VI. KODAVIMO TEORIJOS ĮVADAS

6.1 Iššifruojami kodai

Viena iš svarbesnių diskrečiosios matematikos sričių yra kodavimo teorija. Ankščiau, kodavimo priemonės atlikdavo pagalbinį vaidmenį įvairiose srityse, o kaip atskira matematinė disciplina ėmė vystytis gana nesenai. Kodavimas yra pagrindinė priemonė pateikiant informaciją kompiuteriui bei saugant ją kompiuterio atmintyje. Kodavimo dėka informacija apsaugoma nuo nesankcionuoto naudojimo, kodavimas naudojamas perduodant informaciją ryšio kanalais bei spaudžiant informaciją duomenų bazėse.

Suformuluokime kodavimo uždavinį. Tegu $A = \{a_1, a_2, \ldots, a_n\}$ ir $B = \{b_1, b_2, \ldots, b_m\}$ dvi netuščios aibės. Šias aibes vadinsime abėcėlėmis. Abėcėlės elementai vadinami raidėmis. Abėcėlės raidžių junginį $a_{i_1}a_{i_2}\ldots a_{i_k}$, $i_j \in \{1, \ldots n\}$ vadinsime žodžiu. Visų abėcėlės A žodžių aibę žymėsime simboliu A^*, o abėcėlės B žodžių aibę simboliu B^*. Tarkime, kad $S \subset A^*$ ir funkcija $f : S \to B$. Tada šią funkciją vadinsime kodavimu, aibės S žodžius- pranešimais, o žodžius $f(\alpha) = \beta$, $\alpha \in S$, $\beta \in B^*$ vadinsime kodais. Tiksliau kalbant β, yra pranešimo α kodas.

Jei $|B| = m$, tai funkcija f vadinama $m-$ ainiu kodavimu. Populiariausias atvejis, kai $m = 2$. Tada kodai vadinami dvejetainiais kodais. Funkcijai f atvirkštinė funkcija, jei ji egzistuoja, $f^{-1} : E(f) \to S$ vadinama *dekodavimu*.

Koduojant pranešimą galima elgtis dvejopai. Galima koduoti visą pranešimą arba koduoti atskiras pranešimo dalis. Bet koks abėcėlės simbolis yra vadinamas *elementaria pranešimo dalimi*. Tarkime duota abėcėlė $A = \{a_1, a_2, \ldots, a_n\}$ ir žodžių aibė A^*. Tegu $\alpha \in A^*$. Žodį sudarančių raidžių skaičių vadinsime *žodžio ilgiu*, ir žymėsime: $|\alpha| = |a_{i_1}\ldots a_{i_k}| = k$. Kartais patogu naudoti tuščio žodžio savoką. Žodį vadinsime tuščiu, jei jo ilgis lygus 0.

Jei $\alpha = \alpha_1\alpha_2$, α_1, α_2 du žodžiai, tai α_1 vadinamas *priešdėliu*, o α_2- *priesaga*.

Abėcėliniu kodavimu σ vadinsime kodų lentelę:

$$\sigma := (a_1 \to \beta_1, \ldots, a_n \to \beta_n), \quad a_i \in A, \quad , \beta_i \in K \subset B^*, i = 1, \ldots, n.$$

Abėcėlės raidžių kodai, priklausantys aibei K, vadinami *elementariaisiais kodais*.

Šią kodų lentelę vadinsime *sigma kodavimu*.

Pastebėsime, kad turint abėcėlinį kodavimą, galime koduoti ir bet kokius aibės pranešimus $\alpha \in A^*$:

$$f : A^* \to B^*, \quad \alpha = a_{i_1}\ldots a_{i_k}, \quad f(\alpha) = \beta_{i_1}\ldots\beta_{i_k}.$$

Abėcėlinis kodavimas σ vadinama *iššifruojamu kodavimu (iššifruojama schema)*, jei kiekvienas žodis sudarytas iš elementarių kodų, vieninteliu būdu yra išskaidomas elementariaisiais kodais, t.y. jei

$$\beta_{i_1}\ldots\beta_{i_k} = \beta_{j_1}\ldots\beta_{j_l}, \quad \text{tai} \quad k = l \wedge \forall t \in \{1, \ldots, k\}, \quad i_t = j_t.$$

Pastebėsime, kad jei schema iššifruojama, tai ji dekoduojama.

Tarkime, kad duotos dvi abėcėlės $A : \{1, 2, \ldots, 9\}, \quad B = \{0, 1\}$. Panagrinėkime schemas

$$\sigma_1 : (0 \to 0,\ 1 \to 1,\ 2 \to 10,\ 3 \to 11,\ 4 \to 100,\ 5 \to 101,\ 6 \to 110,)$$

$$7 \to 111,\ 8 \to 1000,\ 9 \to 1001,$$

ir

$$\sigma_2 : (0 \to 0000,\ 1 \to 0001,\ 2 \to 0010,\ 3 \to 0011,\ 4 \to 0100,\ 5 \to 0101,\ 6 \to 0110,)$$

$$7 \to 0111,\ 8 \to 1000,\ 9 \to 1001.$$

Matome, kad kodavimas f schemoje σ_1 nėra apverčiama funkcija, kadangi $f_{\sigma_1}(333) = 111111 = f_{\sigma_1}(77)$. Tuo tarpu schema σ_2 yra iššifruojama ir tuo pačiu dekoduojama.

Schemą σ vadinsime *p- schema*, jei elementarusis vienos raidės kodas nėra kitos raidės elementaraus kodo priešdėliu, t.y.

$$\text{jei } \forall i \neq j; \beta_i, \beta_j \in K, \text{ tai } \forall \beta \in B^*,\ \beta_i \neq \beta_j \beta.$$

p-schemos kodą vadinsime *p- kodu*.

1 Teorema p- schema yra dekoduojama.

$\ominus$

Naudosime prieštaros metodą. Tarkime, kad σ yra $p-schema$, bet nėra iššifruojama. Vadinasi, egzistuoja žodis $\beta \in f_\sigma(A^*)$ toks, kad

$$\beta = \beta_{i_1} \ldots \beta_{i_k} = \beta_{j_1} \ldots \beta_{j_l}, \text{ ir, } (\exists t \forall s (s < t \Rightarrow \beta_{i_s} = \beta_{j_s} \wedge \beta_{i_t} \neq \beta_{j_t})).$$

Kadangi

$$\beta_{i_t} \ldots \beta_{i_k} = \beta_{j_t} \ldots \beta_{j_l},$$

tai $\exists \beta'$ toks, kad $(\beta_{i_t} = \beta j_t \beta' \vee \beta_{j_t} = \beta_{i_t} \beta_{t'})$. Bet paskutinieji sąryšiai prieštarauja tam, kad schema yra p-schema.

$\oplus$

Kokios sąlygos turi būti tenkinamos, kad nagrinėjama schema būtų p-schema, o tuo pačiu ir dekoduojama? Teisinga tokia

2 Teorema (Makmilano nelygybė)*Jei schema* $\sigma : (a_i \to \beta_i)_{i=1}^n$ *iššifruojama, tai*

$$\sum_{i=1}^{n} \frac{1}{2^{l_i}} \leq 1, \quad l_i = |\beta_i|. \tag{1}$$

$\ominus$

Pažymėkime $l = \max\limits_{1\le i\le n} l_i$. Panagrinėkime (1) nelygybės sumos $n-$ąjį laipsnį. Turime, kad

$$(\sum_{i=1}^{n} \frac{1}{2^{l_i}})^n = \sum_{i_1,\ldots,i_n=1}^{n} \frac{1}{2^{l_{i_1}+\ldots+l_{i_n}}},$$

paskutinėje sumoje sumuojama pagal visus elementariųjų kodų rinkinius. Pažymėkime simboliu $\nu(n,t)$ dėmenų $\frac{1}{2^t}$ skaičių sumoje, kai $t = l_{i_1} + \ldots + l_{i_n}$. Taigi, sutraukę panašius narius turi:

$$\sum_{t=1}^{nl} \frac{\nu(n,t)}{2^t}.$$

Pastebėkime, kad kiekvieną dėmenį $2^{-(l_{i_1}+\ldots+l_{i_n})}$ vienareikšmiškai galime susieti su aibės B žodžiu $\beta_{i_1}\ldots\beta_{i_n}$. Matome, kad šio kodo ilgis yra t, taigi $\nu(n,t)$ yra žodžių $\beta_{i_1},\ldots\beta_{i_n}$, $|\beta_{i_1}\ldots\beta_{i_n}| = t$ skaičius. Kadangi nagrinėjame p-schemą, tai $\nu(n,t) \le 2^t$, nes priešingu atveju egzistuotų du vienodi žodžiai $\beta_{i_1}\ldots\beta_{i_n} = \beta_{j_1}\ldots\beta_{j_n}$ turintys skirtingus dėstinius.

Turime, kad

$$\sum_{t=1}^{nl} \frac{\nu(n,t)}{2^t} \le \sum_{t=1}^{nl} \frac{2^t}{2^t} = nl.$$

Tada $\forall n, (\sum\limits_{i=1}^{n} 2^{-l_i})^n \le ln$. Vadinasi,

$$\sum_{i=1}^{n} \frac{1}{2^{l_i}} \le \sqrt[n]{nl} = 1.$$

$\oplus$

Makmilano nelygybė yra ne tik būtina, bet ir tam tikra prasme ir pakankama sąlyga, kad schema būtų iššifruojama.

3 Teorema *Jei skaičiai $l_1,\ldots,l_n$ tenkina (1) nelygybę, tai egzistuoja iššifruojama abėcėlinio kodavimo schema $\sigma : (a_i \to \beta_i)_{i=1}^n$, $\forall i,\ l_i = |\beta_i|$.*

$\oplus$

Šios teoremos įrodymo nepateiksime. Jį galima rasti [2].

Panagrinėkime plačiai ankščiau naudotą informacijos perdavimo priemonę- *Morzės abėcėlę.* Abėcėlė koduojama tokiu būdu:

$$A \to 01, B \to 1000, C \to 1100, D \to 100, E \to 0, F \to 0010, G \to 110, H \to 0000, I \to 00,$$

$$J \to 0111, K \to 101, L \to 0100, M \to 11, N \to 10, O \to 111, P \to 0110, Q \to 1101,$$

$$R \to 010, S \to 000, T \to 1, U \to 001, V \to 0001, W \to 011, X \to 1001, Y \to 1011,$$

$$Z \to 1100.$$

Perduodant informaciją 0 ir 1 vietoje buvo naudojami simboliai $(\cdot)$ ir $(-)$ atitinkamai.

Patikrinkime, ar šiam kodavimui tenkinama Makmilano nelygybė. Turime, kad

$$\frac{1}{4}+\frac{1}{16}+\frac{1}{16}+\frac{1}{8}+\frac{1}{2}+\frac{1}{16}+\frac{1}{8}+\frac{1}{16}+\frac{1}{4}+\frac{1}{16}+$$

$$\frac{1}{8}+\frac{1}{16}+\frac{1}{4}+\frac{1}{4}+\frac{1}{8}+\frac{1}{16}+\frac{1}{16}+\frac{1}{8}+\frac{1}{8}+$$

$$\frac{1}{2}+\frac{1}{8}+\frac{1}{16}+\frac{1}{8}+\frac{1}{16}+\frac{1}{16}+\frac{1}{16}=$$

$$\frac{2}{2}+\frac{4}{4}+\frac{7}{8}+\frac{12}{16}=\frac{29}{8}>1.$$

Matome, kad Makmilano nelygybė nėra tenkinama Morzės abėcėlei, taigi ši schema nėra iššifruojama. Dekoduojant šia schema užkoduotą informaciją yra naudojami papildomi simboliai- pauzės tarp žodžių arba raidžių, kurių dėka informacijos dekodavimas tampa įmanomas.

6.2 Kodavimo nuostoliai. Kodavimo kaina

Koduojant informaciją ir ją perduodant arba saugant labai svarbu, kad raidžių kodai būtų kiek įmanoma trumpesni. Abėcėlinis kodavimas tinka bet kokiam pranešimų $S \subset A^*$ kodavimui. Jeigu apie pranešimų aibę S nėra nieko žinoma, tai spręsti optimizavimo uždavinį gana sudėtinga. Paprastai yra žinoma kokia nors informacija apie pranešimų aibę. Pavyzdžiui žinoma, kokios raidės ir kaip dažnai pasitaiko dideliame informaciniame pranešime. Tad natūralu, kad dažnai pasirodančios raidės būtų koduojamos trumpais kodais, o rečiau- ilgesniais.

Sakykime, kad schema $\sigma : (\alpha_i \to \beta_i)$ yra iššifruojama. Tada bet kokia kodavimo schema $\sigma' : (\alpha_i \to \beta_i^{'})$, čia $(\beta_1^{'}, \ldots, \beta_n^{'})$ yra kodų rinkinio $(\beta_1, \ldots, \beta_n)$ keitinys, taip pat bus iššifruojama. Jeigu elementariųjų kodų ilgiai vienodi, tai atlikus kodų perstatymą (t.y. pradinės abėcėlės raidėms priskirus kitus kodus) mes nepakeisime pranešimo ilgio. Tačiau situacija bus kita, jei elementarieji kodai yra skirtingo ilgio. Pranešimas pailgės, jei dažniau naudojamiems simboliams priskirsime ilgesnius kodus. Jeigu duotas konkretus pranešimas ir konkreti kodavimo schema, tai nesunku parinkti kodus taip, kad pranešimo ilgis būtų trumpiausias.

Tarkime, kad $k_1, \ldots, k_n-$ yra raidžių $a_1, \ldots, a_n$ dažniai pranešimo tekste, o $l_1, \ldots, l_n-$ yra elementariųjų kodų $\beta_1, \ldots, \beta_n-$ ilgiai atitinkamai. Tada, jei $k_i \le k_j$ ir $l_i \ge l_j$, tai

$$k_i l_i + k_j l_j \le k_i l_j + k_j l_i.$$

Įrodykime šią nelygybę. Pažymėkime $k_j = k_i + a$, ir $l_i = l_j + b$. Tada

$$(k_i l_j + k_j l_i) - (k_i l_i + k_j l_j) = (k_i l_j + (k_i + a)(l_j + b)) - (k_i(l_j + b) + l_j(k_i + a)) = ab \geq 0.$$

Iš paskutiniosios nelygybės išplaukia, kad norint minimizuoti pranešimo ilgį, reikia sutvarkyti pradinės abėcėlės raides pagal raidžių dažnumą (mažėjimo prasme), priskiriant šioms raidėms kodus nuo mažiausio iki didžiausio ilgio eilės tvarka. Taip sutvarkytas pranešimo kodas bus trumpiausias.

Naudojant šį metodą galime spręsti minimizavimo uždavinį tik fiksuoto (žinomo) pranešimo kodavimui. Beje, šis kodavimas optimaliausias.

Panagrinėkime dar vieną kodavimo schemą. Tarkime, kad kiekviena abėcėlės $A = \{a_1, \ldots, a_m\}$ raidė susieta funkciniu sąryšiu $g : A \to P$ su aibės $P = \{p_1, \ldots, p_n\}$ elementais, tenkinančiais lygybę

$$p_1 + \ldots + p_n = 1,\ p_1 \geq \ldots \geq p_n > 0.$$

Pastebėsime, kad $E(g) = P$. Jei $a_i \notin D(g)$, tai suprantama, kad abėcėlės raidė pranešime nepasirodo. Be to abėcėlėje raidės yra surikiuojamos (įvedamas tvarkos sąryšis) priklausomai nuo tikimybės dydžio, kalbant tiksliau, tikimybių mažėjimo atžvilgiu.

Kiekvienai atskiriamai kodavimo schemai $\sigma : (a_i \to \beta_i)$ apibrėžkime skaičių:

$$l_\sigma(P) = \sum_{i=1}^{n} p_i |\beta_i|$$

Šį skaičių vadinsime *vidutine kodavimo schemos σ kaina*, kai žinomas raidžių skirstinys P. Vidutinė schemos kaina kartais vadinama *vidutiniu kodavimo schemos ilgiu*.

Pažymėkime:

$$l_*(P) = \inf_\sigma l_\sigma(P), p_* = \min_{1 \leq i \leq n} np_i,\quad L = [\log_2(n-1)] + 1.$$

Visada egzistuoja atskiriama schema $\sigma : (a_i \to \beta_i)$ tokia, kad $\forall i, |\beta_i| = L$ (kodėl?) Tokią schemą vadinsime *tolygaus kodavimo* schema. Taigi, $1 \leq l_*(P) \leq L$ taigi, pakanka nagrinėti tokias schemas, kad $\forall i, p_i l_i \leq L$, kai l_i sveikas skaičius ir be to $l_i \leq L/p_*$. Taigi, egzistuoja baigtinis schemų σ skaičius, kurioms tenkinamos nelygybės: $l_*(P) \leq l_\sigma(P) \leq L$. Taigi, egzistuoja schema σ_* tokia, kad $l_{\sigma_*}(P) = l_*(P)$.

Abėcėlinis kodavimas σ_*, kuriam $l_{\sigma_*}(P) = l_*(P)$, yra vadinamas kodavimu su *minimaliais nuostoliais*, arba *optimaliu kodavimu*, kai duotas skirstinys P.

6.3 Fano algoritmas

Naudodami Fano algoritmą galime sukonstruoja iššifruojamą p- schemą, kuri artima optimaliai.

Įv: P: **array** [1...n] **of real** -"tikimybių, su kuriomis pasirodo raidės pranešime masyvas, kuriame bus surikiuotos tikimybės pagal mažėjimą: $1 \geq P[1] \geq \ldots \geq P[n] > 0,\ P[1] + \ldots + P[n] = 1$."

Išv: C: **array** [1... n, 1... L] **of** 0 ... 1 elementariųjų kodų masyvas.
Fano(1, n, 0) Fano procedūros iškvietimas.

Aprašysime rekursyvinę Fano procedūrą.

Įv: $b-$masyvo P pradžios indeksas, $e-$ masyvo P pabaigos indekas, $k-$ masyve C sukonstruotų kodų ilgiai.

Išv: masyvo C užpildymas

if $e > b$ **then**

$k := k + 1$ (eilinio kodo užkrovimas)

$m := Med(b, e)$ (masyvas dalinamas į dvi dalis)

for i **from** b **to** e **do**

$C[i, k] := i > m$ (pirmoje dalyje pridedame 0, antroje pridedame 1)

end for

Fano(b, n, k) (apdorojama pirmoji dalis)

Fano(m+1,e,k) (apdorojama antroji dalis)

end if

Funkcija Med suranda masyvo $P[b \dots e]$ nurodytos dalies medianą, kitaip tariant randa indeksą m, $(b \leq m < e)$ tokį, kad masyvo $P[b \dots m]$ elementų suma labiausiai artima masyvo $P[m + 1 \dots e]$ elementų sumai.

Įv: $b-$masyvo P nagrinėjamos dalies pradžios indeksas, $e-$ masyvo P nagrinėjamos dalies pabaigos indekas.

Išv: $m-$ medianos indeksas, t.y.

$$\min_{m \in b \dots e-1} |\sum_{i=b}^{m} P[i] - \sum_{i=m+1}^{e} P[i]|.$$

$S_b := 0$ pirmosios dalies elementtų suma

if $e > b$ **then**

for i **from** b **to** $e - 1$ **do**

$S_b := S_b + P[i]$ (visi išskyrus paskutinįjį)

end for

$S_e := P[e]$ (antrosios dalies elementų suma)

m:=e (pradedame ieškoti medianos nuo galo)

reapeat

$d := S_b - S_e$ (pirmosios ir antrosios dalių elementų sumų skirtumas)

$m := m - 1$ (medianą stumtelime žemyn)

$S_b := S_b - P[m]$; $S_e := S_e + P[m]$

until $|S_b - S_e| \geq d$

return m

Kiekvieną kartą vienoje dalyje kodus ilginame prirašydami nulius, o kitoje dalyje-vienetus. Tokiu būdu, pirmosios dalies kodai nebus kitos dalies priešdėliais. Kodus ilginti baigiame tada, kai dalies ilgis tampa lygus 1, taigi lieka vienintelis kodas.

p[i]	$C[i]$	$l[i]$
0.2	00	2
0.2	010	3
0.19	011	3
0.12	100	3
0.11	101	3
0.09	110	3
0.09	111	3
l[i,P]	–	2.80

6.4 Optimalus kodas

1 Lema *Sakykime, kad* $\sigma : (\alpha_i \to \beta_i)$ *yra optimalaus kodo schema, su skirstiniu* $P = \{p_1, \ldots, p_n\}, p_1 \geq \ldots \geq p_n > 0$. *Tada, jei* $p_i > p_j$, *tai* $l_j \geq l_i$.

$\ominus$

Naudosime prieštaros metodą. Tarkime, kad $i < j, p_i > p_j$ ir $l_i > l_j$. Nagrinėkime schemą

$$\sigma' : (a_1 \to \beta_1, \ldots, a_i \to \beta_j, \ldots a_j \to \beta_i, \ldots, a_n \to \beta_n).$$

Tada

$$l_\sigma - l_{\sigma'} = (p_i l_i + p_j l_j) - (p_i l_j + p_j l_i) = (p_i - p_j)(l_i - l_j) > 0.$$

Bet pastaroji nelygybė prieštarauja tam, kad $\sigma-$ optimali schema.

$\oplus$

Taigi, nemažindami bendrumo galime laikyti, kad $l_1 \leq \ldots \leq l_n$.

2 Lema *Jei schema* $\sigma : (a_i \to \beta_i)$ *yra optimalaus p-kodo schema, su skirstiniu* $P = \{p_1, \ldots, p_n\}, \quad p_1 \geq \ldots \geq p_n > 0$, *tai tarp elementarių kodų, kurių kodai maksimalaus ilgio, egzistuoja tik du, kurie skiriasi tik paskutiniu elementu.*

$\ominus$

Naudosime prieštaros metodą.

1. Tarkime, kad maksimalaus ilgio kodas yra vienas ir apibrėžtas tokiu būdu: $\beta_n = \beta b$, čia $b = 0$ arba 1. Tada bet kokiam $i \in \{1, \ldots, n-1\}$, $l_i \leq |\beta|$, t.y. visi kodai yra trumpesni.

Bet tuomet kodas β nėra kodų $\beta_1, \ldots, \beta_{n-1}$ priešdėlis, nes priešingu atveju $\beta = \beta_j$ ir β_j būtų kodo β_n priešdėlis. Vadinasi schema $\sigma' : (a_1 \to \beta_1, \ldots, a_{n-1} \to a \to \beta_{n-1}, a_n \to \beta)$ taip pat p-schema, be to $l_{\sigma'}(P) = l_\sigma - p_n$, o tai prieštarauja schemos σ optimalumui.

2. Sakykime, kad du kodai β_{n-1} ir β_n yra maksimalaus ilgio, besiskiriantys ne paskutiniuoju simboliu, $\beta_{n-1} = \beta' b'$ ir $\beta_n = \beta" b"$, $\beta' \neq \beta"$ ir be to β' ir $\beta"$ nėra kodų $\beta_1, \ldots, \beta_{n-2}$ priešdėliai ir atvirkščiai. Taigi schema $\sigma' : (a_1 \to \beta_1, \ldots, a_{n-2} \to \beta_{n-2}, a_{n-1} \to \beta' b', a_n \to \beta")$ taip pat yra p- schema. Be to $l_{\sigma'}(P) = l_\sigma - p_n$, o tai prieštarauja schemos σ optimalumui.

$\oplus$

4 Teorema *Jei* $\sigma_{n-1} : (a_1 \to \beta_1, \ldots, a_{n-1} \to \beta_{n-1})$ *yra optimali p- schema, su skirstiniu* $P : p_1 \geq \ldots \geq p_{n-1} > 0$ *ir* $p_j = q' + q"$, *kai*

$$p_1 \geq \ldots \geq p_{j-1} \geq p_{j+1} \geq \ldots \geq p_{n-1} \geq q' \geq q" > 0,$$

tai kodavimos schema

$$\sigma_n : (a_1 \to \beta_1, \ldots, a_{j-1} \to \beta_{j-1}, a_{j+1} \to \beta_{j+1}, \ldots, a_{n-1} \to \beta_{n-1}, a_j \to \beta_j 0, a_n \to \beta_j 1)$$

yra optimali p- schema, su skirstiniu $P : p_1 \ldots p_{j-1}, p_{j+1}, \ldots, p_{n-1}, q', q"$.

$\ominus$

1. Jei $\sigma_{n-1}-$ yra p-schema, tai remiantis σ_n konstrukcija taip pat p-schema.

2. $l_{\sigma_n} P_n = p_1 l_1 + \ldots + p_{j-1} l_{j-1} + p_{j+1} l_{j+1} + \ldots + p_{n-1} l_{n-1} + q' l_{j+1} + q" l_{j+1} =$

$$p_1 l_1 + \ldots + p_{j-1} l_{j-1} + (q' + q") l_j + q' + q" =$$

$$p_1 l_1 + \ldots + p_{n-1} l_{n-1} + p_j l_j + p_j = l_{\sigma_{n-1}}(P_{n-1}) + p_j.$$

3. Sakykime, kad schema $\sigma'_n : (a_1 \to \beta_1, \ldots, a_n \to \beta_n)$ yra optimali, su skirstiniu P_n. Tada, remiantis paskutiniąja lema gauname, kad

$$\sigma'_n : (a_1 \to \beta'_1, \ldots, a_{n-2} \to \beta'_{n-2}, a_{n-1} \to \beta 0, a_n \to \beta 1).$$

Pažymėję $l' = |\beta|$ panagrinėkime schemą

$$\sigma'_{n-1} : (a_1 \to \beta'_1, \ldots, a_j \to \beta, \ldots, a_{n-2} \to \beta'_{n-2}),$$

čia j apibrėžtas taip, kad $p_{j-1} \geq q' + q'' \geq p_{j+1}$.

4. $$l_{\sigma'_n} P_n = p_1 l_1 + \ldots + p_{n-2} l_{n-2} + q'(l' + 1) + q''(l' + 1) =$$

$$p_1 l_1 + \ldots + p_{n-2} l_{n-2} + (q' + q'') l' + q' + q'' = l_{\sigma'_{n-1}}(P_{n-1}) + p_j.$$

5. σ'_n p-schema, taigi ir σ'_{n-1} taip pat p-schema.

6. Kadangi σ_{n-1} yra optimali schema, tai $l_{\sigma'_{n-1}}(P_{n-1}) \geq l_{\sigma_{n-1}}(P_{n-1})$.

7. $l_{\sigma_n}(P_n) = l_{\sigma_{n-1}}(P_{n-1}) + p_j \leq l_{\sigma'_{n-1}}(P_{n-1}) + p_j = l_{\sigma'_n}(P_n)$. Žinome, kad σ'_n yra optimali schema, taigi ir σ_n optimali schema.

$\oplus$

6.5 Hafmano (Huffman) algoritmas

Tarkime, kad duotas abėcėlės raidžių skirstinys. Turint fiksuotą skirstinį sudarysime rekursyvinį algoritmą optimaliai p- schemai sudaryti.

Hafmano algoritmas

Įv: $n-$ abėcėlės raidžių skaičius, P: **array** [1...n] **of real** -"tikimybių masyvas, kuriame bus surikiuotos tikimybės pagal mažėjimą: $1 \geq P[1] \geq \ldots \geq P[n] > 0$, $P[1] + \ldots + P[n] = 1$."

Išv: C: **array** [1...n, 1...L] **of** 0...1 elementariųjų kodų masyvas. l: **array** [1...n] **of** 1...L elementariųjų kodų, optimalioje p-schemoje, ilgių masyvas.

if $n = 2$ **then**
$k := k + 1$ (eilinio kodo užkrovimas)
$C[1,1] := 0; l(1) := 1$(pirmas elementas)
$C[2,1] := 1; l(2) := 1$(antras elementas)
else
$q := P[n-1] + P[n]$ (sudedamos dvi paskutinės tikimybės)
$j := Up(n,q)$ (paieška vietos ir ją suradus sumos įstatymas)
Huffman$(P, n-1)$ (rekursyvus iškvietimas)
$Down(n,j)$ (kodo "ugdymas")
end if

Funkcija Up masyve P suranda vietą, kuriame turi būti skaičius q ir įrašę šį skaičių į surastą vietą visus likusius pastumia žemyn.

Įv: $n-$masyvo P nagrinėjamos dalies ilgis, $q-$ įstatoma suma.
Išv: pakeistas masyvas P.
for i **from** $n-1$ **downto , 2 do**
if $P[n-1] \leq q$ **then**
$P[i] := P[i-1]$ (pastumiamas masyvo elementas)
else
$j := i - 1$ (nustatoma įrašomo elemento vieta)
exit for i (viskas atlikta ir ciklo tęsti nereikia)
end if
end for
$P[j] := q$ (įrašomas elementas)
return j

Naudodami procedūrą $Down$ konstruojame optimalų n raidžių kodą, kai optimalus $n-1$ raidės kodas jau sukonstruotas. Realizuojant šį tikslą raidės, kurios numeris j kodas

laikinai pašalinamas iš masyvo $C[]$ pastumiant į viršų raidžių, kurių numeriai didesni negu j kodus, o po to nagrinėjamo masyvo $C[]$ pabaiga papildoma pora kodų, kurie gaunami iš raidės su numeriu j kodo, papildyto 1 arba 0, atitinkamai. Žemiau simboliu $C[i,*]$ žymėsime masyvo $C[]$ $i-$ąją eilutę.

Įv: $n-$masyvo P nagrinėjamos dalies ilgis, $j-$ išskiriamos raidės numeris.

Išv: optimalūs masyvo C pirmųjų n elementų kodai, l.

$c := C[j,*]$ fiksuojamas $j-$ osios raidės kodas

$l := l[j]$ fiksuojamas šio kodo ilgis

for i **from** j **to** $n-2$ **do**

$C[i,*] := C[i+1,*]$ kodas pastumiamas

$l[i] := l[i+1]$

end for

$C[n-1,*] := c;\ C[n,*] := c$ (kopijuojamas $j-$ osios raidės kodas)

$C[n-1,l+1] := 0;\ C[n,l+1] := 1$ (kodas pailginamas)

$l[n-1] := l+1; l[n] := l+1$ (padidinamas ilgis)

Komentaras. Jei raidės tik dvi, tai optimalaus kodavimo schema, esant bet kokioms tikimybėms akivaizdi- pirmąjai raidei priskiriame 1, o antrąjai- 0. Tada atliekama pirmosios dalies procedūra **if**. Rekursyvinė algoritmo dalis atkartoja paskutiniosios teoremos įrodymą. Naudojant procedūrą Up pradiniame sutvarkytame masyve $P[]$ atmetame dvi paskutines tikimybes (mažiausias) ir po to jų suma įrašoma į masyvą $P[]$ išlaikant sutvarkymą. Pastebėsime, kad įrašymo vieta fiksuojama indeksu j. Šis procesas tęsiamas iki tol, kol lieka masyvas tik iš dviejų elementų, kuriem optimalus kodas žinomas. Paskui atvirkštine tvarka konstruojamas optimalus kodas trims, keturiems ir t.t. elementams. Pastebėsime, kad dabar jau tikimybių masyvas $P[]$ jau nebereikalingas, reikalingi tik kodų numerių seka, kurie turi būti ištraukti iš kodų masyvo ir padubliavus po to pabaigoje prirašant simbolį (krūvį). Ši seka saugoma kintamuoju j, iškviečiant procedūrą Huffman.

Sukonstruosime Hafmano kodą, kai $n = 7$. Kairėje lentelėje demonstruojama kaip kinta masyvas $P[]$, o dešinėje- masyvas $C[]$. Juodu šriftu yra išskiriama kintamojo j reikšmė kiekvienu momentu.

0.20	0.20	**0.23**	**0.37**	**0.40**	**0.60**
0.20	0.20	0.20	0.23	0.37	0.40
0.19	0.19	0.20	0.20	0.23	
0.12	**0.18**	0.19	0.20		
0.11	0.12	0.18			
0.09	0.11				
0.09					

0	**1**	**00**	**01**	10	10
1	00	01	10	11	11
	01	10	11	000	000
		11	000	**001**	011
			001	010	011
				011	0010
					0011

,

Vidutinis kodo ilgis (kaina) yra

$$0.2 \cdot 2 + 0.2 \cdot 2 + 0.19 \cdot 3 + 0.12 \cdot 3 + 0.11 \cdot 3 + 0.09 \cdot 4 + 0.09 \cdot 4 = 2.78.$$

6.6 Kodai atsparūs triukšmui

Perduodant signalą kanalu dažnai susiduriama su jo iškraipymo problema. Tai gali būti sąlygota tiek kanalo vidinių savybių, tiek perdavimo bei priėmimo technikos charakteristikų. Pagrindinė problema- teisingai dekoduoti perduotą (perskaityti įrašytą) informaciją.

Tarkime, kad duotas ryšio kanalas C, kuriame yra trukdžių šaltinis:

$$C(S) = S',\ S \in A^*,\ \ S' \in B^*,$$

S perduotos informacijos aibė, $S'-$ priimtos informacijos aibė, kai buvo perduota S. Kodavimo funkcija f turinti savybes:

$$f(S) = K,\ C(K) = K',\ f^{-1}(K') = S,\ \forall s \in S,\ \ f^{-1}(C(f(s))) = s$$

vadinama kodavimo funkcija *atsparia triukšmui* .

Aptarsime klaidų klasifikavimo problemą. Paprastumo dėlei laikykime, kad $A = B = \{0, 1\}$. Kanale pasitaiko tokio pobūdžio klaidos.

1. $0 \to 1,\ 1 \to 0$ (simbolių sukeitimas vietomis);
2. $0 \to \Lambda,\ 1 \to \Lambda$ (simbolių pašalinimas);
3. $\Lambda \to 1,\ \Lambda \to 0-$ (papildomų simbolių įrašymas);

Sutarkime šias klaidas vadinti pirmo, antro ir trečio tipo klaidomis.

Paprastai kanalo klaidų skaičių žymėsime simboliu $\delta = \delta(a, b, c)$, skliaustuose nurodydami aukščiau išvardintų klaidų skaičių, kai a nurodo pirmo tipo klaidų skaičių, $b-$ antro tipo klaidų skaičių ir $c-$ trečio tipo klaidų skaičių.

Panagrinėkime pavyzdį. Tarkime, kad kanalo klaidų charakteristika $\delta = \delta(1, 0, 0)$, t.y. kanale galima pirmo tipo viena klaida, kai perduota n ilgio informacija. Tarkime kodavimo funkcija apibrėžta tokiu būdu:$f(a) = aaa$, o dekodavimas atliekamas atvirkštine funkcija $f^{-1}(abc) := a + b + c > 1$. Ši kodavimo funkcija nėra atspari kanalo triukšmui, kadangi perdavus $3n$ ilgio informaciją, galimos ne daugiau negu trys pirmo tipo klaidos, tačiau šios klaidos kode nebūtinai išsidėstę tolygiai. Pirmo tipo klaidos gali įvykti gretimuose koduose ir ši dekodavimo funkcija raides gali dekoduoti klaidingai.

Panagrinėkime klaidų taisymo galimybę. Pažymėkime simboliu E_s^δ visų galimų žodžių aibę, kurie gali būti gaunami iš žodžio s, kai jis buvo perduotas kanalu, kurio klaidų charakteristika δ. Taigi: $s \in S \subset A^*,\ \ E_s^\delta \subset B^*$. Jeigu $s' \in E_s^\delta$, tai ši konkreti klaidų seka, kurios dėka iš žodžio s gaunamas žodis s' bus žymima $E^\delta(s, s')$. Jei klaidų tipas kanale žinomas, tai tada indeksą δ paprastai praleisime.

5 Teorema Tam, kad egzistuotų triukšmui atsparus kodavimas būtina ir pakankama, kad visiems žodžiams $s_1, s_2 \in S,\ \ E_{s_1} \cap E_{s_2} = \emptyset$ egzistuotų aibės B^* skaidinys B_s, toks, kad $\cap B_s = \emptyset,\ \cup B_s = B^*$ ir $\forall s \in S, E_s \subset B_s$.

$\ominus$

Jei kodavimas yra triukšmui atsparus, tai $E_{s_1} \cap E_{s_2} = \emptyset$. Atvirkščiai. Tarkime, kad egzistuoja minėtas aibės B^* skaidinys. Remiantis šiuo skaidiniu konstruojame funkciją

$$f^{-1}(s') = s,\ \ \forall s' \in B_s.$$

$\oplus$

Panagrinėkime kanalą, kuriame galima pirmo tipo klaida, kuri įvyksta su tikimybe $0 < p < 0.5$. Klaidos įvyksta nepriklausomai. Tokį kanalą vadinsime *binariniu simetriniu* kanalu. Šiuo atveju bet koks žodis $s \in E_2^n$ gali būti pakeistas į žodį $s' \in E_2^n$. Taigi, $\forall s \in E_s = E_2^n$ ir ištaisyti visas klaidas kanale negalima.

Sakykime, kad E bet kokia netuščia aibė, o $\mathcal{R}_+ = [0, \infty)$.

Apibrėžimas *Funkciją $\rho : E \times E \to \mathcal{R}_+$, kuri bet kokiai porai $(x, y) \in E \times E$ tenkina reikalavimus:*

1) $0 \leq \rho(x, y) < \infty$,

2) $\rho(x, y) = 0 \leftrightarrow x = y$,

3) $\rho(y, x) = \rho(x, y)$,

4) $\rho(x, y) \leq \rho(x, z) + \rho(z, y)$, $x, y, z \in X$, *vadinsime metrika apibrėžta aibėje E.*

Sakykime, kad $\beta', \beta'' \in E^\delta(\beta', \beta'')$. Tada *Hamingo atstumu* tarp dviejų kodų vadinsime tokią funkciją:

$$d_\delta(\beta', \beta'') = \min_{E^\delta(\beta', \beta'')} |E^\delta(\beta', \beta'')| = \sum_{\substack{1 \leq i \leq n, \\ \beta_i' \neq \beta_i''}} 1.$$

Parodykime, kad apibrėžta funkcija d_δ yra atstumas.

1. Nesunku suprasti, kad $d_\delta(\beta', \beta'') = 0, \Leftrightarrow \beta' = \beta''$, kadangi klaidos yra simetrinės ir iš sekos $E(\beta', \beta'')$ sukeitus kodus vietomis gauname seką $E(\beta'', \beta')$, t.y.paskutinioji seka gaunama klaidas kode surašant atvirkščia tvarka, taigi:

2. $$d_\delta(\beta', \beta'') = d_\delta(\beta'', \beta').$$

3. $$d_\delta(\beta', \beta'') \leq d_\delta(\beta', \gamma) + d_\delta(\gamma, \beta''),$$

kadangi $E(\beta', \gamma) \cap E(\gamma, \beta'')$ yra seka iš kurios gaunamos sekos β' ir β'', tuo tarpu atstumas tarp sekų $d_\delta(\beta', \beta'')$ yra pats trumpiausias.

Tarkime, kad duota kodavimo schema $\sigma : (a_i \to \beta_i)$ ir d, kokia nors metrika aibėje B^*. Tada mažiausią atstumą tarp elementarių kodų

$$d(\sigma) = \min_{1 \leq i < j \leq n} d(\beta_i, \beta_j),$$

vadinsime *schemos σ atstumu tarp kodų.*

6 Teorema *Schemoje $\sigma : (\sigma_i \to \beta_i)$, kurioje atstumas tarp kodų apibrėžtas formule*

$$d_\delta(\sigma) = \min_{\beta', \beta'' \in V} d_\delta(\beta', \beta''),$$

dekoduojant yra taisoma p, δ tipo kaidų tada ir tik tada, kai $d_\delta(\sigma) = 2p$.

$\ominus$

Tegu $E(p, x)$ yra rutulys, kurio centras taške x, o spindulys p. Nagrinėjame kanalą, kuriame gali būti nedaugiau negu p, δ tipo klaidų. Tada, remiantis paskutiniąja teorema gauname, kad

$$E(p, \beta') \cap E(p, \beta'') = \emptyset \quad \Leftrightarrow d_\delta(\sigma) > 2p.$$

$\oplus$

Panagrinėkime Hamingo kodą, taisantį pirmo tipo vieną klaidą. Norint nustatyti ar perdavus infomaciją įvyko klaida, reikia siųsti papildomą informaciją, su kuria lygindami pagrindinį tekstą galėtume nustatyti ar įvyko klaida.

Tarkime, kad pranešimas $\alpha = a_1 \dots a_m$ koduojamas žodžiu $\beta = b_1 \dots b_n$, $n > m$. Laikome, kad kanale perdavimo metu gali įvykti ne daugiau negu viena klaida. (Paprastai šią savybę turi vidinės kompiuterio magistralės.)

Fiksuotam n, papildomus kodo simbolius parenkame taip, kad $2^k \geq n+1$, čia $k = n - m$. Turime, kad

$$2^k \geq n+1 \;\Leftarrow \frac{2^n}{n+1} \geq 2^{n-k} \Leftarrow \;\frac{2^n}{n+1} \geq 2^m.$$

Suskirstykime natūraliųjų skaičių aibę į poaibius priklausomai nuo to, kokią padėtį šiuose skaičiuose užima 1, jų dvejetainėje išraiškoje.

Tegu V_i- aibės, kurias sudarančių elementų dvejetainių kodų $i-$ oje pozicijoje yra vienetas:

$$V_1 = \{1,3,5,7,9,11,\dots\},\;\; V_2 = \{2,3,6,7,10,\dots\},\;\; V_3 = \{4,5,6,7,12,\dots\},\dots.$$

Panagrinėkime dvejetainio kodo $b_1 \dots b_n$ simbolius, esančius tokiose padėtyse: $2^0 = 1, 2^1 = 2, \dots, 2^{k-1}$. Šiuos elementus vadinsime *kontroliniais*. Likusius m simbolius vadinsime *informaciniais*. Informacinių simbolių vietoje patalpiname pranešimo $a_1 \dots a_m$ visus dvejetainio kodo simbolius eilės tvarka. Tada kontrolinius simbolius apibrėžiame tokiu būdu:

$$b_1 := b_3 + b_5 + b_7 + \dots,\;\; b_2 := b_3 + b_6 + b_7 + \dots,\;\; b_4 := b_5 + b_6 + b_7 + \dots,\;\; \dots$$

$$b_{2^{j-1}} := \sum_{i \in V_j \setminus \{2^{j-1}\}} b_i,$$

beje, čia sudėtis atliekama moduliu 2.

Laikykime, kad buvo gautas informacinis pranešimas $c_1 \dots c_n := C(b_1 \dots b_n)$,

$$\exists I, c_I = b_I \text{ arba } \overline{b_I}, \forall i \neq I, c_i = b_i,$$

čia I yra numeris simbolio, kuriame įvyko klaida. Numerį I užrašykime dvejetaine forma: $I = i_l \dots i_1$.

Apibrėžkime skaičių $J = j_l \dots j_1$ tokiu būdu:

$$j_1 := c_1 + c_3 + c_5 + c_7 + \dots,\;\; j_2 := c_2 + c_3 + c_6 + c_7 + \dots,$$

$$j_3 := c_4 + c_5 + c_6 + c_7 + \dots,\;\; \dots j_p := \sum_{i \in V_p} c_i.$$

7 Teorema *Teisinga lygybė:* $I = J$.

$\ominus$

Tarkime, kad $i_1 = 0$. Tada $I \notin V_1$ ir remiantis b_i apibrėžimu

$$j_1 := c_1 + c_3 + c_5 + c_7 + \ldots = b_1 + b_3 + b_5 + b_7 + \ldots = 0.$$

Tarkime, kad $i_1 = 1$. Tada $I \in V_1$ ir remiantis b_i apibrėžimu

$$j_1 := c_1 + c_3 + c_5 + c_7 + \ldots = b_1 + b_3 + b_5 + b_7 + \ldots \overline{b}_x + \ldots = 1,$$

kadangi pakeitus bent vieną kodo simbolį priešingu (laikome, kad 0 priešingas 1 ir atvirkščiai) visas sumos reikšmė keičiasi į priešingą. Taigi $i_1 = j_1$. Analogišku būdu, naudodami seką V_2 gauname, kad $i_2 = j_2$ ir t.t.. Taigi, gauname, kad $I = J$.

$\oplus$

Naudodamiesi paskutiniąją teoremą gauname teisingo dekodavimo taisyklę: reikia apskaičiuoti skaičių J. Jeigu $J = 0$, tai klaidos nėra, priešingu atveju $c_j := \overline{c_j}$. Po šio veiksmo (ištaisius pranešimą) gauname informacinį pranešimą, kuris klaidų neturi.

6.7 Duomenų spaudimas

Šiame skyrelyje nagrinėsime sąlygas, kurios sudaro prielaidas taupyti atmintį, perduodant informacijos srautus. Kitaip kalbant, kokius veiksmus reikia atlikti, kad laimėti laiko koduojant ir dekoduojant informaciją. Žinoma, šiam pocesui egzistuoja tam tikros ribos. Tam, kad galėtume informaciją suspausti, turime naudoti kitokias negu abėcėlinio kodavimo schemas. Apie tai dabar ir pakalbėsime.

Tarkime, yra nagrinėjamas pranešimas, užkoduotas kokiu nors būdu ir saugomas kompiuterio atmintyje. Pastebėsime, kad tolygus kodavimas nėra optimalus tekstų kodavimui. Paprastai tekstams koduoti naudojama mažiau negu standartiniai 256 simboliai. Be to raidžių dažniai taip pat skirtingi, priklausomai nuo kalbos. Tačiau fiksuotos kalbos tekstui, naudojantis Hafmano algoritmą, galima sukonstruoti optimalų abėcėlinį kodą. Mes to plačiau nenagrinėsime, bet yra įrodyta, kad koduojant plačiai naudojamas kalbas, galima pasiekti, kad kodo vidurkis (kaina) būtų lygi 6. Yra žinoma, kad spaudimo programos (zip. arj. ir kt.) turi žymiai geresnius rodiklius. Taigi, šiose programose naudojama ne abėcėlinio kodavimo schema.

Panagrinėkime tokią kodavimo schemą:

1. Pradinis pranešimas yra skaidomas į atskiras simbolių sekas, kurios vadinamos *žodžiais*.

2. Gauta žodžių aibė yra vadinama naujos abėcėlės raidėmis. Šiai aibei yra konstruojama abėcėlinio kodavimo p-schema (tolygaus kodavimo schema arba optimalaus kodavimo schema, jei kiekvienam žodžiui yra žinomas pasirodymo dažnumas tekste). Gautoji schema vadinama *žodynu*, šioje schemoje kiekvienam žodžiui priskiriamas kodas.

3. Pranešimo kodas sudaromas iš gautojo kodų žodyno kodų sekos.

4. Dekoduojant, pradinis pranešimas atstatomas keičiant žodžių kodus į žodyno žodžius.

Tarkime, kad reikia koduoti kokį nors lietuvišką tekstą. Žinoma, kad kiekvieno teksto žodžiai, vienas nuo kito, skiriami kableliais. Laikysime, kad tekste ne daugiau negu 2^{16}

skirtingų žodžių. Taigi, visus žodžius tolygiai koduojant galime ir numeruoti. Kadangi vidutinis lietuvių kalbos žodžių ilgis yra didesnis negu dvi raidės, tai galima suspausti apie 75% teksto. Pastebėsime, kad jei tekstas gana didelis (milijonai raidžių), vis tik saugojimo išlaidos nėra didelės.

Grįžkime prie kodavimo schemos sudarymo. Aukščiau aptartą metodą galima ir patobulinti, jei 2. žingsnyje konstruojant optimalų kodą, taikysime Hafmano metodą, prieš tai 1. žingsnyje tinkamai suskaidę tekstą, t.y. taip suskaidyti tekstą, kad tarp visų skaidinių nagrinėjamas turėtų mažiausią vidutinį ilgį. Toks kodavimo būdas vadinama *absoliučiai optimaliu*. Paprastai šis kodavimo būdas dėl didelių paruošiamųjų darbų laiko sąnaudų praktikoje vartojamas labai retai.

Aptarsime spaudimo (dažnai vadinamo adityvaus spaudimo) algoritmą, kuris dažnai naudojamas praktikoje. Skaitant tekstą, tuo pat metu konstruojamas žodynas ir koduojamas tekstas. Dar daugiau, žodynas nėra saugomas, kadangi dekoduojant naudojamas tas pat žodyno sudarymo algoritmas ir žodynas atstatomas iš karto.

Algoritmo pradžioje žodynas D : **array** [**init**] **of string** yra tuščias (jame yra žodis, kurio kodas 0.) Toliau tekste nuosekliai išskiriami žodžiai. Išskiriamas žodis, tai pats ilgiausias žodyne esantis žodis plius vienas simbolis. Į suspaustą turinį įrašomas surasto žodžio kodas ir papildomas simbolis, o žodynas papildomas išplėsta informacija.

Lempelo- Zivo algoritmą.

Įv: įvedamas pradinis tekstas, kuris aprašytas kodų masyvu f: **array** $[1 \dots n]$ **of char.**

Įved: suspaustas tekstas, apibrėžtas pora (p, q), čia $p-$ žodžio numeris žodyne, $q-$ papildomo simbolio kodas.

$D[0] :=''; d := 0$ pradinė žodyno būsena;

$k := 1$ eilinės raidės kodas tekste;

while $k \leq n$ **do**;

$p := FD(k)$ p rasto žodžio žodyne indeksas;

$l := Length(D[p])$ $l-$ rasto žodžio žodyne ilgis;

yield(p, f[k+l]) rasto žodžio kodas ir papildoma raidė;

$d := d + 1;\ D[d] := D[p] \cup f[k + l]$ papildome žodyną prijungdami papildomos raidės kodą;

$k := k + l + 1$ pasistumiame pirmyn tekste;

endwhile

Žodyne ieškome žodžio naudodami funkciją FD:

Įv: $k-$ simbolio tekste numeris, nuo kurio pradedant žodyne ieškomas tekstas;

Išv: $p-$ paties ilgiausio žodžio žodyne, sutampančio su simboliais $f[k] \dots f[k + l]$, indeksas, o jei tokio žodžio nėra, tai $p = 0$.

$l := 0,\ p := 0$ pradinė būsena;

for i **from** 1 **to** d **do**

if $D[i] = f[k \dots k + Length(D[i]) - 1] \vee Length(D[i]) > l$ **then**

$p := i; l := Length(D([i]))$ randame tinkamesnį žodį;

end if

end for

return p.

Išpakavimas atliekamas naudojant tokį algoritmą.

Įv: įvedamas suspaustas tekstas, kuris aprašytas porų masyvu g: **array** $[1 \dots m]$ **of record** p : **int;** q : **char endrecord,** čia $p-$ žodžio numeris žodyne, $q-$ papildomos raidės kodas;

Įved: pradinis tekstas, apibrėžtas simbolių ir eilučių seka;

$D[0] :=''; d := 0$ pradinė žodyno būsena;

for $k \leq n$ **do**;

$p := g[k], p$ p žodžio žodyne indeksas;

$q := g[k], q)$ $q-$ papildoma raidė;

yield$(p, \cup q)$ išvedamas žodis ir papildoma raidė;

$d := d + 1;\ D[d] := D[p] \cup q$ papildome žodyną;

endfor

Pastebėsime, kad tekstuose dažnai sutinkamos reguliarios sekos, pavyzdžiui tekstai su tarpais. Tokiais atvejais racionaliau naudoti specialius būdus pavyzdžiui, koduoti tarpų seką simbolių pora: (k, s), k yra tarpų skaičius, $s-$ tarpo kodas.

6.8 Tekstų šifravimas

Informacijos kodavimas siekiant ją apsaugoti nuo nesankcionuoto naudojimosi yra vadinamas *šifravimu*. Procesas, kurio metu informacija yra dekoduojama, vadinamas *dešifravimu*.

Koduota informacija yra vadinamas *šifru*. Pagrindinis reikalavimas keliamas šifruijis turėtų būti dešifruojamas, tik esant leidimui (papildomai informacijai), kuri vadinama *šifro raktu*. Paprastai šifrai yra vertinama priklausomai nuo jų atsparumo dešifravimui, o tai vertinama ekonominiais kriterijais. Šifras yra geras (atsparus dešifravimui), jei šifruota informacija vertinama mažiau, negu pastangos šifrui įveikti (nulaužti).

Šiame apžvalginiame skyrelyje nagrinėsime tik dvejetainės abėcėlės šifrus.

Tarkime turime atsitiktinių skaičių generatorių, kuris generuoja skaičius tokiu algoritmu:

$$T_{i+1} := (a \cdot T_i + b) \mathrm{mod} c,$$

T_i yra atsitiktiniai skaičiai, a, b, c žinomos konstantos. Paprastai yra naudojama $c = 2^n$, n priklauso nuo kompiuterio procesoriaus, b, nelyginis skaičius, $1 \equiv a \mathrm{mod} 4$. Taip apibrėžta skaičių seka turi periodiškumą c. Pseaudo atsitiktinių skaičių seka $T_0, T_1, \dots, T_n \dots$ yra vadinama *šifro gama*.

Šifravimo procesas apibrėžiamas tokiu būdu. Šifruojamas pranešimas apibrėžiamas n ilgio žodžių seka $S_0, S_1, S_2, \dots, S_n, \dots$, kurie sudedami su pseaudo atsitiktiniais skaičiais $T_0, T_1, T_2, \dots$, moduliu 2, t.y.

$$C_i = S_i + T_i.$$

Dešifruojant, šifruotą informaciją dar kartą sudedame su šifro gama:

$$S_i := C_i + T_i.$$

Šifro raktas yra pseudo atsitiktinių skaičių sekos pirmasis elementas T_0, kuris turi būti žinomas tik siuntėjui ir gavėjui. Šifrus vadinsime *simetriniais*, jeigu informacijos šifravimui ir dešifravimui yra naudojami tie patys raktai.

Jeigu pseudo atsitiktinių skaičių periodas c yra pakankamai didelis, t.y. šifro gama ilgesnė už pranešimą, tai dešifruoti informaciją galima tik atspėjus raktą.

Panagrinėkime kriptosistemos atsparumo problemą. Aukščiau pateiktas šifravimo metodas turi vieną esminį trūkumą: jei žinoma pranešimo pradinė dalis, tai visą pranešimą galima nesunkiai dešifruoti. Tarkime, kad žinome žodį S_i. Tada

$$T_i := C_i + S_i.$$

Taigi, galime atstatyti pseudo skaičius, o tuo pačiu ir raktą. Simetrinių šifravimo sistemų atsparumui didinti yra naudojami įvairūs būdai. Sudėtingu būdu sudaromos šifro gamos, vietoje + operacijos yra naudojamos kitos operacijos arba papildomai sukeičiami pradinės informacijos bitai. Šiuo metu patikimiausia simetrinė sistema yra DES (Data Encryption Standart), kurioje naudoje keli dešifravimą "apsunkinantys" būdai.

Pastaruoju metu plačiai naudojami *atviro rakto* šifrai. Jie nėra simetriniai. Informacijos šifravimui ir dešifravimui yra naudojami skirtingi raktai. Tuo tarpu šifravimo raktas yra visiems žinomas. Panagrinėkime šią kripto sistemą.

6.9 Atviro rakto šifrai

Atviro rakto sistemos sudaromos tokiu būdu:

1. Gavus pranešimą yra generuojamas atviras raktas, skaičių (n, e) pora ir slaptasis raktas d. Tai atliekamo tokiu būdu:

1) parenkami du pirminiai skaičiai p ir q;

2) sudaroma pirmoji atviro rakto dalis $n = pq$;

3) sudaroma antroji atviro rakto dalis- parenkamas mažas nelyginis skaičius e, kuris nedalo sandaugos $(p-1)(q-1) = \phi(n)$;

4) skaičiuojamas slaptasis raktas $d = e^{-1} \mathrm{mod}((p-1)(q-1))$;

5) perduodamas pranešimas.

2. Siuntėjas šifruoja pranešimą paprastai skaidydamas, jei reikia, į žodžius S_i, kurių ilgis nedidesnis negu $\log_2 n$:

$$C_i := S_i^e \mathrm{mod} n$$

ir šį pranešimą pasiunčia gavėjui.

3. Gavėjas dešifruoja pranešimą turėdamas slaptąjį raktą d :

$$P_i = C_i^d \mathrm{mod} n.$$

Teorema Šifravimas atviru raktu yra korektiškas, t.y. $P_i = S_i$.

$\ominus$

Nesunku suprasti, kad $P_i = S_i^{ed} \mathrm{mod} n$. Parodysime, kad $\forall M < n,\ M^{ed} \equiv M \mathrm{mod} n$.

Pastebėkime, kad d ir e yra vienas kitam atvirkštiniai skaičiai moduliu $(p-1)(q-1)$, t.y. egzistuoja $k \in \mathcal{N}$, kad

$$ed = 1 + k(p-1)(q-1).$$

Jeigu $M \neq 0 \mathrm{mod} p$, tai remiantis Ferma teorema gauname, kad:

$$M^{ed} \equiv M(M^{(p-1)})^{k(q-1)} \equiv M \cdot 1^{k(q-1)} \equiv M \mathrm{mod} p.$$

Jeigu $M = 0 \mathrm{mod} p$, tai akivaizdu, kad $M^{ed} \equiv M \mathrm{mod} p$.
Tada,

$$\forall M \in [0, n),\ M^{ed} \equiv M \mathrm{mod} p.$$

Samprotaudami analogiškai gauname, kad

$$\forall M \in [0, n),\ M^{ed} \equiv M \mathrm{mod} q.$$

Remdamiesi liekanų teorema gauname, kad

$$\forall M \in [0, n),\ M^{ed} \equiv M \mathrm{mod} n.$$

Kadangi $S_i < n$ ir $P_i < n$, tai gauname, kad visiems i, $P_i = S_i$.
$\oplus$

Panagrinėkime elektroninio parašo problemą. Atviro rakto šifrą galima naudoti ir kitiems tikslams. Organizuojant slaptą daugiakanalinį informacijos keitimąsi, pakanka sugeneruoti raktų poras (slapto ir neslapto) ir visiems ryšio dalyviams paskelbti atvirą raktą. Pastebėsime, kad šifravimo ir dešifravimo operacijos iš esmės yra vienodos ir skiriasi tik rodiklio laipsniu:

$$M = (M^e)^d \mathrm{mod} n = (M^{ed}) \mathrm{mod} n = M^{de} \mathrm{mod} n = (M^e)^d \mathrm{mod} n = M.$$

Panagrinėkime dviejų obonentų X ir Y bendravimo schemą. Siuntėjas X koduoja pranešimą S savo slaptu raktu $C := M^d \mathrm{mod} n$ ir gavėjui Y siunčia porą (S, C), kuri yra vadinama elektroniniu parašu. Gavėjas Y priėmęs šį pranešimą koduoja pranešimo parašą atviru raktu X, t.y. $S' = C^e \mathrm{mod} n$. Jeigu $S = S'$, tai reiškia, kad pradinis nekoduotas pranešimas S iš tiesų buvo siųstas siuntėjo X. Jei $S \neq S'$, tai arba pranešimas buvo iškraipytas arba netikras.

Tokio pobūdžio schemose egzistuoja ir kitų problemų. Tarkime, kad asmuo Z, turintis blogų ketinimų, turi technines galimybes kontroliuoti visą korespondenciją patenkančią į jūsų kompiuterį. Tada, perėmęs pranešimą X, kuriame buvo atviras raktas e gali pakeisti atvirą raktą savo atviru raktu. Po to asmuo Z gali falsifikuoti visus pranešimus X, pasirašydamas savo asmeniniu parašu ir tuo pačiu veikti kito vardu. Kitaip tariant, elektroninis parašas tik patvirtina, kad pranešimas S atėjo iš to paties šaltinio, iš kurio ir buvo siųstas raktas e.

Galima pasirašinėti ir elektroninius pranešimus. Tai atliekama tokiu būdu: siuntėjas X iš pradžių savo slaptu raktu koduoja pranešimą S, ir gauna elektroninį parašą C, o po to koduoja gautą porą (S, C) atviru gavėjo Y raktu. Gavęs šį pranešimą, Y iš pradžių iššifruoja jį savo slaptu raktu, o po to įsitikina gauto pranešimo teisingumu, palyginęs jį su rezultatu kuris buvo gautas taikant atvirą raktą X parašui C. Deja ir šis metodas negarantuoja, kad informacija nebus falsifikuota.

Uždaviniai

1. Ar pateikta abėcėlinio kodavimo schema yra i66ifruojama:

$$(a \to 0, b \to 10, c \to 011, d \to 1011, e \to 1111)?$$

2. Įrodykite, kad funkcija

$$d_\delta(\beta', \beta'') := 2 \min_{E\beta''' \in B^*} \max(\min_{E^\delta(\beta', \beta''')} |E^\delta(\beta', \beta''')|, \min_{E^\delta(\beta''', \beta'')} |E^\delta(\beta''', \beta'')|$$

yra metrika nesimetrinių klaidų aibėje.