MONADIČNA LOGIKA DRUGEGA REDA IN PREPOZNAVNI JEZIKI

ANARA NEMANIČ

Fakulteta za matematiko in fiziko
Univerza v Ljubljani

Končne avtomate se lahko obravnava kot matematične modele, ki berejo besede in odločajo o pripadnosti besed jeziku; monadično logiko drugega reda pa kot logični formalizem za izražanje lastnosti besed z uporabo relacij med pozicijami črk v besedah. Čeprav gre na prvi pogled za različna matematična pojma, ju povezuje presenetljiv rezultat, znan kot Büchijev izrek. Ta pravi, da so jeziki, ki jih prepoznavajo končni avtomati, natanko jeziki, ki se jih lahko opiše s stavki monadične logike drugega reda. V članku so predstavljeni osnovni pojmi teorije avtomatov in monadične logike drugega reda ter podan dokaz Büchijevega izreka v obeh smereh.

MONADIC SECOND ORDER LOGIC AND RECOGNISABLE LANGUAGES

Finite automata can be viewed as mathematical models that read words and decide whether they belong to a given language, while monadic second-order logic can be viewed as a logical formalism for expressing properties of words using relations between positions of letters in words. Although these notions appear to be quite different at first sight, they are connected by a remarkable result known as Büchi's theorem. This theorem states that the languages recognised by finite automata are precisely the languages that can be defined by sentences of monadic second-order logic. This article introduces the basic concepts of automata theory and monadic second-order logic and presents a proof of Büchi's theorem in both directions.

1. Uvod

V začetku 20. stoletja so se številni raziskovalci spraševali, ali je mogoče vse matematične resnice določiti „mehanično“ z uporabo matematičnih strojev, ki so kasneje postali osnova za razvoj računalnikov. Z omejitvijo teh strojev na končen pomnilnik smo dobili model končnega avtomata, katerega lastnosti so raziskovali pionirji, kot so Kleene, Rabin, Scott in drugi. V poznih 50. letih sta Büchi in Elgot odkrila globoko povezavo med končnimi avtomati in matematično logiko, kar je postalo ključno za napredek na področju računalništva.

Büchijev izrek dokazuje, da imajo jeziki, ki jih prepoznavajo končni avtomati, ter jeziki, definirani v monadični logiki drugega reda nad signaturo naslednika, enako izrazno moč, kar nam omogoča prehod med logičnimi opisi in avtomati. S tem, ko lahko logično formulo prevedemo v končni avtomat, postanejo lastnosti, izražene z logiko, dostopne algoritmični obdelavi. Ta povezava predstavlja enega izmed temeljnih mostov med matematično logiko ter teorijo računalništva in je imela pomemben vpliv na razvoj teorije formalnih jezikov, programskih jezikov in analize algoritmov.

Namen članka je predstaviti povezavo med avtomati in logiko drugega reda ter podati podroben dokaz Büchijevega izreka. V razdelku 2. bomo uvedli osnovne pojme teorije formalnih jezikov, nato bomo v razdelku 3. obravnavali končne avtomate, in v razdelku 4. predstavili monadično logiko drugega reda. S tem bomo pripravljeni, da v razdelku 5. dokažemo Büchijev izrek. Temeljni rezultati in pojmi, obravnavani v tem članku, sledijo predvsem viru [1], povsem podrobno pa so obravnavani tudi v diplomskem delu [2].

2. Besede in jeziki

V tem razdelku predstavimo osnovne pojme iz teorije formalnih jezikov po [1, poglavje III: 1 in 2] in [3, razdelek 1.2–4].

2.1 Besede

Uvedimo najprej osnovne pojme o besedah. Naj bo $A = \{a, b, \ldots\}$ množica, imenovana *abeceda*, katere elementi se imenujejo *črke*. Končno zaporedje elementov A imenujemo *(končna) beseda* nad abecedo A. Besedo $u = (a_0, a_1, \ldots, a_n);\ a_i \in A$ označujemo z

$$u = a_0 a_1 \cdots a_n.$$

Množica besed je opremljena z operacijo *stika*, ki besedama $u = a_0 a_1 \cdots a_p$ in $v = b_0 b_1 \cdots b_q$ priredi besedo $uv = a_0 a_1 \cdots a_p b_0 b_1 \cdots b_q$. Operacija stik je asociativna in ima tudi identiteto, *prazno besedo*, označeno z ε, ki je prazno zaporedje. Z A^+ označimo množico nepraznih besed nad A in z A^* množico vseh besed.

2.2 Jeziki

Naj bo A končna abeceda. Podmnožice A^* imenujemo *jeziki*. Na primer, če je $A = \{a, b\}$, so množice $\emptyset, \{\varepsilon\}, \{aba, babaa, bb\}, \{a^n b a^n \mid n \geq 0\}$ in A^* jeziki. Jezik $\{u\}$, ki vsebuje natanko eno besedo u, označimo kar z u.

Nad jeziki lahko definiramo naslednje operacije: unijo, presek, razliko, komplement (glede na množico vseh besed A^*) in stik. *Potence jezika* lahko definiramo induktivno kot $L^0 = \{\varepsilon\}$ in $L^n = L^{n-1}L$ za vsak $n \in \mathbb{N}$. Jezik L^+ definiramo kot unijo vseh neničelnih potenc jezika L, jezik L^* pa kot unijo prav vseh potenc jezika L.

2.2.1 Regularni jeziki

Pri delu z jeziki nas pogosto zanimajo množice besed, ki jih ni mogoče preprosto našteti, temveč jih želimo opisati z določenimi pravili ali vzorci. Zato potrebujemo formalne načine, kako opisati njihove vzorce. Regularni jeziki so najpreprostejši razred takšnih jezikov, ki jih lahko opišemo z regularnimi izrazi. Njihov pomen izhaja iz dejstva, da so dovolj preprosti, da jih lahko učinkovito obdelujemo z algoritmi, hkrati pa dovolj izrazni, da zajamejo številne naravne primere, ki se pojavljajo v računalništvu.

Množica *regularnih jezikov* nad abecedo A je induktivno definirana kot:

1. Prazen jezik $\emptyset$ je regularen jezik.
2. Za vsak $a \in A$ je enojec $\{a\}$ regularen jezik.
3. Če je L regularen jezik, je tudi L^* regularen.
4. Če sta L in L' regularna jezika, potem sta regularna tudi $L \cup L'$ in LL'.
5. Noben drug jezik iz A^* ni regularen.

Končne jezike lahko vedno izrazimo kot končno unijo posameznih besed. Direktno po definiciji sledi, da je vsak končni jezik regularen.

3. Opis jezikov z avtomati

Gradivo tega razdelka temelji predvsem na [1, razdelek III: 3, 4, 5].

3.1 Deterministični končni avtomati

Deterministični končni avtomat je preprost abstrakten model stroja, ki ga uporabljamo za opisovanje in prepoznavanje vzorcev v besedah nad abecedo A. Naloga avtomata je sprejeti ali zavrniti besedo, odvisno od tega, ali se vzorec, ki ga definira avtomat, pojavlja v besedi. Formalno je to peterica

$$\mathcal{A} = (Q, A, E, q_0, F),$$

kjer je Q končna *množica stanj*, A končna abeceda, $q_0 \in Q$ *začetno stanje*, $F \subseteq Q$ *množica sprejemnih stanj* in $E \subseteq Q \times A \times Q$ *množica prehodov*, kjer za vsako stanje $q \in Q$ in za vsako črko $a \in A$ obstaja največ eno stanje q', tako da je $(q, a, q') \in E$.

Tak avtomat si lahko nazorno predstavljamo kot označen usmerjen graf. Vozlišča grafa predstavljajo stanja, vsaka usmerjena povezava pa ustreza enemu prehodu in je označena s simbolom iz abecede. Začetno stanje običajno označimo z dodatno puščico, ki vanje vstopa od zunaj, in napisom „start“; sprejemna stanja pa z dvojno obrobo.

Za razumevanje delovanja avtomata je ključen pojem poti. Če za dva prehoda (p,a,q) in (p',a',q') velja $q = p'$, ju lahko izvedemo zaporedno. Vsako končno zaporedje takšnih prehodov imenujemo *pot*. V bolj pregledni obliki pot $c = (e_0,e_1,\ldots,e_{n-1}), e_i = (q_i,a_i,q_{i+1})$, v avtomatu $\mathcal{A}$ zapišemo kot

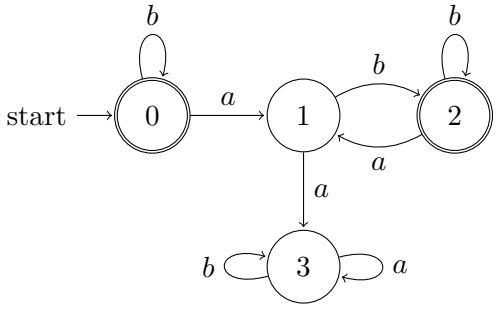
$$c = q_0 \xrightarrow{a_0} q_1 \cdots q_{n-1} \xrightarrow{a_{n-1}} q_n \text{ ali } c = q_0 \xrightarrow{a_0 \cdots a_{n-1}} q_n.$$

Oznaka poti je beseda $a_0a_1 \cdots a_{n-1}$, njena dolžina pa je enaka številu prehodov, torej n. Pot imenujemo *uspešna*, če se začne v začetnem stanju in se konča v enem izmed sprejemnih stanj. Pravimo, da avtomat $\mathcal{A}$ *sprejme* besedo, če obstaja vsaj ena uspešna pot, katere oznaka je ravno ta beseda. Množico vseh sprejetih besed označimo z $L(\mathcal{A})$ in jo imenujemo *jezik, ki ga avtomat $\mathcal{A}$ prepozna*. Dva avtomata sta *ekvivalentna*, kadar prepoznata isti jezik.

3.2 Prepoznavni jeziki

Pri obravnavi jezikov nas ne zanima le, kako jih lahko zapišemo, temveč tudi, kako lahko učinkovito ugotovimo, ali dana beseda pripada izbranemu jeziku ali ne. Zato potrebujemo modele, ki omogočajo odločanje o pripadnosti besed jeziku na sistematičen način. Jezike, za katere obstaja končni avtomat, ki sprejme natanko besede iz tega jezika, imenujemo s končnimi avtomati prepoznavni jeziki. V nadaljevanju jih bomo krajše imenovali kar prepoznavni jeziki. Jezik $L \subseteq A^*$ je *prepoznaven*, če obstaja tak končni avtomat $\mathcal{A}$, da je $L(\mathcal{A}) = L$. Oglejmo si primer:

Zgled 1. Naj bo $\mathcal{A} = (Q, A, E, q_0, F)$ deterministični avtomat, pri čemer so $Q = \{0,1,2,3\}$, $A = \{a,b\}$, $E = \{(0,a,1),(0,b,0),(1,a,3),(1,b,2),(2,a,1),\ (2,b,2),(3,a,3),\ (3,b,3)\}$, $q_0 = 0$, in $F = \{0,2\}$. Avtomat $\mathcal{A}$ je prikazan na sliki 1.



Slika 1. Deterministično avtomat $\mathcal{A}$ prepozna jezik L_{ab}, ki je množica vseh besed nad $\{a,b\}$, v katerih vsaki pojavitvi črke a neposredno sledi črka b.

Jezik L_{ab}, ki ga prepozna avtomat $\mathcal{A}$, je množica vseh besed nad $\{a,b\}$, v katerih vsaki pojavitvi črke a neposredno sledi črka b.

Naj bosta A in B abecedi. Preslikava $h : A^* \to B^*$ je *homomorfizem*, če:

1. $h(\varepsilon) = \varepsilon$;
2. $h(uv) = h(u)h(v)$ za vsaki besedi $u, v \in A^*$.

Če je L regularen jezik, sta regularna tudi $h(L)$ in $h^{-1}(L)$, pri čemer je h^{-1} inverz homomorfizma h. Razred prepoznavnih jezikov je poleg tega zaprt tudi za unijo, stik, komplement, presek in prej omenjena operatorja na jezikih $+$ in $*$.

Iz vsakega avtomata zmoremo izraziti jezik, ki ga prepozna, s pomočjo regularnega izraza. Tudi obratno znamo iz vsakega regularnega izraza sestaviti avtomat, ki tak jezik sprejema. To globoko odvisnost med regularnimi in prepoznavnimi jeziki opisuje Kleenejev izrek, ki ga bomo le navedli, ne pa tudi dokazali:

Izrek 2 (Kleenejev izrek)**.** Jezik je regularen, če in samo če je prepoznaven.

4. Opis jezikov v logiki

V tem razdelku bomo pregledali nekaj osnovnih definicij logike: logika prvega reda, logika drugega reda in monadična logika drugega reda. Večinoma sledimo [1, razdelek IX: 2 in 3], z vmesnimi dodatki iz [4] in [5, razdelek 2].

4.1 Sintaksa

Najprej definirajmo sintakso logike prvega reda, ki jo gradijo *logični simboli*, kamor uvrščamo:

- *logične veznike*: $\wedge$, $\vee$, $\neg$, $\Rightarrow$,
- *kvantifikatorja*: $\exists$, $\forall$,
- *znak enakosti*: $=$,
- *spremenljivke*: $x, y, z, \ldots$,
- *oklepaje*: $(,)$.

Poleg logičnih simbolov uvedemo še množico $\mathcal{L}$, imenovano *signatura jezika* prvega reda, ki jo definiramo kot četverico

$$\mathcal{L} = (L_{\mathrm{R}}, L_{\mathrm{f}}, L_{\mathrm{c}}, \mathrm{ar}),$$

kjer so:

- L_{R} množica *relacijskih simbolov*: na primer $<, \leq, \ldots$,
- L_{f} množica *funkcijskih simbolov*: na primer $\sin, f, \ldots$,
- L_{c} množica *konstantnih simbolov*[1]: na primer $0, 1, \ldots$ in
- $\mathrm{ar} : L_{\mathrm{R}} \cup L_{\mathrm{f}} \to \mathbb{N}$ preslikava, ki vsakemu funkcijskemu in relacijskemu simbolu priredi njegovo *mestnost*. Simbol imenujemo n-mesten, če je njegova mestnost enaka n.

Za interpretacijo formul na besedah nad abecedo A bomo uporabljali dve različni signaturi: signaturo

$$\mathcal{L}_< = (\{<, P_a \mid a \in A\}, \emptyset, \{\min, \max\}, \mathrm{ar}); \qquad \mathrm{ar}(<) = 2, \qquad \mathrm{ar}(P_a) = 1$$

bomo imenovali *signatura urejenosti* in signaturo

$$\mathcal{L}_S = (\{S, P_a \mid a \in A\}, \emptyset, \{\min, \max\}, \mathrm{ar}); \qquad \mathrm{ar}(S) = 2, \qquad \mathrm{ar}(P_a) = 1$$

bomo imenovali *signatura naslednika*.[2] Signatura naslednika je posebej pomembna v povezavi s končnimi avtomati, saj avtomati berejo besedo zaporedno, simbol za simbolom, zato relacija naslednika bolje odraža njihovo lokalno naravo.

[1] Elementi teh treh množic so resnično zgolj simboli, ki jih bomo kot dejanske funkcije, relacije ali konstante interpretirali šele kasneje.

[2] V obeh signaturah oznaka P_a torej predstavlja enomestno relacijo za vsako črko abecede; znaka min in max pa bosta oznaki za konstantna simbola. Pomene bomo razložili ob podani strukturi.

Zdaj bomo podali podroben opis sintaktičnih pravil, po katerih konstruiramo logične formule v treh zaporednih korakih od izrazov, preko osnovnih formul, do formul prvega reda. Najprej definiramo množico $\mathcal{L}$-*izrazov*, ki je induktivno podana kot:

1. Vsaka spremenljivka je izraz.
2. Vsak konstantni simbol je izraz.
3. Če so $t_1, t_2, \ldots, t_n$ izrazi in je f n-mestni funkcijski simbol, potem je izraz tudi $f(t_1, t_2, \ldots, t_n)$.

Osnovne formule so ali formule oblike $t_1 = t_2$, kjer sta t_1 in t_2 izraza, ali oblike $R(t_1, \ldots, t_n)$, kjer so $t_1, \ldots, t_n$ izrazi in je R n-mestni relacijski simbol množice L_{R}. Osnovne formule iz signature urejenosti so torej oblike

$$P_a(x), \qquad x = y, \qquad \min < y,$$

tiste iz signature naslednika pa

$$P_a(x), \qquad x = y, \qquad S(\min, y).$$

Osnovne formule same po sebi še ne zadoščajo za izražanje zahtevnejših lastnosti struktur. Zato iz njih z uporabo logičnih veznikov in kvantifikatorjev postopoma gradimo splošnejše formule, ki jih imenujemo *formule prvega reda*. *Množico vseh formul prvega reda nad signaturo* $\mathcal{L}$ označimo z $\mathbf{FO}[\mathcal{L}]$. To je najmanjša množica, ki vsebuje vse osnovne formule in je zaprta po naslednjem postopku tvorjenja. Če je φ formula prvega reda, $(\varphi_i)_{i\in I}$ končna družina formul prvega reda in x spremenljivka, so tudi

$$\bigwedge_{i\in I} \varphi_i, \qquad \bigvee_{i\in I} \varphi_i, \qquad \neg\varphi, \qquad \varphi_i \Rightarrow \varphi_j, \qquad \exists x\, \varphi \qquad \text{in} \qquad \forall x\, \varphi$$

formule prvega reda. Na ta način lahko iz preprostih osnovnih formul zgradimo poljubno zapletene formule, s katerimi opisujemo lastnosti matematičnih struktur.

Oklepaje pogosto izpustimo in uporabimo pravila o prednosti logičnih veznikov. Spremenljivki x, ki stoji neposredno za kvantifikatorjem v zapisu $\forall x$ ali $\exists x$, bomo rekli kvantifikatorju *lastna spremenljivka*. Posamezen nastop spremenljivke je bodisi prost ali vezan. Nastop spremenljivke v formuli je *vezan*, če gre za lastno spremenljivko ali je spremenljivka v dosegu kvantifikatorja, ki se nanaša nanjo; sicer je nastop *prost*. Spremenljivka je prosta, če je vsaj en od njenih nastopov prost. Logična formula, v kateri so vse spremenljivke vezane, se imenuje *stavek*. Primer stavka je formula: $\exists x\, \forall y\, (y < f(x, 0))$.

Pri zapisu $\varphi(x_1, \ldots, x_n)$ bomo privzeli, da so vse proste spremenljivke formule φ med spremenljivkami $x_1, \ldots, x_n$.[3] V logiki prvega reda uporabljamo spremenljivke, ki jih interpretiramo kot posamezne elemente obravnavane domene (glej razdelek 4.2). Te spremenljivke bomo, kadar bo potrebno razlikovanje, imenovali *spremenljivke prvega reda*. Logika drugega reda pa razširi ta formalizem z dodatno vrsto spremenljivk. Namesto da bi spremenljivke predstavljale posamezne elemente, nove spremenljivke opisujejo relacije med elementi domene. Običajno jih označujemo z velikimi tiskanimi črkami, na primer $X_0, X_1, X_2, \ldots$ Ko poleg spremenljivk prvega reda dovolimo še takšne relacijske spremenljivke, lahko na enak način kot prej uvedemo tudi *formule drugega reda*.

Nabor $\mathcal{L}$-*izrazov* je enak kot v logiki prvega reda, *osnovne formule* pa so ali oblike $t_1 = t_2$, kjer sta t_1 in t_2 izraza, ali oblike $R(t_1, \ldots, t_n)$ ali $X(t_1, \ldots, t_n)$, kjer so $t_1, \ldots, t_n$ izrazi; R je n-mestni relacijski simbol iz L_{R} in X spremenljivka, ki predstavlja n-mestno relacijo.

Množica formul drugega reda nad $\mathcal{L}$ je označena s $\mathbf{SO}[\mathcal{L}]$. Gre za najmanjšo množico, ki vsebuje osnovne formule in je zaprta po naslednjih tvorbenih pravilih. Če je φ formula prvega reda, $(\varphi_i)_{i\in I}$

[3] To pomeni le, da je množica prostih spremenljivk podmnožica množice $\{x_1, \ldots, x_n\}$; ni pa nujno, da se ti množici ujemata.

končna družina formul drugega reda, x spremenljivka prvega reda in X spremenljivka drugega reda, potem so tudi

$$\bigwedge_{i\in I}\varphi_i, \qquad \bigvee_{i\in I}\varphi_i, \qquad \neg\varphi, \qquad \varphi_i \Rightarrow \varphi_j, \qquad \exists x\,\varphi, \qquad \forall x\,\varphi, \qquad \exists X\,\varphi \qquad \text{in} \qquad \forall X\,\varphi$$

formule drugega reda.

Monadična logika drugega reda predstavlja logiko drugega reda, pri kateri dovolimo le takšne spremenljivke drugega reda, ki predstavljajo enomestne relacije. Ker je vsako enomestno relacijo mogoče naravno identificirati z množico elementov, za katere relacija velja, lahko te spremenljivke razumemo kot spremenljivke, ki predstavljajo podmnožice domene. Množico vseh formul te logike nad signaturo $\mathcal{L}$ označimo z $\mathbf{MSO}[\mathcal{L}]$.

Čeprav je ta omejitev na prvi pogled precej stroga, je monadična logika drugega reda še vedno dovolj izrazna, da omogoča opis številnih pomembnih lastnosti struktur. Prav ta raven izrazne moči bo v nadaljevanju ključna pri povezavi med logiko in teorijo avtomatov.

4.2 Semantika

V prejšnjem razdelku smo določili pravila, po katerih gradimo izraze in formule, s čimer smo opisali njihovo sintaktično zgradbo. Da bi lahko o formulah smiselno govorili kot o resničnih ali neresničnih, pa moramo tem simbolnim zapisom pripisati tudi pomen. To dosežemo tako, da izberemo domeno in na njej interpretiramo vse simbole iz signature $\mathcal{L}$.

Struktura $\mathcal{S}$ signature $\mathcal{L} = (L_{\mathrm{R}}, L_{\mathrm{f}}, L_{\mathrm{c}}, \mathrm{ar})$ je urejena četverica

$$\mathcal{S} = (D, \{R^{\mathcal{S}}\}_{R\in L_{\mathrm{R}}}, \{f^{\mathcal{S}}\}_{f\in L_{\mathrm{f}}}, \{c^{\mathcal{S}}\}_{c\in L_{\mathrm{c}}}),$$

kjer je:

- D neprazna množica, imenovana *domena*,
- za vsak n-mestni relacijski simbol $R \in L_{\mathrm{R}}$ je $R^{\mathcal{S}} \subseteq D^n$ njegova interpretacija,
- za vsak n-mestni funkcijski simbol $f \in L_{\mathrm{f}}$ je $f^{\mathcal{S}} \colon D^n \to D$ njegova interpretacija in
- za vsak konstantni simbol $c \in L_{\mathrm{c}}$ je $c^{\mathcal{S}} \in D$ njegova interpretacija.

Zapis $R^{\mathcal{S}}$, $f^{\mathcal{S}}$ in $c^{\mathcal{S}}$ loči med simboli iz signature in njihovo interpretacijo v strukturi $\mathcal{S}$. V nadaljevanju bomo zaradi preglednosti včasih uporabljali isti zapis za simbol in njegovo interpretacijo; iz konteksta bo jasno razvidno, ali simbol označuje sintaktični objekt ali njegovo interpretacijo v strukturi.

Za obravnavo formul na besedah bomo vsako besedo obravnavali kot preslikavo, ki povezuje vsako črko z njenim indeksom. Naj bodo $a_0, a_1, \ldots, a_{n-1}$ črke in $u = a_0a_1\ldots a_{n-1}$ neprazna beseda nad abecedo A. *Domena* neprazne besede u je množica $\mathrm{Dom}(u) = \{0, \ldots, n-1\}$, pri čemer je $n = |u|$. Besedi u priredimo strukturo $\mathcal{S}_u$ nad signaturo $\mathcal{L}_<$ oziroma $\mathcal{L}_S$, definirano kot

$$\mathcal{S}_u = (\mathrm{Dom}(u), \{R^{\mathcal{S}_u}\}, \emptyset, \{c^{\mathcal{S}_u}\}),$$

kjer so interpretacije simbolov podane na naslednji način:

- relacijski simbol $<$ interpretiramo kot relacijo urejenosti: $< = \{(i,j) \in \mathrm{Dom}(u)^2 \mid i < j\}$,
- relacijski simbol S interpretiramo kot relacijo naslednika: $S = \{(i,j) \in \mathrm{Dom}(u)^2 \mid j = i+1\}$,
- za vsak $a \in A$ je $P_a = \{i \in \mathrm{Dom}(u) \mid a_i = a\}$,
- konstantna simbola min in max interpretiramo kot $\min = 0, \max = |u| - 1$.

Spremenljivke prvega reda $x, y, \ldots \in \mathrm{Dom}(u)$ nam torej predstavljajo pozicije v besedi. Enomestne relacije P_a pa predstavljajo množico tistih pozicij znotraj besede u, na katerih se nahaja črka a. Tako z množicami P_a lahko kodiramo pozicije znotraj besede.

Zgled 3. Naj bo $u = abbaab$ beseda nad $A = \{a, b\}$. Tedaj je $\mathrm{Dom}(u) = \{0,1,2,3,4,5\}$ in velja $P_a = \{0,3,4\}$,$P_b = \{1,2,5\}$.

Doslej smo določili pomen relacijskih, funkcijskih in konstantnih simbolov, preostane pa še razlaga, kako interpretiramo spremenljivke. Najprej obravnavajmo spremenljivke prvega reda. Vsaki prosti spremenljivki želimo prirediti konkreten element domene, pri čemer morajo vsi prosti nastopi iste spremenljivke dobiti isto vrednost. Naj bo $\mathcal{S}$ struktura nad signaturo $\mathcal{L}$ z domeno D. *Vrednotenje prvega reda* na strukturi $\mathcal{S}$ je preslikava ν iz množice spremenljivk v množico D. To preslikavo naravno razširimo na vse $\mathcal{L}$-izraze:

1. Če je c konstanten simbol, potem velja $\nu(c) = c^{\mathcal{S}}$.

2. Če so $t_1, \ldots, t_n$ izrazi in je f n-mestni funkcijski simbol, potem velja

$$\nu(f(t_1, \ldots, t_n)) \;=\; f^{\mathcal{S}}(\nu(t_1), \ldots, \nu(t_n)).$$

3. Če je x spremenljivka in $a \in D$, označimo z $\nu[x \mapsto a]$ vrednotenje, ki opisuje zamenjavo vseh prostih nastopov spremenljivke x z elementom $a \in D$ in je definirano z

$$\nu[x \mapsto a](y) = \begin{cases} \nu(y), & \text{če } y \neq x \\ a, & \text{če } y = x. \end{cases}$$

S tem lahko natančno opredelimo, kdaj je formula v dani strukturi resnična. Za formulo prvega reda φ in vrednotenje ν zapišemo $\mathcal{S} \models \varphi[\nu]$, če struktura $\mathcal{S}$ pri vrednotenju ν *zadošča* formuli φ, kar pomeni, da za osnovne formule velja:

$$\begin{aligned} &\mathcal{S} \models (t_1 = t_2)[\nu], && \text{če in samo če } \nu(t_1) = \nu(t_2). \\ &\mathcal{S} \models (R(t_1, \ldots, t_n))[\nu], && \text{če in samo če } (\nu(t_1), \ldots, \nu(t_n)) \in R. \end{aligned}$$

In za formule višje kompleksnosti:

$$\begin{aligned} &\mathcal{S} \models (\neg\varphi)[\nu], && \text{če in samo če } \mathcal{S} \not\models \varphi[\nu]. \\ &\mathcal{S} \models \left(\bigwedge_{i \in I} \varphi_i\right)[\nu], && \text{če in samo če za vsak } i \in I \text{ velja } \mathcal{S} \models \varphi_i[\nu]. \\ &\mathcal{S} \models \left(\bigvee_{i \in I} \varphi_i\right)[\nu], && \text{če in samo če obstaja } i \in I, \text{ da velja } \mathcal{S} \models \varphi_i[\nu]. \\ &\mathcal{S} \models (\varphi \Rightarrow \psi)[\nu], && \text{če in samo če } \mathcal{S} \not\models \varphi[\nu] \text{ ali } \mathcal{S} \models \psi[\nu]. \\ &\mathcal{S} \models (\exists x\, \varphi)[\nu], && \text{če in samo če } \mathcal{S} \models \varphi[\nu[x \mapsto a]] \text{ za neki } a \in D. \\ &\mathcal{S} \models (\forall x\, \varphi)[\nu], && \text{če in samo če } \mathcal{S} \models \varphi[\nu[x \mapsto a]] \text{ za vsak } a \in D. \end{aligned}$$

Resničnost formule je odvisna le od vrednosti njenih prostih spremenljivk, zato pri stavkih običajno vrednotenja sploh ne navajamo. Če formula φ velja v strukturi $\mathcal{S}$ za vsako vrednotenje ν, zapišemo preprosto $\mathcal{S} \models \varphi$.

Semantiko logike drugega reda dobimo tako, da vrednotenju dovolimo še prirejanje relacij spremenljivkam drugega reda. Če ima struktura $\mathcal{S}$ domeno D, potem *vrednotenje drugega reda* vsaki spremenljivki prvega reda priredi element iz D, vsaki n-mestni spremenljivki drugega reda pa neko n-mestno relacijo na domeni, torej neko podmnožico množice D^n.

Če je X spremenljivka drugega reda in R podmnožica D^n, označimo z $\nu[X \mapsto R]$ vrednotenje, ki opisuje zamenjavo vseh prostih nastopov spremenljivke X z relacijo R na domeni D in je definirano z:

$$\nu[X \mapsto R](x) = \nu(x), \qquad \text{če je } x \text{ spremenljivka prvega reda,}$$
$$\nu[X \mapsto R](Y) = \begin{cases} \nu(Y), & \text{če } Y \neq X, \\ R, & \text{če } Y = X, \end{cases} \qquad \text{če je } Y \text{ spremenljivka drugega reda.}$$

Že definiran pojem interpretacije formule za logiko prvega reda je tako dopolnjen z naslednjimi pravili:

$\mathcal{S} \models (X(t_1, \ldots, t_n))[\nu]$, če in samo če $(\nu(t_1), \ldots, \nu(t_n)) \in \nu(X)$.
$\mathcal{S} \models (\exists X\ \varphi)[\nu]$, če in samo če obstaja $R \subseteq D^n$, tako da $\mathcal{S} \models \varphi[\nu[X \mapsto\ R]]$.
$\mathcal{S} \models (\forall X\ \varphi)[\nu]$, če in samo če za vsak $R \subseteq D^n$ velja $\mathcal{S} \models \varphi[\nu[X \mapsto R]]$.

Naj bo φ stavek. Neprazna beseda[4] *u zadošča* φ, če struktura $\mathcal{S}_u$ zadošča formuli φ, ki je enolično določena z besedo u. To označimo z $u \models \varphi$. Za stavek φ lahko definiramo jezik besed $L(\varphi)$, ki mu zadoščajo, kot:

$$L(\varphi) = \{u \in A^+ \mid u \models \varphi\}.$$

Naj bo $L \subseteq A^+$ jezik. Pravimo, da je jezik L *definiran* v $\mathbf{MSO}[\mathcal{L}]$, če obstaja tak stavek φ v $\mathbf{MSO}[\mathcal{L}]$, da je $L(\varphi) = L$. Dve logiki imata *enako izrazno moč*, če definirata isti razred jezikov nad vsemi strukturami: vsaka lastnost, ki jo lahko definiramo v eni logiki, je lahko definirana tudi v drugi.

Stavka φ in ψ sta *logično ekvivalentna*, če imata isto logično vrednost v vseh možnih strukturah, torej če za vsako strukturo $\mathcal{S}$ nad $\mathcal{L}$ velja $\mathcal{S} \models \varphi$, če in samo če $\mathcal{S} \models \psi$. V tem primeru pišemo $\varphi \sim \psi$. Logična ekvivalenca nam omogoča, da formule preoblikujemo v bolj strukturirane oblike. Naslednje formule so logično ekvivalentne:

$$\varphi \Rightarrow \psi \sim \neg\varphi \vee \psi,$$
$$\forall x\ \varphi \sim \neg(\exists x\ \neg\varphi).$$

Posledično lahko formule obravnavamo (do logične ekvivalence natančno), kot da so zgrajene brez simbolov $\Rightarrow$ in $\forall$.

Pri delu s formulami je pogosto koristno, da so vsi kvantifikatorji zbrani na začetku formule. Za formulo prvega reda tako rečemo, da je v *preneksni normalni obliki*, če jo lahko zapišemo v obliki

$$\psi = K_1 x_1\ K_2 x_2\ \cdots\ K_n x_n\ \varphi,$$

kjer je vsak K_i bodisi eksistencialni bodisi univerzalni kvantifikator, formula φ pa ne vsebuje nobenih kvantifikatorjev. Znana trditev pravi, da je vsaka formula prvega reda logično ekvivalentna neki formuli v preneksni normalni obliki in vsaka monadična formula drugega reda je logično ekvivalentna formuli oblike

$$\rho = K_1 X_1 \cdots K_n X_n\ Q_1 x_1 \cdots Q_m x_m\ \varphi,$$

kjer so K_i in Q_i eksistencialni ali univerzalni kvantifikatorji, φ pa formula drugega reda brez kvantifikatorjev.

Zgled 4. Naj bo $A = \{a, b\}$ abeceda. Oglejmo si primer (ne)zadoščanja besede ab različnim stavkom.

[4] Prazno besedo izključimo zaradi tehnične enostavnosti konstrukcij; vse rezultate bi bilo možno razširiti tudi na prazne besede z ustreznimi prilagoditvami.

- $ab \vDash \exists x\ P_a(x)$
- $ab \nvDash \exists x\ \exists y\ \exists z\ (\neg(z = y) \wedge \neg(y = x) \wedge \neg(x = z))$
- $ab \nvDash \exists x\ \exists y\ (x < y \wedge P_b(x) \wedge P_a(y))$
- $ab \nvDash \exists X\ \forall x\ ((X(x) \Rightarrow P_b(x)) \wedge \exists y\ (y = \min \wedge X(y)))$
- $ab \vDash \forall x\ (P_a(x) \Rightarrow \exists y\ (S(x, y) \wedge P_b(y)))$

Sedaj pa za vsakega od zgornjih stavkov poglejmo, kateri jezik definirajo.

φ	$L(\varphi)$
$\exists x\ P_a(x)$	$\{vaw \mid v, w \in A^*\}$
$\exists x\ \exists y\ \exists z\ (\neg(z = y) \wedge \neg(y = x) \wedge \neg(x = z))$	$\{w \mid \lvert w\rvert \geq 3\}$
$\exists x\ \exists y\ (x < y \wedge P_b(x) \wedge P_a(y))$	$\{ubvaw \mid u, v, w \in A^*\}$
$\exists X\ \forall x\ ((X(x) \Rightarrow P_b(x)) \wedge \exists y\ (y = \min \wedge X(y)))$	$\{bw \mid w \in A^*\}$
$\forall x\ (P_a(x) \Rightarrow \exists y\ (S(x, y) \wedge P_b(y)))$	L_{ab}

Formula v zadnjem primeru definira jezik L_{ab}, ki je množica vseh besed nad $\{a,b\}$, v katerih vsaki pojavitvi črke a neposredno sledi črka b.

Če primerjamo zadnji primer zgleda 4 in zgled 1, lahko opazimo, da smo isti jezik opisali tako z avtomatom kot tudi s formulo. To nakazuje na idejo, da med končnimi avtomati in formulami drugega reda obstaja povezava. Izkaže se, da ta dva zgleda nista naključje, saj povezava obstaja tudi v splošnem, o čemer govori Büchijev izrek v razdelku 5..

5. Büchijev izrek

Ta razdelek je posvečen dokazu Büchijevega izreka, ki pravi, da so jeziki nad abecedo A, ki jih lahko definiramo v monadični logiki drugega reda, natanko regularni jeziki. Dokaz desne implikacije je povzet po [6, razdelek 3] in [7, razdelek 5], struktura leve implikacije pa po [1, razdelek IX: 3], s pomočjo [7, razdelek 6].

Izrek 5 (Büchijev izrek)**.** Jezik $L \subseteq A^+$ je prepoznaven, če in samo če je definiran v $\mathbf{MSO}[\mathcal{L}_S]$.

Ker je relacijo naslednika mogoče definirati z relacijo urejenosti in obratno, sta logiki $\mathbf{MSO}[\mathcal{L}_<]$ in $\mathbf{MSO}[\mathcal{L}_S]$ ekvivalentni glede izrazne moči nad besednimi strukturami $\mathcal{S}_u$. To nam omogoča, da v logiki drugega reda prosto prehajamo med obema okoljema. Pri dokazu bomo zaradi preglednosti uporabljali signaturo urejenosti $\mathcal{L}_<$. Utemeljitev ekvivalence izrazne moči si lahko podrobneje ogledamo v [1, razdelek IX: 2.3] in [6, razdelek 2].

Dokaz tega izreka lahko razstavimo na dva dela: prehod iz avtomatov k logičnim formulam in od logičnih formul nazaj k avtomatom. Prvi prehod oziroma implikacija v desno, je lažji, saj preprosto simulira obnašanje avtomata s formulo, kar nam pojasni spodnja trditev 6. Za prehod iz logičnih formul na avtomate pa bomo najprej konstruirali regularni jezik, podan z našim stavkom in se nato oprli na Kleenejev izrek 2, ki pravi, da je vsak regularen jezik tudi prepoznaven. Začnimo z implikacijo v desno.

Trditev 6. Naj bo $\mathcal{A} = (Q, A, E, q_0, F)$ avtomat. Obstaja tak stavek φ v logiki drugega reda nad signaturo naslednika $\mathbf{MSO}[\mathcal{L}_S]$, da velja $L(\varphi) = L(\mathcal{A}) \cap A^+$.[5]

Dokaz. Naj bo $\mathcal{A} = (Q, A, E, q_0, F)$ avtomat, kjer je $Q = \{1, \ldots, n\}$. Naš cilj je skonstruirati stavek φ v $\mathbf{MSO}[\mathcal{L}_S]$, ki definira natanko jezik $L(\mathcal{A}) \cap A^+$; torej množico nepraznih besed, ki so

[5] Ker smo pri definiciji jezika $L(\varphi)$ zaradi tehnične enostavnosti izključili prazno besedo, jo moramo izključiti tudi pri avtomatih. Vse rezultate bi bilo mogoče brez večjih težav razširiti tudi na prazno besedo.

oznake vsaj kakšne uspešne poti v avtomatu $\mathcal{A}$. Spomnimo se, da je uspešna pot predstavljena kot zaporedje stanj, pri čemer je prvo stanje začetno in zadnje sprejemno.

Stavek želimo skonstruirati tako, da bo resničen le, če je podana beseda oznaka uspešne poti v avtomatu $\mathcal{A}$, kar pomeni, da:

- beseda kodira zaporedje stanj avtomata,
- prvo stanje v zaporedju je začetno stanje,
- vsako stanje sledi iz prejšnjega skladno z množico prehodov E,
- ob branju zadnjega simbola preidemo v sprejemno stanje.

Najprej definirajmo formulo ψ, ki izraža zgolj obstoj poti z oznako u. Spomnimo se, da spremenljivke prvega reda $x, y, \ldots \in \mathrm{Dom}(u)$ predstavljajo pozicije v besedi, relacije P_a pa množico tistih pozicij znotraj besede u, na katerih se nahaja črka a. Z množicami P_a lahko kodiramo pozicije znotraj besede, da pa bomo lahko zares izrazili pot avtomata, potrebujemo še način, s katerim bomo opisali pot po stanjih, ki jo avtomat opravi ob branju dane besede.

Za vsak $q \in Q$ uvedemo enomestno relacijo X_q, ki predstavlja množico pozicij v besedi u, pri katerih se avtomat med branjem besede nahaja v stanju q. Zapis $X_q(x)$ pomeni, da je avtomat pred branjem simbola na poziciji x v stanju q.[6] Na primer, če v besedi $u_0u_1 \ldots u_{x-1}u_x \ldots$ po branju predpone $u_0u_1 \cdots u_{x-1}$ avtomat doseže stanje q, potem velja $X_q(x)$:

$$q_0 \xrightarrow{u_0} \cdots \xrightarrow{u_{x-1}} q,$$

kjer je q_0 začetno stanje.

Da beseda u res predstavlja oznako poti avtomata, morajo biti množice X_q paroma disjunktne in prehodi morajo slediti strukturi avtomata. Če je pot v stanju q na poziciji x, v stanju q' na poziciji $x+1$, in je na mestu x simbol a, potem mora veljati $(q, a, q') \in E$. Zapišemo:

$$\psi = \forall x \left(\bigvee_q X_q(x) \wedge \bigwedge_{q \neq q'} \neg \left(X_q(x) \wedge X_{q'}(x) \right) \right) \wedge \left(\forall x \, \forall y \, \left(S(x,y) \Rightarrow \bigvee_{(q,a,q') \in E} \left(X_q(x) \wedge P_a(x) \wedge X_{q'}(y) \right) \right) \right).$$

Poleg tega mora pot biti uspešna. Začetna pozicija (oznaka min) mora tako pripadati začetnemu stanju q_0. Med branjem zadnjega simbola (oznaka max) pa mora obstajati prehod iz zadnjega kodiranega stanja v enega izmed sprejemnih stanj $q_F \in F$.[7] Formulo razširimo v:

$$\psi_+ = \psi \wedge X_{q_0}(\min) \wedge \left(\bigvee_{\substack{(q,a,q_F) \in E \\ q_F \in F}} \left(X_q(\max) \wedge P_a(\max) \right) \right).$$

Končen stavek, ki kodira obstoj uspešne poti, je:

$$\varphi = \exists X_1 \, \exists X_2 \ldots \exists X_n \, \psi_+.$$

S tem smo skonstruirali stavek $\varphi \in \mathbf{MSO}[\mathcal{L}_S]$, ki je resničen natanko na tistih besedah, ki so oznake uspešne poti v avtomatu $\mathcal{A}$. Torej velja $L(\varphi) = L(\mathcal{A}) \cap A^+$, kar zaključi dokaz. □

[6] Relacije X_q tako pokrijejo tudi začetno stanje poti, ki ustreza poziciji 0, preden je prebran prvi simbol. Ker pa je domena besede množica $\{0,1,\ldots,|u|-1\}$, z njimi ne moremo eksplicitno kodirati zadnjega stanja, doseženega po branju celotne besede.

[7] Zadnja pozicija v besedi namreč še pripada domeni, toda naslednji korak avtomata, ki vodi v zadnje stanje, se v modelu ne odraža več kot pozicija v besedi. Zato je zadnje stanje možno le preveriti preko zadnjega simbola in njegovega prehoda.

Zgled 7. Oglejmo si še enkrat avtomat na sliki 1.

V tabeli 1 je za lažjo predstavo primer množic pozicij sprejete besede *bababb*, ki jo opišemo s potjo $0 \xrightarrow{b} 0 \xrightarrow{a} 1 \xrightarrow{b} 2 \xrightarrow{a} 1 \xrightarrow{b} 2 \xrightarrow{b} 2$. Opazimo, da zadnje doseženo stanje 2 ni v nobeni množici.

Tabela 1. Množice pozicij za sprejeto besedo *bababb* v avtomatu $\mathcal{A}$ iz zgleda 1.

pot	0	$\xrightarrow{b}$	0	$\xrightarrow{a}$	1	$\xrightarrow{b}$	2	$\xrightarrow{a}$	1	$\xrightarrow{b}$	2	$\xrightarrow{b}$	2
P_a				$\{1,$				$3\}$					
P_b		$\{0,$				$2,$				$4,$		$5\}$	
X_0		$\{0,$		$1\}$									
X_1						$\{2,$				$4\}$			
X_2								$\{3,$				$5\}$	
X_3		$\{$										$\}$	

Preden se lotimo dokaza implikacije v levo v Büchijevem izreku, najprej predstavimo njegovo osnovno idejo. Dokaz bomo izvedli z indukcijo po zgradbi formul v $\mathbf{MSO}[\mathcal{L}_<]$.[8]

Vsaki od osnovnih formul $x_i = x_j$, $x_i < x_j$, $P_a(x)$ in $X_i(x_j)$ bomo priredili ustrezne regularne jezike označenih besed, tj. besed nad razširjeno abecedo, kjer posamezne pozicije vsebujejo informacijo o tem, katere spremenljivke se nahajajo na posameznem mestu.

V induktivnem koraku se nato opremo na dejstvo, da je razred regularnih jezikov zaprt za logične operacije, kot so konjunkcija, disjunkcija in negacija, ter za homomorfizme. Slednje uporabimo pri odstranjevanju prostih spremenljivk. Na ta način lahko iz avtomatov za osnovne formule postopoma zgradimo avtomat, ki sprejema natanko tiste besede, ki zadoščajo stavku φ. Po Kleenejevem izreku pa vemo, da je vsak regularen jezik tudi prepoznaven. Na naslednjih straneh bomo to konstrukcijo izvedli natančno in s tem zaključili dokaz izreka.

Pri prehodu od stavkov k avtomatom naletimo na prvo težavo že pri sami definiciji jezika $L(\varphi)$. Ta je definirana le za stavke, torej formule brez prostih spremenljivk. V nadaljevanju pa bomo zaradi induktivne konstrukcije avtomatov obravnavali tudi formule s prostimi spremenljivkami. Takim formulam ne bomo neposredno prirejali jezikov nad abecedo A, temveč jezike označenih besed nad razširjeno abecedo, ki poleg simbolov besede kodirajo tudi vrednosti prostih spremenljivk. *Razširjeno abecedo* uvedemo kot: $B_{p,q} = A \times \{0,1\}^p \times \{0,1\}^q$, kjer sta p in q naravni števili, pri čemer je p enak številu prostih spremenljivk prvega reda in q enak številu prostih spremenljivk drugega reda v logični formuli $\varphi \in \mathbf{MSO}[\mathcal{L}_<]$.

Črko $b \in B_{p,q}$ zapišemo kot zaporedje $b = (b_0, b_1, \ldots, b_p, b_{p+1}, \ldots, b_{p+q})$, kjer je $b_0 \in A$ črka osnovne abecede; vrednosti $b_1, \ldots, b_p \in \{0,1\}$ označujejo, ali se na tej poziciji nahaja posamezna prosta spremenljivka prvega reda, in vrednosti $b_{p+1}, \ldots, b_{p+q} \in \{0,1\}$ označujejo, ali je ta pozicija vključena v relacijo, ki ustreza posamezni prosti spremenljivki drugega reda.

Besedo $w \in B_{p,q}^*$ torej lahko identificiramo z $(p+q+1)$-terico zaporedij enake dolžine, kar zapišemo kot $w = (w_0, w_1, \ldots, w_p, w_{p+1}, \ldots, w_{p+q})$.

Osredotočili se bomo na množico *označenih besed* $K_{p,q} \subseteq B_{p,q}^*$ nad razširjeno abecedo, katere elementi so besede z dodanimi oznakami določenih pozicij, oziroma z označenimi mesti v besedi, ki jih zasedejo posamezne spremenljivke. $K_{p,q}$ je množica vseh besed nad $B_{p,q}$, pri katerih vsaka od prvih p komponent $w_1, \ldots, w_p$ označuje pozicijo spremenljivke prvega reda, torej vsebuje natanko eno pojavitev 1; in vsaka od zadnjih q komponent označuje spremenljivke drugega reda, torej poljubno število pozicij. V zgornjem primeru besede w to pomeni, da:

- $w_0 \in A^*$ predstavlja osnovno besedo nad abecedo A,

[8] Konstrukcija avtomatov za osnovne formule je znotraj $\mathbf{MSO}[\mathcal{L}_<]$ nekoliko preglednejša, zato lahko zaradi ekvivalence izražalne moči $\mathbf{MSO}[\mathcal{L}_<]$ in $\mathbf{MSO}[\mathcal{L}_S]$ dokaz izvedemo nad signaturo urejenosti.

- vsako zaporedje $w_i \in \{0,1\}^*$ za $i \in \{1,\ldots,p\}$ označuje pozicijo v besedi w_0, kjer se nahaja prosta spremenljivka prvega reda z indeksom i,
- vsako zaporedje $w_j \in \{0,1\}^*$ za $j \in \{p+1,\ldots,p+q\}$ označuje pozicije v besedi w_0, ki pripadajo enomestni relaciji, ki predstavlja prosto spremenljivko drugega reda z indeksom j.

Zgled 8. Če je $A = \{a, b\}$, vsak izmed stolpcev predstavlja posamezno črko v $B_{p,q}$. Primer črke iz abecede $B_{3,2}$ je torej stolpec: $(a, 0, 0, 1, 0, 1)$. Celotna označena beseda w; $w_0 = abaabab$ iz $K_{3,2} \subseteq B^*_{3,2}$, je predstavljena spodaj.

w_0	a	b	a	a	b	a	b
w_1	0	1	0	0	0	0	0
w_2	0	0	0	0	1	0	0
w_3	1	0	0	0	0	0	0
w_4	0	1	1	0	0	1	1
w_5	1	1	0	1	0	1	0

V nadaljevanju bomo, kadar bo kontekst jasen, pisali B namesto $B_{p,q}$ in K namesto $K_{p,q}$. Za vsaka $p, q \geq 0$ je množica $K_{p,q}$ regularen jezik nad $B_{p,q}$, kar z malo truda lahko dokažemo, vendar bomo za potrebe članka dokaz izpustili.

Formule na besedah $B^*_{p,q}$ si razlagamo na isti način, ki je predstavljen v razdelku 4., le razlago relacije P_a moramo nekoliko spremeniti, da bo zajemala le pozicije, definirane na vhodni besedi w_0. Naj bo A abeceda, $a_0, \ldots, a_{n-1}$ črke in $w_0 = a_0a_1 \ldots a_{n-1}$ beseda iz A^*. Interpretacijo relacijskega simbola P_a na besedi w_0 določimo kot $P_a = \{i \in \mathrm{Dom}(w_0) \mid a_i = a\}$.

Naj bo $\varphi(x_1,\ldots,x_p,X_1,\ldots,X_q)$ formula, katere proste spremenljivke prvega reda so natanko spremenljivke $x_1, \ldots, x_p$ in proste spremenljivke drugega reda natanko $X_1, \ldots, X_q$. Predpostavimo, da je $w \in K_{p,q}$ označena beseda oblike $w = (w_0,w_1,\ldots,w_{p+q})$. Naj bo n_i edina pozicija, na kateri ima beseda w_i vrednost 1 za vsak $i \in \{1,\ldots,p\}$. V zgledu 8 so tako $n_1 = 1, n_2 = 4, n_3 = 0$.

Rečemo, da *označena beseda w zadošča formuli φ*, in zapišemo $w \models \varphi$, če osnovna beseda w_0 zadošča formuli φ pri vrednotenju ν, ki ga določimo na naslednji način:

$$\begin{aligned} \nu(x_i) &= n_i, && 1 \leq i \leq r; \\ \nu(X_j) &= \{\, k \in \mathrm{Dom}(w_0) \mid (w_{p+j})_k = 1 \,\}, && 1 \leq j \leq s. \end{aligned}$$

To pomeni, da vsaka spremenljivka prvega reda x_i predstavlja natanko tisto pozicijo, ki je označena z enico v pripadajoči besedi w_i. Spremenljivka drugega reda X_j pa ustreza množici vseh pozicij, na katerih ima beseda w_{p+j} vrednost 1.[9] Če je $p = q = 0$, s tem dobimo običajno interpretacijo stavkov.

Zgled 9. Naj bo osnovna abeceda $A = \{a, b\}$ ter naj bosta $p = 3$ in $q = 2$. Razširjena abeceda je: $B_{3,2} = A \times \{0,1\}^3 \times \{0,1\}^2$ in označena beseda na tej abecedi je oblike $w = (w_0, w_1, w_2, w_3, w_4, w_5)$. Izberimo konkretno označeno besedo iz zgleda 8. in obravnavajmo logično formulo:

$$\varphi(x_1, x_2, x_3, X_1, X_2) = (x_1 < x_2) \wedge (X_1(x_1)) \wedge P_b(x_2)$$

z definiranim vrednotenjem

$$\begin{aligned} \nu(x_1) &= n_1 = 1, \\ \nu(x_2) &= n_2 = 4, \\ \nu(x_3) &= n_3 = 0, \\ \nu(X_1) &= \{1, 2, 5, 6\}, \\ \nu(X_2) &= \{0, 1, 3, 5\}. \end{aligned}$$

[9] Tudi če formula ne uporablja katere od svojih prostih spremenljivk, to ne vpliva na logično vrednost, saj se ne pojavi. Takrat je njena vrednost v označeni besedi lahko poljubna.

Preverimo, ali označena beseda w zadošča formuli

$$\varphi(x_1, x_2, x_3, X_1, X_2) = (1 < 4) \wedge (1 \in \{1,\, 2,\, 5,\, 6\}) \wedge P_b(4).$$

Označena beseda w zadošča formuli φ.

S strukturo, ki jo enolično določa označena beseda w, lahko definiramo jezike tudi za formule, ki niso nujno stavki. Za formulo $\varphi(x_1, \ldots, x_p, X_1, \ldots, X_q)$ označimo z $L_{p,q}(\varphi) = \{\, w \in K_{p,q} \mid w \models \varphi \,\}$ jezik vseh označenih besed, ki zadoščajo formuli φ. Ponovno bomo, kadar kontekst to dopušča, namesto $L_{p,q}(\varphi)$ uporabljali zapis $L(\varphi)$.

Za nadaljevanje dokaza izreka 5 bomo z indukcijo po zgradbi formul dokazali, da so vsi jeziki $L(\varphi)$ regularni. Začnimo z osnovnimi formulami.

Trditev 10. Naj bo $a \in A$ in naj bo $\varphi(x_1, \ldots, x_p, X_1, \ldots, X_q)$ ena izmed osnovnih formul iz signature $\mathcal{L}_<$, torej ena izmed $x_i = x_j, x_i < x_j, P_a(x_i)$ ali $X_i(x_j)$. Potem je jezik $L(\varphi)$ regularen nad $B_{p,q}$.

Dokaz. Za $i, j \in \{1, \ldots, p+q\}; i < j$ in $a \in A$ definiramo množice črk iz abecede B:

$$\begin{aligned} C_i &= \{b \in B \mid b_i = 1\}, \\ C_{i,j} &= \{b \in B \mid b_i = b_j = 1\}, \\ C_{i,a} &= \{b \in B \mid b_i = 1 \text{ in } b_0 = a\}, \end{aligned}$$

ki označujejo tiste črke iz razširjene abecede B, ki imajo posamezne spremenljivke na ustreznih pozicijah.

Naj bodo zdaj $i, j \in \{1, \ldots, p\}$ in $k \in \{1, \ldots, q\}$. Velja:

$$\begin{aligned} L(x_i = x_j) &= K \cap B^* C_{i,j} B^*, \\ L(x_i < x_j) &= K \cap B^* C_i B^* C_j B^*, \\ L(P_a(x_i)) &= K \cap B^* C_{i,a} B^*, \\ L(X_k(x_i)) &= K \cap B^* C_{i,p+k} B^*. \end{aligned}$$

Ker so množice $C_{i,a}, C_{i,j}, C_i$ končne in je B končna abeceda, so $B^* C_{i,j} B^*$, $B^* C_i B^* C_j B^*, B^* C_{i,a} B^*$ in $B^* C_{i,p+k} B^*$ regularni jeziki. Ker je tudi K regularen in je razred regularnih jezikov zaprt za končne preseke, so vsi zgornji jeziki regularni. □

Za osnovne formule smo torej regularnost njihovih jezikov že dokazali, v induktivnem koraku pa bomo dokazali regularnost kompleksnejših formul, saj se konjunkcije, disjunkcije in negacije zlahka pretvorijo v logične operacije, ki regularnost ohranjajo. Spodnje trditve 11 ne bomo dokazovali, saj dokaz sledi neposredno iz definicij.

Trditev 11. Naj bo $(\varphi_i)_{i \in I}$ končna družina formul. Veljajo naslednje enakosti:

$$\begin{aligned} L\left(\bigvee_{i \in I} \varphi_i\right) &= \bigcup_{i \in I} L(\varphi_i), \\ L\left(\bigwedge_{i \in I} \varphi_i\right) &= \bigcap_{i \in I} L(\varphi_i), \\ L(\neg\varphi) &= K_{p,q} - L(\varphi). \end{aligned}$$

Trditev 11 pokaže, da regularnost ohranjajo tudi logični vezniki. Da pridemo do vseh formul drugega reda, nam preostane le še obravnava eksistenčnih kvantifikatorjev, torej formul oblike $\exists x\, \varphi$ in $\exists X\, \varphi$.

Osnovna ideja je naslednja: označena beseda vsebuje tudi informacijo o vrednostih prostih spremenljivk. Če želimo izraziti, da obstaja neka vrednost spremenljivke, moramo odstraniti ustrezno komponento iz razširjene abecede. To dosežemo s projekcijo, ki izbriše eno komponento in tako realizira eksistenčni kvantifikator. S *projekcijo* π_i torej označimo funkcijo, ki izbriše komponento z indeksom i:

$$\pi_i(b_0, b_1, \ldots, b_i, \ldots, b_{p+q}) = (b_0, b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_{p+q}).$$

S tem π_i obravnavamo kot homomorfizem iz $B^*_{p,q}$ v $B^*_{p-1,q}$, če je $i \leq p$, in v $B^*_{p,q-1}$, če je $p < i \leq p+q$. Homomorfizem deluje točkovno na črkah in izbriše i-to komponento vsake črke v celi besedi. Spodnja trditev 12 spet sledi neposredno iz definicije eksistenčnega kvantifikatorja.

Trditev 12. Naj bo $\varphi(x_1, \ldots, x_p, X_1, \ldots, X_q)$ formula s p prostimi spremenljivkami prvega reda in q prostimi spremenljivkami drugega reda. Veljata naslednji enakosti:

$$L_{p-1,q}(\exists x_p\, \varphi) = \pi_p(L_{p,q}(\varphi)),$$
$$L_{p,q-1}(\exists X_q\, \varphi) = \pi_{p+q}(L_{p,q}(\varphi)).$$

Če je $L(\varphi)$ regularen jezik po indukcijski predpostavki, potem sta regularna tudi $L(\exists x\ \varphi)$ in $L(\exists X\ \varphi)$, saj lahko predpostavimo $x = x_p$ in $X = X_q$.[10] Potem velja $L(\exists x_p\ \varphi) = \pi_p(L(\varphi))$ in $L(\exists X_q\ \varphi) = \pi_{p+q}(L(\varphi))$ po trditvi 12. Ker je projekcija homomorfizem in so regularni jeziki zaprti za homomorfizme, kar je posledica Kleenejevega izreka 2 in lastnosti razreda regularnih jezikov, sta tudi $L(\exists x_p \varphi)$ in $L(\exists X_q\ \varphi)$ regularna.

S trditvijo 10 smo pokazali, da za vsako osnovno formulo φ velja, da je jezik $L(\varphi)$ regularen. Po trditvah 11 in 12 velja, da razred regularnih jezikov ostane zaprt za vse logične operacije, ki nastopajo v $\mathbf{MSO}[\mathcal{L}_<]$, torej za disjunkcijo, konjunkcijo, negacijo in eksistenčno kvantifikacijo. Po indukciji po zgradbi formule sledi, da je za vsako formulo φ v $\mathbf{MSO}[\mathcal{L}_<]$ jezik $L(\varphi)$ regularen.

Ko upoštevamo še ekvivalenco signatur, dobimo še drugo smer Büchijevega izreka: vsak jezik, definiran v $\mathbf{MSO}[\mathcal{L}_S]$, je regularen in po Kleenejevem izreku 2 prepoznaven. Skupaj z obratno smerjo, podano v trditvi 6, sledi ekvivalenca med prepoznavnimi jeziki in jeziki, definiranimi v $\mathbf{MSO}[\mathcal{L}_S]$.

6. Zaključek

Končni avtomati in logika drugega reda na prvi pogled predstavljata povsem različna matematična pojma. Avtomati opisujejo postopek branja besede kot zaporedje prehodov med stanji, medtem ko logične formule izražajo lastnosti besed z uporabo kvantifikatorjev in relacij med njihovimi pozicijami. Büchijev izrek pokaže, da gre v resnici za dva enakovredna pogleda na isti razred jezikov.

Pokazali smo, da lahko vsak prepoznavni jezik opišemo s stavkom monadične logike drugega reda, po drugi strani pa lahko za vsak jezik, definiran v tej logiki, skonstruiramo končni avtomat, ki ga prepoznava. To pomeni, da lahko isti jezik opišemo na dva različna načina: z avtomatom, ki preverja, ali beseda pripada jeziku, ali pa z logično formulo, ki izraža lastnosti besed v tem jeziku.

Povezava med avtomati in logiko je eden od temeljnih rezultatov teorije formalnih jezikov in predstavlja pomemben primer prepletanja matematične logike ter teoretičnega računalništva. Büchijev izrek zato ne podaja zgolj nove karakterizacije prepoznavnih jezikov, temveč razkriva tudi globljo povezavo med dvema na videz zelo različnima načinoma opisovanja matematičnih struktur.

LITERATURA

[1] J.-É. Pin, *Mathematical Foundations of Automata Theory*, (2025), 37–86, 157–168. Dostopno na: https://www.irif.fr/~jep/PDF/MPRI/MPRI.pdf [ogled 15. 5. 2026].

[10] To lahko storimo brez izgube splošnosti, saj formule lahko obravnavamo kot enake do preimenovanja vezanih spremenljivk natančno, torej lahko dosledno preimenujemo vezane spremenljivke.

[2] A. Nemanič, *Monadična logika drugega reda in prepoznavni jeziki*, (2026). Dostopno na: https://repozitorij.uni-lj.si/Dokument.php?id=236524&lang=slv [ogled 11. 8. 2026].

[3] J. Berstel, D. Perrin in C. Reutenauer, *Automata and Formal Languages*, (2009), 1–10. Dostopno na: https://ndl.ethernet.edu.et/bitstream/123456789/33081/1/29.pdf.pdf [ogled 15. 5. 2026].

[4] M. Viswanathan, *First Order Logic*, (2018). Dostopno na: https://courses.physics.illinois.edu/cs498mv/fa2018/FirstOrderLogic.pdf [ogled 15. 5. 2026].

[5] G. Fijavž, *Diskretne strukture*, (2015), 42–52. Dostopno na: https://matematika.fri.uni-lj.si/ds/ds.pdf [ogled 15. 5. 2026].

[6] W. Thomas, *Languages, Automata, and Logic*, (1996), 2–13. Dostopno na: https://www.irif.fr/~serre/Files/thomas96_handbook.pdf [ogled 15. 5. 2026].

[7] M. Mukund, *Regular Languages: From Automata to Logic and Back*, (2011). Dostopno na: https://www.cmi.ac.in/~madhavan/papers/pdf/regular-languages-2011.pdf [ogled 15. 5. 2026].