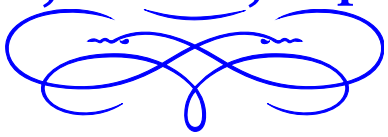
Informacijos kodavimo algoritmai

VILIUS STAKĖNAS

2022

Turinys

Informacijos teorijos pagrindai



Kodai

Žinias norime išsaugoti, perduoti, jomis dalintis. Todėl jas reikia užrašyti sutartiniais ženklais, taigi – koduoti.

Ženklų, kuriais užrašome žinias, aibę vadinsime abėcėle. Taigi abėcėlė – baigtinė simbolių aibė $\mathcal{A} = \{a_1, a_2, \ldots, a_r\}$. Baigtinės aibės elementų skaičių žymėsime rašydami aibės žymenį tarp vertikalių brūkšnelių: $|\mathcal{A}| = r$. Iš abėcėlės simbolių galime sudaryti žodžius:

$$\mathbf{a} = a_{i_1} a_{i_2} \cdots a_{i_n}, \quad a_{i_j} \in \mathcal{A}.$$

n-ilgio žodžių, sudarytų iš abėcėlės $\mathcal{A}$ simbolių aibę žymėsime $\mathcal{A}^n$; tada $|\mathcal{A}^n| = r^n$. Visų žodžių aibę žymėsime $\mathcal{A}^*$; ši aibė yra begalinė:

$$\mathcal{A}^* = \mathcal{A}^1 \cup \mathcal{A}^2 \cup \cdots$$

Perduodami, saugodami duomenis naudojame kokią nors daiktišką, fizinę terpę. Kad galėtume ja pasinaudoti dažniausiai turime perkoduoti informaciją, t.y. išreikšti ją kitos abėcėlės žodžiais. Pavyzdžiui, kad galėtume įrašyti duomenis į skaitmenines laikmenas, turime juos išreikšti dvejetainės abėcėlės $\{0, 1\}$ ženklais. Kad kodavimas nereikštų duomenų praradimo, iš gautų kodo abėcėlės žodžių turi būti galime vienareikšmiškai atkurti pradinius duomenis.

Apibrėžimas. *Tegu $\mathcal{A}, \mathcal{B}$ yra dvi abėcėlės. Kodu vadinsime atvaizdį*

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

Kad duomenys koduojant nebūtų pakeičiami be galimybės juos visada atkurti, atvaizdis turi būti injektyvus. Injektyvumas reiškia, kad jeigu $\mathbf{a}_1, \mathbf{a}_2$ yra du skirtingi aibės $\mathcal{A}^*$ žodžiai, tai $c^*(\mathbf{a}_1), c^*(\mathbf{a}_2)$ irgi skirtingi.

Koduojame ne vien norėdami pasinaudoti informacijos saugojimo ar perdavimo galimybėmis, bet ir norėdami, kad tai vyktų efektyviai, patikimai ir saugiai. Taigi koduodami siekiame:

- taupumo;
- patikimumo;
- apsaugos.

Siekdami taupumo koduojame duomenis taip, kad juos būtų galima užrašyti trumpesniais žodžiais ir šitaip sumažinti saugojimui reikalingą vietą ir perduoti duomenis greičiau. Patikimumas reiškia galimybę pastebėti ir ištaisyti galimas įrašymo ar perdavimo klaidas. Apsauga reiškia, kad duomenų prasmė nebūtų prieinama jais neturintiems teisės naudotis subjektams, taip pat galimybę pastebėti ryšio kanale įvykdytus pakeitimus ir kita.

Taigi duomenų kodavimo uždavinius sprendžia net trys mokslai: informacijos teorija, klaidas taisančių kodų teorija ir kriptografija. Visos jos naudoja daug bendrų sąvokų, tačiau taiko skirtingus požiūrius ir metodus.

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.

Kaip sudaryti kodą?

Kodas yra taisyklė vienos begalinės aibės žodžiams priskirianti kitos begalinės aibės žodžius. Kaip tokią taisyklę galima apibrėžti?

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

Apibrėžkime keletą sąvokų.

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.*

Prisiminkime: $\mathcal{A}$ ilgio n žodžių aibę žymime $\mathcal{A}^n$, 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.

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

Kad galėtume naudotis kodavimo taisykle, turime ją apibrėžti. Kodavimo taisyklių yra be galo daug.

Pasielkime itin praktiškai: nagrinėkime tik tokias kodavimo taisykles, kurias galima apibrėžti nurodžius, kaip koduojami pavieniai abėcėlės $\mathcal{A}$ simboliai.

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

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

vadinsime atvaizdžio c tęsiniu. Abėcėlės $\mathcal{B}$ žodį $c^*(x_1 x_2 \cdots x_n)$ vadinsime žodžio $x_1 x_2 \cdots x_n$ 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):

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

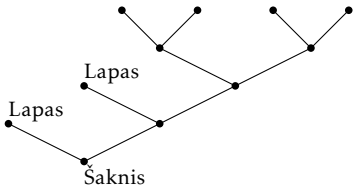
Taigi abėcėlės $\mathcal{A}$ galima ir neminėti; dažnai taip ir darysime – kodu vadinsime tiesiog atitinkamos abėcėlės žodžių gretinį, arba tiesiog žodžių poaibį

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

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**.

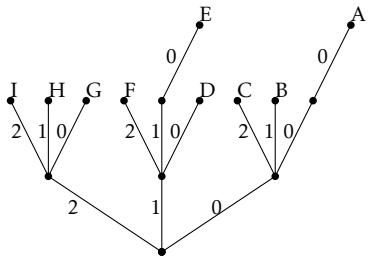


Medį vadinsime **q-nariu**, jei iš kiekvienos viršūnės išeina ne daugiau kaip q šakų; jeigu iš kiekvienos viršūnės, kuri nėra lapas, išeina lygiai q šakų, tai medį vadinsime **reguliariuoju q-nariu**. Jei visų reguliariojo q-nario medžio lapų atstumai iki šaknies yra vienodi, tai medį vadinsime **pilnuoju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

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

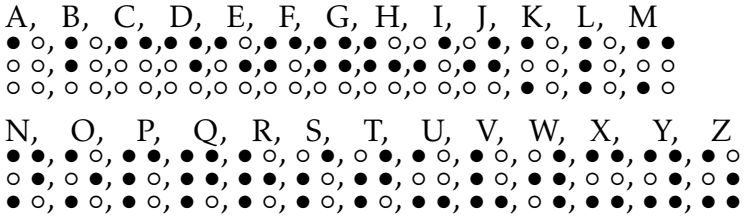
Polibijaus aprašytas ugnies kodas. Norėdami perduoti, pavyzdžiui, raidę β, ryšininkai vienoje kalvos vietoje turėjo iškelti vieną degantį fakelą, o kitoje – du.

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	·–	*j*	·–––	*s*	···
b	–···	*k*	–·–	*t*	–
c	–·–·	*l*	·–··	*u*	··–
d	–··	*m*	––	*v*	···–
e	·	*n*	–·	*w*	·––
f	··–·	*o*	–––	*x*	–··–
g	––·	*p*	·––·	*y*	–·––
h	····	*q*	––·–	*z*	––··
i	··	*r*	·–·		

Morse kodas pritaikytas telegrafui.



Apie 1820 metus Louis Braille sukurtas kodas akliesiems. Iš 3×2 *matricos taškų 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.

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
DC1	0010001	1	0110001	Q	1010001	q	1110001
DC2	0010010	2	0110010	R	1010010	r	1110010
DC3	0010011	3	0110011	S	1010011	s	1110011
DC4	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.

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.

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...

Momentiniai kodai

Jeigu į savo kompiuterį siunčiate suspaustos informacijos archyvą, teks palaukti ryšio pabaigos, kad galėtumėte pradėti skaityti. Jeigu tekstas perduodamas raidė po raidės, kiekvieną žodį atpažįstame, kai tik gauname paskutinę jo raidę. O kartais netgi anksčiau.

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

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

Tai reiškia, kad gavėjas niekada negaus simbolių eilutės, kurią būtų galima suvokti nevienareikšmiškai.

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

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.

Apibrėžimas. *Kodą $c : \mathcal{A} \to \mathcal{B}^*$ vadinsime momentiniu 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.*

Iš tiesų, jeigu kuris nors žodis būtų ilgesnio žodžio priešdėliu, tai gavę jį perduodamų simbolių sraute, dar turėtume palaukti ir įsitikinti, kad jis nėra tik ilgesniojo žodžio pradžia.

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

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*

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

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

$$(b^{-s_1} + \ldots + b^{-s_n})^r = \sum_{l=1}^{rs} N_l b^{-l}. \tag{2}$$

Kokia gi koeficientų N_l prasmė?

Koeficientas N_l lygus skaičiui l ilgio žodžių, 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

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

Dabar iš (2) nelygybės gauname

$$(b^{-s_1} + \ldots + b^{-s_n})^r \leq rs,$$

Ši nelygybė teisinga su visomis r reikšmėmis. Jeigu nelygybė būtų neteisinga, tada su $\epsilon > 0$ turėtume

$$b^{-s_1} + \ldots + b^{-s_n} = 1 + \epsilon$$

ir su visais r

$$(1+\epsilon)^r \le rs.$$

Tačiau $(1+\epsilon)^r > 1+\epsilon r+\epsilon^2 r(r-1)/2$, o nelygybė

$$1+\epsilon r+\epsilon^2 r(r-1)/2 < (1+\epsilon)^r < rs$$

negali būti teisinga su fiksuotais $\epsilon > 0, s \ge 1$ ir visais r.

Dabar jau galime teigti, kad dekoduojamo kodo iš dvinarės abėcėlės žodžių su ilgiais $2;2;2;2;3$ sukonstruoti negalima, nes žodžių ilgiai netenkina (1) sąlygos.

Įsitikinkime, kad (1) nelygybė garantuoja, kad kodas su iš anksto nustatytais žodžių ilgiais $s_1, s_2, \ldots, s_n$ gali būti sukonstruotas.

Teorema. *Tegu $\mathcal{B}$ yra abėcėlė, $b = |\mathcal{B}|$, o $s_1, s_2, \ldots, s_n$ yra natūralieji skaičiai, tenkinantys (1) nelygybę. Tada egzistuoja momentinis 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 (1) nelygybę galime užrašyti taip

$$n_1 b^{-1} + n_2 b^{-2} + \ldots + n_s b^{-s} \le 1. \tag{3}$$

Dauginkime abi (3) nelygybės puses iš $b, b^2, \ldots, b^s$ ir pertvarkykime gautas nelygybes, palikdami kairėje pusėje tik $n_1, n_2, \ldots, n_s$:

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

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

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

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.

(1) nelygybė vadinama Krafto-McMillano nelygybe. L. G. Kraftas 1949 m. ją įrodė momentiniams 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 momentinius kodus. Iš tiesų, bet kokį dekoduojamą kodą galima pakeisti atitinkamu momentiniu kodu, turinčiu tiek pat to paties ilgio žodžių.

Informacijos šaltiniai ir kodai

Nėra programavimo kalbos, kuri visais atvejais būtų geriausia. Nėra ir kodo, kuris geriausiai tiktų visų šaltinių informacijai koduoti. Tačiau kas yra informacijos šaltinis?

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.

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.

Apibrėžimas. *Informacijos šaltiniu vadinsime atsitiktinių dydžių, įgyjančių reikšmes iš tos pačios abėcėlės $\mathcal{A}$, seką*

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

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, taigi 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 momentinius 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.

Apibrėžimas. *Vidutiniu momentinio kodo $c : \mathcal{A} \to \mathcal{B}^*$ žodžių ilgiu vadinsime skaičių*

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

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

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

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

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

$$\lambda(c) = \sum_{i=1}^{r} p_i |c(a_i)| \leqslant t, \tag{4}$$

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 (4) 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

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

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

Sudarant kodą, geriausiai tinkantį šaltiniui, reikia siekti, kad rečiau pasitaikantys simboliai būtų koduojami ilgesniais, o dažniau – trumpesniais žodžiais.

Tarkime, šaltinio tikimybės išrikiuotos didėjimo tvarka:

$$p_1 \leq p_2 \leq \ldots \leq p_r.$$

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

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

Akivaizdu, kad juos galime parinkti vieninteliu būdu. Kadangi

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

tai galima sudaryti momentinį kodą $\hat{c} : \mathcal{A} \to \mathcal{B}^*$, kad $|\hat{c}(a_i)| = s_i$.

Tokie kodai vadinami Shannono kodais. Shannono kodą galime sudaryti 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

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

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.

Naudodami Shannono kodus dažnesnius simbolius koduojame trumpesniais žodžiais, tačiau Shannono kodas ne visada optimalus.

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:

$$c_{r-1} \to c_{r-2} \to \ldots \to c_1; \tag{5}$$

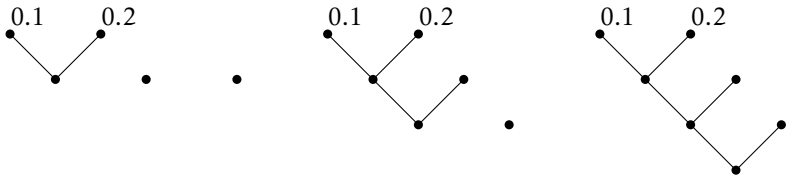
čia c_k – porą $\langle \mathcal{A}_k, P(\mathcal{A}_k) \rangle$ atitinkantis Huffmano kodas.

Abėcėlių redukcijos procese (5) 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 (5) perėjimus, užrašyti.

$\mathcal{A}_1$	c_1	$\mathcal{A}_2$	c_2	$\mathcal{A}_3$	c_3
0.4		0.4		$\underline{0.6}$	
0.3		0.3^*		0.4	
0.2^*		$\underline{0.3^*}$			
0.1^*					

1 lentelė. Huffmano kodo sudarymas

Dabar išsamiau aptarkime (5) procesą, t. y. kodo sudarymą. Jeigu abėcėlių redukciją interpretuojate kaip kodo medžio braižymą, tai kodo sudarymas pagal gautą medį – jau žinomas dalykas. Vis dėlto (5) 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.

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.

Teorema. *Huffmano kodai yra optimalūs.*

Įrodymas. Visi kodai (5) 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

$$\lambda(d) = \sum_{a \in \mathcal{A}_{k-1}} p(a)|d(a)| < \sum_{a \in \mathcal{A}_{k-1}} p(a)|c_{k-1}(a)| = \lambda(c_{k-1}). \tag{6}$$

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 (6) nelygybei, vadinasi, prielaida, kad kodas c_{k-1} nėra optimalus, yra klaidinga.

Teorema įrodyta.

Šaltinio informacijos kiekis

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.

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

$$f(p) = \log_b \frac{1}{p}. \tag{7}$$

Kita vertus, kiekvienam $b > 1$ funkcija $f(p)$, apibrėžta (7), tenkina 1)-4) sąlygas.

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.

Informacijos šaltiniu pavadinome atsitiktinių dydžių seką $\mathcal{U} = \langle U_1, U_2, \ldots \rangle$. Kaip kiekybiškai išreikšti jo informatyvumą? Pradėkime nuo dalinio šaltinio, kurį tapatiname su pirmuoju atsitiktiniu dydžiu U_1, įgyjančiu reikšmes iš baigtinės aibės. Jo reikšmės $U_1 = u$ suteikiamos informacijos kiekis yra $\log_2(1/P(X = u))$. Šaltinio informatyvumą matuosime informacijos kiekių vidurkiu.

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, apibrėžimas tinka ir atsitiktiniams vektoriams, t.y. baigtinėms atsitiktinių dydžių sekoms. Taigi dalinio šaltinio $\mathcal{U}_n = \langle U_1, U_2, \ldots, U_n \rangle$ entropija (suteikiamos informacijos kiekiu) laikysime $H(U_1, U_2, \ldots, U_n)$. Viso šaltinio entropiją apibrėšime taip.

Apibrėžimas. *Jeigu informacijos šaltiniui $\mathcal{U} = \langle U_1, U_2, \ldots$ egzistuoja riba*

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

tai H vadinsime šaltinio entropija.

Nagrinėkime paprasčiausią – monetos mėtymo šaltinį: $U_i = 0$, jei i-ajame metime moneta atvirto herbu į viršų ir $U_i = 1$, jei skaičiumi. Tegu $P(U_i = 0) = p, P(U_i = 1) = q$. Tada

$$H(U_i) = p \log_2 \frac{1}{p} + q \log_2 \frac{1}{q}.$$

Tarkime, dydžiai U_i nepriklausomi,taigi pažymėję $f_p = \log_2(1/p), f_q = \log_2(1/q)$ šaltiniui $\mathcal{U}_2$ gausime

$$\begin{aligned} H(U_1,U_2) &= p^2(f_p+f_p)+2pq(f_p+f_q)+q^2(f_q+f_q) \\ &= 2p(pf_p+qf_q)+2q(pf_p+qf_q) = 2(pf_p+qf_q) = H(U_1)+H(U_2). \end{aligned}$$

Galima taip pat įrodyti, kad

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

Taigi šaltinio entropija

$$H = \lim_{n\to\infty} \frac{H(U_1,U_2,\ldots,U_n)}{n} = p\log_2\frac{1}{p} + q\log_2\frac{1}{q}.$$

Šis šaltinis vadinamas Bernulio šaltiniu.

Tarkime yra du informacijos šaltiniai – du atsitiktiniai dydžiai; abėcėlės yra vienodo dydžio, o simbolių tikimybės atitinkamai

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

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.

Teorema. *Tegu*

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

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

$$\sum_{i=1}^{n} p_i \log_2\frac{1}{p_i} \leqslant \sum_{i=1}^{n} p_i \log_2\frac{1}{q_i}. \tag{8}$$

Iš šios nelygybės gauname tokią išvadą.

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.

Iš tiesų, jei X reikšmių tikimybės yra $p_i, i = 1, 2, \ldots, n$, tai su $q_i = 1/n$ gausime

$$H(X) = \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i} \leqslant \sum_{i=1}^{n} p_i \log_2 \frac{1}{q_i} = \log_2 n.$$

Nagrinėkime du informacijos šaltinius, kuriuos tapatinsime su atsitiktiniais dydžiais X, Y. Tarkime, X įgyja reikšmes $x_1, x_2, \ldots, x_n$, o Y – $y_1, y_2, \ldots, y_m$. Tada

$$H(X,Y) = \sum_{i,j} P(X = x_i, Y = y_j) \log \frac{1}{P(X = x_i, Y = y_j)}.$$

Pritaikę (8) nelygybę su $P(X = x_i, Y = y_j)$ vietoje p_i ir $P(X = x_i)P(Y = y_j)$ bei sutvarkę reiškinius, gautume

$$H(X,Y) \leqslant H(X) + H(Y).$$

Taigi du stebimi informacijos šaltiniai negali suteikti daugiau informacijos kiekio, nei šaltinių atskirai suteikiamų informacijos kiekių suma. Skirtumą

$$H(X,Y) - H(X)$$

galime suvokti, kaip papildomą informacijos kiekį, kurį suteikia šaltinis Y, jeigu jau sužinojome informaciją iš X.

Apibrėžimas. *Tegu X, Y yra atsitiktiniai dydžiai, įgyjantys reikšmes iš baigtinių aibių. Sąlygine Y entropija su sąlyga X vadinsime dydį*

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

Taigi

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

Dydį

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

galime suvokti, kaip tos pačios informacijos, kurią suteikia ir X, ir Y kiekį.

Panagrinėkime pavyzdį. Tarkime, atsitiktinai parenkama eilutės

$$aABBccABbbbbCCCAaBBb$$

raidė. Dydžio X reikšmė – parinktoji raidė, dydis $Y = 0$, jei raidė didžioji, $Y = 0$, jei mažoji. Sudarykime dydžių poros $\langle X, Y\rangle$ reikšmių tikimybių lentelę:

	0	1
A	$\frac{3}{20}$	$\frac{2}{20}$
B	$\frac{5}{20}$	$\frac{5}{20}$
C	$\frac{5}{20}$	0

Taigi

$$\begin{aligned}
H(X,Y) &= 3\cdot\frac{5}{20}\cdot\log_2\frac{20}{5} + \frac{3}{20}\cdot\log_2\frac{20}{3} + \frac{2}{20}\cdot\log_2\frac{20}{2} \\
&= \frac{3}{2} + \frac{3}{20}\log_2\frac{20}{3} + \frac{1}{10}\log_2 10 \approx 2.243, \\
H(X) &= 2\cdot\frac{5}{20}\cdot\log_2\frac{20}{5} + \frac{10}{20}\cdot\log_2\frac{20}{10} = \frac{3}{2}, \\
H(Y) &= \frac{13}{20}\cdot\log_2\frac{20}{13} + \frac{7}{20}\cdot\log_2\frac{20}{7} \approx 0.771.
\end{aligned}$$

Dabar gauname:

$$\begin{aligned}
H(Y|X) &= H(X,Y) - H(X) \approx 0,743, \\
H(X|Y) &= H(X,Y) - H(Y) \approx 1,472, \\
I(X,Y) &= H(X) - H(X|Y) \approx 0,029.
\end{aligned}$$

Šaltinio entropija ir kodo žodžių ilgis

Jeigu šaltinio perduodamos informacijos kiekis yra 4 *bitai, tai pageidautume, kad koduojant jo simbolius dvejetainės abėcėlės žodžiais vidutinis žodžių ilgis irgi būtų bent apytiksliai lygus* 4. *Tokį pageidavimą iš tiesų galima patenkinti.*

Nagrinėkime informacijos šaltinį $\mathcal{U} = \langle U_1, U_2, \ldots \rangle$, čia U_i yra atsitiktiniai dydžiai, įgyjantys reikšmes iš aibės (abėcėlės) $\mathcal{A} = \{a_1, a_2, \ldots, a_r\}$. Jeigu dydžiai U_i yra nepriklausomi ir tikimybės $P(U_i = a_j)$ nepriklauso nuo i, tokį šaltinį vadinsime Bernulio šaltiniu.

Tarkime, egzistuoja šaltinio entropija

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

Jeigu šaltinis yra Bernulio, tai $H = H(U_1)$.

Šaltinį koduosime abėcėlės $\mathcal{B}, |\mathcal{B}| = b$, žodžiais. Iš pradžių nagrinėkime dalinį šaltinį $\mathcal{U}_1$, kurį galime tiesiog tapatinti su atsitiktiniu dydžiu U_1.

Teorema. *Kiekvienam dekoduojamam šaltinio $\mathcal{U}_1$ kodui $c : \mathcal{A} \to \mathcal{B}^*$ teisinga nelygybė*

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

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 $\lambda(c) \geq H(U_1)$ 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 $\ln x\leqslant x-1$ $(x>0)$, gausime

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

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

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

o tikimybių suma lygi vienetui. Taigi gavome

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

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?

Teorema. *Egzistuoja šaltinio $\mathcal{U}_1$ kodas $\hat{c}:\mathcal{A}\to\mathcal{B}^*$, kuriam teisinga nelygybė*

$$\lambda(\hat{c})<\frac{H(U_1)}{\log_2 b}+1. \tag{9}$$

Įrodymas. Tarkime, šaltinio tikimybės išrikiuotos didėjimo tvarka:

$$p_1\leqslant p_2\leqslant\ldots\leqslant p_r.$$

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

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

Akivaizdu, kad juos galime parinkti vieninteliu būdu. Tai jau mums žinomo Shannono kodo žodžių ilgiai. Kadangi

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

tai galima sudaryti momentinį kodą $\hat{c} : \mathcal{A} \to \mathcal{B}^*$, kad $|\hat{c}(a_i)| = s_i$. Įsitikinkime, kad šiam kodui teisinga (9) 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:

> ***Teorema.*** *Optimalus šaltinio $\mathcal{U}_1$ kodas tenkina nelygybę*
>
> $$\frac{H(U_1)}{\log_2 b} \leqslant \lambda(\hat{c}) < \frac{H(U_1)}{\log_2 b} + 1. \tag{10}$$

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.

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:

$$c_n(x_1 x_2 \ldots x_n) = c(x_1)c(x_2)\ldots c(x_n), \quad x_j \in \mathcal{A}. \tag{11}$$

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

$$\frac{H(U_1)}{\log_2 b} \leqslant \bar{\lambda}(c_n) < \frac{H(U_1)}{\log_2 b} + 1. \tag{12}$$

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 įrodytas teoremas, gausime, kad egzistuoja kodas $c_n : \mathcal{A}^n \to \mathcal{B}^*$, tenkinantis sąlygą

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

arba

$$\frac{H(U_1, U_2, \ldots, U_n)}{n\log_2 b} \leqslant \bar{\lambda}(c_n) < \frac{H(U_1, U_2, \ldots, U_n)}{n\log_2 b} + \frac{1}{n}. \tag{13}$$

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 (13) gauname tokį teiginį:

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*

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

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

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

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*

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

Kintamų kodų metodas

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

Huffmano 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

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

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

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

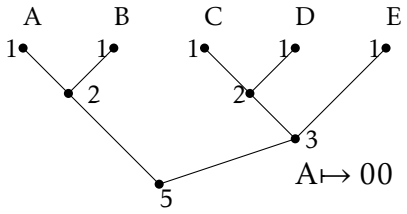
Tada, naudodamiesi šiomis tikimybėmis, galėtume koduoti duomenis Huffmano arba aritmetiniu kodu, tikėdamiesi, kad šitaip jie bus gerai suspausti. Darbo koduojant šiuo būdu padaugėja – juk reikia iš pradžių sudaryti simbolių dažnių lentelę! Kita aplinkybė – kartu su suspaustais duomenimis gavėjui turime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

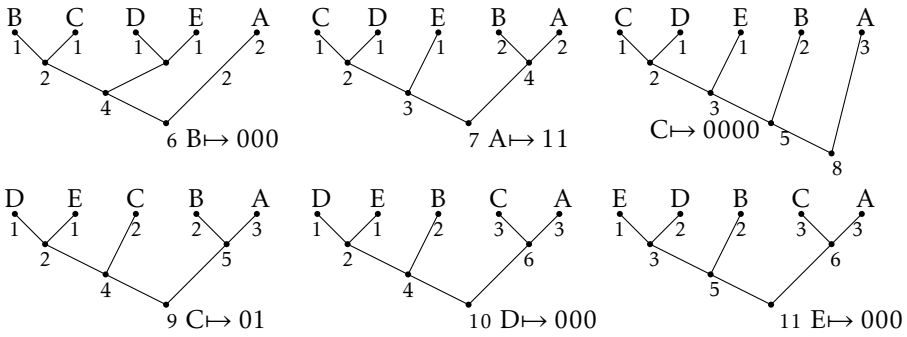
- abėcėlės simbolius visada išdėstysime dažnių didėjimo tvarka iš kairės į dešinę;
- konstruojant kodo medį, jungiant mažiausius dažnius turinčių viršūnių porą, visada bus jungiami patys „kairiausieji“ elementai; sujungus viršūnių porą, gautoji viršūnė bus vaizduojama vienu lygiu žemiau už žemesniojo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

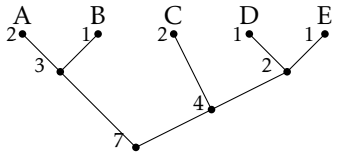


Brėžinyje parodyta, kaip keitėsi kodo medis, koduojant žodį ABACCDE. Pirmajai raidei koduoti buvo panaudotas medis, atitinkantis vienodus abėcėlės simbolių pasirodymo dažnius.

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!).

Žodyno metodai

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 lango 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|VASA|RA 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:

VASAR|IS VASAR|A PA|VASARIS

Visa teksto VASARIS VASARA PAVASARIS eiga parodyta lentelėje

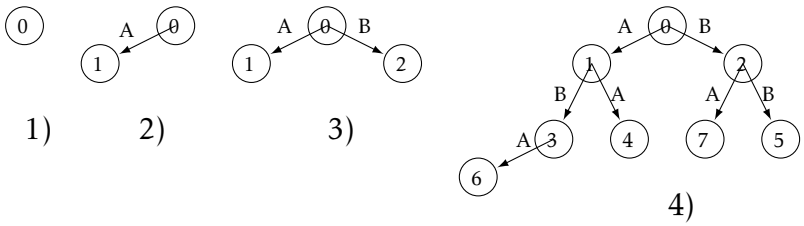
	Kodas
VASA RIS VASARA PAVASARIS	0;0;V;
V ASAR IS VASARA PAVASARIS	0;0;A;
VA SARI S VASARA PAVASARIS	0;0;S;
VAS ARIS VASARA PAVASARIS	2;1;R;
VASAR IS V ASARA PAVASARIS	0;0;I;
VASARI S VA SARA PAVASARIS	4;1; :
VASARIS VASA RA PAVASARIS	8;4;R;
VASAR IS VASAR A PA VASARIS	2;1; ;
VASARIS VASARA PAVA SARIS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

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

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

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.

Štai kaip tai galima padaryti. Tarkime, intervalas, kuriame norime parinkti skaičių – žodžio kodą yra $I(\mathbf{x}) = [a; a + \delta]$. Surasime skaičius c ir n, kad būtų

$$\frac{(c-1)}{2^n} \leqslant a < \frac{c}{2^n} < a + \delta.$$

Tada žodžio $\mathbf{x}$ kodu galėsime imti skaičių $\frac{c}{2^n}$, kurį galėsime perduoti siųsdami skaičiaus c dvejetainės išraiškos n bitų, papildydami nuliais, jei reikia iš kairės. Skaičius n – mažiausias nelygybės $\frac{1}{2^n} < \delta$ sprendinys. Taigi galime imti

$$n = \lceil \log_2(1/\delta) \rceil.$$

Skaičių c rasime pasinaudoję nelygybe:

$$(c-1) \leqslant a2^n < c, \quad c = \lceil a2^n \rceil.$$

Pavyzdyje žodžiui $\mathbf{x} = 213552$ koduoti sudarėme intervalą

$$I(213552) = [0,22946; 0,22962875).$$

Taigi

$$a = 0.22946, \quad \delta = 0.00016875, \quad n = 13.$$

Tada $c = 1880, c/2^n = 0,2294921875$. Skaičiaus c dvejetainę išraišką 0011101011000 galėtume įrašyti kaip žodžio **x** kodą.

Yra dvi aplinkybės, kurios verčia susimąstyti, ar aritmetinis 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 **x** = 21355241. Mastelio keitimą atliksime tik tada, kai ir apatinis ir viršutinis intervalo rėžiai prasidės tuo pačiu skaitmeniu (arba keliais vienodais).

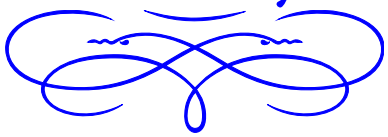
Simbolis	I	I^*	Kodas
2	[0,2;0,45)		
1	[0,2;0,25)	[0;0,5)	2
3	[0,425; 0,5)		
5	[0,4775; 0,5)		
5	[0,49325; 0,5)		
2	[0,4946; 0,4962875)	[0,46; 0,62875)	49
4	[0,56125; 0,578125)	[0,6125; 0,78125)	5
1	[0,6125; 0,64625]	[0,125; 0,4625)	62

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.

Klaidas taisantys kodai



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.

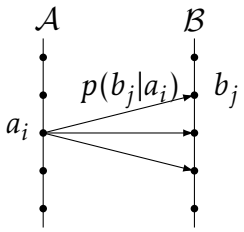
Apibrėžimas. *Perdavimo kanalą vadinsime kanalu be atminties, jeigu visiems žodžiams* $\mathbf{u} = u_1 u_2 \ldots u_n \in \mathcal{A}^n, \mathbf{v} = v_1 v_2 \ldots v_n \in \mathcal{B}^n$ *teisinga lygybė*

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

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

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

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.

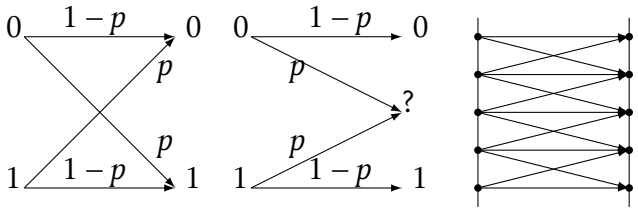
Tarptautinis arba vietinis paštas, telefonas, elektroninis paštas, kompaktinė plokštelė, popieriaus lapas, kurį galime prirašyti, ... , tuščias butelis, į kurį galima įdėti laišką ir paleisti upe pasroviui... Perdavimo kanalų yra įvairių, visų jų neaprėpsime. Panagrinėkime, keletą paprastų kanalų matematinių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

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

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, $0 \leqslant p \leqslant 1$, 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.

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.

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

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

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:

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

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

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

Teorema. *Dvinario simetrinio kanalo talpa yra*

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

Įrodymas. Dydžiai U, V įgyja tas pačias reikšmes $0, 1$. Pažymėkime $P(U = 0) = t$, tada $P(U = 1) = 1 - t$. Skaičiuosime perduodamos kanalu informacijos kiekį

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

Skaičiuokime tikimybes:

$$P(U=0,V=0)=t(1-p), \quad P(U=0,V=1)=tp,$$
$$P(U=1,V=0)=(1-t)p, \quad P(U=1,V=1)=(1-t)(1-p).$$

Tada

$$\begin{aligned} H(U,V) &= t(1-p)\log_2\frac{1}{t(1-p)}+tp\log_2\frac{1}{tp} \\ &+ (1-t)p\log_2\frac{1}{(1-t)p}+(1-t)(1-p)\log_2\frac{1}{(1-t)(1-p)}. \end{aligned}$$

Naudodamiesi logaritmo adityvumu, pav., $\log_2\frac{1}{t(1-p)}=\log_2\frac{1}{t}+\log_2\frac{1}{(1-p)}$, ir sugrupavę narius gausime

$$\begin{aligned} H(U,V) &= t\log_2\frac{1}{t}+(1-t)\log_2\frac{1}{1-t}+p\log_2\frac{1}{p}+(1-p)\log_2\frac{1}{1-p} \\ &= H(U)+p\log_2\frac{1}{p}+(1-p)\log_2\frac{1}{1-p}. \end{aligned}$$

Taigi

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

Perduotos informacijos kiekis yra didžiausias, kai $H(V)$ įgyja didžiausią reikšmę. Ši didžiausia reikšmė lygi 1. Ji įgyjama, kai $t=1/2$, nes tada

$$\begin{aligned} P(V=0) &= P(U=0)P(V=0|U=0)+P(U=1)P(V=0|U=1) \\ &= t(1-p)+(1-t)p=1/2, \\ P(V=1) &= 1-P(V=0)=1/2. \end{aligned}$$

Teiginys įrodytas.

Kodai ir dekodavimo taisyklės

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

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

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!

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 ($n \geqslant m$) 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.

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.

$$q^m \leqslant N.$$

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.

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

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

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

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

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 **D** žodį **d**, turi stengtis atpažinti, koks kodo **C** žodis buvo jam siųstas. Taigi jis turi vadovautis tam tikra dekodavimo taisykle.

Apibrėžimas. *Tegu* **C** *yra kodas, o* **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 **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 **d**:

$$P(klaida|\mathbf{d}) = \sum_{\mathbf{c}\neq f(\mathbf{d})} p(\mathbf{c}|\mathbf{d}) = 1 - p(f(\mathbf{d})|\mathbf{d}). \tag{14}$$

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 (14), gausime

$$p_{vid} = 1 - \sum_{\mathbf{d}} p(f(\mathbf{d})|\mathbf{d})p(\mathbf{d}). \tag{15}$$

Ši vidutinės klaidos išraiška rodo, kaip sudaryti dekodavimo taisyklę, kuri minimizuotų vidutinę klaidą. Iš tikrųjų tam reikia pasiekti, kad sumos reikšmė (15) lygybėje būtų didžiausia. Taip bus, jei žodžiui **d** parinksime $f(\mathbf{d})$ taip, kad tikimybė $p(f(\mathbf{d})|\mathbf{d})$ būtų maksimali.

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*

$$p(f(\mathbf{d})|\mathbf{d}) = \max_{\mathbf{c}} p(\mathbf{c}|\mathbf{d}). \tag{16}$$

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.

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

Sukeitę (16) lygybėje argumentus vietomis, gausime kitos taisyklės apibrėžimą.

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*

$$p(\mathbf{d}|f(\mathbf{d})) = \max_{\mathbf{c}} p(\mathbf{d}|\mathbf{c}). \tag{17}$$

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.

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.

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.

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.

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

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

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

Shannono teoremos dvinariam kanalui

Shannono teoremos apie ryšio galimybes naudojantis nepatikimais kanalais – svarbiausi informacijos ir kodavimo teorijos teiginiai.

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 $N \leqslant q^n$. 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[1], kurį minėjome jau anksčiau, laikomas pirmuoju kodavimo teorijos, kaip atskiros tyrimų srities, darbu.

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.

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*

$$R(\mathbf{C}_m) \geq R, \quad p_{max}(\mathbf{C}_m) \to 0, \quad m \to \infty.$$

[1] „Mathematical theory of communication'", 'The Bell System Technical Journal, 1948.

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“.

Taigi sužinojus, ką įmanoma pasiekti, reikia ieškoti praktiškai svarbių sprendimų.

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

$$N \leqslant \frac{2^n}{C_n^k}.$$

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

$$\ln C_n^k \approx n\left((1-p) \ln \frac{1}{1-p} + p \ln \frac{1}{p}\right), \quad \frac{\log_2 C_n^k}{n} \approx (1-p) \ln \frac{1}{1-p} + p \ln \frac{1}{p}$$

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:

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

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.

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.

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.

Informacijos kiekis, kanalai, dekodavimas

Jeigu kanalas iškraipo perduodamus žodžius, tai juo galima perduoti tik dalį šaltinio informacijos kiekio. Jeigu gavėjas taiko kokią nors protingą dekodavimo taisyklę, pavyzdžiui, minimalaus atstumo, tai intuicija sako, kad tai padidina peduodamos informacijos kiekį. Tačiau skaičiavimų rezultatai skatina peržiūrėti šią intuicijos siūlomą išvadą.

Panagrinėkime tokį gana dirbtinį informacijos perdavimo kanalu pavyzdį. Tarkime, šaltinio perduodami žodžiai – atsitiktinio dydžio U reikšmės, o gavėjo gaunami žodžiai – atsitiktinio dydžio V reikšmės. Pažymėkime

$$P(U = 000) = p_1, \quad P(U = 111) = p_2, \quad p_1 + p_2 = 1.$$

Perdavimo kanalas gana keistas, jis iškraipo perduodamus žodžius taip:

$$000 \mapsto 100, \quad 111 \mapsto 001.$$

Beveik akivaizdu, kad perduodamos informacijos kiekio ir teisingo perdavimo tikimybės reikšmės yra tokios:

$$I(U, V) = H(U), \quad P(\text{perduota teisingai}) = 0.$$

O dabar tarkime, kad gavėjas taiko minimalaus atstumo dekodavimo taisyklę, ir galutinis rezultatas – dydžio Z reikšmė. Akivaizdu, kad $P(Z = 000) = 1$,

$$I(U, Z) = 0, \quad P(\text{dekoduota teisingai}) = p_1.$$

Taigi be dekodavimo perduodama visa šaltinio informacija, bet teisingo perdavimo tikimybė lygi 0. Su dekodavimu – perduodamos informacijos kiekis lygus nuliui, bet teisingo perdavimo (po dekodavimo) tikimybė p_1 teigiama. Jeigu yra du kanalai, vienas be dekodavimo, o kitas su dekodavimu, tai jų šūkiai gali būti tokie:

perduosime didelį informacijos kiekį, bet netiksliai;

perduosime mažiau, bet tiksliau.

Taigi dekodavimas didina ne perduodamos informacijos kiekį, bet tik teisingo perdavimo tikimybę.

Hammingo kodas

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.[2] 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.

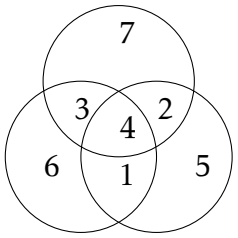
Savo straipsnyje „Error Detecting and Error Correcting Codes“, išspausdintame „The Bell System Technical Journal“ 1950 metų antrajame numeryje jis rašo:

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.

[2] 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

$$P_2 = (q^3 + 3pq^2)^4.$$

Galima įsitikinti, kad 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.

Hammingo kodas suteikia galimybę ištaisyti vieną klaidą septynių bitų dvejetainiame žodyje. Sudarysime vieną klaidą taisančius kodus bet kokio nelyginio ilgio $n > 1$ dvejetainiuose žodžiuose. Tuos kodus taip pat vadinsime Hammingo kodais.

Tegu n yra natūralusis skaičius, $2^m \leqslant n < 2^{m+1}$. Sudarysime kodą iš žodžių

$$x_1 x_2 \dots x_n \in \{0,1\}^n.$$

Kodo sudarymo idėją paaiškinsime pavyzdžiu su $n = 10$. Kadangi $2^3 < n < 2^4$, tai visų skaičių $1 \leqslant m \leqslant n$ dvejetainėse išraiškose dalyvaus tik šie laipsniai: $2^0, 2^1, 2^2, 2^3$. Sudaromo kodo žodyje $x_1x_2...x_{10}$ simboliai $x_3, x_5, x_6, x_7, x_9, x_{10}$ bus informaciniai, t.y. galės būti laisvai parenkami, o simboliai x_1, x_2, x_4, x_8 – kontroliniai, suskaičiuoti panaudojant informacinių simbolių reikšmes. Kontrolinių simbolių reikšmes sudarysime pagal skaičių $3, 5, 6, 7, 9, 10$ dvejetaines išraiškas:

$$\begin{aligned}
3 &= 1\cdot 2^0 + 1\cdot 2^1 + 0\cdot 2^2 + 0\cdot 2^3,\\
5 &= 1\cdot 2^0 + 0\cdot 2^1 + 1\cdot 2^2 + 0\cdot 2^3,\\
6 &= 0\cdot 2^0 + 1\cdot 2^1 + 1\cdot 2^2 + 0\cdot 2^3,\\
7 &= 1\cdot 2^0 + 1\cdot 2^1 + 1\cdot 2^2 + 0\cdot 2^3,\\
9 &= 1\cdot 2^0 + 0\cdot 2^1 + 0\cdot 2^2 + 1\cdot 2^3,\\
10 &= 0\cdot 2^0 + 1\cdot 2^1 + 0\cdot 2^2 + 1\cdot 2^3.
\end{aligned}$$

Sudarysime lentelę, eilutėse išrašydami koeficientų prie 2^i reikšmes skaičių $3, 5, 6, 7, 9, 10$ išraiškose:

	x_3	x_5	x_6	x_7	x_9	x_{10}
2^0	1	1	0	1	1	0
2^1	1	0	1	1	0	1
2^2	0	1	1	1	0	0
2^3	0	0	0	0	1	1

Pastebėkime, kad lentelėje nėra vienodų stulpelių. Pagal lentelės eilutes sudarykime lygybes, kontrolinių simbolių $x_{2^i}, i = 0, 1, 2, 3$, reikšmėms apskaičiuoti:

$$\begin{aligned}
x_1 &= 1\cdot x_3 + 1\cdot x_5 + 0\cdot x_6 + 1\cdot x_7 + 1\cdot x_9 + 0\cdot x_{10},\\
x_2 &= 1\cdot x_3 + 0\cdot x_5 + 1\cdot x_6 + 1\cdot x_7 + 0\cdot x_9 + 1\cdot x_{10},\\
x_4 &= 0\cdot x_3 + 1\cdot x_5 + 1\cdot x_6 + 1\cdot x_7 + 0\cdot x_9 + 0\cdot x_{10},\\
x_8 &= 0\cdot x_3 + 0\cdot x_5 + 0\cdot x_6 + 0\cdot x_7 + 1\cdot x_9 + 1\cdot x_{10}.
\end{aligned}$$

Pavyzdžiui, norėdami perduoti kanalą bitus 011011 turėtume sudaryti žodį:

$$011011 \mapsto **0*110*11 \mapsto 0000110011$$

ir perduoti į kanalą.

Tarkime, kanalas vieną bitą pakeitė. Gavėjas, gavęs žodį, tikrins keturias kontrolines lygybes. Jeigu iškraipytas žodžio bitas x_3, tai

pirmosios dvi lygybės bus nebe teisingos, o dvi paskutinės išliks teisingos. Tada gavėjas nuspręs, kad reikia taisyti x_3. Jeigu būtų pakeistas kitas reikšminis bitas, būtų kitos lygybės neteisingos (visi lentelės stulpeliai skirtingi!) Jeigu būtų iškreiptas kontrolinis simbolis, tik viena kontrolinė lygybė būtų neteisinga.

Šio kodo koeficientas lygus $5/10 = 1/2$. Atidesnis skaitytojas pastebės, kad nedidinant kontrolinių lygybių skaičiaus, kodo žodį galima pailginti iki 15, o informacinių bitų skaičių iki 10.

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...

Stačiakampiai ir trikampiai kodai

Dar du vieną klaidą žodyje ištaisantys kodai, kurių kontrolines lygybes lengva sudaryti.

Tegu m, n yra du natūralieji skaičiai. Sudarysime dvejetainės abėcėlės žodžių kodą $S(m,n)$, kurio kiekvienas žodis „neša" $m \cdot n$ informacinių bitų ir gali ištaisyti 1 klaidą.

Pirmiausia informacinius bitus $x_1, x_2, \ldots, x_{mn}$ surašykime į $m \times n$ lentelę

$$x_1 x_2 \ldots x_{nm} \rightarrow \begin{array}{cccc} x_1 & x_2 & \ldots & x_n \\ x_{n+1} & x_{n+2} & \ldots & x_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ x_{m(n-1)+1} & x_{m(n-1)+2} & \ldots & x_{mn} \end{array}$$

Papildykime lentelę pridėdami prie eilučių bitus $y_1, y_2, \ldots, y_m$, o prie stulpelių – $z_1, z_2, \ldots, z_n$. Tai ir bus kontroliniai bitai. Jų reikšmės – atitinkamų eilučių (stulpelių) informacinių bitų sumos moduliu 2.

Pavyzdžiui, imkime $m = 3, n = 4$. Tada papildytoji lentelė atrodys taip:

$$\begin{array}{ccccc} x_1 & x_2 & x_3 & x_4 & y_1 \\ x_5 & x_6 & x_7 & x_8 & y_2 \\ x_9 & x_{10} & x_{11} & x_{12} & y_3 \\ z_1 & z_2 & z_3 & z_4 & \end{array}$$

Kontrolines lygybes galima užrašyti taip:

$$\begin{array}{rr} x_1 + x_2 + x_3 + x_4 + y_1 = 0, & x_1 + x_5 + x_9 + z_1 = 0, \\ x_5 + x_6 + x_7 + x_8 + y_2 = 0, & x_2 + x_6 + x_{10} + z_2 = 0, \\ x_9 + x_{10} + x_{11} + x_{12} + y_3 = 0, & x_3 + x_7 + x_{11} + z_3 = 0, \\ & x_4 + x_8 + x_{12} + z_4 = 0. \end{array}$$

Taigi norėdami perduoti 12 informacinių simbolių, pridedame dar 7 :

$$x_1 x_2 \ldots x_{12} \mapsto x_1 x_2 \ldots x_{12} y_1 y_2 y_3 z_1 z_2 z_3 z_4.$$

Jeigu kanale būtų iškreiptas, pavyzdžiui, x_7, tai būtų nepatenkintos kontrolinės lygybės, kuriose dalyvauja bitai y_2, z_3. Gavėjas matytų, kad

antros eilutės ir trečio stulpelio sankirtoje yra bitas, kurio reikšmę reikia pakeisti. Jeigu būtų iškreiptas vienas iš kontrolinių bitų, tik viena kontrolinė lygybė būtų nepatenkinta.

Panagrinėkime, kaip sudaromos trikampio kodo $T(r), r \geqslant 1$, kontrolinės lygybės. Šio kodo žodis „neša"

$$r + (r-1) + \cdots + 2 + 1 = \frac{r(r+1)}{2}$$

informacinių bitų, kurie kontrolinių lygybių sudarymui surašomi į trikampį pavidalą.

Pavyzdžiui, kodui $T(4)$ sudaryti bitus reikia surašyti taip:

$$x_1x_2\ldots x_{10} \to \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \\ x_5 & x_6 & x_7 & \\ x_8 & x_9 & & \\ x_{10} & & & \end{array}$$

Po to kiekviena eilutė papildoma kontroliniu bitu ir dar vienas bitas pridedamas pirmojo stulpelio pabaigoje:

$$\begin{array}{ccccc} x_1 & x_2 & x_3 & x_4 & y_1 \\ x_5 & x_6 & x_7 & y_2 & \\ x_8 & x_9 & y_3 & & \\ x_{10} & y_4 & & & \\ y_5 & & & & \end{array}$$

Bito y_j reikšmė gaunama sudėjus moduliu 2 informacinius bitus, kurie yra toje pačioje eilutėje ir stulpelyje kaip ir y_j. Pavyzdžiui,

$$y_3 = x_3 + x_7 + x_8 + x_9.$$

Kontrolines sumas galime užrašyti taip:

$$\begin{aligned} x_1 + x_2 + x_3 + x_4 + y_1 &= 0, \\ x_4 + x_5 + x_6 + x_7 + y_2 &= 0, \\ x_3 + x_7 + x_8 + x_9 + y_3 &= 0, \\ x_2 + x_6 + x_9 + x_{10} + y_4 &= 0, \\ x_1 + x_5 + x_8 + x_{10} + y_5 &= 0. \end{aligned}$$

Ir vėl – jeigu kanale bus iškraipytas informacinis bitas, dvi kontrolinės lygybės bus neteisingos ir galėsime nustatyti, kurį bitą reikia taisyti. Jeigu bus neteisingai perduotas kontrolinis bitas, tik viena kontrolinė lygybė nebus teisinga.

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.

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.

Teorema. *Hamingo atstumas aibėje* $\mathcal{A}_q^n$ *turi šias savybes:*

- $h(\mathbf{x}, \mathbf{x}) = 0, \quad \mathbf{x} \in \mathcal{A}_q^n;$
- $h(\mathbf{x}, \mathbf{y}) = h(\mathbf{y}, \mathbf{x}), \quad \mathbf{x}, \mathbf{y} \in \mathcal{A}_q^n;$
- $h(\mathbf{x}, \mathbf{y}) \leq h(\mathbf{x}, \mathbf{z}) + h(\mathbf{y}, \mathbf{z}), \quad \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathcal{A}_q^n.$

Pirmosios dvi savybės akivaizdžios. Kad būtų aišku, jog ir trečioji teisinga panagrinėkime, kaip skaičiuotume Hammingo atstumus tarp žodžių $\mathbf{x}, \mathbf{y}, \mathbf{z}$, kuriuose a, b, c žymi skirtingus simbolius:

$$\begin{aligned} \mathbf{x} &= a\ a\ a\ a\ \ldots \\ \mathbf{y} &= a\ a\ b\ b\ \ldots \\ \mathbf{z} &= a\ b\ a\ c\ \ldots \end{aligned}$$

Dar kartą prisiminkime kodo apibrėžimą.

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 $f : \mathcal{A}_q^n \to \mathbf{C}$.

Apibrėžimas. *Dekodavimo taisyklę $f : \mathcal{A}_q^n \to \mathbf{C}$ vadinsime minimalaus atstumo taisykle, jei su kiekvienu $\mathbf{d} \in \mathcal{A}_q^n$*

$$h(\mathbf{d}, f(\mathbf{d})) = \min_{\mathbf{c} \in \mathbf{C}} h(\mathbf{d}, \mathbf{c}). \tag{18}$$

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 (18) 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.

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.

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.*

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.

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.

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 $(7, 16, 3)$.

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.

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.

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

$$V_q(n, r) = |B_q(\mathbf{x}, r)|.$$

Apibrėžimas. *Tegu* $\mathbf{C}$ *yra koks nors* (n,N) *kodas. Didžiausią sveikąjį skaičių* t*, kuriam*

$$B_q(\mathbf{c}_1,t)\cap B_q(\mathbf{c}_2,t)=\emptyset,\ \text{ jei }\ \mathbf{c}_1,\mathbf{c}_2\in\mathbf{C},\ \ \mathbf{c}_1\neq\mathbf{c}_2,$$

vadinsime kodo $\mathbf{C}$ *pakavimo spinduliu. Jį žymėsime* $r_p=r_p(\mathbf{C})$.

Taigi rutuliai $B_q(\mathbf{c},t),\mathbf{c}\in\mathbf{C},t\le r_p(\mathbf{C})$, sudaro nesikertančių aibių šeimą; jei $t\ge r_p(\mathbf{C})+1$, bent du rutuliai kertasi.

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 $r_p(\mathbf{C})\le r_d(\mathbf{C})$. 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 $r\ge 1$ 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 $r_p(\mathbf{C})\le r$. 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

$$h(\mathbf{x},\mathbf{c}_0)\le h(\mathbf{x},\mathbf{c})+h(\mathbf{c},\mathbf{c}_0),$$

arba

$$h(\mathbf{x},\mathbf{c})\ge h(\mathbf{x},\mathbf{c}_0)-h(\mathbf{c},\mathbf{c}_0)\ge n-r,$$

t. y. $r_d(\mathbf{C})\ge n-r$.

Paprastas pavyzdys: $(n,2)$ kodui

$$\mathbf{C} = \{000\ldots00, 000\ldots11\}$$

gauname: $r_p = 0, r_d = n-1$.

Apibrėžimas. *Kodą* $\mathbf{C}$ *vadinsime tobulu, jei*

$$r_p(\mathbf{C}) = r_d(\mathbf{C}).$$

Teorema. *(n,N,d) kodas $\mathbf{C}$ yra tobulas tada ir tik tada, kai $d = 2t+1$ ir galioja lygybė*

$$N V_q(n,t) = q^n.$$

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

$$\sum_{\mathbf{c}\in\mathbf{C}} |B(\mathbf{c},t)| = |\mathcal{A}_q^n| \quad \text{arba} \quad N V_q(n,t) = q^n.$$

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

$$N V_q(n,t) = q^n,$$

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.
- 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 ($q = 2, m = 3$).

Kodai su kontroliniu simboliu

Šiuos kodus matome ant knygų, kortelių, ant kiekvienos prekės...

Jeigu minimalus kodo atstumas yra lygus 1, tai klaidų taisymo požiūriu kodas koone bevertis – negali garantuotai ištaisyti nei vienos klaidos. Tačiau jeigu minimalus atstumas lygus 2, tai įvykus vienai klaidai, gavėjas visada įspėjamas. Jeigu galima pakartoti siuntimą, tai nieko daugiau ir nereikia. Tokie kodai paprastai sudaromi prie informacinių simbolių pridedant pagal atitinkamą taisyklę sudarytą kontrolinį simbolį.

Brūkšniniai kodai – tai ir yra kodai su kontroliniu simboliu, o skeneris – kanalas, kuris siunčia informaciją į kompiuterio atmintį. Informaciniai simboliai – skaičiai užrašomi juodų ir baltų juostų seka. Jeigu nuskaitytas žodis nėra kodo žodis, skaitymas kartojamas.

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.

Kontrolinei lygybei užrašyti naudosime lyginio duotu moduliu žymenį. Žymuo $a \equiv b \pmod{n}$ reiškia, kad skaičius a, b dalijant iš n, gaunama ta pati liekana $r, 0 \leqslant r < n$. Pavyzdžiui, $12 \equiv 112 \equiv 22 \equiv 2 \equiv -8 \pmod{10}$, nes

$$12 = 1 \cdot 10 + 2, \quad 112 = 11 \cdot 10 + 2, \quad -8 = (-1) \cdot 10 + 2.$$

Taigi informacijos apie knygą žodis $x_1 x_2 \ldots x_9$, sudarytas iš 9 simbolių, papildomas dešimtuoju taip, kad galiotų lygybė

$$X \cdot x_1 + 9 \cdot x_2 + 8 \cdot x_3 + 7 \cdot x_4 + 6 \cdot x_5 + \ldots + 2 \cdot x_9 + 1 \cdot x_{10} \equiv 0 \pmod{11}.$$

Pavyzdžiui, ISBN - 9986-16-180-0 yra knygos, išleistos Lietuvoje (9986 – Lietuvos kodas), „Tyto alba“ leidykloje (leidyklos kodas – 16).

Turint informcinius simbolius $x_1x_2\ldots x_9$ kontrolinis simbolis sudaromas taip: randamas skaičius $1 \leqslant r \leqslant 11$,

$$r \equiv X \cdot x_1 + 9 \cdot x_2 + 8 \cdot x_3 + 7 \cdot x_4 + 6 \cdot x_5 + \ldots + 2 \cdot x_9 \pmod{11}$$

ir imama $x_{10} = 11 - r$. Skaičius r nuo dalybos iš 11 liekanos skiriasi tik tuo atveju, kai dalybos liekana lygi nuliui, tada imame $r = 11, x_{10} = 0$. ISBN kodas visada „pastebi“, jei įvyko viena klaida. Pavyzdžiui, jeigu nuskaitant žodį $x_1x_2\ldots x_9$ simbolis x_2 pasikeitė į x_2^*, tai tikrinant kontrolinę lygybę gausime

$$\begin{aligned} &X \cdot x_1 + 9 \cdot x_2^* + 8 \cdot x_3 + 7 \cdot x_4 + 6 \cdot x_5 + \ldots + 2 \cdot x_9 + 1 \cdot x_{10} = \\ &X \cdot x_1 + 9 \cdot x_2 + 8 \cdot x_3 + 7 \cdot x_4 + 6 \cdot x_5 + \ldots + 2 \cdot x_9 + 1 \cdot x_{10} + 9(x_2 - x_2^*) \\ &\equiv 0 + 9(x_2 - x_2^*) \pmod{11}. \end{aligned}$$

Tačiau $9(x_2 - x_2^*)$ negali dalytis iš 11, taigi nulio negausime ir klaida bus pastebėta. Tačiau kodas signalizuoja ir apie kitas įvykusias klaidas. Gali pranešti, kad įvyko klaida, kai neteisingai perskaityti du ar daugiau simbolių, tačiau ne visada. Kokios klaidos dažniausiai skaitant informaciją įvyksta?

Nuo 2007 metų išleidžiamos knygos 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

$$1 \cdot x_1 + 3 \cdot x_2 + \ldots + 3 \cdot x_{12} + 1 \cdot x_{13} \equiv 0 \pmod{10}.$$

EAN kodas taip pat atpažįsta, jei vienas simbolis perduodamas neteisingai, o taip pat daug ir kitokios rūšies klaidų.

Olandų matematikas J. Verhoeff 1969 metais[3] paskelbė tyrimų apie dažniausiai įvykstančias klaidas, kai skaitmeninę informaciją perduoda žmonės.

Klaidos rūšis	Pavyzdys	Dažnumas
Pavienės klaidos	$a \mapsto b$	$60 - 95\%$
Gretimos perstatos	$ab \mapsto bb$	$10 - 20\%$
Dvynių klaidos	$aa \mapsto bb$	$0,5 - 1,5\%$
Trijų perstatos	$acb \mapsto bca$	$0,5 - 1,5\%$
Dvynių perstatos	$aca \mapsto bcb$	$< 1\%$
Tarimo klaidos	$50 \mapsto 15$	$0,5 - 1,5\%$
Pridėta ar praleista		$10 - 20\%$

[3]Verhoeff, Jacobus (1969) "Error Detecting Decimal Codes," Mathematical Center Tract 29, Amsterdam.

Banko kreditinėms kortelėms žymėti taip pat naudojamas kodas su kontroliniu simboliu. Penkiolikos skaitmenų žodis $d_1 d_2 \dots d_{15}$ papildomas šešioliktuoju skaitmeniu d_{16}, kad būtų teisinga kontrolinė lygybė

$$s(2 \cdot d_1) + d_2 + s(2 \cdot d_3) + \cdots + d_{14} + s(2 \cdot d_{15}) + d_{16} \equiv 0 \pmod{10};$$

čia $s(n)$ reiškia skaičiaus išraiškos dešimtainių skaitmenų sumą, pavyzdžiui, $s(13) = 4$. Jeigu kortelės informacija būtų užrašyta, pavyzdžiui, skaitmenimis

$$2-3-5-6-3-9-3-4-1-5-7-6-6-9-1,$$

tai kontrolinį skaitmenį skaičiuotume taip:

$$t = s(4)+3+s(10)+6+s(6)+9+s(6)+4+s(2)+5+s(14)+6+s(12)+9+s(2) = 71,$$

$t \equiv 1 \pmod{10}, d_{16} = 10 - 1 = 9$. Šis kodas pastebi visus vieno skaitmens neteisingo perdavimo atvejus ir visus gretimų skaitmenų sukeitimo atvejus, išskyrus vieną – kai sukeičiami greta stovintys skaitmenys 0 ir 9.

Panagrinėsime IBM kodą su kontroliniu simboliu. Jį galima naudoti bet kokio ilgio žodžiui, sudarytam iš dešimtainių skaitmenų.

Kodui sudaryti naudojamas keitinys

$$\sigma = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 0 & 2 & 4 & 6 & 8 & 1 & 3 & 5 & 7 & 9 \end{pmatrix}$$

Taigi $\sigma(0) = 0, \sigma(1) = 2$ ir t.t. Keitinį galime suvokti sudarytą iš ciklų:

$$0 \mapsto 0,\ 1 \mapsto 2 \mapsto 4 \mapsto 8 \mapsto 7 \mapsto 5 \mapsto 1, \quad 3 \mapsto 6 \mapsto 3,\ 9 \mapsto 9$$

ir užrašyti taip:

$$\sigma = (0)(124875)(36)(9).$$

Tarkime, informacija užrašoma dešimtainiais skaitmenimis $d_1 d_2 \dots d_n$ ir n lyginis. Tada kontrolinis skaitmuo d_{n+1} pridedamas taip, kad būtų teisinga lygybė

$$d_1 + \sigma(d_2) + d_3 + \sigma(d_4) + \cdots + d_{n-1} + \sigma(d_n) + d_{n+1} \equiv 0 \pmod{10}.$$

Pavyzdžiui, jei $d_1 d_2 d_3 d_4 = 2314$, tai

$$d_1 + \sigma(d_2) + d_3 + \sigma(d_4) = 2 + 6 + 1 + 8 = 17 \equiv 7 \pmod{10}$$

ir kontrolinis simbolis yra $d_5 = 10 - 7 = 3$.

Jeigu n yra nelyginis, tai kontrolinis simbolis d_{n+1} yra skaitmuo, su kuriuo teisinga lygybė

$$\sigma(d_1) + d_2 + \sigma(d_2) + d_3 + \cdots + d_{n-1} + \sigma(d_n) + d_{n+1} \equiv 0 \pmod{10}.$$

IBM kodas aptinka vieno skaitmens neteisingo nuskaitymo klaidas, taip pat gretimų skaitmenų sukeitimo vietomis klaidas.

Dalybos liekanų kūnas

Skaičiavimo dalybos iš n liekanų aibėje savybių apžvalga.

Teiginys apie sveikųjų skaičių dalybą su liekana – paprastas, bet labai svarbus.

Teorema. *Bet kokiems sveikiesiems skaičiams m, n $(n > 0)$ yra vienintelė sveikųjų skaičių pora q, r, kad*

$$m = qn + r, \quad 0 \le r < b.$$

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

$$m - n < qn \le m, \quad \text{arba} \quad 0 \leqslant m - qn < n.$$

Taigi

$$m = qn + r, \quad r = m - qn, \quad 0 \le r < n.$$

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.

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 $\mathbb{Z}_n$.

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.

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.

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.

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:

Apibrėžimas. *Žodžių* $\mathbf{x}, \mathbf{y} \in V_n, \mathbf{x} = x_1 x_2 \ldots x_n, \mathbf{y} = y_1 y_2 \ldots y_n$, *suma vadinsime žodį* $\mathbf{z} = z_1 z_2 \ldots z_n$, *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.

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.

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.

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.

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.

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

$$\mathbf{e}_1 = 100\ldots00,\ \mathbf{e}_2 = 010\ldots00,\ \ldots,\ \mathbf{e}_n = 000\ldots01 \tag{19}$$

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.

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.

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 (19) 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.

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.

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.

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.

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.

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*

$$\mathbf{L} = \{\mathbf{x} \in V_n : \mathbf{x}\cdot\mathbf{h}_1 = \mathbf{x}\cdot\mathbf{h}_2 = \ldots = \mathbf{x}\cdot\mathbf{h}_{n-k} = 0\}. \tag{20}$$

Sąlygas (20) 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 (20) sąlygos reiškia, kad poerdvis **L** yra sudarytas iš tų žodžių **x**, kurių komponentės $x_1 x_2 \dots x_n$ tenkina tokias lygybes:

$$\begin{aligned}
&h_{11}x_1 + h_{12}x_2 + \dots + h_{1n}x_n = 0,\\
&h_{21}x_1 + h_{22}x_2 + \dots + h_{2n}x_n = 0,\\
&\dots\dots\dots\dots\dots\dots\dots\dots\dots\\
&h_{n-k,1}x_1 + h_{n-k,2}x_2 + \dots + h_{n-k,n}x_n = 0.
\end{aligned}$$

Tiesiniai kodai

Naudingesni yra tie daiktai, su kuriais galime atlikti daugiau veiksmų. Juk mobilus telefonas tikrai naudingesnis už veidrodį. O išimtys patvirtina taisyklę. Daugiau laimėsime, jei kodui sudaryti pasirinksime abėcėlę, su kurios elementais galėsime atlikti daugiau veiksmų.

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 skyriaus apibrėžimas:

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.

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!

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 $\mathbf{a}_1,\dots,\mathbf{a}_k$, kurie sudaro $\mathbf{L}$ kaip tiesinio $\mathbb{F}_q^n$ poerdvio bazę.

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:

$$\mathbf{L}=\{\mathbf{x}G:\mathbf{x}\in\mathbb{F}_q^k\}; \tag{21}$$

čia $\mathbf{x}G$ reiškia žodžio, kaip vektoriaus-eilutės, ir matricos sandaugą.

Panagrinėkime (21) 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.

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.

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 $\mathbf{L}$, t.y., turinčio tuos pačius parametrus ir tas pačias klaidų taisymo galimybes, generuojanti matrica.*

Įsitikinti, kad šis teiginys teisingas, galima 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:

$$G' = \begin{pmatrix} 1 & 0 & \dots & 0 & a_{1,1} & \dots & a_{1,n-k} \\ 0 & 1 & \dots & 0 & a_{2,1} & \dots & a_{2,n-k} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & a_{k,1} & \dots & a_{k,n-k} \end{pmatrix} = (I_k, A); \tag{22}$$

čia: I_k yra vienetinė $k \times k$ matrica, A – kūno $\mathbb{F}_q$ elementų $k \times (n-k)$ matrica. Susitarsime sakyti, jog (22) matrica yra **standartinio pavidalo**.

Iš teoremos apie generuojančios matricos pertvarkius galime daryti tokią išvadą:

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.

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: $w(00\ldots0) = 0$; kitų žodžių svoriai ne mažesni už 1.

Teorema. *Tegu d yra tiesinio kodo $\mathbf{L}$ minimalus atstumas. Tada*

$$d = \min\{w(\mathbf{x}) : \mathbf{x} \in \mathbf{L}, \mathbf{x} \neq 00\ldots0\}.$$

Įrodymas. Tegu $\mathbf{0} = 00\ldots0$ – nulinis žodis; h kaip ir anksčiau žymime Hammingo atstumą. Tada bet kokiam $\mathbf{x} \in \mathbf{L}$, $\mathbf{x} \neq 00\ldots0$,

$$d \leq h(\mathbf{0}, \mathbf{x}) = w(\mathbf{x}), \quad \text{todėl} \quad d \leq \min\{w(\mathbf{x}) : \mathbf{x} \in \mathbf{L}, \mathbf{x} \neq 00\ldots0\}.$$

Tegu dabar $d = h(\mathbf{x}, \mathbf{y})$, $\mathbf{x}, \mathbf{y} \in \mathbf{L}$. Kadangi $\mathbf{x} - \mathbf{y} \in \mathbf{L}$, tai

$$\begin{aligned} d &= h(\mathbf{x}, \mathbf{y}) = h(\mathbf{0}, \mathbf{x} - \mathbf{y}) = w(\mathbf{x} - \mathbf{y}),\ \mathbf{x} - \mathbf{y} \neq \mathbf{0}, \\ d &\geq \min\{w(\mathbf{z}) : \mathbf{z} \in \mathbf{L}, \mathbf{z} \neq 00\ldots0\}. \end{aligned}$$

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.

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 $\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_k$ yra kodo $\mathbf{L}$ bazė, o $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{n-k}$ 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:

$$\mathbf{L} = \{\mathbf{x} : \mathbf{x} \in \mathbb{F}_p^n, \mathbf{x} \cdot \mathbf{h}_j = 0, j = 1, 2, \ldots, n-k\}. \tag{23}$$

Pastebėkime, kad visas (23) 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.

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

$$h_1 x_1 + h_2 x_2 + \ldots + h_n x_n = 0. \tag{24}$$

Kodavimas šiuo kodu – tai informacinių simbolių eilutės papildymas dar vienu simboliu (kontroliniu):

$$x_1 x_2 \ldots x_{n-1} \mapsto x_1 x_2 \ldots x_{n-1} x_n, \quad x_n = -(h_1 h_n^{-1} x_1 + \ldots + h_{n-1} h_n^{-1} x_{n-1}).$$

Nesunku nustatyti, kad minimalus kodo $\mathbf{K}$ atstumas yra $d = 2$, taigi jis negali ištaisyti nei vienos klaidos. Tačiau visada gali vieną klaidą aptikti! Tokius kodus jau nagrinėjome – tai kodai su kontroliniu simboliu.

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:

$$G \cdot H^T = O_{k,n-k}. \tag{25}$$

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 (25), bus šio kodo kontrolinė matrica.

Kontrolinę matricą paprasta surasti, jeigu kodas turi standartinio pavidalo generuojančią matricą.

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!

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.

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 $\mathbf{c} = c_1c_2\dots c_n$ yra kodo žodis, tai $\mathbf{c}H^T = \mathbf{0}$. Šią matricinę lygybę galime užrašyti kaip lygybių sistemą

$$\begin{aligned} h_{11}c_1 + h_{12}c_2 + \dots + h_{1n}c_n &= 0,\\ h_{21}c_1 + h_{22}c_2 + \dots + h_{2n}c_n &= 0,\\ &\dots\dots\dots\\ h_{n-k,1}c_1 + h_{n-k,2}c_2 + \dots + h_{n-k,n}c_n &= 0. \end{aligned}$$

Tarkime, kad perdavimo kanalas simbolių neiškraipo, tačiau gali ištrinti. Tada vietoj žodžio $\mathbf{c} = c_1c_2c_3\dots c_n$ gausime, pavyzdžiui, $\mathbf{d} = ?c_2?\dots c_n$. Dekodavimo, t. y. ištrintų simbolių ieškojimo, metodas kone akivaizdus – perduotas simbolių reikšmes reikia įstatyti į sistemos lygybes, o ištrintų simbolių vietoje – nežinomuosius; pavyzdyje – $x_1, x_3, \dots$ 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 $\mathbf{L}_\mathbf{x}, \mathbf{L}_\mathbf{y}$ iš tikrųjų yra „sluoksniai“, t. y. skirtingiems $\mathbf{x}, \mathbf{y} \in \mathbb{F}_q^n$ jos arba nesikerta, arba sutampa. Pažymėkime sluoksnių aibę

$$\mathbb{F}_q^n/\mathbf{L} = \{\mathbf{L}_\mathbf{x} : \mathbf{x} \in \mathbb{F}_q^n\}.$$

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:

$$\begin{pmatrix} \mathbf{a}_0 & \mathbf{c}_1 & \mathbf{c}_2 & \ldots & \mathbf{c}_N \\ \mathbf{a}_1 & \mathbf{a}_1+\mathbf{c}_1 & \mathbf{a}_1+\mathbf{c}_2 & \ldots & \mathbf{a}_1+\mathbf{c}_N \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{a}_s & \mathbf{a}_s+\mathbf{c}_1 & \mathbf{a}_s+\mathbf{c}_2 & \ldots & \mathbf{a}_s+\mathbf{c}_N \end{pmatrix}. \tag{26}$$

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 (26) matricos sudarymo būdas garantuoja, jog kiekvienoje eilutėje bus surašyti atitinkamos klasės $\mathbf{L_a}$ elementai, o pirmasis iš jų turės mažiausią svorį. Matricą (26) 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 $\mathbf{x}$.

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. Kitaip tariant – žodžio $\mathbf{x}$ sindromas yra kontrolinės matricos stulpelių tiesinė kombinacija su koeficientais, kurie yra $\mathbf{x}$ elementai.

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

$$\alpha_i = \left|\left\{\mathbf{a}_j : w(\mathbf{a}_j) = i\right\}\right|, \quad i = 0, \ldots, n.$$

Tada tikimybė, kad dekodavimo algoritmas duos teisingą atsakymą, lygi:

$$P(\mathbf{L} \text{ žodis bus dekoduotas teisingai}) = \sum_{i=0}^{n} \alpha_i p^i (1-(q-1)p)^{n-i}.$$

Kriptografijos pagrindai

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.

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.

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*

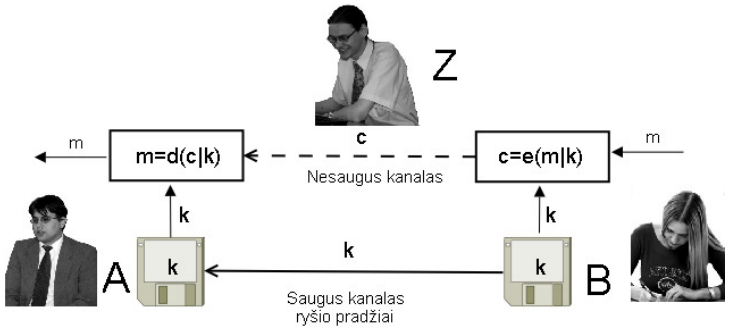
$$e(\cdot|K_e) : \mathcal{M} \to \mathcal{C}, \quad d(\cdot|K_d) : \mathcal{C} \to \mathcal{M}, \quad \langle K_e, K_d \rangle \in \mathcal{K},$$

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[4] 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:

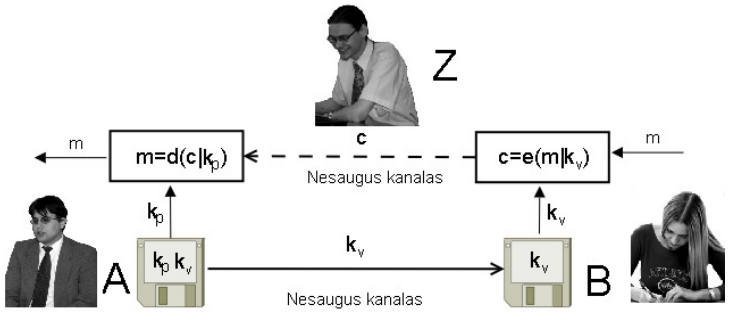
$$K_e = 2, \quad K_d = a_n a_{n+1} \dots a_{n+99}; \quad n = 10^{10^{10}}, \ \sqrt{K_e} = a_0, a_1 a_2 \dots,$$

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.

[4] 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:

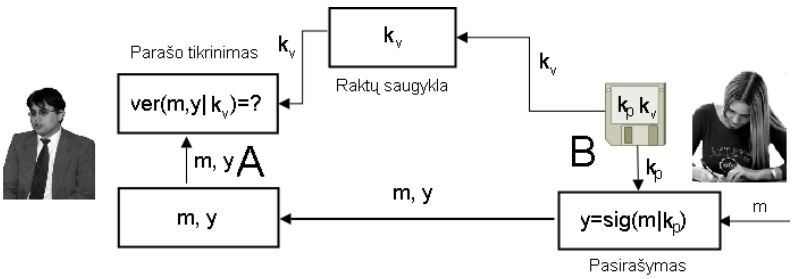
Apibrėžimas. *Skaitmeninių parašų schemą sudaro aibės* $\mathcal{M}, \mathcal{P}, \mathcal{K}$, *čia* $\mathcal{M}$ *– tekstų aibė,* $\mathcal{P}$ *– skaitmeninių parašų aibė,* $\mathcal{K}$ *– raktų* $K = \langle K_v, K_p \rangle$ *aibė (*K_p *– privačioji, parašui sudaryti skirta rakto komponentė,* K_v *– viešoji, parašui tikrinti skirta rakto komponentė) ir parašų sudarymo bei tikrinimo algoritmų šeimos*

$$sig(\cdot|K_p) : \mathcal{M} \to \mathcal{P}, \quad ver(\cdot|K_v) : \mathcal{M} \times \mathcal{P} \to \{0,1\}.$$

Parašo tikrinimo algoritmai turi savybę: $ver(x, sig(x|K_p)|K_v) = 1$. *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* $ver(x, y|K_v) = 0$.

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.

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.

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.

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.

Pavyzdžiui, sutarkime, kad šifravimui tekstas bus rašomas į lentelę, turinčią 7 stulpelius:

K	N	Y	G	O	S	P
E	R	D	U	O	D	A
P	A	T	I	R	T	Į

Jeigu lentelę skaitysime stulpeliais sutarta tvarka, gausime tekstą, gautą iš pradinio, perstačius jo raides. Stulpelių skaitymo tvarką galime nurodyti septynių raidžių žodžiu. Šis žodis ir bus šifro raktas, nurodantis, kokia tvarka skaityti stulpelius. Pavyzdžiui,

$$SNIEGAS \mapsto 6-5-4-2-3-1-7.$$

Kiekvienai rakto raidei priskyrėme skaičių: raidei A – vienetą, nes abėcėlėje ji pirmoji, raidė E gauna skaičių 2, nes po A abėcėlėje ieškodami kitų rakto raidžių pirmiausia sutinkame E ir t.t. Atkreipkime dėmesį, kaip skaičiai priskirti pasikartojančioms rakte raidėms S.

Taigi naudodamiesi šiuo raktu tekstą užšifruosime taip:

$$KNYGOSPERDDUODAPATIRTĮ \mapsto SDTGUIOORYDTNRAKEPPAĮ.$$

Toks perstatų šifras nėra labai saugus. Tačiau galime taikyti dvigubą šifravimą: vieną kartą su vienu, kitą kartą su kitu raktu. Tokius šifrus nelengva iššifruoti.

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	

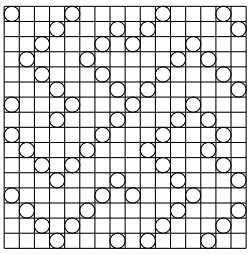
S			L
A			I
	U	M	
	E	K	

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... Kokį tekstą užšifravome?

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![5]

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.

[5] 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.

Keitiniai ir šifrai

Šifrų, paremtų keitiniais, sudarymo idėjas irgi galime įžvelgti Antikos laikų šaltiniuose.

Keitinys tai injektyvus vienos abėcėlės simbolių atvaizdis kitoje abėcėlėje: $\lambda : \mathcal{A} \to \mathcal{B}$. Naudodamiesi juo kaip raktu, galime šifruoti abėcėlės A simbolių eilutes taip:

$$M|\lambda) = \lambda(m_1)\lambda(m_2)\ldots\lambda(m_n).$$

Tačiau reikia sugalvoti, kaip tinkamai užrašyti ir perduoti tokius keitinius. Prisiminkime Polibijaus ugnies kodą:

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

Lentelę galime suvokti kaip keitinį, kuris kiekvieną graikišką raidę pakeičia skaitmenų – koordinačių pora. Šią idėją galime panaudoti kriptosistemoms sudaryti. Surašykime lotyniškos abėcėlės raides į lentelę. Galime pasirinkti eilučių ir stulpelių skaičių. Apibrėžtumo dėlei imkime 5×5 matmenų lentelę ir praleiskime raidę Q:

	1	2	3	4	5
1	*A*	*B*	*C*	*D*	*E*
2	*F*	*G*	*H*	*I*	*J*
3	*K*	*L*	*M*	*N*	*O*
4	*P*	*R*	*S*	*T*	*U*
5	*V*	*W*	*X*	*Y*	*Z*

Dabar kiekvieną teksto raidę galime keisti koordinačių pora:

$$A \mapsto 11, \quad B \mapsto 12, \ldots$$

Tačiau kas iš to? Juk tai tik vienas keitinys ir nėra jokio rakto! Raktą, nuo kurio priklauso keitinys nesunku įvesti. Pavyzdžiui, imkime kaip raktą žodį SNIEGAS ir pradėkime pildyti lentelę pradėdami rakto raidėmis, o po jų įrašydami raides, kurių dar trūksta:

	1	2	3	4	5
1	*S*	*N*	*I*	*E*	*G*
2	*A*	*B*	*C*	*D*	*F*
3	*H*	*J*	*K*	*L*	*M*
4	*O*	*P*	*R*	*T*	*U*
5	*V*	*W*	*X*	*Y*	*Z*

Dabar galime naudoti šią lentelę kaip keitinį. Kad ją galėtume sudaryti, pakanka žinoti raktą. Daugelis klasikinės kriptografijos šifrų naudoja šią ar panašią idėją. Vienas žymiausių pavyzdžių – ADFGVX šifras, kurį vokiečiai naudojo I pasaulinio karo metais.

Šiame šifre keitinį apibrėžia tokia lentelė:

	A	*D*	*F*	*G*	*V*	*X*
A	*A*	*B*	*C*	*D*	*E*	*F*
D	*G*	*H*	*I*	*J*	*K*	*L*
F	*M*	*N*	*O*	*P*	*Q*	*R*
G	*S*	*T*	*U*	*V*	*W*	*X*
V	*Y*	*Z*	1	2	3	4
X	5	6	7	8	9	0

Taikant šią lentelę kiekviena raidė keičiama abėcėlės ADFGVX raidžių pora. Tačiau nėra rakto! Pakeitę pranešimą šešių raidžių seka, šią seką vokiečiai dar kartą užšifruodavo naudodami perstatas. Tai gana saugus šifras. Vis dėlto prancūzų kriptoanlitikas G. Painvinas sugebėjo jį įveikti.

Panagrinėkime dar vieną istorinį šifrą, kuriame keitiniai savotiškai derinami su perstatomis. Tai F. Delastelle'io šifras. Šaltiniuose galima rasti įvairių jo variantų.

Pasinaudosime Polibijaus lentele, sudaryta pagal raktą SNIEGAS. Tai pirmoji rakto dalis. Antroji – bloko ilgis m. Pasirinkime $m = 6$. Tai reiškia, kad pranešimą skaidysime į blokus po 6 raides ir juos šifruosime. Užšifruosime žodį TANKUS. Pirmiausia, užrašysime raidžių koordinates stulpeliais:

T	*A*	*N*	*K*	*U*	*S*
4	2	1	3	4	1
4	1	2	3	5	1

Po to lentelės eilutes sujunkime, išskaidykime į skaitmenų poras ir jas vėl keiskime raidėmis naudodamiesi ta pačia Polibijaus lentele:

$$TANKUS \mapsto 421341412351 \mapsto 42\ 13\ 41\ 41\ 23\ 51 \mapsto PIOOCV.$$

Cezario ir kiti keitinių 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.[6]

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:

$$e(m_1 m_2 \ldots m_r|K) = c_1 c_2 \ldots c_r, \quad c_1 c_2 \ldots c_r = m_1 m_2 \ldots m_r \cdot K,$$

o dešifravimo operaciją

$$d(c_1 c_2 \ldots c_r|K) = m_1 m_2 \ldots m_r, \quad m_1 m_2 \ldots m_r = c_1 c_2 \ldots c_r \cdot K^{-1}.$$

Atvirkštinę matricą K^{-1} reikia skaičiuoti, žinoma, moduliu n, o žodis dauginamas 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.

[6] L. Hill. Cryptography in an Algebraic Alphabet. The American Mathematical Monthly, **36**, 1929, June-July, p. 306–312.

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 $m, 2m, \ldots$ 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.

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!

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, viename 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.

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 $C = c_1 c_2 \dots$

Jeigu jis žinotų rakto ilgį d, galėtų turimą šifrą suskaidyti į d fragmentų

$$\begin{aligned} C_1 &= c_1 c_{1+d} c_{1+2d} \dots \\ C_2 &= c_2 c_{2+d} c_{2+2d} \dots \\ &\dots\dots \\ C_d &= c_d c_{2d} c_{3d} \dots \end{aligned}$$

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 $2, 3, \dots, 10$, sudarytume tokią lentelę:

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.

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 $n = 32$):

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

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

$$\begin{matrix} c_1 & c_2 & c_3\ldots \\ c_4 & c_5 & c_6\ldots \end{matrix}',$$

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 ($d = 1, 2, \ldots$) 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

$$C = c_1 c_2 \ldots c_N.$$

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 $a_1, \ldots, a_n$ pasikartoja šifre atitinkamai $m_1, \ldots, m_n$ 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:

$$\begin{matrix} c_1 & c_{1+d} & c_{1+2d} & \ldots \\ c_2 & c_{2+d} & c_{2+2d} & \ldots \\ \vdots & \vdots & \vdots & \vdots \\ c_d & c_{2d} & c_{3d} & \ldots \end{matrix}.$$

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:

$$\begin{aligned} P(A) &= \frac{1}{C_N^2} \cdot \frac{N}{2} \cdot \left(\frac{N}{d} - 1\right), \\ P(A^c) &= \frac{1}{C_N^2} \cdot \frac{N(N - N/d)}{2}. \end{aligned}$$

Kadangi $P(x = y) = \kappa_c$, tai gausime formulę

$$\kappa_c = \frac{1}{C_N^2} \cdot \frac{N}{2} \cdot \left(\frac{N}{d} - 1\right)\kappa_t + \frac{1}{C_N^2} \cdot \frac{N(N - N/d)}{2}\kappa_0.$$

Iš šios lygybės jau galime surasti apytikslę formulę rakto ilgiui:

$$d \approx \frac{N(\kappa_t - \kappa_0)}{\kappa_c(N - 1) + \kappa_t - N\kappa_0}.$$

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“ [7] 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 correspondants*).

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

[7] Auguste Kerckhoffs. La cryptographie militaire. Journal des sciences militaires, vol. IX, Jan. 1883, p. 5—83, Feb. 1883, p. 161—191.

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.

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

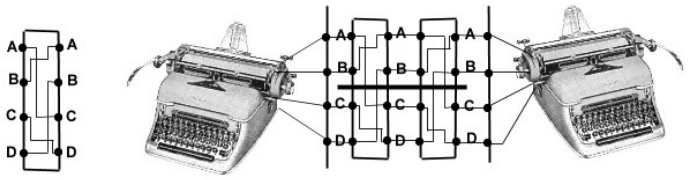
perduoti, tai c yra šifras, gautas panaudojus raktą k. Kaip gavėjas gali iššifruoti gautą šifrą c? Jeigu jo įrenginyje bus tokia pati sudėties grandinė, tai, įvedęs į ją eilutes c ir k, gaus m. Taigi šifruojama ir dešifruojama visiškai vienodai ir šiuos veiksmus gali atlikti pats įrenginys. Tiesa, jeigu 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 grįždavo 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

$$\lambda = \begin{pmatrix} 1 & 2 & \dots & n \\ i_1 & i_2 & \dots & i_n \end{pmatrix}.$$

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

$$\epsilon = \begin{pmatrix} 1 & 2 & \dots & n \\ 1 & 2 & \dots & n \end{pmatrix}.$$

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:

$$\rho_1 = \rho = \begin{pmatrix} 1 & 2 & \dots & n-1 & n \\ 2 & 3 & \dots & n & 1 \end{pmatrix}, \quad \rho^{-1} = \begin{pmatrix} 2 & 3 & \dots & n & 1 \\ 1 & 2 & \dots & n-1 & n \end{pmatrix}.$$

Savo ruožtu šifravimo keitinys bus

$$\sigma_1 = \rho^{-1}\lambda\rho.$$

Pasukę rotorių per $2, 3, \dots, m$ pozicijų, gausime plokštelių ir rotoriaus kontaktų atitikties keitinius $\rho^2, \rho^3, \dots, \rho^m$ ir $\rho^{-2}, \rho^{-3}, \dots, \rho^{-m}$. Vadinasi pasukę rotorių per m pozicijų, gauname tokį šifro keitinį:

$$\sigma_m = \rho^{-m}\lambda\rho^m, \quad m = 0, 1, 2, \dots, \rho^n = \epsilon.$$

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.

Blokiniai šifrai

Blokinių šifrų struktūra – keitinių ir perstatų sluoksniai.

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.

> ***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.*

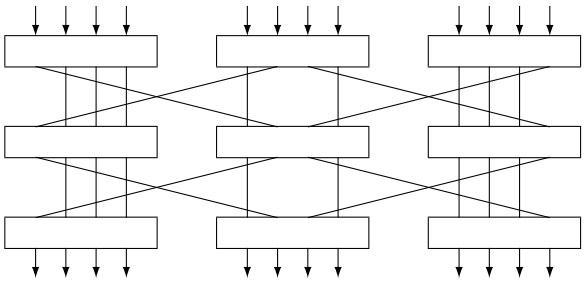
Taigi blokinės kriptosistemos šifravimo ir dešifravimo taisyklės yra funkcijos, kurių argumentai – dvejetainių žodžių poros:

$$e(\cdot|\cdot), d(\cdot|\cdot) : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n.$$

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.

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.

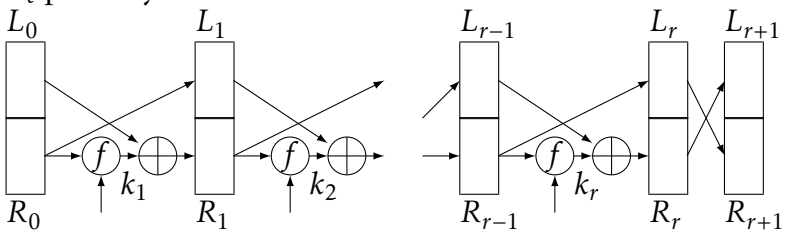
Tegu $M_0 = m_1 m_2 \dots m_n m_{n+1} \dots m_{2n}$ yra pradinis tekstas. Jo šifras gaunamas po r Feistelio iteracijų, kurias valdo daliniai raktai $k_1, k_2, \dots k_r$:

$$M = M_0 \overset{\downarrow k_1}{\longrightarrow} M_1 \overset{\downarrow k_2}{\longrightarrow} M_2 \longrightarrow \dots \longrightarrow M_{r-1} \overset{\downarrow k_r}{\longrightarrow} M_r = C.$$

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:

$$\begin{aligned} L_1 &= R_0,\ R_1 = L_0 \oplus f(R_0, k_1) \\ L_2 &= R_1,\ R_2 = L_1 \oplus f(R_1, k_2) \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ L_r &= R_{r-1}, R_r = L_{r-1} \oplus f(R_{r-1}, k_r) \\ L_{r+1} &= R_r, R_{r+1} = L_r. \end{aligned}$$

Kam gi reikalingas tas paskutinis žingsnis? Tuoj įsitikinsime, kad tai gudrus sumanymas.

Taigi po visų pertvarkymų gavome pradinio bloko $M = L_0R_0$ šifrą $C = L_0'R_0' = R_rL_r$. 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 $C_1 = L_1'R_1'$, čia

$$\begin{aligned} L_1' &= R_0' = L_r = R_{r-1}, \\ R_1' &= L_0' \oplus f(R_0', k_r) = R_r \oplus f(R_{r-1}, k_r) \\ &= L_{r-1} \oplus f(R_{r-1}, k_r) \oplus f(R_{r-1}, k_r) = L_{r-1}. \end{aligned}$$

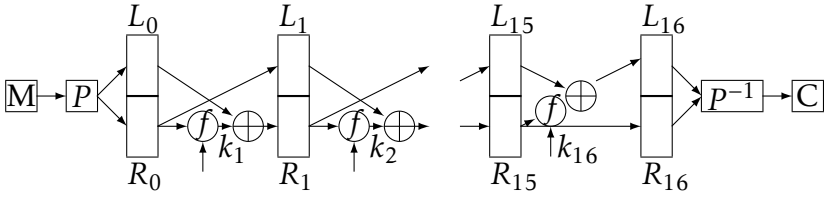
Taigi $C_1 = R_{r-1}L_{r-1}$. Atlikę r iteracijų, gautume $C_r = R_0L_0$, paskutiniame žingsnyje sukeitę bloko puses vietomis, gautume M. Šifravimo iteracijų seka tinka ir dešifravimui – tik raktus reikia naudoti „atbuline“ tvarka. Ir jokių reikalavimų funkcijai f!

DES

DES, arba Data Encryption Standard, yra kriptografijos mokslo brandos ženklas. Tai pirmoji kriptosistema valstybės institucijos įvertinta ir pripažinta šifravimo standartu.

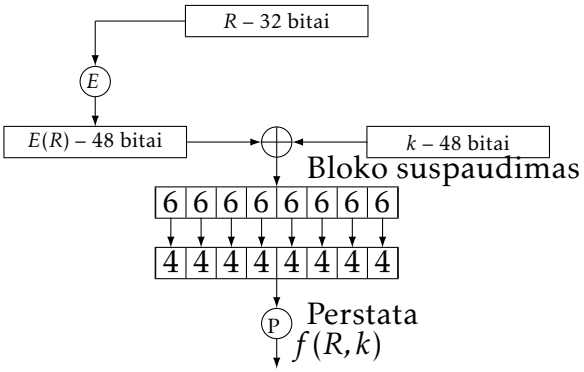
1973 metais JAV vyriausybė nusprendė, kad metas standartizuoti informacijos srautų apdorojimo metodus. Užduotis parengti šiuos standartus teko Nacionaliniam standartų institutui (NBS – National Bureau of Standards). Vienas iš daugelio tikslų, kurie buvo keliami, – parengti duomenų šifravimo standartą. 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:



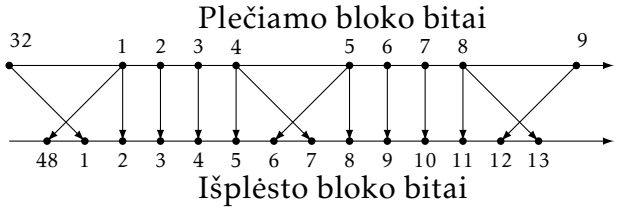
DES kriptosistemos schema. IP žymi tam tikrą pradinių duomenų perstatą, o IP^{-1} – šiai perstatai atvirkštinę.

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. [8]

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).

[8] FIPS 46-3, Data Encryption Standard (DES).http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

Penki režimai

Turite gerą blokinį šifrą? Laikas nuspręsti, kam ir kaip jį naudoti...

Blokinė kriptosistema šifruoja ir dešifruoja vieną fiksuoto ilgio žodį:

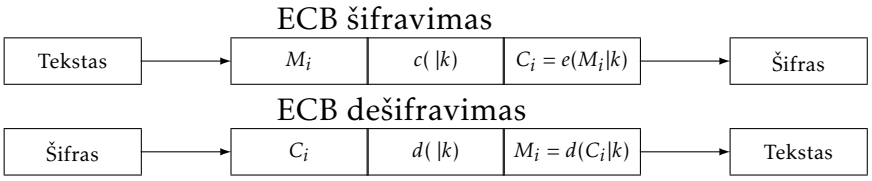
$$C = e(M|K),\ M = d(C|K) \quad M, C \in \{0,1\}^n.$$

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:

$$M_1 M_2 \ldots \mapsto C_1 C_2 \ldots, \quad C_j = e(M_j|K).$$



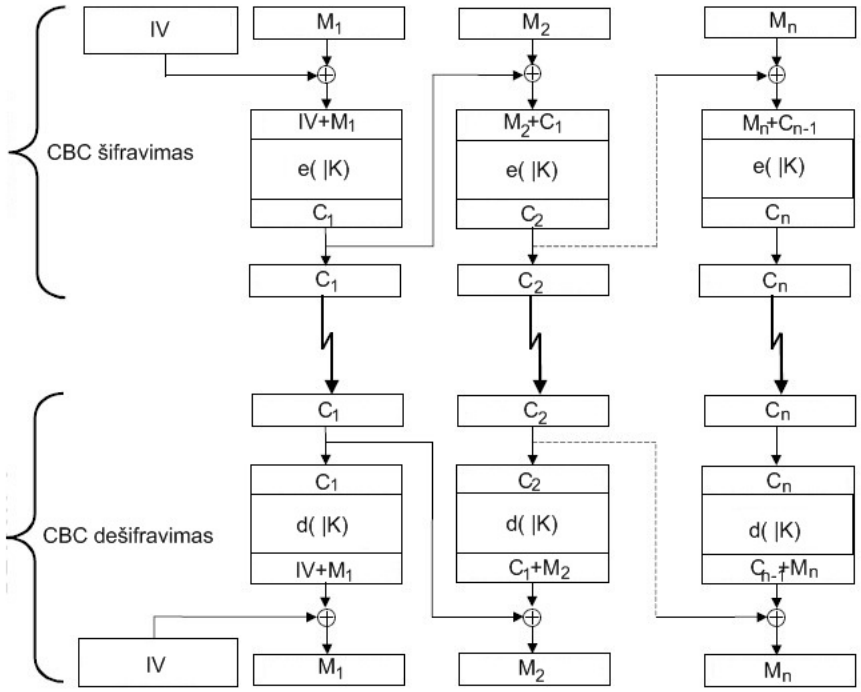
Duomenų šifravimas ir dešifravimas blokine kriptosistema ECB režimu.

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.

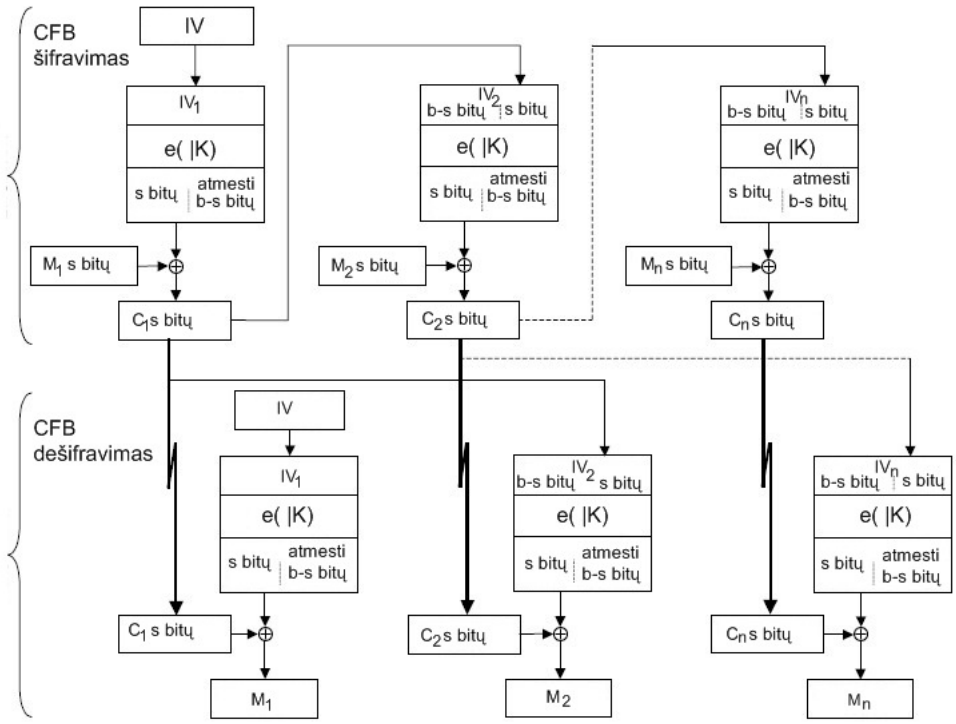
Pirmajam blokui šifruoti naudojamas žodis IV – pradžios (kriptografai sako – inicializacijos) vektorius. Taigi šifro gavėjas turi žinoti ne tik raktą, bet ir inicializacijos vektorių.

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, $1 \le s \le n$, 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.



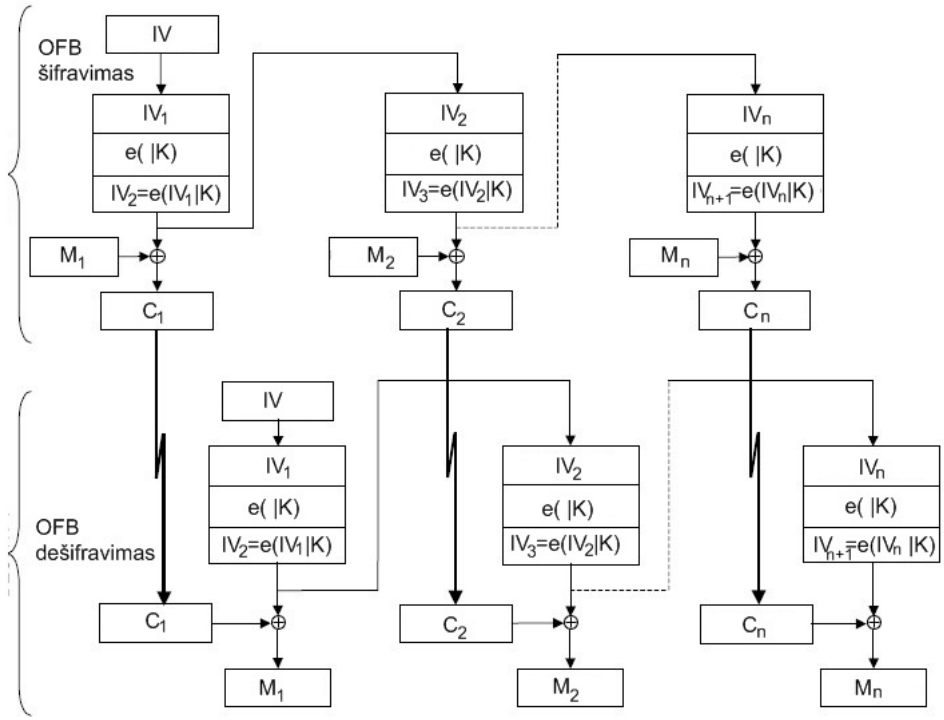
Duomenų šifravimas ir dešifravimas blokine kriptosistema CFB režimu.

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.

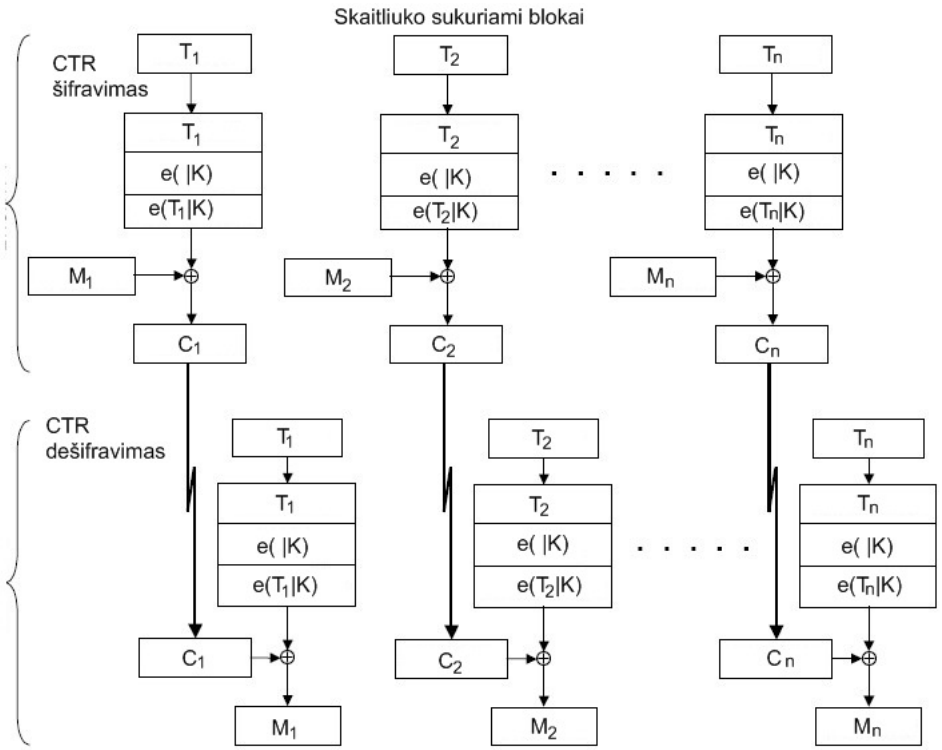


Duomenų šifravimas ir dešifravimas blokine kriptosistema OFB režimu.

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[9].

[9]Morris Dworkin. Recommendation for BlockCipher Modes of Operation Methods and Techniques. NIST Special Publication 800-38A 2001 Edition.

Srautiniai šifrai

Blokinių kriptosistemų kūrėjai sprendžia klausimą: kaip transformuoti duomenų srautą, kad gautume šifrą? Srautinių kriptosistemų kūrėjai atsako į šį klausimą pačiu paprasčiausiu būdu: pakanka sudėti duomenų ir rakto srauto bitus.

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.

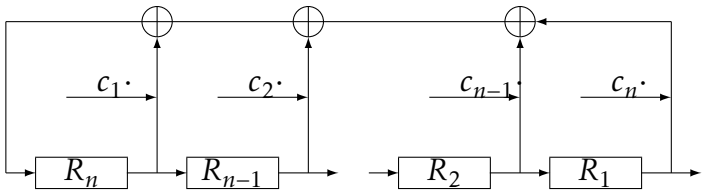
Taigi svarbiausias srautinių šifrų kūrėjų rūpestis – kaip generuoti pseudoatsitiktinių bitų srautus?

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

$$x_i(t) = x_{i+1}(t), \quad 1 \le i \le n-1, \tag{27}$$

$$x_n(t+1) \equiv c_1x_n(t) + \ldots + c_nx_1(t) \pmod 2, \tag{28}$$

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 atvaizduojamas 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

$$x_1(0), x_1(1), x_1(2), \dots \tag{29}$$

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.

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

$$x(0),\ x(0)C,\ \dots,\ x(0)C^m$$

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

$$c_1 = c_4 = 1,\ c_2 = c_3 = 0,\ \ x(0) = \langle 1,0,1,0\rangle.$$

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.

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.

Apibrėžimas. *Tiesinės registrų sistemos, nusakytos (28) lygybėmis charakteringuoju daugianariu vadinsime daugianarį*

$$P_n(x) = 1 + c_1x + \ldots + c_nx^n,\ c_n \neq 0.$$

O dabar – atsakymas į suformuluotą klausimą.

Teorema. *Tiesinė registrų sistema generuoja maksimalaus periodo seką tada ir tik tada, kai jos charakteringasis daugianaris yra primityvus.*

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.

Apibrėžimas. *Bendruoju 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:

$$\begin{aligned}
a &= q_0 b + r_0, \quad 0 < r_0 < b,\\
b &= q_1 r_0 + r_1, \quad 0 < r_1 < r_0,\\
r_0 &= q_2 r_1 + r_2, \quad 0 < r_2 < r_1,\\
&\ldots\ldots\ldots\\
r_{m-2} &= q_m r_{m-1} + r_m, \quad 0 < r_m < r_{m-1},\\
r_{m-1} &= q_{m+1} r_m + r_{m+1}, \quad 0 < r_{m+1} < r_m,\\
&\ldots\ldots\ldots\\
r_{n-2} &= q_n r_{n-1} + r_n, \quad 0 < r_n < r_{n-1},\\
r_{n-1} &= q_{n+1} r_n.
\end{aligned}$$

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 $(a,b) = u r_m + v r_{m+1}$, tai $(a,b) = v r_{m-1} + (u - v q_{m+1}) r_m$.

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}	$v, u - v q_{m+1}$
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: $r_m, r_{m+1}|\ q_{m+2}|\ u; v$, 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		

Apie žiedus $\mathbb{Z}_n$

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.

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.

Štai paprasčiausios lyginių savybės:

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 dažninausiai žymime $\mathbb{F}_p$. 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?

Apibrėžimas. *Tegu $n \geqslant 1$ yra natūralusis skaičius,*

$$\mathbb{Z}_n^* = \{m : 1 \leqslant m < n, (m,n) = 1\}.$$

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

$$\varphi(p) = p - 1, \quad \varphi(p^m) = p^m - p^{m-1}. \tag{30}$$

O dabar imkimės atvirkštinių klausimo.

Teorema. *Kiekvienas $\mathbb{Z}_n^*$ elementas turi atvirkštinį. Jeigu $a \in \mathbb{Z}_n^*$, tai*

$$a^{\varphi(n)} \equiv 1 \ (mod\ n). \tag{31}$$

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

$$a \cdot a_0, a \cdot a_1, \ldots, a \cdot a_{\varphi(n)} \tag{32}$$

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 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 (32) yra tie patys elementai $a_0, a_1, \ldots, a_{\varphi(n)}$, tik kitaip persirikiavę. Sudauginę juos visus ir pasinaudoję lyginių savybe, gausime

$$a^{\varphi(n)} \cdot a_0 \cdot a_1 \cdots a_{\varphi(n)} \equiv a_0 \cdot a_1 \cdots a_{\varphi(n)} \ (\text{mod } n), \quad a^{\varphi(n)} \equiv 1 \ (\text{mod } n).$$

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 skaičių teorijoje vadinamas Eulerio teorema. Vienas jos atvejis nusipelno atskiro vardo.

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.

Teorema. *Jeigu $p_1 < p_2 < \ldots < p_t$ yra visi pirminiai skaičiaus n dalikliai, tai*

$$\varphi(n) = n\left(1 - \frac{1}{p_1}\right) \cdot \left(1 - \frac{1}{p_2}\right) \cdots \left(1 - \frac{1}{p_t}\right).$$

Pasinaudojus šia Eulerio funkcijos išraiška nesunku įrodyti tokią šios funkcijos savybę:

Teorema. *Jeigu natūriniai skaičiai* m, n *yra tarpusavyje pirminiai, t.y.* $(m, n) = 1$*, tai*

$$\varphi(m \cdot n) = \varphi(m)\varphi(n).$$

Funkcijos, apibrėžtos natūraliųjų skaičių aibėje ir turinčios šią savybę, vadinamos multiplikatyviomis. Taigi Eulerio funkcija yra multiplikatyvi.

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

$$x^2 + bx + c \equiv 0 \pmod{n} \tag{33}$$

yra teisingas.

Jeigu pirminis p dalija n ir su kokia nors x reikšme (33) yra teisingas, tai su ta pačia reikšme bus teisingas ir lyginys

$$x^2 + bx + c \equiv 0 \pmod{p}. \tag{34}$$

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 (34) 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

$$x^2 \equiv a \pmod{p} \ (a \neq 0) \tag{35}$$

atvejį. Žinoma, 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 (35) lyginys turi sprendinį? Atsakyti į šį klausimą galima greitai, jeigu pasinaudosime gerais skaičių teorijos įrankiais.

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.

Teorema. *Su bet kokiu pirminiu skaičiumi p teisingos šios lygybės:*

$$\left(\frac{a^2}{p}\right) = 1, \quad \left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right),$$
$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod{p}, \quad \left(\frac{a+kp}{p}\right) = \left(\frac{a}{p}\right),$$
$$\left(\frac{-1}{p}\right) = (-1)^{(p-1)/2}, \quad \left(\frac{2}{p}\right) = (-1)^{(p^2-1)/8}.$$

Ir dar vienas teiginys, kuris padeda skaičiuoti Legendre'o simbolius. Tai Gauso dėsnis – vienas iš pačių įstabiausių skaičių teorijos teiginių.

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

$$x^2 \equiv 46 \pmod{57}.$$

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.

Teorema. *Tegu pirminis skaičius p, dalijant iš 4, duoda liekaną 3, t.y. $p \equiv 3 \pmod 4$. Jeigu lyginys $x^2 \equiv a \pmod p$ turi sprendinių, tai vienas iš jų yra*

$$x_0 \equiv a^{\frac{p+1}{4}} \pmod p.$$

Įrodymas. Kadangi lyginys turi sprendinių, tai iš Legendre'o simbolio savybių gauname

$$\left(\frac{a}{p}\right) = 1, \quad a^{\frac{p-1}{2}} \equiv \left(\frac{a}{p}\right) \pmod p, \quad a^{\frac{p-1}{2}} \equiv 1 \pmod p.$$

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

$$x^2 \equiv a \pmod{n} \tag{36}$$

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

$$x^2 \equiv a \pmod{p}, \quad x^2 \equiv a \pmod{q} \tag{37}$$

sprendinys. Kita vertus, jeigu rastume vieną x reikšmę, tinkančią abiem (37) lyginiams, ji būtų ir (36) sprendinys. Tačiau spręsdami (37) 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.

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*

$$a_i \cdot \frac{n}{n_i} \equiv 1 \pmod{n_i}, \quad i = 1, 2, \ldots, t.$$

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 $M \equiv m_i \pmod{n_i}$, $i = 1, 2, \ldots, t$.

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 (36), (37). 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 (36) 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!

Kuprinės kriptosistema

R. Merkle ir M. Hellmanas sukūrė viešojo rakto kriptosistemą, kuriai prigijo „kuprinės“ vardas[10]*. 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.*

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 W. Diffie, M. Hellman. New Directions in Cryptography. IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, p. 644–654.. 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.

[10] 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.

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.

Viešojo rakto kriptografija remiasi tuo, kad yra paprastai formuluojamų uždavinių, kurių skaitiniam sprendimui reikia milžiniškų skaičiavimo ir laiko išteklių. Vienas iš tokių uždavinių – kuprinės uždavinys.

„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.

Kai įvesties duomenų (svorių) yra daug, ją spręsti sudėtinga: greito algoritmo jai spręsti niekas nežino, o perranka reikalauja labai daug laiko. Tačiau kai „kuprinės“ svoriai w_i turi specialių savybių, „kuprinės“ uždavinys irgi gali būti greitai sprendžiamas.

Apibrėžimas. *Sakysime, kad skaičiai* $w_1, w_2, \ldots, w_n$ *sudaro sparčiai didėjančią seką, jei visiems* $i > 1$ *teisinga nelygybė*

$$w_1 + w_2 + \ldots + w_{i-1} < w_i.$$

Paprasčiausias sparčiai didėjančios svorių sistemos pavyzdys – natūraliojo skaičiaus a laipsniai:

$$a, a^2, a^3, \ldots, a^n.$$

Teorema. *Jei „kuprinės“ svoriai* $W = \{w_1, w_2, \ldots, w_n\}$ *sudaro sparčiai didėjančią seką, tai „kuprinės“ uždavinys sprendžiamas efektyviu 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.

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

$$O(n|v|) = O((|w_1| + \ldots + |w_n|)|v|).$$

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):

$$p > w_1 + w_2 + \ldots + w_n, \tag{38}$$

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

$$\begin{aligned} C_1 \equiv Cs \quad &\equiv m_1 s v_1 + m_2 s v_2 + \ldots + m_n s v_n \\ &\equiv m_1 w_1 + m_2 w_2 + \ldots + m_n w_n \pmod{p}. \end{aligned}$$

(38) 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[11].

„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: $C = e(m_1 m_2 \ldots m_n | K_v) = m_1 v_1 + \ldots + m_n v_n$.
Dešifravimas: $C_1 \equiv Cs \pmod{p}$, $C_1 = m_1 w_1 + \ldots + m_n w_n$, $m_1 \ldots m_n = d(C|K_p)$.

[11] 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

RSA

Ši trijų autorių – R. Rivesto, A. Shamiro ir L. Adlemano sukurta viešojo rakto kriptosistema yra pati populiariausia. Ja naudojamasi jau daugiau kaip keturiasdešimt 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.

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 (1,2,...,32), o tarpą tarp žodžių žymėti nuliu.

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.

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.

> ***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[12].

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.

> ***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 efektyvų algoritmą.*

Žodis „tikimybinis“ čia reiškia, kad pradėjus vykdyti algoritmą, sėkmė nėra garantuota. Jeigu nepasisekė – reikia kartoti algoritmą su kitais pradiniais duomenimis.

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.

[12] M. Wiener. Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information Theory. **36**, 1990, p. 553–558.

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

$$a_v \equiv a^{2^v t} \equiv a_{v-1}^2 \pmod n, \quad \text{tai} \quad a_v - 1 \equiv (a_{v-1}-1)(a_{v-1}+1) \equiv 0 \pmod n.$$

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:

$$C_1 \equiv M^{e_A} \pmod n, \quad C_2 \equiv M^{e_B} \pmod n.$$

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

$$C_1^a C_2^b \equiv M^{ae_A+be_B} \equiv M \pmod n.$$

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.

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.

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.

Shanko 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

$$x = im + j, \quad 0 \le i < m,\ 0 \le j < m. \tag{39}$$

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 (39) 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

$$\log_g y_1 = 2m + 4 = 14, \quad \log_g y_2 = m + 2 = 7.$$

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.

RSA kriptosistemos saugumas pagrįstas sveikųjų skaičių skaidymo uždavinio 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:

$$C_2 (C_1^a)^{-1} \equiv M\beta^k (g^{ka})^{-1} \equiv M g^{ak-ak} \equiv M \pmod{p}.$$

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 $C = e(M|K_v) = \langle C_1, C_2\rangle$. **Dešifravimas:** $d(C|K_p) = C_2(C_1^a)^{-1} = M$.

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.

Skaitmeninių parašų schemos

Skaitmeninio parašo schema yra tarsi „apversta“ kriptosistema: užšifruoti gali tik privataus rakto savininkas, o dešifruoti – visi.

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.

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:

$$ver(x, sig(x|K_p)|K_v) = 1. \tag{40}$$

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.

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 $y = y_1 y_2$ yra pranešimo $x = x_1 x_2$ 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. $R(x_1 x_2) \neq R(x_1) \cdot R(x_2)$.

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.

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

$$\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p. \tag{41}$$

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 (41) 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 (41) 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 (41) lyginį tenkinančius skaičius renkant juos kartu. Vienas būdas yra toks.

Pasirinkime skaičius i, j, tenkinančius sąlygą $0 \leqslant i, j \leqslant p-2, (j, p-1) = 1$. 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 (41) 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 $k, 1 \leqslant k \leqslant p-1$. Tačiau tikrąjį visada galima atsirinkti naudojantis sąlyga $\gamma \equiv \alpha^k \pmod{p}$.

Sužinojus k, jau aptartu būdu galima surasti ir a.

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 $0 \leqslant r < q-1$, 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 $0 \leqslant r < q-1$, 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.

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 $q \nmid \delta$. 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

$$\alpha^{e_1}\beta^{e_2} \pmod p \equiv \gamma \pmod q. \tag{42}$$

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 (42) 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: $\alpha^{e_1}\beta^{e_2} \equiv 8661$, $8661 \equiv 3858 \pmod q$, 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

$$\alpha^{e_1}\beta^{e_2} \pmod{p} \equiv \gamma \pmod{q},$$
$$e_1 \equiv x\delta^{-1} \pmod{q}, e_2 \equiv \gamma\delta^{-1} \pmod{q}.$$

Paslapties dalijimo schemos

Kriptografinėmis priemonėmis galima ne tik šifruoti ar kurti parašus. Galima, pavyzdžiui, padalyti paslaptį...

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.

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.

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 ($1 \leq t \leq n$), 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 $\mathbb{F}_q$.