DODATNA SNOV ZA PREDMET TEORIJA RAČUNSKIH POSTOPKOV za slušatelje, ki jim manjka samo dodatna snov pri predmetu TJI (TOR I) in IA (TOR II)

B. Vilfan

27. januar 1997

Kazalo

Poglavje 1

B. Vilfan: GRAFI S KONČNO ZAKASNITVIJO

1.1 Uvod

V tem poglavju bomo na primeru grafov s končno zakasnitvijo prikazali uporabo nekaterih pojmov s področja končnih avtomatov. Pojem grafa s končno zakasnitvijo je nastal pri raziskavah *diferenčnih Booleovih enačb* kot enega izmed modelov digitalnih vezij. Prvotna reference je delo Wang [3], sledi delo Even in Meyer [1], omenjena uporaba teorije končnih avtomatov pa je opisana v Vilfan [2]. Ker pa so dela [3, 1] težje dostopna, je pričujoči sestavek zasnovan tako, da ni potrebno uporabljati virov.

Izhajamo iz sistema enačb

$$\begin{array}{rcl} dy_1 & = & F_1(x_1,\ldots,x_m,y_1,\ldots,y_n), \\ \vdots & \vdots & \vdots \\ dy_n & = & F_n(x_1,\ldots,x_m,y_1,\ldots,y_n), \\ z & = & F_{n+1}(x_1,\ldots,x_m,y_1,\ldots,y_n), \end{array} \tag{1.1}$$

kjer so $F_1,\ldots,F_{n+1}$ funkcije $\{0,1\}^{m+n} \longrightarrow \{0,1\}$. Možna interpretacija (1.1) je, da predstavlja povezavo med dvojiškimi vhodi in dvojiško zakodiranimi stanji nekega končnega avtomata, na eni strani, ter stanji istega avtomata v naslednjem časovnem trenutku, na drugi. Simbol d v povezavi dy_i ima pomen "bodoča vrednost", z pa predstavlja izhod končnega avtomata. Glede na to, da z ne igra bistvene vloge v naših razpravah, bomo odslej enačbo za z kar izpustili.

Sistem (1.1) lahko prevedemo na eno samo enačbo, v kateri se pojavljajo logični operatorji, če upoštevamo naslednje identitete:

$$A = B \Longleftrightarrow A \oplus B = 0$$
$$A = 0 \textbf{ in } B = 0 \Longleftrightarrow \overline{\overline{A} \wedge \overline{B}} = 0.$$

Na ta način dobimo iz sistema (1.1) eno samo enačbo oblike

$$F(\mathbf{x},\mathbf{y},\mathbf{dy}) = 0, \tag{1.2}$$

pri $\mathbf{x} \in \{0,1\}^m$ ter $\mathbf{y}, \mathbf{dy} \in \{0,1\}^n$ in lahko zapišemo naslednjo definicijo:

1.1 Definicija Diferenčna Booleova enačba *je določena z dvema pozitivnimima celima številoma m in n ter neko funkcijo*

$$F : \{0,1\}^{m+2n} \longrightarrow \{0,1\}.$$

Naj bo dana diferenčna Booleova enačba $\langle m, n, F\rangle$ ter dve zaporedji dolžine l in $l+1$:

$$\mathbf{X} = \mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(l) \tag{1.3}$$
$$\mathbf{Y} = \mathbf{y}(1), \mathbf{y}(2), \ldots, \mathbf{y}(l+1), \tag{1.4}$$

kjer $\mathbf{x}(i) \in \{0,1\}^m$, pri $1 \le i \le l$ in $\mathbf{y}(i) \in \{0,1\}^n$, pri $1 \le i \le l+1$. Potem predstavlja par $\langle \mathbf{X}, \mathbf{Y}\rangle$ *posamezno rešitev (dolžine l)* v primeru, ko velja

$$F(\mathbf{x}(i), \mathbf{y}(i), \mathbf{y}(i+1)) = 0, \text{ pri } 1 \le i \le l.$$

Za enačbo $\langle m, n, F\rangle$ pravimo, da je *rešljiva* v primeru, ko za vse $\mathbf{X}$, oblike (1.3), pri poljubnem l, eksistira $\mathbf{Y}$ oblike (1.4) tako, da je par $\langle \mathbf{X}, \mathbf{Y}\rangle$ rešitev enačbe $\langle m, n, F\rangle$.

Rešljivost Booleovih enačb ima učinkovito interpretacijo z usmerjenimi grafi, ki imajo zaznamovane povezave, kar je enakovredno nedeterminističnim končnim polavtomatom[1]. Pri podani diferenčni Booleovi enačbi $\langle m, n, F\rangle$ definiramo nedeterministični končni polavtomat

$$G(\langle m, n, F\rangle) = \langle Q, \Sigma, \delta\rangle,$$

kjer je $Q = \{0,1\}^n$, $\Sigma = \{0,1\}^m$, in

$$\delta(\mathbf{y}, \mathbf{x}) = \{\mathbf{z} \mid F(\mathbf{x}, \mathbf{y}, \mathbf{z}) = 0\}.$$

Sedaj lahko zapišemo naslednjo trditev:

1.2 Lema $\langle \mathbf{X}, \mathbf{Y}\rangle$ *predstavlja rešitev $\langle m, n, F\rangle$ natanko, ko v $G(\langle m, n, F\rangle)$ obstaja pot*

$$\mathbf{y}(1) \xrightarrow{\mathbf{x}(1)} \mathbf{y}(2) \xrightarrow{\mathbf{x}(2)} \ldots \xrightarrow{\mathbf{x}(l)} \mathbf{y}(l+1).$$

Dokaz je očiten.

Če pri polavtompatu $G(\langle m, n, F\rangle)$ korespondenco δ na običajen način spremenimo v funkcijo $\wp(Q) \times \Sigma^* \longrightarrow \wp(Q)$, pri čemer $\wp(Q)$ predstavlja potenčno množico Q, ima lema 1.2 ima preprosto posledico:

1.3 Posledica $\langle m, n, F\rangle$ *je* rešljiva *natanko, ko*

$$\forall \mathbf{x} \in \Sigma^* \left[\delta(Q, \mathbf{x}) \neq \phi\right],$$

ali z drugimi besedami, avtomat, ki ga dobimo tako, da polavtomatu kot začetna stanja in končna stanja določimo vsa stanja, sprejema Σ^*.

[1] Spomnimo se: polavtomat je avtomat, ki nima določenih začetnega stanja in končnih stanj; torej je določen le z množico stanj, z vhodno abecedo in s funkcijo, oziroma korespondenco prehodov.

	0	1
A	C	D
B	D	C
C	A, B	ϕ
D	ϕ	C, D

(a)

	0	1
A, B, C, D	A, B, C, D	C, D
C, D	A, B	C, D
A, B	C, D	C, D

(b)

Slika 1.1 Končni polavtomat (a) in dosegljivost stanj (b).

1.2 Zakasnitev grafov oziroma polavtomatov

Odslej lahko pojem rešljivosti raziskujemo na končnih nedeterminističnih polavtomatih na podlagi leme 1.2 ne glede na prvotno motivacijo za to početje (diferenčne Booleove enačbe ter zasnova digitalnih končnih avtomatov). Torej

1.4 Definicija *Če so podani polavtomat* $G = \langle Q, \Sigma, \delta \rangle$, $x = \langle a_1, a_2, \ldots, a_l \rangle$, *pri* $a_i \in \Sigma$, *za* $1 \leq i \leq l$, *in* $\underline{q} = \langle q_1, q_2, \ldots, q_{l+1} \rangle$, *pri* $q_i \in Q$ *za* $1 \leq i \leq l+1$, *predstavlja par* $\langle x, \underline{q} \rangle$ posamezno rešitev *za* G *v primeru, ko* $q_{i+1} \in \delta(q_1, a_1 a_2 \ldots a_i)$, *pri* $1 \leq i \leq l$. G *je* rešljiv *v primeru, ko za vse* x *obstaja posamezna rešitev* $\langle x, \underline{q} \rangle$, *ali z drugimi besedami ko za vse* $x \in \Sigma^*, \delta(Q, x) \neq \phi$.

1.5 Pripomba Sedaj za vhodne simbole polavtomata G kakor tudi za njegova stanja ne uporabljamo več krepkega tiska. □

1.6 Primer Vzemimo kot primer polavtomat, čigar prehodi so definirani s tabelo na sl. 1.1a. Na sl. 1.1b je prikazana tabela prehodov ustreznega determinističnega avtomata za stanja, ki so dosegljiva iz začetnega stanja, ki vsebuje vsa stanja prvotnega nedeterminističnega avtomata. Ker na podlagi sl. 1.1b ne moremo priti do prazne množice, sklepamo, da je avtomat s sl. 1.1a rešljiv.

Sedaj se vprašajmo, kako določiti neko zaporedje stanj, ki ustreza konkretni vhodni besedi x? V tem primeru postopamo tako, da najprej poiščemo množico $S = \delta(Q, x)$ in nato za vsak element $s \in S$ poiščemo $R_s = \delta^R(\{s\}, x^R)$, pri čemer je x^R zrcalna slika besede x, δ^R pa inverzna korespondenca korespondence δ. Z drugimi besedami,

$$r \in \delta^R(q, a) \Longleftrightarrow q \in \delta(r, a).$$

Kot začetno stanje q, ki ustreza besedi x, izberemo poljubno stanje iz množice R_s (če velja $R_s \neq \phi$). Naslednje stanje je poljubno stanje iz $\delta(\{q\}, a) \cap \delta^R(\{s\}, y)$, če je $x^R = ya$, nadaljujemo pa na očiten način.

1.7 Definicija *Za* $G = \langle Q, \Sigma, \delta \rangle$ *pravimo, da je* rešljiv s končno zakasnitvijo d *v primeru, ko*

$$\begin{aligned} &\forall x \in \Sigma^d \exists q \in Q \forall y \in \Sigma^* \exists \underline{q} \in Q^{|xy|} \\ &\Big[\langle xy, q\underline{q} \rangle \;\; \textit{predstavlja posamezno rešitev za } G\Big]. \end{aligned} \tag{1.5}$$

	0	1
A	A, B	ϕ
B	ϕ	A, B

(a)

	0	1	2
A	A, B	ϕ	A
B	ϕ	A, B	B

(b)

Slika 1.2 Polavtomat s končno zakasnitvijo (a) in brez (b).

(Z drugimi besedami G je rešljiv s končno zakasnitvijo d, ko za izbiro prvega stanja v zaporedju, ki ustreza w zadostuje, da poznamo prvih d črk w.) Bodimo pozorni na to, da definicija 1.7 ne pomeni, da za besede, ki so krajše od d ni rešitev: take besede je preprosto potrebno na poljuben način podaljšati in nato pri iskanju rešitve zanemariti podaljšek.

1.8 Primer Polavtomat na sl. 1.2a je rešljiv z zakasnitvijo $d = 1$ medtem, ko oni na sl. 1.2b ni rešljiv s končno zakasnitvijo (vaja). □

Vhod:

Končni polavtomat $G = \langle Q, \Sigma, \delta \rangle$ ter neko pozitivno celo število d.

Izhod:

Sodba, ali ima G zakasnitev d.

Postopek:

1. Za vse možne pare $\langle q, ax \rangle$, pri $q \in Q$, $a \in \Sigma$ ter $x \in \Sigma^{d-1}$ poiščemo za vse $b \in \Sigma$ množice
$$S(q, ax, b) = \{\langle r, xb \rangle \mid r \in \delta(q, a)\}.$$
Če je pri nekem b $S(q, ax, b)$ prazno, postane par $\langle q, ax \rangle$ "prepovedan".
2. Množico prepovedanih parov povečamo za tiste pare $\langle q, ax \rangle$, za katere obstaja vsaj en $b \in \Sigma$ tako, da vsebuje $S(q, ax, b)$ same že prepovedane pare. Če se je tako množica prepovedanih parov povečala, korak ponovimo, sicer nadaljujemo z naslednjim korakom.
3. G ima zakasnitev d natanko v primeru, ko za vsak $x \in \Sigma^d$ obstaja vsaj en neprepovedan par $\langle q, x \rangle$.

Slika 1.3 Odločitveni algoritem za zakasnitev d.

Sedaj se bomo lotili problema, kako ugotoviti, ali ima neki graf končno zakasnitev, z drugimi besedami, zakasnitev d, pri nekem končnem d. Za določeno, podano vrednost d lahko ugotovimo, ali ima $G = \langle Q, \Sigma, \delta \rangle$ zakasnitev d na podlagi definicije 1.7. Algoritem, na podlagi katerega se lahko odločamo o tem je prikazan na sl. 1.3. Dokaz njegove pravilnosti prepuščamo bralcu za vajo.

	0	1	2
A	A, B, C	ϕ	B, C
B	ϕ	ϕ	A, C
C	ϕ	A, B, C	A, B

(a)

	0	1	2
A_0	A_0, B_0, C_0	A_1, B_1, C_1	A_2, B_2, C_2
A_1	ϕ	ϕ	ϕ
A_2	B_0, C_0	B_1, C_1	B_2, C_2
B_0	ϕ	ϕ	ϕ
B_1	ϕ	ϕ	ϕ
B_2	A_0, C_0	A_1, C_1	A_2, C_2
C_0	ϕ	ϕ	ϕ
C_1	A_0, B_0, C_0	A_1, B_1, C_1	A_2, B_2, C_2
C_2	A_0, B_0	A_1, B_1	A_2, B_2

(b)

Pripomba: oznaka q_a predstavlja par $\langle q, ax\rangle$.

Slika 1.4 Polavtomat iz tabele (a) nima zakasnitve ena.

1.9 Primer Na sl. 1.4a vidimo primer končnega polavtomata, za katerega želimo ugotoviti, ali ima zakasnitev 1. Tabela na sl. 1.4b prikazuje množice $S(q, ax, b)$ iz algoritma na sl. 1.3. Pri izvajanju koraka 1, postanejo prepovedani pari A_1, B_0, B_1, C_0, Pri prvem izvajanju koraka 2, moramo dodatno prepovedati A_2, C_2; pri drugem izvajanju, prepovemo B_2; in končno, pri tretjem še A_0 in C_1. Sklepamo torej, da G s sl. 1.4a nima zakasnitve 1. □

Kako pa ugotovimo, ali ima neki graf $G = \langle Q, \Sigma, \delta\rangle$ *končno* zakasnitev? Osnovna ideja je, da poiščemo nekakšno zgornjo mejo D za zakasnitev (torej, če ima graf končno zakasnitev, je njegova zakasnitev $\leq D$). Potem pa lahko uporabimo algoritem s sl. 1.3 za $1, 2, \ldots, D$.

Do omenjene zgornje meje pa pridemo na naslednji način. Če je podan $G = \langle Q, \Sigma, \delta\rangle$, najprej definiramo $G^k = \langle Q, \Sigma^k, \delta^k\rangle$, pri čemer je δ^k preprosto k-kratni kompozitum osnovne korespondence δ. Sedaj lahko trdimo

1.10 Lema *$G = \langle Q, \Sigma, \delta\rangle$ ima končno zakasnitev natanko, ko eksistira k, pri katerem ima G^k zakasnitev 1.*

Dokaz. Naj ima G zakasnitev d. Potem je očitno, da ima G^d zakasnitev 1. Po drugi strani pa naj ima G^k zakasnitev 1. To pomeni, da lahko na podlagi prvih k črk neke besede izberemo začetno stanje q. Vendar, ker vhodno besedo sedaj beremo v blokih dolžine k, moramo za naslednje stanje prebrati naslednji blok, kar pomeni, da moramo za drugo stanje poznati naslednjih $2k - 1$ črk. To pa je tudi najhujši primer in tako pridemo do sklepa, da ima G zakasnitev $d \leq 2k - 1$. □

Če je podan $G = \langle Q, \Sigma, \delta\rangle$, bomo z G^Σ označili *množico funkcij* $f_a : Q \longrightarrow \wp(Q)$, pri $a \in \Sigma$ in $f_a(q) = \delta(q, a)$. Sedaj definiramo naslednjo relacijo podobnosti na neki množici polavtomatov: $G = \langle Q, \Sigma, \delta\rangle$ in $F = \langle R, \Delta, \eta\rangle$ sta si *podobna* (simbolično: $G \sim F$) v primeru, ko (1) $Q = R$ in (2) $G^\Sigma = F^\Delta$. Torej dva polavtomata sta si podobna, če imata isto množico stanj in če obe abecedi (ki ni nujno, da sta enaki) definirata isto množico preslikav med stanji in podmnožicami stanj.

Sedaj pri podanem $G = \langle Q, \Sigma, \delta \rangle$, opazujemo zaporedje polavtomatov

$$G^1 = G, G^2, \ldots, G^k, \ldots$$

in zapišemo naslednje trditve:

1.11 Lema *Če velja $G^i \sim G^j$, velja tudi $G^{i+1} \sim G^{j+1}$.*

Dokaz. Očiten.

1.12 Lema *Če velja $G^i \sim G^j$ in če G^i nima zakasnitve 1, jo nima tudi G^j.*

Dokaz. Če je v algoritmu s sl. 1.3 na koncu par $\langle q, a \rangle$ prepovedan, bodo v primeru $f_a = f_b$ prepovedani tudi vsi pari $\langle q, b \rangle$. □

Torej si lahko zamislimo naslednji algoritem za ugotavljanje končne zakasnitve: preverjamo zakasnitev 1 za zaporedje $G^1 = G, G^2, \ldots, G^N$, kjer je N število nepodobnih polavtomatov z množico stanj Q_G. Pri G^N lahko prenehamo, kajti na podlagi lem 1.11 in 1.12, če dotlej nismo našli G^k z zakasnitvijo 1, ga ne bomo našli tudi odtlej.

Sedaj preostane prešteti število nepodobnih polavtomatov z n stanji. Število funkcij $Q \longrightarrow \wp(Q)$ je $(2^n)^n = 2^{n^2}$. Neki podobnostni razred relacije $\sim$ je določen s podmnožico funkcij $Q \longrightarrow \wp(Q)$. Torej je podobnostnih razredov (oziroma nepodobnih polavtomatov) $2^{2^{n^2}}$. To zapišemo v obliki naslednjega izreka:

1.13 Izrek *Če ima $G = \langle Q, \Sigma, \delta \rangle$ končno zakasnitev d, velja $d \leq 2(2^{2^{n^2}}) - 1$.*

1.3 Izboljšana ocena zakasnitve

Sedaj bomo uporabili teorijo končnih avtomatov za izboljšavo ocene zakasnitve, ki je vsebovana v izreku 1.13. Naj ima $G = \langle Q, \Sigma, \delta \rangle$ končno zakasnitev d. Na podlagi definicije 1.7 velja, da je (1) vsakemu stanju $q \in Q$ prirejena množica $S(q) \subset \Sigma^d$, ki predstavlja začetke besed, katerih pot lahko začnemo pri q, neodvisno od tega, kako besedo kasneje podaljšujemo in (2) $\bigcup_{q \in Q} S(q) = \Sigma^d$. Naj ponovimo nekaj, kar smo že poudarili: tudi za krajše besede obstaja pot skozi graf; velja le, da krajše besede ne moremo poljubno podaljševati.

Sedaj vzemimo množice $T(q) = S(q)\Sigma^*$. Množica $T(q)$ vsebuje tiste besede, za katere lahko začnemo pot pri q. Očitno sedaj velja

$$\bigcup_{q \in Q} T(q) = \Sigma^d \cdot \Sigma^*, \tag{1.6}$$

ali drugače povedano,

$$\overline{\bigcup_{q \in Q} T(q)} \text{ je končno.} \tag{1.7}$$

Na tem mestu je bistvena ugotovitev, da lahko $T(q)$ *zapremo* glede na (Myhill-Nerodeovo) desno-invariantno ekvivalenčno relacijo $R_{G(q)}$, ki jo določa G, z začetnim stanjem q. (Spomnimo se: $xR_{G(q)}y \Longleftrightarrow \delta(q,x) = \delta(q,y)$) Namreč, če velja $x \in T(q)$ ter $xR_{G(q)}y$, lahko tudi y priključimo k $T(q)$, kajti če obstaja pot skozi graf za x, ki se pričenja pri q, obstaja (ista) pot tudi za y. Množice $T(q)$, ki smo jih na ta način morebiti povečali, označimo z $U(q)$.

Ni se težko prepričati tudi o nasprotnem: če obstajajo množice $U(q)$, ki so zaprte glede na omenjeno (Myhill-Nerodeovo) desno-invariantno ekvivalenčno relacijo $R_{G(q)}$ in če je $\overline{\bigcup_{q\in Q} U(q)}$ končno, ima G končno zakasnitev.

Torej dosedanje razmišljanje lahko strnemo v naslednjo lemo:

1.14 Lema *Graf $G = \langle Q, \Sigma, \delta\rangle$ ima končno zakasnitev natanko, ko obstajajo množice $U(q)$, ki so zaprte glede Mihill-Nerodeove desno-invariantne ekvivalenčne relacije, ki jo določa G z začetnim stanjem q, in imajo še dodatno lastnost*

$$\overline{\bigcup_{q\in Q} U(q)} \text{ je končno.} \tag{1.8}$$

Sedaj ugotovimo, da je, prvič, $U(q)$ regularna množica (na podlagi konstrukcije) in, drugič, da ima končni (deterministični) avtomat, ki sprejema $U(q) \leq 2^n$ stanj, če ima G n stanj. Koliko stanj pa ima končni avtomat, ki sprejema množico $\bigcup_{q\in Q} U(q)$ oziroma njen komplement? Za unijo regularnih množic obstaja standardna konstrukcija, ki iz avtomatov za komponente unije, pridobi avtomat za unijo, katerega množica stanj je kartezični produkt množic stanj avtomatov za komponente. Isti avtomat lahko z zamenjavo končnih in nekončnih stanj uporabimo za komplement te množice. Na ta način ugotovimo, da za razpoznavanje komplementa množice $\bigcup_{q\in Q} U(q)$ obstaja avtomat, ki ima $\leq 2^{n^2}$ stanj.

Na tem mestu se zopet sklicujemo na znan rezultat teorije končnih avtomatov, ki pravi: če je jezik L končen in če ga sprejema avtomat z n stanji, potem noben element L po dolžini ne dosega n. Torej če ima število d lastnost, da množica $\bigcup_{q\in Q} U(q)$ vsebuje vse besede ki so dolge d ali več (d je potemtakem zakasnitev grafa) in torej velja (1.8), je na podlagi pravkar povedanega $d \leq 2^{n^2} - 1$. S tem smo dokazali izrek

1.15 Izrek *Če ima graf $G = \langle Q, \Sigma, \delta\rangle$ končno zakasnitev, je le-ta manjša ali enaka $2^{n^2} - 1$.*

Izrek 1.15 nam bistveno skrajša postopek ugotavljanja, ali ima podani graf končno zakasnitev (na logaritem vrednosti, ki sledi iz izreka 1.13).

Poglavje 2

Parcialne in splošno rekurzivne funkcije (zapisal A. Karalič)

2.1 Parcialne rekurzivne funkcije

Definicija: *Parcialne rekurzivne funkcije* (PRF) ali kar *Turingove funkcije* (TF) so vse funkcije, ki jih lahko izračuna turingov stroj na naslednji način: Za model stroja vzamemo enotračni stroj s trakom, ki je neskončen v desno (knjiga, stran 148), pri čemer argumente predstavljamo v unarnem zapisu; torej pri izračunu $f(x_1, x_2, ..., x_k)$ ponazarjamo argumente v obliki $0^{x_1}10^{x_2}1\ldots10^{x_k}$. Ko se stroj ustavi, mora na traku biti zapisana beseda $0^{f(x_1,x_2,...,x_k)}$. Turingove funkcije s k argumenti označujemo s $\mathcal{T}_k$, Turingove funkcije nasploh pa s $\mathcal{T} = \bigcup_{k=1}^{\infty} \mathcal{T}_k$. $\square$

Iz predavanj vemo, da za računanje $f \in \mathcal{T}_k$ lahko izberemo neki stroj z notranjo abecedo $\{0, 1, B\}$.

2.2 Splošno rekurzivne funkcije

Definicija: *Splošno rekurzivna funkcija* (SRF) ali kar *Rekurzivna funkcija* (RF) je vsaka funkcija $\mathcal{N}^k \to \mathcal{N}$, ki jo lahko dobimo iz začetnih funkcij:

1. *Ničelna funkcija*: $(\forall x)(Z(x) = 0)$,
2. *Naslednik*: $(\forall x)(N(x) = x + 1)$,
3. *Projekcija*: $(\forall n)(\forall i) : [(n \geq 1) \wedge (1 \leq i \leq n) \Rightarrow (U_i^n(x_1, x_2, \ldots, x_n) = x_i)]$.

s končnim številom substitucij, primitivnih rekurzij in minimalizacij. Rekurzivne funkcije s k argumenti označujemo z $\mathcal{R}_k$, vse rekurzivne funkcije pa z $\mathcal{R} = \bigcup_{k=1}^{\infty} \mathcal{R}_k$. $\square$

Definicija: (*Substitucija* oziroma *funkcijska kompozicija*) Če iz že pridobljenih SRF $g : \mathcal{N}^m \to \mathcal{N}$ in $h_i : \mathcal{N}^n \to \mathcal{N}$ pri $1 \leq i \leq m$ konstruiramo novo SRF f:

$$f(x_1, \ldots, x_n) = g(h_1(x_1, \ldots, x_n), \ldots, h_m(x_1, \ldots, x_n))$$

pravimo, da smo jo pridobili s substitucijo iz g in $h_1, \ldots, h_m$. □

Definicija (*Primitivna rekurzija*) Podani sta dve SRF $g : \mathcal{N}^n \to \mathcal{N}$ ter $h : \mathcal{N}^{n+2} \to \mathcal{N}$. Potem lahko definiramo novo SRF $f : \mathcal{N}^{n+1} \to \mathcal{N}$:

$$f(x_1, \ldots, x_n, 0) = g(x_1, \ldots, x_n)$$

$$f(x_1, \ldots, x_n, y + 1) = h(x_1, \ldots, x_n, y, f(x_1, \ldots, x_n, y))$$

Za f pravimo, da smo jo dobili iz g ter h z rekurzijo oziroma primitivno rekurzijo. □

Definicija (*Minimalizacija*) Za $f : \mathcal{N}^n \to \mathcal{N}$ pravimo, da je pridobljena z minimalizacijo iz SRF $g : \mathcal{N}^{n+1} \to \mathcal{N}$, če je definirana takole:

$$f(x_1, \ldots, x_n) = \mu_y g(x_1, \ldots, x_n, y) = \begin{cases} z & \text{če } (\exists z) : [g(\ldots, z) = \min_y (g(\ldots, y) = 0)] \\ \text{nedefinirano} & \text{sicer} \end{cases}$$

□

2.3 Ekvivalenca

Izrek 3.1 $\mathcal{T} = \mathcal{R}$. (Turingove funkcije so natanko rekurzivne funkcije.) □

Dokaz razpade na dokaza dveh trditev:

Trditev 3.1.1: $\mathcal{R} \subset \mathcal{T}$. (Vsaka rekurzivna funkcija je obenem Turingova funkcija.) □

Trditev 3.1.2: $\mathcal{T} \subset \mathcal{R}$. (Vsaka Turingova funkcija je obenem rekurzivna funkcija.) □

Na tem mestu ugotovimo korist od ekvivalence med različnimi verzijami Turingovih strojev, ki smo jih opisali med predavanji. Namreč Trditev 3.1.1 je tem lažje dokazati, kolikor imamo zmogljivejši stroj na voljo; medtem, ko je Trditev 3.1.2 tem lažje dokazati, kolikor imamo opravka s preprostejšim strojem.

Dokaz Trditve 3.1.1: Trditev dokažemo tako, da sprogramiramo vse elemente, ki nastopajo v definiciji SRF (ničelna funkcija, substitucija, primitivna rekurzija, minimalizacija) na neki verziji TS. Čeprav je naloga lahko tehnično zamudna, je pa načelno trivialna in jo prepuščamo bralcu. □

Dokaz Trditve 3.1.2 pa terja, da pri podanem stroju, ki računa funkcijo $f(x_1, \ldots, x_n)$, definiramo f kot SRF. To bomo storili tako, da bomo postopoma prišli do preprostih funkcij, za katere ne bo težko pokazati, da so SRF.

2.3.1 Kodiranje traku in stanj

Besedo ali del besede, ki je zapisana na traku, kodiramo z naravnimi števili tako, da definiramo $c : \Gamma^* \to \mathcal{N}$ na naslednji način:

1. Naj bo $|\Gamma| = b$.

2. Vsakemu simbolu $a \in \Gamma$ pripišemo neko cifro $d(a) \in \{0, 1, \ldots, b-1\}$ z omejitvijo, da presledku B pripišemo cifro 0.

3. Če je $x = a_1 a_2 \ldots a_n$, potem $c(x) = d(a_1)b^{n-1} + d(a_2)b^{n-2} + \ldots + d(a_{n-1})b + d(a_n)$ (Številčni zapis z bazo b.)

(Pripomba: iz definicije sledi, da zaporedje presledkov poljubne dolžine kodiramo z 0.)

Odslej bomo za $x \in \Gamma^*$ namesto $c(x)$ pisali $\widetilde{x}$, inverzno funkcijo $\mathcal{N} \to \Gamma^*$ pa bomo označevali z $\widetilde{\widetilde{x}}$ pri $x \in \mathcal{N}$.

Sedaj definirajmo nekaj važnih funkcij $\mathcal{N} \to \mathcal{N}$, ki se nanašajo na številčni zapis v bazi b in pokažimo da so SRF.

(a) $\text{dolžina}(x) = \mu_k[x \dot{-} b^k]$. Pri čemer velja:

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

(b) $\text{prvi}(x) = \mu_k[x \dot{-} k \cdot b^{\text{dolžina}(x) \dot{-} 1}] \dot{-} 1$

(c) $\text{zadnji}(x) = x \bmod b$

(d) $\text{rep}(x) = x \dot{-} \text{prvi}(x) \cdot b^{\text{dolžina}(x) \dot{-} 1}$

(e) $x \circ y = b^{\text{dolžina}(y)} x + y$ (Konkatenacija)

Pokažimo npr., da je $f(x, y) = x + y$ SRF:

$$f(x, 0) = g(x) = U_1^1(x) = x$$

$$f(x, y+1) = N(x + y)$$

Kaj pa $x \cdot y$?

$$x \cdot 0 = Z(x)$$

$$x \cdot (y+1) = x \cdot y + x$$

Sedaj pa $x \dot{-} y$: najprej definirajmo $\delta(x) = x \dot{-} 1$

$$\delta(0) = 0$$

$$\delta(x+1) = x$$

$$x \dot{-} 0 = x$$

$$x \dot{-} (y+1) = \delta(x \dot{-} y)$$

Za $x \bmod y$:

$$x \bmod y = x \dot{-} (\mu_z[x \dot{-} y \cdot z] \dot{-} 1) \cdot y$$

Dokaz, da je x^y SRF:

$$x^0 = N(Z(x))$$

$$x^{y+1} = x^y \cdot x$$

Iz tega sklepamo, da so tudi 'dolžina', 'prvi', 'zadnji', 'rep' in '$\circ$' SRF. Pri dokazih, da so zapisane funkcije SRF smo uporabljali minimalizacijo. Te funkcije se dajo definirati tudi brez minimalizacije, vendar je strogi dokaz nekoliko zamuden.

Končno lahko tudi stanja zakodiramo s števili iz nekega intervala $[0, |Q|-1]$ in tudi tukaj bomo uporabljali simbola $\tilde{q}$ ter $\tilde{\tilde{q}}$. (Na ta način je $\tilde{\tilde{q}}$ dvoumno, kajti v tem primeru je $q \in \mathcal{N}$ in ni jasno ali uporabljamo inverzno funkcijo za c, ki je bila prej definirana ali pa inverzno funkcijo od pravkar omenjene $Q \to \mathcal{N}$. Vendar bo iz konteksta jasno za kaj gre.)

2.3.2 Dokaz Trditve 3.1.2

Naj bo $M = \langle Q, \Sigma, \Gamma, \delta, q_0, B, F\rangle$ Turingov stroj, ki računa $f \in \mathcal{T}_k$.

Definirajmo sedaj funkciji T_1 in T_2:

$$T_1(x,q,y,n) = \begin{cases} 0 & \text{pri } \tilde{\tilde{x}}\tilde{\tilde{q}}\tilde{\tilde{y}} \overset{n}{\vdash}_M \tilde{\tilde{x}}'\tilde{\tilde{q}}'\tilde{\tilde{y}}' \text{ in } \tilde{\tilde{q}}' \in F \\ 1 & \text{sicer} \end{cases}$$

Torej nam T_1 z vrednostjo 0 signalizira končno stanje čez n korakov. Druga funkcija je:

$$T_2(x,q,y,n) = \begin{cases} z & \text{pri } \tilde{\tilde{x}}\tilde{\tilde{q}}\tilde{\tilde{y}} \overset{n}{\vdash}_M \tilde{\tilde{x}}'\tilde{\tilde{q}}'\tilde{\tilde{y}}' \text{ in } \tilde{\tilde{x}}'\tilde{\tilde{y}}' = 0^z \\ \uparrow & \text{sicer} \end{cases}$$

kjer $\uparrow$ pomeni "nedefinirano". T_2 torej izračuna vrednost (po n korakih) zapisano v unarnem zapisu. Potem lahko funkcijo, ki jo računa stroj, zapišemo kot:

$$f(x_1,\ldots,x_k) = T_2(0, \tilde{q}_0, \widetilde{0^{x_1}1\ldots10^{x_k}1}, \mu_n T_1(0,\tilde{q}_0, \widetilde{0^{x_1}1\ldots10^{x_k}1}, n))$$

Definirajmo na tem mestu funkciji e in d:

$$e(x,y) = \begin{cases} 0 & \text{pri } x = y \\ 1 & \text{sicer} \end{cases}$$

$$d(x,y) = 1 \dot{-} e(x,y)$$

Za T_1 dobimo izraz

$$T_1(x,q,y,n) = \prod_{\tilde{p}\in F} e(p, T_3(x,q,y,n))$$

T_3 pa izračuna stanje po n korakih:

$$T_3(x,q,y,n) = \begin{cases} q & \text{pri } n = 0 \\ T_3(N_l(x,q,y), N_Q(x,q,y), N_d(x,q,y), n-1) & \text{sicer} \end{cases}$$

N_l, N_Q, N_d nam računajo levi del, stanje in desni del v trenutnem opisu, ki neposredno sledi za $\tilde{\tilde{x}}\tilde{\tilde{q}}\tilde{\tilde{y}}$.

$$N_l(x,q,y) = \sum_{\substack{0\le p\le |Q|-1\\ 0\le a\le b-1}} d(p,q)d(a,\mathrm{prvi}(y))N_{l,p,a}(x,q,y)$$

kjer velja

$$N_{l,p,a}(x,q,y) = \begin{cases} x \operatorname{div} b & \text{pri } \delta(p,a) = (r,s,L) \\ x \circ \tilde{s} & \text{pri } \delta(p,a) = (r,s,R) \end{cases}$$

$$N_Q(x,q,y) = \sum_{\substack{0\le p\le |Q|-1\\ 0\le a\le b-1}} d(p,q)d(a,\mathrm{prvi}(y))N_{Q,p,a}(x,q,y)$$

$$N_{Q,p,a} = \tilde{r} \text{ pri } \delta(p,a) = (r,s,d)$$

$$N_d(x,q,y) = \sum_{\substack{0\le p\le |Q|-1\\ 0\le a\le b-1}} d(p,q)d(a,\mathrm{prvi}(y))N_{d,p,a}(x,q,y)$$

$$N_{d,p,y}(x,q,y) = \begin{cases} \mathrm{zadnji}(x) \circ \tilde{s} \circ \mathrm{rep}(y) & \text{pri } \delta(p,a) = (r,s,L) \\ \mathrm{rep}(y) & \text{pri } \delta(p,a) = (r,s,R) \end{cases}$$

Sedaj zapišimo T_2:

$$T_2(x,q,y,n) = \mathrm{dolžina}(\mathrm{Fl}(\mathrm{Trak}(x,q,y,n)))$$

$$\mathrm{Trak}(x,q,y,n) = \begin{cases} x \circ y & \text{pri } n = 0 \\ \mathrm{Trak}(N_l(x,q,y), N_Q(x,q,y), N_d(x,q,y), n-1) & \text{sicer} \end{cases}$$

Funkcija 'Fl' ugotavlja, če je trak pravilno oblikovan.

$$\mathrm{Fl}(x) = \begin{cases} x & \text{pri } \tilde{\tilde{x}} = 0 \ (0 \in \Gamma) \\ \uparrow & \text{pri } 0 \le x \le b-1 \text{ in } \tilde{\tilde{x}} \ne 0 \\ \mathrm{Fl}(\mathrm{prvi}(x)) \circ \mathrm{Fl}(\mathrm{rep}(x)) & \text{sicer} \end{cases}$$

Sedaj pa pokažimo, da so nekatere funkcije, ki jih uporabljamo SRF.

$$e(x,y) = 1 \dot{-} (1 \dot{-} ((x \dot{-} y) + (y \dot{-} x)))$$

$$d(x,y) = 1 \dot{-} e(x,y)$$

$$\uparrow = \mu_n[N(Z(n))]$$

Na ta način smo vse funkcije izrazili kot SRF in je Trditev 3.2.1 dokazana. □

S tem pa smo tudi končali dokaz Izreka 3.1. □

2.4 Vse Splošno rekurzivne funkcije niso Primitivno rekurzivne

Definicija: *Primitivno rekurzivne funkcije* (PrRF) so tiste SRF, ki so pridobljene iz začetnih funkcij brez uporabe minimalizacije. □

Naj P_i predstavlja podrazred vseh primitivno rekurzivnih funkcij, ki jih lahko pridobimo z največ i substitucijami ali primitivnimi rekurzijami iz začetnih funkcij. P_0 vsebuje le začetne funkcije.

Sedaj bomo definirali funkcijo $A(x,i)$ z naslednjima lastnostima:

$$A(x,i) \quad \text{monotono narašča po obeh argumentih} \tag{2.1}$$

$$A(x,i) > f(x_1,\ldots,x_n) \Leftarrow x \geq \max(x_1,\ldots,x_n) \wedge f \in P_i \tag{2.2}$$

Recimo, da je $A(x,k)$ pri $k \leq i$ že definirana. Naj bo $f \in P_{i+1}$. Potem je bilo zadnje pravilo v definiciji f bodisi substitucija bodisi primitivna rekurzija.

Če je bilo zadnje pravilo substitucija, imamo:

$$f = g(h_1(x_1,\ldots,x_n),\ldots,h_m(x_1,\ldots,x_n)) \;\text{ pri }\; g,h_1,\ldots,h_m \in P_i$$

in torej (ker $g \in P_i$):

$$A(x,i) > f(x_1,\ldots,x_n) \quad \Leftarrow \quad x \geq \max(h_1(x_1,\ldots,x_n),\ldots,h_m(x_1,\ldots,x_n))$$

Iz lastnosti (2.1) in (2.2) sledi, da $A(x,i) > x$ in

$$A(x,i) > h_i(x_1,\ldots,x_n) \quad \Leftarrow \quad x \geq \max(x_1,\ldots,x_n)$$

Torej

$$A(A(x,i),i) > f(x_1,\ldots,x_n) \;\text{ pri }\; x \geq \max(x_1,\ldots,x_n) \tag{2.3}$$

V primeru, ko je bilo zadnje pravilo primitivna rekurzija, imamo

$$f(x_1,\ldots,x_n,0) = g(x_1,\ldots,x_n)$$

$$f(x_1,\ldots,x_n,y+1) = h(x_1,\ldots,x_n,y,f(x_1,\ldots,x_n,y))$$

pri $g,h \in P_i$. Velja torej

$$A(x,i) > f(x_1,\ldots,x_n,0) \quad \Leftarrow \quad x \geq \max(x_1,\ldots,x_n) \tag{2.4}$$

$$A(x,i) > f(x_1,\ldots,x_n,y) \quad \Leftarrow \quad x \geq \max(x_1,\ldots,x_n,y-1,f(x_1,\ldots,x_n,y-1)) \tag{2.5}$$

Torej zaradi (2.1), (2.2) tembolj velja

$$A(A(x,i),i) > f(x_1,\ldots,x_n,y) \Leftarrow x \geq \max(x_1,\ldots,x_n,y-1,f(x_1,\ldots,x_n,y-2)) \tag{2.6}$$

Z verižno uporabo relacije (2.5) dobimo

$$\underbrace{A(A(\ldots A(}_{y\,\text{krat}}x,i),\ldots,i),i) > f(x_1,\ldots,x_n,y) \quad \Leftarrow \quad x \geq \max(x_1,\ldots,x_n,y-1,0)$$

oziroma, ker $x \geq y-1$ in (2.1) tembolj

$$\underbrace{A(A(\ldots A(}_{x+2\,\text{krat}}x,i),\ldots,i),i) > f(x_1,\ldots,x_n,y) \;\Leftarrow\; x \geq \max(x_1,\ldots,x_n,y) \tag{2.7}$$

Sedaj pa je očitno, da če postavimo

$$A(x,0) = x+2 = N(N(x)) \tag{2.8}$$

(ker mora biti $A(x,0)$ večja od katerekoli začetne funkcije) in

$$A(x,i+1) = \underbrace{A(A(\ldots A(}_{x+2\,\text{krat}}x,i),\ldots,i),i) \tag{2.9}$$

izpolnjuje $A(x,i)$ zahtevane pogoje (glej (2.3) in (2.7)).

Definirajmo $\mathrm{B}(x) = A(x,x)$. Sedaj pa trdimo, da $\mathrm{B}(x)$ <u>*ni*</u> primitivno rekurzivna funkcija. Recimo, da velja nasprotno. Potem $\mathrm{B}(x) \in P_i$ pri nekem določenem i in velja:

$$A(x,i) > \mathrm{B}(y) \quad \Leftarrow \quad x \geq y$$

Sedaj pa izberemo $(x=y) \;\wedge\; (x>i)$ in dobimo:

$$A(x,i) > \mathrm{B}(x) = A(x,x) > A(x,i)$$

Prvi '>' velja zaradi (2.2) drugi '>' pa velja zaradi (2.1) in $x > i$. Dobili smo protislovje.

Po drugi strani pa se ni težko prepričati v to, da je A izračunljiva oziroma splošno rekurzivna funkcija (vaja). Funkciji 'A' pravimo *Ackermannova funkcija*. Je SRF in narašča hitreje kot katerakoli PrRF.

2.5 Dodatek, B. Vilfan

$$p(x) = \begin{cases} 0 & \text{pri } x=0 \\ 1 & \text{sicer} \end{cases}$$
$$q(x) = \begin{cases} 1 & \text{pri } x=0 \\ 0 & \text{sicer} \end{cases}$$

$$x \dot{-} y$$

Izpeljava $p(x)$:

1. $f^0 = Z^0$
2. $g^2(x, y) = N^1(Z^0)$
3. $p(x)$ Na podlagi 1 in 2 ter primitivne rekurzije.

Izpeljava $q(x)$:

1. $f^0 = N^1(Z^0)$
2. $g^2(x, y) = Z^0$
3. $p(x)$ Na podlagi 1 in 2 ter primitivne rekurzije.

Izpeljava $x + y$:

1. $f^1(x) = N_1^1(x)$
2. $g^2(x, y, z) = N^1(P_3^3(x, y, z))$
3. $x + y$ Na podlagi 1 in 2 ter primitivne rekurzije.

Izpeljava $x \dot{-} y$:

1. $f^0 = Z^0$
2. $g^2(x, y) = P_1^2(x, y)$
3. $d(x)$ Na podlagi 1 in 2 ter primitivne rekurzije.
4.

Literatura

[1] S. Even and A. R. Meyer. Sequential boolean equations. Technical report, Carnegie-Mellon University, Pittsburgh, Pa., USA, March 1968.

[2] Boštjan Vilfan. An improved upper bound for the finite delay of graphs (short note). *IEEE TRANS. ON COMPUTERS*, 19(5):453–456, May 1970.

[3] H. Wang. Circuit synthesis by solving sequential boolean equations. *Zeitschr. f. mat. Logik und Grundlagen d. Math.*, 5:291–322, 1959. also in *Studies in Logic and the Foundations of Mathematics.*, Science Press, Peking and North-Holland Publ. Co., Amsterdam: 1963, pp. 269-305.