4. Turingi masinad

4.1. Turingi masina mõiste

Algoritm on eeskiri, mis määrab teatavat liiki ülesannete lahendamiseks vajalikud operatsioonid ja nende järjekorra. Üks vanemaid ja tuntumaid on antiikajast pärinev Eukleidese algoritm kahe naturaalarvu suurima ühisteguri leidmiseks. Sajandite vältel algoritme on koostatud palju, näiteks kuupvõrrandi lahendamiseks, arvude tegurdamiseks, esemete optimaalse valiku või paigutuse leidmiseks jne.

Teiselt poolt aga kogunes ajapikku ülesandeid, mille jaoks ei õnnestunud leida ei efektiivset ega üleüldse ühtki algoritmi. Kõige tuntum sellistest ülesannetest on Hilberti kümnes probleem: leida meetod, millega saaks kontrollida, kas suvalisel täisarvuliste kordajatega n-muutuja polünoomil on täisarvuline nullkoht.

Niisuguses olukorras on mõistlik püüda välja selgitada ülesannete klassid, milliste lahendamiseks leidub algoritm ja milliste jaoks ei leidu. Kõigepealt tuleb aga täpselt fikseerida, mida mõista algoritmi all. Erinevaid definitsioone on mitmeid, meie vaatleme nendest inglise matemaatiku A. Turingi (1912–1954) oma, mille ta esitas aastal 1936.

Turingi masin koosneb järgmistest komponentidest: lint, lugev-kirjutav pea, sisemälu ja tabel.

Lint on mõlemast suunast lõpmatu ning jagatud ühesuurusteks lahtriteks. Igasse lahtrisse võib olla kirjutatud üks sümbol lõplikust tähestikust $\{s_1, \ldots, s_m\}$ või tühisümbol s_0. Eeldame, et alguses on lindil vaid lõplik hulk tühisümbolist erinevaid sümboled ja et kõik ülejäänud lahtrid on täidetud tühisümbolitega.

Lugev-kirjutav pea võib liikuda mööda linti lahtrist lahtrisse, lugeda lahtris oleva sümboli või kirjutada lahtrisse seal oleva sümboli asemele teise.

Sisemälu on igal ajamomendil ühes lõplikust hulgast seisunditest. Seisundid jagatakse kahte liiki: aktiivsed seisundid $q_1, \ldots, q_k$ ja passiivsed $q_0^{(1)}, \ldots, q_0^{(l)}$. Kui passiivseid seisundeid on ainult üks, siis jätame ülemise indeksi kirjutamata. Masin alustab tööd seisundis q_1, töö käigus võib ta siirduda ühest seisundist teise.

Tabel sisaldab instruktsioone pea tegevuse ja seisundite muutumise kohta. Iga selline instruktsioon esitub *käsu* kujul

$$s_a q_b \to s_c q_d K,$$

kus s_a ja s_c tähistavad sümboleid ($0 \le a, c \le m$), q_b ja q_d seisundeid ($1 \le b \le k$, $0 \le d \le k$) ning K on üks tähtedest L, R, C. Sama vasaku poolega käske

ei tohi olla rohkem kui üks. Vasaku poole järgi koondatakse käsud tabelisse

	s_1	...	s_a	...	s_n
q_1					
⋮					
q_b			$s_c q_d K$		
⋮					
q_m					

Mõned lahtrid võivad ka tühjaks jääda, kui vastava vasaku poolega käsku ei ole. Käskude tabelit nimetatakse ka masina *programmiks*.

Turingi masin töötab taktide kaupa. Oletame, et pea all on sümbol s_a ning masin on seisundis q_b. Kui tabeli lahtris, mis vastab sümbolile s_a ning seisundile q_b, leidub käsk $s_c q_d K$, siis kirjutatakse lindile sümboli s_a asemele sümbol s_c, masin läheb seisundist q_b seisundisse q_d ning pea liigub ühe positsiooni võrra vasakule, ühe positsiooni võrra paremale või ei liigu üldse vastavalt K väärtusele L, R või C. Sellega on üks takt läbi. Järgmisel taktil kogu tegevus kordub. Masin lõpetab töö, kui ta läheb mingi käsu toimel passiivsesse seisundisse või kui täitmisjärg jõuab tühja lahtrisse.

Vaatamata sellele, et Turingi masina operatsioonid on väga elementaarsed, pole siiani leitud algoritmi, mida ei saaks temal realiseerida (jätame kõrvale niisugused, mis kasutavad näiteks juhuslikkust mingi füüsikalise protsessi näol). Algoritmi täitmiseks kulunud taktide arv sobib seega tema keerukuse mõõduks.

4.2. Turingi mõttes arvutatav funktsioon

Kuna Turingi masina mõiste on kasutusele võetud algoritmide teoreetiliseks uurimiseks, on tema praktilise kasutamisega seotud küsimused teisejärgulised. Teoreetilisest seisukohast pole tähtsust, et Turingi masinaga ei saa lahendada mitmeid tänapäeva arvutite standardülesandeid, näiteks joonistada ekraanile pilte või printida teksti. Kooskõlas masina ehitusega on uuritavad küsimused põhimõttelist ja fundamentaalset laadi. Järgnevas võtame vaatluse alla naturaalarvuliste funktsioonide arvutatavuse. Kõigepealt tuleb aga kokku leppida, kuidas arve Turingi masina lindil esitada.

Fikseerime tähestiku järgmiselt: $s_0 =$ `-`, $s_1 =$ `0`, $s_2 =$ `I`. Naturaalarvu x esitus lindil olgu

$$\texttt{0}\underbrace{\texttt{II}\ldots\texttt{I}}_{x \text{ kriipsu}}$$

ja naturaalarvude järjendi $(x_1, \ldots, x_n)$ esitus

$$\mathtt{0}\underbrace{\mathtt{II}\ldots\mathtt{I}}_{x_1 \text{ kriipsu}}\mathtt{0}\underbrace{\mathtt{II}\ldots\mathtt{I}}_{x_2 \text{ kriipsu}}\ldots\mathtt{0}\underbrace{\mathtt{II}\ldots\mathtt{I}}_{x_n \text{ kriipsu}}.$$

Neid kokkuleppeid arvestades saab defineerida, mis on Turingi masina poolt arvutatav funktsioon ja mis on Turingi mõttes arvutatav funktsioon. Funktsioonid võivad olla mitme argumendiga (n muutuja funktsioonid) ning võivad teoreetiliste konstruktsioonide hõlbustamiseks olla ka mitte kõikjal määratud.

Definitsioon 1. Turingi masina $\mathcal{M}$ poolt arvutatav funktsioon $f_{\mathcal{M}}(x_1, \ldots, x_n)$ defineeritakse järgmiselt. Eeldame, et masina töö alguses on lindil

$$\mathtt{0}\underbrace{\mathtt{II}\ldots\mathtt{I}}_{x_1 \text{ kriipsu}}\ldots\mathtt{0}\underbrace{\mathtt{II}\ldots\mathtt{I}}_{x_n \text{ kriipsu}}$$

ja pea asub viimase nulli kohal. Kui masin peatub lõpliku arvu sammude järel, töö lõpus on lindil

$$\mathtt{0}\underbrace{\mathtt{II}\ldots\mathtt{I}}_{x_1 \text{ kriipsu}}\ldots\mathtt{0}\underbrace{\mathtt{II}\ldots\mathtt{I}}_{x_n \text{ kriipsu}}\mathtt{0}\underbrace{\mathtt{II}\ldots\mathtt{I}}_{y \text{ kriipsu}}$$

ja pea asub viimase nulli kohal, siis funktsiooni $f_{\mathcal{M}}(x_1, \ldots, x_n)$ väärtuseks on y. Vastasel korral on funktsiooni väärtus määramata.

Definitsioon 2. Ütleme, et funktsioon $f(x_1, \ldots, x_n)$ on Turingi mõttes arvutatav, kui leidub Turingi masin $\mathcal{M}$, et $f = f_{\mathcal{M}}$.

Turingi masina $\mathcal{M}$ poolt arvutatav funktsioon $f_{\mathcal{M}}$ on määratud argumentide $x_1, \ldots, x_n$ korral vaid siis, kui masin $\mathcal{M}$, alustades tööd standardsest algseisust, peatub lõpliku arvu sammude järel standardses lõppseisus. Standardne alg- ja lõppseis tähendavad seda, et lindil on kõik arvud kirjutatud üksteise järele ilma tühikuteta ning pea asub viimase arvu nulli kohal. Lõppseisus peavad seejuures olema alles kõik algsed argumendid ja funktsiooni väärtus peab olema kirjas nende järel.

Kui need nõuded ei ole täidetud, siis on funktsiooni $f_{\mathcal{M}}$ väärtus argumentide $x_1, \ldots, x_n$ korral määramata. Määramatus tähendab siin sisuliselt kolme võimalust: 1) masin ei peatu ja töötab lõpmatult; 2) masin peatub, kuid lindil pole vajaliku kujuga sümbolijada; 3) masin peatub ja lindil on vajaliku kujuga sümbolijada, kuid pea asub vales kohas. Eriti oluline on esimene võimalus, mida ühegi võttega vältida ei saa. Ülejäänud kahel juhul võiksime funktsiooni väärtuseks valida mingi kindla arvu, näiteks nulli.

Definitsioonis 2 eraldatakse kõigi naturaalarvuliste funktsioonide hulgast välja need, mille jaoks leidub Turingi masin, mis neid arvutab. Võib olemas olla (nagu me järgnevas näeme, ongi) funktsioone, mida ei arvuta ükski Turingi masin.

Näitame nüüd, et mõned funktsioonid on Turingi mõttes arvutatavad. Vaatleme kõigepealt konstantset funktsiooni $f(x) = 3$. Seda funktsiooni arvutava Turingi masina tabel on

	-	0	I
q_1	0q_2R	0q_1R	Iq_1R
q_2	Iq_3R		
q_3	Iq_4R		
q_4	Iq_5L		
q_5		0q_0C	Iq_5L

Programmi toimel liigub pea argumendi nulli kohalt argumendi lõppu ning kirjutab sinna nulli ja kolm kriipsu. Seejärel liigub pea tagasi viimase nulli kohale, millega tekib standardne lõppseis.

Kahe arvu liitmiseks, st. funktsiooni $f(x, y) = x + y$ väärtuse arvutamiseks, tuleb mõlema argumendi kõik kriipsud kopeerida argumentide järele. Kui argumendid on pärast töö lõppu rikutud, siis tuleb nad taastada.

	-	0	I
q_1	0q_2L	0q_1R	Iq_1R
q_2	-q_2L	0q_6L	-q_3R
q_3	-q_3R	0q_4R	
q_4	Iq_5L		Iq_4R
q_5		0q_2L	Iq_5L
q_6	-q_6L	0q_8R	-q_7R
q_7	-q_7R	0q_3R	
q_8	Iq_8R	0q_9R	
q_9	Iq_9R	0q_0C	

Ka kahe arvu vahet saab Turingi masinaga arvutada. Kuna vahe võib olla negatiivne, lindil tohivad aga olla ainult mittenegatiivsed arvud, siis vaatleme tavalise vahe asemel nn. lõigatud vahet

$$x \dot{-} y = \begin{cases} x - y, & \text{kui } x \geq y \\ 0, & \text{kui } x < y \end{cases}$$

Funktsiooni $f(x,y) = x \dot{-} y$ arvutamiseks sobib järgmine programm.

	-	0	I
q_1	$\texttt{0}q_2L$	$\texttt{0}q_1R$	$\texttt{I}q_1R$
q_2		$\texttt{0}q_3L$	$\texttt{I}q_2L$
q_3	$\texttt{-}q_3L$	$\texttt{0}q_8R$	$\texttt{-}q_4R$
q_4	$\texttt{-}q_4R$	$\texttt{0}q_5R$	
q_5		$\texttt{0}q_6R$	$\texttt{I}q_5R$
q_6	$\texttt{I}q_7L$		$\texttt{I}q_6R$
q_7		$\texttt{0}q_2L$	$\texttt{I}q_7L$
q_8	$\texttt{I}q_8R$	$\texttt{0}q_9R$	
q_9		$\texttt{0}q_{10}L$	$\texttt{I}q_9R$
q_{10}	$\texttt{-}q_{10}L$	$\texttt{0}q_{15}R$	$\texttt{-}q_{11}R$
q_{11}	$\texttt{-}q_{11}R$	$\texttt{0}q_{12}R$	
q_{12}	$\texttt{-}q_{13}L$		$\texttt{I}q_{12}R$
q_{13}		$\texttt{0}q_{16}L$	$\texttt{-}q_{14}L$
q_{14}		$\texttt{0}q_{10}L$	$\texttt{I}q_{14}L$
q_{15}	$\texttt{I}q_{15}R$	$\texttt{0}q_0C$	
q_{16}	$\texttt{I}q_{16}L$	$\texttt{0}q_{17}R$	$\texttt{I}q_{17}R$
q_{17}		$\texttt{0}q_0C$	$\texttt{I}q_{17}R$

Peale Turingi masina on olemas veel mitmeid algoritmi mõiste formalisatsioone, näiteks lambda-arvutus (A. Church), osaliselt rekursiivsed funktsioonid (S. C. Kleene), normaalalgoritmid (A. Markov). On aga osutunud, et kõik need formalisatsioonid on omavahel ekvivalentsed, st. määravad ühe ja sama funktsioonide klassi. Seepärast on algoritmiteoreetikud arvamusel, et kõik nimetatud formalisatsioonid kirjeldavadki igaüks erineval viisil algoritmi mõiste intuitiivselt tajutavat sisu. Niisugust väidet nimetatakse Churchi teesiks. Formuleerime selle Turingi masinate jaoks.

Churchi tees. Iga algoritmiliselt arvutatav funktsioon on arvutatav Turingi masina abil.

Churchi teesi ei ole võimalik tõestada, sest „algoritmiliselt arvutatav“ on mittematemaatiline mõiste ja väljendab inimeste tavaarusaama algoritmidest. Vastupidiselt on „arvutatav Turingi masina abil“ defineeritud rangelt ja tal on kindel matemaatiline sisu. Küll aga on Churchi teesi võimalik ümber lükata, näiteks juhul, kui avastatakse funktsioon, mis on ilmselgelt algoritmiliselt arvutatav, kuid mida ei saa arvutada ühegi Turingi masinaga. Vaatamata pikaajalisele uurimistööle algoritmiteoorias, pole niisugust funktsiooni leitud.

4.3. Turingi masinate numeratsioon

Järgnevas seame endale eesmärgiks näidata, et mitmed programmeerimises huvipakkuvad ülesanded ei ole Turingi masinaga lahendatavad. Churchi teesi põhjal pole siis olemas üldse mingit algoritmi nende ülesannete üldkujuliseks lahendamiseks. Esimese sammuna omistame igale Turingi masinale teataval viisil unikaalse numbri, mida Austria loogiku K. Gödeli järgi nimetatakse Gödeli numbriks.

Vaatleme Turingi masinaid, mille lindil võivad olla sümbolid s_0, s_1, ..., s_m. Esmalt nummerdame kõik Turingi masina käsud. Käsu

$$s_a q_b \to s_c q_d K$$

Gödeli numbriks loeme arvu

$$g(s_a q_b \to s_c q_d K) = 2^a 3^b 5^c 7^d 11^k,$$

kus $k = 1, 2, 3$ vastavalt sellele, kas $K = L, R, C$.

Kui on antud number, siis saab käsu üheselt taastada. Tuleb vaid Gödeli number teguriteks lahutada ja kirjutada algarvude 2, 3, 5, 7 ja 11 astmete järgi vajalik käsk välja. Leidub ka arve, mis pole ühegi käsu Gödeli numbriks, näiteks 13. Selleks, et arv oleks mingi käsu Gödeli number, peavad 2 ja 5 astmed asuma 0 ja m vahel, 3 aste peab olema vähemalt 1, sest käsu vasakul poolel on aktiivne seisund, ning 11 aste kas 1, 2 või 3.

Kui on määratud käskude numbrid, siis saab defineerida Turingi masina numbri. Oletame, et Turingi masin $\mathcal{M}$ koosneb käskudest $C_1, \ldots, C_t$. Masina $\mathcal{M}$ Gödeli numbriks loeme arvu

$$G(\mathcal{M}) = 2^{g(C_1)} 3^{g(C_2)} \ldots p_t^{g(C_t)},$$

kus p_t on järjekorras t-s algarv. Astmealusteks on seega t järjestikust algarvu.

Masina numbri järgi saab üheselt taastada masina kõigi käskude numbrid ning viimaste abil omakorda kõik masina käsud. Kui on antud Turingi masina number, siis saab selle järgi välja kirjutada masina tabeli. Leidub arve, mis pole ühegi masina numbriks. Samuti nagu käskude puhul, võib ka siin leida tingimused, mida arv rahuldama peab, et ta oleks mingi Turingi masina Gödeli number.

Teoreetiliste küsimuste selgitamiseks on otstarbekam numeratsioon, milles ei esine tühimikke ja kus iga arv on mingi Turingi masina number. Seetõttu defineerime Gödeli numbrite alusel uue, tihendatud numeratsiooni. Olgu nimelt $\mathcal{T}_0$ vähima Gödeli numbriga Turingi masin ja iga $i = 0, 1, \ldots$ korral $\mathcal{T}_{i+1}$ vähima Gödeli numbriga Turingi masin, mis ei kuulu hulka $\{\mathcal{T}_0, \ldots, \mathcal{T}_i\}$.

Üleminekud Gödeli numbrilt uuele numbrile ja vastupidi on samuti põhimõtteliselt lihtsasti teostatavad. Kui on antud Gödeli number, siis võime kontrollida kõiki sellele eelnevaid arve ja teha kindlaks, kui mitu neist kujutab endast mingi Turingi masina numbrit. Vastupidi, kui on antud masina järjekorranumber i, siis võime naturaalarve järjest läbi vaadates välja selgitada suuruselt i-nda Gödeli numbri.

4.4. Mittearvutatavad funktsioonid

Nüüd võime tõestada, et mõned konkreetsed funktsioonid on Turingi mõttes mittearvutatavad. Nende funktsioonide hulgas leidub ka selliseid, mis on olulised programmeerimise seisukohalt. Oletame, et Turingi masinad on nummerdatud eelmises jaotises näidatud viisil $\mathcal{T}_0$, $\mathcal{T}_1$, ...

Definitsioon 3. Turingi masinat $\mathcal{T}_x$ nimetatakse eneselerakendatavaks, kui ta peatub sisendargumendil x lõpliku arvu sammude järel.

Masina $\mathcal{T}_x$ argumendiks anname seega tema enda numbri. Kui arvutus kestab lõpmatult, siis masin $\mathcal{T}_x$ ei ole eneselerakendatav; kui arvutus lõpeb (ükskõik, kas standardses lõppseisus või mitte), siis masin on eneselerakendatav.

Teoreem 1. Ei leidu Turingi masinat $\mathcal{M}$, et

$$f_{\mathcal{M}}(x) = \begin{cases} 1, & \text{kui masin } \mathcal{T}_x \text{ on eneselerakendatav} \\ 0, & \text{kui masin } \mathcal{T}_x \text{ ei ole eneselerakendatav} \end{cases}$$

See tähendab, ei leidu üldist algoritmi, mis Turingi masina numbri järgi tunneks ära, kas masin on eneselerakendatav või mitte.

Tõestus. Oletame vastuväiteliselt, et selline masin $\mathcal{M}$ on olemas. Siis võime konstrueerida masina $\mathcal{N}$, et

$$f_{\mathcal{N}}(x) = \begin{cases} \text{määramata}, & \text{kui } f_{\mathcal{M}}(x) = 1 \\ 0, & \text{kui } f_{\mathcal{M}}(x) = 0 \end{cases}$$

Masina $\mathcal{N}$ võib saada, ühendades teineteise järele masina $\mathcal{M}$ ja masina tabeliga

	`-`	`0`	`I`
q_1	`-`q_0L	`0`q_1R	`I`q_1C

Kui masin $\mathcal{M}$ annab vastuseks `I`, siis käivitataks lõpmatu tsükkel, kui ta annab vastuseks `0`, siis lõpetatakse töö.

Uurime, kas masin $\mathcal{N}$ on eneselerakendatav või mitte. Olgu n selle masina number, st. $\mathcal{N} = \mathcal{T}_n$. Oletame, et $\mathcal{N}$ on eneselerakendatav. See tähendab, et $f_{\mathcal{M}}(n) = 1$, millest saame, et $f_{\mathcal{N}}(n)$ on määramata. Järelikult $\mathcal{N}$ ei peatu

omaenda numbril ega ole eneselerakendatav. Oletame, et $\mathcal{N}$ ei ole eneselerakendatav. See tähendab, et $f_{\mathcal{M}}(n) = 0$, millest saame, et $f_{\mathcal{N}}(n) = 0$. Järelikult $\mathcal{N}$ peatub omaenda numbril ja on eneselerakendatav. Mõlemal juhul saime vastuolu. Seega ei saa masinat $\mathcal{M}$ olemas olla. □

Tõestatud teoreemil iseenesest kuigi suurt praktilist väärtust ei ole. Küll aga saab tema abil tõestada *peatumise probleemi* mittelahenduvuse. Peatumise probleem on järgmine: leida algoritm, mis etteantud naturaalarvude x ja y puhul kontrollib, kas Turingi masin $\mathcal{T}_x$ peatub sisendil y.

Teoreem 2. Ei leidu Turingi masinat $\mathcal{M}$, et

$$f_{\mathcal{M}}(x,y) = \begin{cases} 1, & \text{kui masin } \mathcal{T}_x \text{ peatub argumendil } y \\ 0, & \text{kui masin } \mathcal{T}_x \text{ ei peatu argumendil } y \end{cases}$$

Tõestus. Oletame vastuväiteliselt, et selline masin $\mathcal{M}$ on olemas. Siis võime konstrueerida masina $\mathcal{N}$, et

$$f_{\mathcal{M}}(x) = \begin{cases} 1, & \text{kui masin } \mathcal{T}_x \text{ on eneselerakendatav} \\ 0, & \text{kui masin } \mathcal{T}_x \text{ ei ole eneselerakendatav} \end{cases}$$

Masina $\mathcal{N}$ võib saada, ühendades üksteise järele järgmised kolm masinat.

1. Masin, mis arvutab samasusfunktsiooni ehk dubleerib sisendargumendi. Selle masina tabel on

	`-`	`0`	`I`
q_1	`0`q_2L	`0`q_1R	`I`q_1R
q_2	`-`q_2L	`0`q_6R	`-`q_3R
q_3	`-`q_3R	`0`q_4R	
q_4	`I`q_5L		`I`q_4R
q_5		`0`q_2L	`I`q_5L
q_6	`I`q_6R	`0`q_0C	

2. Masin $\mathcal{M}$.

3. Masin, mis kustutab vahepealt teise argumendi ja nihutab vastuse (`0` või `I`) esimese argumendi järele. Selle masina tabel on

	`-`	`0`	`I`
q_1	`-`q_4L	`I`q_1R	`-`q_2L
q_2		`0`q_3R	`-`q_2L
q_3	`I`q_0L		
q_4		`0`q_0C	`-`q_4L

Olgu alguses lindil arv x. Kõigepealt see dubleeritakse, nii et lindile jääb xx, kusjuures pea asub teise x nulli kohal. Seejärel rakendatakse masinat $\mathcal{M}$, mis jätab lindile $xx\mathtt{0I}$, kui $\mathcal{T}_x$ peatub argumendil x või $xx\mathtt{0}$, kui $\mathcal{T}_x$ ei peatu argumendil x. Lõpuks kustutatakse x üleliigne eksemplar ja lindile jääb vastavalt kas $x\mathtt{0I}$ või $x\mathtt{0}$.

Nüüd aga saame vastuolu teoreemiga 1, mis väidab, et masinat $\mathcal{N}$ ei saa olemas olla. Järelikult ei saa olemas olla ka masinat $\mathcal{M}$. □

Tõestatud teoreemi võib interpreteerida nii, et programmeerimises leidub ülesandeid, mis ei ole lahendatavad algoritmiliselt, vaid nõuavad lahendamiseks loovust ja intuitsiooni. Niisuguseks ülesandeks on näiteks ka programmide testimine, mille käigus tuleb kontrollida, kas programm on kirjutatud korrektselt ja annab tulemuse suvalise sisendi korral. Muidugi võib koostada algoritme, mis lahendavad peatumise probleemi teatavatel erijuhtudel või mingi kindla ülesannete klassi jaoks. Ent kui ka kõik need klassid kokku võtta, kõiki ülesandeid ei ammenda nad ikkagi.