tai gudrus sumanymas.

Taigi po visų pertvarkymų gavome pradinio bloko $M = L_0R_0$ šifrą $C = L'_0R'_0 = R_rL_r$. Įsivaizduokime, kad tą patį pertvarkymų ciklą atliekame su C, tik raktus naudojame atvirkščia tvarka: $k_r, k_{r-1}, \ldots, k_1$. Atlikę pirmąjį žingsnį, gausime bloką $C_1 = L'_1R'_1$, čia

$$\begin{aligned} L'_1 &= R'_0 = L_r = R_{r-1}, \\ R'_1 &= L'_0 \oplus f(R'_0, k_r) = R_r \oplus f(R_{r-1}, k_r) \\ &= L_{r-1} \oplus f(R_{r-1}, k_r) \oplus f(R_{r-1}, k_r) = L_{r-1}. \end{aligned}$$

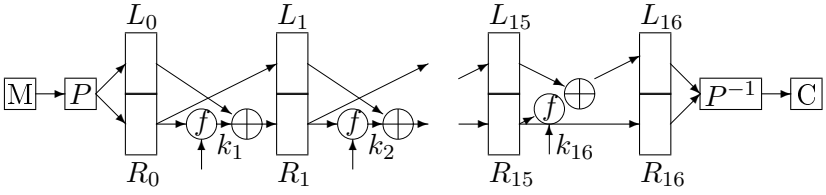
Taigi $C_1 = R_{r-1}L_{r-1}$. Atlikę r iteracijų, gautume $C_r = R_0L_0$, paskutiniame žingsnyje sukeitę bloko puses vietomis, gautume M. Šifravimo iteracijų seka tinka ir dešifravimui – tik raktus reikia naudoti „atbuline“ tvarka. Ir jokių reikalavimų funkcijai f!

16.2. DES

DES, arba Data Encryption Standard, yra kriptografijos mokslo brandos ženklas. Tai pirmoji kriptosistema valstybės institucijos įvertinta ir pripažinta šifravimo standartu.

1973 metais JAV vyriausybė nusprendė, kad metas standartizuoti informacijos srautų apdorojimo metodus. Užduotis parengti šiuos standartus teko Nacionaliniam standartų institutui (NBS – National Bureau of Standards). Vienas iš daugelio tikslų, kurie buvo keliami, – parengti duomenų šifravimo standartą. Buvo paskelbtas konkursas, bet jo rezultatai iškeltų sąlygų netenkino. Konkursas buvo pakartotas. Vieną iš pasiūlymų pateikė IBM. Siūloma kriptosistema buvo sukurta naudojantis H. Feistelio kriptosistemos LUCIFER idėjomis. Svarstymai truko pusantrų metų, o konkursas pasibaigė IBM pergale. Jų pasiūlymas tapo pirmuoju pasaulyje šifravimo standartu DES (Data Encryption Standard, patvirtinta 1976 metais).

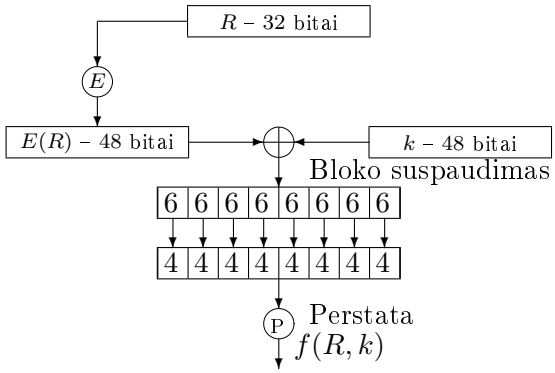
DES yra Feistelio struktūros blokinė kriptosistema, šifruojanti 64 bitų ilgio blokus ir naudojanti 56 bitų ilgio raktus. Jos struktūrinė schema yra tokia:



DES kriptosistemos schema. IP žymi tam tikrą pradinių duomenų perstatą, o IP^{-1} – šiai perstatai atvirkštinę.

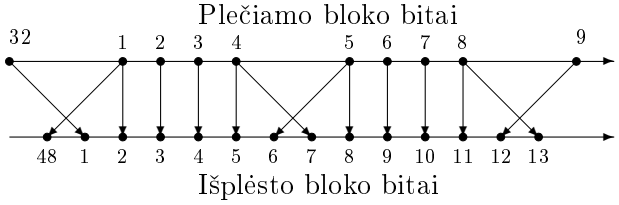
DES naudoja 16 iteracijų ir 56 bitų ilgio raktą. Formaliai rakto ilgis yra toks pat kaip ir blokų – 64 bitai. Tačiau aštuoni bitai sudarant dalinius

raktus nedalyvauja, jie yra tiesiog kontroliniai. Paskutinės iteracijos sudarytas blokas R_{16} sudaro kairiąją šifro pusę, L_{16} – dešiniąją (prisiminkime aptartąją Feistelio schemos pabaigą!), perstatos IP ir IP^{-1} yra fiksuotos, t. y. nepriklauso nuo rakto, todėl kriptosistemos saugumas nuo jų nepriklauso. Jos įtrauktos į schemą techniniais sumetimais – norėta, kad duomenys į DES mikroschemą būtų įkeliami greičiau. Taigi visa esmė glūdi f dėžėse. Kas gi jose vyksta?



Transformacijų, atliekamų DES f-dėžėje schema.

Pirmas faktas: į šią dėžę įvedamas 32 bitų ilgio duomenų blokas ir 48 bitų – dalinis raktas. Pirma operacija skirta duomenų blokui išplėsti iki tokio pat ilgio kaip raktas. Tai daroma paprastai – keletas bitų panaudojami du kartus.



Duomenų išplėtimo operacija DES dėžėje.

Tada ateina laikas pasinaudoti raktu. Išplėstas 48 bitų raktas tiesiog sudedamas moduliu 2 su daliniu raktu (XOR operacija) ir gaunamas naujas 48 bitų blokas. Tačiau iš dėžės turi išeiti 32 bitai, taigi reikia bloko ilgį sumažinti. Tiesiog nubraukti šešiolika bitų būtų prastas sprendimas. DES kūrėjai sugalvojo kitaip: 48 bitai paskirstomi po šešis ir kiekvienas šešetukas eina į savo dėžutę. Iš kiekvienos dėžutės išeina tik 4 bitai. Šitaip gaunamas reikalingo ilgio duomenų blokas. Kiekvienai iš aštuonių dėžučių DES kūrėjai sudarė 4×16 dydžio lentelę, kurios elementai – keturiais bitais užrašomi natūriniai skaičiai (taigi skaičiai nuo 1 iki 15). Jeigu į dėžutę įeina bitų šešetas $b_1 b_2 \ldots b_6$, tai skaičius $e = b_1 + 2b_6$ nurodo lentelės eilutės numerį, o

$s = b_2 + 2b_3 + 4b_4 + 8b_5$ – stulpelio. Vadinasi, iš šios dėžutės turi būti išvestas skaičius (keturi bitai) užrašytas e-ojoje eilutėje ir s-ajame stulpelyje! Štai šitaip veikia DES. Kam įdomu, kaip sudarytos tos lentelės bei kaip iš 56 bitų rakto sudaroma 16 dalinių raktų, pavartykite DES aprašymą. [11]

DES buvo naudojamas beveik trisdešimt metų. Ne tik naudojamas, tačiau ir nuodugniai tyrinėjamas. Konstrukcija pasirodė esanti tokia gera, kad iš esmės nebuvo rasta praktiškai reikšmingų saugumo spragų. Ir vis dėlto – mūsų dienomis DES nebegalima laikyti saugiu šifru. Kodėl? Nes gerokai išaugo kompiuterių galia. Paprastas raktų perrankos atakas, kurioms nepakako išteklių tuomet, kai DES buvo sukurtas, dabar jau galima vykdyti.

Perrankos ataka yra teksto-šifro porų ataka. Jeigu žinomas tam tikras kiekis nešifruotų tekstų $M_1, M_2, \ldots, M_r$ ir juos atitinkančių šifrų $C_1 = e(M_1|K)$, $C_2 = e(M_2|K), \ldots, C_r = e(M_r|K)$, gautų panaudojus nežinomą raktą K, tai galima bandyti visus galimus raktus ir tikrinti lygybes:

$$d(C_i|K) \stackrel{?}{=} M_i,\ i = 1, 2, \ldots, r;\ K \in \mathcal{K}.$$

Blogiausiu atveju tektų atlikti maždaug 2^{56} tokių tikrinimų. Tai didžiulis skaičius, tačiau dabartiniai kompiuteriai irgi labai galingi.

Tačiau ir su DES dar galima pasiekti tinkamą saugumo lygį, tiksliau su 3DES. Skaičius 3 reiškia, kad DES taikomas tris kartus:

$$C = e(d(e(M|K_1)|K_2)|K_1).$$

Jeigu $K_1 = K_2$, tai toks šifravimas tolygus DES šifravimui su vienu raktu. Kai $K_1 \neq K_2$, gauname algoritmą, kurio rakto ilgis yra 112 (raktų K_1 ir K_2 ilgių suma).

16.3. AES

1997 JAV Nacionalinis standartų ir technologijų institutas (NIST – National Institute of Standards and Technology) paskelbė naują konkursą garbingojo DES vietai užimti. Atsirado 15 kandidatų. Į finalą išėjo šie: MARS, RC6, Rijndael, Serpent ir Twofish.

O nugalėjo Rijndael. Šifro pavadinimas „Rijndael“ sudarytas sujungus jo kūrėjų – dviejų belgų kriptografų V. Rijmen ir J. Daemen – pavardžių skiemenis. Jų šifrui ir buvo suteiktas titulas AES (Advanced Encryption Standard). Standartas paskelbtas 2001 metais[12], po 5 metus trukusio vertinimo proceso. Tačiau tai nereiškia, kad tai vienintelis geras šifras iš penkių finalininkų. Kai kurie iš jų pagal atskirus kriterijus netgi geresni. Tačiau tų kriterijų

[11] FIPS 46-3, Data Encryption Standard (DES).http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

[12] ADVANCED ENCRYPTION STANDARD (AES). Federal Information Processing Standards Publication 197 November 26, 2001.

buvo daug: saugumas, operacijų greitis, diegimo paprastumas, struktūrinės savybės... Iš kitų blokinių šifrų AES išsiskiria savo struktūros paprastumu bei „matematiškumu“ . Prisiminę DES dėžes, negalėtume paaiškinti, kokiais matematiniais dėsniais remtasi jas sudarant. O štai visos AES transformacijos labai paprastai aprašomos naudojantis matematiniais objektais. Tai privalumas, o galbūt ir trūkumas. Juk niekas negali užginčyti, kad sudėtingiems, bet tiksliai suformuluotiems matematiniams uždaviniams gali atsirasti ir elegantiški bei efektyvūs sprendimai. Kitaip tariant, naujos matematinės AES atakos... Tačiau kol kas tai tik svarstymai. Geriau panagrinėkime AES struktūrą.

AES kriptosistema atlieka veiksmus su baitais. Kiekvieną baitą, t. y. aštuonių bitų žodį $b = b_7b_6b_5b_4b_3b_2b_1b_0$, interpretuokime kaip erdvės $\mathbb{F}_{2,8}[x]$ daugianarį

$$b(x) = b_7x^7 + b_6x^6 + \ldots + b_1x + b_0.$$

Du tokius daugianarius sudėję, vėl gausime tos pačios erdvės daugianarį. Taigi du baitus galime sudėti; tiesą sakant, ta sudėtis – įprastinė žodžių sudėtis moduliu 2 (XOR operacija) ir tiek. Imkime aštunto laipsnio daugianarį

$$f(x) = x^8 + x^4 + x^3 + x + 1$$

ir apibrėžkime daugianarių iš $\mathbb{F}_{2,8}[x]$ (baitų) sandaugą, kaip tai darėme nagrinėdami kūnų plėtinius.

$$a(x) \times_f b(x) = a(x)b(x) \text{ dalybos iš } f(x) \text{ liekana.}$$

Taigi dabar turime du veiksmus su baitais: sudėtį ir daugybą. Su šiais veiksmais baitų aibė sudaro žiedą. O dabar gera naujiena – daugianaris $f(x)$ yra neskaidus virš kūno $\mathbb{F}_2$. Tai reiškia (žvilgtelėkite į daugianariams skirtą skyrelį), kad daugianarių erdvė $\mathbb{F}_{2,8}[x]$ su sudėties ir daugybos operacijomis sudaro kūną, turintį $2^8 = 256$ elementus. Šis kūnas yra izomorfiškas kūnui $\mathbb{F}_{2^8}$. Taigi veiksmus su baitais, kuriuos naudoja AES, galime interpretuoti kaip veiksmus su kūno $\mathbb{F}_{2^8}$ elementais.

DES kriptosistemą galima naudoti su fiksuotu duomenų bloko ir rakto ilgiu. AES yra šiuo požiūriu kiek lankstesnė: galima naudoti kelis kriptosistemos variantus su skirtingais rakto ilgiais ir skirtingu iteracijų skaičiumi. Kriptosistemos kūrėjai taip pat numatė galimybę naudoti ir įvairius duomenų bloko ilgius (128, 160, 192, 224 ir 256 bitų), tačiau standartizuotas tik variantas su 128 bitų ilgio duomenų blokais.

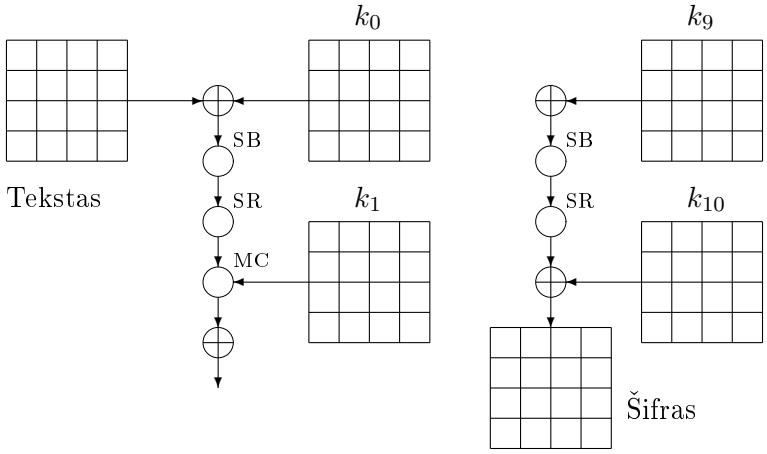
	Rakto ilgis	Bloko ilgis	Iteracijų skaičius
AES-128	128	128	10
AES-192	192	128	12
AES-256	265	128	14

Pradinių duomenų blokas prieš pradedant transformacijas surašomas į lentelę.

s_{00}	s_{01}	s_{02}	s_{03}
s_{10}	s_{11}	s_{12}	s_{13}
s_{20}	s_{21}	s_{22}	s_{23}
s_{30}	s_{31}	s_{32}	s_{33}

AES duomenų blokas; lentelės elementai – aštuonių bitų ilgio žodžiai (baitai).

Šifruojant su šia lentele atliekamos keturių rūšių operacijos: baitų keitimo, eilučių postūmio, stulpelių maišymo ir sudėties su iteracijos raktu. Dešifruojant vykdomos analogiškos (bet ne tokios pat) operacijos. Bendra AES-128 algoritmo schema atrodo taip:



AES-128 sudaro 10 vienodos struktūros žingsnių. Kiekvienai operacijai iš kriptosistemos bendro rakto sudaromas dalinis raktas. Šifravimas prasideda sudėties su pradžios raktu veiksmu. Pirmieji devyni žingsniai vienodi – atliekamos baitų keitimo (SB), eilučių postūmio (SR) ir stulpelių maišymo (MC) operacijos. Paskutiniame žingsnyje stulpelių maišymo nėra.

Baitų keitimo operacija kiekvieną lentelės baitą pakeičia nauju. Kiekvienas baitas (interpretuojamas kaip kūno $\mathbb{F}_{2^8}$ elementas) keičiamas atvirkštiniu, kuriam dar be to atliekama tam tikra nesudėtinga transformacija. Nulinis baitas neturi atvirkštinio, todėl jis nekeičiamas. Baito $b_0b_1b_2b_3b_4b_5b_6b_7$ keitimo baitu $b'_0b'_1b'_2b'_3b'_4b'_5b'_6b'_7$ operaciją galima užrašyti matricine forma

taip:

$$\begin{pmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}.$$

Baitų keitimo operacijos matricinė išraiška. Antroji matricos eilutė gauta iš pirmosios, atlikus jos elementų postūmį, analogiškai iš antrosios eilutės gaunama trečioji ir t. t.

Eilučių postūmio operacija yra pati paprasčiausia: duomenų eilutės baitai cikliškai perstumiami ir tiek.

s_{00}	s_{01}	s_{02}	s_{03}
s_{10}	s_{11}	s_{12}	s_{13}
s_{20}	s_{21}	s_{22}	s_{23}
s_{30}	s_{31}	s_{32}	s_{33}

$\longrightarrow$

s_{00}	s_{01}	s_{02}	s_{03}
s_{11}	s_{12}	s_{13}	s_{10}
s_{22}	s_{23}	s_{20}	s_{21}
s_{33}	s_{30}	s_{31}	s_{32}

Eilučių postūmio operacija

Kiek sudėtingesnė yra stulpelių maišymo operacija. Ji atliekama su kiekvienu duomenų lentelės stulpeliu.

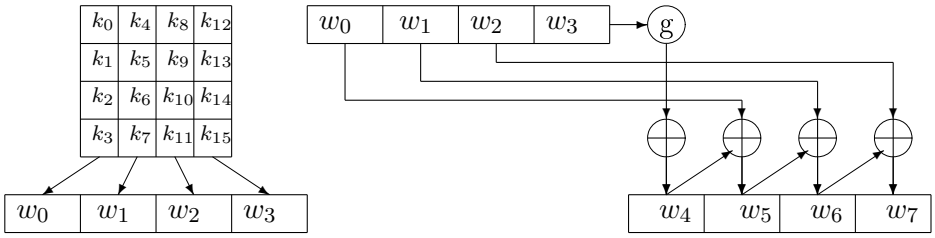
Šią operaciją galima apibrėžti naudojant daugianarių veiksmus arba matricinę lygybę

$$\begin{pmatrix} s'_{0c} \\ s'_{1c} \\ s'_{2c} \\ s'_{3c} \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \cdot \begin{pmatrix} s_{0c} \\ s_{1c} \\ s_{2c} \\ s_{3c} \end{pmatrix}.$$

AES stulpelių maišymo operacijos matricinė išraiška. Matricos elementus reikia interpretuoti kaip šešioliktaine sistema užrašytus baitus. Baitų daugyba atliekama interpretuojant juos kaip kūno $\mathbb{F}_{2^8}$ elementus.

Paskutinė – sudėties su raktu – operacija – tai paprasta matricų su elementais iš $\mathbb{F}_{2^8}$ sudėties operacija.

Štai ir viskas. Trūksta tik dalinių raktų sudarymo iš kriptosistemos rakto algoritmo ir galima pradėti programuoti!



Vienas rakto išplėtimo algoritmo ciklas. Pradinis AES raktas išsaugomas žodžiuose w_0, w_1, w_2, w_3, o naudojantis jais sukuriami dar keturi žodžiai

Pradinis AES 128 bitų raktas išsaugomas keturiuose 32 bitų žodžiuose w_0, w_1, w_2, w_3, naudojantis jais, sukuriami dar keturi žodžiai, naudojantis pastaraisiais – dar keturi... Iš viso sukuriami 44 žodžiai $w_i, i = 0, 1, \ldots, 43$. Pradinei sudėties su raktu operacijai panaudojami pirmieji keturi žodžiai, antrajai sudėčiai raktas sudaromas iš sekančių žodžių ir t. t. Rakto išplėtimo schemoje naudojama funkcija g. Ji veikia taip:

$$g(w) = SB(rot(w)) + R_j,$$

čia rot baitų postūmio operacija, $rot(b_0b_1b_2b_3) = b_1b_2b_3b_0$; SB – baitų keitimo operacija, apibrėžta anksčiau, o R_j – j-ojo žingsnio konstanta. Šios konstantos sudaromos pagal paprastą taisyklę (baitus dauginame interpretuodami juos kaip $\mathbb{F}_{2^8}$ elementus):

$$R_j = r_j000000, \quad r_1 = 01, \quad r_j = 2 \cdot r_{j-1}.$$

Dabar tai jau tikrai viskas apie AES. Kuo vadovavosi šios kriptosistemos kūrėjai, kodėl panaudojo tokias operacijas, o ne kitokias? Tikriausiai išbandė ne vieną. Pagrindinis kriterijus – kad kriptosistema atlaikytų visas žinomas atakas ir, žinoma, veiktų greitai. Struktūros paprastumas irgi svarbus kriterijus. Juk sudėtingas schemas sudėtinga ir analizuoti, vadinasi, sudėtinga ir vertinti.

16.4. Penki režimai

Turite gerą blokinį šifrą? Laikas nuspręsti, kam ir kaip jį naudoti...

Blokinė kriptosistema šifruoja ir dešifruoja vieną fiksuoto ilgio žodį:

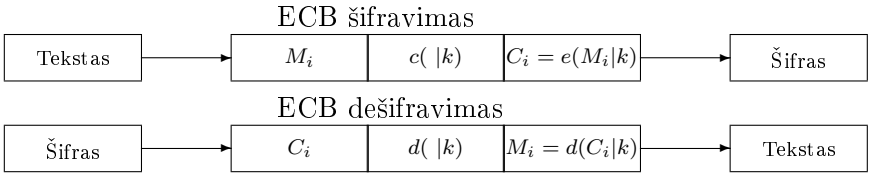
$$C = e(M|K), \; M = d(C|K) \quad M, C \in \{0,1\}^n.$$

Tačiau duomenis, kuriuos norime apsaugoti, sudaro daugybė žodžių. Kaip taikyti kriptosistemą tokiam srautui? Kriptografai žino keletą būdų.

Elektroninė kodų knyga (ECB, Electronic Codebook)

Tai pats paprasčiausias blokinės kriptosistemos naudojimo būdas. Savo duomenų srautą skaidote į vienodo ilgio žodžius ir juos vieną po kito šifruojate:

$$M_1 M_2 \ldots \mapsto C_1 C_2 \ldots, \quad C_j = e(M_j|K).$$



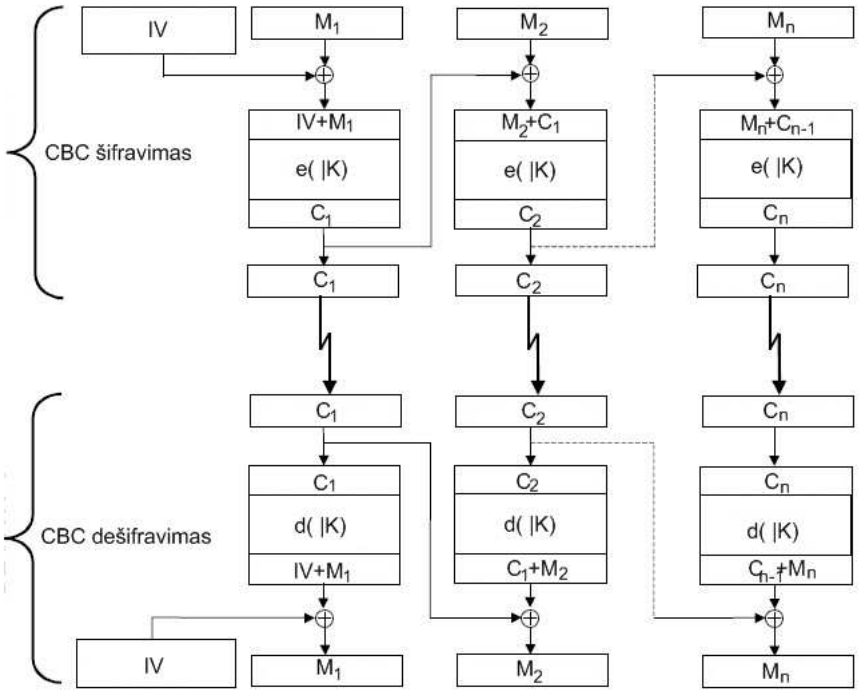
Duomenų šifravimas ir dešifravimas blokine kriptosistema ECB režimu.

Paprastumas – turbūt vienintelis šio režimo privalumas. O trūkumų daug. Pavyzdžiui, vienodi duomenų srauto žodžiai visada užšifruojami vienodai. Jeigu kas nors pašalins kelis šifro srauto žodžius – nepastebėsite. Žymūs kriptografai N. Fergusonas ir B. Shneieris savo knygoje „Praktinė kriptografija“ rašo taip: minime jį tik todėl, kad žinotumėte, kokio režimo niekada nereikia naudoti.

Šifrų blokų grandinės režimas (CBC, Cipher Block Chaining)

Naudojantis šiuo režimu, kiekvieno žodžio M_i šifras priklauso nuo anksčiau šifruotų žodžių:

$$\begin{aligned} C_1 &= e(M_1 \oplus IV|K),\ C_i = e(M_i \oplus C_{i-1}|K),\ i \geq 2, \\ M_1 &= d(C_1 \oplus K|IV), M_i = e(C_i|K) \oplus C_{i-1},\ i \geq 2. \end{aligned}$$



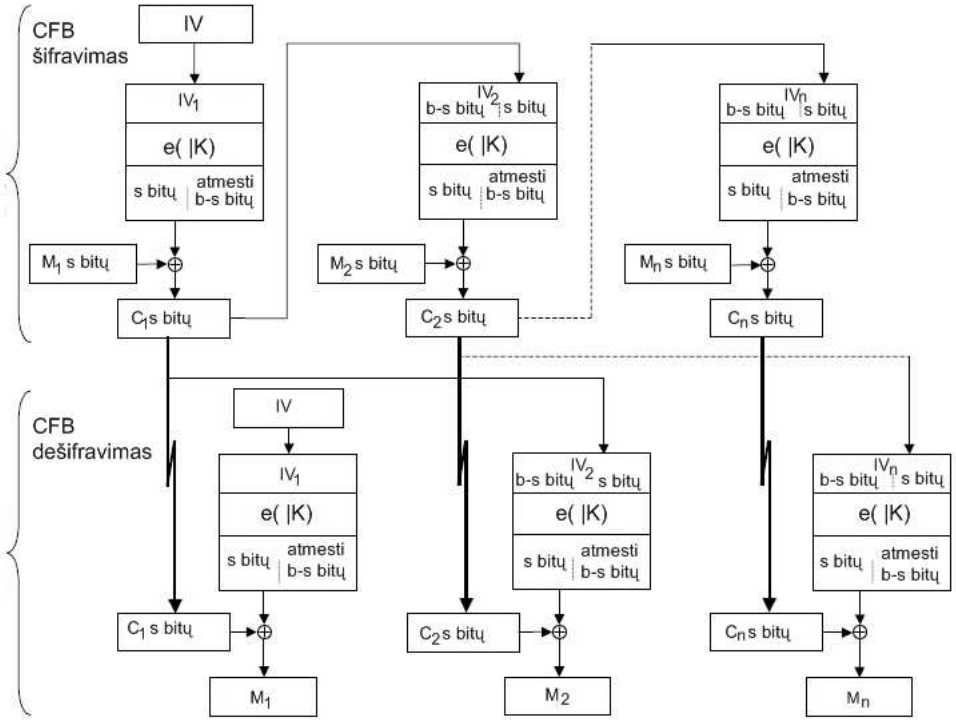
Duomenų šifravimas ir dešifravimas blokine kriptosistema CBC režimu.

Pirmajam blokui šifruoti naudojamas žodis IV – pradžios (kriptografai sako – inicializacijos) vektorius. Taigi šifro gavėjas turi žinoti ne tik raktą, bet ir inicializacijos vektorių.

Jeigu naudojamasi vienu iš šių režimų, tai šifravimo įrenginys perskaito visą šifruojamą bloką, paskui jį šifruoja ir tada pasiunčia kanalu arba įrašo į sudaromą šifro failą. Padidinto saugumo reikalavimų atveju gali būti neleidžiama perskaityti ir nors trumpam saugoti šifruojamą bloką. Duomenys turi būti skaitomi bitas po bito, bitai šifruojami ir iš karto siunčiami į kanalą. Tokiam atvejui, pavyzdžiui, tinka

Šifro atgalinio ryšio režimas (CFB, Cipher Feedback)

Naudojantis šiuo režimu duomenų srautas gali būti šifruojamas „blokeliais“ po s, $1 \leq s \leq n$, bitų, čia n yra naudojamo blokinio šifro bloko ilgis. Duomenys šifruojami sudedant moduliu 2 (kitaip tariant, atliekant XOR operaciją) s ilgio duomenų blokelius su tokio pat ilgio blokeliais, kuriuos generuoja kriptosistema. Darbo pradžiai taip pat reikalingas inicializacijos vektorius.



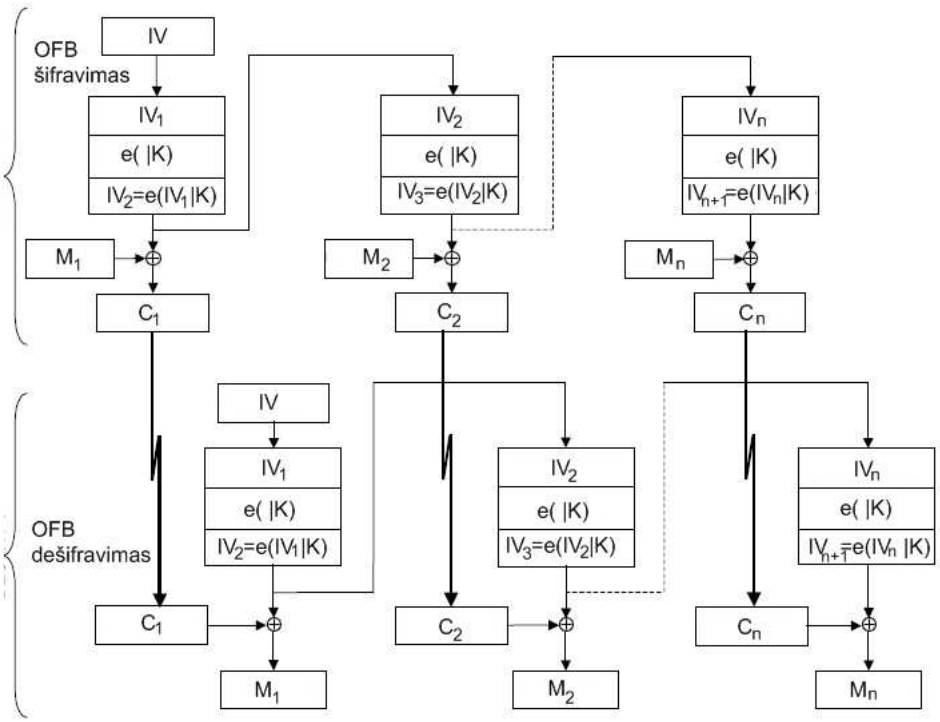
Duomenų šifravimas ir dešifravimas blokine kriptosistema CFB režimu.

Pirmajame žingsnyje blokinė kriptosistema šifruoja inicializacijos vektorių. Gautojo šifro s bitų panaudojama XOR operacijai, kiti atmetami. Tada pradinis inicializacijos vektorius pakeičiamas: pirmieji jo s bitų atmetami, o prie sutrumpinto vektoriaus prijungiama s gautojo šifro bitų. Galima įsivaizduoti, kad šifro bitai išstumia pirmuosius s inicializacijos vektoriaus bitų. Analogiškai inicializacijos bitai keičiami ir kituose žingsniuose. Dešifruojama lygiai taip pat kaip šifruojama. Tai net paprasčiau negu Feistelio šifre – net raktų tvarkos nereikia keisti.

Panašus į šį režimą yra

Srauto atgalinio ryšio režimas (OFB, Output Feedback)

Esminis skirtumas tik tas, kad srautas, kuris sudaromas XOR operacijai su šifruojamais duomenimis, generuoja pats save, taigi šifras nebenaudojamas.

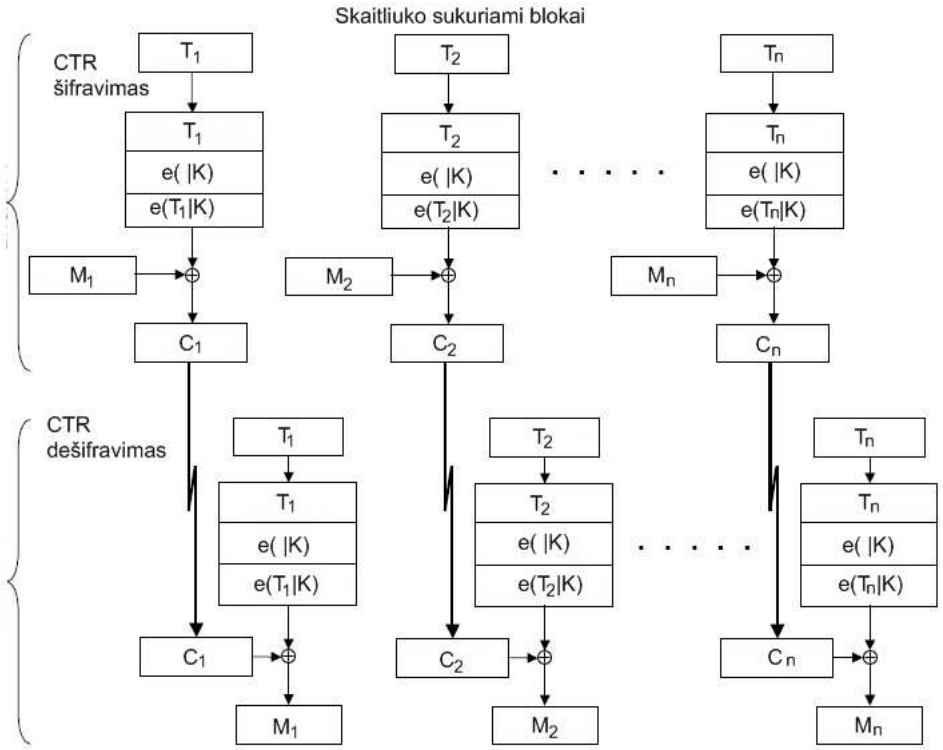


Duomenų šifravimas ir dešifravimas blokine kriptosistema OFB režimu.

Vienas šio režimo privalumų – bitų srautą, kuris naudojamas XOR operacijoje su duomenų srautu, galima generuoti iš anksto, t. y. neturint nei duomenų, nei šifro.

Skaitliuko režimas (CTR, Counter)

Šios paprastos schemos esmė tokia: blokinis šifras naudojamas blokams, kuriuos generuoja koks nors atskiras įrenginys (skaitiklis), šifruoti. Gautieji šifro žodžiai naudojami XOR operacijai su duomenų srautu.



Duomenų šifravimas ir dešifravimas blokine kriptosistema CTR režimu

Reikalaujama, kad skaitiklio generuojami blokai būtų skirtingi (žinoma, po daugelio žingsnių pasikartojimai neišvengiami). Paprasčiausias pavyzdys: dvejetainės skaičių $g, g+1, g+2, \ldots$ išraiškos. Šifravimas ir dešifravimas ir šiame režime yra vienodos operacijos. Tačiau reikia, kad šifro gavėjas turėtų taip pat veikiantį skaitiklį kaip siuntėjo.

Skaitytojas, žinoma, atkreipė dėmesį į tai, kad paskutinieji trys režimai priverčia blokinę kriptosistemą veikti kitaip. Iš tiesų, šie režimai iš blokinių kriptosistemų sukuria srautinius šifrus.

Režimų, žinoma, yra ir daugiau. Ir jūs galite sugalvoti savo režimą. Čia aptartieji režimai pasirinkti todėl, kad jie yra įtraukti į oficialų JAV Nacionalinio standartų ir technologijų instituto patvirtintą standartą[13].

[13]Morris Dworkin. Recommendation for BlockCipher Modes of Operation Methods and Techniques. NIST Special Publication 800-38A 2001 Edition.

17 Srautiniai šifrai

Blokinių kriptosistemų kūrėjai sprendžia klausimą: kaip transformuoti duomenų srautą, kad gautume šifrą? Srautinių kriptosistemų kūrėjai atsako į šį klausimą pačiu paprasčiausiu būdu.

Tegu $M = m_1 m_2 \ldots$ dvejetainės abėcėlės simbolių srautas, generuokime tokio pat ilgio rakto žodį $K = x_1 x_2 \ldots$ ir apibrėžkime M šifrą taip:

$$C = e(M|K) = c_1 c_2 \ldots, \quad c_i = m_i \oplus x_i,\ i = 1, 2, \ldots$$

Taigi teksto šifras – tai jo ir rakto srauto XOR operacijos rezultatas. Rakto simbolių srautas tiesiog paslepia teksto simbolius. Dešifravimas – ta pati XOR operacija, tik ją atlikti reikia su šifro ir rakto srautais:

$$M = d(C|K) = m_1 m_2 \ldots, \quad m_i = c_i \oplus x_i,\ i = 1, 2, \ldots$$

Tačiau šiuo atsakymu klausimai tik prasideda. Kaip sukurti tą rakto srautą K? Jeigu generuosime rakto srautą visiškai atsitiktinai ir naudosime jį tik vieną kartą, realizuosime Vernamo kriptosistemą. Tačiau tuomet bus reikalingas saugus kanalas raktui perduoti. Ar gavėjas negalėtų pats generuoti rakto srauto?

Taigi priartėjame prie pseudoatsitiktinių bitų srauto generavimo idėjos. Reikia sugalvoti algoritmą, kuris iš fiksuoto ilgio žodžio $k = k_1 k_2 \ldots k_m$ sukurtų reikiamo ilgio rakto srautą K:

$$k_1 k_2 \ldots k_m = k \longrightarrow K = x_1 x_2 \ldots$$

Šį srautą galėtume naudoti XOR operacijai su teksto simbolių srautu. Rakto srautas jau nebebus sudarytas iš atsitiktinai generuotų bitų, taigi jį vadinsime pseudoatsitiktinių bitų seka. Kuo ši seka panašesnė į tikrai atsitiktinę, tuo geriau.

Taigi svarbiausias srautinių šifrų kūrėjų rūpestis – kaip generuoti pseudoatsitiktinių bitų srautus?

17.1. Golombo pseudoatsitiktinės sekos

> *Tikrai atsitiktinę bitų seką gautume mėtydami simetrišką monetą ir užsirašydami rezultatus. Tai idealus, bet nepraktiškas būdas. Yra daug geresnių. Tačiau kaip nuspręsti, ar sudarytas bitų srautas yra panašus į atsitiktinį?*

Tarkime, sugalvojome kokį nors būdą generuoti nepriklausomų vienodai pasiskirsčiusių atsitiktinių dydžių $X_1, X_2, \ldots$

$$P(X_i = 0) = P(X_i = 1) = \frac{1}{2},$$

reikšmes. Naudodamiesi šiuo būdu, gavome n reikšmių

$$x_1x_2\dots x_n, \quad x_i \in \{0,1\}. \tag{81}$$

Gali pasitaikyti taip, kad, atlikę tiesiog tobulas dydžių reikšmių generavimo operacijas, mes gausime, pavyzdžiui, tokią bitų seką: $00011\dots11$. Ir niekas nepatikės, kad tai nepriklausomų atsitiktinių dydžių sekos reikšmės! Tačiau taip įvyks labai retai. Dažniausiai gausime sekas, turinčias tam tikras tipines savybes. Norėdami pagrįsti, kodėl tos savybės yra tipinės, turėtume formuluoti matematinius teiginius apie atsitiktinių dydžių tikimybes.

Pavyzdžiui, viena iš tokių tipinių savybių (už jos slypi svarbus tikimybių teorijos teiginys – didžiųjų skaičių dėsnis) yra tokia:

• *maždaug pusė (81) sekos elementų yra vienetai, kiti – nuliai.*

Vieną ar kelias tipines atsitiktinių dydžių reikšmių sekos savybes gali turėti labai „taisyklingos“ sekos. Pavyzdžiui, suformuluotą savybę turės seka, kuri sudaryta iš dviejų vienodo ilgio blokų: $00\dots0$ ir $11\dots1$. Todėl tiriant, ar bitų seka panaši į atsitiktinę, reikia pasitelkti daugiau kriterijų.

Pavadinkime r nulių grupę, „saugomą“ dviejų vienetų, (t. y. grupę $10\dots01$) 10_r1 bloku, o grupę $01\dots10$, kurioje nulių apsuptyje yra r vienetų, – 01_r0 vienetų bloku. Jeigu sekos pirmieji r nariai lygūs nuliui, o po jų seka vienetas, tai tokią seką vadinsime 0_r1 bloku; jeigu seka baigiasi r nuliais, tokią pabaigą vadinsime 10_r bloku. Analogiškai apibrėšime 1_r0 ir 01_r blokus. Blokus $10_r1, 0_r1, 10_r$ vadinsime tiesiog 0_r blokais. Analogiškai apibrėšime 1_r blokus.

Tarkime, N yra bendras (81) sekos blokų skaičius, t. y.

$$N = 0_1 \text{ skaičius } + 1_1 \text{ skaičius } + 0_2 \text{ skaičius } + \dots$$

Tada tipinėje sekoje

• *blokų skaičiai tenkintų lygybes:*

$$0_1 \textit{ skaičius } + 1_1 \textit{ skaičius } \approx \frac{N}{2}, \quad 0_2 \textit{ skaičius } + 1_2 \textit{ skaičius } \approx \frac{N}{2^2} \textit{ ir t.t.}$$

Jeigu sugretintume (81) su cikliškai pastumta seka

$$\begin{array}{ccccc} x_1 & x_2 & x_3 & \dots & x_n \\ x_t & x_{t+1} & x_{t+2} & \dots & x_{n+t-1} \end{array}$$

ir suskaičiuotume, kiek yra pozicijų su vienodais simboliais ir kiek su skirtingais, tai tipinėje sekoje

• *sutapimų ir nesutapimų būtų beveik po lygiai.*

Šios trys tipinės savybės panaudotos bandant apibūdinti bitų sekas, panašias į atsitiktinai generuotas.

97 apibrėžimas. *Tegu* $x_1x_2\dots x_n, \quad x_i \in \{0,1\}$, *yra bitų seka,* $\nu(1), \nu(0), \nu(0_r), \nu(1_r)$ *yra atitinkamai vienetų, nulių ir* r *ilgio blokų skaičiai*

šioje sekoje, o N – bendras blokų skaičius. Seką vadinsime pseudoatsitiktine Golombo seka, jeigu ji tenkina tokias sąlygas:

1) $|\nu(1) - \nu(0)| \le 1$;

2) $\nu(0_1) + \nu(1_1) \ge \dfrac{N}{2},\ \ldots,\ \nu(0_r) + \nu(1_r) \ge \dfrac{N}{2^r},\ \ldots$;

3) *dydis*

$$C(t) = \sum_{i=1}^{n}(2x_i - 1)(2x_{i+t} - 1), \quad t = 0, 1, \ldots, n-1,\ x_{k+n} = x_k, k \ge 1$$

įgyja tik dvi reikšmes: $C(0) = n,\ C(t) = m\ (t \ne 0)$.

Antrojoje sąlygoje nelygybes turi tenkinti, žinoma, tik sekoje egzistuojančių blokų kiekiai. Be to, dar pageidautina, kad būtų $\nu(0_r) \approx \nu(1_r)$. Trečioji sąlyga reikalauja, kad sutapimų ir nesutapimų skaičių skirtumas būtų nedidelis (lygus m). Golombo sąlygos yra gana griežtos. Tačiau jeigu seka tenkina griežtas sąlygas, tai ją nedrąsu vadinti atsitiktine. Golombo sąlygas tenkinančias sekas galima sukonstruoti. Pavyzdžiui, seka

011001000111101

yra pati tikriausia Golombo seka. Bendras blokų skaičius joje $N = 8$.

17.2. Statistiniai testai

Vertindami bitų seką, sprendžiame klausimą, ar ji tinka kriptografijos reikmėms, ar ne. Hipotezių tikrinimo terminologija ir metodai – geri įrankiai ieškant atsakymo.

Dauguma kriterijų (testų), kurie taikomi tiriamai bitų sekai, siekiant nustatyti, ar ji yra panaši į tipinę nepriklausomų atsitiktinių dydžių generuotą reikšmių seką, naudoja tam tikrus tikimybių teorijos rezultatus ir iš matematinės statistikos pasiskolintą hipotezių tikrinimo schemą. Trumpai aptarkime šio metodo esmę.

Tarkime, $X_1, X_2, \ldots, X_n$ yra „tikri“ nepriklausomi atsitiktiniai dydžiai, su vienodomis tikimybėmis įgyjantys reikšmes 0 ir 1. Sudaromas naujas atsitiktinis dydis (paprastai vadinamas tiesiog statistika)

$$T = T(X_1, X_2, \ldots, X_n),$$

kurio reikšmių pasiskirstymas paklūsta gerai žinomiems ir ištyrinėtiems dėsniams. Svarbiausi ir dažniausiai pasitaikantys iš jų – standartinis normalusis dėsnis $\mathcal{N}(0,1)$ ir $\chi^2(m)$ – chi-kvadrat dėsnis su m laisvės laipsnių (čia $m \ge 1$, taigi turime visą dėsnių šeimą). Visi šie dydžiai yra absoliučiai tolydūs, t. y. turi tikimybinius tankius – neneigiamas funkcijas, kuriomis reiškiamos su dydžiais susijusios tikimybės:

$$P(X<u) = \int_{-\infty}^{\infty} p_X(x)\mathrm{d}x, \quad p_X(x) = \frac{1}{\sqrt{2\pi}}\mathrm{e}^{-\frac{x^2}{2}},$$
$$P(Y<u) = \int_{0}^{\infty} p_Y(x)\mathrm{d}x, \quad p_Y(x) = \frac{1}{2^{n/2-1}\Gamma(n/2)}x^{n-1}\mathrm{e}^{-\frac{x^2}{2}},\ x>0,$$

čia $\Gamma(t)$ yra speciali funkcija (kai t natūrinis skaičius, tai $\Gamma(t)=(t-1)!$, todėl ją galima interpretuoti kaip faktorialo tęsinį). Dydis Y neįgyja neigiamų reikšmių, todėl $P(Y<u)=0$, kai $u<0$.

Sudarius geras savybes turinčią statistiką $T(X_1,X_2,\ldots,X_n)$, galima konstruoti testą, kurį taikant bitų sekai

$$\mathbf{x} = x_1x_2\ldots x_n, \tag{82}$$

priimama viena iš dviejų hipotezių:

H_0 : **x yra** tipinė dydžių $X_1,X_2,\ldots,X_n$ generuota seka,
H_1 : **x nėra** tipinė dydžių $X_1,X_2,\ldots,X_n$ generuota seka.

Jeigu priimama hipotezė H_0, sakome, kad seka praėjo testą, t. y. jos panašumas į atsitiktinai generuotą seką yra patvirtintas. Sprendimas priimamas apskaičiavus statistikos reikšmę

$$t = T(x_1,x_2,\ldots,x_n).$$

Testui taikyti dar reikia pasirinkti nedidelį teigiamą skaičių α, vadinamą reikšmingumo lygmeniu, (pavyzdžiui, $\alpha = 0,01; 0,005$) ir, naudojantis juo, sudaryti reikšmių aibę, į kurią patekus $t=T(x_1,x_2,\ldots,x_n)$, hipotezė H_0 yra atmetama. Ši sritis matematinėje statistikoje vadinama kritine sritimi.

Jeigu statistika T pasiskirsčiusi pagal standartinį normalinį dėsnį, tai hipotezė H_0 atmetama, jei

$$|T(x_1,x_2,\ldots,x_n)| \geq z_\alpha,$$

čia z_α randamas iš lygybės

$$P(|X|>z_\alpha)=\alpha, \quad X\sim\mathcal{N}(0,1).$$

Jeigu statistika T pasiskirsčiusi pagal chi-kvadrat dėsnį su m laisvės laipsnių, tai hipotezė H_0 atmetama, jei

$$T(x_1,x_2,\ldots,x_n) \geq z_{n,\alpha},$$

$z_{n,\alpha}$ randamas iš lygybės

$$P(X>z_{n,\alpha})=\alpha, \quad X\sim\chi^2(m).$$

Štai tokia statistinių testų ideologija. Reikia pabrėžti, kad apskritai statistikos $T(X_1, X_2, \ldots, X_n)$ pasiskirstymo dėsnis iš tiesų niekada nesutampa nei su standartiniu normaliuoju, nei su chi-kvadrat dėsniu. Tačiau kai n yra pakankamai didelis skaičius, skirstiniai yra panašūs, todėl, juos tiesiog sutapatinant, paklaidos neturi praktinės reikšmės.

O dabar liko išsiaiškinti, kaip konstruoti tas statistikas T, t. y. naujus gerus atsitiktinius dydžius iš nepriklausomų vienodai pasiskirsčiusių atsitiktinių dydžių sekos

$$X_1, X_2, \ldots, \quad X_i \in \{0,1\},\ P(X_i = 0) = P(X_i = 1) = \frac{1}{2}. \tag{83}$$

Statistinių testų iš tiesų yra sugalvota labai įvairių. Jei prireiks, teks paieškoti jiems skirtų publikacijų[14]. O mes aptarkime tik penkis.

Pavienių bitų testas

Tegu N_0 yra nulių skaičius atsitiktinių dydžių sekos (82) reikšmių aibėje, o N_1 – vienetų. Tada dydžio

$$T_1 = \frac{(N_1 - N_0)^2}{n}$$

pasiskirstymo dėsnis, kai $n \geq 10$ yra labai artimas dėsniui $\chi^2(1)$.

Taigi seka „praeis“ šį testą, jeigu joje vienetų ir nulių kiekiai bus panašūs. Galime sugretinti testą su pirmuoju Golombo reikalavimu. Kuris iš jų rodo daugiau tolerancijos?

Bitų porų testas

Tegu dydžių N_0, N_1 reikšmės yra tos pačios kaip pavienių bitų teste, o $N_{00}, N_{01}, N_{10}, N_{11}$ yra atitinkamai bitų blokų $00, 01, 10, 11$ kiekiai (82) reikšmių sekoje. Kadangi poros gali turėti vieną bendrą bitą, tai

$$N_{00} + N_{01} + N_{10} + N_{11} = n - 1.$$

Apibrėžkime statistiką

$$T_2 = \frac{4}{n-1}\left(N_{00}^2 + N_{01}^2 + N_{10}^2 + N_{11}^2\right) - \frac{2}{n}\left(N_0^2 + N_1^2\right) + 1.$$

Tikimybių teorija garantuoja, kad su $n \geq 21$ statistikos T_2 pasiskirstymo dėsnis artimas dėsniui $\chi^2(2)$.

Pokerio testas

Atliekant pavienių bitų testą, skaičiuojami nulių ir vienetų kiekiai reikšmių sekoje. Kodėl nepaskaičiavus, kiek kartų pasikartoja ilgesni žodžiai?

Fiksuokime m ($m < n$); visų skirtingų žodžių aibė yra $\mathbb{F}_2^m = \{a_1, a_2, \ldots, a_{2^m}\}$. Padalykime (82) reikšmių seką į $k = \left[\frac{n}{m}\right]$ m ilgio žodžių ir, peržiūrėję

[14] Pavyzdžiui, NIST rekomendacijos apie statistinių testų taikymą, žr. [66].

juos, nustatykime, kiek kartų pasitaiko žodžiai $a_1, a_2, \ldots, a_{2^m}$. Šių žodžių pasitaikymo kiekius pažymėkime $N_1, N_2, \ldots, N_{2^m}$. Dabar jau galime sudaryti statistiką

$$T_3 = \frac{2^m}{k} \sum_{i=1}^{2^m} N_i^2 - k.$$

Jeigu $k \geq 5 \cdot 2^m$, tai T_3 pasiskirstymo dėsnis artimas $\chi^2(2^m - 1)$.

Blokų testas

Antroji Golombo pseudoatsitiktinės sekos apibrėžimo sąlyga reikalauja, kad sekoje blokai pasirodytų gana dažnai. Pažymėkime F_r blokų 10_r1 skaičių (82) reikšmių sekoje, o G_r blokų 01_r0 skaičių. Pažymėkime

$$E_i = \frac{n - i + 3}{2^{i+2}}$$

ir k – didžiausią natūrinį skaičių, su kuriuo $E_k \geq 5$. Statistiką apibrėšime taip:

$$T_4 = \sum_{i=1}^{k} \left\{ \frac{(F_i - E_i)^2}{E_i} + \frac{(G_i - E_i)^2}{E_i} \right\}.$$

Statistikos T_4 pasiskirstymo dėsnis yra artimas $\chi^2(2k - 2)$.

Autokoreliacijos testas

Trečioji Golombo sąlyga kelia griežtą reikalavimą sutampančių bitų kiekiui, kada lyginame pradinę seką su paslinkta. Autokoreliacijos testas taip pat yra reikalavimas sutapimams, tačiau jį lengviau įvykdyti.

Tegu $1 \leq d \leq [n/2]$ ir

$$X(d) = \sum_{i=1}^{n-d+1} X_i \oplus X_{i+d-1}.$$

Dydis X_d yra lygus nesutampančių bitų skaičiui, kai lyginame atsitiktinių dydžių (83) reikšmių seką su ja pačia, paslinkta per $d-1$ poziciją į kairę (x_1 lyginamas su x_d). Jeigu $n - d \geq 10$, tai statistikos

$$T_5 = \frac{2X(d) - n + d}{\sqrt{n - d}}$$

pasiskirstymo dėsnis yra artimas standartiniam normaliajam dėsniui $\mathcal{N}(0, 1)$.

17.3. Tiesinių registrų sistemos

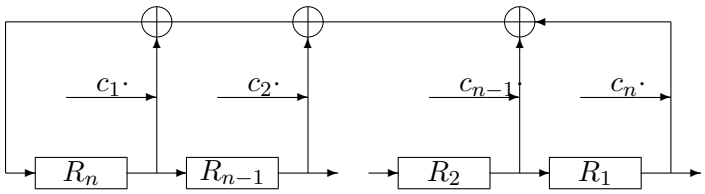
Vienas paprasčiausių instrumentų bitų srautams, panašiems į atsitiktinius, generuoti – tiesinių registrų įrenginys. Juo naudojamasi ir kodavimo teorijoje, ir kriptografijoje... todėl, kad juo paprasta naudotis.

Yra daug būdų generuoti bitų sekas, kurios tenkintų atsitiktinumo testus. Tikriausiai pats paprasčiausias ir geriausiai išnagrinėtas yra tiesinių registrų metodas.

Tarkime, kad sistemą sudaro n sujungtų tarpusavyje įrenginių, kuriuos vadinsime registrais ir žymėsime $R_1, \ldots, R_n$. Kiekvienas registras gali saugoti vieną informacijos bitą bei jį perduoti kitam registrui. Kiekvieno registro R_i turinį laiko momentu t žymėsime $x_i(t)$, $x_i(t) \in \{0,1\}$, $t = 0,1,2,\ldots$ Tegu pradinį sistemos būvį nusako vektorius

$$x(0) = \langle x_1(0), \ldots, x_n(0) \rangle.$$

Registrų sistemos turinio kitimą laikui bėgant nusakysime rekurenčiosiomis lygybėmis.



Tiesinių registrų sistema. Kiekviename žingsnyje atliekamas registrų bitų postūmis: pirmojo registro bitu papildomas generuojamas bitų srautas, antrojo registro bitas perrašomas į pirmąjį registrą, ..., n-ojo – į $n-1$-ąjį, o į n-ąjį registrą įrašoma registrų bitų tiesinė kombinacija.

Jei

$$x(t) = \langle x_1(t), \ldots, x_n(t) \rangle$$

yra sistemos padėtis laiko momentu t, tai sekančiame žingsnyje bus atlikti tokie veiksmai

$$x_i(t) = x_{i+1}(t), \quad 1 \le i \le n-1, \tag{84}$$

$$x_n(t+1) \equiv c_1 x_n(t) + \ldots + c_n x_1(t) \pmod 2, \tag{85}$$

čia $c_i \in \{0,1\}$, $1 \le i \le n$. Sakysime, kad $c_n \neq 0$, nes priešingu atveju registrų sistemą galėtume sutrumpinti. Tokia registrų sistema yra techniškai lengvai realizuojama: net ir daug registrų turintis įrenginys yra kompaktiškas ir dirba greitai.

Registrų sistemai veikiant, registrų turiniai nuolat keičiasi. Galime įsivaizduoti, kad kiekviename žingsnyje žodis $x(t)$ (dvejetainis vektorius) yra atvaizduo-

jamas į žodį $x(t+1)$. Atvaizdį galime užrašyti naudodami matricas taip:

$$x(t+1) = x(t)C, \quad C = \begin{pmatrix} 0 & 0 & \dots & 0 & c_n \\ 1 & 0 & \dots & 0 & c_{n-1} \\ 0 & 1 & \dots & 0 & c_{n-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & c_1 \end{pmatrix}.$$

Ši lygybė apibrėžia tiesinį atvaizdį $\mathbb{F}_2^n \to \mathbb{F}_2^n$. Kadangi matrica yra neišsigimusi, tai atvaizdis yra abipus vienareikšmis, t. y. bijekcija.

Aptarę tiesinių registrų sistemos konstrukciją, panagrinėkime jos darbo rezultatą – pseudoatsitiktinę seką

$$x_1(0), x_1(1), x_1(2), ... \tag{86}$$

98 apibrėžimas. *Elementų seką $\{y_i\}$ $(i = 0, 1, \dots)$ vadinsime periodine, jeigu egzistuoja toks natūralusis skaičius p, kad su visais $i \geq 0$ teisinga lygybė $y_{i+p} = y_i$. Mažiausią natūralųjį skaičių p, su kuriuo visos lygybės teisingos vadinsime sekos periodu.*

Tiesinių registrų seka gali generuoti tik periodines sekas.

120 teorema. *Tiesinė n registrų sistema generuoja periodines sekas, kurių periodas yra ne didesnis už $2^n - 1$.*

Įrodymas. Jei $x(0)$ (pradinė registrų sistemos padėtis) yra nulinis vektorius, tai teoremos tvirtinimas akivaizdus. Todėl toliau sakysime, kad $x(0) \neq \langle 0, \dots, 0 \rangle$.

Pasinaudoję lygybe $x(s+1) = x(s)C$ t kartų, gausime

$$x(t) = x(t-1)C = x(t-2)C^2 = \dots = x(0)C^t.$$

Pažymėję $m = 2^n - 1$, galime tvirtinti, kad vektorių

$$x(0),\ x(0)C,\ ...,\ x(0)C^m$$

sekoje būtinai bus pasikartojimų, nes iš viso yra tik m skirtingų nenulinių n-mačių vektorių. Taigi galima rasti tokius s ir t, kad $0 \leq s < s+t \leq m$ ir

$$x(0)C^s = x(0)C^{s+t}.$$

Kadangi neišsigimusi matrica C^s turi atvirkštinę matricą C^{-s}, tai

$$x(t) = x(0)C^t = x(0)C^{s+t}C^{-s} = x(0).$$

Dabar su bet kokiu $r \geq 0$ gausime

$$x(r+t) = x(0)C^{r+t} = x(0)C^tC^r = x(t)C^r = x(0)C^r = x(r).$$

Tai rodo, kad generuojama seka yra periodinė, o jos periodas yra ne didesnis už $t, t \leq m = 2^n - 1$.

Ar visada galima sukonstruoti tiesinę n registrų sistemą, kuri generuotų maksimalaus periodo $m = 2^n - 1$ seką? Tikėtis teigiamo atsakymo į šį klausimą leidžia kad ir toks pavyzdys.

Pavyzdys. Nagrinėkime keturių registrų sistemą, kurioje

$$c_1 = c_4 = 1,\ c_2 = c_3 = 0,\quad x(0) = \langle 1, 0, 1, 0\rangle.$$

Paeiliui skaičiuodami sistemos būsenas, gausime: $x(1) = \langle 1, 1, 0, 1\rangle$, $x(2) = \langle 0, 1, 1, 0\rangle$, $x(3) = \langle 0, 0, 1, 1\rangle$ ir t. t., kol pasieksime $x(15) = x(0) = \langle 1, 0, 1, 0\rangle$. Taigi tokia registrų sistema generuoja maksimalaus periodo seką. Kokios gi tiesinių registrų sistemos generuoja tokias sekas? Pirmiausia prisiminkime vieną sąvoką.

99 apibrėžimas. *n-ojo laipsnio daugianaris $f(x) \in \mathbb{F}_2[x]$ vadinamas primityviuoju, jeigu jis yra neskaidus ir nėra jokio daugianario $x^d + 1$ su $d < 2^n - 1$ daliklis.*

Rasti n-ojo laipsnio primityviuosius daugianarius nėra paprastas algebros uždavinys. Tačiau žinoma, kad visiems natūraliesiems n jie egzistuoja.

100 apibrėžimas. *Tiesinės registrų sistemos, nusakytos (85) lygybėmis charakteringuoju daugianariu vadinsime daugianarį*

$$P_n(x) = 1 + c_1 x + \ldots + c_n x^n,\ c_n \neq 0.$$

O dabar – atsakymas į suformuluotą klausimą.

121 teorema. *Tiesinė registrų sistema generuoja maksimalaus periodo seką tada ir tik tada, kai jos charakteringasis daugianaris yra primityvus.*

Jeigu jau ketiname naudoti tiesinių registrų sistemą pseudoatsitiktinių bitų srautui generuoti, tai verta pasirūpinti, kad šio srauto periodas būtų maksimalus. O kaipgi su atsitiktinumo testais? Štai viena tiesinių registrų sistemos generuoto srauto savybė:

122 teorema. *Jeigu tiesinių registrų sistema generuoja maksimalaus periodo m seką, tai bet kuri šios sekos m ilgio atkarpa yra Golombo pseudoatsitiktinių bitų seka.*

Pasirodo, yra be galo daug ir kiek norima ilgų sekų, kurios tenkina griežtuosius Golombo reikalavimus! Be to, jas ne taip jau sudėtinga ir sukonstruoti.

Įrodysime, kad maksimalaus periodo tiesinių registrų sistemos generuota seka tenkina pirmąsias dvi Golombo sąlygas. Tai išplauks iš tokio teiginio:

123 teorema. *Tegu tiesinės registrų sistemos generuotos sekos periodas yra maksimalus ir lygus $m = 2^n - 1$. Tuomet bet kurioje šios sekos m ilgio atkarpoje yra:*

1. $2^{n-1}-1$ *nulių ir* 2^{n-1} *vienetų;*

2. 2^{n-t-2} *blokų* 10_t1 *ir tiek pat blokų* 01_t0 *kiekvienam* t, $1 \le t \le n-2$.

Įrodymas. n registrų sistemos padėtį kiekviename žingsnyje galima išreikšti n-ženkliu dvejetainiu skaičiumi, priklausančiu intervalui $[1, 2^n-1]$. Kiekvienas eilinis generuotos sekos elementas bus lygus pirmojo registro turiniui, t. y. šio dvejetainio skaičiaus pirmajam skaitmeniui. Kadangi registrų sistema generuoja maksimalaus periodo seką, tai ji turi pereiti visas galimas padėtis nuo 1 iki 2^n-1. Tačiau tarp šių skaičių yra $2^{n-1}-1$ lyginių (paskutinis bitas 0) ir 2^{n-1} nelyginių (paskutinis bitas 1). Iš čia išplaukia pirmasis teiginys.

Kad sekoje pasirodytų t ilgio vienetų blokas, reikia, kad kuriame nors žingsnyje registrų sistemos padėtis būtų $011\ldots10x_1\ldots x_{n-t-2}$, čia $x_i \in \{0,1\}$. Iš viso tokių padėčių yra 2^{n-t-2}. Taigi tiek t ilgio vienetų blokų ir bus viename sekos periode. Analogiškai randamas ir t ilgio nulių blokų skaičius.

Taigi maksimalaus periodo m ilgio sekoje bendras blokų $10_r1, 01_r0$ skaičius yra

$$2(2^{n-3}+2^{n-2}+\ldots+1) = 2^{n-1}-2.$$

Pridėję dvejetą (sekos pradžios ir pabaigos blokai), gauname, kad bendras blokų skaičius lygus $N = 2^{n-1}$. Dabar jau nesudėtinga įsitikinti, kad antroji Golombo sąlyga yra patenkinta.

Pažiūrėkime į (86) seką kriptoanalitiko akimis. Tarkime, kad pranešimas yra dvejetainė seka $m_1m_2\ldots$ Ją užšifravę, gausime šifrą $y_1y_2\ldots$, čia

$$y_i \equiv m_i + x_i \pmod 2, \quad i = 1, 2, \ldots$$

Žinodami m_i ir y_i, nesunkiai randame x_i:

$$x_i \equiv m_i + y_i \pmod 2.$$

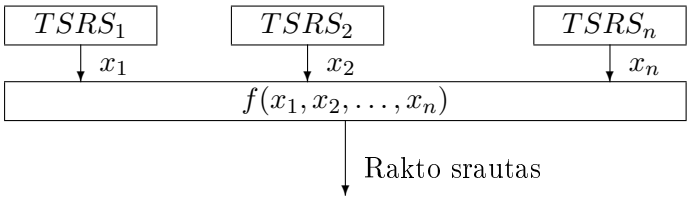
Suradę bet kuriuos $2n$ iš eilės einančius simbolius x_i, galėsime sudaryti n tiesinių lygčių su nežinomaisiais $c_1, \ldots, c_n$. Jas išsprendę rasime registrų sistemos koeficientus ir galėsime atkurti visą rakto srautą. Taigi kriptosistemą nesudėtinga įveikti turint pakankamai ilgą teksto ir jo šifro fragmentą.

Išvada. *Jeigu vienkartinio rakto kriptosistemoje raktas generuojamas tiesinių registrų sistema, tai pavienės teksto-šifro poros ataka tokia kriptosistema yra įveikiama.*

17.4. Iš paprastų – sudėtingesni

Keletas idėjų, kaip, jungiant paprastus pseudoatsitiktinių skaičių generatorius, galima sukurti sudėtingesnes rakto srauto generavimo sistemas.

Tiesinių registrų sistemos nėra vienintelis būdas generuoti pseudoatsitiktines bitų sekas. Tačiau tai lengviausiai realizuojamas būdas ir todėl dažnai naudojamas. Pagrindinis tokių sistemų trūkumas – generuoto srauto tiesiškumas, t. y. tiesinė generuoto simbolio priklausomybė nuo ankstesniųjų. Vienas iš būdų panaikinti tiesiškumą – kelių tiesinių registrų sistemų generatorių jungimas į sudėtingesnę sistemą.



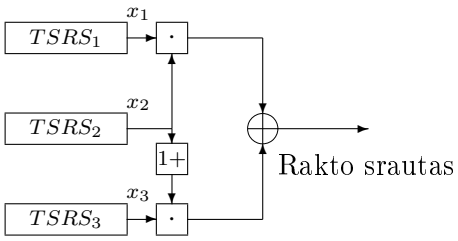
Kelių tiesinių registrų sistemų lygiagretus jungimas

Rakto srauto bitus generuoja Būlio funkcija $f(x_1, x_2, \ldots, x_n)$, ją visada galima užrašyti kaip n kintamųjų daugianarį. Pavyzdžiui, tris tiesinių registrų sistemas jungiančiame Geffe generatoriuje

$$f(x_1, x_2, x_3) = x_1 x_2 \oplus (1 \oplus x_2) x_3.$$

Jeigu Geffe schemoje visos sujungtos tiesinių registrų sistemos generuoja maksimalaus periodo sekas, o registrų kiekiai sistemose L_1, L_2, L_3 yra tarpusavyje pirminiai skaičiai, tai Geffe generatorius sukuria srautą, kurio periodas yra

$$(2^{L_1} - 1)(2^{L_2} - 1)(2^{L_3} - 1). \tag{87}$$



Geffe generatorius

Tačiau šioje schemoje yra vienas plika akimi nematomas trūkumas, kuris gali palengvinti tokios schemos kriptoanalizę.

Tarkime, turime pakankamai ilgą Geffe generatoriaus sukurtą rakto srauto fragmentą ir galime naudotis pačiu generatoriumi. Tikslas – sužinoti, kokie buvo registrų turiniai, kai buvo sudaromas rakto srautas, kurio fragmentą pavyko gauti. Sužinoję iš viso $L_1 + L_2 + L_3$ bitų, galėtume patys generuoti kokio tik reikia ilgio rakto srautą ir dešifruoti su juo gautą šifrą. Galima būtų tiesiog bandyti visus variantus: užpildžius registrus, generuoti raktų srautą ir lyginti jį su jau turimu. Iš viso blogiausiu atveju tektų atlikti (87)

tikrinimų. Tačiau kiekvieną iš trijų sistemų, pasirodo, galima „lukštenti“ skyrium. Pažymėkime $x_1(t), x_2(t), x_3(t)$ registrų sistemų generuotus bitus t žingsnyje, o $k(t)$ – iš jų gautą rakto srauto bitą. Tarkime, kad visos trys sistemos generuoja pakankamai „atsitiktinumo savybių“ turinčius srautus, pavyzdžiui, tarkime, kad $x_i(t)$ ir $x_j(t)$ kai, $i \neq j$, yra nepriklausomi ir

$$P(x_i(t) = 0) = P(x_i(t) = 1) = \frac{1}{2}.$$

Nesunku įsitikinti, kad

$$P(k(t) = x_1(t)) = P(x_2(t) = 1) + P(x_2(t) = 0)P(x_3(t) = x_1(t)) = \frac{1}{2} + \frac{1}{2}\cdot\frac{1}{2} = \frac{3}{4}.$$

Taigi pirmosios sistemos generuotas bitas labai dažnai sutampa su rakto bitu. Tai teisinga ir trečiosios sistemos generuotam srautui:

$$P(k(t) = x_3(t)) = \frac{3}{4}.$$

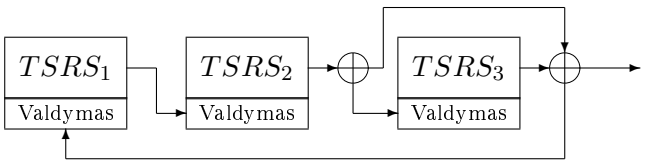
Nustačius šią savybę, schemos ataką galima vykdyti taip: užpildžius pirmosios sistemos registrus ir generavus srautą, reikia jį lyginti su rakto srautu ir tikrinti, ar sutapimų yra maždaug trys ketvirtadaliai. Jeigu taip – pirmosios registrų sistemos raktas įspėtas. Blogiausiu atveju prireiks $2^{L_1} - 1$ tikrinimų. Analogiškai galima nustatyti ir trečiosios sistemos pradines registruose saugotas reikšmes, o paskui ir antrosios sistemos. Taigi blogiausiu atveju prireiks

$$(2^{L_1} - 1) + (2^{L_2} - 1) + (2^{L_3} - 1)$$

tikrinimų, tai yra daug mažiau už (87) dydį.

Šis paprastas pavyzdys demonstruoja koreliacinių atakų, kurios taikomos srautinėms kriptosistemoms, esmę: jeigu yra priklausomybė tarp rakto srauto ir vieno schemos elemento generuoto srauto, tai tokią priklausomybę galima panaudoti daliai rakto bitų nustatyti.

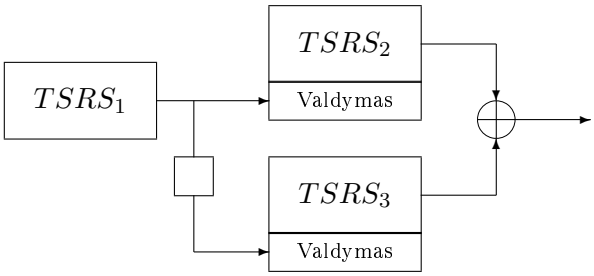
Panagrinėkime kelias nuoseklaus tiesinių registrų sistemų jungimo idėjas. Golmano generatorių grandinėje kiekviena sistema „klauso“ pirmesnės sistemos.



Golmano tiesinių registrų sistemų grandinė. Jeigu į tiesinės registrų sistemos „valdymo“ skyrių perduoto bito reikšmė lygi vienetui, sistema generuoja eilinį bitą ir atlieka registrų turinių postūmį; jeigu valdymo bitas lygus nuliui, registrų turiniai nepasikeičia. Rakto srautas – dviejų paskutinių registrų bitų suma.

Jeigu, pavyzdžiui, pirmoji sistema t žingsnyje generuoja $x_1(t) = 0$, tai antroji sistema, generavusi savo simbolį, nepakeičia registrų būsenų, jeigu $x_1(t) = 1$ – pakeičia. Analogišku būdu antroji sistema valdo trečiąją ir t.t.

Vienos registrų sistemos generuotas srautas gali būti naudojamas ne vienai, bet kelioms kitoms registrų sistemoms valdyti.



Pirmosios registrų sistemos generuotas srautas valdo antrosios ir trečiosios sistemų būsenų kaitą. Jeigu $x_1(t) = 1$, tai iš naujo užpildomi antrosios sistemos registrai, o trečiosios – lieka kaip buvę. Jeigu $x_1(t) = 0$, keičiasi trečiosios sistemos registrų turiniai, o antroji sistema „nepajuda“.

Dar viena dviejų sistemų derinimo idėja: pirmosios sistemos generuotą srautą galima naudoti antrosios srautui „praretinti“. Pavyzdžiui, jeigu pirmoji sistema generavo srautą

$$x_1 = 0100111001100011001\ldots,$$

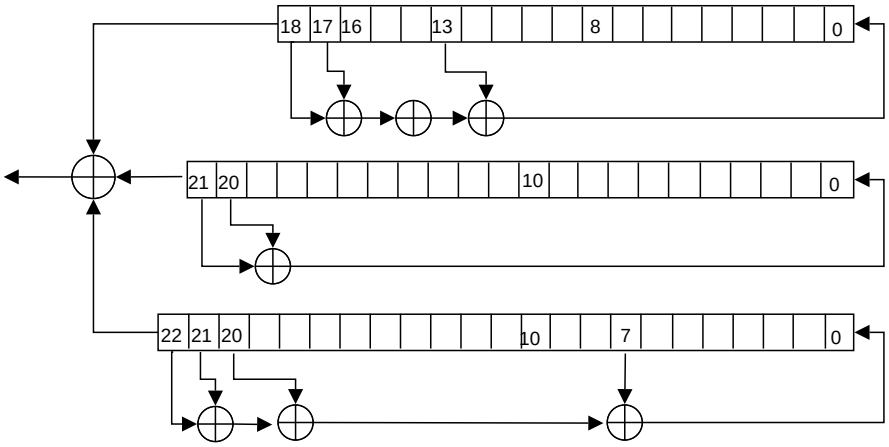
tai rakto srautas bus sudaromas iš antrosios sistemos generuoto srauto bitų:

$$k = x_2(2)x_2(5)x_2(6)x_2(7)x_2(10)x_2(11)x_2(15)x_2(16)x_2(19)\ldots$$

17.5. A5 ir Bluetooth E0

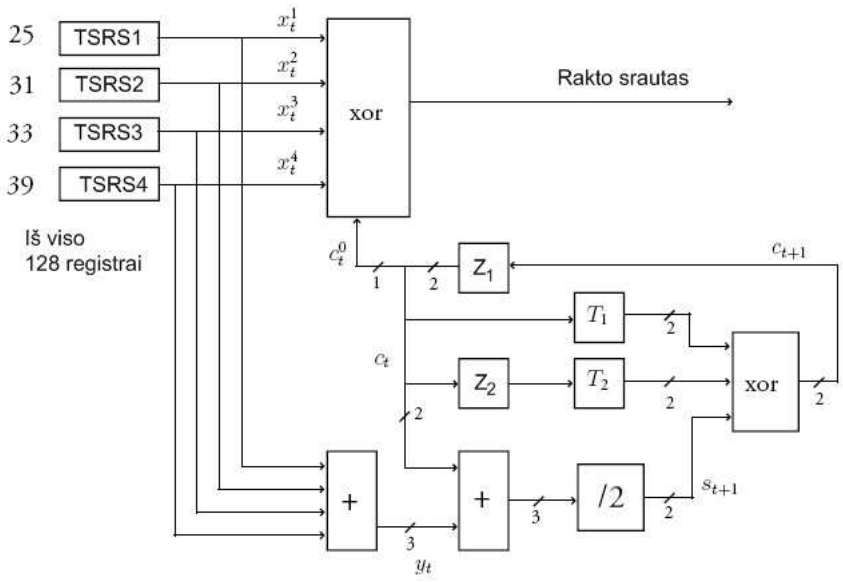
Operacijos su bitais srautinių šifrų sistemose vykdomos greitai ir nereikalauja didelių išteklių. Todėl jie naudojami tose ryšio priemonėse, kur reikia šifruoti daug ir greitai. Pavyzdžiui, bevieliam kompiuterių tinklui arba pokalbiams mobiliaisiais telefonais apsaugoti.

Srautinių šifrų yra daug ir įvairių. Panagrinėsime dvi šiuolaikiniuose ryšiuose naudojamas kriptosistemas. A5 sistema naudojama pokalbiams telefonu šifruoti, o Bluetooth E0 – bevielio kompiuterių tinklo protokoluose.



A5/1 kriptosistema

Srautinio šifro A5/1 rakto srautas sukuriamas atlikus sudėties moduliu 2 operaciją su bitais, kuriuos generuoja trys tiesinių registrų sistemos. Tačiau kiekviena iš šių sistemų nebūtinai kiekviename žingsnyje pakeičia savo būseną, t. y. iš naujo suformuoja registrų turinius. Sudarius rakto srauto bitą fiksuojami pirmosios sistemos aštuntojo registro, antrosios ir trečiosios sistemų – dešimtųjų registrų bitai. Pavyzdžiui, tegu šie bitai yra $0, 1, 0$. Nulių yra daugiau, todėl pirmosios ir trečiosios sistemų registrų turiniai yra pakeičiami, o antrosios – lieka kaip buvę. Jeigu visuose registruose būtų nuliai arba vienetai – visų trijų sistemų registrai būtų perkrauti.



Bluetooth E0 srautinio šifro schema

Bluetooth E0 rakto srauto bitai k_t gaunami atlikus sudėties moduliu operaciją su keturių tiesinių registrų generatorių pateiktais bitais ir vienu sistemos vidinės būsenos bitu:

$$k_t = x_t^1 \oplus x_t^2 \oplus x_t^3 \oplus x_t^4 \oplus c_t^0.$$

Kartu sudėjus iš viso yra 128 registrai, kiek jų kiekviename iš generatorių matyti iš jų charakteringųjų daugianarių, kurie visi yra primityvūs:

$$\begin{aligned} p_1(x) &= x^{25} + x^{20} + x^{12} + x^8 + 1, \\ p_2(x) &= x^{31} + x^{24} + x^{16} + x^{12} + 1, \\ p_3(x) &= x^{33} + x^{28} + x^{24} + x^4 + 1, \\ p_4(x) &= x^{39} + x^{36} + x^{28} + x^4 + 1. \end{aligned}$$

Taigi kiekviena iš keturių tiesinių registrų sistemų generuoja maksimalaus periodo pseudoatsitiktines sekas. Tiesinių registrų sistemos „išorėje“, dėžutėse Z_1 ir Z_2, saugomi keturi sistemos vidinės atminties bitai. Dėžutėje Z_1 saugoma bitų pora $c_t = c_t^0 c_t^1$, o dėžutėje $\mathbb{Z}_2$ – bitų pora $c_{t-1} = c_{t-1}^0 c_{t-1}^1$. Bitas c_t^0 naudojamas t žingsnio rakto srauto bitui sudaryti, o kiti – vidinei būsenai pakeisti. Vidinė būsena t žingsnyje pasikeičia taip: dėžutės Z_2 bitų pora $c_{t-1} = c_{t-1}^0 c_{t-1}^1$ pakeičiama į $c_t = c_t^0 c_t^1$, o į dėžutę Z_1 įkeliami du bitai $c_{t+1} = c_{t+1}^0 c_{t+1}^1$, gauti atlikus XOR operaciją įrenginyje X_2. Ši operacija atliekama su bitų porų trejetu. Pirmoji iš jų – per T_1 perduodama pora $c_t = c_t^0 c_t^1$. Antrąją iš dėžutės Z_2 perduotos poros $c_{t-1} = c_{t-1}^0 c_{t-1}^1$ sukuria įrenginys T_2. Jame bitų pora transformuojama taip:

$$T_2 : z_0 z_1 \mapsto z_2 z_0, \quad z_2 = z_0 \oplus z_1.$$

Daugiausia vargo dėl trečiosios poros, kurią pažymėsime $s_{t+1} = s_{t+1}^0 s_{t+1}^1$. Pirmiausia sudedant keturių registrų bitus kaip natūraliuosius skaičius, sudaromas trimis bitais užrašomas skaičius

$$y_t = x_t^1 + x_t^2 + x_t^3 + x_t^4.$$

Tada, pasinaudojus dėžutės Z_1 bitais, sudaromas skaičius $c_t = c_t^0 + 2c_t^1$ ir suskaičiuojama suma $y_t + c_t$. Paskutinė operacija jau sukuria trečiąją bitų porą:

$$s_{t+1} = s_{t+1}^0 s_{t+1}^1 = \left[\frac{y_t + c_t}{2} \right].$$

18 Sudėtingumo teorijos pradmenys

Dešifruodamas šifrą, teisėtas gavėjas naudojasi raktu ir sprendžia nesunkų uždavinį, o neteisėtas, bandydamas iššifruoti šifrą be rakto, – sunkų. Jeigu jums tokio paaiškinimo užtenka, galite šio skyriaus ir neskaityti. Tačiau jeigu norėtumėte sužinoti, pagal kokius požymius nesunkūs uždaviniai skiriami nuo sunkių – tada nepraleiskite ir šių puslapių. Tuo labiau kad vienoje nuošalioje šio skyriaus vietelėje yra paaiškinta, kaip galima užsidirbti milijoną dolerių.

18.1. Uždaviniai ir jų sprendimai

Uždavinį sudaro sąlyga ir klausimas. Sąlygą žinome, o atsakymo į klausimą – ne. Uždavinio sprendimas – tai planas, kuris visada nuo sąlygos duomenų nuveda iki atsakymo. Uždavinio sudėtingumas matuojamas šio plano ilgiu. Štai toks trumpas šio skyrelio turinys. Prisiminkite tai, jeigu paklysite matematiniuose tankumynuose!

Tiek teisėtas šifro gavėjas, tiek kriptoanalitikas dešifruodami sprendžia uždavinį: kokį pranešimą atitinka šifras? Gavėjui, turinčiam kriptosistemos raktą, ši problema nesunki, o kriptoanalitikui gali būti net teoriškai neišsprendžiama (kaip vienkartinio rakto kriptosistemos atveju). Šiame skyriuje nagrinėsime tik turinčius sprendimus uždavinius. Tačiau uždavinių būna įvairių: vieni turi vieną, kiti – kelis sprendimus. Kita vertus, ką reiškia išspręsti uždavinį? Tad iš pradžių pabandykime formalizuoti uždavinio ir jo sprendimo sąvokas.

Uždavinį paprastai sudaro sąlygos duomenys (dažnai vadinsime juos įeities duomenimis, santrumpa ID) ir užduotis arba klausimas (santrumpa U), į kurį reikia pateikti atsakymą. Tiek įeities duomenis, tiek atsakymą reikia užrašyti kokios nors abėcėlės simboliais. Dvejetainės abėcėlės $\mathcal{B} = \{0, 1\}$ šiam tikslui visiškai pakaks. Visų įmanomų šios abėcėlės žodžių aibę žymėsime, kaip visada, $\mathcal{B}^*$, $\mathcal{B}^* \times \mathcal{B}^*$ reiškia visų žodžių porų aibę.

101 apibrėžimas. *Uždaviniu vadinsime aibę $\pi \subset \mathcal{B}^* \times \mathcal{B}^*$, turinčią savybę: kiekvienam $\mathbf{x} \in \mathcal{B}^*$ atsiras $\mathbf{y} \in \mathcal{B}^*$, kad $\langle \mathbf{x}, \mathbf{y} \rangle \in \pi$. Jei $\langle \mathbf{x}, \mathbf{y} \rangle \in \pi$, tai $\mathbf{x}$ vadinsime įeities duomenimis, o $\mathbf{y}$ – atsakymu.*

Taigi įeities duomenys pateikiami dvinarės abėcėlės žodžiu. Kiekvienu konkrečiu atveju tokią duomenų pateiktį nebūna sudėtinga sudaryti. Akivaizdu, pavyzdžiui, kaip nulių-vienetų žodžiais užrašyti natūraliuosius skaičius, šiek tiek reikia pagalvoti, kaip pateikti orientuotus grafus. Gali atrodyti keista, kad bet kuris žodis įeina į uždavinį kaip įeities duomenys. Juk kad ir kokį, pavyzdžiui, grafų užrašymo būdą naudotume, ne visi dvinarės abėcėlės žodžiai reikš egzistuojančią struktūrą. Tačiau tokiais atvejais galime numatyti, kad atsakymas A reiškia, kad įeities duomenys neturi prasmės.

102 apibrėžimas. *Uždavinį $\pi \subset \mathcal{B}^* \times \mathcal{B}^*$ vadinsime funkcija, jei kiekvienam $\mathbf{x} \in \mathcal{B}^*$ atsiras vienintelis $\mathbf{y} \in \mathcal{B}^*$, kad $\langle \mathbf{x}, \mathbf{y} \rangle \in \pi$. Jei galimos tik dvi atsakymo reikšmės, funkciją vadinsime išsprendžiamumo uždaviniu.*

Išsprendžiamumo uždavinio atsakymo reikšmes interpretuosime kaip teigiamą bei neigiamą atsakymus T („taip“, arba galima žymėti tiesiog 1) ir N (ne, arba 0).

Visus uždavinius formaliai galima pertvarkyti į išsprendžiamumo uždavinius, t. y. kiekvienam uždaviniui galima rasti „orakulą“, kuris į pateikiamus klausimus atsakinėja tik „taip“ ir „ne“, bet iš tokių jo atsakymų jūs galite sukurti savo uždavinio sprendimą.

Apsiribosime natūraliojo skaičiaus skaidymo pirminiais daugikliais (faktorizacijos) uždavinio pavyzdžiu:

ID: *natūralusis skaičius n*;

U: *rasti netrivialų n daliklį arba nustatyti, kad tokio daliklio nėra.*

Šitaip suformuluotas uždavinys nėra, žinoma, išsprendžiamumo uždavinys. Tarkime, turime gerą algoritmą tokiam išsprendžiamumo uždaviniui spręsti:

ID: *natūralieji skaičiai n, k*;

U: *ar egzistuoja n daliklis m, tenkinantis nelygybę $m \geq k$?*

Pastarasis uždavinys yra išsprendžiamumo uždavinys, juk prašoma atsakyti tik „taip“ arba „ne“. Pasinaudosime šio uždavinio sprendimo algoritmu skaičiaus skaidiniui rasti.

Apsiribokime skaitiniu pavyzdžiu. Tarkime, reikia rasti netrivialų $n = 63$ daliklį m. Kadangi $n < 2^6$, tai daliklį, jeigu jis egzistuoja, galima užrašyti taip:

$$m = \epsilon_5 \cdot 2^5 + \epsilon_4 \cdot 2^4 + \epsilon_3 \cdot 2^3 + \epsilon_2 \cdot 2^2 + \epsilon_1 \cdot 2^1 + \epsilon_0,$$

čia $\epsilon_i \in \{0, 1\}$. Žymėkime $A(n, k)$ atsakymą, kurį duoda antrojo uždavinio algoritmas įeities duomenims n, k.

Kadangi $A(n, 2^5) = N$, tai galime teigti, kad $\epsilon_5 = 0$; $A(n, 2^4) = T$, tai $\epsilon_4 = 1$; $A(n, 2^4 + 2^3) = N$, todėl $\epsilon_3 = 0$; $A(n, 2^4 + 2^2) = T$, taigi $\epsilon_2 = 1$; $A(n, 2^4 + 2^2 + 2) = N$, $\epsilon_1 = 0$; pagaliau $A(n, 2^4 + 2^2 + 1) = T$, $\epsilon_0 = 1$. Ieškomas netrivialus daliklis toks:

$$m = 2^4 + 2^2 + 1 = 31.$$

103 apibrėžimas. *Išsprendžiamumo uždavinio $\pi \subset \mathcal{B}^* \times \mathcal{B}^*$ kalba vadinsime poaibį*

$$\mathcal{L}_\pi = \{\mathbf{x} : \mathbf{x} \in \mathcal{B}^*, \langle \mathbf{x}, T \rangle \in \pi\}.$$

Taigi išsprendžiamumo uždavinį galime interpretuoti kaip klausimą, ar žodis $\mathbf{x} \in \mathcal{B}^*$ įeina į kalbą $\mathcal{L}_\pi$.

Uždavinio sprendimą intuityviai suvokiame kaip tam tikrų nurodymų rinkinį, kuris valdo procesą, pasibaigiantį atsakymu (kartais nesėkme). Sudėtingumo teorijoje naudojama ne viena formali uždavinio sprendimo sam-

prata. Apsiribosime determinuotomis Turingo mašinomis. Mums užteks neformalaus jų apibrėžimo[15].

Galime įsivaizduoti, kad Turingo mašiną sudaro begalinė į kvadratėlius padalyta juosta ir slankiojanti skaitymo-rašymo galvutė. Kiekviename kvadrate gali būti įrašytas vienas abėcėlės $\mathcal{B}$ simbolis. Galvutė gali nuskaityti simbolį iš kvadrato, ties kuriuo ji stovi. Mašina gali būti vienoje iš padėčių, nusakomų aibe $Q = \{q_0, \ldots, q_m\}$. Mašina gali atlikti kelis paprastus veiksmus, kuriuos vadinsime elementariomis operacijomis:

- pakeisti nuskaitytą simbolį kitu $\mathcal{B}^*$ simboliu;
- paslinkti galvutę per vieną kvadratą į kairę ar į dešinę;
- pakeisti mašinos padėtį iš q_i į q_j.

Kuris iš šių veiksmų bus atliktas i-ajame žingsnyje, vienareikšmiškai lemia nuskaitytas simbolis ir tuometinė padėtis q_i.

Darbo pradžioje į juostos kvadratus nuo 1 iki n užrašomas pradinis žodis $\mathbf{x} \in \mathcal{B}^*$, $|\mathbf{x}| = n$, čia $|\mathbf{x}|$ – žodžio $\mathbf{x}$ ilgis. Mašina pastatoma į pradinę padėtį q_0, o jos galvutė ties pirmuoju kvadratu. Toliau mašina atlieka elementarias operacijas ir baigia darbą, patekusi į padėtį q_m. Juostos turinys tuo metu ir bus mašinos darbo rezultatas. Jį žymėsime $f(\mathbf{x})$ ir sakysime, kad Turingo mašina skaičiuoja funkciją $f : \mathcal{B}^* \to \mathcal{B}^*$. Mašinai M reikšmei skaičiuoti reikalingą elementariųjų operacijų skaičių žymėsime $t_{\mathrm{M}}(\mathbf{x})$.

104 apibrėžimas. *Turingo mašinos* M *darbo laiku vadinsime funkciją* $T_{\mathrm{M}} : \mathbf{N} \to \mathbf{N}$, *kurios reikšmė kiekvienam natūraliajam* n *yra lygi*

$$T_{\mathrm{M}}(n) = \max\{t : \quad \textit{egzistuoja } \mathbf{x} \in \mathcal{B}^*,\ |\mathbf{x}| = n,\ t_{\mathrm{M}}(\mathbf{x}) = t\}.$$

Taigi uždavinio (funkcijos, atskiru atveju – išsprendžiamumo uždavinio) sprendimą matematiškai suvoksime kaip atitinkamą Turingo mašiną.

Tą pačią funkciją $f(\mathbf{x})$ gali skaičiuoti įvairios Turingo mašinos priklausomai nuo to, koks algoritmas yra naudojamas. Konstruoti Turingo mašiną ir skaičiuoti jos darbo laiką – keblus darbas. Sudėtingesniems algoritmams patogiau konstruoti Turingo mašiną, kuri yra paprastesnių Turingo mašinų sistema (panašiai kaip programa, sudaryta iš paprogramių). Lengviau skaičiuoti ne Turingo mašinos elementarias operacijas, o įprastinių matematinių veiksmų kiekį. Pavyzdžiui, dviejų n dvejetainiais skaitmenimis užrašomų skaičių sudėčiai atlikti reikia n operacijų su simboliais. Kiekvienam tokiam veiksmui atlikti Turingo mašinai prireikia tam tikru dydžiu c_1 aprėžto elementariųjų operacijų kiekio. Galime tarti, kad dar reikia papildomai atlikti

[15] Matematinio griežtumo norintis skaitytojas turėtų pavartyti R. Lassaigne'o ir M. de Rougemont'o knygą „Logika ir algoritmų sudėtingumas“, kurią į lietuvių kalbą išvertė S. Norgėla, žr. [47].

ne daugiau kaip c_2 „techninių“ operacijų, pavyzdžiui, skaičių pabaigos ženklui nuskaityti. Taigi skaičių sudėčiai atlikti prireiks

$$T_{\mathrm{M}}(n) \le c_1 n + c_2$$

operacijų.

Panagrinėkime daugiau pavyzdžių. Vertindami operacijų skaičių, naudosime žymenį $\mathrm{O}(g)$, kuris reiškia tam tikrą dydį, ne didesnį kaip $c \cdot g$, čia $c > 0$ yra tam tikra konstanta, kurios tiksli reikšmė dažniausiai nėra svarbi.

1 pavyzdys. *Sveikųjų skaičių daugyba ir dalyba.* Tarkime, kad skaičiai $\mathbf{x} = x_1 \dots x_n$, $\mathbf{y} = y_1 \dots y_n$ užrašyti dvejetainėje sistemoje. Paeiliui daugindami x_i iš $y_1, y_2, \dots, y_n$, atliksime n^2 bitų daugybos operacijų. Gautas sandaugas pastumdami ir sudėdami atliksime dar maždaug n^2 operacijų. Taigi reikalingas bitų operacijų skaičius bus $T(n) = \mathrm{O}(n^2)$.

Jeigu reikia skaičių $\mathbf{x} = x_1 \dots x_n$ padalyti su liekana iš skaičiaus $\mathbf{y} = y_1 \dots y_m$ $(m \le n)$, tai, pagalvoję, kaip atliekamas „dalybos kampeliu“ algoritmas, įsitikinsime, kad reikalingas operacijų su bitais skaičius neviršys

$$T(n) = \mathrm{O}\left((n - m + 1) \cdot m\right) = \mathrm{O}(n^2).$$

Tačiau tiek dalybai, tiek daugybai galima sukonstruoti ir greitesnius algoritmus.

Panagrinėkime daugybos veiksmą. Kaip ir anksčiau $T(n)$ žymėsime kiekį elementarių operacijų, kurių reikia dviem skaičiams, užrašomiems n ilgio bitų žodžiu, sudauginti. Tarkime, kad reikia sudauginti skaičius $\mathbf{x} = x_1 \dots x_n$, $\mathbf{y} = y_1 \dots y_n$, užrašytus dvejetainėje sistemoje, ir n yra lyginis. Jeigu n nėra lyginis, visada galime pridėti gale nulį. Skaičius $\mathbf{x}$ ir $\mathbf{y}$ perskelsime pusiau:

$$\mathbf{x} = a2^{n/2} + b, \quad \mathbf{y} = c2^{n/2} + d.$$

Dabar sandaugą $\mathbf{x} \cdot \mathbf{y}$ galėsime skaičiuoti taip:

$$\mathbf{x} \cdot \mathbf{y} = (ac)2^n + (ac + bd - (a - b)(c - d))2^{n/2} + bd.$$

Taigi reikia apskaičiuoti tris sandaugas $(ac, bd, (a-b)(c-d))$, kuriose daugikliai yra ne ilgesni kaip $n/2$, bitų ir atlikti kelias sudėties operacijas. Kadangi daugyba iš 2^n yra labai paprasta, tai gausime nelygybę

$$T(n) \le 3T\Big(\frac{n}{2}\Big) + \mathrm{O}(n).$$

Tačiau perpus trumpesnių skaičių daugybai taip pat galime taikyti tą patį metodą, taigi

$$\begin{aligned} T(n) &\le 3\Big(3T\Big(\frac{n}{4}\Big) + \mathrm{O}\Big(\frac{n}{2}\Big)\Big) + +\mathrm{O}(n), \\ T(n) &\le 4 \cdot 3^k + \mathrm{O}\Big(n + 3 \cdot \frac{n}{2} + \ldots + 3^{k-1}\frac{n}{2^{k-1}}\Big), \end{aligned}$$

čia k yra natūrinis skaičius, $n/2^k \le 2$; galime imti $k = [\log_2 n]$. Kiek paskaičiavę gautume

$$3^k \le 3^{\log_2 n} = n^{\log_2 3}, \quad n + 3 \cdot \frac{n}{2} + \ldots + 3^{k-1}\frac{n}{2^{k-1}} \le \frac{2}{3} \cdot \left(\frac{3}{2}\right)^k < n^{\log_2 3}.$$

Taigi

$$T(n) = \mathrm{O}\big(n^{\log_2 3}\big).$$

Šis daugybos metodas yra kiek greitesnis už tiesioginį. Tačiau yra ir dar greitesnių. Palyginimui pasakysime, kad geriausiam iki šiol žinomam Schˋonhage-Strasseno algoritmui[16]

$$T(n) = \mathrm{O}(n \ln n \ln\ln n). \tag{88}$$

Natūraliojo skaičiaus dalybai iš kito skaičiaus irgi yra algoritmas, kurio veiksmų skaičiui teisingas (88) įvertis.

2 pavyzdys. *Patikrinti, ar natūralusis skaičius N yra pirminis.* Paprasčiausias (bet ne greičiausias!) būdas toks: patikrinti, ar N dalosi iš 2 ir visų nelyginių k, $3 \le k \le \sqrt{N}$. Tam blogiausiu atveju reikės apie $\sqrt{N}/2$ dalybos operacijų. Kadangi N dvejetainėje išraiškoje bus $n = [\log_2 N] + 1$ dvejetainių skaitmenų, tai reikalingą operacijų skaičių $T(n)$ galime vertinti taip:

$$T(n) = \mathrm{O}(\sqrt{N}) = \mathrm{O}\big(\mathrm{e}^{an}\big),$$

čia a – teigiama konstanta. Įvertis rodo, kad ilginant įeities duomenis skaičius operacijų, reikalingų „blogiausio“ atvejo sprendimui, didėja labai greitai. Tiesa, yra ir geresnių būdų šiam uždaviniui spręsti, žr. [3].

18.2. P ir NP uždavinių klasės

Vienus uždavinius sprendžiame nuosekliai atlikdami veiksmus iš pateikto sąrašo. Kitiems geriau taikyti tokį būdą: perskaičius sąlygą, nueiti miegoti arba žvejoti. Gal gera idėja kils netyčia, o tada jau uždaviniui išspręsti reikės visai mažų pastangų. Pirmuoju būdu sprendžiami **P** *klasės uždaviniai, o antruoju –* **NP**.

Ankstesniame skyrelyje išdėstytu požiūriu uždavinio sprendimas yra (determinuota) Turingo mašina. Ji visada realizuoja tam tikrą instrukcijų rinkinį, t. y. algoritmą. Dažnai vietoj Turingo mašinų kalbėsime tiesiog apie algoritmus.

Algoritmus klasifikuosime pagal jų Turingo mašinų darbo laiką.

105 apibrėžimas. *Algoritmas, kurio Turingo mašinos darbo laikas yra $T_{\mathrm{M}}(n)$, vadinamas polinominio laiko algoritmu, jei egzistuoja daugianaris*

[16] A. Schˋonhage, V. Strassen. Schnelle Multiplikation grosser Zahlen. Computing, 7(1971), p. 281–292.

p, kiekvienam $n \in \mathbf{N}$ tenkinantis nelygybę $T_{\mathrm{M}}(n) \leq p(n)$. Sakysime, kad uždavinys priklauso polinominio laiko uždavinių klasei **P**, *jei jos sprendimui žinomas polinominio laiko algoritmas*[17].

Paprastai laikomasi nuomonės, kad polinominio laiko algoritmas yra greitas ir jį galima praktiškai realizuoti. Todėl uždavinio priklausymas klasei **P** yra svarbi jo savybė. Tačiau praktiniam taikymui polinominis algoritmas gali būti net blogesnis už nepolinominį. Juk darbo laiko rėžius nustatančio daugianario laipsnis gali būti didelis!

Galima įsitikinti, kad pagrindinių aritmetinių bei palyginimo operacijų algoritmai priklauso klasei **P**. Todėl norint įrodyti, kad uždavinys priklauso klasei **P**, pakanka rasti jo sprendimo algoritmą, kuris turėtų polinominį laiką, išreikštą šių pagrindinių operacijų t.y sudėties, daugybos, palyginimo ir t. t. – skaičiumi. Kitaip sakant, nėra būtina kiekvieną kartą konstruoti Turingo mašiną. Prisiminkime ankstesnius pavyzdžius. Dviejų skaičių sandaugos, dalmens, liekanos ir bendrojo didžiausiojo daliklio radimas yra polinominio laiko uždaviniai. Tiktai visai neseniai buvo sugalvotas polinominio laiko algoritmas, nustatantis, ar duotasis skaičius yra pirminis, žr. [3].

O dabar panagrinėsime uždavinius, kuriems spręsti iki šiol nėra surasta polinominio laiko algoritmų. Tačiau tai būtų per daug platus ir neapibrėžtas tyrimo objektas. Todėl kalbėsime tik apie vadinamuosius nedeterminuoto polinominio laiko (NP) uždavinius. Apsiribosime išsprendžiamumo uždaviniais, nes į juos formaliai gali būti pertvarkyti kiti uždaviniai. Pradėsime vėl nuo pavyzdžių.

1 pavyzdys. *Patikrinti, ar natūralusis skaičius N yra sudėtinis.*

Jeigu skaitėme minėtą straipsnį apie polinominį algoritmą, tikrinantį, ar duotasis skaičius yra pirminis, tai ir šį uždavinį, žinoma, priskirsime **P** klasei. Juk klausimą, ar skaičius yra sudėtinis, galime pakeisti klausimu, ar jis pirminis.

Tarkime, kad to dar nesužinojome, todėl uždavinio nepriskiriame šiai klasei.

Taigi mums duotas didelis natūralusis skaičius N ir reikia pateikti atsakymą, ar jis sudėtinis. Tarkime, kad mes kažkaip sužinojome (galbūt susapnavome) problemos sprendimo raktą: kokį nors skaičiaus N daliklį d, $1 < d < N$. Aišku, kad šiuo atveju atsakymui gauti pakanka vienos dalybos operacijos, taigi jį išsprendžiame polinominio laiko algoritmu. Tačiau jeigu mums duotas skaičius nebūtų sudėtinis, tai gauti atsakymą per polinominį laiką nepadėtų jokie sapnai.

[17] Man patinka vietoj lotyniško žodžio „polinomas“ vartoti lietuvišką „daugianaris“. Jis skambus ir gerai perteikia prasmę. Tačiau termino „daugianarinis“ laikas pavartoti vis dėlto nesiryžau. Galima, žinoma, vartoti ilgą ir todėl nepatogų terminą, pavyzdžiui, „daugianariu apribotas laikas“. Tačiau tegu šiame skyrelyje bus „polinominis laikas“. Skaitytojams, kurie studijuos sudėtingumo teoriją iš kitomis kalbomis parašytų knygų bus lengviau atpažinti žinomas sąvokas (polynomial problem, algorithm angl.

Šis išsprendžiamumo uždavinys turi tokią savybę: jeigu į pateiktus duomenis atsakymas yra teigiamas, tai jį galima nustatyti per polinominį laiką, sužinojus tam tikrą papildomą informaciją (raktą), tačiau jeigu atsakymas yra neigiamas, tai surasti jį per polinominį laiką gali ir nepavykti. Tokius uždavinius vadinsime **NP** (nedeterminuoto polinominio laiko) uždaviniais.

Gautojo šifro dešifravimas paprastai būna **NP** klasės uždavinys: jei raktą žinome, galime jį išspręsti lengvai, jeigu ne – kartais sprendimas iš viso neįmanomas.

Pastebėsime, kad kiekvienas **P** klasės uždavinys yra ir **NP** klasės uždavinys.

2 pavyzdys. *Patikrinti, ar grafas yra Hamiltono, t. y. ar egzistuoja ciklas, jungiantis visas viršūnes ir einantis per kiekvieną iš jų tik po vieną kartą.*

Bendruoju atveju tai yra sudėtingas uždavinys, kuriam iki šiol nėra surastas polinominio laiko algoritmas. Akivaizdu, kad tai yra **NP** uždavinys. Jo raktas – Hamiltono ciklas.

Tačiau jeigu šį uždavinį suformuluotume šitaip: **ID:** *grafas* G;

U: *ar* G *nėra Hamiltono grafas?*

tai uždavinys jau nebepriklausytų klasei **NP**!

O dabar – kiek painokas matematinis **NP** uždavinio apibrėžimas. Primename, kad nagrinėjame tik išsprendžiamumo uždavinius, taigi klausimus galime interpretuoti taip: ar žodis $\mathbf{x} \in \mathcal{B}^*$ (įeities duomenys) priklauso uždavinio kalbai $\pi \subset \mathcal{B}^*$ (aibei duomenų, į kuriuos atsakymas yra teigiamas).

106 apibrėžimas. *Sakysime, kad* π *yra* **NP** *uždavinys (žymėsime* $\pi \in \mathbf{NP}$*), jei egzistuoja funkcija* $f : \mathcal{B}^* \times \mathcal{B}^* \to \{0, 1\}$ *tokia, kad*

1. $\mathbf{x} \in \pi$ *tada ir tik tada, kai egzistuoja toks* $\mathbf{y} \in \mathcal{B}^*$, *kad* $f(\mathbf{x}, \mathbf{y}) = 1$;
2. $f(\mathbf{x}, \mathbf{y})$ *skaičiuojama* $|\mathbf{x}|$ *atžvilgiu polinominio laiko algoritmu.*

Šiame apibrėžime žodis $\mathbf{y} \in \mathcal{B}^*$ ir yra uždavinio sprendimo raktas, kurį reikia kaip nors įspėti (kartais jis vadinamas sertifikatu).

Dabar įsitikinsime, kad **NP** uždavinių klasė turi savotišką struktūrą. Bet prieš tai dar viena nauja sąvoka.

107 apibrėžimas. *Sakoma, kad uždavinys* π_1 *yra polinomiškai pertvarkomas į uždavinį* π_2, *jei egzistuoja polinominio laiko funkcija* $\varphi : \mathcal{B}^* \to \mathcal{B}^*$, *tenkinanti sąlygą* $\mathbf{x} \in \pi_1$ *tada ir tik tada, kai* $\varphi(\mathbf{x}) \in \pi_2$.

Kitaip sakant, polinomiškai pertvarkyti uždavinį į kitą reiškia per polinominį laiką performuluoti klausimą, kad teigiamas atsakymas į jį reikštų taip pat teigiamą atsakymą į pirmąjį klausimą, o neigiamas – neigiamą.

Pavyzdžiui, visi žinome, kaip uždavinį

ID: *sveikieji skaičiai* a, b, c;

U: *ar lygtis* $ax^2 + bx + c = 0$ *turi du skirtingus sprendinius?*

galima pertvarkyti į uždavinį

ID: *sveikas skaičius* D;
U: *ar* D *teigiamas?*

124 teorema. *Jei* π_1 *yra polinomiškai pertvarkomas į* π_2 *ir* $\pi_2 \in \mathbf{P}$, *tai ir* $\pi_1 \in \mathbf{P}$.

Įrodymas. Tarkime, kad $\mathbf{x} \in \mathcal{B}^*$, $|\mathbf{x}| = n$, φ – polinominio laiko funkcija, pertvarkanti π_1 į π_2. Tada $|\varphi(\mathbf{x})| \leq g(n)$, čia g – daugianaris. Į klausimą, ar $\mathbf{x} \in \pi_1$, atsakysime patikrinę, ar $\varphi(\mathbf{x}) \in \pi_2$. Prisiminę, kad $\pi_2 \in \mathbf{P}$, galime teigti, kad šį tikrinimą galima atlikti algoritmu, kurio laikas neviršija $p(|\varphi(\mathbf{x})|) \leq q(n)$, čia p, q – daugianariai. Taigi $\pi_1 \in \mathbf{P}$.

Dabar jau galime suformuluoti vieną įžymią teoremą apie santykius **NP** uždavinių aibėje.

125 teorema. *(S. A. Cook, 1973) Egzistuoja* **NP** *uždavinys, į kurį polinomiškai galima pertvarkyti bet kurį kitą* **NP** *uždavinį.*

Toks universalus uždavinys vadinamas *pilnuoju* **NP** uždaviniu. Tiesą sakant, Cookas suformulavo šią teoremą kiek kitaip. Jis pateikė ir **NP** pilno uždavinio pavyzdį. Dabar tokių pavyzdžių žinoma labai daug. 1979 m. Garey ir Johnsonas pateikė katalogą [30], kuriame yra virš 3000 pilnųjų NP problemų iš skaičių teorijos, logikos, kombinatorikos ir automatų teorijos. Tarp jų yra ir mūsų jau nagrinėta grafo Hamiltono ciklo egzistavimo uždavinys.

Polinominių uždavinių aibė sudaro **NP** uždavinių aibės poaibį. Tiesą sakant, niekas pasaulyje nežino, ar **NP** uždavinių yra iš tikrųjų daugiau negu **P** klasės uždavinių! Juk, pavyzdžiui, niekas negali paneigti, kad kas nors nesuras polinominio laiko algoritmo Hamiltono ciklo radimo grafe uždaviniui spręsti. Jeigu tai įvyktų, būtų nustatyta, kad šis uždavinys priklauso klasei **P**, o tuo pačiu metu... kad visi **NP** uždaviniai sprendžiami polinominio laiko algoritmais, t. y. kad $\mathbf{P} = \mathbf{NP}$!

Prieš keletą metų Clay matematikos institutas Kembridže (JAV, Masachusetts), kurį finansuoja verslininkas Landonas T. Clay, paskelbė septynių matematinių uždavinių sąrašą. Atlygis už kiekvieno iš jų sprendimą – milijonas JAV dolerių. Vienas iš šių uždavinių formuluojamas itin paprastai:

įrodyti arba paneigti, kad $\mathbf{P} = \mathbf{NP}$.

Taigi milijoną dolerių galima uždirbti tyrinėjant paprasčiausius grafus.

18.3. Nepilnas NP pilnų problemų sąrašėlis

> **NP** *pilnų uždavinių žinoma labai daug. Ir iš nedidelio skyrelyje pateikiamo sąrašo susidarysite įspūdį apie tokių uždavinių įvairovę.*

Minėjome, kad **NP** pilnų uždavinių dabar jau žinoma keli tūkstančiai. Jų galima rasti visose diskrečiosios matematikos srityse. Tuo įsitikinsite peržvelgę ir šį nedidelį **NP** uždavinių rinkinį.

Visur $G = \langle V, E \rangle$ reiškia grafą, V – jo viršūnių aibė, E – briaunų aibė.

Viršūnių denginys (vertex cover)

ID: $G = \langle V, E\rangle$, $k \leq |V|$;

U: *ar egzistuoja* $V' \subset V, |V'| \leq k$, *kad kiekvienos briaunos viršūnė priklausytų* V'*?*

Grafo k-spalvinimas (graph k-colorability)

ID: $G = \langle V, E\rangle$, $k \leq |V|$;

U: *ar galima viršūnes nuspalvinti* k *spalvomis, kad bet kurios briaunos dvi viršūnės būtų nuspalvintos skirtingai?*

Vienspalvis trikampis (monochromatic triangle)

ID: $G = \langle V, E\rangle$;

U: *ar galima nuspalvinti visas grafo briaunas dviem spalvomis, kad iš tos pačios spalvos briaunų ir atitinkamų viršūnių negalėtume sudaryti vieno trikampio?*

Komercinis keliautojas (traveling salesman)

ID: *miestų aibė* C, *atstumai tarp miestų* $d(c_i, c_j)$, *skaičius* $B > 0$;

U: *ar egzistuoja būdas apeiti visus miestus, kad viso kelio ilgis neviršytų* B*?*

Aibių pakavimas (set packing)

ID: *baigtinės aibės poaibių rinkinys* C, *skaičius* $k, 0 < k \leq |C|$;

U: *ar rinkinyje* C *egzistuoja* k *poromis nesikertančių aibių?*

Aibių skaidymas (set splitting)

ID: *baigtinės aibės* S *poaibių rinkinys* C;

U: *ar galima* S *išskaidyti į dviejų nesikertančių aibių sąjungą* $S = S_1 \cup S_2$, *kad jokia* C *aibė nebūtų nei* S_1, *nei* S_2 *poaibis?*

Trumpiausias viršžodis (shortest common supersequence)

ID: *baigtinė abėcėlė* $\Sigma, R \subset \Sigma^*$ *ir natūralusis skaičius* k;

U: *ar egzistuoja žodis* $w \in \Sigma^*$, *kad* $|w| \leq k$ *ir kiekvienas žodis* $x \in R$ *būtų* w *fragmentas (dalis)?*

Skaidymas (partition)

ID: *baigtinė aibė* A *ir neneigiami sveikieji skaičiai (elementų svoriai)* $s(a), a \in A$;

U: *ar egzistuoja* $A' \subset A$, *kad*

$$\sum_{a \in A'} s(a) = \sum_{a \in A \setminus A'} s(a)?$$

Kvadratiniai lyginiai (quadratic congruences)

ID: *natūralieji skaičiai* a, b, c;

U: *ar egzistuoja natūralusis skaičius* $x, |x| < c$, *kad* $x^2 \equiv a \pmod{b}$?

Dalumo palyginimas (comparative divisibility)

ID: *du natūraliųjų skaičių rinkiniai* $\{a_1, \ldots, a_n\}$ *ir* $\{b_1, \ldots, b_m\}$;

U: *ar yra toks natūralusis skaičius* c, *kad skaičių iš pirmojo rinkinio, kuriuos dalo* c, *yra daugiau nei analogiškų skaičių iš antrojo rinkinio?*

Kvadratinės Diofanto lygtys (quadratic diophantine equations)

ID: *natūralieji skaičiai* a, b, c;

U: *ar egzistuoja natūralieji skaičiai* x, y *su sąlyga* $ax^2 + by = c$*?*

Kryžiažodžio konstravimas (crossword puzzle construction)

ID: *baigtinė abėcėlė* Σ*, baigtinė žodžių aibė* $W \subset \Sigma^*$*,* $n \times n$ *matrica* A*, kurios elementai yra vienetai;*

U: *ar galima iš* W *žodžių sukonstruoti kryžiažodį pagal* A *kaip šabloną (rašant raides į langelius, kurie pažymėti* 0*, o langelius, pažymėtus* 1*, laikant užtušuotais)?*

18.4. Tikimybiniai algoritmai

Deterministinio algoritmo taikytojas apie sunkų uždavinį galvoja taip: kadangi uždavinys sunkus, tai, norint gauti atsakymą, reiks daug dirbti. Tikimybinio algoritmo taikytojo požiūris kitoks: nors uždavinys sunkus, bet jeigu pasiseks, išspręsiu jį lengvai.

Tikimybinių algoritmų idėja labai paprasta, juos mes gyvenime neretai naudojame. Tarkime, reikia atsakyti į klausimą, ar ežere yra žuvų. Algoritmas paprastas – užmerkti meškerę ir palaukti. Galimos dvi baigtys: pagausime žuvį arba nepagausime. Pirmuoju atveju atsakymas į klausimą yra teigiamas ir visiškai teisingas, antruoju – atsakymas neigiamas, tačiau jo teisingumas yra tik tikėtinas, o ne absoliutus.

Sudarysime panašias savybes turintį algoritmą tokiam uždavinius spręsti:

ID: N *– nelyginis skaičius;*

U: *ar* N *sudėtinis?*

Pasinaudosime Fermat teorema (šį labai svarbų visai viešojo rakto kriptografijai teiginį įrodysime kitame skyriuje).

126 teorema. *Jei* N *– nelyginis pirminis skaičius,* $(a, N) = 1$*, tai* $a^{N-1} \equiv 1 \ (mod\ N)$.

Dabar galime tikrinti, ar N yra pirminis, taip:

- parinkime nelyginį $w, 1 < w < N$; jei $w|N$, tai N – sudėtinis skaičius;
- jei $(w, N) = 1$, skaičiuokime $w^{N-1} (\text{mod } N)$. Jei $w^{N-1} \not\equiv 1 (\text{mod } N)$, tai N sudėtinis, jei $w^{N-1} \equiv 1 (\text{mod } N)$, darome išvadą, kad N yra pirminis.

Aišku, kad teigdami, jog N pirminis, ne visada būsime teisūs. Galima atveju $w^{N-1} \equiv 1 \ (\text{mod } N)$ kartoti testą, parinkus kitą w reikšmę. Dažnai šitokiu būdu pavyksta „demaskuoti“ pirminiu apsimetusį sudėtinį skaičių. Deja, ne visada. Egzistuoja be galo daug skaičių[18] N, kurie su visais $(w, N) = 1$ tenkina sąlygą $w^{N-1} \equiv 1 \ (\text{mod } N)$. Mažiausias Carmichaelio skaičius yra 561. Tačiau visiems kitiems sudėtiniams skaičiams nepavyks išlikti neatpažintiems, jeigu tik testas bus pakankamai daug kartų pakartotas. Tai išplaukia iš tokios teoremos:

[18] Jie vadinami Carmichaelio skaičiais; R. D. Carmichael (1879–1967).

127 teorema. *Tegu N – nelyginis skaičius. Tada sąlyga $w^{N-1} \equiv 1 \ (mod\ N)$ teisinga arba visiems $w, (w,N) = 1, 1 \leq w < N$, arba daugiausiai pusei tokių w.*

Dabar apibrėžkime tikimybinio algoritmo sąvoką.

108 apibrėžimas. *Algoritmą išsprendžiamumo uždaviniui spręsti vadinsime tikimybiniu, jeigu įeities duomenims jo pateikiamas atsakymas $TAIP$ yra visada teisingas, o NE – teisingas su tikimybe, didesne už $1/2$.*

Teiginys *teisingas su tikimybe, didesne už* $1/2$, reiškia, kad, su įeities duomenimis, kuriems teisingas atsakymas yra NE, atlikus algoritmą visais įmanomais būdais, šis atsakymas bus gautas daugiau kaip pusę kartų. Taigi dėl Carmichaelio skaičių negalime mūsų algoritmo, pagrįsto Fermat teorema, pavadinti tikimybiniu. Juk jeigu pasitaikė taip, kad mūsų įeities duomuo yra Carmichaelio skaičius, tai tikimybė, kad algoritmas duos teisingą atsakymą, lygi nuliui!

Jeigu tikimybinio algoritmo naudojamo laiko kiekis yra polinominis įeities duomenų dydžio atžvilgiu, tai algoritmą vadinsime tikimybiniu polinominiu algoritmu, o jo sprendžiamą uždavinį – tikimybiniu polinominiu uždaviniu. Tikimybinių polinominių uždavinių klasė literatūroje žymima **RP** (*random polynomial* angl.).

Dabar sukonstruosime tikrą tikimybinį polinominį algoritmą sudėtinių skaičių atpažinimo uždaviniui spręsti.

Tegu N – nelyginis skaičius. Suraskime skaičius $s, t, (t,2) = 1$, kad būtų $N - 1 = 2^s t$. Parinkime $a, (a,N) = 1$ ir patikrinkime, ar $a^t \not\equiv 1 \pmod N$. Jei galioja lygybė, parinkime kitą a. Suskaičiuokime

$$a^t \pmod N,\ a^{2t} \pmod N,\ \ldots \quad a^{2^{s-1}t} \pmod N,\ a^{N-1} \pmod N. \quad (89)$$

Tarkime, N yra pirminis skaičius. Kiek vienetų yra sekoje (89)? Iš Fermat teoremos gauname, kad paskutinis šios sekos skaičius yra būtinai vienetas. Vienetas gali pasitaikyti ir anksčiau (pirmas skaičius tikrai ne vienetas). Tegu pirmasis sekos vienetas yra

$$a^{2^l t} \equiv 1 \pmod N,$$

tada prieš jį einantis skaičius $b \equiv a^{2^{l-1}t} \pmod N$ yra lyginio $x^2 \equiv 1 \pmod N$ sprendinys. Kadangi N yra pirminis, tai šis lyginys turi tik du sprendinius: 1 ir $N - 1$. Iš sąlygos $b \not\equiv 1 \pmod N$, gauname, kad $b \equiv N - 1 \pmod N$.

Taigi pirminiam skaičiui N teisingi tokie teiginiai:

1. paskutinis (89) sekos skaičius yra 1;
2. prieš pirmąjį (89) sekos vienetą stovi $N - 1$.

Dabar algoritmą galime aprašyti taip:

- surasti skaičius $s, t, (t, 2) = 1$, kad $N - 1 = 2^s t$;
- parinkti $1 < a < N$; jei $a|N$, tai N – sudėtinis;
- jei $a^t \equiv 1 \pmod{N}$, parinkti kitą a;
- sudaryti seką (89);
- jei bent vienas iš teiginių 1), 2) (89) sekai neteisingas, tai N – sudėtinis skaičius; jei abu teisingi – pirminis.

Šis algoritmas vadinamas Millerio (arba Millerio-Rabino) testu. Algoritmas polinominis. Ar jį galime pavadinti tikimybiniu, t. y. ar jis tenkina mūsų apibrėžimo sąlygas? Teigiamas atsakymas, žinoma, visada yra teisingas. Reikia įsitikinti, kad neigiamas atsakymas teisingas su tikimybe, didesne už 1/2. Tai išplaukia iš tokio teiginio:

128 teorema. *Tegu N – nelyginis sudėtinis skaičius, $4 < N$. Tada ne mažiau kaip $3(N-1)/4$ skaičių $a, 1 < a < N$, arba teiginys 1), arba 2) neteisingas.*

Taigi Millerio-Rabino testas – tikimybinis polinominis algoritmas. Tikimybė, kad sudėtinį N pripažinsime pagal šį testą pirminiu, yra ne didesnė už 1/4. Pakartojus testą k kartų, klaidingo sprendimo tikimybė bus ne didesnė už $1/4^k$. Matome, kad, pakartoję testą daug kartų, ją galima kiek norime sumažinti.

19 Skaičių teorijos sąvokos ir algoritmai

Viešojo rakto kriptosistemų, skaitmeninių parašų ir kitų moderniosios kriptografijos schemų pagrindas – veiksmai baigtinėse algebrinėse struktūrose. Tos struktūros – dalybos liekanų žiedai $\mathbb{Z}_n$, baigtiniai kūnai $\mathbb{F}_p$, $\mathbb{F}_{p^m}$... Jas tyrinėjo pačios „gryniausios“ matematikos atstovai, niekada turbūt nemanę, kad jų teiginiai bus pritaikyti labai praktiškiems tikslams.

Uždavinius sprendžiame naudodamiesi tam tikromis taisyklėmis – algoritmais. Jie nurodo, kokius veiksmus reikia atlikti su duomenimis, kad gautume rezultatą. Veiksmų skaičius priklauso nuo uždavinio duomenų dydžio. Kuo matuoti duomenų dydį? Natūrali idėja – dvejetainės abėcėlės simbolių skaičiumi, reikalingu duomenims užrašyti. Kai duomenų dydis auga (pavyzdžiui, skaičiuojame su vis didesniais skaičiais) veiksmų skaičius irgi didėja. Jeigu veiksmų skaičius, reikalingas algoritmui su n ilgio duomenimis atlikti neviršija tam tikro daugianario f reikšmės $f(n)$, toks algoritmas vadinamas polinominiu. Paprastai laikoma, kad uždavinys, kuriam spręsti žinome polinominį algoritmą, yra sprendžiamas greitai. Tačiau tai tik paviršutiniška, nors dažnai ir teisinga nuomonė. Algoritmų vertinimo pagal sudėtingumą uždaviniai nagrinėjami sudėtingumo teorijoje. Norintys į ją įsigilinti turėtų pavartyti, pavyzdžiui, [30], [47] knygas. Mes tik retkarčiais pabandysime

įvertinti algoritmui atlikti reikalingų veiksmų skaičių. Tokiuose įverčiuose patogu naudoti žymenį $f = \mathrm{O}(g)$, kuris reiškia, kad dydis f neviršija dydžio $c \cdot g$, čia $c > 0$ yra konstanta, kurios tiksli reikšmė paprastai nėra svarbi.

19.1. Euklido algoritmas

Bendrąjį didžiausiąjį dviejų skaičių daliklį Antikos graikai suvokė kaip bendrąjį skaičių matą. Euklido algoritmas jam rasti – nedaug matematinių kūrinių gali jam prilygti paprastumu, teorine ir praktine reikšme.

Kas yra bendrasis didžiausiasis dviejų natūrinių skaičių daliklis, visi žinome. Nuo jo ir pradėkime.

109 apibrėžimas. *Bendruuoju didžiausiuoju natūrinių skaičių a, b dalikliu vadinamas didžiausias skaičius, kuris dalija ir a, ir b. Bendrąjį didžiausiąjį daliklį žymėsime (a, b). Jeigu $(a, b) = 1$, šiuos skaičius vadinsime tarpusavyje pirminiais.*

Jeigu b dalijasi iš a, tai, žinoma, $(a, b) = b$.

Darni natūrinių skaičių dalumo teorija išdėstyta Euklido „Pradmenyse“. Visi skaičių teorijos vadovėliai, kuriuose teiginiai dėstomi nuosekliai ir išsamiai, ją pakartoja. Bet mums juk pirmiausia rūpi ne loginiai teorijos pagrindai, bet jos teiginiai, kuriuos kaip įrankius galima panaudoti kriptografijoje. Taigi praleiskime įrodymus tų teiginių, kurie tiesiog „akivaizdūs“. Pavyzdžiui,

jei $(a, b) = d$, tai $a = da', b = db'$ ir $(a', b') = 1$.

Skaičių dalybą su liekana jau aptarėme kodavimo teorijai skirtoje knygos dalyje. Jeigu $a > b$ yra natūralieji skaičiai, tai

$$a = qb + r, \quad 0 \le r < b.$$

Euklido algoritmas bendrajam didžiausiajam skaičių a, b dalikliui rasti remiasi teiginiu: $(a, b) = (b, r)$.

Jei $a > b$ – natūralieji skaičiai, tai, kartodami dalybos su liekana veiksmus, gausime:

$$\begin{aligned}
a &= q_0 b + r_0, \quad 0 < r_0 < b, \\
b &= q_1 r_0 + r_1, \quad 0 < r_1 < r_0, \\
r_0 &= q_2 r_1 + r_2, \quad 0 < r_2 < r_1, \\
&\cdots\cdots\cdots\cdots \\
r_{m-2} &= q_m r_{m-1} + r_m, \quad 0 < r_m < r_{m-1}, \\
r_{m-1} &= q_{m+1} r_m + r_{m+1}, \quad 0 < r_{m+1} < r_m, \\
&\cdots\cdots\cdots\cdots \\
r_{n-2} &= q_n r_{n-1} + r_n, \quad 0 < r_n < r_{n-1}, \\
r_{n-1} &= q_{n+1} r_n.
\end{aligned}$$

Tada paskutinė nelygi nuliui liekana ir yra bendrasis didžiausiasis daliklis:

$$(a,b) = (b,r_0) = (r_0,r_1) = \ldots = (r_{n-1},r_n) = r_n.$$

Štai ir visa Euklido algoritmo esmė.

Iš priešpaskutinės lygybės gausime:

$$(a,b) = r_n = r_{n-2} + (-q_n)r_{n-1}.$$

Įstatę į šią lygybę r_{n-1} išraišką per r_{n-2}, r_{n-3}, gausime bendrojo didžiausiojo daliklio išraišką per liekanas su mažesniais indeksais. Galų gale, šitaip kopdami į viršų, surasime sveikuosius x, y, kad

$$(a,b) = xa + yb.$$

Pastebėkime, kad jei $(a,b) = ur_m + vr_{m+1}$, tai $(a,b) = vr_{m-1} + (u - vq_{m+1})r_m$.

Pasinaudoję šia pastaba, galime ieškoti skaičių x, y taip. Surašykime Euklido algoritmo skaičiavimus tokioje lentelėje:

r_{i-1}, r_i	q_{i+1}	u_{i-1}, u_i
a, b	q_0	
b, r_0	q_1	
r_0, r_1	q_2	
$\ldots$	$\ldots$	$\ldots$
r_{m-1}, r_m	q_{m+1}	$v, u - vq_{m+1}$
r_m, r_{m+1}	q_{m+2}	u, v
$\ldots$	$\ldots$	$\ldots$
r_{n-2}, r_{n-1}	q_n	$1, -q_n$
r_{n-1}, r_n	q_{n+1}	

Trečiąjį stulpelį pildome nuo apačios į viršų. Jeigu paskutinė užpildyta eilutė yra tokia: $r_m, r_{m+1} \mid q_{m+2} \mid u; v$, tai į viršutinės eilutės trečiąjį stulpelį rašome skaičius $v; u - vq_{m+1}$. Su kiekvienos eilutės skaičiais $r_{i-1}, r_i, u_{i-1}, v_{i-1}$ teisinga lygybė

$$(a,b) = u_{i-1}r_{i-1} + v_{i-1}r_i.$$

Pirmosios eilutės skaičiai duoda lygybę

$$ax + by = (a,b).$$

Lentelėje pateiktas skaičiavimo pavyzdys su $a = 57, b = 10$.

$57; 10$	5	$3; -17;$	$1 = 57 \cdot 3 + 10 \cdot (-17)$
$10; 7$	1	$-2; 3;$	$1 = 10 \cdot (-2) + 7 \cdot 3$
$7; 3$	2	$1; -2;$	$1 = 7 \cdot 1 + 3 \cdot (-2)$
$3; 1$	3		

Panagrinėkime, kiek daugiausia Euklido algoritmo žingsnių reikia atlikti, kad surastume bendrąjį didžiausiąjį daliklį. Apibrėžkime skaičius

$$f_0 = 1,\ f_1 = 2,\ f_i = f_{i-1} + f_{i-2}\ (i \geq 2).$$

Kas nors, be abejonės, pastebėjo, kad tai – Fibonačio skaičiai. Įsitikinsime, kad mūsų dalybos liekanoms teisingos nelygybės:

$$r_n \geq f_0,\ r_{n-1} \geq f_1,\ \ldots,\ r_{n-m} \geq f_n,\ \ldots, r_0 \geq f_n,\ b \geq f_{n+1}.$$

Iš tikrųjų, nelygybės $r_n \geq f_0,\ r_{n-1} \geq f_1$ akivaizdžios. Jeigu teisingos nelygybės $r_{n-m+2} \geq f_{m-2}, r_{n-m+1} \geq f_{m-1}$, tai

$$r_{n-m} = q_{n-m+1} r_{n-m+1} + r_{n-m+2} \geq r_{n-m+1} + r_{n-m+2} \geq f_{m-1} + f_{m-2} = f_m.$$

Taigi teisinga nelygybė $b \geq f_{n+1}$. Tačiau savo ruožtu Fibonačio skaičiai tenkina nelygybes

$$f_m \geq \alpha^m, \quad \alpha = \frac{1+\sqrt{5}}{2}, \quad \alpha^2 = \alpha + 1.$$

Iš tikrųjų, nelygybės $f_0 \geq \alpha^0, f_1 \geq \alpha^1$ yra akivaizdžios. Jeigu teisingos nelygybės $f_{m-1} \geq \alpha^{m-1}, f_m \geq \alpha^m$, tai

$$f_{m+1} = f_m + f_{m-1} \geq \alpha^m + \alpha^{m-1} = \alpha^{m-1}(\alpha + 1) = \alpha^{m-1}\alpha^2 = \alpha^{m+1}.$$

Matematinės indukcijos metodu nelygybės Fibonačio skaičiams įrodytos. Tada

$$\alpha^{n+1} \leq f_{n+1} \leq b, \quad n \leq \log_\alpha b - 1.$$

Jeigu skaičiui a užrašyti reikia N dvejetainių skaitmenų, o vienam Euklido žingsniui atlikti reikalingų operacijų skaičių vertinsime dydžiu $\mathrm{O}(N^2)$, tai visam algoritmui reikalingų operacijų skaičių galime įvertinti dydžiu

$$\mathrm{O}(n \cdot N^2) = \mathrm{O}(\log_\alpha b \cdot N^2) = O(N^3).$$

Taigi bendrojo didžiausiojo daliklio radimo algoritmas yra polinominis; iš tikrųjų, vertinant operacijų skaičių tiksliau, galima gauti įvertį $O(N^2)$.

Skaičius α, kuris padėjo išspręsti uždavinį, yra labai įžymus: tai aukso pjūvio skaičius.

19.2. Apie žiedus $\mathbb{Z}_n$

Šio skyrelio teiginių atradėjai sudaro ryškiausią matematikų žvaigždyną. Lyginius sukūrė Gausas, Fermat ir Euleris įrodė svarbias skaičių teorijai (ir kriptografijai) teoremas ir, žinoma, vėl – Euklidas.

Jeigu sveikųjų skaičių a ir b skirtumas dalijasi iš n, rašome

$$a \equiv b \pmod{n}.$$

Tokį sąryšį vadiname lyginiu, skaitome: a lygsta b moduliu n. Toks sąryšis teisingas tada ir tik tada, kai, dalijant a ir b iš n, gaunama ta pati liekana.

Baigtiniams kūnams skirtame skyrelyje suformuluotas teiginys apie paprasčiausias lyginių savybes:

jei $a \equiv b \pmod{n}$ ir $c \equiv d \pmod{n}$, tai $a + c \equiv b + d \pmod{n}$;
jei $a \equiv b \pmod{n}$ ir c yra sveikasis skaičius, tai $ca \equiv cb \pmod{n}$;
jeigu $ca \equiv cb \pmod{n}$ ir $(c, n) = 1$, tai $a \equiv b \pmod{n}$.

Naudodamiesi šiomis savybėmis, galime atlikti veiksmus su lyginiais, t.y. iš vienų lyginių gauti kitus. Kiekvieną lyginį moduliu n, kurio abi pusės sudarytos iš skaičių ar simbolių, naudojant sudėties ir daugybos veiksmus, galime interpretuoti kaip dalybos liekanų žiedo $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$ elementų sąryšį. Pakanka skaičius pakeisti jų dalybos liekanomis, o veiksmus – liekanų žiedo veiksmais. Primename, kad šiame žiede apibrėžtos tokios sudėties ir daugybos operacijos:

$$\begin{aligned} a +_n b &= a + b \text{ dalybos iš } n \text{ liekana}, \\ a \times_n b &= a \cdot b \text{ dalybos iš } n \text{ liekana}. \end{aligned}$$

Taip pat ir reiškinius su šiais veiksmais galime skaičiuoti naudodamiesi lyginių žymeniu ir savybėmis. Pavyzdžiui, apskaičiuokime $5^3 \times_9 7^6$:

$$5^3 \cdot 7^6 \equiv 25 \cdot 5 \cdot (49)^3 \equiv 7 \cdot 5 \cdot 4^3 \equiv (-1) \cdot 1 \equiv 8 \pmod{9}.$$

Taigi $5^3 \times_9 7^6 = 8$.

Dažniausiai veiksmams liekanų žiede žymėti vartosime įprastinės skaičių sudėties ir daugybos ženklus.

Kodavimo teorijoje svarbiausias veiksmas buvo žiedo elementų sudėtis ir daugyba, kriptografijoje, kaip netrukus pamatysime, – kėlimas laipsniu, t.y. daugelio vienodų elementų daugyba.

Kėlimo laipsniu operaciją žiede $\mathbb{Z}_n$ galime atlikti labai sparčiu „kėlimo kvadratu“ algoritmu. Panagrinėkime skaitinį pavyzdį. Tarkime, žiede $\mathbb{Z}_{11}$

reikia apskaičiuoti 10^{31}, t.y. reikia rasti skaičiaus $100\ldots0$ su 31 nuliu dalybos iš 11 liekaną. Kas ims dalyti – pražus. Skaičiuokime kitaip.

Užrašę 31 dvejetainėje skaičiavimo sistemoje, gausime:

$$31 = 1+1\cdot2^1+1\cdot2^2+1\cdot2^3+1\cdot2^4, \quad 10^{31} = 10^1\cdot10^2\cdot(10^2)^2\cdot((10^2)^2)^2\cdot(((10^2)^2)^2)^2.$$

Dabar skaičiuojame

$$10^2 \equiv 7 \pmod{31}, \quad 7^2 \equiv 18 \pmod{31},$$
$$18^2 \equiv 14 \pmod{31}, \quad 14^2 \equiv 10 \pmod{31},$$

taigi

$$10^{31} \equiv 10\cdot7\cdot18\cdot14\cdot10 \equiv 25 \pmod{31}.$$

Jeigu $n = p$ yra pirminis skaičius, tai žiedas $\mathbb{Z}_p$ yra kūnas; jį paprastai žymėdavome $\mathbb{F}_p$. Žinome, kad kiekvienas nenulinis šio kūno elementas a turi atvirkštinį, t.y. egzistuoja $b \in \mathbb{F}_p$, paprastai žymimas a^{-1}, kad

$$a \times_n b = 1, \quad \text{t.y. } a \times_n a^{-1} = 1.$$

O kaipgi bendruoju atveju?

110 apibrėžimas. *Tegu $n \geq 1$ yra natūralusis skaičius,*

$$\mathbb{Z}_n^* = \{m : 1 \leq m < n, (m,n) = 1\}.$$

Funkciją $\varphi(n) = |\mathbb{Z}_n^*|$, apibrėžtą natūraliųjų skaičių aibėje, vadinsime Eulerio funkcija.

Eulerio funkcijos $\varphi(n)$ reikšmė lygi mažesnių už n ir tarpusavyje pirminių su juo skaičių kiekiui. Pavyzdžiui,

$$\varphi(1) = \varphi(2) = 1, \ \varphi(3) = \varphi(4) = 2, \ldots$$

Jeigu p yra pirminis, tai nesunku suvokti, kad

$$\varphi(p) = p-1, \quad \varphi(p^m) = p^m - p^{m-1}. \tag{90}$$

O dabar imkimės atvirkštinių klausimo.

129 teorema. *Kiekvienas $\mathbb{Z}_n^*$ elementas turi atvirkštinį.*
Jeigu $a \in \mathbb{Z}_n^*$, tai

$$a^{\varphi(n)} \equiv 1 \ (mod\ n). \tag{91}$$

Įrodymas. Tegu

$$\mathbb{Z}_n^* = \{a_0, a_1, \ldots, a_{\varphi(n)}\}, \quad a_0 = 1,$$

ir $a \in \mathbb{Z}_n^*$. Pirmiausia reikia įsitikinti, kad visi skaičiai

$$a\cdot a_0, a\cdot a_1, \ldots, a\cdot a_{\varphi(n)} \tag{92}$$

atitinka skirtingus $\mathbb{Z}_n^*$ elementus, t.y. duoda skirtingas dalybos iš n liekanas. Padarę prielaidą, kad taip nėra, tuoj pat gautume prieštaravimą. Tačiau jeigu visi (92) atitinka skirtingus $\mathbb{Z}_n^*$ elementus, tai vienas atitinka $a_0 = 1$. Todėl atsiras toks $b \in \mathbb{Z}_n^*$, kad $a \times_n b = 1$.

Taigi (92) yra tie patys elementai $a_0, a_1, \ldots, a_{\varphi(n)}$, tik kitaip persirikiavę. Sudauginę juos visus ir pasinaudoję lyginių savybe, gausime

$$a^{\varphi(n)} \cdot a_0 \cdot a_1 \cdots a_{\varphi(n)} \equiv a_0 \cdot a_1 \cdots a_{\varphi(n)} \pmod{n}, \quad a^{\varphi(n)} \equiv 1 \pmod{n}.$$

Dar keletas pastabų apie šią teoremą. Iš tikrųjų įrodėme kiek daugiau negu teigėme. Įrodėme, kad daugybos veiksmo atžvilgiu aibė $\mathbb{Z}_n^*$ yra grupė.

Teiginys (91) skaičių teorijoje vadinamas Eulerio teorema. Vienas jos atvejis nusipelno atskiro vardo.

130 teorema. *(Mažoji Fermat teorema.) Jeigu p yra pirminis skaičius ir $(a, p) = 1$, tai*

$$a^{p-1} \equiv 1 \ (mod\ p).$$

Kriptografijai nepakanka žinoti, kokie elementai turi atvirkštinius, reikia mokėti juos greitai surasti. Tačiau greitas metodas jau yra!

Tegu $a \in \mathbb{Z}_n^*$, reikia rasti jo atvirkštinį. Kadangi $(a, n) = 1$, tai, pasinaudoję Euklido algoritmu, galime rasti sveikuosius skaičius x, y, kad būtų

$$ax + ny = 1.$$

Tačiau iš šios lygybės išplaukia lyginys $ax \equiv 1 \pmod{n}$. Tada x dalybos iš n liekana yra a atvirkštinis.

Pavyzdžiui, su $n = 57, a = 10$ iš lygybės $1 = 57 \cdot 3 + 10 \cdot (-17)$ gauname

$$10 \cdot (-17) \equiv 1 \pmod{57}, \quad 10 \cdot 40 \equiv 1 \pmod{57}, \quad 10^{-1} \equiv 40 \pmod{57}.$$

Kitas būdas surasti atvirkštinį – pasinaudoti Eulerio teorema: jei $(a, n) = 1$, tai

$$a^{\varphi(n)} \equiv 1 \pmod{n}, \quad a \cdot a^{\varphi(n)-1} \equiv 1 \pmod{n},$$

taigi $a^{-1} \equiv a^{\varphi(n)-1} \pmod{n}$.

O dabar dar patyrinėkime Eulerio funkciją. Kiekvieną natūralųjį skaičių n galime užrašyti pirminių skaičių laipsnių sandauga:

$$n = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_t^{\alpha_t}, \quad p_1 < p_2 < \ldots < p_t,$$

čia p_i yra pirminiai skaičiai. Šis teiginys vadinamas pagrindine aritmetikos teorema. Nors tokiu pavidalu jis suformuluotas gana neseniai, juo naudojosi ir Euklido laikų matematikai. Žinodami visus pirminius skaičiaus n daliklius, galime jais pasinaudoti ir užrašyti Eulerio funkcijos išraišką.

131 teorema. *Jeigu $p_1 < p_2 < \ldots < p_t$ yra visi pirminiai skaičiaus n dalikliai, tai*

$$\varphi(n) = n\left(1 - \frac{1}{p_1}\right) \cdot \left(1 - \frac{1}{p_2}\right) \cdots \left(1 - \frac{1}{p_t}\right). \tag{93}$$

Įrodymas. Pažymėkime $A = \{m : 1 < m \leq n, (m, n) > 1\}$. Tada $\varphi(n) = n - |A|$. Savo ruožtu jei $A_i = \{m : 1 < m \leq n, (p_i, n) = p_i\}$, tai $A = \bigcup_{i=1}^{t} A_i$. Taigi reikia apskaičiuoti, kiek skaičių yra šioje aibių sąjungoje. Pasinaudokime kombinatorine „pliuso-minuso“ taisykle:

$$\left|\bigcup_{i=1}^{t} A_i\right| = \sum_{i=1}^{t} |A_i| - \sum_{1 \leq i_1 < i_2 \leq t} |A_{i_1} \cap A_{i_2}| + \ldots + (-1)^{t+1} |A_1 \cap A_2 \cap \ldots \cap A_t|.$$

Aibių sankirtai $A_{i_1} \cap \ldots \cap A_{i_r}$ priklauso tie ne didesni už n skaičiai, kurie dalijasi iš pirminių skaičių sandaugos $p_{i_1} p_{i_2} \ldots p_{i_r}$. Tokių skaičių yra $n/(p_{i_1} p_{i_2} \ldots p_{i_r})$. Taigi

$$n - |A| = n - \sum_{i=1}^{t} \frac{n}{p_i} + \sum_{n \leq i_1 < i_2 \leq t} \frac{n}{p_{i_1} p_{i_2}} - \ldots + (-1)^t \frac{n}{p_1 p_2 \ldots p_t}. \tag{94}$$

Belieka įsižiūrėti ir įsitikinti, kad reiškiniai (93) ir (94) yra lygūs.

Pasinaudojus gautąja Eulerio funkcijos išraiška nesunku įrodyti tokią šios funkcijos savybę:

132 teorema. *Jeigu natūriniai skaičiai m, n yra tarpusavyje pirminiai, t.y. $(m, n) = 1$, tai*

$$\varphi(m \cdot n) = \varphi(m)\varphi(n). \tag{95}$$

Funkcijos, apibrėžtos natūraliųjų skaičių aibėje ir turinčios šią savybę, vadinamos multiplikatyviomis. Taigi Eulerio funkcija yra multiplikatyvi.

19.3. Kvadratiniai lyginiai

Štai tokia padėtis: didžiuliame ir sudėtingų sąryšių pilname realiųjų skaičių kūne kvadratines lygtis spręsti galime greitai, o paprastame dalybos iš skaičiaus liekanų kūne greito būdo nėra.

Kvadratines lygtis jau prieš kelis tūkstančius metų sprendė babiloniečių moksleiviai. Mūsų dienų moksleivius gelbsti diskriminantas: į jį pažiūrėjus galima pasakyti, ar lygtis turi sprendinių ir kokie jie.

O mes panagrinėkime kvadratinės lygties

$$x^2 + bx + c = 0$$

sprendimą žiede $\mathbb{Z}_n$, čia, žinoma, b, c yra sveikieji skaičiai. Kitaip tariant, panagrinėkime, su kokiomis x reikšmėmis lyginys

$$x^2 + bx + c \equiv 0 \pmod{n} \tag{96}$$

yra teisingas.

Jeigu pirminis p dalija n ir su kokia nors x reikšme (96) yra teisingas, tai su ta pačia reikšme bus teisingas ir lyginys

$$x^2 + bx + c \equiv 0 \pmod{p}. \tag{97}$$

Taigi galima pradėti nuo kvadratinių lygčių tyrinėjimo kūnuose $\mathbb{F}_p$ tikintis (ir pagrįstai!), kad, išnagrinėjus tokius atvejus, bus nesunku pereiti ir prie bendrojo.

Tarkime, kad $p > 2$ yra pirminis, ir nagrinėkime (97) lyginį. Atlikime keletą paprastų pertvarkių:

$$\begin{aligned}
&x^2 + bx + c \equiv 0 \pmod{p},\\
&4x^2 + 2(2x)b + b^2 \equiv b^2 - 4c \pmod{p},\\
&(2x + b)^2 \equiv b^2 - 4c \pmod{p},\\
&y^2 \equiv D \pmod{p},\ y = 2x + b,\ D = b^2 - 4c.
\end{aligned}$$

Taigi diskriminantas pasirodo ir čia. Jis lemia sprendinių skaičių, tačiau kaip – nelengva pasakyti.

Matome, kad svarbiausia išnagrinėti visų paprasčiausio lyginio

$$x^2 \equiv a \pmod{p}\ (a \neq 0) \tag{98}$$

atvejį. Žinome, kad kūne n-ojo laipsnio lygtis negali turėti daugiau kaip n sprendinių. Mūsų atveju padėtis tokia: jeigu sprendinių yra – tai jų lygiai du. Išties, jeigu x_0 tenkina lyginį, tai ir $x_1 = p - x_0$ tenkins.

Kada (98) lyginys turi sprendinį? Atsakyti į šį klausimą galima greitai, jeigu pasinaudosime gerais skaičių teorijos įrankiais.

111 apibrėžimas. *Tegu p yra pirminis skaičius. Legendre'o (Ležandro) simboliu vadinama funkcija*

$$\left(\frac{a}{p}\right) = \begin{cases} 0, & \text{jei } a \equiv 0 \ (mod\ p),\\ 1, & \text{jei lyginys } x^2 \equiv a \ (mod\ p),\ \text{turi sprendinių,}\\ -1, & \text{jei lyginys } x^2 \equiv a \ (mod\ p),\ \text{neturi sprendinių.} \end{cases}$$

Atrodo, kokia nauda iš tokio apibrėžimo? Ar jis nėra tik paprastas trijų atvejų ženklinimas skaičiais? Tačiau su skaičiais galima atlikti veiksmus. O veikiant visada galima ką nors pasiekti. Šiuo atveju veiksmų su Legendre'o simboliais efektyvumą lemia geros jų savybės.

133 teorema. *Su bet kokiu pirminiu skaičiumi p teisingos šios lygybės:*

$$\left(\frac{a^2}{p}\right) = 1, \qquad \left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right),$$
$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \ (mod\ p), \qquad \left(\frac{a+kp}{p}\right) = \left(\frac{a}{p}\right),$$
$$\left(\frac{-1}{p}\right) = (-1)^{(p-1)/2}, \qquad \left(\frac{2}{p}\right) = (-1)^{(p^2-1)/8}.$$

Ir dar vienas teiginys, kuris padeda skaičiuoti Legendre'o simbolius. Tai Gauso dėsnis – vienas iš pačių įstabiausių skaičių teorijos teiginių.

134 teorema. *Su bet kokiais skirtingais pirminiais skaičiais p, q teisinga lygybė*

$$\left(\frac{p}{q}\right) \cdot \left(\frac{q}{p}\right) = (-1)^{(p-1)(q-1)/4}.$$

Kelios iš suformuluotų simbolio savybių visiškai akivaizdžios, kitų esmės lengvai neįžvelgsi. Įrodymus galite susirasti skaičių teorijos vadovėliuose.

O dabar išbandykime Legendre'o simbolių skaičiavimo techniką. Tarkime, mums reikia nustatyti, ar turi sprendinių lyginys

$$x^2 \equiv 46 \ (\mathrm{mod}\ 57).$$

Skaičiuokime naudodamiesi Legendre'o simbolio savybėmis:

$$\begin{aligned}\left(\frac{46}{57}\right) &= \left(\frac{2\cdot 23}{57}\right) = \left(\frac{2}{57}\right)\cdot\left(\frac{23}{57}\right) = \left(\frac{23}{57}\right)\\ &= (-1)^{\frac{(23-1)(57-1)}{4}}\left(\frac{57}{23}\right) = -\left(\frac{57}{23}\right) = -\left(\frac{2\cdot 23+11}{23}\right)\\ &= -\left(\frac{11}{23}\right) = -(-1)^{\frac{(11-1)(23-1)}{4}}\left(\frac{23}{11}\right) = \left(\frac{1}{11}\right) = 1.\end{aligned}$$

Taigi lyginys sprendinį turi. Kaip jį galima surasti? Tiesa yra tokia: greito metodo, tinkančio visiems pirminiams, nėra! Taigi belieka perrinkti pretendentus, tikintis, kad vienas iš dviejų sprendinių ilgai nesislapstys. Tiesa, yra vienas atvejis, kai sprendinį galima rasti skaičiavimais.

135 teorema. *Tegu pirminis skaičius p, dalijant iš 4, duoda liekaną 3, t.y. $p \equiv 3\ (mod\ 4)$. Jeigu lyginys $x^2 \equiv a\ (mod\ p)$ turi sprendinių, tai vienas iš jų yra*

$$x_0 \equiv a^{\frac{p+1}{4}}\ (mod\ p).$$

Įrodymas. Kadangi lyginys turi sprendinių, tai iš Legendre'o simbolio savybių gauname

$$\left(\frac{a}{p}\right) = 1, \quad a^{\frac{p-1}{2}} \equiv \left(\frac{a}{p}\right)\ (\mathrm{mod}\ p), \quad a^{\frac{p-1}{2}} \equiv 1\ (\mathrm{mod}\ p).$$

Tada

$$x_0^2 \equiv a^{\frac{p+1}{2}} \equiv a \cdot a^{\frac{p-1}{2}} \equiv a \pmod{p}.$$

Tai beveik viskas, ką galima pasakyti apie kvadratinių lyginių sprendimą pirminiu moduliu. Greito sprendimo būdo tiesiog nėra.

O dabar panagrinėkime sudėtinio modulio atvejį. Apsiribokime atveju, kai $n = pq$, čia p, q – du skirtingi pirminiai skaičiai. Surasti lyginio

$$x^2 \equiv a \pmod{n} \tag{99}$$

sprendinį reiškia rasti tokią x reikšmę v, kad $v^2 - a$ dalytųsi iš n. Tačiau tada ir p bei q dalytų $v^2 - a$, taigi reikšmė $x = v$ būtų abiejų lyginių

$$x^2 \equiv a \pmod{p}, \quad x^2 \equiv a \pmod{q} \tag{100}$$

sprendinys. Kita vertus, jeigu rastume vieną x reikšmę, tinkančią abiem (100) lyginiams, ji būtų ir (99) sprendinys. Tačiau spręsdami (100) lyginius atskirai, vargu ar galime tikėtis, kad gausime tą patį skaičių. Gautume, pavyzdžiui, kad pirmajam lyginiui tinka visi skaičiai m, tenkinantys sąlygą $m \equiv m_1 \pmod{p}$, o antrajam – visi skaičiai, tenkinantys sąlygą $m \equiv m_2 \pmod{q}$. Taigi reikia skaičiaus, kuris tenkintų abi sąlygas! Išspręsti šį uždavinį galime pasinaudoję dar vienu labai senu, bet kriptografijai labai naudingu įrankiu – kiniškąja liekanų teorema.

136 teorema. *Tegu $n_1, n_2, \ldots, n_t$ yra tarpusavyje pirminiai skaičiai, o $m_1, m_2, \ldots, m_t$ – bet kokie sveikieji skaičiai. Tegu $n = n_1 n_2 \cdots n_t$, o sveikieji skaičiai $a_1, a_2, \ldots, a_t$ tenkina sąlygas*

$$a_i \cdot \frac{n}{n_i} \equiv 1 \ (mod\ n_i), \quad i = 1, 2, \ldots, t.$$

Sudarykime skaičių

$$M = m_1 a_1 \cdot \frac{n}{n_1} + m_2 a_2 \cdot \frac{n}{n_2} + \ldots + m_t a_t \cdot \frac{n}{n_t}.$$

Tada šis skaičius tenkina visus lyginius $M \equiv m_i \ (mod\ n_i)$, $i = 1, 2, \ldots, t$.

Įrodymas. Įsitikinkime, pavyzdžiui, kad $M \equiv m_1 \pmod{n_1}$. Kadangi visi dėmenys skaičiaus M išraiškoje, išskyrus pirmąjį, dalijasi iš n_1, tai

$$M \equiv m_1 a_1 \cdot \frac{n}{n_1} \pmod{n_1}.$$

Tačiau sandaugą $a_1 \cdot \frac{n}{n_1}$ galime pakeisti vienetu, taigi $M \equiv m_1 \pmod{n_1}$.

Žinoma, pastebėjote, kad skaičiai a_i yra skaičių $\frac{n}{n_i}$ atvirkštiniai moduliu n_i. Kaip juos rasti, jau žinome.

O dabar sugrįžkime prie lyginių (99), (100). Tarkime, pirmojo lyginio sprendiniai yra $\pm x_1$, o antrojo – $\pm x_2$. Tada suradę skaičius a, b, kad

$$aq \equiv 1 \pmod{p}, \quad bp \equiv 1 \pmod{q},$$

galėsime pagal kinų liekanų teoremą sudaryti ir (99) sprendinius

$$x = \pm x_1 aq \pm x_2 bp.$$

Taigi gauname net keturis sprendinius. Taisyklė „sprendinių nėra daugiau nei lygties laipsnis“ galioja tik kūnuose!

19.4. Natūrinių skaičių skaidymas

Skaidyti natūraliuosius skaičius pirminiais daugikliais – sunkus skaičiavimo uždavinys. Tačiau tai nereiškia, kad skaidyti kiekvieną didelį skaičių yra sunku.

Tarkime, n yra didelis natūralusis skaičius. Uždavinys – surasti jo pirminius daliklius. Pirmiausia, kas ateina į galvą – bandyti jį dalyti iš pirminių, išrikiuotų didėjimo tvarka: $p_1 < p_2 \ldots$ Jeigu n yra sudėtinis skaičius, tai $n = n_1 n_2$ ir vienas iš skaičių n_1, n_2 yra ne didesnis už $\sqrt{n}$. Taigi ieškant pirminių daliklių tokiu elementarios perrankos būdu, tikrai neteks atlikti daugiau kaip $\sqrt{n}$ perrankų. Tačiau lyginant su dydžiu $\log_2 n$, nusakančiu bitų skaičių, reikalingą n užrašyti, perrankų gali tekti atlikti labai daug. Todėl toks algoritmas nėra efektyvus.

Kai nėra visiems atvejams tinkančių efektyvių sprendimo taisyklių, kliautis atsitiktine sėkme nėra blogas sprendimas. Panagrinėkime Pollardo sugalvotą skaičių skaidymo algoritmą, kuriame sėkmė vaidina svarbų vaidmenį.

Pollardo ρ metodas

Tarkime, n yra natūralusis skaičius, o p – mums nežinomas jo pirminis daliklis. Atsitiktinai pasirinkime skaičius $s_1, s_2, \ldots, s_m$. Kai m yra pakankamai didelis skaičius, tikėtina, kad atsiras du skaičiai s_i, s_j, kad $s_i \equiv s_j \pmod{p}$, tačiau $s_i \not\equiv s_j \bmod n$. Tada skirtumas $s_i - s_j$ dalysis iš p, bet nesidalys iš n. Bendrasis didžiausiasis skirtumo ir n daliklis $n_1 = (s_i - s_j, n)$ bus nelygus vienam, taigi tokiu atveju gautume skaidinį $n = n_1 n_2$ ir toliau galėtume ieškoti mažesnių skaičių n_1, n_2 pirminių daliklių. Tačiau ar toks būdas nėra toks pat neefektyvus kaip ir tiesioginė perranka? Iš tikrųjų, toks jis ir būtų, jeigu skaičiuotume visus bendruosius didžiausiuosius daliklius $(s_i - s_j, n)$. Viena gera idėja paverčia jį maždaug dukart spartesniu už tiesioginę daliklių perranką.

Visų pirma tarkime, kad atsitiktiniams skaičiams parinkti sudarėme tam tikrą funkciją $f : \mathbb{Z}_n \to \mathbb{Z}_n$, kurios reikšmės išsibarsčiusios aibėje $\mathbb{Z}_n$ gana „atsitiktinai“. Be to, tarkime, ši funkcija turi savybę: bet kokiam natūraliajam a iš $u \equiv v \pmod{a}$ seka $f(u) \equiv f(v) \pmod{a}$. Šią savybę turi, pavyzdžiui, daugianariai. Daugelis aukštesnio laipsnio daugianarių f neblogai tinka šiam metodui. Atsitiktinai pasirinkę $s_0 \in \mathbb{Z}_n$, kitas reikšmes skaičiuokime taip:

$$s_1 \equiv f(s_0) \pmod{n},\ s_2 \equiv f(s_1) \pmod{n},\ \ldots,\ s_m \equiv f(s_{m-1}) \pmod{n}.$$

Tarkime su skaičiais $0 \le i < j$ teisingas sąryšis $s_i \equiv s_j \pmod p$, čia p yra mums nežinomas pirminis n daliklis. Pažymėkime $k = j - i$, tada $j = i + k$, $s_i \equiv s_{i+k} \pmod n$. Panagrinėkime tokią teisingų lyginių lentelę:

$$\begin{array}{rcll} f(s_i) & \equiv & f(s_{i+k}) \pmod p & \text{arba} \quad s_{i+1} \equiv s_{i+1+k} \pmod p \\ f(s_{i+1}) & \equiv & f(s_{i+1+k}) \pmod p & \text{arba} \quad s_{i+2} \equiv s_{i+2+k} \pmod p \\ & \cdots & & \cdots \qquad \cdots \\ f(s_{i+k-1}) & \equiv & f(s_{i+k+k-1}) \pmod p & \text{arba} \quad s_{i+k} \equiv s_{i+2k} \pmod p \end{array}$$

Tačiau iš lyginių $s_i \equiv s_{i+k} \pmod p, s_{i+k} \equiv s_{i+2k} \pmod p$ gauname, kad $s_i \equiv s_{i+2k} \pmod p$. Samprotaudami toliau, gautume, kad lyginys $s_i \equiv s_{i+tk} \pmod p$ teisingas su bet kokiu natūraliuoju t. Jeigu yra toks t, kad $tk = i$, tai tada turime teisingą lyginį $s_i \equiv s_{2i} \pmod p$. Taigi daliklio paiešką galime atlikti taip: skaičiuojame s_i ir s_{2i} ir $d = (s_{2i} - s_i, n)$, kol gauname $d > 1$.

Skaičiavimus organizuoti patogu taip:

1. parenkame funkciją $f : \mathbb{Z}_n \to \mathbb{Z}_n$ ir atsitiktinį $r \in \mathbb{Z}_n$;
2. priskiriame reikšmes $a = r, b = r$;
3. skaičiuojame $a = f(a), b = f(f(b))$;
4. randame $d = (b - a, n)$; jei $d \neq 0, n$ – rastas netrivialus n daliklis; jei $d = 1$, kartojame 3) žingsnį.

Trečiajame žingsnyje skaičiuojami dydžiai s_i ir s_{2i}.

Išbandykime šį metodą, pavyzdžiui, su $n = 34571317791$. Pasirinkta funkcija $f(x) \equiv x^2 + 732 \pmod n$. Kelių skaičiavimo žingsnių rezultatai pateikti lentelėje

a	b	$(b - a, n)$
142107	20194400180	1
20194400180	21542042492	3
17674479849	10362019826	1
21542042492	16189403312	141

Jau antrajame žingsnyje suradome daliklį 3. Žinoma, kad skaičius dalijasi iš 3, galėjome nustatyti iš karto, naudodamiesi dalybos iš 3 požymiu: skaitmenų suma dalijasi iš 3. Taigi

$$34571317791 = 141 \cdot 34571317791 = 3 \cdot 47 \cdot 34571317791.$$

Jeigu bandytume skaidyti skaičių 34571317791 – nieko nelaimėtume. Šis skaičius yra pirminis.

Pollardo $p-1$ metodas

Pollardas sugalvojo dar vieną skaičių skaidymo metodą, kuriuo naudojantis kartais pirminį daliklį galima rasti labai greitai.

112 apibrėžimas. *Tegu m, B yra du natūriniai skaičiai. Skaičių m vadinsime B-glodžiu, jeigu visi jo pirminiai dalikliai yra ne didesni už B.*

Pollardo $p-1$ metodas skaičiui n skaidyti yra efektyvus tada, kai n turi pirminį daliklį, kuriam $p-1$ yra B-glodus su nedidele B reikšme. Kitaip tariant – kai bent vienam pirminiam dalikliui p visi skaičiaus $p-1$ pirminiai dalikliai yra maži. Todėl šis Pollardo metodas ir vadinamas $p-1$ metodu.

Tarkime, n yra skaičius, kurį reikia išskaidyti, o p – mums nežinomas pirminis daliklis. Mažoji Fermat teorema teigia: jei skaičius a nesidalija iš p, tai $a^{p-1} \equiv 1 \pmod{p}$. Tačiau tada ir su bet kokiu natūraliuoju skaičiumi t lygybė $a^{t(p-1)} \equiv 1 \pmod{p}$ taip pat teisinga. Metodo esmė yra tokia: darant prielaidą, kad $p-1$ sudarytas iš mažų pirminių daliklių, galima pabandyti sukonstruoti kokį nors $p-1$ kartotinį $N = t(p-1)$. Tada būtų teisinga lygybė $a^N \equiv 1 \pmod{p}$, t.y. $a^N - 1$ dalytųsi iš p. Kadangi ir n dalijasi iš p, tai galėtume Euklido algoritmu suskaičiuoti $(a^N - 1, n) = d$ ir rasti netrivialų n daliklį. Žinoma, skaičius $a^N - 1$ gali būti tiesiog milžiniškas, tačiau jo reikšmė skaičiuojant bendrąjį didžiausiąjį daliklį visai nereikalinga, pakanka laipsnius skaičiuoti moduliu n.

Parinkti skaičių N galima įvairiais būdais. Pavyzdžiui, galima pasirinkus B apibrėžti $N = BMK(2, 3, \ldots, B)$, čia $BMK(a_1, a_2, \ldots, a_k)$ žymi bendrąjį mažiausią skaičių kartotinį. Šio didelio skaičiaus reikšmę irgi nebūtina skaičiuoti iš anksto, pakanka surasti visus pirminius skaičius q, kurie yra mažesni už B ir reikšmę a^N apskaičiuoti palaipsniui. Štai tokio algoritmo aprašymas:

1. pasirenkamas skaičius $a_0 = a$ ir surandami visi pirminiai skaičiai $q_1, q_2, \ldots, q_h$, ne didesni už B;

2. atliekama h algoritmo žingsnių, i-ajame žingsnyje skaičiuojama

$$r_i = \left[\frac{\log B}{\log q_i}\right], \quad a_i \equiv a_{i-1}^{r_i} \pmod{n};$$

3. skaičiuojama $d = (a_i - 1, n)$; jeigu $d \neq 1, n$ – skaičiaus n daliklis rastas.

Pabandykime išskaidyti šiuo metodu gana didelį natūralųjį skaičių

$$n = 406222941815702227.$$

Galima iš pradžių pasirinkti nelabai didelį B, o tada, jeigu nepasiseka, skaičiuoti su didesniu. Imkime, pavyzdžiui, $B = 20$. Šis skaičius yra nedidelis,

todėl galime visų skaičių iki B bendrąjį mažiausiąjį kartotinį suskaičiuoti ir tiesiogiai:

$$N = 2^4 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19 = 232792560.$$

Imkime, pavyzdžiui, $a = 2$; skaičiuodami gausime:

$$\begin{aligned} &2^{232792560} \equiv 327811457795757939 \pmod{n}, \\ &(327811457795757939 - 1, n) = 8893, \\ &n = 8893 \cdot 45678954437839. \end{aligned}$$

Žinoma, tokia greita sėkmė nusišypsos retai. Šįkart pasisekė todėl, kad iš anksto buvo parinktas geras skaičius.

Taigi natūraliųjų skaičių skaidymo uždavinys ne visada yra sudėtingas. Jeigu reikalingas sudėtinis skaičius, kurį būtų sunku išskaidyti dauginamaisiais, jo pirminiai dalikliai, sumažinti vienetu, neturi būti itin glodūs.

Efektyviausi šiuo metu žinomi skaičių skaidymo metodai naudoja skaičių laukų ir elipsinių kreivių savybes. Jų esmę paaiškinti būtų sudėtingiau nei nagrinėtųjų paprastų, bet dažnai greitai duodančių rezultatą Pollardo metodų.

19.5. Generuojantys elementai ir diskretieji logaritmai

Skaičiuodami su realiaisiais skaičiais, dažnai naudojamės apytikslėmis reikšmėmis. Tačiau baigtinėse grupėse apytikslis skaičiavimas tiesiog neturi prasmės, o paprastai formuluojami uždaviniai neturi paprastų ir efektyvių sprendimų. Logaritmo skaičiavimas baigtinėse grupėse – tokio uždavinio pavyzdys.

Tegu G yra baigtinė grupė, joje apibrėžta elementų daugyba. Jeigu elementus $a, b \in G$ sieja lygybė

$$a^x = b,$$

čia x yra sveikasis skaičius, tai jį pagal realiųjų skaičių pavyzdį natūralu pavadinti elemento b logaritmu pagrindu a. Tačiau verta tokį apibrėžimą patikslinti: būtų gerai, kad logaritmas būtų apibrėžtas vienareikšmiškai ir nereiktų papildomai nagrinėti, ar logaritmas duotuoju pagrindu egzistuoja, ar ne.

113 apibrėžimas. *Tegu G yra baigtinė ciklinė grupė, o $g \in G$ jos generuojantis elementas, t.y. toks elementas, kad*

$$G = \{g^0, g^1, \ldots, g^{N-1}\},$$

čia $N = |G|$ yra grupės eilė, t.y. jos elementų skaičius. Elemento y diskrečiuoju logaritmu pagrindu g vadinsime aibės $\mathbb{Z}_{N-1}$ skaičių x, su kuriuo $y = g^x$. Logaritmą žymėsime $x = \log_g y$.

Taigi diskrečiuosius logaritmus apibrėžėme tik ciklinėse grupėse, jų pagrindais gali būti tik generuojantys elementai. Ne visos baigtinės grupės yra ciklinės, tačiau ir ciklinės grupės ne retenybė. Pavyzdžiui, bet kokio baigtinio kūno $\mathbb{F}_{p^m}$ nenulinių elementų aibė $\mathbb{F}_{p^m}^*$ daugybos atžvilgiu sudaro ciklinę grupę. Dažniausiai kriptografijos reikmėms naudosime grupę $\mathbb{F}_p^* = \{1, 2, \ldots, p-1\}$.

Diskrečiojo logaritmo pagrindas turi būti generuojantis elementas. Kaip surasti bent vieną? Kadangi tų generuojančių elementų yra pakankamai daug, tai ieškojimas pasikliaujant sėkme nėra bloga išeitis. O nustatyti, ar pasirinktas elementas yra generuojantis, galime naudodamiesi tokiu kriterijumi:
tegu N yra ciklinės grupės G eilė, $g \in G$ jos elementas; jei $g^d \neq 1$ su visais netrivialiais N dalikliais d, tai g yra generuojantis elementas.

Panagrinėkime, pavyzdžiui, atvejį $G = \mathbb{F}_7, g = 2$. Kadangi grupės eilė $N = 6$, tai pakanka patikrinti, ar nei vienas iš laipsnių $2^2, 2^3$ nelygus vienetui. Kadangi $2^3 \equiv 1 \pmod 7$, tai $g = 2$ nėra generuojantis elementas. Tačiau $h = 3$ tenkina šį kriterijų, taigi yra generuojantis elementas.

Diskretieji logaritmai turi panašias savybes kaip ir logaritmai realiųjų skaičių aibėje. Tačiau yra vienas esminis skirtumas: apytikslis diskrečiųjų logaritmų skaičiavimas neturi prasmės, o efektyvių būdų surasti tikslias reikšmes kol kas niekas nesugalvojo.

137 teorema. *Tegu G yra ciklinė grupė, N – jos eilė, o g – generuojantis elementas. Tada*

1. *su visais $x, y \in G$ teisinga lygybė $\log_g(x \cdot y) \equiv \log_g x + \log_g y \ (mod\ N)$;*
2. *su visais $x \in G$ ir su visais sveikaisiais k teisinga lygybė $\log_g x^k \equiv k \log_g x \ (mod\ N)$.*

Teiginiai beveik akivaizdūs, panagrinėkime pirmąjį. Tegu $u = \log_g x, v = \log_g y, w = \log_g(x \cdot y)$, tada

$$x = g^u,\ y = g^v,\ x \cdot y = g^u \cdot g^v = g^{u+v} = g^w.$$

Tačiau iš laipsnių $g^a = g^b$ lygybės išplaukia $a \equiv b \pmod N$, taigi $u + v \equiv w \pmod N$.

Efektyvių metodų diskretiesiems logaritmams skaičiuoti nėra. Visada galima ieškoti diskrečiojo logaritmo reikšmės perrankos būdu: tikrinti galimas reikšmes, kol pasitaikys tikroji. Yra ir šiek tiek išradingesnių metodų, kurie, nors ir negarantuoja greitos sėkmės, kartais padeda rasti diskretųjį logaritmą gana greitai.

Shankso metodas

Šiuo metodu ieškoma elementų iš $\mathbb{F}_p^*$ diskrečiųjų logaritmų pagrindu g, čia p – pirminis skaičius. Bet kurio elemento $y \in \mathbb{F}_p^*$ diskretusis logaritmas

x yra skaičius, ne didesnis už $p-1$. Pažymėkime $m = [\sqrt{p-1}]+1$; padaliję x iš m su liekana, gautume

$$x = im + j, \quad 0 \le i < m,\ 0 \le j < m. \tag{101}$$

Taigi

$$g^{mi+j} = y, \quad g^{mi} = yg^{-j}.$$

Šia lygybe ir remiasi algoritmo idėja: galima iš anksto apskaičiuoti reikšmes ir susidaryti porų $\langle i, g^{mi}\rangle$, $i = 0, 1, \ldots, m-1$, lentelę; tada sudaryti kitą lentelę iš porų $\langle j, yg^{-j}\rangle$ ir ieškoti abiejose lentelėse įrašų su vienodomis antrosiomis komponentėmis, t.y. lygybės $g^{mi} = yg^{-j}$. Suradę atitinkamus i ir j, pagal (101) galime apskaičiuoti diskrečiojo logaritmo reikšmę. Blogiausiu atveju reiktų maždaug $\sqrt{p}$ skaičiavimų; ieškant logaritmo tiesioginės perrankos būdu, didžiausias tikrinimų skaičius, kurio gali prireikti, yra maždaug p. Šį metodą dar kartais vadina mažylio-milžino žingsnių metodu (baby-step-giant-step method). Mažais žingsneliais sudarome lenteles, randame i, j ir, žengę didelį žingsnį, apskaičiuojame diskrečiojo logaritmo reikšmę.

Pavyzdys. Apskaičiuokime skaičių $y_1 = 13, y_2 = 17, y_1, y_2 \in \mathbb{F}_{23}^*$ diskrečiuosius logaritmus pagrindu $g = 5$. Mūsų atveju $m = 5$; taigi lentelėse bus tik po penkias poras.

$i, j =$	g^{mi}	$y_1 g^{-j}$	$y_2 g^{-j}$
0	1	13	17
1	20	21	8
2	9	18	20
3	19	22	4
4	12	9	10

Iš lentelės matome

$$g^{2m} = y_1 g^{-4}, \quad g^{1m} = y_2 g^{-2},$$

taigi

$$\log_g y_1 = 2m + 4 = 14, \quad \log_g y_2 = m + 2 = 7.$$

Pollardas, kurio natūraliųjų skaičių skaidymo logaritmus nagrinėjome ankstesniame skyriuje, sugalvojo ir du diskrečiųjų logaritmų skaičiavimo metodus. Vienas vadinamas ρ-metodu, kitas – λ-metodu. Panagrinėsime pirmąjį, kuris primena ρ metodą skaičiams skaidyti.

Pollardo ρ-metodas

Reikia surasti skaičiaus $y \in \mathbb{F}_p^*$ diskretųjį logaritmą pagrindu g, čia g – generuojantis elementas. Metodo esmė tokia: vienas po kito skaičiuojami

elementai $x_{i+1} = f(x_i), x_i \in \mathbb{F}_p^*$, kad su atitinkamais a_i, b_i būtų teisinga lygybė

$$x_i \equiv y^{a_i} g^{b_i} \pmod{p},$$

čia skaičiai a_i, b_i irgi skaičiuojami vienas po kito. Jeigu gautume, kad su kokiais nors i, j teisinga lygybė

$$x_i \equiv x_j \pmod{p}, \quad y^{a_i} g^{b_i} \equiv y^{a_j} g^{b_j} \pmod{p}, \quad y^{a_i - a_j} \equiv g^{b_j - b_i} \pmod{p},$$

tai, pažymėję $x = \log_g y$, gautume lyginį

$$(a_i - a_j)x \equiv b_j - b_i \pmod{p}$$

reikšmei x rasti. Kaip ir naudojant panašų skaičių skaidymo metodą, daugelio tikrinimų $x_i \equiv x_j \pmod{p}$ galima išvengti skaičiuojant iš karto x_k ir x_{2k} bei tikintis, kad jie sutaps:

$$x_{k+1} = f(x_k), \quad x_{2k+2} = f(f(x_k)).$$

Jeigu $x_i = x_{2i}$, tai gautume tokią lygybę:

$$(a_i - a_{2i})x \equiv b_{2i} - b_i \pmod{p}.$$

Belieka apibrėžti sekų x_i, a_i, b_i skaičiavimo taisykles. Štai jos: $x_0 = 1$, $a_0 = b_0 = 0$,

$$x_{i+1} = \begin{cases} yx_i \pmod{p}, & \text{jei } 0 < x_i < p/3 \\ x_i^2 \pmod{p}, & \text{jei } p/3 < x_i < 2p/3 \\ gx_i \pmod{p}, & \text{jei } 2p/3 < x_i < p; \end{cases}$$

$$a_{i+1} = \begin{cases} 1 + a_i \pmod{p-1}, & \text{jei } 0 < x_i < p/3 \\ 2a_i \pmod{p-1}, & \text{jei } p/3 < x_i < 2p/3 \\ a_i \pmod{p-1}, & \text{jei } 2p/3 < x_i < p; \end{cases}$$

$$b_{i+1} = \begin{cases} b_i \pmod{p-1}, & \text{jei } 0 < x_i < p/3 \\ 2b_i \pmod{p-1}, & \text{jei } p/3 < x_i < 2p/3 \\ 1 + b_i \pmod{p-1}, & \text{jei } 2p/3 < x_i < p. \end{cases}$$

Lyginius $x_i \equiv y^{a_i} g^{b_i} \pmod{p}$ patikrinti nesudėtinga.

Pavyzdys. Tegu $p = 23$, tada $g = 5$ yra generuojantis elementas. Pabandykime Pollardo ρ-metodu surasti $x = \log_g y, y = 18$. Štai skaičiavimų

eiga:

i	x_i	a_i	b_i	x_{2i}	a_{2i}	b_{2i}
0	1	0	0	1	0	0
1	18	1	0	21	1	1
2	21	1	1	13	1	2
3	13	1	2	21	4	9
4	8	2	4	8	8	20

Taigi jau ketvirtajame žingsnyje pasisekė gauti lygybę $x_4 = x_8$. Dabar galime ieškoti ir logaritmo:

$$y^2g^4 \equiv y^8g^{20} \pmod{23}, \quad y^6 \equiv g^{-16} \pmod{23}, \quad 6x \equiv -16 \pmod{22},$$
$$3x \equiv -8 \pmod{11}, \quad 3x \equiv 3 \pmod{11}, \quad x \equiv 1 \pmod{11}.$$

Gauname du lyginio $6x \equiv -16 \pmod{22}$ sprendinius: $x = 1$ ir $x = 1 + 11 = 12$. Antroji reikšmė ir yra diskretusis logaritmas: $\log_5 18 = 12$.

Tačiau ne visada Pollardo metodo procesas baigiasi taip sėkmingai. Pavyzdžiui, skaičiuodami analogiškai su $y = 14$, gautume:

i	x_i	a_i	b_i	x_{2i}	a_{2i}	b_{2i}
0	1	0	0	1	0	0
1	14	1	0	12	2	0
1	12	2	0	6	4	0
2	6	4	0	15	5	0
3	15	5	0	15	5	0

Taigi gavome, kad sutampa ne tik $x_i = x_{2i}$, bet ir $a_i = a_{2i}, b_i = b_{2i}$. Iš to mums nėra jokios naudos. Tačiau galime pakartoti Pollardo algoritmą su kita pradine x_0 reikšme. Pavyzdžiui, imdami $x_0 = 3$, jau antrajame žingsnyje gauname:

$$x_2 = x_4, \quad a_2 = b_2 = 1, \quad a_4 = b_4 = 2,$$

ir logaritmą randame iš lyginio $yg \equiv y^2g^2 \pmod{23}$, arba $x \equiv -1 \pmod{22}$, $x = 21$.

20 Viešojo rakto kriptosistemos

Jeigu nėra saugaus būdo perduoti raktams, simetrinės kriptosistemos naudojimas ryšiui apsaugoti tiesiog neįmanomas. Kai didžiulių informacijos srautų judėjimas kompiuteriniais tinklais tapo kasdienybe, saugius ryšio kanalus

teko tiesiog užmiršti. Tiesa, diplomatinis bagažas egzistuoja ir dabar, tačiau dauguma iš mūsų nesame nei diplomatai, nei ruošiamės jais tapti. Taigi kompiuterių amžius iškėlė kriptografijai kone neįmanomą iššūkį: reikia šifruoti, bet saugiai perduoti raktų neįmanoma!

Galime prisiminti, kaip buvo atsakyta į klausimą: ar galima palaikyti labai patikimą ryšį nepatikimu kanalu, turint tik ribotus išteklius? C. Shannonas įrodė, kad tai įmanoma. Teigiamai atsakyta ir į klausimą apie šifravimą, kuriam nebūtini saugūs raktų perdavimo būdai.

Naująsias kompiuterių amžiaus kriptografijos kryptis 1976 metais pirmieji nurodė W. Diffie ir M. Hellman[19]. Idėja labai paprasta: kadangi neįmanoma sukurti tokios kriptosistemos, kurios šifravimui ir dešifravimui skirti raktai būtų nesusiję, tai reikia padaryti bent taip, kad, norint iš dešifravimo rakto nustatyti šifravimui skirtą raktą, tektų sugaišti tiek laiko, kad žmonių pasaulyje jis prilygtų kone amžinybei. Nereikia netgi amžinybės, pakanka, kad tokiam darbui prireiktų laiko, per kurį ir naudojamas raktas būtų pakeistas, ir užšifruotas pranešimas taptų nebesvarbus. Tada šifravimui skirtą raktą galima būtų paskelbti viešai, nebijant, kad iš jo bus greitai nustatytas dešifravimo raktas. Todėl tokia ideologija pagrįstos kriptosistemos ir vadinamos viešojo rakto kriptosistemomis.

Štai tokio viešo, bet saugaus paskelbimo pavyzdys. Tarkime, sugalvojau kriptosistemą, kurioje šifruojama naudojant raktą

$$\begin{aligned} k_e = \quad & 11739085287969531650666649599035831993898213898740 79 \\ & 90187848339927925066408945441063107863029870718402 3, \end{aligned}$$

o dešifravimo raktas – mažesnysis skaičiaus k_e pirminis daliklis. Teks gerokai pasidarbuoti jūsų programoms, kol nustatysite, kad

$$k_d = 10000000000000000000000000000000000000000000000151.$$

Neilgai trukus po Diffie ir Hellmano idėjų paskelbimo atsirado ir pirmosios viešojo rakto kriptosistemos. Jas sukūrė R. Merkle ir M. Hellmanas, o taip pat trijų autorių grupė: R. Rivestas, A. Shamiras ir L. Adlemanas. Jie laikomi viešojo rakto kriptografijos pionieriais. Tačiau būti laikomam pirmuoju – tai dar ne juo būti. Visai neseniai paaiškėjo, kad viešojo rakto kriptosistemų principai buvo sukurti maždaug dešimtmečiu anksčiau. Jų autoriai – Jungtinės Karalystės vyriausybinės organizacijos darbuotojai Jamesas Ellisas, Cliffordas Cocksas ir Malcolmas Williamsonas. Tačiau jų darbai buvo įslaptinti, todėl jie galėjo tik iš šalies stebėti, kaip laisvi akademinio sluoksnio žmonės plėtoja naujųjų laikų kriptografiją.

[19] W. Diffie, M. Hellman. New Directions in Cryptography. IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, p. 644–654.

20.1. Kuprinės ir šifrai

R. Merkle ir M. Hellmanas sukūrė viešojo rakto kriptosistemą, kuriai prigijo „kuprinės“ vardas[20]*. Ją verta panagrinėti dėl kelių priežasčių: viena vertus tai pirmoji viešojo rakto kriptosistema, antra – ją labai paprasta paaiškinti, trečia – tai kriptosistemos, kuri buvo įveikta pasitelkus ne didžiulius skaičiavimo išteklius, bet matematines idėjas, pavyzdys.*

„Kuprinės“ uždavinys formuluojamas taip:
ID: *natūraliųjų skaičių seka* $W = \{w_1, w_2, \ldots, w_n\}$ *ir skaičius* v.
U: *rasti skaičius* $x_i \in \{0, 1\}$, *kad*

$$v = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n,$$

arba nustatyti, kad tokių skaičių nėra.

Ši problema yra **NP** pilna, taigi kai įvesties duomenų (svorių) yra daug, ją spręsti sudėtinga. Tačiau kai „kuprinės“ svoriai w_i turi specialių savybių, „kuprinės“ uždavinys irgi gali būti greitai sprendžiamas.

114 apibrėžimas. *Sakysime, kad skaičiai* $w_1, w_2, \ldots, w_n$ *sudaro sparčiai didėjančią seką, jei visiems* $i > 1$ *teisinga nelygybė*

$$w_1 + w_2 + \ldots + w_{i-1} < w_i.$$

138 teorema. *Jei „kuprinės“ svoriai* $W = \{w_1, w_2, \ldots, w_n\}$ *sudaro sparčiai didėjančią seką, tai „kuprinės“ uždavinys sprendžiamas polinominiu algoritmu. Jeigu skaičių* v *galima išreikšti sistemos* W *svoriais, tai tokia išraiška yra vienintelė.*

Įrodymas. Tegu v – natūralusis skaičius. Ieškosime jo išraiškos

$$v = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n, \quad x_i \in \{0, 1\}.$$

Pirmiausia reikia patikrinti, ar $v \geq w_n$. Jeigu ši nelygybė teisinga, tai svorį w_n būtinai teks įtraukti į sumą, t. y. $x_n = 1$. Jeigu neteisinga, tai $x_n = 0$. Tada reikia lyginti skaičių $v - x_n w_n$ su w_{n-1} ir nustatyti x_{n-1} reikšmę. Matome, kad skaičių x_j reikšmės yra nustatomos vienareikšmiškai; taigi jei išraiška egzistuoja, tai ji vienintelė.

Uždavinio sprendimo algoritmą galima užrašyti taip:

1. $w := v, j := n$;
2. jei $w \geq w_j$, tai $x_j = 1$; jei $w < w_j$, tai $x_j = 0$; $w := w - x_j w_j$; $j := j - 1$;
3. jei $w = 0$, tai išraiška rasta; jei $w \neq 0, j = 0$, išraiška neegzistuoja; kitais atvejais reikia kartoti 2 žingsnį.

[20] R. C. Merkle, M. Hellman. Hiding Information and Signatures in Trapdoor Knapsacks. IEEE Transactions on Information Theory, v. 24, n. 5, 1978, p. 525–530.

Skaičiavimą sudaro ne daugiau kaip n žingsnių. Galime tarti, kad vienam žingsniui atlikti reikia $O(|v|)$ operacijų su bitais, čia $|v|$ žymi v užrašymui reikalingų bitų kiekį. Tada bendras operacijų skaičius bus

$$O(n|v|) = O((|w_1| + \ldots + |w_n|)|v|).$$

Taigi algoritmas yra polinominis.

Turint svorių sistemą $W = \{w_1, w_2, \ldots, w_n\}$, galima taip šifruoti nulių-vienetų blokus:

$$x_1 x_2 \ldots x_n \to c = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n.$$

Tačiau tenka pasukti galvą dėl dviejų dalykų. Kaip parinkti svorius, kad dviem skirtingiems blokams visada gautume du skirtingus skaičius c? Kaip dešifruoti c, t. y. kaip spręsti „kuprinės“ uždavinį, kad jis būtų sunkus kriptoanalitikui ir lengvas teisėtam šifro gavėjui?

Jei naudosime sparčiai augančią svorių sistemą, tiek šifravimo, tiek dešifravimo veiksmai bus atliekami sparčiai, tačiau ir kriptoanalitikas, ir teisėtas gavėjas turės vienodas galimybes.

Vieną iš sprendimo būdų 1976 metais pasiūlė Merkle ir Hellmanas.

Tegu $W = \{w_1, w_2, \ldots, w_n\}$ yra sparčiai didėjanti svorių sistema, p – skaičius, didesnis už visų svorių sumą (nebūtinai pirminis):

$$p > w_1 + w_2 + \ldots + w_n, \tag{102}$$

s, t – du tarpusavyje pirminiai su p skaičiai, tenkinantys sąlygą $st \equiv 1 \pmod{p}$. Vieną iš šių skaičių galime parinkti laisvai, o kitą surasti pasinaudoję Euklido algoritmu.

Algio privatųjį (slaptąjį) raktą sudaro svorių sistema W ir skaičius s, taigi $K_p =< W, s >$. Viešasis raktas bus svorių sistema

$$V = \{v_1, v_2, \ldots, v_n\}, \quad v_i \equiv w_i t \pmod{p}, \quad K_v = \langle V \rangle.$$

Ši svorių sistema nebėra sparčiai didėjanti, taigi galime tikėtis, kad paprasto algoritmo „kuprinės“ uždaviniui spręsti nėra. Pranešimas $M = m_1 m_2 \ldots m_n \in \{0,1\}^n$ bus šifruojamas taip:

$$C = e(M|K_v) = m_1 v_1 + m_2 v_2 + \ldots + m_n v_n.$$

Taigi šifras yra skaičius, ne didesnis už $n(p-1)$.

Kaip A gali dešifruoti C? Visų pirma jis turi suskaičiuoti

$$\begin{aligned} C_1 \equiv Cs \quad &\equiv m_1 s v_1 + m_2 s v_2 + \ldots + m_n s v_n \\ &\equiv m_1 w_1 + m_2 w_2 + \ldots + m_n w_n \pmod{p}. \end{aligned}$$

(102) sąlyga garantuoja, kad skirtingiems pranešimams M skaičiai C_1 irgi bus skirtingi. Kad surastų pranešimą M, Algis turi spręsti „kuprinės“

uždavinį su sparčiai didėjančių svorių sistema W ir skaičiumi C_1. Jau žinome, kad toks uždavinys sprendžiamas polinominiu algoritmu.

Deja, ši paprasta ir elegantiška kriptosistema turi didelį trūkumą. Pasirodė, kad iš viešojo rakto svorių sistemos V, pasinaudojus tam tikromis matematinėmis idėjomis įmanoma greitai rasti privatųjį raktą, t. y. sparčiai didėjančią svorių sistemą. Taigi ši kriptosistema nėra saugi. Buvo sugalvota įvairių kitų „kuprinės“ kriptosistemos variantų. Tačiau ir jie vienas po kito buvo įveikti. „Kuprinės“ tipo kriptosistemų ir jų analizės apžvalga yra pateikta straipsnyje[21].

„Kuprinės“ kriptosistema
Pranešimų aibė $\mathcal{M} = \{0,1\}^n$, šifrų aibė $\mathcal{C} \subset \mathbb{N}$. **Privatusis raktas**: $K_p = \langle W, s\rangle$, čia $W = \langle w_1, w_2, \ldots, w_n\rangle$ – sparčiai didėjanti svorių sistema; $w_1 + w_2 + \ldots + w_n < p, \quad (s,p) = 1$. **Viešasis raktas:** $K_v = \langle v_1, v_2, \ldots, v_n\rangle$, $v_i \equiv w_i s^{-1} \pmod p$. **Šifravimas:** $C = e(m_1 m_2 \ldots m_n \mid K_v) = m_1 v_1 + \ldots + m_n v_n$. **Dešifravimas:** $C_1 \equiv Cs \pmod p$, $C_1 = m_1 w_1 + \ldots + m_n w_n$, $m_1 \ldots m_n = d(C \mid K_p)$.

20.2. RSA

Ši trijų autorių – R. Rivesto, A. Shamiro ir L. Adlemano sukurta viešojo rakto kriptosistema yra pati populiariausia. Ja naudojamasi jau daugiau kaip dvidešimt metų. Dvidešimt metų trunkantys jos tyrinėjimai neatskleidė esminių saugumo spragų.

Šioje kriptosistemoje ir pranešimai, ir jų šifrai yra tos pačios aibės skaičiai.

Raktų parinkimas. Ryšio dalyvis A pasirenka du pirminius skaičius p, q ir sudaugina juos: $n = p \cdot q$. Dar reikia pasirinkti natūralųjį skaičių $e = e_A$, kad jis būtų tarpusavyje pirminis su $\phi(n) = (p-1)(q-1)$, t. y. $(e, (p-1)(q-1)) = 1$. Naudojantis Euklido algoritmu, reikia surasti skaičių $d = d_A$, kad būtų

$$e \cdot d \equiv 1 \pmod{(p-1)(q-1)}.$$

Dabar jau galima sudaryti raktus: viešąjį $K_v = \langle e, n\rangle$ ir privatųjį $K_p = \langle d\rangle$. Skaičius p, q geriausia iš viso ištrinti. Jų nebeprireiks, tačiau jeigu jie taptų žinomi kriptoanalitikui Z, jis surastų ir privatųjį raktą.

[21] Andrew M. Odlyzko. The Rise and Fall of Knapsack Cryptosystems. In: Cryptology and Computational Number Theory, Proceedings of Symposia in Applied Mathematics, vol. 42. American Mathematics Society, Providence, RI, 1990, p. 75–88. http://www.research.att.com/ amo/doc/arch/knapsack.survey.ps

Šifravimas. Pranešimai, kuriuos bus galima siųsti A, yra aibės $\mathcal{M} = \{1, 2, \ldots, n\}$ skaičiai. Šifravimas apibrėžiamas lygybe:

$$C = e(M|K_v) \equiv M^e \pmod{n}.$$

Dešifravimas. Dešifravimo algoritmas visiškai toks pat kaip ir šifravimo:

$$d(C|K_p) \equiv C^d \pmod{n}.$$

Įsitikinkime, kad dešifruojant visada gaunama $C^d \equiv M \pmod{n}$. Kadangi $n = pq$, tai pakanka įsitikinti, kad $C^d \equiv M \pmod{p}, C^d \equiv M \pmod{q}$.

Jei $M \equiv 0 \pmod{p}$, tai akivaizdu, kad $C \equiv 0 \pmod{p}$ ir

$$C^d \equiv 0 \equiv M \pmod{p}.$$

Tegu $(M, p) = 1$. Tada

$$C \equiv M^{ed} \equiv M^{1+t(p-1)(q-1)} \equiv M(M^{p-1})^{q-1} \pmod{p}.$$

Tačiau pagal Fermat teoremą $M^{p-1} \equiv 1 \pmod{p}$, taigi $C^d \equiv M \pmod{p}$. Aišku, kad analogiški samprotavimai tinka ir skaičiui q.

Kriptosistemos saugumas remiasi tuo, kad Z, norėdamas surasti privatųjį raktą d, turi spręsti lyginį $e \cdot d \equiv 1 \pmod{(p-1)(q-1)}$. Tačiau tam reikia žinoti $(p-1)(q-1)$. Kad šį skaičių surastų, Z turi išskaidyti n pirminiais daugikliais. Tai yra sunkus skaičiavimo uždavinys. Taigi kol efektyvus didelių pirminių skaičių skaidymo pirminiais daugikliais algoritmas nėra žinomas, tol RSA kriptosistema yra saugi. Gali būti, kad RSA galima įveikti ir be greito natūrinių skaičių skaidymo algoritmo, tačiau to niekas kol kas nežino.

RSA kriptosistema
Pranešimų ir šifrų aibės $\mathcal{M} = \mathcal{C} = \mathbb{Z}_n$, $n = pq$, skaičiai p, q yra pirminiai.
Privatusis raktas: $K_p = \langle d \rangle$, $(d, \varphi(n)) = 1, \varphi(n) = (p-1)(q-1)$.
Viešasis raktas: $K_v = \langle n, e \rangle$, $ed \equiv 1 \pmod{\varphi(n)}$.
Šifravimas: $C = e(M\|K_v) \equiv M^e \pmod{n}$.
Dešifravimas: $M = d(C\|K_p) \equiv C^d \pmod{n}$.

Norint RSA kriptosistema šifruoti, pavyzdžiui, lietuviškus tekstus, reikia susitarti, kaip jie bus verčiami skaičiais. Galima, pavyzdžiui, raides interpretuoti kaip skaičiavimo sistemos su pagrindu $N = 33$ skaitmenis, o tarpą tarp žodžių žymėti nuliu:

A	Ą	B	C	Č	D	E	Ę	Ė	F	G	H	I	Į	Y	J
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
K	L	M	N	O	P	R	S	Š	T	U	Ų	Ū	V	Z	Ž
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

Tada kiekvieną žodį galėsime užrašyti skaičiumi, pavyzdžiui,

$$KNYGA = 17 \cdot 33^4 + 20 \cdot 33^3 + 15 \cdot 33^2 + 11 \cdot 33 + 1 = 20896096.$$

Galima ilgą tekstą skaidyti sakiniais ir pastaruosius keisti skaičiais. Galima tiesiog versti tekstą dvejetainių simbolių srautu, pastarąjį skaidyti blokais ir interpretuoti juos kaip natūraliųjų skaičių dvejetaines išraiškas.

20.3. RSA saugumas

Keletas paprastų RSA kriptosistemos atakų, kurių nesudėtinga išvengti.

Jau minėjome, kad esminių RSA kriptosistemos saugumo spragų kol kas neaptikta. Tačiau neatsargiai ja naudojantis, kriptoanalitikas gali įgyti galimybę surasti privatųjį raktą. Pavyzdžiui, jei pranešimas M nėra tarpusavyje pirminis su n, tai jo šifras C irgi turės su n netrivialų bendrą daliklį (bent vieną iš skaičių p, q) ir jį galima suskaičiuoti Euklido algoritmu. Tiesa, tikimybė, kad pranešimas bus p arba q kartotinis, labai nedidelė, viso labo tik

$$\frac{1}{p} + \frac{1}{q} - \frac{1}{pq}.$$

Reikia šiek tiek atsargumo parenkant pirminius skaičius p, q. Tarkime, pavyzdžiui, A pavyko surasti vieną, jo nuomone, pakankamai didelį pirminį skaičių p, pavyzdžiui, $p = 3138428376749$ ir jis nutarė kito pirminio ieškoti netoli p, t. y. tikrinti, ar $p + 2, p + 3, \ldots$ yra pirminiai. Neilgai trukus jis tokį pirminį surado ir, apskaičiavęs RSA modulį

$$n = 984973267632859020525139 1,$$

jį paskelbė. Kriptoanalitikas Z, turėdamas marias laisvo laiko ir gerų mokyklinės matematikos žinių, užsirašė paprastas lygybes

$$4n = (p+q)^2 - (p-q)^2, \quad (p+q)^2 = 4n + (p-q)^2$$

ir sukūrė paprastą programą, kuri tikrina, ar kartais kuris nors iš skaičių

$$4n + 2^2, 4n + 3^2, 4n + 4^2, \ldots$$

nėra pilnas kvadratas. Ilgai jo kompiuteriui dirbti nereikėjo:

$$4n + 110^2 = 6276856753608^2.$$

Dabar liko vieni niekai:

$$\begin{cases} p + q = 6276856753608, \\ p - q = 110 \end{cases}$$

ir $p = 3138428376749, q = 3138428376859$. Šio pavyzdžio moralas toks: jeigu norite saugumo, skaičius $|p - q|$ turi būti pakankamai didelis.

Atskirais atvejais gali būti sėkmingos ir kitokios atakos. Pavyzdžiui, efektyvią ataką galima atlikti, kai privačiojo rakto komponentė d yra maža palyginus su n.

139 teorema. *Jeigu $K_v = \langle n, e\rangle, K_p = \langle d\rangle$ yra RSA kriptosistemos raktai ir p, q, d tenkina sąlygas*

$$q < p < 2q, \quad d < \frac{1}{3} n^{\frac{1}{4}},$$

tai iš viešojo rakto polinominiu algoritmu galima rasti privatųjį.

Šią mažo privačiojo rakto ataką sugalvojo M. Wieneris[22]. Nesigilindami į teoremos įrodymą, panagrinėkime, kaip vykdoma privačiojo rakto paieška.

Kiekvieną paprastąją nesuprastinamą trupmeną $\frac{a}{b}$ $(a < b)$ galima užrašyti baigtine grandinine trupmena. Kaip tai daroma, pamatysite iš pavyzdžio. Tarkime, $a = 17,\ b = 57$. Pirmiausia atlikime didžiausiojo bendrojo daliklio ieškojimo Euklido algoritmu žingsnius:

$$\begin{aligned} 57 &= 3 \cdot 17 + 6, \quad \frac{57}{17} = 3 + \frac{6}{17} \\ 17 &= 2 \cdot 6 + 5, \quad \frac{17}{6} = 2 + \frac{5}{6}, \\ 6 &= 1 \cdot 5 + 1, \quad \frac{6}{5} = 1 + \frac{1}{5}. \end{aligned}$$

Skaičiaus skleidinys grandinine trupmena atrodys taip:

$$\frac{17}{57} = \frac{1}{\frac{57}{17}} = \frac{1}{3 + \frac{6}{17}} = \ldots = \frac{1}{3 + \cfrac{1}{2 + \cfrac{1}{1 + \frac{1}{5}}}}.$$

[22] M. Wiener. Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information Theory. **36**, 1990, p. 553–558.

Grandininę trupmeną visiškai apibrėžia skaičiai vardikliuose. Taigi kad nereiktų rašyti tokios ilgos grandinės, galime ją žymėti tiesiog

$$\frac{17}{57} = [3; 2; 1; 5].$$

Sumažindami koeficientų kiekį, sudarysime dar tris grandinines trupmenas:

$$t_1 = [3] = \frac{1}{3},\ t_2 = [3;2] = \frac{1}{3+\frac{1}{2}} = \frac{2}{7},\ t_3 = [3;2;1] = \frac{3}{10},\ t_4 = [3;2;1;5] = \frac{17}{57}.$$

Šias trupmenas vadinsime skaičiaus $\frac{17}{57}$ konvergentėmis. Paskutinioji konvergentė lygi pačiam skaičiui. Bet kokio skaičiaus konvergentės skaičiuojamos efektyviai.

Ryšys su RSA kriptosistema yra toks:

jeigu RSA dydžiai tenkina Wienerio teoremos sąlygas, tai kriptoanalitikui žinomos trupmenos $\frac{e}{n}$ vienos konvergentės vardiklis lygus privačiajam raktui d!

Įveikti RSA reiškia sugalvoti efektyvų būdą, kaip dešifruoti šifrą, kai dešifravimo raktas nežinomas. Jeigu turėtume efektyvų algoritmą kriptosistemos moduliui n skaidyti pirminiais daugikliais, surastume $\varphi(n)$, o tada jau ir privatųjį raktą. Taigi skaičių skaidymo uždavinio sprendimas duoda būdą įveikti ir RSA. Galbūt įmanoma RSA šifrą dešifruoti ir nenustačius privataus rakto? Ar toks „aplinkinis“ RSA įveikimo metodas taip pat duotų galimybę efektyviai skaidyti natūraliuosius skaičius pirminiais daugikliais? Tai nėra žinoma. Tačiau jeigu RSA būtų įveikiama nustatant privatųjį raktą, tai ir modulį būtų galima efektyviai skaidyti.

140 teorema. *Jeigu žinomi abu RSA kriptosistemos raktai $K_v = \langle n, e\rangle$ ir $K_p = \langle d\rangle$, tai n galima išskaidyti naudojant tikimybinį polinominį algoritmą.*

Įrodymas. Kadangi kriptoanalitikas žino ir e, ir d, jis gali apskaičiuoti $ed-1 = k\varphi(n)$. Kiekvienam skaičiui a, $(a,n) = 1$, teisingas lyginys $a^{ed-1} \equiv 1 \pmod{n}$. Tačiau gali būti, kad parinktajam a lyginys $a^m \equiv 1 \pmod{n}$ teisingas ir su mažesniu laipsnio rodikliu m.

Galime greitai nustatyti, iš kokio dvejeto laipsnio dalijasi $ed-1$, ir surasti tokią išraišką:

$$ed - 1 = 2^s t, \quad (2,t) = 1.$$

Parinkime $a, (a,n) = 1$, ir skaičiuokime:

$$a_0 \equiv a^t \pmod{n}, \quad a_1 \equiv a_0^2 \pmod{n}, \quad \ldots \quad a_i \equiv a_{i-1}^2 \pmod{n}, \ldots$$

Kuris nors iš elementų a_j bus lygus 1. Tarkime, v yra mažiausias indeksas, su kuriuo

$$a_{v-1} \not\equiv 1 \pmod{n}, \quad a_v \equiv 1 \pmod{n}.$$

Kadangi

$$a_v \equiv a^{2^v t} \equiv a_{v-1}^2 \,(\mathrm{mod}\, n), \quad \text{tai} \quad a_v - 1 \equiv (a_{v-1}-1)(a_{v-1}+1) \equiv 0 \,(\mathrm{mod}\, n).$$

Dabar galima bandyti sėkmę: sandauga $(a_{v-1}-1)(a_{v-1}+1)$ dalijasi iš n; jeigu nei vienas iš abiejų daugiklių nesidalytų iš n, tai vienas turėtų dalytis iš p, kitas iš q. Tada Euklido algoritmu surastume:

$$p = (a_{v-1}-1, n), \quad q = (a_{v-1}+1, n).$$

O jeigu abu daugikliai dalytųsi iš n? Tada tektų bandyti laimę su kitu a. Todėl šis algoritmas ir yra tikimybinis.

Taigi organizuojant grupės dalyvių tarpusavio ryšių apsaugą su RSA, reikia pasirūpinti, kad moduliai būtų skirtingi. Jeigu dviejų dalyvių moduliai bus vienodi, tai tas, kuris išmano kriptografiją, galės sužinoti savo kolegos privatųjį raktą, išskaidęs bendrąjį modulį pirminiais daugikliais (jeigu ne jis pats sukūrė savo raktus, tai to skaidinio ir pats nežino).

Bendras kelių vartotojų modulis kelia pavojų ir dėl kitos priežasties. Tegu dviejų ryšio dalyvių RSA viešieji raktai yra $K_{v,A} = \langle n, e_A\rangle, K_{v,B} = \langle n, e_B\rangle, (e_A, e_B) = 1$. Tarkime, kažkas pasiuntė jiems abiem to paties pranešimo šifrus:

$$C_1 \equiv M^{e_A} \,(\mathrm{mod}\, n), \quad C_2 \equiv M^{e_B} \,(\mathrm{mod}\, n).$$

Kriptoanalitikas Zigmas žino $K_{v,A}, K_{v,B}, C_1, C_2$. Atskleisti M jam visai nesunku: suradęs Euklido algoritmu skaičius a, b, su kuriais $ae_A + be_B = 1$ jis lengvai apskaičiuos

$$C_1^a C_2^b \equiv M^{ae_A + be_B} \equiv M \,(\mathrm{mod}\, n).$$

20.4. Pohligo-Hellmano ir Massey-Omura kriptosistemos

Tai dvi įdomios variacijos RSA kriptosistemos tema.

Jeigu kriptosistema yra viešojo rakto, tai nereiškia, kad šifravimui skirtą raktą būtina paskelbti. Jeigu nenorite – neskelbkite. Pavyzdžiui, perduokite RSA šifravimo raktą savo draugams saugiu būdu ir naudokitės RSA kaip paprasta simetrine kriptosistema. Jeigu daug šifruoti nereikia, toks viešojo rakto kriptosistemos naudojimas „ne pagal paskirtį“ visiškai tinkamas. Tačiau jeigu tenka šifruoti didelius duomenų srautus, geriau jau pasitelkti kokią nors įprastinę simetrinę kriptosistemą, nes ji tiesiog daug greitesnė.

O RSA naudojimo būdas nepaskelbiant raktų kriptografijoje netgi turi atskirą vardą – tai Pohligo-Hellmano simetrinė kriptosistema. Tiesa, dažniau taip vadinamas kiek pakeistas kriptosistemos variantas.

Pohligo-Hellmano kriptosistema

Pranešimai ir šifrai – aibės $\mathbb{Z}_p^*$ skaičiai, p – didelis pirminis skaičius.

Šifravimo ir dešifravimo raktai:
$K_e = e,\ K_d = d,\ ed \equiv 1 \pmod{p-1}$ – abu raktus gauna ir A, ir B.

Šifravimas: $C \equiv e(M|K_e) \equiv M^e \pmod{p}$.

Dešifravimas: $M \equiv d(C|K_d) \equiv C^d \pmod{p}$.

Pohligo-Hellmano kriptosistemos modulis p netgi gali būti paskelbtas, galimybių sužinoti e, d kriptoanalitikui toks paskelbimas nesuteiks. Susitarti dėl modulio A ir B gali ir nesaugiu kanalu, pavyzdžiui, telefonu. Skaičiai e ir d jiems turi būti įteikti saugiai. Įdomu, kad galima šia kriptosistema naudotis visiškai neperdavus skaičių e ir d! Žinoma, bent jau dešifravimo algoritmą teks pakeisti, todėl turėsime naują kriptosistemą. Tai Massey-Omura kriptosistema; iš tiesų tai tam tikras kriptografinis protokolas, suteikiantis simetrinei kriptosistemai viešojo rakto kriptosistemos savybes.

Massey-Omura kriptosistema

Pranešimai ir šifrai – aibės $\mathbb{Z}_p^*$ skaičiai, p – didelis pirminis.

Viešasis raktas: pirminis skaičius p.
Privatieji raktai: ryšio dalyviai A, B, naudodami p, sudaro savo privačiuosius raktus iš dviejų komponenčių:

$$K_A = \langle e_A, d_A\rangle, K_B = \langle e_B, d_B\rangle,$$
$$e_A d_A \equiv 1 \pmod{p-1}, e_B d_B \equiv 1 \pmod{p-1}.$$

Šifravimas: A šifruoja $C \equiv e(M|K_A) \equiv M^{e_A} \pmod{p}$ ir siunčia B.
Dešifravimas: B gavusi C, šifruoja $C_1 \equiv e(C|K_B) \equiv C^{e_B} \pmod{p}$ ir siunčia A. A gavęs C_1, dešifruoja $C_2 \equiv d(C_1|K_A) \equiv C_1^{d_A} \pmod{p}$ ir siunčia B. B, gavusi C_2, dešifruoja $d(C_2|K_B) \equiv C_2^{d_B} \equiv M \pmod{p}$.

Įsitikinkime, kad dešifruojama bus tikrai teisingai:

$$\begin{aligned} C_2^{d_B} &\equiv ((C_1)^{d_A})^{d_B} \equiv ((C^{e_B})^{d_A})^{d_B} \equiv (((M^{e_A})^{e_B})^{d_A})^{d_B} \\ &\equiv (M^{e_A d_A})^{e_B d_B} \equiv M^{e_B d_B} \equiv M \pmod{p}. \end{aligned}$$

RSA kriptosistemoje visi ryšio dalyviai savo viešuosius raktus turi paskelbti su skirtingais moduliais. Jeigu dviejų dalyvių moduliai vienodi, jie gali

nustatyti vienas kito privačiuosius raktus. Bendro modulio pavojaus nėra Massey-Omura kriptosistemoje. Įdomi šios kriptosistemos savybė – vienas asmuo gali paskelbti visiems ryšio dalyviams bendrą modulį. Tada bet kurie du dalyviai, naudodamiesi Massey-Omura protokolu, galės užmegzti šifruotą ryšį. Šia ypatybe pasinaudojama kai kuriuose kriptografiniuose protokoluose.

Iš ryšio kanalo, kuriuo naudojasi A ir B, kriptoanalitikas gali gauti šifrus C, C_1, C_2. Tačiau jam iš to mažai naudos, nes, pavyzdžiui, lyginyje

$$C \equiv M^{e_A} \pmod{p}$$

yra netgi du nežinomieji.

20.5. Rabino kriptosistema

Viešojo rakto kriptosistemų kūrėjai naudojasi skaičiavimo uždaviniais, kuriuos sunku spręsti. Vienas tokių uždavinių – kvadratinių lygčių sprendimas žieduose $\mathbb{Z}_n$. Šis uždavinys – Rabino kriptosistemos pagrindas.

Matėme, kad RSA kriptosistemos įveikimo uždavinys nėra sunkesnis už natūraliųjų skaičių skaidymo pirminiais daugikliais uždavinį. Tačiau nežinoma, ar jie ekvivalentūs, t. y. nežinoma, ar suradus efektyvų RSA kriptosistemos šifro dešifravimo be privataus rakto algoritmą, jį būtų galima panaudoti natūraliųjų skaičių skaidymo pirminiais daugikliais uždaviniui.

Šiame skyrelyje išnagrinėsime kriptosistemą, kurios „sulaužymo“ uždavinys sudėtingumo požiūriu ekvivalentus natūraliųjų skaičių skaidymo pirminiais daugikliais uždaviniui.

Raktų parinkimas. Parinkę du didelius pirminius skaičius p, q, iš jų sudarome privatųjį raktą $K_p = \langle p, q\rangle$ ir viešąjį raktą $K_v = \langle n\rangle$, $n = pq$.

Šifravimas. Pranešimų ir šifrų aibė ta pati: $\mathcal{M} = \mathcal{C} = \{0, 1, \ldots, n-1\}$. Šifravimui pasirinkime parametrą $a, 0 \le a < n$ (jis yra viešas). Tada šifras sudaromas taip:

$$c = e(m|K_v) \equiv m(m+a) \pmod{n}.$$

Dešifravimas. Dešifruojant reikia spręsti lyginį

$$x(x+a) \equiv c \pmod{n}.$$

Šį lyginį galima pertvarkyti į paprastesnį:

$$x(x+a) \equiv x^2 + 2\cdot 2^{-1}\cdot x\cdot a + (2^{-1}a)^2, -(2^{-1}a)^2 \equiv c \pmod{n}$$
$$y^2 \equiv d \pmod{n}, \quad y \equiv x + 2^{-1}a \pmod{n}, \quad d \equiv c + (2^{-1}a)^2 \pmod{n}.$$

Tačiau greito algoritmo lyginio $y^2 \equiv d \pmod{n}$ sprendiniui rasti taip pat nėra. Taigi kriptoanalitikas turi spręsti sudėtingą skaičiavimo uždavinį.

Sprendžiant variantų perrinkimo algoritmu, blogiausiu atveju prireiks maždaug n perrankos operacijų.

Privataus rakto savininkas gali vietoj lyginio $y^2 \equiv d \pmod{n}$ spręsti du lyginius $u^2 \equiv d \pmod{p}$ ir $v^2 \equiv d \pmod{q}$. Net ir perrinkimo algoritmu jis greičiau ras sprendinius, nes perrankų skaičius $p+q$ daug mažesnis skaičius už $n = pq$. Iš rastųjų sprendinių jis gali sudaryti lyginio $y^2 \equiv d \pmod{n}$ sprendinį, naudodamasis kiniškąja liekanų teorema. Tačiau ši teorema duoda net keturis sprendinius. Iš jų reikia atsirinkti vieną. Jeigu buvo šifruotas koks nors tekstas, tai mažai tikėtina, kad visi keturi sprendiniai reikš ką nors prasminga. Jeigu vertinti pranešimų pagal prasmę negalime, tada reikia kokio nors susitarimo, kaip atskirti tikrąjį pranešimą nuo „pašalinių“ . Pavyzdžiui, galima susitarti, kad tikrasis pranešimas turi baigtis tam tikra galūne.

Rabino kriptosistema

Pranešimai ir šifrai – aibės $\mathbb{Z}_n$, $n = pq$, skaičiai; čia p, q yra pirminiai, tenkinantys sąlygas $p, q \equiv 3 \pmod{4}$.

Privatusis raktas: $K_p = \langle p, q \rangle$.

Viešasis raktas: $K_v = \langle n \rangle$.

Šifravimas: $C = e(M|K_v) \equiv M^2 \pmod{n}$.
Dešifravimas: randami skaičiai u, v,

$$u \equiv M \pmod{p}, \quad v \equiv M \pmod{q},$$
$$u \equiv C^{(p+1)/4} \pmod{p}, \quad v \equiv C^{(q+1)/4} \pmod{q},$$

naudojantis kiniškąja liekanų teorema, randami keturi skaičiai M_i

$$M_1 \equiv u \pmod{p},\ M_1 \equiv v \pmod{q},$$
$$M_2 \equiv -u \pmod{p},\ M_2 \equiv v \pmod{q},$$
$$M_3 \equiv u \pmod{p},\ M_3 \equiv -v \pmod{q},$$
$$M_4 \equiv -u \pmod{p},\ M_4 \equiv v \pmod{q}$$

ir iš jų atrenkamas pranešimas $M = d(C|K_p)$.

Dešifruojant tenka spręsti lyginius pirminiais moduliais. Tai irgi gali būti nelengva. Tačiau ankstesniame skyrelyje įrodėme teiginį, kad atskiru atveju lyginio sprendinį galima surasti visai lengvai. Prisiminkime tą teoremą.

141 teorema. *Jei $p \equiv 3 \pmod{4}$ ir $u^2 \equiv d \pmod{p}$ turi sprendinį, tai vienas iš sprendinių yra $u_0 = d^{(p+1)/4}$.*

Taigi parinkus Rabino kriptosistemos pirminius iš progresijos $4m+3$, dešifravimui perrankos neprireiks, abu lyginio $u^2 \equiv d \pmod{p}$ sprendiniai gaunami panaudojus vieną kėlimo laipsniu veiksmą. Todėl kriptosistema dažniausiai ir naudojama su tokiais pirminiais.

Įveikti Rabino kriptosistemą reiškia sudaryti efektyvų algoritmą lyginiui $x^2 \equiv d \pmod{n}, n = pq$, spręsti. Jeigu sprendinys egzistuoja, tai jų yra net keturi. Gavę iš algoritmo sprendinius x_1, x_2, x_3, x_4, galime sudaryti dvi jų poras, pavyzdžiui, $x_1 \equiv -x_3 \pmod{n}, x_2 \equiv -x_4 \pmod{n}$. Tada su tam tikrais sveikaisiais skaičiais a, b, u, v bus teisingos lygybės:

$$x_1 \equiv vap + ubq \pmod{n}, \quad x_2 \equiv -vap + ubq \pmod{n},$$

čia p, q – skaičiaus n pirminiai dalikliai. Tuomet $x_1 + x_2 \equiv 0 \pmod{q}$, bet $x_1 + x_2$ nesidalija iš n. Todėl skaičiuodami didžiausiąjį bendrąjį skaičių n ir $x_1 + x_2$, daliklį surasime vieną iš n pirminių daliklių: $(x_1 + x_2, n) = q$.

Taigi Rabino kriptosistema yra viena iš nedaugelio viešojo rakto kriptosistemų, kurių saugumas yra įrodytas, t. y. matematiškai nustatyta, kad kriptosistemos įveikimas tolygus sudėtingo skaičiavimo uždavinio sprendimui.

20.6. Blomo-Goldwasserio tikimybinė kriptosistema

Dar viena kriptosistema, kurioje naudojamasi kvadratiniais sprendiniais. Ji vadinama tikimybine todėl, kad pranešimo šifras priklauso nuo atsitiktinio skaičiaus, kurį pasirenka šifruotojas.

Kriptosistemos raktai parenkami kaip Rabino kriptosistemoje.

Raktų sudarymas. Tegu p, q yra du pirminiai skaičiai su sąlyga $p, q \equiv 3 \pmod{4}$. Tada viešasis raktas $K_v = \langle n \rangle$, o privatusis – $K_p = \langle p, q \rangle$.

Šifravimas. Šifruojami bet kokio ilgio dvejetainės abėcėlės žodžiai, taigi $\mathcal{M} = \{0,1\}^*$. Šifruojamas pranešimas skaidomas į h ilgio fragmentus:

$$M = m_1 m_2 \dots m_t, \quad m_i \in \{0,1\}^h, \quad h = [\log_2 k], \quad k = [\log_2 n].$$

Šifruojama taip. Pasirinkus atsitiktinį skaičių $r \in \mathbb{Z}_n^*$, apskaičiuojamas kvadratas $x_0 \equiv r^2 \pmod{n}$. Kiekvienas fragmentas m_i šifruojamas atskirai:

$$x_i \equiv x_{i-1}^2 \pmod{n}, \quad c_i = e(m_i | K_v) = m_i \oplus y_i,$$

čia y_i yra žodis, sudarytas iš skaičiaus x_i dvejetainės išraiškos h paskutinių (mažiausiai reikšmingų) bitų. Baigus šifruoti visus blokus, dar surandamas skaičius $x_{t+1} \equiv x_t^2 \pmod{n}$ ir siunčiamas šifras

$$C = \langle c_1 c_2 \dots c_t, x_{t+1} \rangle.$$

Pastebėkime, kad visi skaičiai yra tarpusavyje pirminiai su n, taigi ir su p bei q.

Dešifravimas. Akivaizdu, kad, norint dešifruoti C, pakanka rasti x_0. Tada galima skaičiuoti kvadratus $x_i \equiv x_{i-1}^2 \pmod{n}$, surasti y_i ir dešifruoti pranešimo fragmentus: $m_i = c_i \oplus y_i$.

Skaičiui x_i rasti ir reikalingas x_{t+1}. Pirmiausia randami skaičiai $u \equiv x_0 \pmod{p}, v \equiv x_0 \pmod{p}$, o tada, naudojantis kiniškąja liekanų teorema, apskaičiuojamas ir x_0.

Kaip žinant x_{t+1}, surasti $u, u \equiv x_0 \pmod{p}$? Kadangi skaičiai x_i susieti lyginiais $x_i \equiv x_{i-1}^2 \pmod{n}$, tai jiems taip pat bus teisingi ir lyginiai $x_i \equiv x_{i-1}^2 \pmod{p}$. Taigi visi mūsų skaičiai $x_0, x_1, \ldots, x_{t+1}$ nesidalija iš p ir yra kvadratai moduliu p.

Šiek tiek paskaičiuokime:

$$x_{t+1}^{(p+1)/4} \equiv x_t^{(p+1)/2} \equiv x_t (x_t)^{(p-1)/2} \equiv x_t (x_{t-1})^{(p-1)} \equiv x_t \pmod{p}.$$

Taigi keldami x_{t+1} laipsniu $(p+1)/4$, galime rasti $x_t \pmod{p}$. Keldami x_t tuo pačiu laipsniu, galime rasti $x_{t-1} \pmod{p}$ ir t. t. Arba iš karto

$$x_0 \equiv (x_t)^{((p+1)/4)^{(t+1)}} \pmod{p}.$$

Skaičiuodami visų pirma galime surasti $a \equiv ((p+1)/4)^{(t+1)} \pmod{p-1}$, o tada apskaičiuoti $u \equiv x_{t+1}^a \pmod{p}$, t. y. $u \equiv x_0 \pmod{p}$.

Skaičiavimas su q, žinoma, yra analogiškas. Šifruotojas gali pasirinkti parametrą r. Nuo šio skaičiaus priklauso ir šifras. Taigi tas pats pranešimas su tuo pačiu raktu gali būti šifruojamas įvairiai.

20.7. ElGamalio kriptosistema

Ši kriptosistema, tiksliau tariant, įvairūs jos variantai, populiarumu rungiasi netgi su RSA. Jos kūrėjas Taheras ElGamalis yra vienas žymiausių mūsų laikų kriptografijos autoritetų. Kasdieną milijonai žmonių, naršydami po Internetą, naudojasi SSL protokolu, kuriame įdiegtos ElGamalio idėjos.

Nagrinėjome kriptosistemas, kurių saugumas pagrįstas sveikųjų skaičių skaidymo bei kvadratinių lyginių sprendimo uždavinių sudėtingumu. Laikas patyrinėti kriptosistemas, kuriose panaudojamas dar vienas sudėtingas skaičiavimo uždavinys – diskrečiojo logaritmo radimo.

Raktų sudarymas. Tegu p – didelis pirminis skaičius, kad rasti diskretųjį logaritmą moduliu p būtų sunku, g – primityvioji šaknis moduliu p, kitaip tariant – multiplikatyvios grupės $\mathbb{F}_p^*$ generuojantis elementas. Pasirinkime skaičių $a, 0 < a \leq p-1$ ir sudarykime viešąjį raktą K_v pranešimams šifruoti ir privatųjį dešifravimo raktą K_p:

$$K_v = \langle p, g, \beta \rangle, \quad \beta \equiv g^a \pmod{p}, \quad K_p = \langle a \rangle.$$

Šifravimas. Pranešimų aibė $\mathcal{M}$ – visi nenuliniai $\mathbb{F}_p^*$ elementai. Prieš šifruojant parenkamas skaičius $k \in \mathbb{F}_p^*$ ir pranešimo M šifras sudaromas taip:

$$e(M|K_v) = \langle C_1, C_2 \rangle = C, \quad C_1 \equiv g^k \pmod{p}, \quad C_2 \equiv M\beta^k \pmod{p}.$$

Dešifravimas. Šifro $C = \langle C_1, C_2 \rangle$ dešifravimas:

$$d(C|K_p) \equiv C_2(C_1^a)^{-1} \pmod{p}.$$

Nesunku patikrinti, kad dešifravimo procedūra tikrai veikia:

$$C_2(C_1^a)^{-1} \equiv M\beta^k(g^{ka})^{-1} \equiv Mg^{ak-ak} \equiv M \pmod{p}.$$

Jeigu kriptoanalitikui pavyktų iš lyginio $C_1 \equiv g^k \pmod{p}$ surasti $k = \log_g C_1$, tai jis be vargo nustatytų ir M. Tačiau kriptosistema nebūtų įveikta, nes kitą kartą šifruotojas panaudotų kitą k reikšmę. Žinoma, kriptosistema būtų visiškai įveikta, jeigu kriptoanalitikas apskaičiuotų diskretųjį logaritmą $a = \log_g \beta$. Tačiau bent kol kas diskrečiajam logaritmui skaičiuoti greitų būdų nėra.

Ar būtina kiekvienam šifruojamam pranešimui parinkti vis kitą k reikšmę? Panagrinėkime, kas gali atsitikti, jeigu du skirtingus pranešimus M ir M^* šifruotume ElGamalio kriptosistema su tuo pačiu k. Tada jų šifrai būtų

$$C = e(M|K_v) = \langle C_1, C_2 \rangle, \quad C^* = e(M^*|K_v) = \langle C_1, C_2^* \rangle.$$

Kriptoanalitikas kaipmat pastebėjęs, kad pirmosios komponentės sutampa galėtų apskaičiuoti

$$C_2^{-1}C_2^* \equiv (M\beta^k)^{-1}M^*\beta^k \equiv M^{-1}M^* \pmod{p}.$$

Šis lyginys nustato ryšį tarp pranešimų: jeigu vieną iš jų sužinotume, surastume ir kitą. O jeigu su tuo pačiu k būtų užšifruota ne du, bet keli šimtai pranešimų? Kad Z visus juos atskleistų, pakanka sužinoti vieno iš jų turinį.

ElGamalio kriptosistemoje skaičiuojama su kūno $\mathbb{F}_p$ multiplikatyviosios grupės elementais. Vietoj šios grupės galima naudoti bet kokią baigtinę ciklinę grupę, kurioje diskrečiojo logaritmo uždavinį yra sunku spręsti. Tokių grupių yra daug, tačiau yra ir tokių, kuriose diskretųjį algoritmą $\log_g x$ rasti vienas juokas. Štai tokios kriptografijai netinkamos ciklinės grupės pavyzdys: $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$ su sudėties moduliu n veiksmu. Kokie elementai generuoja šią grupę ir kaip skaičiuojami diskretieji logaritmai?

Bendroji ElGamalio kriptosistemos schema
Pranešimai ir šifrai – baigtinės ciklinės grupės G, kurioje diskrečiojo logaritmo skaičiavimo uždavinys yra sunkus, elementai, t. y. $\mathcal{M} = \mathcal{C} = G$.
Privatusis raktas: $K_p = \langle a \rangle$, a natūralusis skaičius, $a < \lvert G \rvert$.
Viešasis raktas: $K_v = \langle G, g, \beta \rangle$, $\beta = g^a$.
Šifravimas: pranešimui $M \in G$ šifruoti atsitiktinai parenkamas natūralusis skaičius k, skaičiuojama $C_1 = g^k$ ir $C_2 = M\beta^k$. Sudaromas šifras $C = e(M\lvert K_v) = \langle C_1, C_2 \rangle$.
Dešifravimas: $d(C\lvert K_p) = C_2(C_1^a)^{-1} = M$.

Galime ElGamalio kriptosistemą sukonstruoti, pavyzdžiui, panaudoję kokio nors kūno $\mathbb{F}_{p^m}$ multiplikatyviąją grupę.

Šiuo metu labai intensyviai tyrinėjamos galimybės kriptografijai panaudoti baigtines grupes, susijusias su kreivėmis, kurių lygtys yra

$$y^2 = ax^3 + ax + b.$$

Šios kreivės vadinamos elipsinėmis. Naudojant tokias grupes tam pačiam saugumo lygiui kaip kriptosistemose su $\mathbb{F}_p$ garantuoti, pakanka trumpesnių raktų ir reikia mažiau skaičiavimų. Ši savybė yra labai svarbi diegiant kriptosistemas įrenginiuose su ribotais skaičiavimo resursais, pavyzdžiui, autentifikavimui naudojamose kortelėse.

20.8. McElliece'as: dešifravimas yra dekodavimas

Šifravimas yra duomenų pradinės struktūros „sugadinimas“ , dešifravimas – atstatymas. Panašūs veiksmai vyksta perduodant informaciją nepatikimu kanalu. Robertui McElliece'ui kilo mintis, kad perdavimo trūkumas – simbolių iškraipymai – tampa privalumu, kai norime pranešimus apsaugoti nuo tų, kam jie nėra skirti.

Kodavimo teorijos skyriuje aptarėme, kaip perdavimo klaidoms taisyti naudojami tiesiniai kodai. Jeigu klaidų skaičius neviršija pasirinktojo kodo galimybių, tai pagal gautąjį iškraipytą kodo žodį galime atkurti tą, kurį mums siuntė siuntėjas. Nagrinėjome visiems tiesiniams kodams tinkantį lyderių-sindromų metodą. Ar klaidų taisymas šiuo metu vyksta greitai? Kartais iš tikrųjų greitai, pavyzdžiui, dvejetainių Hammingo kodų atveju. Tačiau apskritai dekodavimo algoritmas – ne polinominis. Galima pasakyti dar daugiau – tiesinių kodų dekodavimas, t. y. siųstų žodžių radimas pagal gautuosius, kai iškraipymų kiekis nėra didesnis už to kodo taisomų klaidų skaičių, yra **NP** pilnas uždavinys.

Tačiau yra kodų, kurie specialiais kodų savybes panaudojančiais metodais dekoduojami greitai. Panašiai yra ir su „kuprinės uždaviniu“: nors

tai **NP** pilnas uždavinys, bet sparčiai didėjančių svorių sistemos atveju jis sprendžiamas polinominiu algoritmu.

Šiomis priešingybėmis – dekoduoti yra sunku, bet kartais lengva – ir pasinaudojama McElliece'o kriptosistemoje.

Raktų sudarymas. Tegu $\mathbf{C} \subset \mathbb{F}_q^n$ yra koks nors kodas, kuriam dekoduoti yra greitas algoritmas (pavyzdžiui, BCH kodų šeimos narys). Tegu kodo dimensija yra k, generuojanti matrica G, o taisomų klaidų skaičius t. Sudarykime dar dvi matricas: neišsigimusią $k \times k$ matricą S iš kūno $\mathbb{F}_q$ elementų ir $n \times n$ matricą P, kuri gaunama iš vienetinės, kokia nors tvarka sukeičiant stulpelius. Matrica P bus naudojama žodžio komponentėms sukeisti: jeigu $\mathbf{x} \in \mathbb{F}_q^n$, tai $\mathbf{x}P$ yra žodis, sudarytas iš tų pačių komponenčių, tik išdėstytų kita tvarka.

Dabar jau galime sudaryti raktus:

$$K_p = \langle S, G, P \rangle, \quad K_v = \langle G^* \rangle, \quad G^* = SGP.$$

Matrica G^* yra tam tikro tiesinio $[k, n]$ kodo $\mathbf{C}^* \subset \mathbb{F}_q^n$ generuojanti matrica. Tai „sugadintas“ geras kodas $\mathbf{C}$, taigi dekoduoti iškraipytus jo žodžius turėtų būti sudėtinga.

Šifravimas ir dešifravimas. Šifruojami pranešimai yra k simbolių ilgio žodžiai iš aibės $\mathcal{M} = \mathbb{F}_q^k$; šifravimą sudaro žodžio kodavimas $\mathbf{C}^*$ žodžiu ir t (arba mažiau) simbolių iškraipymas:

$$C = e(M|K_v) = MG^* + \mathbf{e}, \quad \mathbf{e} \in \mathbb{F}_q^n.$$

Klaidų žodis $\mathbf{e}$ parenkamas atsitiktinai ir jo svoris ne didesnis už t, t. y. jame ne daugiau kaip t nenulinių komponenčių.

Gavėjas iškraipytą žodį (šifrą) dešifruoja taip: apskaičiavęs $C_1 = CP^{-1} = (MS)G + \mathbf{e}P^{-1}$, jis gali manyti, kad C_1 yra kanalu siųstas, bet iškraipytas kodo $\mathbf{C}$ žodis $\mathbf{c} = (MS)G$. Svarbu, kad naujojo klaidų žodžio $\mathbf{e}P^{-1}$ svoris yra ne didesnis už t, todėl klaidas galima ištaisyti. Pritaikęs greitąjį dekodavimo algoritmą, jis gali surasti $M' = MS$ ir $M = M'S^{-1} = d(C|K_p)$.

McElliece'o kriptosistema

Sukonstruojamas tiesinis $[n,k]$ kodas $\mathbf{C} \subset \mathbb{F}_q^n$, taisantis t klaidų, kurio žodžiams dekoduoti yra greitai veikiantis algoritmas, G generuojanti kodo matrica. Pranešimų aibė $\mathcal{M} = \mathbb{F}_q^k$.

Privatusis raktas: $K_p = \langle S, G, P\rangle$, S neišsigimusi $k \times k$ matrica iš $\mathbb{F}_q$ elementų, P – n-os eilės matrica, gauta iš vienetinės, kokia nors tvarka perstačius stulpelius.

Viešasis raktas: $K_v = \langle G^*\rangle, G^* = SGP$.

Šifravimas: $C = e(M|K_v) = MG^* + \mathbf{e}, \mathbf{e} \in \mathbb{F}_q^n, w(\mathbf{e}) \leq t$, žodis $\mathbf{e}$ parenkamas atsitiktinai.

Dešifravimas: $C_1 = CP^{-1}$, žodyje C_1 ištaisius klaidas, randama

$$M' = MS, \quad M = d(C|K_p) = M'S^{-1}.$$

Štai ir viskas. Ši kriptosistema – viena pirmųjų viešojo rakto kriptosistemų. Esminių saugumo spragų kol kas niekas neįžvelgė. Tinkamai parinkus kodą šifravimo ir dešifravimo greičiu ji gali pranokti ir RSA. Tačiau yra ir keletas trūkumų, dėl kurių ji nėra populiari. Viena vertus, tinkamam saugumui garantuoti reikalingi dideli kodo parametrai. Pavyzdžiui, pats R. McElliece'as rekomendavo naudoti kodui iš dvejetainės abėcėlės žodžių

$$n = 1024, \quad k = 524, \quad t = 30.$$

Tada viešąjį raktą sudarys net $nk = 30720$ bitų! Kita vertus, šifro dydis gerokai viršija pranešimo. Tiesa, taip yra ir kitose kriptosistemose, pavyzdžiui, ElGamalio.

21 Skaitmeninių parašų schemos

Prieš šifruojant pranešimą viešojo rakto kriptosistema reikia jį užrašyti schemoje numatytu būdu. Kartais šifruojami dvejetainės abėcėlės žodžiai, kartais pranešimą reikia paversti skaičiumi.

Tą pradinį duomenų užrašymo veiksmą tenka atlikti ir prieš sudarant skaitmeninį parašą. Parengtus pasirašymui skaitmeniniu parašu duomenis vadinsime tekstais. Skaitmeninis teksto x parašas – tai tam tikri duomenys, sukurti naudojant tekstą bei privatųjį pasirašymui skirtą raktą. Tikrinant parašą, naudojamas tekstas x, jo parašas y ir viešasis parašui tikrinti skirtas raktas K_v. Parašas priimamas, jeigu šie trys dydžiai tenkina tam tikrą skaitmeninio parašo schemoje numatytą sąlygą.

115 apibrėžimas. *Skaitmeninių parašų schemą sudaro tekstų aibė $\mathcal{M}$, skaitmeninių parašų aibė $\mathcal{P}$, raktų $K = \langle K_v, K_p\rangle$ aibė $\mathcal{K}$ (čia K_p –*

privačioji parašui sudaryti skirta komponentė, K_v – viešoji parašui tikrinti skirta rakto komponentė) ir parašų sudarymo bei tikrinimo algoritmų šeimos

$$\begin{aligned} sig(\cdot|K_p) &: \mathcal{M} \to \mathcal{P}, \\ ver(\cdot|K_v) &: \mathcal{M} \times \mathcal{P} \to \{0,1\}. \end{aligned}$$

Parašo tikrinimo algoritmai turi savybę:

$$ver(x, sig(x|K_p)|K_v) = 1. \tag{103}$$

Jeigu $y \in \mathcal{P}$ pateikiamas kaip teksto $x \in \mathcal{M}$ parašas, tai parašas pripažįstamas galiojančiu, jeigu $ver(x, y|K_v) = 1$, ir pripažįstamas negaliojančiu, jei $ver(x, y|K_v) = 0$.

Taigi teksto $x \in \mathcal{M}$ skaitmeninis parašas yra $y = sig(x|K_p)$. Įprastiniai parašai tikrinami lyginant juos su pavyzdžiu, o skaitmeniniai – atliekant skaičiavimus, kurie parodo, ar x ir y tenkina tam tikrą matematinį sąryšį.

Daugelį viešojo rakto kriptosistemų galima paversti skaitmeninio parašo schemomis. Iš viešojo rakto K_v nustatyti privatųjį K_p praktiškai neįmanoma. Jeigu neįmanoma ir iš privačiojo rakto nustatyti viešąjį, tai tokią kriptosistemą galime paversti skaitmeninio parašo schema tiesiog paskelbdami dešifravimui skirtą raktą K_p ir paslėpdami viešąjį K_v.

Tada teksto parašas bus jo šifras, kurį gali sukurti tik turintis raktą K_v:

$$y = e(x|K_v).$$

Tikrinant parašą, pateikiama pora $\langle x, y\rangle$. Jeigu reikšmė $x' = d(y|K_p)$ sutampa su x, parašas priimamas. Jeigu x yra tekstas, kurį galime vertinti prasmės požiūriu, tai tikrinimui galima pateikti vien tik parašą y. Jeigu jis „prasmingai“ išsišifruoja, tai gautąjį tekstą galime laikyti pasirašytu to asmens, kurio raktą tikrinimui naudojome. Tikimybė, kad gerai išsišifruos be K_v sudarytas parašas y, labai maža.

Tačiau ne visų viešojo rakto kriptosistemų raktai K_v, K_p turi minėtą savybę. Pavyzdžiui, Rabino kriptosistemoje $K_p = \langle p, q\rangle$, o $K_p = \langle n\rangle, n = pq$. Aišku, kad, paskelbus pirminius p, q, slėpti jų sandaugą nebėra jokios prasmės. Tačiau ir tokiu atveju, kai privačiojo rakto negalima paskelbti, yra paprastas būdas paversti kriptosistemą skaitmeninio parašo schema. Galima dešifravimo algoritmo, pritaikyto pranešimui x, rezultatą $y = d(x|K_p)$ laikyti skaitmeniniu parašu. Tada, tikrinant parašą, pakaks įsitikinti, ar $x = e(y|K_v)$.

21.1. RSA skaitmeniniai parašai

Beveik nieko nereikia keisti RSA kriptosistemoje, jeigu norime ją naudoti kaip skaitmeninių parašų schemą. Vis dėlto, diegdami ją praktiškai, tam tikrų keblumų neišvengtume.

Kadangi RSA kriptosistemoje šifruojama ir dešifruojama tuo pačiu algoritmu tik su skirtingais raktais, tai, norint RSA naudoti kaip skaitmeninio parašo schemą, nieko nereikia keisti.

Jeigu A viešasis RSA raktas yra $K_{v,A} = \langle n_A, e_A \rangle$, o privatusis $K_{p,A} = \langle d_A \rangle$, tai pranešimo x parašą A gali sudaryti tiesiog taip:

$$y = sig(x|K_{p,A}) \equiv x^{d_A} \pmod{n_A},$$

ir siųsti B porą $\langle x, y \rangle$ arba tiesiog y. Parašo tikrinimas – dešifravimas su viešuoju raktu:

$$x \equiv y^{e_A} \pmod{n_A}.$$

Tikrinti A parašą ir skaityti pasirašytą tekstą galės visi, kas panorės. Kaip pasiekti, kad tik B galėtų perskaityti A laišką ir, patikrinusi parašą, būtų tikra, kad laišką atsiuntė tikrai A?

Patarkime B irgi susikurti RSA kriptosistemą. Tegu B raktai bus $K_{v,B} = \langle n_B, e_B \rangle$ ir $K_{p,B} = \langle d_B \rangle$. Dabar A, norėdamas siųsti ir šifruotą, ir pasirašytą laišką x, gali elgtis dvejopai: siųsti c_1 arba c_2:

$$c_1 = e(sig(x|d_A)|e_B), \quad c_2 = sig(e(x|e_B)|d_A).$$

Kurį būdą pasirinkti? Abu būdai ne tik nėra lygiaverčiai, bet vienas netgi gali neveikti. Tarkime, pavyzdžiui, $n_A > n_B$. Kad, sudarę parašą $y = sig(x|d_A)$, galėtume jį sėkmingai užšifruoti, turi būti teisinga nelygybė $y < n_B$. Tačiau $n_A > n_B$, todėl gali būti ir $y > n_B$. Tada šifruodami sugadinsime savo laišką. Taigi tokiu atveju reikia pirma šifruoti, o tada pasirašyti, t. y. siųsti c_2. Tačiau šis variantas irgi turi šiokį tokį trūkumą. Perėmęs siunčiamą c_2, kriptoanalitikas Z gali jį dešifruoti A viešuoju raktu, t. y. surasti $c = e(x|e_B)$, ir, jeigu jis irgi yra šios ryšių sistemos dalyvis, pasiųsti jį B savo vardu: $c_3 = sig(c|d_Z)$. Birutė bus kiek suklaidinta.

Tokių keblumų galima išvengti. Kiekvienam dalyviui sukurkime po du komplektus RSA raktų su skirtingais moduliais. Pirmoji raktų pora skirta skaitmeniniams parašams, o antroji – šifravimui. Jeigu parašams sudaryti skirtų raktų moduliai bus mažesni už tam tikrą nustatytą slenksčio reikšmę T, o šifravimui skirtų raktų moduliai didesni už T – išvengsime visų nesusipratimų pirma pasirašydami, o paskui šifruodami. Tada A, norėdamas pasiųsti pasirašytą ir šifruotą laišką B, turėtų siųsti $c_1 = e(sig(x|d_A^{(1)})|e_B^{(2)})$, čia $d_A^{(1)}$ yra A parašams sudaryti skirtas privatusis raktas, o $e_B^{(2)}$ – šifruotiems laiškams rašyti B viešasis raktas.

RSA skaitmeninio parašo schema
Tekstų aibė $\mathbb{Z}_n$, $n = pq$, p, q – du pakankamai dideli pirminiai skaičiai. **Privatusis parašams sudaryti skirtas raktas:** $K_p = \langle d \rangle$, $(d, \varphi(n)) = 1$. **Viešasis parašams tikrinti skirtas raktas:** $K_v = \langle n, d \rangle$, $ed \equiv 1 \pmod{\varphi(n)}$. **Parašo sudarymas:** tekstas $x \in \mathbb{Z}_n$, jo parašas $y \equiv x^d \pmod{n}$. **Parašo tikrinimas:** parašas priimamas, jeigu $y^e \equiv x \pmod{n}$ arba jeigu x neatsiųstas, įvertinus $y^e \pmod{n}$ pagal prasmę.

RSA schema turi multiplikatyvumo savybę: jei $y_1 = sig(x_1|K_{v,A}), y_2 = sig(x_2|K_{v,A})$, tai $y = y_1y_2$ yra pranešimo $x = x_1x_2$ parašas. Taigi Z turi galimybę iš dviejų A pasirašytų pranešimų sudaryti trečiojo pranešimo parašą. To galima išvengti, pavyzdžiui, pasirašinėjant ne pačius pranešimus, bet jų vaizdus, gautus naudojant visiems schemos dalyviams žinomą injekciją $R : \mathcal{M} \to \mathcal{M}_S$, jei tik ši funkcija parinkta taip, kad neturėtų multiplikatyvumo savybės, t. y. $R(x_1x_2) \neq R(x_1) \cdot R(x_2)$.

Tam tikruose kriptografiniuose protokoluose reikia, kad subjektas sukurtų skaitmeninį parašą, nematydamas paties teksto. Tai tarsi pasirašymas ant užklijuoto voko, kuriame įdėta kalkė ir pasirašymui parengtas dokumentas. Kalkė perkelia parašą nuo voko ant paties dokumento. Tokie parašai vadinami aklais parašais. Jų prireikia finansinėje kriptografijoje bei elektroninių rinkimų sistemose. Pavyzdžiui, rinkiminė komisija savo parašu turi patvirtinti, kad skaitmeninis balsavimo biuletenis yra galiojantis, tačiau neturi sužinoti, už ką rinkėjas balsavo.

Sukurti aklą parašą su RSA sistema labai paprasta. Tarkime, B nori, kad A sukurtų galiojantį teksto m RSA parašą, bet nepamatytų paties teksto. Tegu A raktai yra $K_{v,A} = \langle e_A, n_A \rangle$ ir $K_{p,A} = \langle d_A \rangle$. Parinkusi skaičių $r, (r, n) = 1$, B apskaičiuoja

$$x \equiv r^{e_A} m \pmod{n_A}$$

ir nusiunčia A pasirašyti. A pasirašo ir atsiunčia B parašą

$$z = sig(x|K_{p,A}) \equiv x^{d_A} \pmod{n_A}.$$

Dabar B skaičiuoja

$$y \equiv r^{-1}z \equiv r^{-1}x^{d_A} \equiv r^{-1}(r^{e_A}m)^{d_A} \equiv m^{d_A} \pmod{n_A}.$$

Taigi $y = sig(m|K_{p,A})$ yra galiojantis teksto m parašas.

21.2. Rabino skaitmeninis parašas

Pranešimo x Rabino skaitmeninis parašas – tiesiog lyginio $y^2 \equiv x \,(mod\ n)$ sprendinys. Šiek tiek keblumų sudaro tai, kad ne su visais x šį lyginį galima išspręsti.

Rabino kriptosistemoje privatųjį raktą sudaro pirminių skaičių pora $K_p = \langle p, q\rangle$, o viešąjį – jų sandauga $K_v = \langle n\rangle, n = pq$. Šifruojamų pranešimų aibė yra $\mathcal{M} = \mathbb{Z}_n$. Pranešimo x šifravimas – tiesiog kėlimas kvadratu

$$C = e(x|K_v) \equiv x^2 \pmod{n}.$$

Galime Rabino kriptosistemą paversti skaitmeninių parašų schema, kurioje parašo tikrinimas yra toks pat kėlimas kvadratu. Tegu $x \in \mathbb{Z}_n$ yra pranešimas, kurį norime pasirašyti. Naudodamiesi privačiuoju raktu, galime pabandyti išspręsti lyginį

$$y^2 \equiv x \pmod{n}. \tag{104}$$

Jeigu tai pavyko, tai gautą sprendinį y galime siųsti kaip pranešimo x parašą: $y = sig(x|K_p)$. Parašas bus priimtas, jeigu $y^2 \pmod{n}$ ir x sutaps.

Tačiau yra vienas keblumas: (104) lyginys ne su visais x yra išsprendžiamas. Kaip elgtis tuo atveju, kai sprendinio nėra? Žinome, kad nustatyti, ar lyginys išsprendžiamas, naudojantis Legendre'o simboliais galima labai greitai. Jeigu šifruojame kokį nors tekstą, galima bandyti jį kiek pakaitalioti, pavyzdžiui, įterpiant daugiau intervalų, šitaip tikintis, kad galų gale gausime x, kuriam sprendinys egzistuoja. Iš tikrųjų, ilgai vargti tikrai nereikės. Tačiau jeigu tokia empirinė paieška yra nepriimtina, tai reikia sugalvoti kokią nors funkciją $R : \mathbb{Z}_n \to Q_n$, čia $Q_n \subset \mathbb{Z}_n$ yra poaibis tų elementų x, su kuriais (104) lyginys turi sprendinį. Ši funkcija turi būti vieša. Tada pranešimo x skaitmeniniu parašu laikytume lyginio

$$y^2 \equiv R(x) \pmod{n}$$

sprendinį. Tikrinimui tektų pateikti porą $\langle x, y\rangle$, o tikrinant pirmiausia būtų apskaičiuojama reikšmė $R(x)$ ir tikrinama, ar lyginys

$$y^2 \equiv R(x) \pmod{n}$$

yra teisingas. Jeigu teisingas – parašas priimamas.

Kriptografai yra sukonstravę tokių funkcijų R. Tačiau tai padaryti nėra paprasta. Viena iš būtinų sąlygų tokiai funkcijai: lygtį $R(x) = r$ su žinomu r turi būti sunku spręsti. Jeigu būtų lengva, piktavalis Z galėtų siuntinėti prasmingus ir neprasmingus tekstus su mūsų parašais. Iš tikrųjų, parinkime bet kokį y ir raskime lygties $R(x) = y^2$ sprendinį x. Tada $y = sig(x|K_p)$ – parašas suklastotas!

Rabino skaitmeninio parašo schema
Pasirašomi tekstai – skaičiai $x \in \mathbb{Z}_n$, su kuriais lyginys $y^2 \equiv x \pmod{n}$ turi sprendinį; $n = pq$, p, q – du dideli pirminiai skaičiai.
Privatusis parašams sudaryti skirtas raktas: $K_p = \langle p, q \rangle$.
Viešasis parašams tikrinti skirtas raktas: $K_v = \langle n \rangle$.
Parašo sudarymas: teksto $x \in \mathbb{Z}_n$ parašas – lyginio $y^2 \equiv x \pmod{n}$ sprendinys.
Parašo tikrinimas: parašas priimamas, jeigu $y^2 \equiv x \pmod{n}$.

21.3. ElGamalio skaitmeninio parašo schema

Geras ElGamalio schemos idėjas ne vieną kartą panaudojo kitų skaitmeninio parašo sistemų kūrėjai. Schema gera ir tuo, kad ją galima įdiegti naudojant įvairias baigtines ciklines grupes.

Šios schemos saugumas priklauso nuo atitinkamo diskrečiojo logaritmo uždavinio sudėtingumo.

Raktų parinkimas. Tegu p – pirminis skaičius, pakankamai didelis, kad diskrečiojo logaritmo uždavinį multiplikatyvioje $\mathbb{F}_p$ grupėje, t. y. grupėje $\mathbb{F}_p^* = \{1, 2, \ldots, p-1\}$, būtų sunku spręsti. Tegu α yra šią grupę generuojantis elementas (primityvioji vieneto šaknis). Parinkę $a \in \mathbb{Z}_{p-1}$, skaičiuojame $\beta \equiv \alpha^a \pmod{p}$ ir sudarome viešąjį raktą $K_v = \langle p, \alpha, \beta \rangle$. Privatusis raktas sudarytas tik iš vienos komponentės: $K_p = \langle a \rangle$.

Parašų sudarymas. Pranešimai, kuriuos galima pasirašyti – aibės $\mathcal{M} = \mathbb{F}_p^*$ skaičiai, parašų aibė – $\mathcal{P} = \mathbb{F}_p^* \times \mathbb{Z}_{p-1}$. Pranešimo x parašui sudaryti pasirenkame atsitiktinį (slaptą) skaičių $k \in \mathbb{Z}_{p-1}^*$ (taigi $(k, p-1) = 1$) ir skaičiuojame:

$$\gamma \equiv \alpha^k \pmod{p}, \quad \delta \equiv (x - a\gamma)k^{-1} \pmod{(p-1)}.$$

Ši skaičių pora ir yra pranešimo x parašas: $sig(x|K_p) = \langle \gamma, \delta \rangle$.

Matome, kad pranešimo parašas priklauso nuo to, kokį atsitiktinį parametrą k parenkame. Taigi tam pačiam pranešimui gali būti sudaryta daug skirtingų, bet galiojančių parašų.

Parašų tikrinimas. Skaičių pora $y = \langle \gamma, \delta \rangle$ laikoma galiojančiu pranešimo x parašu tada ir tik tada, kai

$$\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod{p}. \tag{105}$$

Iš tikrųjų, jei y sudarytas tinkamai, tai $x \equiv a\gamma + k\delta \pmod{(p-1)}$, taigi

$$\beta^\gamma \gamma^\delta \equiv \alpha^{a\gamma + k\delta} \equiv \alpha^x \pmod{p}.$$

ElGamalio skaitmeninio parašo schema
Pasirašomų tekstų aibė $\mathcal{M} = \mathbb{F}_p^*$, čia p – didelis pirminis skaičius; parašų aibė $\mathcal{P} = \mathbb{F}_p^* \times \mathbb{Z}_{p-1}$.
Privatusis raktas: $K_p = \langle a \rangle, a \in \mathbb{Z}_{p-1}$.
Viešasis raktas: $K_v = \langle p, \alpha, \beta \rangle$, čia α – generuojantis elementas, $\beta \equiv \alpha^a \pmod p$.
Parašo sudarymas: pasirenkamas atsitiktinis $k, (k, p-1) = 1$ ir skaičiuojama: $\gamma \equiv \alpha^k \pmod p$, $\delta \equiv (x - a\gamma)k^{-1} \pmod{(p-1)}$, $\langle \gamma, \delta \rangle = sig(x\|K_p)$.
Parašo tikrinimas: parašas priimamas tada ir tik tada, kai $\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p$.

Kriptoanalitikas, norėdamas pranešimui x sudaryti be privačiojo rakto galiojantį parašą, turi parinkti γ, δ, kad būtų teisingas (105) lyginys. Galima pasirinkti, pavyzdžiui, γ ir ieškoti tinkamo skaičiaus δ. Tačiau

$$\gamma^\delta \equiv \alpha^x \beta^{-\gamma} \pmod p, \quad \delta \equiv \log_\gamma(\alpha^x \beta^{-\gamma}) \pmod{(p-1)},$$

t. y. tenka spręsti diskrečiojo logaritmo uždavinį.

Jeigu pasirenkamas δ ir iš (105) ieškoma γ, tai problema yra dar sudėtingesnė. Jeigu pasirinksime abi parašo komponentes γ, δ ir ieškosime pranešimo x, kuriam $y = \langle \gamma, \delta \rangle$ būtų tinkamas parašas, tai vėl teks ieškoti diskrečiojo logaritmo.

Tačiau vis dėlto galima parinkti (105) lyginį tenkinančius skaičius renkant juos kartu. Vienas būdas yra toks.

Pasirinkime skaičius i, j, tenkinančius sąlygą $0 \le i, j \le p-2, (j, p-1) = 1$. Dabar suskaičiuokime:

$$\gamma \equiv \alpha^i \beta^j \pmod p, \quad \delta \equiv -\gamma j^{-1} \pmod{p-1}, \quad x \equiv -\gamma i j^{-1} \pmod{p-1}.$$

Nesunku įsitikinti, kad tada $y = \langle \gamma, \delta \rangle$ yra galiojantis x parašas, t. y. skaičiai x, γ, δ tenkina (105) lyginį.

Skaičių k, kurį naudojame parašui sudaryti, geriausia, baigus skaičiuoti, iškart ištrinti. Jeigu jis taps žinomas kriptoanalitikui, tai jis, naudodamasis pranešimu x ir jo parašu $\langle \gamma, \delta \rangle$, nesunkiai suras slaptąjį skaičių a:

$$a \equiv (x - k\delta)\gamma^{-1} \pmod{p-1}.$$

Toks pat pavojus kyla, jeigu du skirtingus pranešimus x_1 ir x_2 pasirašėme naudodami tą patį k: $\langle \gamma, \delta_1 \rangle = sig(x_1\|K_p)$, $\langle \gamma, \delta_2 \rangle = sig(x_2\|K_p)$. Tada iš lyginių

$$\beta^\gamma \gamma^{\delta_1} \equiv \alpha^{x_1} \pmod p, \quad \beta^\gamma \gamma^{\delta_2} \equiv \alpha^{x_2} \pmod p$$

gauname $\alpha^{x_1 - x_2} \equiv \gamma^{\delta_1 - \delta_2} \pmod p$. Kadangi $\gamma \equiv \alpha^k \pmod p$, tai nežinomam parametrui k rasti gauname lyginį $x_1 - x_2 \equiv k(\delta_1 - \delta_2) \pmod{p-1}$.

Tegu $d = (\delta_1 - \delta_2, p-1)$, suprastinę iš d, skaičiui k rasti gauname lyginį $x' \equiv k\delta' \pmod{p'}$, čia $x' = (x_1 - x_2)/d, \delta' = (\delta_1 - \delta_2)/d, p' = (p-1)/d$. Jeigu $d > 1$, tai gautąjį lyginį tenkina ne vienintelis skaičius $k, 1 \le k \le p-1$. Tačiau tikrąjį visada galima atsirinkti naudojantis sąlyga $\gamma \equiv \alpha^k \pmod{p}$.

Sužinojus k, jau aptartu būdu galima surasti ir a.

21.4. Schnorro skaitmeninio parašo schema

Tai variacija ElGamalio skaitmeninio parašo tema. Kriptografinėje literatūroje, o taip pat ir taikymuose galima surasti ir variacijų Schnorro skaitmeninio parašo tema.

Šios schemos saugumas taip pat remiasi diskrečiojo logaritmo uždavinio sudėtingumu.

Raktų sudarymas. Kaip ir ElGamalio schemoje, pirmiausia reikia parinkti didelį pirminį skaičių p. Toliau – surasti kokį nors pakankamai didelį pirminį q, kuris dalija $p-1$. Gali pasitaikyti, kad $p-1$ skaidinys bus sudarytas tik iš mažų pirminių daugiklių. Toks p Schnorro schemos kūrimui netiktų. Geriausia, ko galime tikėtis: $p - 1 = 2q$, čia q yra pirminis. Tokio pavidalo pirminiai skaičiai yra gera žaliava įvairioms kriptografinėms schemoms konstruoti. Jie vadinami saugiais pirminiais. Jų, žinoma, nėra daug, tačiau daug ir nereikia. Štai pirmasis saugių pirminių, didesnių už milijoną, penketukas:

$$1000919,\ 1001003,\ 1001159,\ 1001387,\ 1001447.$$

Radę pakankamai didelį q, suraskime q-osios eilės kūno $\mathbb{F}_p$ elementą α, t. y. tokį, kuriam $\alpha^q \equiv 1 \pmod{p}$ ir $\alpha^j \not\equiv 1 \pmod{p}$, jei $0 < j < q$. Pranešimų aibė $\mathcal{M} = \mathbb{F}_q$. Pasirinkę dar vieną skaičių $e, 0 < e < q$, jau galime sudaryti raktus:

$$K_p = \langle a \rangle, \quad K_v = \langle p, q, g, \beta \rangle, \quad \beta \equiv \alpha^{-a} \pmod{p}.$$

Parašo sudarymas ir tikrinimas. Parinkę skaičių $0 \le r < q-1$, skaičiuojame:

$$\gamma \equiv \alpha^r \pmod{p}, \quad \delta \equiv r + ax \pmod{q}, \quad sig(x|K_p) = \langle \gamma, \delta \rangle.$$

Parašas bus priimtas tada ir tik tada, kai teisingas lyginys

$$\alpha^\delta \beta^x \equiv \gamma \pmod{p}.$$

Schnorro skaitmeninio parašo schema
Pranešimų aibė $\mathcal{M} = \mathbb{F}_q$, čia q yra pirminis skaičiaus $p-1$ daliklis; p irgi yra pirminis.
Privatusis raktas: $K_p = \langle a \rangle, 0 < a < q-1$.
Viešasis raktas: $K_v = \langle p, q, \alpha, \beta \rangle$, $\alpha \in \mathbb{F}_p$ yra q-osios eilės elementas, $\beta \equiv \alpha^{-a} \pmod{p}$.
Parašo sudarymas: pasirašymui skirtas tekstas yra x; parinkus skaičių $0 \leq r < q-1$, skaičiuojama: $\gamma \equiv \alpha^r \pmod{p}, \delta \equiv r + ax \pmod{q}, sig(x\|K_p) = \langle \gamma, \delta \rangle$.
Parašo tikrinimas: pranešimo x parašas $sig(x\|K_p) = \langle \gamma, \delta \rangle$ priimamas tada ir tik tada, kai $\alpha^\delta \beta^x \equiv \gamma \pmod{p}$.

Nesunku įsitikinti, kad pagal taisykles sudarytas parašas bus visada priimtas. Sudarant parašą, reikia apskaičiuoti dydžius γ ir δ. Pirmasis dydis nesusijęs su pranešimu, todėl jį galima apskaičiuoti ir išsaugoti iš anksto. Antrajam dydžiui skaičiuoti reikia visai nedaug veiksmų – vienos daugybos ir sudėties moduliu q. Taigi parašui sudaryti nereikia daug skaičiavimo išteklių. Todėl tokią schemą galima diegti autentifikavimui naudojamose kortelėse su nedidelio galingumo mikroprocesoriais.

21.5. DSA

Šią parašo schemą savo reikšme galima palyginti su DES. DSA (Digital Signature Algorithm) yra pirmoji kriptografijos istorijoje vyriausybiniu lygiu pripažinta skaitmeninių parašų schema.

1991 metais JAV Nacionalinis standartų ir technologijos institutas pasiūlė skaitmeninių parašų schemą, kuri JAV buvo pripažinta skaitmeninių parašų standartu (DSA – Digital Signature Algorithm). Tai pirmoji vyriausybiniu lygiu pripažinta skaitmeninių parašų schema. Teoriniu požiūriu DSA yra ElGamalio skaitmeninio parašo variantas. Vienas iš privalumų, lyginant DSA su ElGamalio schema – DSA parašai yra trumpesni. Ankstesniame skyrelyje matėme, kad ElGamalio skaitmeniniai parašai gali būti net dvigubai ilgesni už patį pranešimą.

Raktų sudarymas. Parinkime du pirminius skaičius p, q, kad q dalytų $p-1$, t. y. $q|p-1$. Kad kriptosistema būtų saugi, diskrečiojo logaritmo uždavinys moduliu p turi būti sunkus. Rekomenduojama naudoti apie 512 bitų skaičių p ir apie 160 bitų skaičių q.

Tegu $\alpha \in \mathbb{F}_p^*$ yra q-osios eilės elementas. Parinkę $a \in \mathbb{F}_q$, apskaičiuojame $\beta \equiv \alpha^a \pmod{p}$, sudarome viešąjį raktą K_v ir slaptąjį K_p:

$$K_v = \langle p, q, \alpha, \beta \rangle, \quad K_p = \langle a \rangle.$$

Parašo sudarymas. Pranešimų, kuriuos bus galima pasirašyti, aibė yra $\mathcal{M} = \mathbb{F}_p^*$, parašų aibė – $\mathcal{P} = \mathbb{F}_q \times \mathbb{F}_q$. Pranešimo $x \in \mathcal{M}$ parašas sudaromas taip. Parinkę atsitiktinį $k \in \mathbb{F}_q^*$, skaičiuojame:

$$\gamma \equiv \alpha^k \pmod{p} \pmod{q}, \quad \delta \equiv (x + a\gamma)k^{-1} \pmod{q}, \quad sig(x|K_p) = \langle \gamma, \delta \rangle.$$

Be to, reikia pasirūpinti, kad būtų patenkinta sąlyga $q \not| \delta$. Jeigu $q|\delta$, reikia pasirinkti naują k ir vėl skaičiuoti parašą.

Parašo tikrinimas. Tegu reikia patikrinti, ar $y = \langle \gamma, \delta \rangle$ yra pranešimo x parašas. Iš pradžių skaičiuojame $e_1 \equiv x\delta^{-1} \pmod{q}, e_2 \equiv \gamma\delta^{-1} \pmod{q}$. Parašas pripažįstamas tada ir tik tada, kai

$$\alpha^{e_1}\beta^{e_2} \pmod{p} \equiv \gamma \pmod{q}. \tag{106}$$

Iš tikrųjų, jei parašas sudarytas teisingai, tai tikrindami gausime

$$\alpha^{e_1}\beta^{e_2} \equiv \alpha^{\delta^{-1}(x+a\gamma)} \pmod{p}.$$

Tačiau iš parašo sudarymo lygybės gauname $\delta^{-1}(x + a\gamma) \equiv k \pmod{q}$, taigi

$$\alpha^{e_1}\beta^{e_2} \equiv \alpha^k \pmod{p}.$$

Tačiau dešinioji pusė moduliu q lygi γ, taigi (106) lygybė teisinga.

Jeigu pasirinktasis pirminis skaičius p užrašomas 512 bitų ilgio žodžiais, o q – 160, tai 512 bitų ilgio teksto skaitmeninis parašas yra sudarytas iš maždaug $2 \times 160 = 320$ bitų.

Pavyzdys. Pasirinksime nedidelius p, q. Skaičius $p = 10007$ yra saugus pirminis, t. y. $p = 2q + 1$, čia $q = 5003$ irgi yra pirminis skaičius. Dabar reikia parinkti q-osios eilės elementą α. Iš pradžių suraskime generuojantį elementą moduliu p. Jų yra daug, pavyzdžiui, $\rho = 51$ yra vienas jų. Taigi galime imti $\alpha \equiv 51^2 \pmod{p}$, t. y. $\alpha = 2601$ yra q eilės elementas. Dabar parinkime a, pavyzdžiui, $a = 300$, tada $\beta \equiv \alpha^a \pmod{p}, \beta = 2774$. Taigi

$$K_v = \langle p, q, \alpha, \beta \rangle = \langle 10007, 5003, 51, 2774 \rangle, \quad K_p = \langle 300 \rangle.$$

Sudarykime pranešimo $x = 1111$ skaitmeninį parašą. Pasirinkime $k = 44$, $k^{-1} \equiv 1933 \pmod{q}$. Tada

$$\gamma \equiv 51^{44} \pmod{p}, \quad \gamma \equiv 8661 \pmod{p}, \quad \gamma \equiv 3658 \pmod{q},$$
$$\delta \equiv (1111 + 300 \cdot 3658) \cdot 1933 \pmod{q}, \quad \delta = 3476.$$

Sąlyga $(\delta, q) = 1$ patenkinta, taigi $sig(1111|K_p) = \langle 3658, 3476 \rangle$.

Tikrindami parašą, pirmiausia surandame $\delta^{-1} \equiv 2051 \pmod{q}$. Dabar skaičiuojame:

$$e_1 \equiv 1111 \cdot 2051 \pmod{q}, \quad e_1 = 2296, \quad e_2 \equiv 3658 \cdot 2051 \pmod{q}, \quad e_2 = 3061.$$

Toliau tikriname: $\alpha^{e_1}\beta^{e_2} \equiv 8661, \quad 8661 \equiv 3858 \pmod{q}$, parašas priimamas.

DSA

Pranešimų aibė $\mathcal{M} = \mathbb{F}_p^*$, parašų aibė – $\mathcal{P} = \mathbb{F}_q \times \mathbb{F}_q$, čia q yra pirminis $p-1$ daliklis.

Privatusis raktas: $K_p = \langle a \rangle,\ 0 < a < q-1$.

Viešasis raktas: $K_v = \langle p, q, \alpha, \beta \rangle$, $\alpha \in \mathbb{F}_p$ yra q-osios eilės elementas, $\beta \equiv \alpha^a \pmod{p}$.

Parašo sudarymas: pranešimui pasirašyti parenkamas skaičius $k \in \mathbb{F}_q^*$ ir skaičiuojama: $sig(x|K_p) = \langle \gamma, \delta \rangle$,

$$\gamma \equiv \alpha^k \pmod{p} \pmod{q},\ \delta \equiv (x + a\gamma)k^{-1} \pmod{q}.$$

Turi būti patenkinta sąlyga $(\delta, q) = 1$.

Parašo tikrinimas: parašas pripažįstamas tada ir tik tada, kai

$$\alpha^{e_1}\beta^{e_2} \pmod{p} \equiv \gamma \pmod{q},$$
$$e_1 \equiv x\delta^{-1} \pmod{q}, e_2 \equiv \gamma\delta^{-1} \pmod{q}.$$

21.6. Nepaneigiami skaitmeniniai parašai

Tai schema, kurioje skaitmeninis B parašas tikrinamas jai „dalyvaujant“. Tačiau asmens dalyvavimas kriptografiniame protokole tiesiogine prasme yra beveidis: nepažiūrėsi į akis ir nenuspręsi, teisingai elgiasi ar sukčiauja. Vadinasi, reikalingi tam tikri matematiniai saugikliai...

Kartais pageidautina, kad, atliekant skaitmeninio parašo tikrinimą, dalyvautų ir parašo autorius. Pavyzdžiui, jeigu asmuo kreipiasi į banką savo sąskaitos tvarkymo klausimu, saugiau, jeigu informacija tikrinama jam dalyvaujant. Tai galima atlikti naudojantis klausimų-atsakymų protokolais (*challenge-and-response protocol*). Tokioje situacijoje svarbu, kad abi pusės (tikrintojas ir parašo autorius) laikytųsi protokolo taisyklių. Tačiau šiame tikrinimo procese abiejų pusių padėtys skirtingos. Tikrintojas remiasi tik vieša informacija, jei jis netinkamai atlieka protokolo veiksmus, autorius gali pareikalauti jį pakeisti ir pasiekti, kad jo parašas bus pripažintas. Parašo autorius atlieka veiksmus naudodamasis tik jam prieinama (slapta) informacija, taigi atlikdamas protokolo veiksmus, ne taip, kaip numatyta, jis gali pasiekti, kad jo parašas bus pripažintas negaliojančiu. Pavyzdžiui, šitaip jis gali neprisiimti kokių nors jo pasirašytų įsipareigojimų.

Chaum ir van Antverpeno nepaneigiamo parašo schemoje yra galimybė nustatyti, kad parašo autorius elgiasi protokolo vykdymo metu netinkamai, taigi bando paneigti savo parašą. Ši schema paskelbta 1989 metais.

Raktų parinkimas. Parinkime p – pakankamai didelį pirminį skaičių, kad diskrečiojo logaritmo uždavinys grupėje $\mathbb{F}_p^*$ būtų sunkus. Tegu q – pirminis skaičiaus $p-1$ daliklis, o $\alpha \in \mathbb{F}_p^*$ – q-osios eilės elementas. Parinkime skaičių $a, 1 \le a \le q-1$, ir suskaičiuokime $\beta \equiv \alpha^a \pmod p$. Dabar jau galime sudaryti raktus:

$$K_p = \langle a \rangle, \quad K_v = \langle p, \alpha, \beta \rangle.$$

Parašų sudarymas. Pranešimus reikia koduoti aibės

$$\mathcal{G} = \{\alpha^m : m = 0, \ldots, q-1\}$$

skaičiais. Parašai – taip pat šios aibės elementai, taigi $\mathcal{M} = \mathcal{P} = \mathcal{G}$. Jei $x \in \mathcal{M}$, tai $sig(x|K_p) \equiv x^a \pmod p$.

Parašo tikrinimas. Tarkime, A turi patikrinti, ar y yra B parašas. Tikrinimo protokolas vykdomas taip:

1. *A parenka atsitiktinius skaičius* $e_1, e_2 \in \mathbb{F}_q^*$, *skaičiuoja* $c \equiv y^{e_1}\beta^{e_2} \pmod p$ *ir siunčia B;*

2. *B skaičiuoja* $d \equiv c^{a^{-1}(\mathrm{mod} q)} \pmod p$ *ir siunčia A;*

3. *parašas priimamas tada ir tik tada, kai* $d \equiv x^{e_1}\alpha^{e_2} \pmod p$.

Jeigu parašas yra tikras, t. y. $y \equiv x^a \pmod p$, tai nesunku įsitikinti, kad jis bus priimtas. Iš tikrųjų,

$$d \equiv c^{a^{-1}(\mathrm{mod} q)} \equiv y^{e_1 a^{-1}}\beta^{e_2 a^{-1}} \equiv x^{e_1 a a^{-1}}\alpha^{e_2 a a^{-1}} \equiv x^{e_1}\alpha^{e_2} \pmod p.$$

Tačiau gali atsitikti, kad bus už teisingą priimtas ir negaliojantis parašas, t. y. toks, kuriam $y \not\equiv x^a \pmod p$. Tačiau tokia galimybė mažai tikėtina.

142 teorema. *Jei* $y \not\equiv x^a \pmod p$, *tai tikimybė, kad* y *bus pripažintas pranešimo* x *parašu, lygi* $1/q$.

Parašo tikrinimo protokolas gali duoti neigiamą atsakymą dviem atvejais: kai parašas iš tikrųjų netikras, t. y. $y \not\equiv x^a \pmod p$, arba B nesilaiko protokolo. Žinoma, ir tikrintojas A gali nesilaikyti protokolo, tačiau jis skaičiuodamas nesinaudoja jokia slapta informacija, taigi jei jis netinkamai elgiasi, gali būti pakeistas kitu.

Šioje parašo schemoje yra galimybė nustatyti, kada parašas yra tikrai netikras, o kada B mėgina (nesvarbu, ar sąmoningai ar dėl skaičiavimo klaidų) to parašo atsisakyti. Jeigu B siekia, kad tikrinimo protokolas duotų neigiamą rezultatą, tai 2-ajame žingsnyje ji pasirenka d „pažvelgusi į lubas“, t. y. bet kokį. Jeigu tikrinimo protokolas duoda neigiamą atsakymą, protokolas dar kartą kartojamas:

1. *jeigu* $d \not\equiv x^{e_1}\alpha^{e_2} \pmod p$ *A parenka atsitiktinius skaičius* $f_1, f_2 \in \mathbb{F}_q^*$, *skaičiuoja* $C \equiv y^{f_1}\beta^{f_2} \pmod p$ *ir siunčia B;*

2. *B skaičiuoja* $D \equiv C^{a^{-1}(mod\ q)} \pmod p$ *(arba vėl pasirenka D „nuo lubų“) ir siunčia A;*

3. *jeigu* $D \equiv x^{f_1}\alpha^{f_2} \pmod p$, *A parašą pripažįsta;*

4. *jeigu* $D \not\equiv x^{f_1}\alpha^{f_2} \pmod p$, *A laiko parašą klastote tada ir tik tada, kai*

$$(d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \pmod p,$$

priešingu atveju A apkaltina B protokolo nesilaikymu arba bandymu suklastoti svetimą parašą.

143 teorema. *Jei* $y \not\equiv x^a \pmod p$, *tačiau ir A, ir B laikosi protokolo, tai*

$$(d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \pmod p.$$

Įrodymas.
Apskaičiuokime dydį $(d\alpha^{-e_2})^{f_1} \pmod p$. Kadangi B laikosi protokolo, tai

$$(d\alpha^{-e_2})^{f_1} \equiv (c^{a^{-1}}\alpha^{-e_2})^{f_1} \equiv (y^{e_1a^{-1}}\beta^{e_2a^{-1}}\alpha^{-e_2})^{f_1} \equiv y^{e_1f_1a^{-1}} \pmod p.$$

Skaičiuodami $(D\alpha^{-f_2})^{e_1} \pmod p$, gautume lygiai tą patį. Taigi lyginys yra teisingas. Galbūt kiek keista, kad skaičiuodami niekur nepanaudojome sąlygos $y \not\equiv x^a \pmod p$. Jeigu ji nebūtų patenkinta, t. y. jeigu y tikrai būtų parašas, tai iki šio lyginio tikrinimo nė neprieitume – parašas būtų pripažintas galiojančiu jau anksčiau.

Ši teorema rodo, kad B negali būti apkaltinta nekaltai. Kokia gi tikimybė B apgauti A – įrodyti, kad jos galiojantis parašas yra klastotė, t. y. atsisakyti savo parašo? Tam ji turėtų atsiųsti ne pagal protokolo taisykles suskaičiuotus d, D, kad lyginys būtų vis dėlto teisingas. Tikimybė, kad tai pavyks yra nedidelė.

144 teorema. *Jei* $y \equiv x^a \pmod p$ *ir B parenka atsitiktinius* d, D, *kad*

$$d \not\equiv x^{e_1}\alpha^{e_2} \pmod p, \quad D \not\equiv x^{f_1}\alpha^{f_2} \pmod p,$$

tai tikimybė, kad

$$(d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \pmod p,$$

lygi $1/q$.

Taigi galimybė atsisakyti savo parašo yra menka.

21.7. Slaptieji kanalai skaitmeninių parašų schemose

Skaitmeninis parašas gali kartais būti ir šifras. Tuoj sužinosite, kaip tai padaryti.

Vienose skaitmeninio parašo schemose teksto parašas apibrėžtas vienareikšmiškai (pavyzdžiui, RSA). Kitose parašas priklauso nuo papildomai parenkamo dydžio. Parašas – tai tam tikri duomenys. Tikrinant parašą, su pranešimu ir tais duomenimis atliekami tam tikri skaičiavimai. Priklausomai nuo jų rezultatų parašas priimamas arba atmetamas. Tačiau tie duomenys gali būti kito pranešimo slaptavietė! Taigi skaitmeniniu parašu galima pasinaudoti kaip slaptu kanalu informacijai perduoti. Tokį slaptą kanalą galima įrengti beveik kiekvienoje skaitmeninio parašo schemoje, kurioje parašas nėra vienareikšmiškai apibrėžtas. Panagrinėkime, kaip tai galima padaryti su ElGamalio skaitmeninių parašų schema.

ElGamalio schemos raktai

$$K_p = \langle a \rangle, \quad K_v = \langle p, \alpha, \beta \rangle, \quad \beta \equiv \alpha^a \pmod{p},$$

čia $\alpha \in \mathbb{F}_p^*$ yra generuojantis elementas. Privatusis raktas sudarytas tik iš vienos komponentės: $K_p = \langle a \rangle$.

Pranešimo x parašas $sig(x|K_p) = \langle \gamma, \delta \rangle$ sudaromas pasirinkus skaičių $k \in \mathbb{Z}_{p-1}^*$ ir apskaičiavus

$$\gamma \equiv \alpha^k \pmod{p}, \quad \delta \equiv (x - a\gamma)k^{-1} \pmod{(p-1)}.$$

Parašas pripažįstamas, jeigu teisingas lyginys

$$\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod{p}. \tag{107}$$

Dabar tarkime, kad A veiksmai yra kontroliuojami ir jis negali pasiųsti B jokio nors menkiausią įtarimą keliančio pranešimo. Tarkime, kad šiek tiek anksčiau A sugebėjo pranešti B savo ElGamalio skaitmeninio parašo raktą $K_p = \langle a \rangle$. Norėdamas perduoti B slaptą pranešimą x, A ketina pasielgti taip: pasirašyti kokį nors nekaltą pranešimą x^* ir paskelbti jį kartu su parašu $y = sig(x^*|K_p) = \langle \gamma, \delta \rangle$. Parašas tenkins tikrinimo sąlygą, todėl nekels įtarimo jokiems kontrolieriams. Tačiau čia ir slypi gudrybė – paraše glūdės slaptasis pranešimas x, kurį B galės atskleisti žinodama $K_p = \langle a \rangle$.

Taigi A ketina sudaryti x^* parašą. Tarkime, $(x, p-1) = 1$ (jeigu taip nėra, galima x šiek tiek pakeisti). Tada parašą galima kurti su $k = x$. Jei

$$\gamma \equiv \alpha^x \pmod{p-1}, \quad \delta \equiv (x^* - a\gamma)x^{-1} \pmod{p-1},$$

tai $sig(x^*|K_p) = \langle \gamma, \delta. \rangle$. Tačiau žinodama pranešimą, parašą ir raktą, B gali surasti jai skirtą žinutę x:

$$x \equiv (x^* - a\gamma)\delta^{-1} \pmod{p-1}.$$

Tiesa, B gali kilti šiokių tokių sunkumų, jei δ nėra tarpusavyje pirminis su $p-1$. Tada δ^{-1} neegzistuos. Tačiau tai nedideli sunkumai. Be to, jų išvengti gali padėti ir A, pasirūpindamas, kad būtų $(\delta, p-1) = 1$.

22 Maišos funkcijos

Skaitmeninių parašų schemos, kokias nagrinėjome, kone visiškai netinkamos praktiniam naudojimui! Iš tiesų, kiek reiktų atlikti sudėtingų skaičiavimų, kad pasirašytume, pavyzdžiui, 1Mgb dydžio tekstą! Todėl reikia ieškoti būdo, kaip tas nepraktiškas schemas paversti lengvai pritaikomomis. Išeitis gali būti paprasta: pasirašinėti reikia ne visus tekstus, bet fiksuoto ilgio santraukas. Taigi reikia tų santraukų sudarymo metodų.

Tarkime, tekstai, kuriuos reikia pasirašinėti, gali būti bet kokio ilgio dvejetainės abėcėlės žodžiai, o skaimeninio parašo schema sukuria tik n bitų ilgio žodžių parašus. Todėl reikalinga funkcija

$$h : \{0,1\}^* \to \{0,1\}^n,$$

sudaranti bet kokio pranešimo santrauką. Tokios funkcijos kriptografijoje vadinamos maišos funkcijomis (hash functions, angl.). Jeigu naudojama maišos funkcija, tai pranešimo x parašu laikome jo santraukos parašą $y = sig(h(x)|K_p)$. Gavęs porą $\langle x, y\rangle$, parašo tikrintojas pirmiausia suskaičiuoja $x' = h(x)$, o tada tikrina, ar y yra x' parašas. Parašo sudarymo ir tikrinimo veiksmų sąnaudos nepriklauso nuo pranešimo ilgio. Kadangi pranešimų yra daugiau negu santraukų, tai yra tekstų su vienodomis santraukomis. Tada šių tekstų skaitmeniniai parašai bus vienodi. Maišos funkcijų naudojimas, be abejonės, susilpnina skaitmeninio parašo schemos saugumą. Todėl, konstruojant maišos funkcijas reikia gerai apsvarstyti, ar parašo schema su šiomis funkcijomis išlieka praktiškai saugi.

22.1. Kokios maišos funkcijos yra saugios?

Kad maišos funkcija būtų saugi, ji turi tenkinti kelias sąlygas, kurias paprasta suformuluoti, bet sunku įgyvendinti.

Tarkime, $h : \mathcal{M} \to \mathcal{H}$ yra maišos funkcija, kurianti tekstų santraukas. Kokius reikalavimus ji turėtų tenkinti? Visų pirma, sudaryti santrauką turi būti nesudėtinga, t. y. funkcijos reikšmė $h(x)$ turi būti skaičiuojama efektyviu algoritmu. Tarkime, ir funkcijos pirmavaizdžio elementai gali būti randami greitai, t. y. yra efektyvus algoritmas duotam z, randantis lygties

$$h(x) = z, \quad z \in \mathcal{H},$$

sprendinį $x \in \mathcal{M}$. Tada kriptoanalitikas galėtų lengvai atlikti skaitmeninio parašo schemos su maišos funkcija ataką, visiškai neanalizuodamas parašo sudarymo algoritmų. Iš tiesų, išsaugojęs teksto ir jo parašo porą $\langle x, y\rangle, y = sig(h(x)|K_p)$, jis galėtų ieškoti lygties $h(u) = y$ sprendinių ir, radęs $u = x', x' \neq x$, pateikti porą $\langle x', y\rangle$ kaip naują tekstą su galiojančiu parašu. Kad tokios atakos tikimybė būtų maža, pirmavaizdžio elemento skaičiavimo uždavinys turi būti sudėtingas.

116 apibrėžimas. *Funkciją $h : \mathcal{M} \to \mathcal{H}$ vadinsime vienakrypte, jeigu jos reikšmės skaičiuojamos efektyviu algoritmu, o lygties $h(x) = z$ su žinomu z sprendiniams rasti efektyvaus algoritmo nėra.*

Efektyviais algoritmais paprastai vadiname polinominius algoritmus. Apibrėžimas aiškus, tačiau labai griežtas. Iš tiesų nežinome nei vienos funkcijos, kuri patenkintų jo reikalavimus! Tiesa, yra daug funkcijų, kurių reikšmės yra efektyviai skaičiuojamos, o pirmavaizdžio elementams rasti greitas būdas nežinomas. Šiandien nežinomas, o rytoj gal atsiras! Iš tiesų tos funkcijos, kurias vadiname vienakryptėmis tėra tik kandidatės į jas.

Taigi maišos funkcijos turėtų būti vienakryptės. Suformuluokime dar porą reikalavimų.

117 apibrėžimas. *Tegu $h : \mathcal{M} \to \mathcal{H}$ yra maišos funkcija. Jeigu $x, x^* \in \mathcal{M}, x \neq x^*$, bet $h(x) = h(x^*)$, tai šią porą vadinsime sutapimo pora, arba tiesiog sutapimu (collision, angl.). Funkciją vadinsime atsparia sutapimams, jeigu nėra efektyvaus algoritmo duotajam $x \in \mathcal{M}$, randančio $x^* \in \mathcal{M}, x^* \neq x$, kad $h(x) = h(x^*)$. Funkciją vadinsime labai atsparia sutapimams, jeigu nėra efektyvaus algoritmo, randančio kokią nors sutapimų porą.*

Angliškoje kriptografijos literatūroje funkcijos, atsparios sutapimams, vadinamos *weekly collision free*, o labai atsparios – *strongly collision free* funkcijomis.

Jeigu maišos funkcija nėra labai atspari sutapimams, tai galima tokia ją naudojančios skaitmeninio parašo schemos ataka. Tarkime, A sutinka pasirašyti sutartį s su tam tikrais įsipareigojimais. B norėtų, kad tie įsipareigojimai būtų griežtesni, todėl yra parengusi kitą sutarties variantą s^*. Tačiau A tikrai nesutiks jo pasirašyti. Tada B gali kiek padirbėti su pora s, s^*, nežymiai kaitaliodama tekstus, pavyzdžiui, įterpdama daugiau tuščių tarpų, ir skaičiuodama $h(s)$ ir $h(s^*)$. Jeigu maišos funkcija nėra labai atspari sutapimams, galbūt pavyks gauti $h(s) = h(s^*)$. Tada B gali pateikti pasirašymui tekstą s, o sprendžiant teisinius ginčus, pateikti s^* su galiojančiu A parašu.

Jeigu funkcija yra atspari sutapimams, tai ji nebūtinai yra vienakryptė. Tačiau jeigu ji nėra vienakryptė, tai sutapimui rasti yra efektyvus (tiesa, tikimybinis, t. y. priklausantis nuo sėkmės) būdas. Tokia yra teiginio, kurį tuoj suformuluosime ir įrodysime, prasmė.

145 teorema. *Tegu $h : \mathcal{M} \to \mathcal{H}$ yra maišos funkcija, pranešimų ir santraukų aibės yra baigtinės ir $|\mathcal{M}| \geq 2|\mathcal{H}|$. Jeigu yra efektyvus algoritmas lygties $h(x) = z$ sprendiniui x rasti, tai egzistuoja tikimybinis algoritmas, randantis sutapimą su tikimybe, ne mažesne už $1/2$.*

Įrodymas. Sąlyga $|\mathcal{M}| \geq 2|\mathcal{H}|$ išvertus į kasdienę kalbą reiškia, kad, užrašius pranešimą ir santrauką dvejetainės abėcėlės žodžiais, santrauka bus bent vienu bitu trumpesnė. Tikriausiai sutiksite, kad tai natūralus noras.

Sutapimų radimo algoritmą, kurį dabar nagrinėsime, net algoritmu nelabai tinka vadinti. Jis toks. Pažymėkime efektyviai randamą lygties $h(x) = z$

sprendinį $A(z)$. Sutapimo ieškosime taip: atsitiktinai parinkę $x \in \mathcal{M}$, skaičiuosime $y = h(x)$, o tada $x^* = A(y)$. Jeigu $x^* \neq x$, sutapimas rastas. Jeigu ne – veiksmus galime kartoti. Reikia įsitikinti, kad $P(x^* \neq x) \geq 1/2$.

Pažymėkime $\mathcal{M}_z = \{u : h(u) = z\}$. Skirtingiems z, z' aibės $\mathcal{M}_z, \mathcal{M}_{z'}$ arba nesikerta, arba sutampa. Šių aibių yra tiek, kiek gali būti skirtingų santraukų, pažymėkime šį skaičių t. Taigi

$$\mathcal{M}_i = \mathcal{M}_{z_i}, \quad \mathcal{M}_i \cap \mathcal{M}_j = \emptyset \ (i \neq j), \quad \bigcup_{i=1}^{t} \mathcal{M}_i = \mathcal{M}, \quad \frac{t}{|\mathcal{M}|} \leq \frac{1}{2}.$$

Kadangi, ieškant sutapimo, visi pranešimai pradiniame žingsnyje gali būti parinkti su vienodomis tikimybėmis, tai

$$\begin{aligned} P(x \neq x^*) &= \sum_{x \in \mathcal{M}} P(A(h(x)) \neq x) \frac{1}{|\mathcal{M}|} = \frac{1}{|\mathcal{M}|} \sum_{x \in \mathcal{M}} \frac{|\mathcal{M}_{h(x)}| - 1}{|\mathcal{M}_{h(x)}|} \\ &= \frac{1}{|\mathcal{M}|} \sum_{i=1}^{t} \sum_{x \in \mathcal{M}_i} \frac{|\mathcal{M}_i| - 1}{|\mathcal{M}_i|} = \frac{1}{|\mathcal{M}|} \sum_{i=1}^{t} (|\mathcal{M}_i| - 1) = 1 - \frac{t}{|\mathcal{M}|} \geq \frac{1}{2}. \end{aligned}$$

Jeigu maišos funkcijos reikšmių aibė turės nedaug elementų, tai daug tekstų turės tą pačią santrauką. Tada sutapimas gali pasitaikyti kriptoanalitikui kaip aklai vištai grūdas. Panagrinėkime, kokio ilgio turėtų būti santraukos, kad šis metodas neveiktų. Tiesa, kriptografijoje jis vadinamas ne aklos vištos metodu, bet gimtadienių ataka. Taip yra todėl, kad ataka siejama su žinomu elementarios tikimybių teorijos uždaviniu: kiek mažiausiai žmonių turi susirinkti, kad įvykio, jog atsiras gimusių tą pačią metų dieną, tikimybė būtų didesnė už 1/2? Beveik visi, bandantys atsakyti vadovaudamiesi „sveiku protu“, apsigauna. Atsakymas – 23.

Taigi panagrinėkime tokį sutapimo ieškojimo būdą: atsitiktinai pasirinkime skirtingus $x_1, x_2, \ldots, x_n$ ir apskaičiuokime $z_1 = h(x_1), z_2 = h(x_2), \ldots, z_n = h(x_n)$. Jeigu nors dvi reikšmės sutapo, sutapimas rastas.

Tegu iš viso santraukų yra m, t. y. $|\mathcal{H}| = m$. Padarykime prielaidą, kad toks santaukų skaičiavimas ekvivalentus atsitiktiniam jų pasirinkimui iš visos aibės su grąžinimu. Tada sutapimo tikimybę galime skaičiuoti spręsdami tokį uždavinį: kokia tikimybė, kad, iš urnos su m skirtingų rutulių paeiliui su grąžinimu traukdami n rutulių, nors vieną rutulį ištrauksime pakartotinai?

Lengviau suskaičiuoti tikimybę q, kad visi rutuliai bus skirtingi:

$$\begin{aligned} q &= \frac{m(m-1)\ldots(m-n+1)}{m^n} = \left(1 - \frac{1}{m}\right)\left(1 - \frac{2}{m}\right)\ldots\left(1 - \frac{n-1}{m}\right) \\ &\approx \exp\left\{-\frac{1}{m}\sum_{i=1}^{n-1} i\right\} \approx \exp\left\{-\frac{1}{2m}n^2\right\}. \end{aligned}$$

Jeigu sutapimo radimo tikimybė šiuo metodu yra ϵ, tai $q = 1 - \epsilon$ ir

$$m \approx \frac{-n^2}{2\ln(1-\epsilon)}.$$

Jeigu mūsų kriptoanalitikas gali atlikti sutapimų paiešką su milijardu pasirinktų tekstų, kiek turėtų būti santraukų, kad jo sėkmės tikimybė būtų tik $0,001$? Užrašant santraukas kaip dvejetainius žodžius, koks turėtų būti jų ilgis? Įstatę $n = 10^9, \epsilon = 10^{-3}$ ir kiek paskaičiavę gautume:

$$m \approx 5 \cdot 10^{20}, \quad \text{santraukų ilgis bitais} \geq 69.$$

Tokio ilgio santraukos yra pernelyg trumpos. Praktiškai dabar naudojamos $128, 256$ ir 512 bitų ilgio santraukos.

22.2. Blokiniai šifrai ir maišos funkcijos

Maišos funkcijas naudojame kaip pagalbinius skaitmeninių parašų schemų įrankius. O blokinius šifrus galime naudoti kaip pagalbinius įrankius maišos funkcijoms konstruoti!

Dažnai maišos funkcijos sudaromos taip, kad santrauka sukuriama atliekant iteracinio proceso žingsnius. Tarkime, turime tam tikrą funkciją

$$H : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n. \tag{108}$$

Tada, panaudoję ją, galime sukonstruoti maišos funkciją $h : \{0,1\}^* \to \mathcal{H}, \mathcal{H} = \{0,1\}^n$. Bet kokio ilgio tekstą $x \in \{0,1\}^*$ padalykime n ilgio žodžiais, jeigu reikia, papildydami paskutinįjį žodį iki reikiamo ilgio:

$$x = x_1 x_2 \ldots x_t, \quad x_i \in \{0,1\}^n.$$

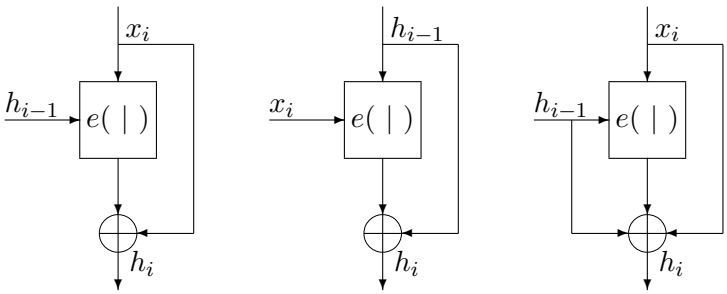
Tegu $h_0 \in \{0,1\}^n$ yra koks nors pradinis fiksuotas žodis. Apibrėžkime

$$h_i = H(h_{i-1}, x_i), \quad i = 1, 2, \ldots, t, \quad h(x) = h_t.$$

Šitaip sudarant santrauką „dalyvaus“ kiekvienas pranešimo simbolis. Tačiau kaip sudaryti (108) funkcijas? Panašios funkcijos kažkur matytos... Iš tiesų – jos naudojamos blokinėse kriptosistemose! Pavyzdžiui, jeigu blokinės kriptosistemos rakto ir bloko ilgiai sutampa, galime imti

$$H(u, v) = e(u|v),$$

t. y. šios funkcijos reikšmė – bloko u šifras panaudojant bloką v kaip raktą. Taigi blokinės kriptosistemos gali būti maišos funkcijų konstrukcijos pagrindas. Jas galima panaudoti ne vien tik minėtu būdu. Panagrinėkime kelis maišos funkcijų sudarymo naudojantis kriptosistemomis metodus.



Matyaso-Meyerio-Oseaso, Davieso-Meyerio ir Miyaguchi-Preneelio metodai maišos funkcijoms konstruoti iš blokinių kriptosistemų.

Brėžiniuose pavaizduotose schemose iteraciniai žingsniai apibrėžiami lygybėmis

$$\begin{aligned} h_i &= e(x_i|h_{i-1}) \oplus x_i \\ h_i &= e(h_{i-1}|x_i) \oplus h_{i-1} \\ h_i &= e(x_i|h_{i-1}) \oplus x_i \oplus h_{i-1}. \end{aligned}$$

Metodų, žinoma, yra ir daugiau. Pavyzdžiui, parinkę dydžiams $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2$ reikšmes iš $\mathbb{F}_2$, galime iteracinį žingsnį apibrėžti taip:

$$h_i = e(\alpha_1 x_i \oplus \alpha_2 h_{i-1}|\beta_1 x_i \oplus \beta_2 h_{i-1}) \oplus \gamma_1 x_i \oplus \gamma_2 h_{i-1}.$$

Ne su visais parametrų rinkiniais tokiu būdu gauname saugias maišos funkcijas.

22.3. SHA-1

Panagrinėjus praktiškai naudojamų maišos funkcijų konstrukcijas, susidaro įspūdis, kad jų kūrėjai veikiau kliovėsi savo intuicija nei kokiais nors objektyviais moksliškais kriterijais. Galite patys tuo įsitikinti.

Skaitmeninių parašų schemų naudojimas be saugių maišos funkcijų yra praktiškai neįmanomas. Nenuostabu, kad skaitmeninių parašų schemų kūrėjai susirūpino ir maišos funkcijų konstravimu. Vienas iš pirmųjų praktiniams taikymams skirtų maišos funkcijų kūrėjų – Ronaldas Rivestas. Jo idėjos panaudotos kuriant MD (Message Digest) maišos funkcijų šeimą.

Kita kriptografijoje gerai žinoma maišos funkcijų šeima – SHA (Secure Hash Algorithm) funkcijos. Funkcija SHA-1 buvo patvirtinta kaip standartinė skaitmeninio parašo DSA schemai. Dabar šios šeimos funkcijos žymimos nurodant, kiek bitų turi su funkcija sudaryta teksto santrauka: SHA-256, SHA-384 ir SHA-512.

Panagrinėkime maišos funkcijos SHA-1, sudarančios 160 bitų ilgio santraukas, veikimo principus.

Ši funkcija atlieka iteracinius 512 bitų ilgio blokų pertvarkius. Todėl pirmiausia tekstas, kurio santrauką reikia sudaryti, pailginamas (jeigu reikia), kad ilgis bitais būtų skaičiaus 512 kartotinis.

Santrauka formuojama penkiuose registruose A,B,C,D,E, kiekviename iš jų saugomi 32 bitai, taigi santraukos ilgis $5 \times 32 = 160$. Prieš pradedant darbą registrai užpildomi pradinėmis standartinėmis reikšmėmis. Pirmasis teksto 512 bitų ilgio blokas padalijamas į 16 žodžių po 32 bitus: $m[0], m[1], \ldots, m[15]$.

Su šiuo bloku bus atliekama 80 operacijų, kiekvienai operacijai reikalinga 32 bitų žodžių pora $K_t, W_t,\ t = 0, 1, \ldots 79$. Jos sudaromos taip:

$$K_t = \begin{cases} \left[2^{30}\sqrt{2}\right], & 0 \le t \le 19, \\ \left[2^{30}\sqrt{3}\right], & 20 \le t \le 39, \\ \left[2^{30}\sqrt{5}\right], & 40 \le t \le 59, \\ \left[2^{30}\sqrt{10}\right], & 60 \le t \le 79, \end{cases}$$

$$W_t = \begin{cases} m[t], & 0 \le t \le 15, \\ (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) \leftarrow\!\!\shortmid 1, & 15 < t < 80, \end{cases}$$

čia ir kitur $\leftarrow\!\!\shortmid r$ reiškia ciklinį postūmį į kairę per r bitų; skaičiai K_t užrašomi 32 bitų žodžiais. Kiekviename iš 80 žingsnių panaudojama sava funkcija

$$f_t : \{0,1\}^{32} \times \{0,1\}^{32} \times \{0,1\}^{32} \to \{0,1\}^{32}.$$

Štai jos visos, apibrėžtos naudojant logines OR, AND ir NOT operacijas su bitais:

$$f_t(x,y,z) = \begin{cases} (x \wedge y) \vee ((\neg x) \wedge z), & 0 \le t \le 19, \\ x \oplus y \oplus z, & 20 \le t \le 39, \\ (x \wedge y) \vee (x \wedge z) \vee (y \wedge z), & 40 \le t \le 59, \\ x \oplus y \oplus z, & 60 \le t \le 79. \end{cases}$$

O dabar jau galima nusakyti funkcijos darbą labai trumpai: kiekviename žingsnyje $t = 0, 1, \ldots, 79$ atliekami tokie veiksmai:

$$\begin{aligned} T &= (A \leftarrow\!\!\shortmid 5) + f_t(B,C,D) + E + W_t + K_t, \\ E &= D, \\ D &= C, \\ C &= B \leftarrow\!\!\shortmid 30, \\ B &= A, \\ A &= T. \end{aligned}$$

Sudėtingiausias veiksmas yra žodžių sudėtis: žodžiai interpretuojami kaip natūralieji skaičiai ir sudedami moduliu 2^{32}. Atlikus visas 80 iteracijų, registruose A, B, C, D, E esančios reikšmės naudojamos atliekant kito 512 bitų ilgio bloko pertvarkius. Tai, kas šiuose registruose atsiranda paskutiniojo bloko pertvarkymo pabaigoje, ir yra maišos funkcijos reikšmė.

Įveikti maišos funkciją reiškia išmokti efektyviai rasti sutapimus. Kartais pasklinda viena kita žinia, kad tokį sutapimą pavyko surasti. Tačiau tokie pavieniai sutapimai praktiniam schemų, kuriose šios maišos funkcijos naudojamos, saugumui grėsmės kol kas nekelia.

22.4. Aritmetinė maišos funkcija

Ar yra maišos funkcija, kurios saugumas kokiu nors būdu gali būti įrodytas? Viena kita atsiranda...

Panagrinėsime Chaumo, van Heijsto ir Pfitzmann sugalvotą maišos funkciją, kurios saugumą galime susieti su diskrečiojo logaritmo uždavinio sudėtingumu.

Tegu p yra saugus pirminis skaičius, t. y. $p = 2q + 1$, čia q irgi yra pirminis skaičius. Tegu $\alpha, \beta \in \mathbb{F}_p$ yra du generuojantys elementai.

Apibrėšime funkciją $h : \{0, 1, \ldots, q-1\} \times \{0, 1, \ldots, q-1\} \to \mathbb{F}_p$ taip:

$$h(x_1, x_2) \equiv \alpha^{x_1} \beta^{x_2} \pmod{p}.$$

Jeigu užrašytume funkcijos argumentus x_1, x_2 ir reikšmę $h(x_1, x_2)$ bitais, matytume, kad reikšmei užrašyti reikia maždaug dvigubai mažiau simbolių. Taigi h galime laikyti maišos funkcija. Tiesa, ji apibrėžta tik riboto ilgio duomenims, tačiau yra metodų, kaip tokias funkcijas pratęsti.

Šios funkcijos saugumo patvirtinimu galime laikyti tokį teiginį:

146 teorema. *Suradę funkcijos $h(x_1, x_2)$ sutapimą, t. y. argumentus $\langle x_1, x_2 \rangle \neq \langle x_1^*, x_2^* \rangle$ su sąlyga $h(x_1, x_2) = h(x_1^*, x_2^*)$, galime rasti $\log_\alpha \beta$.*

Taigi sutapimo radimas tolygus diskrečiojo logaritmo uždavinio atvejo sprendimui. Kadangi šis uždavinys laikomas sudėtingu, tai šia maišos funkcija galima pasikliauti.

Įrodymas. Tegu $\langle x_1, x_2 \rangle \neq \langle x_1^*, x_2^* \rangle$, tačiau

$$\alpha^{x_1} \beta^{x_2} \equiv \alpha^{x_1^*} \beta^{x_2^*} \pmod{p}.$$

Jeigu būtų $x_2 = x_2^*$, tai iš karto gautume, kad $x_1 = x_1^*$. Taigi galime daryti prielaidą, kad $x_2 \neq x_2^*$. Pažymėję $u = \log_\alpha \beta$, gausime

$$\alpha^{x_1 + u x_2} \equiv \alpha^{x_1^* + u x_2^*} \pmod{p}$$

arba

$$u(x_2 - x_2^*) \equiv x_1^* - x_1 \pmod{p-1}. \tag{109}$$

Pažymėkime $d = (x_2 - x_2^*, p-1)$. Jeigu $d = 1$, tai diskretųjį logaritmą gauname iš karto:

$$u \equiv (x_1^* - x_1)(x_2 - x_2^*)^{-1} \pmod{p}.$$

Jeigu $d = 2$, tai (109) turės du sprendinius, iš kurių atsirinksime, kuris mums tinka. Atvejo $d = q$ iš viso negali būti. Išties, kadangi $0 \leq x_2 \leq q-1, 0 \leq x_2^* \leq q-1$, tai $-(q-1) \leq x_2 - x_2^* \leq q-1$ ir šis skirtumas negali dalytis iš q. Teorema įrodyta.

23 Paslapties dalijimo schemos

Penki keliautojai rado didelės vertės brangakmenį. Kadangi kelionės vargai suartina, tai nesutarimų dėl netikėto radinio nekilo, kol jie sugrįžo. O sugrįžę jie nutarė brangakmenį laikyti seife, kol nuspręs, ką su juo veikti. Tačiau kasdienis gyvenimas jau nebe kelionė. Ar nekils kam nors noras pasisavinti brangenybę? Kaip padaryti, kad seifą jie galėtų atverti tik susirinkę visi? Vienas sprendimas – reikia, kad būtų penki užraktai su skirtingais raktais. O jeigu užraktas vienas, be to, atrakinamas surinkus, pavyzdžiui, dešimties skaitmenų kodą? Kaip nors paskirstyti skaitmenis po porą, kad kiekvienas žinotų tik savo? Tai įmanoma, tačiau nelabai saugu. Vienas iš tų penkių gal ir negalės apgauti likusių keturių, tačiau keturi vieną – visai lengvai. Juk susirinkus keturiems tektų patikrinti daugiausiai šimtą variantų, kad seifas atsivertų. Pažiūrėkime, kaip toks paslapties padalijimo uždavinys sprendžiamas kriptografijoje.

23.1. Shamiro paslapties dalijimo schemos su slenksčiais

Kartais paslapčiai atkurti būtina, kad dalyvautų visi, kartais pakanka, kad susirinktų ne mažiau kaip t dalyvių.

Paslapties padalijimo schema negali apsieiti be dalytojo(s). Tarkime, kad dalytojas yra D (Dalia arba Dalius), kuri(s) po paslapties dalijimo išvyksta kur nors labai ilgam. Paslapties dalių gavėjus žymėkime $D_1, D_2, \ldots, D_n$.

Jeigu paslaptis yra pakankamai ilgas dvejetainės abėcėlės žodis $S \in \{0,1\}^m$, o padalyti paslaptį reikia taip, kad tik visi dalyviai susirinkę galėtų ją atkurti, sprendimas gali būti labai paprastas. Dalytojas atsitiktinai parenka $n-1$-ą žodį $S_1, \ldots, S_{n-1} \in \{0,1\}^m$ ir apskaičiuoja

$$S_n = S \oplus S_1 \oplus S_2 \oplus \ldots \oplus S_{n-1}.$$

Dabar kiekvienas dalyvis D_i gauna savo paslapties dalį S_i. Susirinkę visi kartu gali atkurti paslaptį taip:

$$S = S_1 \oplus S_2 \oplus \ldots \oplus S_{m-1} \oplus S_n.$$

O štai Shamiro pasiūlyta schema, naudojanti lyginius.

Tegu m yra didelis natūrinis skaičius, o paslaptis – koks nors skaičius $S \in \mathbb{Z}_m$. Dalytojas D atsitiktinai parenka skaičius $S_1, S_2, \ldots, S_{n-1} \in \mathbb{Z}_m$ ir apskaičiuoja

$$S_n \equiv S - S_1 - S_2 - \ldots - S_{n-1} \pmod{m}.$$

Kiekvienas dalyvis D_i gauna savo paslapties dalį S_i. Jas visas sudėjus gaunama tikroji paslaptis:

$$S \equiv S_1 + S_2 + \ldots + S_n \pmod{m}.$$

Aišku, kad susirinkusiems $n-1$ dalyviui atspėti paslaptį taip pat sunku kaip ir tam vienam, neatėjusiam į susitikimą. Taigi sugadinti paslapties atkūrimo misteriją gali bet kuris neatvykęs į renginį dalyvis. Galima sugadinti ir kitaip – jeigu bent vienas iš dalyvių nurodytų savo paslapties dalį neteisingai, paslaptis būtų nesužinota, nesėkmės kaltininkas irgi nebūtų nustatytas.

Panagrinėkime, kaip bent iš dalies galima išvengti tokių nepatogumų.

118 apibrėžimas. *Tegu S yra paslaptis, o $S_1, S_2, \ldots, S_n$ yra jos dalys, kurias dalytojas D įteikė dalyviams $D_1, D_2, \ldots, D_n$. Šį paslapties padalijimą vadinsime paslapties dalybomis su slenksčiu t ($1 \le t \le n$), jeigu S galima atkurti tik iš ne mažiau kaip t bet kurių paslapties dalių.*

Abiejų nagrinėtų paslapties padalijimo schemų slenkstis lygus dalyvių skaičiui. Jeigu $t = 1$, tai paslapties dalijimo būdas irgi aiškus: $S_i = S$, t. y. kiekvienam įteikiama paslapties kopija. O kaip padalyti paslaptį, kai $1 < t < n$? Panagrinėsime Shamiro pasiūlytą metodą.

Dalytojas D parenka pakankamai didelį pirminį skaičių p, $p > n$, parenka skirtingus skaičius $x_1, x_2, \ldots, x_n \in \mathbb{F}_p$ ir kiekvienam dalyviui D_i įteikia x_i. Šie skaičiai nėra slapti, todėl galima, pavyzdžiui, visą skaičių sąrašą kartu su p paskelbti paslapties dalyboms skirtame tinklalapyje. Paslaptis yra skaičius $S \in \mathbb{F}_p$. Dalytojas atsitiktinai pasirenka skaičius $a_1, a_2, \ldots, a_{t-1}$ ir sudaro daugianarį

$$a(x) = S + a_1 x + a_2 x^2 + \ldots + a_{t-1} x^{t-1} \in \mathbb{F}_p[x].$$

Paslapties dalis, kuri įteikiama dalyviui D_i, yra $S_i \equiv a(x_i) \pmod p$.

Paslaptį S, t. y. laisvąjį daugianario narį, galima nustatyti iš bet kurių t paslapties dalių $S_{i_1}, S_{i_2}, \ldots, S_{i_t}$. Iš tikrųjų, susirinkę t dalyvių gali pasinaudodami savo paslapties dalimis sudaryti t lygčių su t nežinomųjų sistemą

$$\begin{aligned}
&S + a_1 x_{i_1} + a_2 x_{i_1}^2 + \ldots + a_{t-1} x_{i_1}^{t-1} = S_{i_1},\\
&S + a_1 x_{i_2} + a_2 x_{i_2}^2 + \ldots + a_{t-1} x_{i_2}^{t-1} = S_{i_2},\\
&\ldots\ldots\ldots\ldots\ldots\ldots\\
&S + a_1 x_{i_t} + a_2 x_{i_t}^2 + \ldots + a_{t-1} x_{i_t}^{t-1} = S_{i_t}
\end{aligned}$$

su nežinomaisiais $S, a_1, a_2, \ldots, a_{t-1}$. Ši sistema moduliu p turi vienintelį sprendinį. Taigi t dalyvių visada gali surasti S.

Paslaptį galima apskaičiuoti ir iš karto, pasinaudojus Lagrange'o interpoliacine formule. Kadangi daugianario $a(x)$ laipsnis yra $t-1$, tai jį galime surasti žinodami t jo reikšmių, įgyjamų su skirtingomis argumento reikšmėmis. Tegu $v_i = a(u_i), i = 1, 2, \ldots, t$, čia $u_i \in \mathbb{F}_p$ yra skirtingi elementai. Tada

$$a(x) = \sum_{i=1}^{t} v_i \prod_{\substack{1 \le j \le t \\ j \ne i}} \frac{x - u_j}{u_i - u_j} \pmod p.$$

Skaičiuojant dalybą reikia suprasti, žinoma, kaip daugybą iš daliklių atvirkštinių elementų kūne $\mathbb{F}_p$. Jeigu įstatysime $x = 0$, gausime laisvąjį narį, t. y. paslaptį, kurią reikia atskleisti:

$$S \equiv \sum_{i=1}^{t} v_i \prod_{\substack{1 \le j \le t \\ j \ne i}} \frac{u_j}{u_j - u_i} \pmod{p}.$$

Taigi susirinkę t dalyvių $D_{i_1}, D_{i_2}, \ldots, D_{i_t}$ gali įstatyti į šią lygybę $v_i = S_{j_i}, u_i = x_{j_i}$ ir paslaptis bus sužinota.

Shamiro paslapties dalijimo schema su slenksčiu t

Paslapties dalijimas: D – dalytojas, $D_1, D_2, \ldots, D_n$ – dalyviai. D parenka pirminį p, skirtingus $x_1, x_2, \ldots, x_n \in \mathbb{F}_p$ ir D_i įteikia skaičių x_i. Pirminis p yra viešas, paslaptis – skaičius $S \in \mathbb{F}_p$.
D parenka skaičius $a_1, a_2, \ldots, a_{t-1}$, sudaro daugianarį

$$a(x) = S + a_1 x + a_2 x^2 + \ldots + a_{t-1} x^{t-1}$$

ir, apskaičiavęs paslapties dalis $S_i \equiv a(x_i) \pmod{p}$, įteikia jas D_i.

Paslapties atkūrimas: susirinkę t dalyvių $D_{i_1}, D_{i_2}, \ldots, D_{i_t}$ paslaptį gali atkurti taip:

$$S \equiv \sum_{i=1}^{t} S_{j_i} \prod_{\substack{1 \le k \le t \\ k \ne i}} \frac{x_{j_k}}{x_{j_k} - x_{j_i}} \pmod{p}.$$

Pastebėkime, kad Shamiro paslapties padalijimo schemą galime sukurti imdami vietoj $\mathbb{F}_p$ bet kokį baigtinį kūną $\mathbb{F}_q$.

Jeigu bent vienas iš susirinkusių dalyvių nurodys neteisingą savo paslapties dalį, paslaptis bus neatkurta. Tarkime, į paslapties atkūrimo „renginį" susirinko dalyviai $D_1, D_2, \ldots, D_{t+r}$ ($r \ge 2$), taigi dalyvių yra šiek tiek daugiau, negu iš tikrųjų reikia. Samprotavimai, žinoma, nepasikeistų, jeigu vietoje šios grupės susirinktų kita tokio pat dydžio dalyvių grupė.

Iš šių dalyvių paslapčių sudarykime žodį $\mathbf{c} = \langle S_1, S_2, \ldots, S_{t+r} \rangle$. Prisiminę, kaip dalytojas sukūrė šias paslapties dalis, galime užrašyti:

$$\mathbf{c} = S\langle 1, 1, \ldots, 1 \rangle + a_1 \langle x_1, x_2, \ldots, x_{t+r} \rangle + \ldots + a_{t-1} \langle x_1^{t-1}, x_2^{t-1}, \ldots, x_{t+r}^{t-1} \rangle.$$

Prisiminę kodavimo teoriją, galime padaryti išvadą:

$\mathbf{c}$ yra Reedo-Solomono kodo su abėcėle $\mathbb{F}_p$ žodis!

Šio kodo parametrai yra $[t+r, t, r+1]$, taigi kodas gali ištaisyti $v = [r/2]$ klaidų. Jeigu ne visi dalyviai nurodys teisingas savo paslapčių dalis, tai gausime ne žodį $\mathbf{c}$, bet iškraipytą žodį $\mathbf{d}$. Tačiau jeigu nesąžiningų dalyvių

nėra daugiau kaip v, tai, pritaikę klaidų taisymo algoritmą, ne tik nustatysime tuos, kurie melavo, bet ir sužinosime, kokias savo paslapčių dalis jie turėjo nurodyti!

23.2. Dar dvi schemos

Vienoje schemoje naudojamasi algebra, o kitoje – skaičių teorija. Jeigu tiksliau – kinų liekanų teorema.

Galima tarti, kad, atkuriant paslaptį, padalytą pagal Shamiro schemą su slenksčiu, sprendžiama t lygčių sistema su t nežinomųjų. Kiekvienas dalyvis į šią sistemą „atneša" savo lygtį. Taigi galima įsivaizduoti, kad, dalijant paslaptį, kiekvienas dalyvis gauna duomenis savo lygčiai – paslapties atkūrimo sistemos daliai – sudaryti.

Panagrinėkime Blakely paslapties padalijimo schemą, kurioje dalyviai gauna jau dalytojo sudarytas lygtis. Tegu p yra pakankamai didelis pirminis skaičius, paslaptis – skaičius $S < p$. Dalytojas atsitiktinai parenka skaičius $s_2, s_3, \ldots, s_t$, sudaro vektorių

$$\mathbf{s} = \langle s_1, s_2, \ldots, s_t \rangle, \quad s_1 = S,$$

ir pradeda dalyti paslaptį. Dalyvio D_j paslapties dalį jis sukuria taip: parenka atsitiktinius elementus $a_1^{(j)}, a_2^{(j)}, \ldots, a_{t-1}^{(j)}$, apskaičiuoja

$$c_j \equiv s_t - \sum_{i=1}^{t-1} a_i^{(j)} s_i \pmod{p}$$

ir įteikia dalyviui D_j lygtį

$$x_t \equiv c_j + \sum_{i=1}^{t-1} a_i^{(j)} x_i \pmod{p}.$$

Susirinkę t dalyvių gali sudaryti t lygčių su t nežinomųjų sistemą ir ją išsprendę rasti paslaptį $m_1 = S$. Tiesa, kad tai jiems pavyktų, t. y. kad sprendinys būtų vienintelis, sistemos koeficientų matrica turi būti neišsigimusi. Jeigu dalytojas nesinaudojo kokiu nors specialiu koeficientų parinkimo būdu, kartais taip gali ir nebūti. Shamiro paslapties padalijimo schema iš tiesų yra atskiras šios sistemos atvejis su garantija, kad bet kuri t lygčių sistema turės vienintelį sprendinį.

O dabar panagrinėkime paslapties padalijimo schemą, kurioje panaudojama senoji geroji kinų liekanų teorema. Metodą vadinsime autorių vardais: Asmutho-Bloomo paslapties padalijimo schema su slenksčiu.

Tegu n yra dalyvių skaičius, o t – slenksčio parametras. Dalytojas parenka $n+1$-ą skaičių $p < p_1 < \ldots < p_n$; p_i turi būti tarpusavyje pirminiai (kad būtų pirminiai nebūtina), be to, reikia, kad būtų patenkinta sąlyga:

$$p_1 p_2 \ldots p_t > p p_n p_{n-1} \ldots p_{n-t+2}. \tag{110}$$

Taigi t mažiausiųjų skaičių p_j sandauga turi būti didesnė už $t-1$ didžiausiųjų sandaugą. Visi šie skaičiai yra vieši.

Tegu skaičius S, $S < p$, yra paslaptis, kurią reikia padalyti. Pažymėkime $N = p_1 p_2 \dots p_t$. Kadangi N yra mažiausiųjų t sekos $p_1, p_2, \dots, p_n$ skaičių sandauga, tai, parinkę bet kuriuos kitus t skaičių, tikrai gausime daugiau.

Dabar parinkime bet kokį (geriau didelį) skaičių r, tenkinantį sąlygą $0 < r < N/p - 1$, ir sudarykime skaičių

$$S^* = S + rp, \quad S^* < S + N - p < N.$$

Taigi S^* yra mažesnis už bet kurių t skaičių $p_{i_1}, p_{i_2}, \dots, p_{i_t}$ sandaugą.

Paslapčiai atskleisti pakanka sužinoti S^*, nes $S \equiv S^* \pmod p$. Dalyvio D_j paslapties dalis sudaroma taip: $S_j \equiv S^* \pmod{p_j}$. Taigi kiekvienas dalyvis žino paslapties dalybos iš savo modulio liekaną.

Susirinkę dalyviai $D_{i_1}, D_{i_2}, \dots, D_{i_t}$ gali pasinaudoti kinų liekanų teorema ir surasti S^* dalybos iš $p_{i_1} p_{i_2} \dots p_{i_t}$ liekaną; tačiau S^* yra mažesnis už šią sandaugą skaičius, tai liekana ir bus šis skaičius.

Galbūt kilo klausimas, o kam reikia (110) sąlygos? Atsakymas: kad paslapties negalėtų atskleisti mažesnė nei t dalyvių grupė!

Asmutho-Bloomo paslapties padalijimo schema su slenksčiu t

Paslapties dalijimas: dalytojas D parenka skaičius

$$p < p_1 < p_2 < \ldots < p_n, \ (p_i, p_j) = 1, \ i \neq j,$$

$$p_1 p_2 \dots p_t > p p_n p_{n-1} \dots p_{n-t+2}.$$

Paslaptis – skaičius $S, S < p$. Dalytojas parenka $r, r < N/p - 1$, ir apskaičiuoja $S^* = S + rp$. Dalyviui D_i paskiriamas skaičius p_i ir saugiu kanalu perduodama paslapties dalis $S_i \equiv S^* \pmod p$.

Paslapties atkūrimas: t dalyvių, naudodamiesi kiniškąja liekanų teorema išsprendžia lyginių sistemą $x \equiv S_{i_j} \pmod{p_{i_j}}$ $(j = 1, 2, \dots, t)$ ir randa sprendinį $x = S^*$. Paslaptis $S \equiv S^* \pmod p$.

23.3. Leidimų struktūros

Kaip padalyti seifui atidaryti naudojamą slaptažodį, kad bet kuri darbuotojų pora galėtų jį atidaryti, išskyrus du labai gerus draugus?

Nagrinėtose paslapties padalijimo schemose visi dalyviai turi „vienodą svorį", t. y. nei vieno iš jų paslapties dalis nėra svarbesnė už kitų. Nesudėtinga paslaptį padalyti taip, kad, dalyvaujant svarbesniems dalyviams, pakaktų mažiau dalyvių nei numatyta. Pavyzdžiui, kai kuriems dalyviams galime duoti po daugiau paslapties dalių. Tačiau ir toks paslapties padalijimas kartais gali būti nepatikimas.

Tarkime, nuspręsta paslaptį padalyti n dalyviams, kad ją galėtų atkurti ne mažiau kaip trys, t. y. slenksčio parametras $t = 3$. Tačiau grupėje yra du svarbūs asmenys A ir B, kuriems mes norėtume suteikti daugiau galių. Galime jiems duoti po dvi paslapties dalis. Tada paslapčiai atverti pakaks A (arba B) ir dar vieno dalyvio. Tačiau A ir B dviese taip pat gali atverti paslaptį! O jeigu Algis ir Birutė – brolis ir sesuo?

Panagrinėkime paslapties padalijimo metodus, labiau atsižvelgiančius į dalyvių tarpusavio santykius.

119 apibrėžimas. *Tegu* $\mathbf{D} = \{D_1, D_2, \ldots, D_n\}$ *yra dalyvių aibė. Jos poaibių sistemą* $\mathcal{G}$ *vadinsime leidimų struktūra, jeigu kiekvienos aibės* $A \subset \mathcal{G}$ *viršaibis* B *(t. y. aibė, tenkinanti sąlygą* $A \subset B$*) taip pat įeina į* $\mathcal{G}$.

Akivaizdu, kad norint apibrėžti leidimų struktūrą $\mathcal{G}$, pakanka nurodyti tik tokią jos posistemę $\Gamma \subset \mathcal{G}$, kad su kiekviena $A \in \Gamma$ jos poaibiai nepriklausytų Γ. Tokią posistemę vadinsime leidimų struktūros branduoliu. Pavyzdžiui, visų poaibių, kurie turi lygiai po t elementų, sistema yra leidimų struktūros branduolys.

120 apibrėžimas. *Sakysime, kad paslapties padalijimo schema realizuoja leidimų struktūrą* $\mathcal{G}$*, jeigu paslaptį iš savo dalių gali atkurti tik tokie dalyviai, iš kurių sudarytas poaibis priklauso* $\mathcal{G}$.

Nesunku įsitikinti, kad norint realizuoti leidimų struktūrą $\mathcal{G}$ reikia paslaptį padalyti taip, kad ją galėtų atkurti poaibių iš branduolio Γ dalyviai, tačiau negalėtų atkurti tokie dalyviai, kurie nesudaro jokio B iš Γ viršaibio.

Bet kokią leidimų struktūrą galima realizuoti! Panagrinėkime labai paprastą metodą, kurį pasiūlė Benalohas ir Leichteris. Jų idėją suprasime panagrinėję pavyzdį.

Tegu $\mathbf{D} = \{D_1, D_2, D_3, D_4, D_5\}$ yra dalyvių aibė, o

$$G_1 = \{D_1, D_4, D_5\},\ G_2 = \{D_1, D_2, D_4\},\ G_3 = \{D_3, D_4\}$$

yra leidimų struktūros branduolys. Tegu paslaptis yra skaičius S. Pasinaudosime Shamiro paslapties padalijimo schema, kurioje paslapčiai atkurti reikia visų dalyvavimo.

Tegu n yra pakankamai didelis skaičius, $S < n$. Grupės G_1 dalyviams įteikime skaičius

$$S_{11},\ S_{14},\ S_{15} \equiv S - S_{11} - S_{14} \pmod{n},$$

čia S_{11}, S_{14} yra atsitiktinai parinkti skaičiai. Sudėję savo paslapčių dalis, šios grupės dalyviai visada galės atkurti paslaptį. Grupės G_2 dalyviams įteikime

$$S_{21},\ S_{22},\ S_{24} \equiv S - S_{21} - S_{22} \pmod{n},$$

o paskutiniosios grupės dalyviams

$$S_{33},\ S_{34} \equiv S - S_{33} \pmod{n}.$$

Paslaptis padalyta, leidimų struktūra realizuota. Kiekvienas dalyvis gavo tiek paslapties dalių, keliose branduolio grupėse jis dalyvauja: trečiasis tik vieną, o ketvirtasis tris. Taigi kuo asmuo „svarbesnis", t. y. į kuo daugiau grupių jis įtrauktas, tuo didesnis jo raktų ryšulys.

Galima pasistengti sudaryti leidimų struktūrą realizuojančią paslapties padalijimo schemą, kurioje kiekvienas dalyvis gauna po tokio paties dydžio paslapties dalį, tinkančią paslapčiai atkurti visose grupėse. Tačiau tikėtis, kad ją bus paprasta sudaryti, neverta.

Panagrinėkime Brickelio paslapties padalijimo schemą. Tegu dalyvių aibė yra $\mathbf{D} = \{D_1, D_2, \ldots, D_n\}$, o Γ – leidimų struktūros, kurią reikia realizuoti, branduolys. Tegu paslaptis vėl yra skaičius S. Dalytojas turi parinkti pakankamai didelį pirminį skaičių p, natūrinį skaičių d ir sukonstruoti funkciją $\delta : \mathbf{D} \to \mathbb{F}_p^d$, tenkinančią tokią sąlygą:

$$\langle 1, 0, \ldots, 0\rangle \in \mathcal{L}(\delta(D_{i_1}), \delta(D_{i_2}), \ldots, \delta(D_{i_r}))$$

tada ir tik tada, kai dalyvių aibė $\{D_{i_1}, D_{i_2}, \ldots, D_{i_r}\}$ yra kokios nors aibės iš branduolio viršaibis. Ši funkcija gali būti vieša. Čia $\mathcal{L}(\ldots)$ žymi tiesinį žodžių sistemos apvalką.

Šios funkcijos konstravimas yra sunkiausia schemos dalis. Bandymų kelias tikriausiai geriausias būdas tokiai funkcijai sudaryti.

Sudarius funkciją, paslaptis padalijama labai paprastai. Dalytojas, parinkęs $a_2, a_3, \ldots, a_d \in \mathbb{F}_p$ atsitiktinai, sudaro dar vieną vektorių

$$\mathbf{a} = \langle a_1, a_2, \ldots, a_d\rangle, \quad a_1 = S,$$

ir apskaičiuoja paslapties dalis $S_i \equiv \mathbf{a} \cdot \delta(D_i) \pmod p$. Ženklas $\cdot$ čia žymi skaliarinę vektorių sandaugą.

Dabar, tarkime, susirinko dalyviai, kurių sudaroma aibė G priklauso leidimų struktūros branduoliui arba yra jo viršaibis. Tada, išsprendę tiesinių lygčių sistemą, jie gali surasti skaičius c_i, kad būtų

$$\langle 1, 0, \ldots, 0\rangle = \sum_{D_i \in G} c_i \delta(D_i).$$

Jeigu padauginsime skaliariškai šią lygybę iš $\mathbf{a}$, gausime

$$\langle S, 0, \ldots, 0\rangle \equiv \sum_{D_i \in G} c_i(\mathbf{a} \cdot \delta(D_i)) \equiv \sum_{D_i \in G} c_i S_i \pmod p.$$

Tačiau dešiniąją lyginio pusę dalyviai gali apskaičiuoti! Taigi jie gali atkurti ir paslaptį.

Shamiro paslapties padalijimo schema su slenksčiu iš tikrųjų yra atskiras šios schemos atvejis. Funkcija $\delta : \mathbf{D} \to \mathbb{F}_p^t$ apibrėžiama taip:

$$\delta(D_i) = \langle 1, x_i, x_i^2, \ldots, x_i^{t-1}\rangle.$$

24 Kriptografiniai protokolai

Kriptografinės priemonės – kriptosistemos, skaitmeniniai parašai, maišos funkcijos ir kitos schemos – skirtos įvairiems duomenų apsaugos uždaviniams spręsti. Šie uždaviniai kyla labai konkrečiose aplinkose. Mums reikia apsaugoti ne duomenis apskritai, bet, pavyzdžiui, duomenis, kurie perduodami kompiuteriniais tinklais arba įrašomi į duomenų bazę. Taigi kriptografinės apsaugos priemonės tampa bendros informacinės sistemos dalimi, kurios efektyvumą lemia ne vienintelio asmens veiksmai. Todėl būtinos taisyklės, kurios numato, kokius veiksmus ir kokia tvarka turi atlikti asmenys, kad naudojama kriptografinė priemonė iš tikrųjų užtikrintų tokią duomenų apsaugą, kokią ji galėtų suteikti. Ta taisyklių visuma, numatanti asmenų (arba juos atstovaujančių kompiuterių ar programų), naudojančių kriptografinę schemą, veiksmų seką, ir vadinama kriptografiniu protokolu. Su kriptografiniais protokolais jau susidūrėme, pavyzdžiui, nagrinėdami nepaneigiamo parašo schemą.

24.1. Raktų paskirstymas

Norėtume susitarti su draugu dėl simetrinės kriptosistemos rakto, bet mūsų ryšio kanalas kontroliuojamas. Išeitis yra!

Viešojo rakto kriptosistemoms saugaus kanalo raktams perduoti nereikia. Tačiau šifravimo ir dešifravimo veiksmai naudojant viešojo rakto kriptosistemas atliekami daug lėčiau negu simetrinėse kriptosistemose. Galbūt galima suderinti abiejų kriptosistemų privalumus? Pavyzdžiui, naudojantis viešojo rakto kriptosistema, galima perduoti simetrinės sistemos raktus, o kai raktai perduoti – šifruoti ir dešifruoti greičiau veikiančia simetrine sistema.

Pirmąjį būdą simetrinės kriptosistemos raktams perduoti nesaugiu kanalu pasiūlė Diffie ir Hellmanas. Panagrinėkime jų protokolą.

A ir B nori sudaryti simetrinės kriptosistemos raktą, tačiau negali naudotis saugiu kanalu. Pirmiausia jie turi susitarti dėl pakankamai didelio pirminio skaičiaus p ir generuojančio elemento $g \in \mathbb{F}_p$. Tai jie gali padaryti nesaugiais kanalais. Tada kiekvienas pasirenka po skaičių: tarkime, A pasirinko skaičių $x_A, 0 < x_A < p-1$, o B – skaičių $x_B, 0 < x_B < p-1$. Dabar A skaičiuoja $X \equiv g^{x_A} \pmod p$ ir siunčia B, o B savo ruožtu – skaičiuoja $Y \equiv g^{x_B} \pmod p$ ir siunčia A. Tiek A, tiek B gali surasti tą patį skaičių ir naudoti jį kaip simetrinės kriptosistemos raktą:

$$K \equiv Y^{x_A} \equiv g^{x_A x_B} \pmod p, \quad K \equiv X^{x_B} \equiv g^{x_A x_B} \pmod p.$$

Diffie-Hellmano rakto nustatymo protokolas

Pasirengimas: A ir B viešu kanalu susitaria dėl pirminio skaičiaus p ir generuojančio elemento g. A pasirenka skaičių $x_A, 0 < x_A < p-1$, o B – skaičių $x_B, 0 < x_B < p-1$.
Raktų nustatymas:

1. A skaičiuoja $X \equiv g^{x_A} \pmod{p}$ ir siunčia B; B skaičiuoja $Y \equiv g^{x_B} \pmod{p}$ ir siunčia A.
2. A apskaičiuoja raktą $K \equiv Y^{x_A} \pmod{p}$, B apskaičiuoja raktą $K \equiv X^{x_B} \pmod{p}$.

Šį protokolą galime naudoti ir didesnės grupės bendram simetrinės kriptosistemos raktui nustatyti. Tegu, pavyzdžiui, A, B ir C nesaugiu kanalu nori sudaryti bendrą raktą. Kaip ir anksčiau, pirmiausia jiems reikia susitarti dėl pirminio skaičiaus p ir generuojančio elemento g ir pasirinkti po skaičių. Pažymėkime A, B ir C pasirinktus skaičius x_A, x_B ir x_C. Tada jiems reikia atlikti šiuos protokolo žingsnius:

1. A siunčia B $g^{x_A} \pmod{p}$, gauna iš C $g^{x_C} \pmod{p}$; B siunčia C $g^{x_B} \pmod{p}$, gauna iš A $g^{x_A} \pmod{p}$; C siunčia A $g^{x_C} \pmod{p}$, gauna iš B $g^{x_B} \pmod{p}$.
2. A siunčia B $g^{x_A x_C} \pmod{p}$, gauna iš C $g^{x_B x_C} \pmod{p}$; B siunčia C $g^{x_A x_B} \pmod{p}$, gauna iš A $g^{x_A x_C} \pmod{p}$; C siunčia A $g^{x_B x_C} \pmod{p}$, gauna iš B $g^{x_A x_B} \pmod{p}$.
3. kiekvienas, keldamas 2-ajame žingsnyje gautą skaičių laipsniu savo rodikliu, apskaičiuoja bendrą raktą $K \equiv g^{x_A x_B c_C} \pmod{p}$.

Trumpai protokolą galima apibūdinti taip: gavęs skaičių, dalyvis kelia jį laipsniu savuoju rodikliu ir siunčia toliau. Pakėlę laipsniu paskutinįjį kartą, visi dalyviai gauna tą patį skaičių – raktą.

Šifravimas simetrine kriptosistema galimas tik baigus rakto nustatymo protokolą. Hugeso raktų nustatymo protokolas suteikia galimybę A užšifruoti ir nusiųsti B pranešimą šiandien, o susitarimą dėl rakto nukelti rytdienai.

Tarkime, raktas, kurį panaudojo šifravimui A, sudarytas taip: $K \equiv g^x \pmod{p}$. Dėl pirminio p ir generuojančio elemento g turi būti susitarta anksčiau arba A gali šiuos dydžius perduoti B nesaugiu kanalu.

Gavusi šifrą, kurio kol kas negali iššifruoti, B pradeda raktų nustatymo protokolą:

1. B parenka skaičių $y, (y, p-1) = 1$, ir A siunčia $Y \equiv g^y \pmod{p}$;
2. A, gavęs Y, siunčia B $X \equiv Y^x \pmod{p}$;

3. B skaičiuoja $z \equiv y^{-1} \pmod{p-1}$ ir randa raktą $K \equiv X^z \pmod{p}$.

Jeigu raktui nustatyti naudojamas šis protokolas, tai A gali nusiųsti tą patį šifruotą pranešimą keliems draugams, o kai jie norės jį perskaityti, įvykdyti rakto nustatymo protokolą.

Jeigu grupė dalyvių $D_1, D_2, \ldots, D_n$ nori naudotis simetrine kriptosistema su atskiru kiekvienai porai raktu, jie gali susitarti dėl p ir g ir paskelbti savo grupės tinklalapyje skaičius

$$X_1 \equiv g^{x_1} \pmod{p}, X_2 \equiv g^{x_2} \pmod{p}, \ldots, X_n \equiv g^{x_n} \pmod{p},$$

čia $0 < x_i < p-1$ dalyvio D_i pasirinktas skaičius. Tada D_i, norėdamas rašyti šifruotą laišką D_j, gali naudoti kaip raktą skaičių $K_{ij} \equiv X_j^{x_i} \pmod{p}$.

Nors šie protokolai išsprendžia rakto nustatymo uždavinį, jie yra bejėgiai prieš „vidurio ataką“. „Vidurio atakos“ esmė tokia: kai A siunčia B skaičių $X \equiv g^{x_A} \pmod{p}$, Z gali jį perimti, išsaugoti ir pasiųsti B $X' \equiv g^{x_Z} \pmod{p}$. Tą patį Z gali padaryti ir su B siunčiamu skaičiumi $Y \equiv g^{x_B} \pmod{p}$. Kai raktų nustatymo protokolas bus užbaigtas, A turės bendrą su Z raktą $K_1 \equiv g^{x_A x_Z} \pmod{p}$, o B – bendrą su Z raktą $K_2 \equiv g^{x_B x_Z} \pmod{p}$. Tada Z galės gautus iš A laiškus iššifruoti su K_1 ir, vėl užšifravęs su K_2, siųsti B. Analogiškai Z galės elgtis su B laiškais. Taigi Z taps nepagaunamu ryšio kontrolieriumi, jeigu tik nesukels įtarimo delsdamas persiųsti laiškų.

Kovai prieš vidurio atakas reikalingos papildomos priemonės ir protokolai. Galima išvengti vidurio atakų, naudojant raktų nustatymo protokoluose skaitmeninius parašus.

24.2. Įrodymai, nesuteikiantys žinių

Tokius įrodymus kartai pateikia informatikai profesionalai. Jeigu paklausite, kaip dirbama su kokia nors sudėtinga programa, tikriausiai išvysite virtuozišką „grojimą“ klaviatūra, greitą meniu ir langų kaitą... Jums bus įrodyta, kokia gera ši programa, tačiau kažin ar daug suteikta žinių...

Įrodymai, nesuteikiantys žinių (Zero Knowledge Proofs (ZKP), angl.), – tai įrodymai, kuriais įrodinėtoja I (Irena) tikrintojui T (Tomui) įrodo, kad žino tam tikrus duomenis d, tačiau jų neatskleidžia. Dar daugiau, iš tos informacijos, kurią I suteikia T, jo (o taip pat ir pasyviai stebinčių protokolo eigą) žinios apie d nepasikeičia. Galime suprasti, kad tai iš tiesų įmanoma, prisiminę, kaip naudojamasi viešojo rakto kriptosistema. Tarkime, I sukūrė viešojo rakto kriptosistemą: paskelbė viešąjį raktą K_v ir išsaugojo savo kompiuteryje privatųjį K_d. T nori įsitikinti, ar kriptosistema, kurią sukūrė I, „tikra“, ar I nepaskelbė rakto K_v šiaip sau... T gali užšifruoti tekstą M ir pasiųsti I šifrą $C = e(M|K_v)$, prašydamas, kad I atsiųstų M. Jeigu Irena įvykdė prašymą, tai Tomas gali pripažinti tai kaip įrodymą, kad Irena tikrai žino dešifravimui skirtą raktą.

Taigi įrodymas, nesuteikiantis žinių, yra tam tikras protokolas, kuriam pasibaigus T įsitikina, kad I tikrai žino tai, ką teigia žinanti, tačiau iš duomenų, naudotų vykdant protokolą, jos žinių nustatyti negalima. Panagrinėkime keletą pavyzdžių.

Grafą G galime užrašyti nurodydami jo viršūnių aibę V ir viršūnių, kurios yra sujungtos briaunomis, porų aibę E. Šias viršūnių poras vadinsime tiesiog briaunomis.

Grafai $G_1 = \langle V_1, E_1\rangle$ ir $G_2 = \langle V_2, E_2\rangle$ vadinami izomorfiškais, jeigu $|V_1| = |V_2|$ ir egzistuoja abipus vienareikšmė viršūnių atitiktis $\sigma : V_1 \to V_2$, kad

$$\langle v_1, v_2\rangle \in E_1 \text{ tada ir tik tada, kai } \langle \sigma(v_1), \sigma(v_2)\rangle \in E_2.$$

Turint grafą $G = \langle V, E\rangle$, sudaryti jam izomorfiškus grafus nesunku: parinkime kokią nors viršūnių perstatą $\sigma : V \to V$ ir apibrėžkime naują briaunų aibę $E' = \{\langle \sigma(v_1), \sigma(v_2)\rangle : \langle v_1, v_2\rangle \in E\}$. Naujasis grafas $H = \langle V, E'\rangle$ bus izomorfiškas G. Žymėsime jį $H = \sigma(G)$. Savo ruožtu G galime gauti iš H keitiniu σ^{-1}, taigi $G = \sigma^{-1}(H)$.

Tačiau, turint du grafus su tiek pat viršūnių, nustatyti, ar jie yra izomorfiški – sudėtingas uždavinys.

Tarkime, Irena teigia, kad grafai $G_1 = \langle V, E_1\rangle$ ir $G_2 = \langle V, E_2\rangle$ yra izomorfiški; kadangi viršūnių skaičiai abiejuose grafuose tie patys, galime jas žymėti tais pačiais simboliais, pavyzdžiui, skaičiais. Taigi I teigia žinanti abipus vienareikšmę atitiktį $\sigma : V \to V$, „išsaugančią“ briaunas. Ji gali įrodyti Tomui, kad nemeluoja, neatskleisdama atitikties σ.

Įrodymo protokolas vykdomas taip

I: įrodysiu, kad žinau $\sigma : G_1 \to G_2$
1. I pasirenka atsitiktinę perstatą $\lambda : V \to V$ ir, sudariusi naują grafą $H = \lambda(G_1)$, siunčia jį T; 2. T atsitiktinai parenka skaičiaus $i \in \{1, 2\}$ reikšmę ir siunčia I; 3. jeigu $i = 1$, tai I atsiunčia perstatą $\pi = \lambda$, jeigu $i = 2$ – perstatą $\pi = \lambda \circ \sigma^{-1}$. 4. gavęs π, T tikrina, ar $H = \pi(G_i)$;

Kadangi grafai G_1 ir G_2 yra izomorfiški, tai naujai sudarytas grafas H yra izomorfiškas jiems abiem. T antrajame žingsnyje pareikalauja, jog I atskleistų arba H ir G_1, arba H ir G_2 izomorfizmą. Jeigu ji žino G_1 ir G_2 izomorfizmą, ji nesunkiai gali įvykdyti abu reikalavimus. Kokia galimybė apgauti tikrintoją, t. y. pasiekti, kad jis patikėtų nesamomis žiniomis?

Tikimybė, kad antrajame žingsnyje T pasirinks $i = 1$, lygi $\frac{1}{2}$. Tai ir yra apgavystės tikimybė, nes I atsakymui visai nereikia žinių apie G_1 ir G_2 izomorfizmą. Protokolą reikia kartoti. Jeigu jis kartojamas k kartų, tai apgavystės tikimybė lygi $\frac{1}{2^k}$. Labai svarbu, kad I negalėtų nuspėti, kokį skaičių atsiųs T. Jeigu, pavyzdžiui, ji tikrai žino, kad T siųs skaičių $i = 2$, ji protokolo pradžioje gali pasiųsti grafą $H = \lambda(G_2)$ ir, gavusi $i = 2$, siųsti tikrintojui $\pi = \lambda$.

Siuntinėti grafus ir atlikti su jais skaičiavimus nelabai patogu. Geriau siuntinėti skaičius. Panagrinėkime Fiato-Shamiro protokolą, kuriuo naudodamasi I įrodo, kad žino lyginio

$$x^2 \equiv a \pmod{n}, \quad n = pq,$$

sprendinį v, čia p, q yra pirminiai skaičiai, jų sandauga n paskelbiama viešai. Taigi I žino skaičių $v, v^2 \equiv a \pmod{p}$, ir nori jo neatskleisdama tai įrodyti T. Protokolas vykdomas taip:

I: įrodysiu, kad žinau $v, v^2 \equiv a$ **(mod** n**)**
1. I pasirenka atsitiktinį skaičių r ir siunčia T $b \equiv r^2 \pmod{n}$;
2. T atsitiktinai pasirenka $i \in \{0, 1\}$ ir siunčia I;
3. jeigu $i = 0$, I siunčia T skaičių $h = r$, jeigu $i = 1$, ji siunčia $h = rv$;
4. T tikrina, ar $h^2 \equiv a^i b \pmod{n}$; jeigu lyginys teisingas – įrodymą priima.

Jeigu I laikosi protokolo, T visada priims įrodymą. Iš tikrųjų, jei $i = 0$, tai T tikrina, ar $b \equiv r^2 \pmod{n}$. Jeigu $i = 1$, tai T tikrina, ar $ab \equiv (rv)^2 \pmod{n}$.

Apgavystės tikimybė ir čia lygi $\frac{1}{2}$. Jeigu $i = 0$, tai I atsakymui dydžio v nereikia. Tačiau jeigu ji iš tiesų nežinotų v, bet žinotų, kad T atsiųs skaičių $i = 2$, ji irgi galėtų apgauti. Prieš pirmąjį žingsnį ji galėtų pasirinkti bet kokį h, suskaičiuoti $b \equiv h^2 a^{-1} \pmod{n}$ ir nusiųsti šį skaičių pirmajame žingsnyje, o antrajame – h. Lyginys, kurį tikrins T, bus teisingas.

Kad apgavystės tikimybė būtų maža, protokolą reikia pakartoti kelis kartus.

Iš šių pavyzdžių matyti, kad žinios, kurių turėjimą įrodinėja I, yra sudėtingo skaičiavimo uždavinio sprendinys. Panagrinėkime dar vieną protokolą, kuriuo naudodamasi I įrodo, kad žino skaičiaus diskretųjį logaritmą.

Tegu p yra didelis pirminis skaičius, $g \in \mathbb{F}_p$ yra generuojantis elementas, $\beta \in \mathbb{F}_p$ – bet koks nenulinis elementas. Dydžiai p, g, β yra vieši. I žino diskretųjį logaritmą $u = \log_g \beta$ ir nori tai įrodyti T.

Protokolo eiga tokia:

I: įrodysiu, kad žinau u, $g^u \equiv \beta$ **(mod** p**)**
1. I pasirenka atsitiktinį skaičių $0 < r < p-1$ ir, apskaičiavusi $\gamma \equiv g^r \pmod{p}$, siunčia T; 2. T atsitiktinai pasirenka $i \in \{0,1\}$ ir siunčia I; 3. I skaičiuoja $h \equiv r + iu \pmod{p-1}$ ir siunčia T; 4. T tikrina, ar $g^h \equiv \beta^i \gamma \pmod{p}$. Jei lyginys teisingas, T priima įrodymą.

Apgavystės tikimybė šiame protokole ta pati kaip ankstesniuose. Ir šiame protokole svarbu, kad I negalėtų nuspėti, kokius skaičius atsiųs T.

24.3. Skaitmeniniai pinigai

Palyginkime du atsiskaitymo būdus. Pirmasis – atsiskaitymas parduotuvėje banko kortele. Antrasis toks: jūs rašote laišką broliui, seseriai ar tėvams, prašydami, kad jie iš tos metalinės dėžutės, kurioje laikote savo santaupas, paimtų tam tikrą sumą ir ją atiduotų A, kuriam esate skolingas. Ar skiriasi šie du būdai? Iš esmės ne!

Taigi banko kortelės su skaitmeniniais pinigais neturi nieko bendro. O kas gi yra tada tie skaitmeniniai pinigai? O kas yra popieriniai pinigai? Tam tikri dokumentai, kuriuos duodame pardavėjui, jis patikrina juos specialiu aparatu ir, ir nepaklausęs nei mūsų vardo, nei pavardės, leidžia pasiimti prekes. Vadinasi, skaitmeniniai pinigai turėtų būti tam tikri skaitmeniniai duomenys, kuriuos mes duodame pardavėjui, o jis – atitinkamai patikrinęs – priima ir nereikalauja mus identifikuojančių duomenų.

Jeigu norime sukurti skaitmeninius pinigus, turinčius pagrindines popierinių pinigų savybes, tačiau sudarytus vien tik iš skaitmenų (arba bitų, jeigu norite), reikia sugalvoti, kaip galima garantuoti tokius reikalavimus:

- autentiškumas: niekas negali pasiimti skaitmeninių pinigų mūsų vardu;
- vientisumas: sukurta pinigas negali būti pakeistas (prie 1 negali būti „prirašyta“ keletas nulių;
- tiesioginis mokėjimas: atsiskaitant skaitmeniniais pinigais, nereikia kreiptis į banką;
- saugumas: tie patys skaitmeniniai pinigai negali būti išleisti pakartotinai;

- anonimiškumas: atsiskaitant skaitmeniniais pinigais, neturi būti reikalinga informacija apie mokėtojus.

Reikalavimų daug ir jie sunkiai įgyvendinami. Juk, pavyzdžiui, kopijuoti skaitmeninius duomenis nereikia nei ypatingų prietaisų, nei sugebėjimų!

Panagrinėkime vieną kiek supaprastintą skaitmeninių pinigų sistemą ECash, kurią sukūrė kompanija DigiCash.

Taigi A nori gauti skaitmeninių pinigų ir jais atsiskaityti už prekes ar paslaugas. Žinoma, jis turi pirmiausia uždirbti tikrų pinigų ir atsidaryti sąskaitą banke B, kad galėtų šiuos tikruosius pinigus versti skaitmeniniais. Kad A vardu į banką, prašydamas skaitmeninių pinigų, negalėtų kreiptis Z, A turi naudotis kokia nors nustatyta autentifikavimo schema, pavyzdžiui, skaitmeniniais parašais. Skaitmeninis parašas būtinas ir bankui B, kad būtų galima patvirtinti išduotų skaitmeninių banknotų tikrumą. Tarkime, kad naudojama RSA skaitmeninių parašų schema. Naudojantis ja, galima kurti aklus parašus, duodant pasirašyti „užmaskuotus“ dokumentus. Protokoluose daugybos bei kėlimo laipsniu veiksmai atliekami atitinkamu moduliu.

ECash schemos skaitmeninių banknotų išdavimo protokolas

Tarkime, A nori gauti 100 EU skaitmeninį banknotą. Įdomu, kad jam ne tik nedraudžiama jį susikurti, bet jis netgi privalo pats tai padaryti.

1. A parengia n (banko nustatytą kiekį, pavyzdžiui, $n = 100$) skaitmeninių eilučių rinkinių $S_j = (I_{j1}, I_{j2}, ..., I_{jn})$, $j = 1, ..., n$; kiekvienoje eilutėje I_{jk} užrašyta A identifikuojanti informacija.

2. Kiekviena eilutė I_{jk} kaip paslaptis padalijama į dvi dalis (L_{jk}, R_{jk}).

3. A parengia n banknotų po 100 EU: $M_j = (m_j, (L_{jk}, R_{jk})_{k=1,...,n})$, čia m_j yra skirtingi banknotų identifikavimo numeriai, juose sutartu būdu nurodyta ir banknoto vertė.

4. A maskuoja banknotų numerius ir siunčia bankui $M_j^* = (z_j^e m_j, (L_{jk}, R_{jk})_{k=1,...,n})$, čia e yra banko B skaitmeninio parašo schemos viešasis raktas, z_j – A pasirinktas skaičius.

5. B atrenka $n-1$ atsiųstą banknotą (pvz., $M_1^*, \ldots, M_{99}^*$) ir pareikalauja, kad A nurodytų šiems banknotams kurti panaudotus maskuojančius daugiklius z_j.

6. B patikrina, ar atrinktieji banknotai sudaryti teisingai: ar visuose juose nurodytos vienodos vertės ir visi numeriai skirtingi.

7. Jeigu atskleistieji kvitai sudaryti teisingai, B pasirašo likusįjį ir siunčia A $((z_{100}^e m_{100})^d, (L_{100,k}, R_{100,k})_{k=1,...,n})$.

8. A pašalina maskuojamąjį daugiklį ir turi skaitmeninį banknotą $(m_{100}, (m_{100})^d)$ ir dar tam tikrą jo „priedą“ $I_{100} = (L_{100,k}, R_{100,k})_{k=1,...,n}$.

Įvykdyto protokolo rezultatas: A turi banko parašu patvirtintą 100 eurų vertės banknotą su numeriu, kurio bankas nežino. Kai bankas šį numerį sužinos, negalės jo susieti su A. Apskritai skaitmeninio banknoto struktūra turėtų būti sudėtingesnė; I_{100} irgi turėtų būti banko pasirašytas.

ECash mokėjimo protokolas

1. A pateikia pardavėjui P banknotą $(m_{100}, (m_{100})^d)$.
2. P tikrina banko B parašą $m_{100} = ((m_{100})^d)^e$.
3. P generuoja atsitiktinių bitų seką $b_1 b_2 \dots b_{100}$ ir perduoda A. Jeigu $b_i = 0$, A turi atskleisti $L_{100,i}$; jei $b_i = 1$, A turi atskleisti $R_{100,i}$.
4. P siunčia B $(m_{100}, (m_{100})^d)$ ir atskleistas $I_{100,i}$ dalis.
5. B taip pat patikrina parašą ir peržiūri savo duomenų bazę, ar pinigas su numeriu m_{100} dar nebuvo išleistas. Jeigu ne – perveda į P sąskaitą atitinkamą sumą, o į savo duomenų bazę įrašo atsiųstą informaciją. Jeigu jau išleistas – aiškinasi, kas kaltas.

Trečiojo žingsnio vykdymas reikalauja, kad A negalėtų sumeluoti atskleisdamas paslapčių dalis. Todėl ir jos turėtų būti patvirtintos banko parašu.

Kas gi atsitinka, jeigu B nustato, kad banknotas su tuo numeriu jau išleistas? Tuomet jis lygina atsiųstas paslapčių dalis su įrašytomis duomenų bazėje. Jeigu visos jos sutampa – pardavėjas P bando banknotą išgryninti pakartotinai. O jeigu nesutampa, tai banknotą bando pakartotinai panaudoti A. Tada iš pirmų nesutampančių paslapties dalių, tarkime, $L_{100,k}$ ir $R_{100,k}$, bankas nustato nesąžiningo kliento tapatybę. Gali atsitikti, kad visos atsiųstos paslapčių dalys sutaps su jau įrašytomis ne dėl pardavėjo kaltės, bet todėl, kad A bando skaitmeninį pinigą išleisti pakartotinai. Tačiau tokio atvejo tikimybė yra maža – tik 2^{-n}.

Ši schema įgyvendina visus minėtus skyrelio pradžioje reikalavimus skaitmeniniams pinigams. Tačiau, vykdant protokolo žingsnius, tenka atlikti gana daug skaičiavimų. Daugiausiai laiko sugaištama, bankui tikrinant atsiųstus banknotų variantus.

Yra ir kitokių skaitmeninių pinigų schemų. Visos jos yra gana sudėtingos. Tačiau kitaip ir būti negali. Norint praktiškai įdiegti aptartąją schemą, reiktų ją padaryti dar sudėtingesnę. Juk joje nenumatyta, pavyzdžiui, grąžos davimo galimybė.

24.4. Elektroniniai rinkimai

Balsavimo kabina su užuolaidėle – klasikinio saugaus balsavimo protokolo elementas. Kuo ją galima pakeisti, jeigu norėtume dalyvauti rinkimuose neatsitraukdami nuo savo kompiuterio?

Jeigu ketinate sukurti patikimą elektroninio balsavimo sistemą, teks pagalvoti, kaip įgyvendinti ne mažiau kaip skaitmeninių pinigų schemose sąlygų. Žinoma, jeigu pasitikite ryšio kanalais, rinkėjais ir rinkimine komisija, tai galite nesukti sau galvos. Jeigu ne – tada teks sugalvoti, kokių priemonių reikia imtis, kad

1. tik registruoti rinkėjai galėtų balsuoti;
2. kiekvienas rinkėjas galėtų balsuoti tik kartą;
3. niekas kitas negalėtų sužinoti, kaip rinkėjas balsavo;
4. kiekvienas rinkėjas galėtų įsitikinti, kad jo balsas įskaitytas;
5. joks rinkėjas negalėtų dubliuoti kito rinkėjo biuletenio;
6. niekas negalėtų pakeisti rinkėjo biuletenio.

Nors galima sukurti elektroninio balsavimo sistemą, kurioje nėra centrinės rinkimų komisijos, t. y. patys rinkėjai gali sekti, ar balsavimas vyksta korektiškai, protokolai supaprastės, jeigu tokią komisiją (toliau CRK) įsteigsime.

Paprasčiausias balsavimo protokolas galėtų būti toks. CRK paskelbia savo viešąjį raktą, skirtą biuleteniams šifruoti.

1. Rinkėjai šifruoja savo biuletenius viešuoju CRK raktu ir išsiunčia juos CRK;
2. CRK dešifruoja biuletenius, skaičiuoja ir skelbia rezultatus.

Toks balsavimas ne ką geresnis už balsavimą SMS žinutėmis įvairiuose televizijos renginiuose. Kas nori, gali balsuoti kiek nori kartų. Įgyvendinama tik paskutinė sąlyga: niekas negali pakeisti rinkėjo biuletenio (tačiau CRK gali jo neįskaityti).

Jeigu kiekvienam rinkėjui bus suteikta galimybė naudotis skaitmeninio parašo schema, galima naudoti geresnį protokolą.

1. Kiekvienas rinkėjas pasirašo biuletenį savo privačiuoju raktu.
2. Pasirašytą biuletenį rinkėjas šifruoja viešuoju CRK raktu ir jį išsiunčia CRK;
3. CRK dešifruoja biuletenius, tikrina parašus, skaičiuoja ir skelbia rezultatus.

Toks protokolas kur kas geresnis, tačiau CRK jame turi labai daug galios: ji žino, kas ir kaip balsavo, jeigu nori – gali neįskaityti dalies balsų.

Taigi vien skaitmeninių parašų neužtenka. Prisiminkime, kaip skaitmeninių pinigų sistemose naudojami aklieji parašai. Jie praverčia ir elektroninio balsavimo schemose. Štai dar vienas sudėtingesnis balsavimo protokolas.

1. Kiekvienas rinkėjas parengia n biuletenių rinkinių (pavyzdžiui, $n = 10$); kiekviename rinkinyje yra visiems balsavimo variantams parengti biuleteniai; rinkinyje kiekvienas biuletenis pažymėtas numeriu; skirtingų rinkinių numeriai yra skirtingi, jie renkami iš didelių skaičių aibės.

2. Rinkėjas maskuoja kiekvieną biuletenių rinkinį, parengdamas aklam parašui, ir išsiunčia CRK.

3. CRK tikrina, ar rinkėjas nebuvo atsiuntęs biuletenių anksčiau. Jeigu ne – atsitiktinai atrenka $n-1$-ą rinkinį ir paprašo atsiųsti šių rinkinių atskleidimo dydžius. Rinkėjas įvykdo prašymą, CRK patikrina, ar atskleistieji biuleteniai tinkamai sudaryti. Jeigu jie sudaryti pagal taisykles, CRK pasirašo paskutinį neatskleistąjį rinkinį ir nusiunčia rinkėjui.

4. Rinkėjas atskleidžia atsiųstąjį rinkinį ir gauna CRK parašu patvirtintus visų balsavimo variantų biuletenius.

5. Rinkėjas iš turimo rinkinio atrenka tą biuletenį, kuris atitinka jo valią, šifruoja jį viešuoju CRK raktu ir išsiunčia.

6. CRK dešifruoja biuletenį, patikrina savo parašą, perskaito numerį ir patikrina, ar jau priimtų biuletenių numerių sąraše tokio numerio nėra, skaičiuoja ir skelbia rezultatus, nurodydama biuletenių numerius ir kurio kandidato ar partijos naudai jie įskaityti.

Šis protokolas įgyvendina visus šešis anksčiau suformuluotus reikalavimus. Vis dėlto ir juo naudojantis gali kilti nesusipratimų. Rinkėjas gali patikrinti, ar jo balsas teisingai įskaitytas. Tačiau jeigu komisija suklydo arba neteisingai įskaitė tyčia, kaip rinkėjas gali tai įrodyti?

Galima sudaryti sudėtingesnius protokolus, kurie leidžia išvengti tokių situacijų. Galima suteikti galimybę rinkėjui, nepatenkintam savo balso įskaitymu ir norinčiam pakeisti savo nuomonę, tai padaryti. Elektroninis balsavimas pagal tokį protokolą galėtų padėti subalansuoti balsus ir išvengti pakartotinių rinkimų. Tačiau taip bus daroma, matyt, dar negreitai.

24.5. Pokeris telefonu

> *Algis ir Birutė kalbasi telefonu ir niekaip negali susitarti, kas iš jų turėtų grįžti namo anksčiau ir išvesti pasivaikščioti namuose paliktą šunį. Algis sako: „Metu monetą, jeigu ji atvirs herbu, sugrįšiu aš. Vienas, du, trys – atvirto skaičiusį' Ar tikrai Birutė niekaip negali patikrinti, kaip iš tikrųjų buvo?*

Monetos metimo protokolą telefonu garantuojant, kad baigtis bus paskelbta teisingai, tikrai galima surengti. Ir netgi daugeliu būdų. Telefonas, žinoma, čia tik dėl didesnio įspūdžio. Geriau už telefoną tokiems protokolams atlikti tinka elektroninis paštas. Panagrinėkime keletą.

1. A parenka didelį pirminį skaičių p ir praneša jį B.
2. B randa du generuojančius elementus $h, t \in \mathbb{F}_p$ ir siunčia A.
3. A parenka atsitiktinį skaičių x, skaičiuoja $y \equiv h^x \pmod{p}$ arba $y \equiv t^x \pmod{p}$ ir siunčia šį skaičių B.
4. B spėja, ar, skaičiuojant y, buvo naudotas h, ar t.
5. A patikrinimui siunčia B reikšmę x.

Galbūt geriau šį protokolą vadinti „įspėk, kurioje rankoje“ protokolu. Birutė turi galimybę pasirinkti ir pasitikrinti, ar neapsiriko. Tačiau galima jos rinkimąsi interpretuoti ir kaip monetos metimą. Jeigu ji įspėja – „moneta atvirto skaičiumi“, jeigu ne – „atvirto herbu“. Svarbu, kad baigties negali nei vienas iš dalyvių užginčyti.

O štai dar vienas protokolas, kuriame naudojamasi Pohligo-Hellmano simetrine kriptosistema. Prisiminkime, kaip ji veikia.

Parenkamas didelis pirminis skaičius p ir dar du skaičiai $k_1 = e, k_2 = d$, $e \cdot d \equiv 1 \pmod{p-1}$. Pranešimas m šifruojamas ir dešifruojamas taip:

$$c = e(m|k_1) \equiv m^{k_1} \pmod{p}, \quad m = d(c|k_2) \equiv c^{k_2} \pmod{p}.$$

„Monetos metimo“ telefonu protokolas naudojant Pohligo-Hellmano kriptosistemą

1. A ir B susitaria dėl bendro pirminio skaičiaus p ir kiekvienas pasirenka po porą kriptosistemos raktų: $K_A = \langle e_A, d_A \rangle, \quad K_B = \langle e_B, d_B \rangle$.
2. A užšifruoja dvi žinutes: m_1=„herbas“, m_2=„skaičius“

   $$c_1 \equiv m_1^{e_A} \pmod{p}, \quad c_2 \equiv m_2^{e_A} \pmod{p}$$

   ir siunčia B.
3. B pasirenka iš atsiųstųjų vieną žinutę c_i ir ją šifruoja $c_i' \equiv c_i^{e_B} \pmod{p}$, tada siunčia A.
4. A c_i' dešifruoja su raktu d_A $c_i^* \equiv (c_i')^{d_A} \pmod{p}$, išsaugo c_i^*, o taip pat pasiunčia B.
5. B iššifruoja $m_i \equiv (c_i^*)^{d_B} \pmod{p}$ ir praneša A, kas „iškrito“. Kad A galėtų patikrinti, ar B nemeluoja, B siunčia A savo raktą d_B.

Galima telefonu (arba elektroniniu paštu) netgi organizuoti lošimą kortomis, garantuojant, kad dalyviai negalės sukčiauti. Tokiam lošimui organizuoti reikia tokių dalinių protokolų: kortų dalijimo, kortų atskleidimo, lošimo korektiškumo tikrinimo.

Protokolui panaudosime tą pačią Pohligo-Hellmano simetrinę kriptosistemą. Iš pradžių A ir B turi susitarti dėl bendro pirminio skaičiaus p ir pasirinkti po porą raktų: $K_A = \langle e_A, d_A \rangle$, $K_B = \langle e_B, d_B \rangle$.

Kortų dalijimas

Tikslas: A ir B reikia duoti po k atsitiktinai parinktų kortų.

1. A užšifruoja žinutes-kortų pavadinimus $m_1, m_2, \ldots, m_n$, surašytus atsitiktine tvarka, ir siunčia B šifrus $c_i \equiv m_i^{e_A} \pmod{p}$.

2. B atrenka k skaičių c_i ir nusiunčia A. A iššifruoja c_i ir žino, kokios kortos jam teko.

3. B atrenka dar k kortų, jas užšifruoja: $c_i' \equiv c_i^{e_B} \pmod{p}$ ir siunčia A.

4. A iššifuoja gautas žinutes $c_i^* \equiv (c_i)'^{d_A} \pmod{p}$ ir siunčia atgal B.

5. B iššifruoja $m_i \equiv (c_i^*)^{d_B} \pmod{p}$ ir žino, kokios kortos B teko.

Protokolas pasibaigė – abu lošėjai gavo po k kortų, kokias kortas gavo priešininkas – nežino. Tačiau kiekvienas žino priešininko kortų šifrus. Galima pradėti lošti, t. y. vieną po kitos atskleisti kortas.

Kortos atskleidimas

Kai A daro ėjimą (atskleidžia kortą), siunčia partneriui porą $\langle m_i, c_i \rangle$, $c_i \equiv m_i^{e_A} \pmod{p}$. Jis negali pasiųsti tokios kortos, kurios negavo, nes visi jo kortų šifrai žinomi B. Tačiau kol kas B negali patikrinti, ar c_i tikrai yra m_i šifras.

Analogiškai atskleisdama savo kortas elgiasi ir B.

Lošimo korektiškumo tikrinimas

Kai lošimas baigiasi, partneriai turi pasikeisti raktais, kad galėtų patikrinti, ar buvo lošta tomis kortomis, kurias jie turėjo.

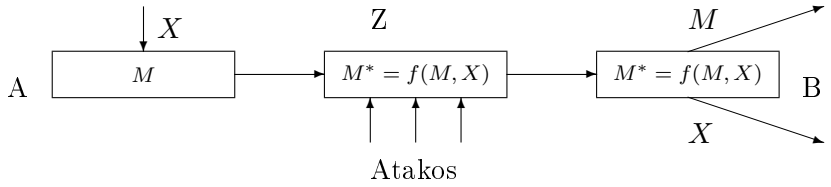
25 Quo vadis?

Kokia bus XXI amžiaus kriptografija? O gal jos iš viso nebereikės? Veltui spėliotume. Geriau pabandykime aptarti dvi duomenų apsaugos kryptis, kurios, be abejonės, bus plėtojamos XXI amžiaus kriptografijoje. Viena iš jų iš tiesų labai sena, senesnė už pačią civilizaciją. Na, o kita – vos dviejų dešimčių metų amžiaus.

25.1. Slėpimo menas

Žuvys, paukščiai ir žvėrys... – amžių amžius trunkanti kova už būvį išmokė juos pasislėpti, kad nepastebėtų priešai. Informacijos srautų judėjimo nepaslėpsi, tačiau galima jais pasinaudoti tarsi konteineriais su dvigubu dugnu ir perduoti nepastebėtus duomenis.

Jeigu kriptoanalitikas Z yra ne šiaip pasyvus kanalo stebėtojas, bet turi galią jį valdyti (pavyzdžiui, jeigu jūsų įstaigos darbuotojų elektroninis paštas yra kontroliuojamas), tai šifrai nepadės. Įtartino turinio siuntiniai bus tiesiog nesiunčiami. Tokiu atveju pranešimui apie negeroves pasiųsti galite pasinaudoti steganografija. *Steganos* graikiškai reiškia slėpti. Kriptografiniais metodais duomenys yra transformuojami, kad būtų paslėpta jų prasmė, tačiau pats duomenų siuntimas yra tikras faktas. Steganografiniais metodais slapta siunčiami duomenys paslepiami kituose, kurie kanalo kontrolieriams neturi kelti įtarimų. Taigi steganografija – tai sritis, kuri siūlo būdus, kaip vienus duomenis panaudoti kaip konteinerį kitiems duomenims siųsti.



Slapto pranešimo X steganografinio perdavimo schema.

Steganografiniai metodai buvo vienaip ar kitaip naudojami visuomet. Juos galima sąlyginai suskirstyti į dvi grupes: lingvistinius ir techninius. Lingvistiniai metodai kaip slapto pranešimo perdavimo konteinerį naudoja kalbą. Tekstai suteikia daug galimybių atlikti nedidelius pakeitimus, šitaip paslepiant reikalingus duomenis. Tarpų tarp žodžių skaičius, kiek nelygiai išdėstytos raidės – visa tai gali būti panaudota duomenims paslėpti.

F. L. Baueris savo knygoje „Decrypted secrets“ pateikia 1976 metais tuometinėje Vokietijos Demokratinėje Respublikoje išleisto kombinatorinės logikos vadovėlio puslapio nuotrauką. Gerai įsižiūrėję galime pastebėti, kad kai kurios raidės yra šiek tiek žemiau negu kitos (spausdinant mechanine spausdinimo mašinėle tai įprastas dalykas). Surašius visas tokias raides, galima perskaityti *„nieder mit dem sowjetimperialismus“* (šalin tarybinį imperializmą).

Viktorijos laikų Anglijoje buvo galima siuntinėti laikraščius paštu nemokamai. Žmonės sugalvojo pasinaudoti tuo žinioms perduoti: pakanka virš laikraščio straipsnių raidžių adata pradurti skylutes, kad, skaitant šitaip pažymėtas raides, susidarytų perduoti skirtas tekstas. Toks metodas buvo naudojamas netgi II pasaulinio karo metais ir po jo. Tik adatos skylutes pakeitė nematomo rašalo taškeliai.

Techninės steganografijos metodai duomenims paslėpti naudoja specialias technines priemones arba įrankius. Škotijos karalienei Marijai skirti laiškai buvo perduodami alaus bokaluose. Šio steganografinio metodo netobulumas irgi buvo viena iš priežasčių, dėl kurių karalienė buvo pasmerkta nukirsdinti.

Nematomas rašalas, mikrofilmai – kur kas geresnės priemonės, bet ir dėl jų daug kas nukentėjo.

Mūsų laikų steganografija – vienų skaitmeninių duomenų slėpimo kituose metodai. Kokie duomenys kaip konteineriai geriausiai tinka steganografijai? Žinoma, tokie, kuriuose nedidelio skaičiaus bitų pokyčiai sunkiai pastebimi – grafikos, audio- ir videoduomenys. Kiekvienam nuotraukos taškui (pikseliui) koduoti naudomi 24 bitai. Jei pakeisime vieną ar kelis jų – akis to nepastebės, tačiau tie pakeistieji gali būti panaudoti slaptam pranešimui (tekstui, vaizdui, garsui) perduoti. Yra visokių metodų ir tuos metodus realizuojančių programų.

Steganografinio metodo vertę lemia trys jo savybės: aptinkamumas (detectability, angl.), atsparumas (robustness, angl.), informacinė talpa (bit rate, angl.).

Steganografinis metodas yra tuo patikimesnis, kuo sunkiau, turint konteinerį (duomenis, kurie gali būti panaudoti kitiems paslėpti), nustatyti, ar jame yra kas nors, ar jis „tuščias“, t. y. ar jame paslėpti kiti duomenys, ar ne. Paslėptus duomenis galime interpretuoti kaip „triukšmą“, padidinantį entropiją. Atlikdamas konteinerio analizę, analitikas gali vertinti, ar turimų duomenų dydis yra būdingas tokio tipo duomenims. Pavyzdžiui, jeigu grynos spalvos 300×300 taškų kvadratas, kuriam saugoti jpg formatu turi užtekti 2 kilobaitų duomenų, siunčiamas kaip dvigubai daugiau duomenų užimantis failas, tai natūralu įtarti, kad kvadratas yra konteineris, kuriame kažkas yra.

Analitikas gali atlikti įtariamo konteinerio ataką, siekdamas jį „iškratyti“. Pavyzdžiui, paveikslą galima pasukti, pakeisti mastelį, atlikti kitas transformacijas, mažai keičiančias jo išvaizdą. Ar paslėpti duomenys nedings? Paslėptų duomenų savybė išsilaikyti po tam tikrų konteinerio modifikacijų ir yra metodo atsparumas.

Taip pat svarbu, kiek bitų įmanoma paslėpti, kad steganografinio metodo panaudojimą dar būtų sunku pastebėti. Slepiamų bitų kiekio ir konteinerio bitų kiekio santykiu galima nusakyti informacinę metodo talpą.

Steganografijai dera priskirti ir vandens ženklų (watermarks, angl.) kūrimo uždavinį. Vandens ženklas – tai liudijimas apie autorystę. Vandens ženklą popieriuje galime pamatyti pažvelgę į jį prieš šviesą. Skaitmeninis vandens ženklas – tai tam tikra informacija, siejama su duomenimis (kūriniu), kuri netrukdo juos naudoti nustatytiems tikslams (klausyti muzikinio kūrinio ar žiūrėti filmą), tačiau ji gali būti registruojama naudojant specialius analizės metodus. Tai šiek tiek panašu į skaitmeninius parašus, tačiau parašai paprastai pridedami kaip priedas (ir todėl gali būti nesunkiai pašalinami), o vandens ženklas turi būti „išlietas“ visame kūrinyje.

25.2. Kvantai ir bitai

Ženklai ant popieriaus, duomenys, perduodami elektros srovės impulsais, elektromagnetinėmis ar garso bangomis – visi šie kanalai turi

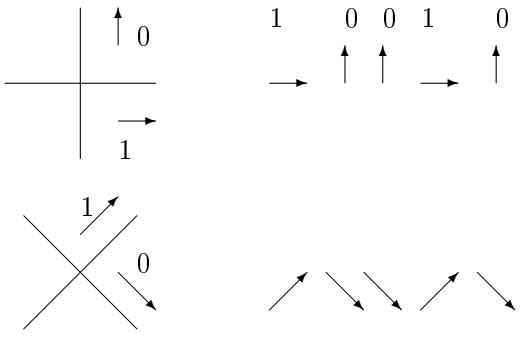
vieną bendrą savybę: duomenis galima nukopijuoti nepakeičiant originalų. Visai kas kita, jeigu duomenų bitus „neša“ fotonai...

Įsivaizduokime fotonus kaip nedalomas šviesos daleles. Idėja, kad reiškinių struktūros pagrindas – mažos dalelės, sena kaip mūsų civilizacija. Prisiminkime, kaip pasaulį aiškino Demokritas... Tačiau šviesos dalelės – fotonai – skiriasi nuo visų dalelių, kurias žmonės įsivaizdavo, tuo, kad jų elgesį valdo visai kitokie dėsniai. Tų dėsnių sąvadas dėstomas kvantinėje mechanikoje. Viena svarbiausių fotonų savybių yra tokia: negalima „išmatuoti“ fotonų nepakeičiant jų savybių. Fotonas prieš matavimą ir po jo – nebe toks pats!

Pirmąsias kvantinės mechanikos dėsnių panaudojimo kriptografijai idėjas išdėstė S. Wiesneris maždaug apie 1970 metus. Tačiau tuomet tos idėjos atrodė, matyt, pernelyg „beprotiškos“. S. Wiesnerio straipsnis buvo paskelbtas tik po gero dešimtmečio. Ir greitai sulaukė tęsinio: 1984 metais Ch. Bennettas (IBM) ir G. Brassard'as (Monrealio universitetas) pasiūlė pirmąjį kvantinės kriptografijos protokolą (pirmoji publikacija [6]). Kriptografijoje jis vadinamas BB84 protokolu. Šiame skyrelyje pabandysime išsiaiškinti, kaip jis vykdomas. Pamatysime, kad kvantinės kriptografijos rūpesčiai visai kitokie negu klasikinės.

Tačiau iš pradžių aptarkime, kaip bitams transportuoti galima panaudoti fotonus. Žinoma, teks labai supaprastinti fizinę realybę, tačiau mums reikia tik tų jos savybių, kurios panaudojamos kriptografijoje.

Pirmiausia įsivaizduokime fotoną kaip dalelę, kurią, naudojant tam tikrų filtrų sistemą, galima poliarizuoti, t.y. suteikti vieną iš dviejų savybių. Naudosime dvi tokių filtrų sistemas, kurias vadinsime atitinkamai + baze ir $\times$ baze. Fotonų poliarizacijas, kurias suteikia + bazė, žymėsime $\uparrow$ ir $\rightarrow$, o $\times$ bazės poliarizacijas – $\nearrow$ ir $\searrow$. Jeigu fotonui poliarizuoti naudota + bazė, sakysime, kad fotonas su poliarizacija $\uparrow$ „neša“ 0, o fotonas $\rightarrow$ „neša“ 1. Atitinkamai $\times$ bazės poliarizacijoms $\searrow$ ir $\nearrow$ irgi priskirkime 0 ir 1.



Tą patį bitų srautą gali „pernešti“ abiejų bazių poliarizuoti fotonai.

Tarkime, A nori perduoti bitų srautą B, pasinaudodamas poliarizuotais fotonais. Mums nesvarbi tikroji fizinė tų fotonų sklidimo terpė – jie

gali sklisti erdve, gali sklisti optiniu kabeliu, svarbu, kad sklistų. Tarkime, visiems fotonams poliarizuoti A pasinaudojo + baze. Gautų fotonų poliarizacijai matuoti turi būti naudojama viena iš dviejų bazių. Tarkime, B visus fotonus registruoja su ta pačia + baze. Visi jo poliarizacijų matavimai bus teisingi ir B gaus tą bitų srautą, kurį siuntė A. Tačiau toks ryšio kanalas niekuo ne geresnis už įprastinius nesaugius kanalus. Kriptoanalitikas Z, žinodamas, kad A ir B fotonams matuoti naudoja tą pačią + bazę, irgi ja pasinaudos: registruos visus A siunčiamus fotonus, užsirašys gautus bitus, o tada su ta pačia + baze generuos tokį pat fotonų srautą ir nusiųs B. Taigi Z bus nepastebimai įsiterpęs į ryšio kanalą, kaip paprastai būna klasikinėje kriptografijoje.

Aptarkime, kokio rezultato galima laukti, jeigu A poliarizavo fotoną su + baze, o B registravo su × baze. Tarkime, A pasiuntė fotoną su poliarizacija →, t.y. nori perduoti bitą, kurio reikšmė 1. Čia ir prasideda visas įdomumas: kvantinės mechanikos dėsniai sako, kad su vienodomis tikimybėmis bus registruotas fotonas su poliarizacijomis ↘ ir ↗ . Kitaip tariant, jeigu B nepataikė pasirinkti bazės, tai matavimo rezultatas bus toks pat, koks būtų paprasčiausiai metant simetrinę monetą!

O dabar jau žinome iš esmės viską, ko reikia, kad suprastume, kaip veikia kvantinės kriptografijos BB84 protokolas. Jo tikslas – perduoti tam tikrą bitų kiekį taip, kad siuntėjas A ir gavėjas B būtų užtikrinti, kad tie bitai nebuvo nukopijuoti Z. O tada galima iš perduotų bitų sudarytą žodį naudoti kaip įprastinės simetrinės kriptosistemos raktą. Juk geros simetrinės kriptosistemos yra labai saugios, jeigu saugiai perduodami raktai!

Kvantinis bitų perdavimo protokolas BB84

1. A rengiasi perduoti B bitų srautą $x_1x_2\ldots x_n$. A atsitiktinai pasirenka bazių seką $a_1a_2\ldots a_n$ ir perduoda B fotonus: bitą x_i „neša“ fotonas, poliarizuotas bazėje a_i.

2. B atsitiktinai pasirenka bazių seką $b_1b_2\ldots b_n$ ir, naudodamasis šiomis bazėmis, matuoja gaunamų fotonų poliarizacijas: i-asis fotonas matuojamas bazėje b_i. Pagal matavimų rezultatus B sudaro bitų seką $y_1y_2\ldots y_n$.

3. B susisiekia su A nesaugiu kanalu ir nurodo, kokią bazių seką $b_1b_2\ldots b_n$ ji buvo pasirinkusi matavimams.

4. A nurodo, kurie pasirinkimai buvo teisingi.

5. Jeigu $i_1, i_2, \ldots, i_k$ yra teisingai pasirinktų bazių numeriai, tai B sudaro iš atitinkamų bitų seką $\mathbf{y} = y_{i_1}y_{i_2}\ldots y_{i_k}$.

6. A ir B susitaria dėl tam tikro kiekio sekos $\mathbf{y}$ bitų, kuriuos reikia patikrinti: ar $x_{i_j} = y_{i_j}$? Jeigu visi jie sutampa, A ir B nutaria, kad Z nebuvo įsiterpęs į fotonų perdavimo procesą, ir, sutrumpinę $\mathbf{y}$,

išbraukdami tikrinant panaudotus bitus, naudojasi gautuoju žodžiu kaip saugiai perduotu raktu.

A siunčiami bitai	0	1	1	0	1	1	0	1	0	0
A bazės	+	×	×	+	×	+	×	+	×	×
A siunčiami fotonai	↑	↗	↗	↑	↗	→	↘	→	↘	↘
B bazės	+	+	×	+	+	+	×	×	+	×
B registruoti fotonai	↑	↑	↗	↑	→	→	↘	↘	→	↘
B registruoti bitai	0	0	1	0	1	1	0	0	1	0
Bendros bazės	+		×	+		+	×			×
Perduoti bitai	0		1	0		1	0			0
Tikrinami bitai	0			0						
Rakto srautas			1			1	0			0

BB84 protokolo pavyzdys: A saugiai perdavė B keturis bitus.

Kaip į šį protokolą gali įsiterpti Z? Jis gali įrengti savo filtrus ir matuoti fotonų poliarizaciją. Tačiau jis nežino, kokias bazes pasirinko A. Jeigu A fotonui poliarizuoti pasirinko + bazę ir Z pasirinko tą pačią bazę, tai, išmatavęs fotoną, jis pasiųs lygiai taip pat poliarizuotą fotoną, kokį gavo. Taigi tokiu atveju jo įsiterpimas tikrai liks nepastebėtas.

Dabar tarkime, kad A poliarizavo fotoną su + baze ir suteikė jam reikšmę 1. Jeigu Z matavimui naudos × bazę, tai jis gali registruoti tiek 1, tiek 0. Registravęs fotoną, jis lygiai tokį pat pasiųs B. Dabar B eilė pasirinkti bazę. Jeigu B pasirinks × bazę, tai trečiajame protokolo žingsnyje jam bus pasakyta, kad pasirinkimas klaidingas, ir šio žingsnio bitas bus išbrauktas. Taigi Z iš savo užrašų neturės jokios naudos. Jeigu B pasirinko tokią pat bazę kaip A, tai B įtrauks registruotą bitą į gautų bitų žodį. Jo reikšmė bus tokia pati, kokią jam suteikė Z, taigi tikimybė, kad ta reikšmė sutaps su A suteikta reikšme, yra $\frac{1}{2}$. Štai dėl ko reikalingas dalies bitų tikrinimas. Jeigu jo nebūtų, galėtų susidaryti tokia padėtis, kad B registruotų ne tokį pat bitų rinkinį, kokį siuntė A. Žinoma, jeigu tas bitų rinkinys bus naudojamas kaip simetrinės kriptosistemos raktas, apgaulė greitai išryškėtų – šifro, kurį sudarė B, A negalėtų iššifruoti. Tačiau verčiau užbėgti tokiai padėčiai už akių: jeigu bent vienoje tikrinamų bitų poroje jie nesutampa – Z buvo įsiterpęs ir protokolas nedavė rezultato. Jokio rezultato, žinoma, nepasiekė ir Z.

Štai toks protokolas ženklina kvantinės kriptografijos pradžią. Jau ir dabar yra praktinių jos taikymų.

O kas bus toliau? Nekantriai laukiame tęsinio...

26 Pastabos ir nuorodos

Kodavimo teorijai vos penkiasdešimt metų, o štai kriptografija – tūkstantmetė. Išsamiausią jos istoriją parašė D. Kahnas [42]. Daug istorinių faktų, o taip pat klasikinės (ir moderniosios) kriptografijos ir kriptoanalizės idėjų dėstoma F. L. Bauerio knygoje [5]. Kriptografijos istorija įdomiai dėstoma S. Singho knygoje [71]. Jeigu norėtumėte paskaityti apie kriptografijos istoriją lietuviškai – galiu pasiūlyti tik savo knygelę [72].

O dabar nebe apie kriptografijos istoriją, bet apie mokslą. Mokslinių kriptografijos metodų pradininkai – W. Friedmanas ir C. Shannonas. Galima teigti, kad C. Shannono darbu [69] prasidėjo kriptologijos matematinių metodų raida.

Šiuolaikinės kriptologijos uždavinius kelia informacinių technologijų naudojimo praktika. Duomenų perdavimas kompiuteriniais tinklais – štai pagrindinis uždavinių šaltinis. Svarbus įvykis kriptografijoje – pirmojo šifravimo standarto patvirtinimas [58]. DES (Data Encryption Standard) buvo intensyviai tyrinėjamas kone ketvirtį amžiaus, sukurta naujų svarbių kriptoanalizės metodų (skirtuminė kriptoanalizė [9], tiesinė kriptanalizė [49] ir kt.), tačiau šio standarto pakeitimo nauju priežastis – ne surastos kriptosistemos spragos, bet išaugusi skaičiavimo technikos galia. Po ilgai trukusio vertinimo proceso naujasis standartas įsigaliojo 2001 metais [2]. Kriptosistemos autoriai – belgų kriptografai – analizuoja šifro struktūrą savo knygoje [19]. Vykdomi nauji kriptosistemų vertinimo ir standartizavimo projektai, žr. pvz., NESSIE (New European Schemes for Signatures, Integrity and Encryption) projekto medžiagą [59], taip pat srautinių kriptosistemų tyrimo projektą eSTREAM [26].

Kriptologijos straipsniai skelbiami įvairiuose matematikos ir informatikos žurnaluose. Yra keletas specialių leidinių: leidžiamas atskiras kriptologijos istorijai, raidai ir įtakai skirtas žurnalas „Cryptologia“. „Journal of Cryptology“ – tikriausiai pagrindinis žurnalas, skirtas kriptografijos profesionalams. Nuo 2007 metų pradedamas leisti dar vienas leidinys „Journal of Mathematical Cryptology“, tai turėtų būti aukšto teorinio lygio straipsnių žurnalas. Tarptautinė organizacija IACR (International Association for Cryptologic Research) kasmet organizuoja kelias kriptologijai skirtas konferencijas (Crypto, Eurocrypt, Asiacrypt), jų darbai išleidžiami atskirais tomais. Skelbiama daug skaitmeninių publikacijų, pvz., portale „Cryptology ePrint-Archive“ (http://eprint.iacr.org/).

Viešojo rakto kriptosistemų idėjas pirmieji paskelbė W. Diffie, M. E. Hellmanas [22] ir nepriklausomai nuo jų R. Merkle [53]. R. Merkle ir M.E. Hellmanas sukūrė pirmąją viešo rakto kriptosistemą – „kuprinės“ kriptosistemą [54]. Vėliau paaiškėjo, kad jų schema nėra saugi. Viešojo rakto sistemą, kurios saugumo spragų nerasta iki šiol, pirmieji paskelbė R. Rivestas, A. Shamiras, L. Adlemanas [64]. RSA kriptoanalizės rezultatai apžvelgti straipsnyje [11].

1997 metais paskelbti faktai rodo, kad viešojo rakto kriptografijos idėjos kilo anglų slaptosioms tarnyboms dirbusiems kriptografams J. Ellisui, C. Cocksui, M. Williamsonui, žr. pav. [16].

Viešojo rakto kriptosistemoms tik 30 metų. Tačiau tai buvo sparčios raidos metai. Parašyta daug išsamių knygų, skirtų šiuolaikinės kriptografijos raidai. Visų pirma reikia paminėti pirmąjį išsamų naujųjų laikų kriptografijai skirtą B. Schneierio veikalą [70], o taip pat – solidų žinyną [52], kurio skaitmeninės kopijos platinamos laisvai. Parašyta ir vadovėlių, ir specialioms kriptografijos kryptims skirtų monografijų; paminėsime keletą, [18], [32], [46], [45], [79], [55], [73], [60], [74].

Praktiškai diegiant kriptografinius apsaugos metodus, pravers knyga, skirta kriptografinių schemų standartams [20].

Bibliografija

[1] N. Abramson. Information Theory and Coding. New York, McGraw-Hill, 1963.

[2] Advanced encryption standard (AES) FIPS-Pub. 197. November 26, 2001.

[3] M. Agrawal, N. Kayal, N. Saxena. PRIMES is in P. Annals of Mathematics, 160 (2004), p. 781–793.

[4] R. B. Ash. Information theory. New York: Dover 1990.

[5] F. L. Bauer. Decrypted secrets. Springer, 1997.

[6] C.H. Bennett, G. Brassard. Quantum cryptography: Public-key distribution and coin tossing. Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, Bangalore, India, December 1984, p. 175 – 179.

[7] E. Berlekamp. Nonbinary BCH decoding. In: Proc. Int. Symp. on Info. Th. San Remo, Italy, 1967.

[8] C. Berrou, A. Glavieux, P. Thitimajshima. Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes. In: Proc. 1993 IEEE Int. Conf. on Communications, Geneva, Switzerland, 1993, p. 1064–1070.

[9] E. Biham, A. Shamir. Differential Cryptanalysis of the Data Encryption Standard. Springer Verlag, 1993.

[10] J. Bierbrauer, L. Kelly. Introduction to Coding Theory. CRC Press, 2005.

[11] D. Boneh. Twenty Years of Attacks on the RSA Cryptosystem. Notices of the AMS, vol. 46, 2, 1999, p. 203–213.

[12] R. Bose, D. Ray-Chaudhuri. On a Class of Error-Correcting Binary Codes. Inf. and Control, vol. 3, 1960, p. 68–79.

[13] M. Burrows, D. Wheeler. A block-sorting losseless data compression algorithm. Digital Systems Research Center report 124, Palo Alto, 1994.

[14] A. R. Calderbank. The art of signaling: fifty years of coding theory. IEEE Trans. Information Theory, 1998, IT-44 (6), p. 2561–2595.

[15] P. J. Cameron, J. H. van Lint. Graphs, codes and designs. Cambridge University Press, 1980.

[16] C. Cocks. A Note on Non-Secret Encryption. 1973. http://www.cesg.gov.uk/site/publications/index.cfm

[17] T. Cover, J. Thomas. Elements of Information Theory. New York: Wiley-Interscience, 1991, 2006.

[18] R. Crandall, C. Pomerance. Prime numbers. A computational perspective. Springer, 2001.

[19] J. Daemen, S. Borg, V. Rijmen. The Design of Rijndael: AES - The Advanced Encryption Standard. Springer-Verlag, 2002.

[20] A. Dent, C. Mitchell. User's Guide to Cryptography and Standards. Artech House, 2004.

[21] S. X. Descamps. A computational primer on block error-correcting codes. Springer, 2003. http://www.wiris.com/cc/

[22] W. Diffie, M. E. Hellman. New directions in Cryptography. IEEE Transactions on Information Theory, v. IT-22, n. 6, 1976, p. 644–654.

[23] Duomenų spūda.
Tinklalapis: http://www.data-compression.info/index.htm

[24] Elektroninis žurnalas „Entropy". http://www.mdpi.org/entropy/

[25] P. Elias. Coding for Noisy Chanels. IRE Conv. Rep., Pt. 4, p. 37–47, 1955.

[26] eSTREAM. http://www.ecrypt.eu.org/stream/

[27] N. Faller. An adaptive system for data compression. Record of the 7th Asilomar Conference on Circuits, Systems and Computers, p. 593–597.

[28] R. Gallagher. Variations on a theme by Huffman. IEEE Transactions on Information Theory, 1978, IT-24 (6), November, p. 668–674.

[29] R. Gallagher. Claude E. Shannon: A retrospective on his life, work, and impact. IEEE Trans. Information Theory, 2001, IT-47, p. 2681–2695.

[30] M. R. Garey, D. S. Johnson. Computers and Intractability. A Guide to the Theory of NP-Completeness, W. H. Freeman and Co., 1979.

[31] M. J. E. Golay. Notes on digital coding. Proceedings of the IRE, June 1949.

[32] O. Goldreich. Modern Cryptography, Probabilistic Proofs and Pseudorandomness. Springer, 1999.

[33] GAP - Groups, Algorithms, Programming. http://www-gap.mcs.st-and.ac.uk/

[34] R. M. Gray. Entropy and Information Theory. 2002. http://www-ee.stanford.edu/% gray/it.html

[35] R. W. Hamming. Error detecting and error correcting codes. The Bell System Technical Journal, 1950, No. 2, April, p. 147–160.

[36] R. W. Hamming. Coding and Information Theory. Prentice Hall, 1986.

[37] D. Hankersson et al. Introduction to Information Theory and Data Compression. CRC Press, 1997.

[38] A. Hockuenghem. Codes Correcteurs D'erreurs. Chiffres, vol. 2, 1959, p. 147–156.

[39] D. Huffman. A method for the construction of minimum redundancy codes. Proceedings of the IRE. 1952, 40(9), p. 1098–1101.

[40] W. C. Huffman, V. S. Pless. Fundamentals of error-correcting codes. Cambridge University Press, 2003.

[41] R. V. L. Hartley. Transmission of information. The Bell System Technical Journal, 7(3), 1928, p. 535–563.

[42] D. Kahn. The codebreakers. Scribner, 1967, 1996.

[43] G. J. Klir. Uncertainty and information. Foundations of Generalized Information Theory. Wiley&sons, 2006.

[44] D. E. Knuth. Dynamic Huffman coding. Journal of Algorithms. 1985, 6, p. 163–180.

[45] N. Koblitz. A course in number theory and cryptography. Springer, 1987.

[46] N. Koblitz. Algebraic Aspects of Cryptography. Springer, 1998.

[47] R. Lassaigne, M. Rougemont. Logika ir algoritmų sudėtingumas. Žara, Vilnius 1999.

[48] J. H. van Lint. Introduction to Coding Theory. Springer, 1982, 1999.

[49] M. Matsui. Linear Cryptanalysis Method for DES Cipher. EUROCRYPT 1993, p. 386–397.

[50] R. J. McEliece. The Theory of Information and Coding. Cambridge University Press, 2002.

[51] D. J. C. MacKay. Information Theory, Inference and Learning Algorithms. Cambridge University Press, 2003. http://www.inference.phy.cam.ac.uk/mackay/

[52] A. J. Menezes, P. C. van Oorshot, S. A. Vanstone. Handbook of applied cryptography. CRC Press, 1997. http://www.cacr.math.uwaterloo.ca/hac/

[53] R. C. Merkle. Secure communication over insecure channels. Communications of the ACM, v. 21, n. 4, 1978, p. 294–299.

[54] R. C. Merkle, M. Hellman. Hiding information and signatures in trapdoor knapsacks. IEEE Transactions on Information Theory, v. IT-24, n. 5, 1978, p. 525–530.

[55] R. A. Mollin. RSA and public-key cryptography. Chapman&Hall CRC, 2003.

[56] T. K. Moon. Error Correction Coding. Mathematical methods and algorithms. Wiley, 2005.

[57] D. Muller. Application of Boolean Switching Algebra to Switching Circuit Design. IEEE Trans. on Computers, vol. 3, Sept. 1974, p. 6-12.

[58] National Bureau of Standards, Data Encryption Standard, FIPS-Pub. 46. National Bureau of Standards, U. S. Department of Commerce, Washington D. C., January 1977.

[59] NESSIE, https://www.cosic.esat.kuleuven.be/nessie/

[60] R. Oppliger. Contemporary cryptography. Artech House, 2005.

[61] E. Prange. Cyclic Error-Correcting Codes in Two Symbols. Air Force Cambridge Research Center, Cambridge, MA, Tech. Rep. TN-58-156, 1958.

[62] I. Reed. A Class of Multiple-Error-Correcting Codes and a Decoding Scheme. IEEE Trans. Information Theory, vol. 4, Sept. 1954, p. 38–49.

[63] I. Reed, G. Solomon, Polynomial Codes over Certain Finite Fields. J. Soc. Indust. Appl. Math., vol. 8, 1960, p. 300-304.

[64] R. Rivest, A. Shamir, L. Adleman. A method of obtaining digital signatures and public-key cryptosystems. Communications of the ACM, v. 21, n. 2, 1978, p. 120–126.

[65] S. Roman. Coding and Information Theory. Springer, 1992.

[66] A. Rukhin et al. A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST Special Publication 800-22, 2001. http://csrc.nist.gov/rng/SP800-22b.pdf

[67] D. Salomon. Data compression. The complete Reference. Springer, 1988, 2000, 2004.

[68] C.E. Shannon. A Mathematical Theory of Communication, The Bell System Technical Journal, XXVII, 1948, N. 3.

[69] C. E. Shannon. Communication theory of secrecy systems. Bell Syst. Tech. J. 28 (1949), p. 656-715.

[70] B. Schneier. Applied Cryptography. Willey, New York 1994.

[71] S. Singh. The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography. Anchor Books, 2000.

[72] V. Stakėnas. Šifrų istorijos. TEV, 2005.

[73] W. Stallings. Cryptography and Network Security. Principles and Practices, Fourth Edition, Prentice Hall, 2005.

[74] D. Stinson. Cryptography Theory and Practice. CRC Press, 1995, 2002, 2005.

[75] P. Sweeney, Error Control Coding: From Theory to Practice. John Wiley and Sons, 2002.

[76] H. van Tilborg. Fundamentals of cryptology. Kluwer, 2002.

[77] A. J. Viterbi. Convolutional Codes and Their Performance in Communication Systems. IEEE Trans. Com. Techn., vol. 19, n. 5, 1971, p. 751–771.

[78] S. Verdu. Fifty years of Shannon theory. IEEE Trans. Information Theory, 1998, IT-44 (6), p. 2057–2078.

[79] S. Wagstaff. Cryptanalysis of number theoretic ciphers. Chapman&Hall CRC, 2002.

[80] J. Ziv, A. Lempel. A universal algorithm for sequential data compression. IEEE Transactions on Information Theory, 1977, IT-24(5), p. 530–536.

Rodyklė