IV dalis
Turing'o mašina

Šios kurso dalies pradinis variantas buvo iš dalies paruoštas pagal dr. Valdo Dičiūno diskrečiosios matematikos paskaitų konspektus. Autorius norėtų jam padėkoti už suteiktą galimybę pasinaudoti išeities tekstais ruošiant šiuos konspektus.

„Algoritmas“ yra pirminė matematikos bei informatikos sąvoka, panašiai kaip sąvokos „skaičius“, „aibė“ ir kitos. Neformaliai (intuityviai) algoritmas suprantamas kaip baigtinis taisyklių arba komandų rinkinys (dar vadinamas programa), nusakantis, kaip vienus objektus (duomenis), atlikus baigtinį skaičių nesudėtingų operacijų, perdirbti į kitus objektus (rezultatus).

Norint formaliai nagrinėti algoritmus kaip matematinius objektus, buvo pasiūlyti įvairūs skaičiavimo modeliai: Turing'o mašinos (beje, apibrėžtos gerokai anksčiau už realių kompiuterių atsiradimą), RAM (tiesioginės kreipties į atmintį) mašinos, dalinės rekursyviosios funkcijos, Markovo algoritmai. Šioje dalyje mes panagrinėsime paprasčiausią Turing'o mašinų variantą — vienajuostę determinuotą Turing'o mašiną.

Apibrėžimas. 1-juoste determinuota Turing'o[1] mašina *(arba tiesiog* Turing'o mašina*) vadiname rinkinį*

$$M = \langle \Sigma, Q, \delta, q_0, F \rangle,$$

kur:

- $\Sigma = \{a_1, \ldots, a_n\}$ *yra baigtinė aibė, vadinama* abėcėle*;*
- $Q = \{q_0, q_1, \ldots, q_k\}$ *yra baigtinė* būsenų aibė*;*
- $\delta : Q \times \Sigma \to Q \times \Sigma \times \{\mathrm{K}, \mathrm{N}, \mathrm{D}\}$ *yra dalinė*[2] pérėjimų funkcija*;*
- $q_0 \in Q$ – pradinė būsena*; ir*
- $F \subseteq Q$ *yra* galutinių būsenų aibė*.*

Turing'o mašina interpretuojama kaip mašina, turinti begalinę juostą, suskirstytą į ląsteles, ir palei juostą slenkančią skaitymo bei rašymo galvutę, kuri fiksuotu laiko momentu mato vieną juostos ląstelę ir gali perskaityti, koks abėcėlės Σ ženklas ten įrašytas, vietoje jo įrašyti kitą ženklą bei pasislinkti per vieną ląstelę į kairę arba į dešinę, arba likti stovėti toje pat vietoje. Be to, kiekvienu laiko momentu mašina yra kurioje nors būsenoje iš aibės Q ir ją gali pakeisti į bet kurią būseną iš aibės Q.

Vienas iš Turing'o mašinos abėcėlės ženklų vadinamas *tuščiu* („blank“) ir žymimas b (b $\in \Sigma$). Jis yra skirtas tuščioms ląstelėms žymėti ir negali būti naudojamas koduoti pradinius duomenis bei

[1] Alan Mathison Turing (Alanas Tiuringas, 1912-1954) – žymus britų matematikas, logikas, kriptografas ir karo didvyris, laikomas vienu iš informatikos mokslo tėvų. Antrojo pasaulinio karo metu vadovavo darbo grupei, iššifravusiai Enigmos kodą, naudotą vokiečių karo laivyno.

[2] Atitiktis $F \subseteq A \times B$ (žr. II dalies 2 skyrių) apibrėžia *dalinę funkciją* $f : A \to B$, jei kiekvienam $a \in A$ egzistuoja ne daugiau kaip vienas vektorius $(a, b) \in F$. Kaip matome, nuo „įprastų“ funkcijų dalinės skiriasi tuo, kad kai kurie elementai $a \in A$ gali neturėti vaizdų, t. y. dalinė funkcija gali būti apibrėžta ne visoje aibėje A, o tik kuriame nors jos poaibyje.

tarpinius rezultatus, su kuriais dirba Turing'o mašina. Duomenys tarp savęs atskiriami vienu ar daugiau tuščių ženklų ar kitais sutartiniais abėcėlės ženklais. Į kairę nuo pirmųjų ir dešiniau nuo paskutinių duomenų visa juosta yra užpildyta tuščiais ženklais. Taigi kiekvienu laiko momentu tik baigtinė juostos sritis gali turėti netuščias ląsteles (kadangi po baigtinio žingsnių skaičiaus galvutė bus pasiekusi tik baigtinį ląstelių skaičių į kairę ir į dešinę nuo pradinės ląstelės). Ši netuščia sritis (pradedant pirmąja iš kairės netuščia ląstele ir baigiant paskutiniąja dešinėje netuščia ląstele) dar yra vadinama *reikšmine juostos dalimi*.

Pradiniu laiko momentu mašina visada yra pradinėje būsenoje q_0 ir mato pirmą iš kairės netuščią ląstelę. Dirbdama mašina keičia būsenas, nuskaito ir įrašo simbolius į juostą ir juda palei juostą. Ką jinai turi daryti, nurodo perėjimų funkcijos δ reikšmės. Tarkime, kažkuriuo laiko momentu mašina yra būsenoje $q_{i_1} \in Q$, o jos skaitymo ir rašymo galvutė mato simbolį $a_{j_1} \in \Sigma$. Tada mašina elgiasi priklausomai nuo to, kokia yra reikšmė $\delta(q_{i_1}, a_{j_1})$. Tarkime, $\delta(q_{i_1}, a_{j_1}) = (q_{i_2}, a_{j_2}, J)$, kur $q_{i_2} \in Q$, $a_{j_2} \in \Sigma$ ir $J \in \{\mathrm{K}, \mathrm{N}, \mathrm{D}\}$. Tai reiškia, kad mašina turi pereiti į būseną q_{i_2}, vietoje simbolio a_{j_1} juostoje užrašyti a_{j_2}, ir pasislinkti į kurią nors pusę per vieną poziciją priklausomai nuo J reikšmės (K — į kairę, D — į dešinę, N — nejudėti, likti vietoje). Užrašas $\delta(q_{i_1}, a_{j_1}) = (q_{i_2}, a_{j_2}, J)$ vadinamas *komanda* (arba *instrukcija*).

Mašina sustoja, kai pereina į kurią nors galutinę būseną, t. y. į būseną iš aibės F. Mašinai sustojus jos *darbo rezultatu* vadinsime žodį, užrašytą netuščiose ląstelėse. Jei mašina visai nesustoja, jos darbo rezultatą laikome *neapibrėžtu*.

Turing'o mašina perdirba abėcėlės $\Sigma \setminus \{\mathrm{b}\}$ žodžius į kitus tos pačios abėcėlės žodžius.

Kiekvienu laiko momentu Turing'o mašiną galima visiškai apibrėžti nurodant jos būseną, galvutės vietą ir reikšminės juostos dalies turinį. Tokį momentinį Turing'o mašinos aprašą dar vadina *konfigūracija*. Taigi Turing'o mašinos konfigūracija yra žodis pavidalo

$$K = a_{j_1} \dots a_{j_{l-1}} q_{i_1} a_{j_l} \dots a_{j_m},$$

kuris reiškia, kad duotu momentu juostos reikšminėje dalyje stovi žodis $a_{j_1} \dots a_{j_{l-1}} a_{j_l} \dots a_{j_m}$, mašina yra būsenoje q_{i_1}, o jos skaitymo ir rašymo galvutė mato ženklą a_{j_l}. Tokiu atveju mašina ieško komandos, kurios kairėje pusėje stovi pora (q_{i_1}, a_{j_l}). Tarkime, mašina rado tokią komandą pavidalo $\delta(q_{i_1}, a_{j_l}) = (q_{i_2}, a_{j_t}, J)$. Tarkime, nagrinėjamoje komandoje judesys $J = \mathrm{D}$. Įvykdžius šią komandą mašinos konfigūracija pasikeičia iš konfigūracijos K į naują konfigūraciją

$$K' = a_{j_1} \dots a_{j_{l-1}} a_{j_t} q_{i_2} a_{j_{l+1}} \dots a_{j_m}.$$

Jei nauja mašinos būsena q_{i_2} yra iš aibės F, mašina sustoja. Jei ne, ieško kitos komandos.

Taigi Turing'o mašinos darbą nuo pradinio momento iki sustojimo (tarkime, po s žingsnių) galima visiškai aprašyti nurodant konfigūracijų seką $K_0 \to K_1 \to \dots \to K_s$, kur K_0 yra pradinė konfigūracija, o K_s yra galutinė konfigūracija po sustojimo.

Tarkime, duota Turing'o mašina M ir abėcėlės $\Sigma \setminus \{\mathrm{b}\}$ žodis (t. y. vektorius) u. Jei pradžioje juostoje buvo užrašytas žodis u, mašinos M darbo rezultatą žymėsime $M(u)$. Jei $M(u)$ neapibrėžtas, žymėsime ∞.

Toliau naudosime abėcėlę $\Sigma = \{0, 1, \mathrm{b}\}$ ir apibrėšime, kaip Turing'o mašina skaičiuoja dalines funkcijas $f : \mathbb{N} \to \mathbb{N}$. Natūralųjį skaičių u juostoje vaizduosime jo dvejetaine išraiška $\tilde{u}$. Sakysime, kad mašina M *skaičiuoja funkciją* $f : \mathbb{N} \to \mathbb{N}$, jei $\forall u \in \mathbb{N}$:

(1) $f(u)$ reikšmė apibrėžta tada ir tik tada, kai $M(\tilde{u})$ apibrėžtas, ir

(2) jei $f(u)$ apibrėžta, tai $M(\tilde{u})$ yra $f(u)$ dvejetainė išraiška.

Dalinę funkciją $f : \mathbb{N} \to \mathbb{N}$ vadiname *apskaičiuojama (pagal Turing'ą)*, jei egzistuoja Turing'o mašina M, skaičiuojanti f. Galima rasti funkcijų, kurios nėra apskaičiuojamos.

Panagrinėkime porą Turing'o mašinos pavyzdžių.

Pavyzdys. Parodysime, kad funkcija

$$f(x) = \begin{cases} 1, & x \text{ lyginis,} \\ 0, & \text{priešingu atveju} \end{cases}$$

yra apskaičiuojama pagal Turing'ą.

Turing'o mašina, skaičiuojanti šią funkciją, turėtų patikrinti paskutinio skaitmens reikšmę. Jei paskutinis skaitmuo yra 0, tai duotas skaičius yra lyginis, todėl reiktų užrašyti atsakymą 1. Priešingu atveju, jei paskutinis skaitmuo yra 1, duotas skaičius yra nelyginis, todėl reiktų užrašyti atsakymą 0. Mašina galėtų veikti taip: iš karto viską triname, važiuodami į dešinę ir įsimindami paskutinio nuskaityto simbolio reikšmę. Kaip įsimename? Ogi keisdami būseną. Pavyzdžiui, jei nuskaitome 0, keičiame būseną į q_0, o jei 1, tai į q_1. Kai viską ištrynę pasiekiame žodžio pabaigą, užrašome atsakymą priklausomai nuo to, kurioje būsenoje esame (t. y. priklausomai nuo to, koks buvo paskutinis nuskaitytas simbolis, kartu ir paskutinis žodžio simbolis), ir baigiame darbą. Užrašykime tai darančią Turing'o mašiną. Jos komandos bus tokios:

$$\begin{aligned} \delta(q_0, 0) &= (q_0, \mathrm{b}, \mathrm{D}) \\ \delta(q_0, 1) &= (q_1, \mathrm{b}, \mathrm{D}) \\ \delta(q_0, \mathrm{b}) &= (q_2, 1, \mathrm{N}) \\ \delta(q_1, 0) &= (q_0, \mathrm{b}, \mathrm{D}) \\ \delta(q_1, 1) &= (q_1, \mathrm{b}, \mathrm{D}) \\ \delta(q_1, \mathrm{b}) &= (q_2, 0, \mathrm{N}) \end{aligned}$$

o galutinių būsenų aibė $F = \{q_2\}$. Patogiausia komandas užrašyti lentele, kurioje stulpeliai sunumeruoti abėcėlės Σ simboliais, o eilutės — būsenomis:

	0	1	b
q_0	$q_0, \mathrm{b}, \mathrm{D}$	$q_1, \mathrm{b}, \mathrm{D}$	$q_2, 1, \mathrm{N}$
q_1	$q_0, \mathrm{b}, \mathrm{D}$	$q_1, \mathrm{b}, \mathrm{D}$	$q_2, 0, \mathrm{N}$

Beje, šio pavyzdžio funkciją apskaičiuoti galima ir kitu būdu. Nesunku įsitikinti, kad šią funkciją skaičiuos tokia mašina:

	0	1	b
q_0	$q_0, 0, \mathrm{D}$	$q_0, 1, \mathrm{D}$	$q_1, \mathrm{b}, \mathrm{K}$
q_1	$q_2, 1, \mathrm{K}$	$q_2, 0, \mathrm{K}$	—
q_2	$q_2, \mathrm{b}, \mathrm{K}$	$q_2, \mathrm{b}, \mathrm{K}$	$q_3, \mathrm{b}, \mathrm{N}$

kurios galutinių būsenų aibė yra $F = \{q_3\}$.

Iš tikro, būsenoje q_0 galvutė tiesiog važiuoja į dešinę, kol pasiekia žodžio x pabaigą, t. y. nuskaito b. Nuskaičiusi b, galvutė pasislenka per vieną poziciją į kairę, t. y. atsistoja ties paskutiniu netuščiu simboliu, ir pakeičia būseną į q_1. Būdama būsenoje q_1 mašina patikrina, ar tas paskutinis simbolis yra 0, ar 1. Jei 0, tai skaičius x yra lyginis, todėl atsakymas turėtų būti 1. Užrašome jį ir pereiname į būseną q_2. Analogiškai darome, jei paskutinis simbolis yra 1. Būdama būsenoje q_2, galvutė viską trina, važiuodama į kairę. Ištrynusi viską, t. y. radusi b, pereina į galutinę būseną, t. y. baigia darbą. Juostoje lieka užrašytas tik atsakymas — 0 ar 1. □

Pavyzdys. Sudarykime Turing'o mašiną, skaičiuojančią funkciją $f(x) = x+1$, čia x — natūralusis skaičius, užrašytas dvejetainiu pavidalu.

Kaip ta mašina turėtų veikti? Kad būtų lengviau įsivaizduoti, imkime pavyzdį. Tegu $x = 1100111$. Tada $x+1 = 1101000$. Matome, kad funkcijos f skaičiavimo algoritmas galėtų būti toks:

visus dešinėje skaičiaus pusėje esančius vienetus verčiame nuliais, pirmą iš dešinės nulį verčiame vienetu, o kitkas lieka nepakitę. Toks būtų bendras algoritmas. Bet nepamirškime, kad būna išskirtinių atvejų, kai bendras algoritmas netinka. Šiuo atveju bendras algoritmas netinka, jei nėra jokio pirmo iš dešinės nulio, t. y. jei skaičius sudarytas vien iš vienetukų, pavyzdžiui, $x = 1111$. Tokiu atveju atsakymas turėtų būti $x+1 = 10000$. Taigi vienetu verčiame pirmą iš dešinės nulį, o jei tokio nėra, tai pirmą tuščią simbolį. Apibendrinkim: galvutė iš pradžių turi nuvažiuoti iki skaičiaus pabaigos (nes pradžioje ji stovi ties pirmu iš kairės netuščiu simboliu) — tai vyksta būsenoje q_0, o tada, važiuodama iš dešinės į kairę, visus vienetus verčia nuliais, o pirmą aptiktą nulį arba tuščią simbolį verčia vienetu ir sustoja. Tai daro tokia mašina: $F = \{q_2\}$,

	0	1	b
q_0	$q_0, 0, \mathrm{D}$	$q_0, 1, \mathrm{D}$	$q_1, \mathrm{b}, \mathrm{K}$
q_1	$q_2, 1, \mathrm{N}$	$q_1, 0, \mathrm{K}$	$q_2, 1, \mathrm{N}$