Nebojša Ikodinović, Tatjana Stojanović

# Formalni jezici i automati

Kragujevac
2018

# Sadržaj

# Uvod

Računarske nauke počivaju na nekoliko fundamentalnih pitanja:

- *Šta je algoritam?*
- *Šta je moguće izračunati, a šta ne?*
- *Kada algoritam možemo smatrati praktično upotrebljivim?*

Više od 80 godina naučnici iz oblasti računarstva se bave ovim pitanjima. Pre nastanka prvih računara, 30tih godina prošlog veka, Alan Tjuring (eng. *Alan Turing*, 1912 -— 1954) je proučavao abstraktne mašine koje su, posmatrano sa stanovišta problema koje bi mogle da izračunaju, imale iste mogućnosti kao savremeni računari. Tjuring je imao za cilj da jasno definiše granicu izmedju problema koje mašine mogu da reše i onih koje ne mogu. Njegovi zaključi su primenljivi ne samo na abstraktne *Tjuringove mašine* već i na savremene računare.

Teorija automata se bavi izučavanjem abstraktnih „mašina" za izračunavanje. Definicije i osobine matematičkih modela izračunavanja su osnov savremene računarske nauke. Teorija izračunljivosti se bazira na svega nekoliko elementarnih i diskretnih koncepata, poput *konačni skupovi* i *nizovi*, od se kojih se, sloj po sloj, gradi pojam - *računar*. Razvoju teorije izračunljivosti, osim matematičara, doprinose i druge oblasti, poput biologije i lingvistike. Već 40tih i 50tih godina dvadestog veka su proučavane jednostavne mašine, koje danas zovemo *konačni automati*. U početku su ovi automati uvedeni da bi se modelovale moždane funkcije. Klini (eng. *Stephen Kleene*, 1909 – 1994 ) je 1956 godine objavio rad u kome je pokazao ekvivalenciju modela mreže neurona i konačnih automata. Kasnije se ispostavile da su konačni automati korisni i za druge namene, pa je trenutno najpopularnija njihova upotreba u tekst editorima i pri leksičkoj analizi teksta. Kasnih 50tih godina dvadestog veka lingvist Noam Čomski (eng. *Noam Chomsky*, 1928 – ) je počeo da proučava formalne gramatike. U svom radu iz 1956 godine, Čomski je dao matematički model za opis prirodnih jezika koristeći stabla. Iako nisu mašine, formalne gramatike su u bliskoj vezi sa automatima i kao takve postale su osnov za razvoj veštačkih jezika i time posale deo kompajlera programskih jezika. Proširujući Tjuringova istraživanja o tome šta može, a šta ne može biti sračuanto, Kuk (eng. *Stephen Cook*, 1939 –) je 1969

godine razdvojio probleme na one koje mogu uz pomoću računara biti efikasno rešeni i one koji u principu mogu biti rešeni, ali u praksi zahtevaju previše vremena čak i za male primere posmatranog problema. Kasnije je klasa problema koji se ne mogu efikasno rešiti nazvana **NP**-teški problemi. Vrlo je verovatno da čak ni eksponencijalni poboljšanja u brzini rada računara neće uticati na našu mogućnost da rešimo velike primere iz ove klase problema.

Sva pomenuta teorijska dostignuća igraju važnu ulogu u savremenoj računarskoj nauci. Pojmovi poput konačnih automata i formalnih gramatika se koriste u razvoju softvera, dok Tjuringove mašine daju bolje razumevanje šta možemo da očekujemo od softvera. Potrebno je za problem koji se rešava, prepoznati kojoj klasi problema pripada i ako je rešiv napisati softver koji će ga rešiti ili u slučaju NP-problema naći približno rešenje ili iskoristiti neku heuristiku ili nekim drugim metodom smanjiti vreme potrebno za rešavanje problema.

Postoji nekoliko razloga zašto je proučavanje automata važan deo osnove računarskih nauka. Konačni automati se mogu koristiti za:

- softver za dizajn i proveru ponašanja digitalnih kola
- leksički analizator kompajlera
- softver koji u velikom tekstu, poput kolekcije veb strana, pronalazi ključne reči, fraze i druge šablone
- softver za proveru sistema koji imaju konačan niz stanja, poput komunikacionih protokola ili protokola za bezbednu razmenu informacija

S obzirom na trend razvoja računarske nauke, specifično tehničko znanje postaje neupotrebljivo posle nekoliko godina. Sa druge strane, poznavanje teorijskih osnova koje se, u ovom slučaju, tiču teorije automata, problema izračunljivosti i kompleksnosti, omugućava lako usvajanje novih tehnologija i aktivno učešće u razvoju istih.

# Glava 1

# Alfabet, reč, jezik

Uopšteno govoreći, računari rade sa tekstovima koje možemo posmatrati kao nizove simbola nad nekim zadatim alfabetom. Programi su tekstovi nad alfabetom tastature, ulazi i izlazi su takodje neki tekstovi nad istim alfabetom, pri čemu program transformiše ulazni tekst u izlazni. Osnovni cilj ovog poglavlja jeste da uvedemo formalizam koji je pogodan za rad sa tekstovima. Osnovni pojmovi su *alfabet*, *reč* i *jezik*, koji će biti polazni za formalno uvodjenje fundamentalnih pojmova računarstva kao što su *algoritam (program)*, *računar*, *izračunavanje* itd.

## Alfabet

**Definicija 1.1.** *Alfabet (azbuka)* $\Sigma$ *je neki konačan, neprazan skup elemenata. Elementi skupa se nazivaju simboli.*

Svaki alfabet koristimo da bismo napisali neki tekst.

PRIMER 1.1. [**Važni alfabeti**] U nastavku, često ćemo posmatrati sledeće alfabete:

- $\Sigma_U = \{|\}$ je tzv. *unarni alfabet*, koji sadrži samo jedan simbol – vertikalnu crtu;
- $\Sigma_{\text{bool}} = \{0,1\}$ je *Bulov alfabet*, veoma važan za računarstvo;
- $\Sigma_m = \{0,1,\ldots,m-1\}$, za $m \geqslant 1$, jeste alfabet koji se koristi za zapisivanje prirodnih brojeva u bazi $m$ (primetimo da je $\Sigma_2 = \Sigma_{\text{bool}}$);
- $\Sigma_{\text{lat}} = \{a,b,c,\ldots,z\}$ je latinica;
- $\Sigma_{\text{keyboard}} = \{\mathtt{A},\mathtt{a},\mathtt{B},\mathtt{b},\ldots,\mathtt{Z},\mathtt{z},\sqcup,>,<,(,),\ldots,!\}$ je alfabet svih simbola tastature, pri čemu $\sqcup$ označava blanko znak;
- $\Sigma_{\text{logic}} = \{0,1,p,(,),\wedge,\vee,\neg\}$ jeste jezik koji možemo koristiti za prikaz iskaznih formula.

Naravno, pored nabrojanih koristićemo po potrebi i neke druge alfabete.

## Reč

Za razliku od prirodnih jezika, gde reč označava osnovnu jedinicu jezika, u računarskoj terminologiji *reč* nad alfabetom $\Sigma$ predstavlja svaki konačni niz simbola iz $\Sigma$.

**Definicija 1.2.** ***Reč*** *nad alfabetom $\Sigma$ je svaki konačan niz simbola iz $\Sigma$. Prazna reč, u oznaci $\varepsilon$ ($\lambda$, e), je reč koja ne sadrži ni jedan simbol. Dužina reči $w$, u oznaci $|w|$, je broj simbola u reči $w$.*

Ako je $\Sigma$ alfabet, za $m \geqslant 2$, skup $\Sigma^m = \underbrace{\Sigma \times \cdots \times \Sigma}_{m \text{ puta}}$ svih $m$-torki simbola nazivamo skupom svih *reči nad $\Sigma$ dužine $m$*, i umesto $(a_1, \cdots, a_m)$ pišemo $a_1 \cdots a_m$. Skup $\Sigma^1$, tj. skup svih reči nad $\Sigma$ dužine 1, identifikujemo sa skupom $\Sigma$. Skup $\Sigma^0$ je jednočlani skup čiji ćemo element označavati sa $\varepsilon$ i pri tome ćemo smatrati da je i $\varepsilon$ reč nad $\Sigma$ i to *prazna reč*. Usvajamo i sledeće oznake:

$$\Sigma^+ = \bigcup_{m \geqslant 1} \Sigma^m, \qquad \Sigma^* = \bigcup_{m \geqslant 0} \Sigma^m = \Sigma^+ \cup \{\varepsilon\}.$$

Dakle, elementi skupa $\Sigma^*$ su *reči* nad $\Sigma$.

PRIMER 1.2.

$$\Sigma^0_{\text{bool}} = \{\varepsilon\}, \Sigma^1_{\text{bool}} = \{0,1\}, \Sigma^2_{\text{bool}} = \{00,01,10,11\}, \ldots$$

$$\begin{aligned} \Sigma^*_{\text{bool}} &= \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 100, 011, \ldots\} \\ &= \{\varepsilon\} \cup \{x_1 x_2 \cdots x_n \mid n \in \mathbb{N}^+, x_i \in \Sigma_{\text{bool}} \text{ za } i = 1, \ldots, n\} \end{aligned}$$

$$\Sigma^+_{\text{bool}} = \{0, 1, 00, 01, 10, 11, 000, 001, 010, 100, 011, \ldots\}$$

Reči koristimo za prikazivanje različitih objekata kao što su brojevi, formule, grafovi, programi, itd.

PRIMER 1.3. [**Unarne reprezentacije brojeva**] Svaku reč

$$\underbrace{|| \cdots |}_{n \text{ puta}} = |^n \in \Sigma^*_{\text{U}}$$

možemo posmatrati kao unarnu reprezentaciju prirodnog broja $n$. Primetimo da je $||^n| = n$.

PRIMER 1.4. [**Binarne reprezentacije brojeva**] Reč $x = x_1 x_2 \ldots x_n \in \Sigma^*_{\text{bool}}$ možemo posmatrati kao binarnu reprezentaciju prirodnog broja različitog od nule:

$$\text{num}(x) = \sum_{i=1}^{n} 2^{n-i} x_i.$$

Za svaki prirodan broj $m$ različit od nule, neka je $\text{bin}(m) \in \Sigma^*_{\text{bool}}$ najkraća binarna reprezentacija broja $m$, tj. reč alfabeta $\Sigma_{\text{bool}}$ koja počinje simbolom 1. Dakle,

$$\text{num}(\text{bin}(m)) = m.$$

Nije teško pokazati da je $|\text{bin}(m)| = [\log_2 m] + 1$, za $m \geqslant 1$. Po dogovoru uzimamo da je $\text{bin}(0) = 0$.

PRIMER 1.5. [**Reprezentacije konačnih nizova prirodnih brojeva**] Konačan niz $a_1, \ldots, a_n$, $n \in \mathbf{N}^+$ i $a_i \in \mathbb{N}$, možemo predstaviti kao reč

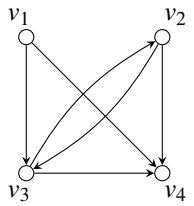
$$\text{bin}(a_1)\#\text{bin}(a_2)\#\cdots\#\text{bin}(a_n) \in \{0,1,\#\}^*.$$

PRIMER 1.6. [**Reprezentacija usmerenih grafova**] Neka je $\mathbb{G} = (V, E)$ *usmeren graf*, pri čemu je $V$ skup čvorova i $E \subseteq \{(u,v) \mid u,v \in V, u \neq v\}$ skup ivica. Neka je $|V| = n$ broj elemenata skupa $V$. Graf $\mathbb{G}$ možemo predstaviti tzv. *matricom susedstva* $M_{\mathbb{G}} = [a_{ij}]$, dimenzije $n \times n$, pri čemu je

$$a_{ij} = 1 \text{ ako } (v_i, v_j) \in E \text{ i } a_{ij} = 0 \text{ ako } (v_i, v_j) \notin E.$$

Dakle, $a_{ij} = 1$ znači da $\mathbb{G}$ sadrži ivicu $(v_i, v_j)$ od $v_i$ do $v_j$, a $a_{ij} = 0$ znači da $\mathbb{G}$ ne sadrži ivicu od $v_i$ do $v_j$. Matrica susedstva $M_{\mathbb{G}}$ može se predstaviti kao reč nad $\{0,1,\#\}$ tako što redom navodimo vrste matrice označavajući kraj vrste simbolom #.

Posmatrajmo graf na narednoj slici.




Njegova matrica susedstva je

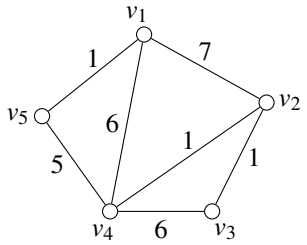
$$\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Predložena reprezentacija ove matrice jeste reč (nad $\{0,1,\#\}$):

0011#0011#0101#0000#

PRIMER 1.7. [**Reprezentacija težinskih neusmerenih grafova**] Veoma važnu vrstu objekata predstavljaju tzv. *težinski neusmereni grafovi* $\mathbb{G} = (V, E, h)$, gde je $E$ podskup skupa dvočlanih podskupova od $V$ i $h : E \to \mathbb{N} \setminus \{0\}$. Vrednost $h(e)$ nziva se *težina* ivice $e \in E$. I ovakvi grafovi se mogu predstaviti matricom susedstva $M_{\mathbb{G}} = [a_{ij}]$:

$$a_{ij} = 0 \text{ ako } \{v_i, v_j\} \notin E \text{ i } a_{ij} = h(\{v_i, v_j\}) \text{ ako } \{v_i, v_j\} \in E.$$




Matrica grafa prikazanog na prethodnoj slici je

$$\begin{bmatrix} 0 & 7 & 0 & 6 & 1 \\ 7 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 6 & 0 \\ 6 & 1 & 6 & 0 & 5 \\ 1 & 0 & 0 & 5 & 0 \end{bmatrix}.$$

Ovu matricu možemo predstaviti kao reč nad $\{0, 1, \#\}$ tako što težine $a_{ij}$ zapišemo u binarnom sistemu razdvajajući ih simbolom #, i označavajući kraj vrste sa ##:
0#111#0#110#1##111#0#1#1#0##0#1#0#110#0##110#1#110#0#101##1#0#0#101#0.

Matrica susedstva $M_{\mathbb{G}} = [a_{ij}]$ je očigledno simetrična, $a_{ij} = a_{ji}$, za sve $i, j$, odakle sledi da se odgovarajuća reč može skratiti (izostavljajući suvišna ponavljanja). Naime, dovoljno je navesti elemente iznad glavne dijagonale:

$$\#111\#0\#110\#1\#\#1\#1\#0\#\#110\#0\#\#101.$$

PRIMER 1.8. [**Reprezentacija iskaznih formula**] Pretpostavimo da su iskazne formule izgradjene nad iskaznim slovima $p_0, p_1, p_2, p_3, \ldots$ upotrebom veznika $\wedge$, $\vee$ i $\neg$. Ako iskazno slovo $p_i$ zapišemo kao reč $p\text{bin}(i)$, onda svaku iskaznu formulu možemo zapisati kao reč nad alfabetom

$$\Sigma_{\text{logic}} = \{p, 0, 1, (, ), \wedge, \vee, \neg\}.$$

Na primer, formuli

$$(p_3 \vee p_{11}) \wedge \neg p_7$$

odgovara reč:

$$(p11 \vee p1011) \wedge \neg p111.$$

PRIMER 1.9. [**Reprezentacija programa**] Svaki program jezika C++ jeste jedna reč nad $\Sigma_{\text{keyboard}}$.

**Definicija 1.3.** *Neka je $\Sigma$ bilo koji alfabet i neka su $u, v \in \Sigma^*$. Reč dobijena **konkatenacijom** (dopisivanjem) reči $u$ i $v$, čije su dužine $m$ i $n$ respektivno, u oznaci $uv$ ili $u \cdot v$, je reč nad alfabetom $\Sigma$ dužine $m+n$, takva da je $i$-to slovo reči $uv$ jednako $i$-tom slovu reči $u$ ako je $i \leqslant m$, odnosno $i-m$-tom slovu reči $v$ ako je $i > m$.*

Osobine operacije dopisivanja su:

- Operacija je asocijativna $(\alpha\beta)\gamma = \alpha(\beta\gamma)$
- $\varepsilon$ je neutralni element $\alpha\varepsilon = \varepsilon\alpha = \alpha$
- Nad $\Sigma^*$ definisan jedan monoid, takozvani *slobodni monoid*
- Za $\alpha \in \Sigma^*$ i $n \in \mathbb{N}$ na uobicajen način definišemo $\alpha^n$ sa $\alpha^0 = \varepsilon,\ \alpha^{n+1} = \alpha^n\alpha$
- Ako $\Sigma$ ima bar dva simbola, onda operacija dopisivanja *nije komutativna*

**Definicija 1.4.** *Neka je $\Sigma$ bilo koji alfabet i neka su $u, v \in \Sigma^*$.*

- *$u$ je **prefiks** reči $w$ ako postoji reč $v$ takva da važi $w = uv$*
- *$u$ je **sufiks** reči $w$ ako postoji reč $v$ takva da važi $w = vu$*
- *$u$ je **podreč** reči $w$ ako postoje reči $v_1$ i $v_2$, maguće i prazne, takve da važi $w = v_1uv_2$.*

## Jezik

**Definicija 1.5.** ***Jezik nad alfabetom** $\Sigma$ jeste bilo koji podskup od $\Sigma^*$.*

Trivijalni jezici nad alfabetom $\Sigma$ jesu:

- $L_\emptyset = \emptyset$ – prazan jezik;
- $L_\varepsilon = \{\varepsilon\}$ – jezik koji sadrži samo praznu reč;
- $L_{\text{full}} = \Sigma^*$ – skup svih reči nad $\Sigma$.

Naravno nama će posebno biti važni netrivijalni jezici.

PRIMER 1.10. Navodimo nekoliko netrivijalnih jezika nad $\{a, b\}$:

- $L_1 = \{\varepsilon, ab, bab\}$;

- $L_2 = \Sigma^+$;
- $L_3 = \{a, aa, aaa, aaaa, \ldots\} = \{a^n \mid n \geqslant 1\}$;
- $L_4 = \{a^p \mid p \text{ je prost broj}\}$;
- $L_5 = \{a^i b^{i+j} a^j \mid i, j \geqslant 1\}$;
- $L_6 = \Sigma^3 = \{aaa, aab, aba, baa, abb, bab, bba, bbb\}$;
- $L_7 = \{w \in \{a,b\}^* \mid |w|_a = |w|_b\}$, pri čemu $|w|_s$, za $s \in \{a,b\}$, označava broj pojavljivanja simbola $s$ u reči $w$.

Iako jezici mogu biti i konačni, u velikom broju slučajeva zanimljivi su upravo oni beskonačni.

**Definicija 1.6.** *Neka je $\Sigma$ alfabet. Na skupu $\mathscr{P}(\Sigma^*)$ svih jezika nad $\Sigma$ definišemo sledeće operacije:*

1. *skupovne:*
   - *unija jezika $L_1$ i $L_2$ jezik $L_1 \cup L_2 = \{w \in \Sigma^* \mid w \in L_1$ ili $w \in L_2\}$;*
   - *presek jezika $L_1$ i $L_2$ jezik $L_1 \cap L_2 = \{w \in \Sigma^* \mid w \in L_1$ i $w \in L_2\}$;*
   - *komplement jezika $L_1$ jezik $L_1^{\complement} = \Sigma^* \setminus L_1 = \{w \in \Sigma^* \mid w \notin L_1\}$.*
2. *nadovezivanje (proizvod): $L_1 \cdot L_2 = \{uv | u \in L_1, v \in L_2\}$*
3. *stepen: $L^0 = \{\varepsilon\}$, $L^{i+1} = L^i \cdot L$, $\quad i \in \mathbb{N}$*
4. *iteracija (Klinijeva zvazdica): $L^* = \bigcup_{n \geqslant 0} L^n = \{u_1 \ldots u_n | n \in \mathbf{N}, u_1, \ldots, u_n \in L\}$*

Dodatno, uvodimo oznaku $L^+ = \bigcup\limits_{i>0} L^i$.

PRIMER 1.11. Ako je $L_1 = \{\varepsilon, b, ab\}$ i $L_2 = \{aa, ab\}$, onda je

$$\begin{aligned} L_1 L_2 &= \{\varepsilon aa, \varepsilon ab, baa, bab, abaa, abab\} \\ &= \{aa, ab, baa, bab, abaa, abab\}. \end{aligned}$$

Izdvajamo neke očigledne jednakosti:

- $L^+ = LL^*$;
- $LL_\emptyset = L_\emptyset L = L_\emptyset = \emptyset$;
- $LL_\varepsilon = L_\varepsilon L = L$

PRIMER 1.12. $\{a\}^* = \{\varepsilon, a, aa, aaa, \ldots\} = \{a^n \mid n \in \mathbb{N}\}$;
$\{a\}^*\{b\}^* = \{a^i b^j \mid i, j \in \mathbb{N}\}$;
$(\{a\}\{b\})^* = \{ab\}^* = \{\varepsilon, ab, abab, ababab, \ldots\} = \{(ab)^i \mid i \in \mathbb{N}\}$;
$\{a,b\}^*$ - skup svih reči nad $\{a,b\}$.

**Lema 1.1.** *Neka su $L_1$, $L_2$, $L_3$ jezici nad istim alfabetom $\Sigma$. Tada važi:*

- $L_1L_2 \cup L_1L_3 = L_1(L_2 \cup L_3)$;
- $L_1(L_2 \cap L_3) \subseteq L_1L_2 \cap L_1L_3$.

PRIMER 1.13. U drugom tvrdjenju prethodne leme, inkluzija se ne može zameniti jednakošću. Posmatrajmo sledeće jezike nad $\Sigma_{\text{bool}}$:

$$L_1 = \{\varepsilon, 1\}, \quad L_2 = \{0\}, \quad L_3 = \{10\}.$$

Tada je $L_2 \cap L_3 = \emptyset$, pa je $L_1(L_2 \cap L_3) = \emptyset$. Kako je

$$L_1L_2 = \{0, 10\} \text{ i } L_1L_3 = \{10, 110\},$$

imamo da je $L_1L_2 \cap L_1L_3 = \{10\}$. Dakle, u ovom slučaju, $L_1(L_2 \cap L_3) \subsetneq L_1L_2 \cap L_1L_3$.

Na kraju ovog odeljka, navodimo nekoliko jezika koji su veoma važni sa praktičnog stanovišta.

PRIMER 1.14.
- Nad alfabetom $\Sigma_{\text{keyboard}}$ posebno je važan jezik $L_{\text{C++}}$ koji sadrži sve sintaksno korektne programe zaisane na jeziku C++.
- Nad alfabetom $\{0, 1, \#\}$ posebno je važan jezik $L_{\text{Hamilton}}$ koji sadrži sve reprezentacije usmerenih grafova koji sadrže Hamiltonov ciklus.
- Nad jezikom $\Sigma_{\text{logic}}$ posebno je važan jezik $L_{\text{taut}}$ koji sadrži sve reprezentacije iskaznih tautologija.

U vezi sa navedenim jezicima, posebno su značajni algoritmi koji rešavaju *problem pripadanja*:

Ako je $L \subset \Sigma^*$ naći algoritam koji za ulaz $w \in \Sigma^*$ daje odgovor DA ako $w \in L$,
odnosno odgovor NE ako $w \notin L$.

Pomenuti algoritam treba da odredi funkciju

$$A : \Sigma^* \to \{0,1\}, \qquad A(w) = \begin{cases} 0, & w \notin L \\ 1, & w \in L \end{cases}$$

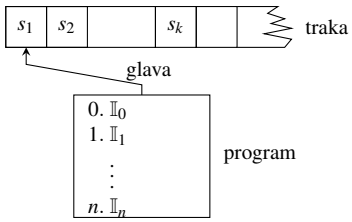
Ovom problemu će posebna pažnja biti posvećena u narednim poglavljima.

# Glava 2

# Regularni jezici

## 2.1 Konačni automati

Konačni automati reprezentuju najjednostavnije algoritme (programe) kojima se rešavaju problemi pripadanja za pojedine jezike. Da bismo slikovitije opisali konačne automate, zamislimo mašinu koja zadatu reč (unapred odredjenog alfabeta) čita simbol po simbol i pri tome izvršava veoma jednostavne instrukcije, koje ćemo uskoro uvesti. Konačan niz instrukcija $\mathbb{I}_0$, $\mathbb{I}_1$, ..., $\mathbb{I}_n$ (pri čemu je početna instrukcija numerisana brojem 0) predstavlja *program* koji je zadat mašini. Ulaznu reč zadajemo na *traci*, koja je neograničena sa jedne strane i izdeljena na ćelije, tako što ćelije popunjavamo redom simbolima koji čine zadatu reč $w = s_1 s_2 s_3 \dots s_k$. Pretpostavimo da mašina raspolaže i *glavom* koja u svakom trenutku može da pročita sadržaj samo jedne ćelije i u skladu sa pročitanim simbolom izvršava tekuću instrukciju.




Razmotrimo najpre jednostavan slučaj kada su ulazne reči zadate nad alfabetom $\Sigma_{\text{bool}} = \{0, 1\}$. U ovom slučaju, programi su konačni nizovi instrukcija oblika:

`IF` *`symbol`*`=0 THEN goto` *i* `ELSE goto` *j*

pri čemu *i* i *j* označavaju redne brojeve nekih instrukcija tog programa (uz dogovor da je 0 redni broj prve instrukcije programa). Način izvršavanja navedene instrukcije je očigledan: *ako čitaš simbol* 0*, onda predji na izvršavanje i-te instukcije programa, a u suprotnim (ako čitaš simbol* 1*) predji na izvršavanje j-te instukcije programa*. Primetimo da je navedeni opšti oblik instrukcije ekvivalentan sa `IF` *`symbol`*`=1 THEN goto` *j* `ELSE goto` *i*.

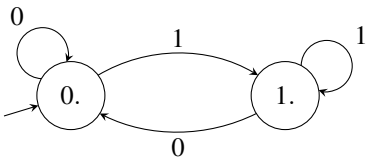
Na početku rada, glava čita sadržaj prve ćelije i izvršava instrukciju $\mathbb{I}_0$. Nakon toga, glava se pomera za jedno mesto udesno i izvršava instrukciju na koju je upućena

u prethodnom koraku. Ponovo se pomera za jedno mesto udesno i izvršava instrukciju na koju je upućena u prethodnom koraku itd. Mašina se zaustavlja kada učita sve simbole sa ulaza, tj. kada glava stigne do praznog polja.

PRIMER 2.1. Analizirajmo rad sledećeg programa.

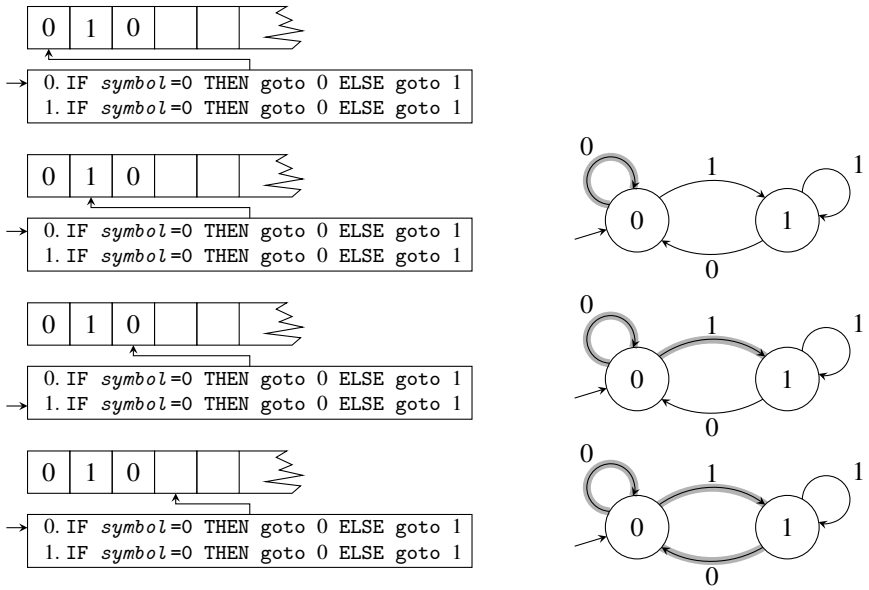
0. `IF` *symbol*`=0 THEN goto 0 ELSE goto 1`
1. `IF` *symbol*`=0 THEN goto 0 ELSE goto 1`

Zbog bolje preglednosti, dati program možemo prikazati grafom. Čvorovi predstavljaju instrukcije programa, a strelice koje povezuju čvorove (ivice) ilustruju značenje instrukcije iz koje polaze.

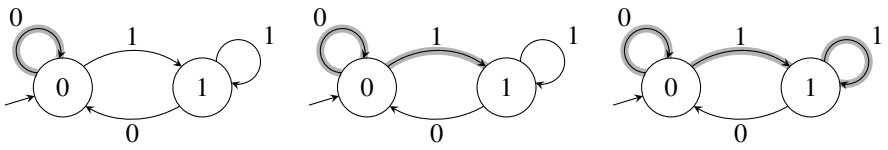



Strelice koje izlaze iz nekog čvora označene su simbolima 0 ili 1 i usmerene ka drugim čvorovima u skladu sa značenjem instrukcije koju taj polazni čvor reprezentuje. Stralica koja dolazi iz spoljašnjosti i završava se u čvoru 0. ukazuje da taj čvor predstavlja prvu instrukciju programa.

Rad mašine po zadatom programu za ulaz 010 prikazan je korak po korak na narednim slikama.




Zbog očiglenih prednosti, rad mašine za ulaz 011 prikazaćemo samo grafom.




Vidimo da se za ulaz 010 mašina zaustavila na instrukciji 0, a za ulaz 011 na instrukciji 1. Nije teško doći do sledećih opštih zaključka:

- ako se ulazna reč završava sa 0, onda se mašina zaustavlja na instrukciji 0, a
- ako se ulazna reč završava sa 1, onda se mašina zaustavlja na instrukciji 1.

Ovaj zaključak nas navodi na pomisao da zadati program posmatramo kao algoritam koji rešava problem pripadanja reči jeziku:

$$L = \{w \in \Sigma^*_{\text{bool}} \mid w \text{ se završava sa } 1\}.$$

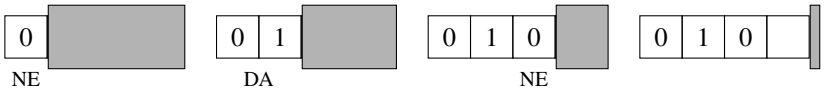
Ako redni broj instrukcije na kojoj se mašina zaustavlja posmatramo kao odgovor mašine za zadati ulaz, onda za jezik $L$

- zaustavljanje na instrukciji 0 tumačimo kao odgovor „NE" (ulaz ne pripada jeziku $L$), a
- zaustavljanje na instrukciji 1 tumačimo kao odgovor „DA" (ulaz pripada jeziku $L$).

Drugim rečima, smatramo da program izračunava vrednosti karakteristične funkcije jezika $L$, tj. funkcije $A : \Sigma^*_{\text{bool}} \to \{0,1\}$ definisane sa

$$A(w) = \begin{cases} 0, & w \in L, \\ 1, & w \notin L. \end{cases}$$

Da bismo pojasnili kako funkcioniše konstruisani automat, pokušajmo sebe da postavimo u njegovu poziciju. Zamislite da vam neko kaže da je u polja trake upisao neki niz nula i jedinica, ali vam reč prikazuje simbol po simbol i zahteva da odgovorite da li se uneta reč završava sa 1. Otkrivanjem svakog novog simbola, vi ćete u mislima pripremiti potencijalni odgovor očekujući da je simbol koji ste upravo videli i poslednji. Kako se simboli menjaju i vi ćete menjati svoj potencijalni odgovor dok god ne vidite prazno polje, kada ćete zaključiti da ste videli čitavu zapisanu reč i da je poslednji potencijalni odgovor ujedno i odgovor na postavljeno pitanje.

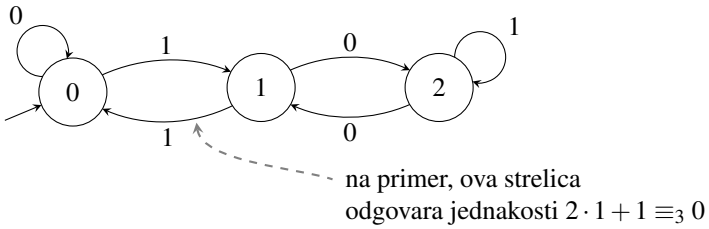


U trenucima u kojima nameravate da date odgovor „NE" smatraćemo da ste u stanju 0, a u trenucima kada je potencijalni odgovor „DA" da ste u stanju 1. Promena vašeg stanja tokom čitanja ulazne reči simbol po simbol u potpunosti odgovara promenama rednih brojeva instrukcija programa dok glava mašine čita ulaznu reč. Zato se kaže i da radni broj instrukcije koju mašina u nekom trenutku izvršava zapravo *stanje* u kome se ona u tom trenutku nalazi.

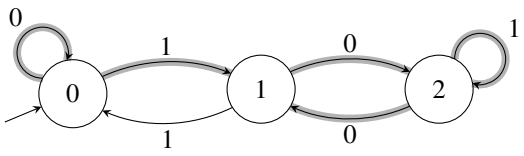
PRIMER 2.2. Napišimo program za konačni automat, koji za ulaz $w \in \Sigma^*_{\text{bool}}$ „računa" ostatak pri deljenju broja $\text{num}(w)$ sa 3. Ideju za pisanje željenog programa dobijamo ako uzmemo u obzir da ulaz čitamo simbol po simbol sleva nadesno. Zamislimo trenutak u kome smo videli prefiks $v$ ulazne reči i imamo spreman potencijalni odgovor (u slučaju da je $v$ zapravo ulazna rač). Neka je $r$ potencijalni odgovor u tom trenutku, tj. neka je $\text{num}(v) \equiv r \mod 3$, ili kraće $\text{num}(v) \equiv_3 r$. Ako u sledećem trenutku vidimo još jedan simbol, moraćemo da potražimo novi potencijalni odgovor uzimajući u obzir prethodni potencijalni odgovor ali i to koji je novi simbol koji smo uočili. Koristeći jednakost $\text{num}(vs) = 2\text{Num}(v) + s$, $s \in \{0,1\}$ i činjenicu da iz $\text{num}(v) \equiv_3 r$ sledi da je $\text{num}(vs) \equiv_3 2r + s$, jednostavno dolazimo do tabele na osnovu koje menjamo potencijalni odgovor („stanje"):

| | $r = 0$ | $r = 1$ | $r = 2$ |
|---|---|---|---|
| $s = 0$ | $2 \cdot 0 + 0 \equiv_3 0$ | $2 \cdot 1 + 0 \equiv_3 2$ | $2 \cdot 2 + 0 \equiv_3 1$ |
| $s = 1$ | $2 \cdot 0 + 1 \equiv_3 1$ | $2 \cdot 1 + 1 \equiv_3 0$ | $2 \cdot 2 + 1 \equiv_3 2$ |

Rezultate prethodne tabele možemo na jednostavan način prikazati grafom u čijim se čvorovima nalaze potencijalni odgovori, a strelice odgovaraju promenama potencijalnih odgovora prema tome koji je novi simbol učitan.




Na primer, ako je na ulazu data reč 01010, automat će se zaustaviti u čvoru 1, tj. daće odgovor 1.



Uzimajući u obzir kako je automat konstruisan, njegov odgovor je i očekivan jer je $\text{num}(01010) = 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 = 10 \equiv_3 1$.

Konstruisani graf reprezentuje sledeći program.

0. `IF` *`symbol`*`=0 THEN goto 0 ELSE goto 1`
1. `IF` *`symbol`*`=0 THEN goto 2 ELSE goto 0`
2. `IF` *`symbol`*`=0 THEN goto 1 ELSE goto 2`

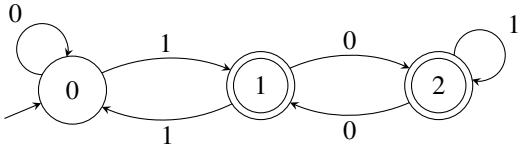
Navedeni program možemo iskoristiti za algoritamsko rešavanje problema pripadanja za nekoliko jezika. Posmatrajmo jezik

$$L = \{w \in \Sigma^*_{\text{bool}} \mid \text{num}(w) \text{ nije deljivo sa } 3\}.$$

Ako

- zaustavljanje na instrukciji 0 tumačimo kao odgovor „ulaz NE PRIPADA jeziku $L$", a
- zaustavljanje na instrukciji 1 ili na instrukciji 2 tumačimo kao odgovor „ulaz PRIPADA jeziku $L$",

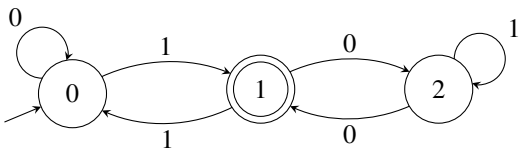
onda program zaista ispituje pripadnost ulazne reči datom jeziku. Dakle, proglašavanjem pojedinih odgovora *potvrdnim*, i samim tim preostale *odrečnim*, program smo na veoma jednostavan način prilagodili problemu pripadnosti jeziku $L$. Uobičajeno je da se potvrdni odgovori nazivaju *završnim stanjima* i da se odgovarajući čvorovi grafa ističu dvema koncentričnim kružnicama.




Na analogan način, program možemo iskoristiti i za neke druge probleme pripadanja; na primer, za problem pripadanja jeziku

$$L_1 = \{w \in \Sigma^*_{\text{bool}} \mid \text{num}(w) \equiv_3 1\}.$$

U slučaju ovog jezika, potvrdan odgovor, tj. završno stanje jeste samo 1.



Na potpuno analogan način razmatramo i opšti slučaj kada su na ulazu dozvoljene reči nad bilo kojim alfabetom $\Sigma = \{s_1, s_2, \ldots, s_k\}$. U tom slučaju za pisanje programa koristimo naredbe oblika:

```
select  symbol=s1 goto i1
        symbol=s2 goto i2
        ⋮
        symbol=sk goto ik
```

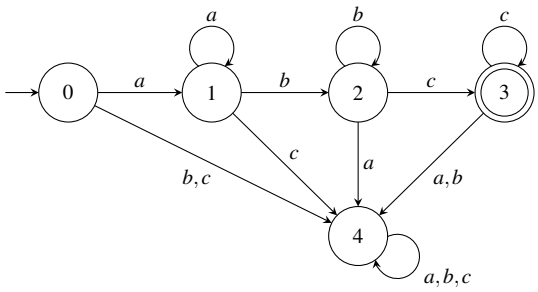
pri čemu je način izvršavanja ovakve naredbe očigledan: *ako čitaš simbol $s_j$ idi na instrukciju $i_j$*. Ako je instrukcija ovog oblika $i$-ta u programu, onda je grafički prikazujemo tako što iz čvora $i$ izlaze strelice označene sa $s_1, s_2, \ldots, s_k$ i završavaju se redom u čvorovima $i_1, i_2, \ldots, i_k$. Primetimo da je u slučaju alfabeta $\Sigma_{\text{bool}}$, naredba `IF symbol=0 THEN goto i ELSE goto j` zapravo samo kraće zapisana naredba:

```
select  symbol=0 goto i
        symbol=1 goto j.
```

PRIMER 2.3. Nad alfabetom $\Sigma_3 = \{a, b, c\}$ posmatramo jezik

$$L = \{a^k b^\ell c^m \mid k, \ell, m \geqslant 1\}.$$

Konstruišimo automat koji rešava problem pripadanja jeziku $L$.




Program kome odgovara dati graf izostavljamo.

Jednostavno je uočiti osnovne ideje kojima je rukovodjena konstrukcija ovog automata: automat može stići do završnog stanja samo ako

- najpre učita bar jedan simbol $a$ nakon čega prelazi u stanje 1 i u njemu može ostati dok god učitava simbole $a$,
- a zatim učita bar jedan simbol $b$ i prelazi u stanje 2 u kome ostaje ako ima još simbola $b$,
- i najzad kada učita simbol $c$ prelazi u završno stanje 3 u kome ostaje samo ako se ulaz završava simbolima $c$,

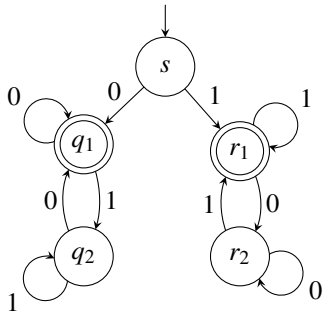
dok u ga u svim ostalim slučajevima šaljemo u takozvano *mrtvo stanje* 4 (iz koga nema izlaska).

Uopšteno govoreći, konačne automate nad alfabetom $\Sigma = \{s_1, \ldots, s_k\}$ reprezentuju konačni „težinski" usmereni grafovi:

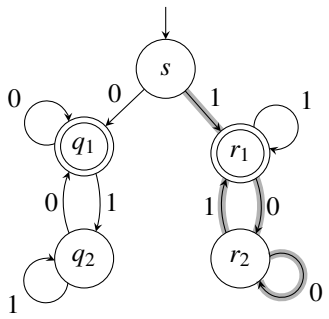
- čije čvorove nazivamo *stanjima*, i medju kojima je izdvojeno jedno *početno stanje* (na koje ukazuje strelica iz spoljašnjosti) i nekoliko *završnih stanja* (koje prikazujemo dvema koncentričnim kružnicama), pri čemu dolazi u obzir i mogućnost da nema završnih stanja, i
- iz svakog čvora (stanja) polazi tačno $k$ ivica (ka istom stanju ili nekim drugim stanjima) koje su označene simbolima $s_1, \ldots, s_k$.

Kažemo da neki konačni automat nad $\Sigma$ prihvata reč $w \in \Sigma^*$ ako se put koji polazi iz početnog stanja i odgovara reči $w$ završava u nekom od završnih stanja. Automat prihvata jezik $L \subseteq \Sigma^*$ ako prihvata sve reči iz $L$, i ne prihvata reči iz $\Sigma^* \setminus L$.

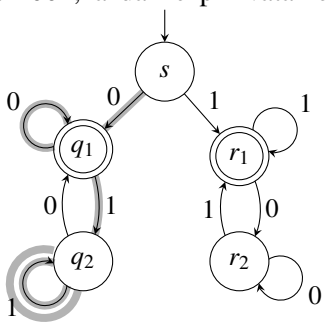
PRIMER 2.4. Posmatrajmo automat nad $\Sigma_{\text{bool}}$ prikazan na narednoj slici.




Nije teško uočiti da dati automat prihvata 1001, a da ne prihvata reč 00111.




prihvata 1001




ne prihvata 00111

Uopšte, automat prihvata reči koje počinju i završavaju se istim simbolom.

PRIMER 2.5. Konstruisati graf za automat nad $\Sigma_{\text{bool}} = \{0,1\}$ koji prihvata:

1. $L_1 = \{01, 101\}$;
2. $L_2 = \{w \in \Sigma_{\text{bool}}^* \} \mid w$ se završava sa $01\}$;
3. $L_3 = \{w \in \Sigma_{\text{bool}}^* \} \mid |w|_0$ je paran broj$\}$.

Nije teško uočiti da ima jezika $L \subseteq \Sigma^*$ za koje ne postoje konačni automati koji ih prihvataju. Kasnije ćemo mnogo detaljnije pisati o takvim jezicima. Sada navodimo samo jedan primer $L = \{0^n 1^n \mid n \geqslant 1\}$. Iako u ovom trenutku nećemo strogo dokazati da ne postoje konačni automat koji prihvata ovaj jezik, to je intuitivno jasno, jer bi takav automat morao nakon učitavanja nula da zapamti njihov broj, a znamo da konačni automati ne raspolažu nikakvom memorijom.

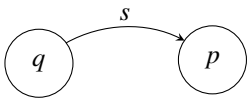
## Formalna definicija konačnog automata

Stroga definicija *konačnog automata* (skraćeno KA) jeste samo matematička formalizacija pojma koji smo uveli oslanjajući se na grafičke prikaze.

**Definicija 2.1.** *Konačni automat je uredjena petorka* $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$, *gde je*

- $Q$ *konačan skup stanja,*
- $q_0 \in Q$ *početno stanje,*
- $F \subseteq Q$ *skup završnih stanja,*
- $\Sigma$ *ulazni alfabet,*
- $\delta : Q \times \Sigma \to Q$ *funkcija tranzicije.*

Prethodnom definicijom je obuhvaćeno sve što je potrebno da bismo konstruisali neki automat: odrediti njegova stanja, precizirati koje stanje je početno, proglasiti pojedina stanja završnim, zadati alfabet čije reči mogu biti ulazi i opisati kako se prelazi iz jednog stanja u drugo što se postiže funkcijom tranzicije: jednakost $\delta(q,s) = p$ odgovara strelici iz stanja $q$ u stanje $p$ pri čemu je strelica označena sa $s$.




PRIMER 2.6. Svaki od automata koji smo do sada grafički predstavljali jednostavno se zadaje i u skladu sa prethodnom definicijom. Na primer, automat iz prethodnog

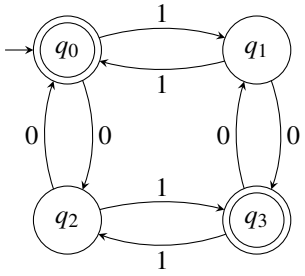
primera jeste uredjena petorka $(\{0,1,2,3,4\},0,\{3\},\{a,b,c\},\delta)$, pri čemu je $\delta:\{0,1,2,3,4\}\times\{a,b,c\}\to\{0,1,2,3,4\}$ funkcija definisana sledećom tabelom:

| $\delta$ | $a$ | $b$ | $c$ |
|---|---|---|---|
| 0 | 1 | 4 | 4 |
| 1 | 1 | 2 | 4 |
| 2 | 4 | 2 | 3 |
| 3 | 4 | 4 | 3 |
| 4 | 4 | 4 | 4 |

PRIMER 2.7. Ako je automat zadat kao uredjena petorka, jednostavno ga grafički prikazujemo. Posmatrajmo automat $\mathbb{M}=(\{q_0,q_1,q_2,q_3\},q_0,\{q_0,q_3\},\{0,1\},\delta)$, gde je $\delta$ definisano sledećom tabelom.

| $\delta$ | 0 | 1 |
|---|---|---|
| $q_0$ | $q_2$ | $q_1$ |
| $q_1$ | $q_3$ | $q_0$ |
| $q_2$ | $q_0$ | $q_3$ |
| $q_3$ | $q_1$ | $q_2$ |

Grafički prikaz automata $\mathbb{M}$ dat je na narednoj slici.




Rad datog automata za zadati ulaz, na primer za 010110, pratimo prolaskom kroz graf, polazeći iz $q_0$ i sledeći ivice označene redom simbolima ulaza (čitanog sleva nadesno). Zaključujemo da automat prihvata ulaz, jer rad završava u stanju $q_3$ koje je završno. Prolazak kroz graf možemo prikazati i na sledeći način:

$$q_0010110\vdash_{\mathbb{M}} q_210110\vdash_{\mathbb{M}} q_30110\vdash_{\mathbb{M}} q_1110\vdash_{\mathbb{M}} q_010\vdash_{\mathbb{M}} q_10\vdash_{\mathbb{M}} q_3,$$

pri čemu su momenti rada automata okarakterisani tekućim stanjem i ostatkom ulaza koji treba da se učita. Ovakve karakterizacije koraka pri radu automata $\mathbb{M}$ nazivaju se *konfiguracijama*. Konfiguracije su povezane znakom $\vdash_{\mathbb{M}}$ koji označava odgovarajuće promene konfiguracija. Svake dve susedne konfiguracije predstavljaju jedan *računski korak*. Čitav niz nazivamo *izračunavanjem* automata $\mathbb{M}$ za ulaz 010110.

Pojmove istaknute u prethodnom primeru uvodimo sledećom definicijom.

**Definicija 2.2.** *Neka* $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ *konačni automat.*

- *Konfiguracija automata* $\mathbb{M}$ *je svaki element iz* $Q \times \Sigma^*$. *Za bilo koje* $q \in Q$ *i* $w \in \Sigma^*$, *konfiguraciju* $(q,w)$ *kraće označavamo* $qw$. *Specijalno konfiguraciju* $q\varepsilon$ *označavamo* $q$.
- *Računski korak automata* $\mathbb{M}$ *jeste binarna relacija* $\vdash_{\mathbb{M}}$ *na skupu svih konfiguracija definisana na sledeći način:*

$$qw \vdash_{\mathbb{M}} pv \Leftrightarrow \text{ za neki } s \in \Sigma, w = sv \text{ i } \delta(q,s) = p.$$

- *Izračunavanje automata* $\mathbb{M}$ *jeste svaki konačan niz konfiguracija* $C_0, C_1, \ldots, C_n$, $n \geqslant 1$, *takav da je* $C_i \vdash_{\mathbb{M}} C_{i+1}$, *za svaki* $i$, $0 \leqslant i < n$.
- *Izračunavanje automata* $\mathbb{M}$ *za ulaz* $w$ *jeste svako izračunavanje* $C_0$, $C_1$, $\ldots$, $C_n$ *takvo da je* $C_0 = q_0 w$ *i* $C_n \in Q \times \{\varepsilon\}$. *Tada se konfiguracije* $C_0$ *naziva početna konfiguracija, a* $C_n$ *završna konfiguracija.*

Pojam izračunavanja automata $\mathbb{M}$ koristimo da bismo uveli takozvano refleksivno i tranzitivno zatvorenje relacije $\vdash_{\mathbb{M}}$, tj, relaciju $\vdash^*_{\mathbb{M}}$ medju konfiguracijama definisanu sa: $qw \vdash^*_{\mathbb{M}} pv$ akko

- važi $q = p$ i $w = v$, ili
- postoji izračunavanje $C_0, C_1, \ldots, C_n$ takvo da je $C_0 = qw$ i $C_n = pv$.

**Definicija 2.3.** *Neka* $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ *konačni automat.*

- *Automat* $\mathbb{M}$ *prihvata reč* $w$, *ako stanje završne konfiguracije izračunavanja automata* $\mathbb{M}$ *za ulaz* $w$ *pripada skupu (završnih stanja)* $F$. *U suprotnom, automat* $\mathbb{M}$ *ne prihvata reč* $w$.
- *Jezik koji prihvata automat* $\mathbb{M}$ *jeste skup svih reči nad* $\Sigma$ *koje prihvata automat* $\mathbb{M}$*:*

$$L(\mathbb{M}) = \{w \in \Sigma^* \mid \mathbb{M} \text{ prihvata reč } w\}.$$

Jezik koji prihvata automat $\mathbb{M}$ možemo opisati i relacijom $\vdash^*_{\mathbb{M}}$:

$$L(\mathbb{M}) = \{w \in \Sigma^* \mid q_0 w \vdash^*_{\mathbb{M}} p \text{ za neko } p \in F\}.$$

Veoma je korisno definisati i svojevrsno **zatvorenje funkcije tranzicije**. Neformalno, za $q \in Q$ i $w \in \Sigma^*$, $\hat{\delta}(q,w)$ je stanje u koje se stiže nakon učitavanja reči $w$ polazeći iz stanja $q$, tj.

$$\hat{\delta}(q,w) = p \text{ je ekvivalentno sa } qw \vdash^*_{\mathbb{M}} p.$$

Specijalno je $\hat{\delta}(q,s) = \delta(q,s)$, za bilo koje $s \in \Sigma$.

**Definicija 2.4.** *Neka je $(Q, q_0, F, \Sigma, \delta)$ neki konačni automat. Zatvorenje funkcije $\delta : Q \times \Sigma \to Q$ jeste funkcija $\hat{\delta} : Q \times \Sigma^* \to Q$ data sledećim jednakostima:*

- $\hat{\delta}(q, \varepsilon) = q$, $q \in Q$,
- $\hat{\delta}(q, ws) = \delta(\hat{\delta}(q, w), s)$, $q \in Q$, $w \in \Sigma^*$, $s \in \Sigma$.

Jezik koji prihvata automat $\mathbb{M}$ ima i sledeći opis:

$$L(\mathbb{M}) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F\}.$$

**Lema 2.1.** *Neka je $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ konačni automat. Ako su $u, v \in \Sigma^*$ reči takve da je $\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v)$, tada za svaku reč $w \in \Sigma^*$ važi:*

$$uw \in L(\mathbb{M}) \Leftrightarrow vw \in L(\mathbb{M}).$$

Dokaz leme je prepušten čitaocu.

U narednom primeru ilustrovaćemo jedan koristan metod koji se koristi prilikom odredjivanja jezika koji prihvata neki konačni automat. Pre toga ističemo nekoliko opštih zapažanja. Svaki konačni automat $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ razbija skup svih $\Sigma^*$ na $|Q|$ skupova[1)] (koji su neprazni, disjunktni i čija je unija jednaka $\Sigma^*$):

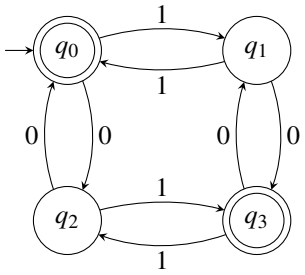
$$C[q] = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) = q\}, q \in Q.$$

Ovo razbijanje skupa $\Sigma^*$ indukovano je relacijom ekvivalencije $\sim_\delta$ datom sa:

$$w \sim_\delta v \Leftrightarrow \hat{\delta}(q_0, w) = \hat{\delta}(q_0, v),$$

čije su klase upravo skupovi $C[q]$, $q \in Q$. Pošto se izračunavanje automata $\mathbb{M}$ za bilo koji ulaz $w \in \Sigma^*$ mora završiti u jednom od stanja iz $Q$, za $q \in Q$ skup $C[q]$ predstavlja zapravo skup svih reči iz $\Sigma^*$ za koje se pomenuto izračunavanje završava u stanju $q$. Očigledno je da kada odredimo skupove $C[q]$, $q \in Q$, odredili smo i jezik $L(\mathbb{M})$, jer je $L(\mathbb{M}) = \bigcup_{q \in F} C[q]$.

PRIMER 2.8. Odredimo jezik koji prihvata konačni automat iz primera 2.7.




[1)] $|Q|$ je broj elemenata skupa $Q$.

Kako automat ima četiri stanja, odredićemo najpre skupove $C[q_0], C[q_1], C[q_2], C[q_3]$. Označimo sa $|w|_s$ broj simbola $s$ u reči $w$, $s \in \{0,1\}$. Analizirajući dati automat uočavamo da bi ovi skupovi mogli biti:

$$\begin{aligned}
C[q_0] &\stackrel{?}{=} \{w \in \Sigma^*_{\text{bool}} \mid |w|_0 \text{ je paran broj i } |w|_1 \text{ je paran broj}\}\\
C[q_1] &\stackrel{?}{=} \{w \in \Sigma^*_{\text{bool}} \mid |w|_0 \text{ je paran broj i } |w|_1 \text{ je neparan broj}\}\\
C[q_2] &\stackrel{?}{=} \{w \in \Sigma^*_{\text{bool}} \mid |w|_0 \text{ je neparan broj i } |w|_1 \text{ je paran broj}\}\\
C[q_3] &\stackrel{?}{=} \{w \in \Sigma^*_{\text{bool}} \mid |w|_0 \text{ je neparan broj i } |w|_1 \text{ je neparan broj}\}.
\end{aligned}$$

Da su navedene pretpostavke tačne moramo i strogo dokazati. Dokaz sprovodimo indukcijom po dužini reči $w \in \Sigma^*_{\text{bool}}$.

*Baza indukcije.* Dokazaćemo da su jednakosti tačne za sve reči dužine 0, 1 ili 2. Ove činjenice direktno proveravamo.

$$\begin{aligned}
&\hat{\delta}(q_0,\varepsilon) = q_0 \Rightarrow \varepsilon \in C[q_0]\\
&\hat{\delta}(q_0,0) = q_2 \Rightarrow 0 \in C[q_2] \qquad \hat{\delta}(q_0,1) = q_1 \Rightarrow 1 \in C[q_1]\\
&\hat{\delta}(q_0,00) = q_0 \Rightarrow 00 \in C[q_0] \quad \hat{\delta}(q_0,01) = q_3 \Rightarrow 01 \in C[q_3]\\
&\hat{\delta}(q_0,10) = q_3 \Rightarrow 10 \in C[q_3] \quad \hat{\delta}(q_0,11) = q_0 \Rightarrow 11 \in C[q_0]
\end{aligned}$$

*Induktivni korak.* Pretpostavimo da jednakosti koje dokazujemo važe za sve reči čija je dužina $n$, gde je $n \geqslant 2$. Neka je $w \in \Sigma^*_{\text{bool}}$ takva da je $|w| = n+1$. Tada je $w = vs$, gde je $v \in \Sigma^*_{\text{bool}}$, $|v| = n$ i $s \in \Sigma_{\text{bool}}$. Razlikujemo četiri slučaja u zavisnosti od parnosti brojeva $|v|_0$ i $|v|_1$:

1. $|v|_0$ i $|v|_1$ su parni brojevi;
2. $|v|_0$ je paran, a $|v|_1$ neparan;
3. $|v|_0$ je neparan, a $|v|_1$ paran;
4. $|v|_0$ i $|v|_1$ su neparni brojevi.

Razmotrićemo samo prva dva slučaja, dok ostala dva izostavljamo, jer se analogno postupa.

1. Ako su $|v|_0$ i $|v|_1$ parni brojevi, onda prema induktivnoj pretpostavci $v \in C[q_0]$, tj. $\hat{\delta}(q_0,v) = q_0$. Ukoliko je $s = 0$, imamo da je

   $$\hat{\delta}(q_0,w) = \hat{\delta}(q_0,v0) = \delta(\hat{\delta}(q_0,v),0) = \delta(q_0,0) = q_2,$$

   tj. $w = v0 \in C[q_2]$, što je i trebalo pokazati. Ako je $s = 1$, iz

   $$\hat{\delta}(q_0,w) = \hat{\delta}(q_0,v1) = \delta(\hat{\delta}(q_0,v),1) = \delta(q_0,1) = q_1,$$

   sledi da $w = v1 \in C[q_1]$, što je i trebalo pokazati.

2. Ako je $|v|_0$ paran, a $|v|_1$ neparan, onda prema induktivnoj pretpostavci $v \in C[q_1]$, tj. da je $\hat{\delta}(q_0, v) = q_1$. Tada iz jednakosti:

$$\hat{\delta}(q_0, w) = \hat{\delta}(q_0, vs) = \delta(\hat{\delta}(q_0, v), s) = \delta(q_1, s) = \begin{cases} q_3, & s = 0, \\ q_0, & s = 1, \end{cases}$$

sledi da je $v0 \in C[q_3]$ i $v1 \in C[q_0]$.

Dakle, zaključujemo da je:

$$\begin{aligned} L(\mathbb{M}) &= C[q_0] \cup C[q_3] \\ &= \{w \in \Sigma^*_{\text{bool}} \mid |w|_0 \text{ i } |w|_1 \text{ su oba parni ili su oba neparni brojevi}\} \\ &= \{w \in \{0,1\}^* \mid |w|_0 + |w|_1 \equiv_2 0\} \end{aligned}$$

Iako se često izostavljaju formalni dokazi poput onog iz prethodnog primera, osnovne ideje u navedenom dokazu mogu biti veoma korisne prilikom konstrukcije nekog automata. Naime, dobra strategija za konstrukciju željenog automata zasnovana je na particiji skupa $\Sigma^*$ svih ulaza na podklase u odnosu na neka (pogodno izabrana za odgovarajući kontekst) svojstva reči, i odredjivanju tranzicija izmedju klasa u skladu sa dopisivinjem simbola iz $\Sigma$ rečima iz klasa.

## 2.2 NKA

Nedeterministički konačni automati (skraćeno NKA) predstavljaju uopštenje konačnih automata o kojima je bilo reči u prethodnom odeljku. Pored naredbi oblika:

```
select  symbol=s1 goto i1
        symbol=s2 goto i2
        ⋮
        symbol=sk goto ik
```

nedeterministički automati izvršavaju i naredbe u kojima su podvučene instrukcije zamenjene sa:

$$\texttt{choose goto } i_{m_1} \texttt{ or goto } i_{m_2} \texttt{ or } \cdots \texttt{ or goto } i_{m_j}$$

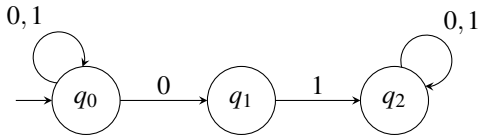
ili sa `STOP`. Drugim rečima, NKA nakon učitavanja simbola nastavlja sa radom nakon izbora jedne od više ponudjenih mogućnosti $(i_{m_1}, i_{m_2}, \ldots, i_{m_j})$ ili se zaustavlja (`STOP`).

Pre nego što ih strogo definišemo, navodimo jedan primer koji se odnosi na jezik $\Sigma_{\text{bool}}$, pa koristimo sažetije zapise naredbi.

PRIMER 2.9. Program:

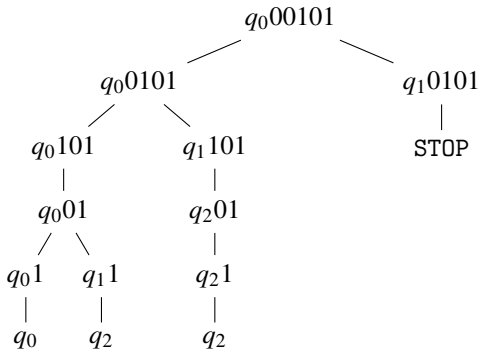
$q_0$ `IF symbol=0 THEN choose goto` $q_0$ `or goto` $q_1$ `ELSE goto` $q_0$
$q_1$ `IF symbol=0 THEN STOP ELSE goto` $q_2$
$q_2$ `IF symbol=0 THEN goto` $q_2$ `ELSE goto` $q_2$

grafički je prikazan na slici ispod.

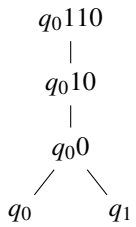



Nedeterminizam ovog programa se uočava u instrukciji $q_0$ budući da se ova instrukcija učitavanjem simbola 0 izvršava tako što se najpre izabere da li će naredna biti $q_0$ ili $q_1$. Pored toga, razlika u odnosu na konačne automate se ogleda i u tome što instrukcija $q_1$ zaustavlja izvršavanje programa (rad automata) ukoliko se učita simbol 0.

Za zadati ulaz, nedeterministički program se može izvršiti na više načina. Ako je ulaz 00101, sva moguća izvršavanja datog programa prikazuje naredno drvo.




Dakle, za ulaz 00101 automat može dati odgovor $q_0$ ili $q_2$ ili da se zaustavi pre nego što učita čitav izlaz. Ako kao i kod konačnih automata, pojedine odgovore (stanja) proglasimo *potvrdnim* (završinim stanjima), onda kažemo da automat prihvata dati ulaz ukoliko postoji bar jedno izračunavanje u kome daje potvrdan odgovor. Na primer, ako samo $q_2$ proglasimo završinim stanjem (potvrdnim odgovorom), dati automat će prihvatiti ulaz 00101. Nasuprot tome, ulaz 110 neće prihvatiti jer se nijedno izvršavanje programa za ovaj ulaz ne završava odgovorom $q_2$.

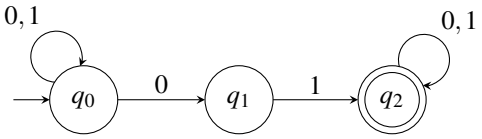



Formalna definicija nedeterminističkog automata razlikuje se od definicije konačnog automata samo po tome kako se uvodi funkcija tranzicije.

**Definicija 2.5.** *Nedeterministički konačni automat je uredjena petorka* $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$, *gde je*

- $Q$ *konačan skup stanja,*
- $q_0 \in Q$ *početno stanje,*
- $F \subseteq Q$ *skup završnih stanja,*
- $\Sigma$ *ulazni alfabet,*
- $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ *funkcija tranzicije, pri čemu je* $\mathcal{P}(Q)$ *skup svih podskupova od* $Q$.

PRIMER 2.10. Automat iz prethodnog primera,




za koji je $q_2$ izabrano za završno stanje, formalno se opisuje kao uredjena petorka $(\{q_0, q_1, q_2\}, q_0, \{q_2\}, \{0,1\}, \delta)$, gde je $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ definisano sa:

| | 0 | 1 |
|---|---|---|
| $q_0$ | $\{q_0, q_1\}$ | $\{q_0\}$ |
| $q_1$ | $\emptyset$ | $\{q_2\}$ |
| $q_2$ | $\{q_2\}$ | $\{q_2\}$ |

**Definicija 2.6.** *Neka* $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ *nedeterministički konačni automat.*

- *Računski korak automata* $\mathbb{M}$ *jeste binarna relacija* $\vdash_{\mathbb{M}}$ *na skupu svih konfiguracija* $Q \times \Sigma^*$ *definisana na sledeći način:*

$$qw \vdash_{\mathbb{M}} pv \Leftrightarrow \text{ za neki } s \in \Sigma, w = sv \text{ i } p \in \delta(q, s).$$

- *Izračunavanje automata* $\mathbb{M}$ *je svaki konačan niz konfiguracija* $C_0, C_1, \ldots, C_n$, $n \geqslant 1$, *takav da je* $C_i \vdash_{\mathbb{M}} C_{i+1}$, *za svaki* $i$, $0 \leqslant i < n$.
- *Izračunavanje automata* $\mathbb{M}$ *za ulaz* $w$ *jeste svako izračunavanje* $C_0, C_1, \ldots, C_n$ *takvo da je* $C_0 = q_0 w$ *i*

  *i)* $C_n \in Q \times \{\varepsilon\}$ *ili*

  *ii)* $C_n = qsv$, *za neke* $q \in Q$, $s \in \Sigma$, $v \in \Sigma^*$, *pri čemu je* $\delta(q, s) = \emptyset$ *i* $sv$ *je sufiks ulazne reči* $w$.

*Tada se konfiguracija $C_0$ naziva početna konfiguracija, a $C_n$ završna konfiguracija. (Dakle, izračunavanje nedeterminističkog automata $\mathbb{M}$ za $w$ može se zaustaviti ako je ceo ulaz pročitan ili (za razliku od konačnih automata) ako ne postoji mogućnost da se izračunavanje nastavi za argument $(q,s)$.)*

Kao i u slučaju konačnih automata, pojam izračunavanja automata $\mathbb{M}$ koristimo da bismo uveli takozvano refleksivno i tranzitivno zatvorenje relacije $\vdash_{\mathbb{M}}$, tj, relaciju $\vdash^*_{\mathbb{M}}$ medju konfiguracijama definisanu sa: $qw \vdash^*_{\mathbb{M}} pv$ akko

- važi $q = p$ i $w = v$, ili
- postoji izračunavanje $C_0, C_1, \ldots, C_n$ takvo da je $C_0 = qw$ i $C_n = pv$.

**Definicija 2.7.** *Neka $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ nedeterministički konačni automat.*

- *Automat $\mathbb{M}$ prihvata reč $w$, ako postoji izračunavanje $C_0, C_1, \ldots, C_n$ automata $\mathbb{M}$ za ulaz $w$, takvo da je $C_n \in F \times \{\varepsilon\}$, odnosno ako postoji $p \in F$ tako da $q_0 w \vdash^*_{\mathbb{M}} p$. U suprotnom, automat $\mathbb{M}$ ne prihvata reč $w$.*
- *Jezik koji prihvata automat $\mathbb{M}$ jeste skup svih reči nad $\Sigma$ koje prihvata automat $\mathbb{M}$:*

$$\begin{aligned} L(\mathbb{M}) &= \{w \in \Sigma^* \mid \mathbb{M} \text{ prihvata reč } w\} \\ &= \{w \in \Sigma^* \mid q_0 w \vdash^*_{\mathbb{M}} p \text{ za neko } p \in F\}. \end{aligned}$$

Primetimo da, u skladu sa prethodnom definicijom, NKA prihvata ulaz jedino u slučaju da je čitav ulaz pročitan i izračunavanje se zaustavlja u završnom stanju.

Definišimo i zatvorenje funkcije tranzicije.

**Definicija 2.8.** *Neka je $(Q, q_0, F, \Sigma, \delta)$ neki nedeterministički konačni automat. Zatvorenje funkcije $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ jeste funkcija $\hat{\delta} : Q \times \Sigma^* \to \mathcal{P}(Q)$ data sledećim jednakostima:*

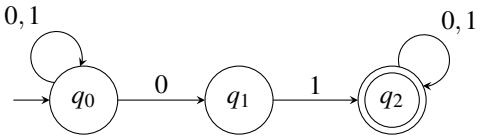
- $\hat{\delta}(q, \varepsilon) = \{q\}$, $q \in Q$,
- *za $q \in Q$, $w \in \Sigma^*$, $s \in \Sigma$:*

$$\begin{aligned} \hat{\delta}(q, ws) &= \{p \in Q \mid \text{ postoji } r \in \hat{\delta}(q,w) \text{ takvo da } p \in \delta(r,s)\} \\ &= \bigcup_{r \in \hat{\delta}(q,w)} \delta(r,s). \end{aligned}$$

Primetimo da je $\hat{\delta}(q,w)$ skup svih stanja u koja se može stići polazeći iz $q$ nakon učitavanja reči $w$. Naravno, ako se izračunavanje automata $\mathbb{M}$ za ulaz $w$ uvek zaustavlja pre učitavanja ulaza, onda je $\hat{\delta}(q_0, w) = \emptyset$, a u tom slučaju je $\hat{\delta}(q_0, wu) = \emptyset$, za bilo koju reč $u \in \Sigma^*$. Jezik koji prihvata neki NKA opisujemo i na sledeći način:

$$L(\mathbb{M}) = \{w \in \Sigma^* \mid \hat{\delta}(q,w) \cap F \neq \emptyset\}.$$

PRIMER 2.11. Nije teško odrediti jezik koji odredjuje NKA iz prethodnog primera.




Naime, NKA može stići u završno stanje $q_2$ samo ako ulaz sadrži podreč 01, dok u suprotnom ne može dostići ovo stanje. Ako sa $\mathbb{M}$ označimo dati NKA, dokazaćemo da je

$$L(\mathbb{M}) = \{w \in \Sigma^*_{\text{bool}} \mid w \text{ sadrži podreč } 01\}.$$

Najpre dokazujemo da je $L(\mathbb{M}) \supseteq \{w \in \Sigma^*_{\text{bool}} \mid w \text{ sadrži podreč } 01\}$.

Neka $w \in \{w \in \Sigma^*_{\text{bool}} \mid w \text{ sadrži podreč } 01\}$, tj. $w = x01y$, za neke $x, y \in \Sigma^*_{\text{bool}}$. Dokazaćemo da postoji izračunavanje automata $\mathbb{M}$ kojim se prihvata reč $w$. Pošto $q_0 \in \delta(q_0,0) \cap \delta(q_0,1)$, za svako $x \in \Sigma^*_{\text{bool}}$:

$$q_0 x \vdash^*_{\mathbb{M}} q_0.$$

Pošto $q_2 \in \delta(q_2,0) \cap \delta(q_2,1)$, za svako $y \in \Sigma^*_{\text{bool}}$:

$$q_2 y \vdash^*_{\mathbb{M}} q_2.$$

Dakle,

$$q_0 x01y \vdash^*_{\mathbb{M}} q_0 01y \vdash_{\mathbb{M}} q_1 1y \vdash_{\mathbb{M}} q_2 y \vdash^*_{\mathbb{M}} q_2.$$

Dalje dokazujemo da je $L(\mathbb{M}) \subseteq \{w \in \Sigma^*_{\text{bool}} \mid w \text{ sadrži podreč } 01\}$.

Neka $w \in L(\mathbb{M})$. Dakle, postoji izračunavanje automata $\mathbb{M}$ kojim se prihvata reč $w$. To izračunavanje započinje u stanju $q_0$, a završava se u (u jedinom) završnom stanju $q_2$. Svaki put iz $q_0$ u stanje $q_2$ mora proći kroz $q_1$, pa je pomenuto izračunavanje sledeég oblika:

$$q_0 w \vdash^*_{\mathbb{M}} q_1 z \vdash^*_{\mathbb{M}} q_2.$$

Svako izračunavanje automata $\mathbb{M}$ može da sadrži najviše jednu konfiguraciju u stanju $q_1$, jer:

- $q_1 \notin \delta(q_1, s)$, za svako $s \in \Sigma_{\text{bool}}$, i
- ako $\mathbb{M}$ napusti $q_1$, onda se više ne može vratiti u $q_1$.

Zato izračunavanje prihvatanja reči $w$ mora biti oblika:

$$q_0 w \vdash^*_{\mathbb{M}} q_0 abu \vdash_{\mathbb{M}} q_1 bu \vdash_{\mathbb{M}} q_2 u \vdash^*_{\mathbb{M}} q_2,$$

pri čemu $a, b \in \Sigma_{\text{bool}}$ i $u \in \Sigma^*_{\text{bool}}$. Jedini način da se dostigne $q_1$ jeste primena tranzicije $q_1 \in \delta(q_0,0)$, tj. čitanjem simbola 0 u stanju $q_0$. Dakle, $a = 0$. Kako je $\delta(q_1,0) = \emptyset$

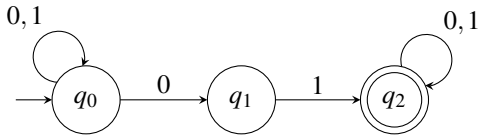
i $\delta(q_1, 1) = \{q_2\}$, jedina mogućnost da se izvrši računski korak iz $q_1$ jeste da se učita 1, pa je $b = 1$. Dakle, izračunavanje prihvatanja reči $w$ ima sledeći oblik:

$$q_0 w \vdash^*_{\mathbb{M}} q_0 01u \vdash_{\mathbb{M}} q_1 1u \vdash_{\mathbb{M}} q_2 u \vdash^*_{\mathbb{M}} q_2,$$

što znači da $w$ mora da sadrži 01 kao podreč.

Iako se na prvi pogled čini da su NKA „moćniji" od KA, nije tako jer se ispostavlja da se svaki nedeterministički automat može simulirati nekim konačnim automatom. Pre nego što to i formalno pokažemo, u narednom primeru ćemo ilustrovati ideje na kojima se zasniva dokaz.

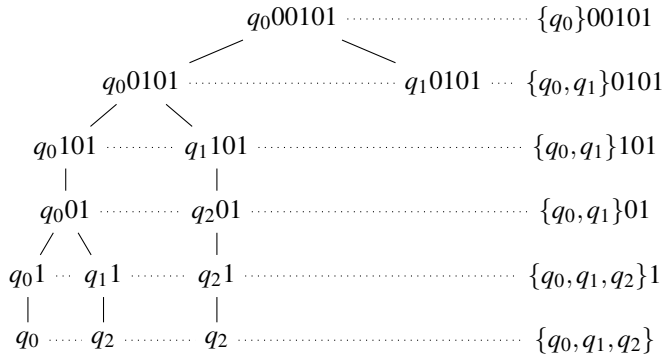
PRIMER 2.12. I ovoga puta ćemo razmatrati NKA iz prethodnih primera:




Funkcija tranzicije $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ data je sledećom tabelom.

| | 0 | 1 |
|---|---|---|
| $q_0$ | $\{q_0, q_1\}$ | $\{q_0\}$ |
| $q_1$ | $\emptyset$ | $\{q_2\}$ |
| $q_2$ | $\{q_2\}$ | $\{q_2\}$ |

Izračunavanja ovog automata za ulaz 00101 prikazujemo narednim drvetom.




Isprekidanim linijama su povezane konfiguracije koje su na istom nivou drveta, odnosno koje se mogu dostići nakon učitavanja istog prefiksa. Konfiguracije na istom nivou mogu imati različita stanja, ali je isti ostatak ulaza (sufiks) koji tek treba da bude učitan. Ovo znači da se svaki nivo može opisati nekim skupom stanja i odgovarajućim sufiksom ulaza kao što je prikazano na slici iznad. Ističemo nekoliko opštih zapažanja:

- koren drveta odgovara početnoj konfiguraciji $\{q_0\}w$,
- ako je $\{q_{i_1},\ldots,q_{i_k}\}sv$ opis jednog nivoa, naredni nivo je opisan sa

$$(\delta(q_{i_1},s)\cup\cdots\cup\delta(q_{i_k},s))v.$$

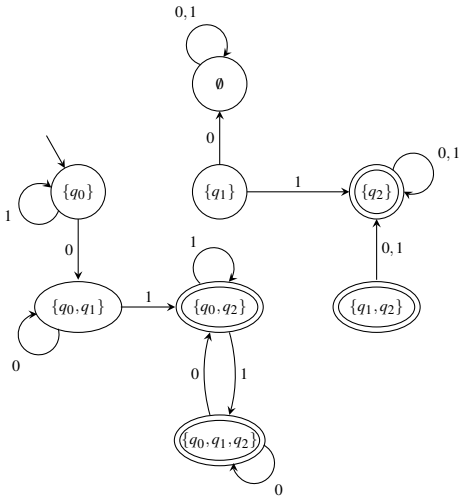
Ova zapažanja nas navode na pomisao da skupove stanja možemo smatrati stanjima novog konačnog automata. Posmatrajmo konačni automat definisan sa: $\mathbb{A} = (\mathcal{P}(Q),\{q_0\},\mathcal{F},\Sigma,\Delta)$ pri čemu je $\Delta:\mathcal{P}(Q)\times\Sigma\to\mathcal{P}(Q)$ definisana sa

$$\Delta(P,s)=\bigcup_{p\in P}\delta(p,s), P\in\mathcal{P}(Q), s\in\Sigma,$$

a skup završnih stanja je $\mathcal{F}=\{P\in\mathcal{P}(Q)\mid F\cap P\neq\emptyset\}$. U narednoj tabeli u potpunosti je opisana funkcija $\Delta$.

| | 0 | 1 |
|---|---|---|
| $\emptyset$ | $\emptyset$ | $\emptyset$ |
| $\{q_0\}$ | $\{q_0,q_1\}$ | $\{q_0\}$ |
| $\{q_1\}$ | $\emptyset$ | $\{q_2\}$ |
| $\{q_2\}$ | $\{q_2\}$ | $\{q_2\}$ |
| $\{q_0,q_1\}$ | $\{q_0,q_1\}$ | $\{q_0,q_2\}$ |
| $\{q_0,q_2\}$ | $\{q_0,q_1,q_2\}$ | $\{q_0,q_2\}$ |
| $\{q_1,q_2\}$ | $\{q_2\}$ | $\{q_2\}$ |
| $\{q_0,q_1,q_2\}$ | $\{q_0,q_1,q_2\}$ | $\{q_0,q_2\}$ |

Grafički prikaz ovog automata dat je na narednoj slici.




Primetimo da se dobijeni konačni automat uz jednostavne modifikacije može pojednostaviti odstranjivanjem nepotrebnih stanja.

**Teorema 2.1.** *Za svaki nedeterministički konačni automat* $\mathbb{M}$ *postoji konačni automat* $\mathbb{A}$ *takav da je* $L(\mathbb{M}) = L(\mathbb{A})$.

DOKAZ. Neka je $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ neki nedeterministički konačni automat. Dokazaćemo da je traženi konačni automat $\mathbb{A} = (\mathcal{P}(Q), \{q_0\}, \mathcal{F}, \Sigma, \Delta)$, pri čemu je $\Delta : \mathcal{P}(Q) \times \Sigma \to \mathcal{P}(Q)$ definisano sa

$$\Delta(P, s) = \bigcup_{p \in P} \delta(p, s), P \in \mathcal{P}(Q), s \in \Sigma,$$

i $\mathcal{F} = \{P \in \mathcal{P}(Q) \mid P \cap F \neq \emptyset\}$. Da bismo to dokazali, dovoljno je dokazati da za svaku reč $w$ važi:

$$(\star) \qquad \hat{\Delta}(\{q_0\}, w) = \hat{\delta}(q_0, w).$$

Odavde sledi željena ekvivalencija:

$$\hat{\Delta}(\{q_0\}, w) \in \mathcal{F} \Leftrightarrow \hat{\delta}(q_0, w) \cap F \neq \emptyset,$$

tj. $\mathbb{A}$ prihvata reč $w$ akko $\mathbb{M}$ prihvata $w$.

Dokaz ekvivalencije $(\star)$ izvodimo indukcijom po dužini reči $w$.

*Baza indukcije.* Ako je $|w| = 0$, onda je $w = \varepsilon$ i jednakost $(\star)$ očigledno važi jer je $\hat{\Delta}(\{q_0\}, \varepsilon) = \{q_0\} = \hat{\delta}(q_0, \varepsilon)$.

*Induktivni korak.* Pretpostavimo da je jednakost $(\star)$ tačna za sve reči dužine $n$, $n \geqslant 0$. Neka je $|w| = n + 1$. Tada je $w = vs$, gde je $|v| = n$ i $s \in \Sigma$. Tada je

$$\begin{array}{rcll} \hat{\Delta}(\{q_0\}, vs) & = & \Delta(\hat{\Delta}(\{q_0\}, v), s) & [\text{def. zatvorenja } \hat{\Delta}] \\ & = & \Delta(\hat{\delta}(q_0, v), s) & [\text{IP}] \\ & = & \bigcup\limits_{p \in \hat{\delta}(q_0, v)} \delta(p, s) & [\text{def. funkcije } \Delta] \\ & = & \hat{\delta}(q_0, vs) & [\text{def. zatvorenja } \hat{\delta}], \end{array}$$

što je i trebalo dokazati. □

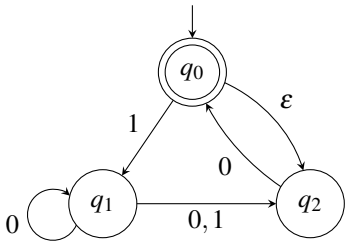
## 2.3 $\varepsilon$-NKA

Nedeterministički konačni automati sa $\varepsilon$-prelazom (skraćeno $\varepsilon$-NKA) predstavljaju uopštenje nedeterminističkih konačnih automata jer je prilikom sastavljanja programa u naredbama oblika

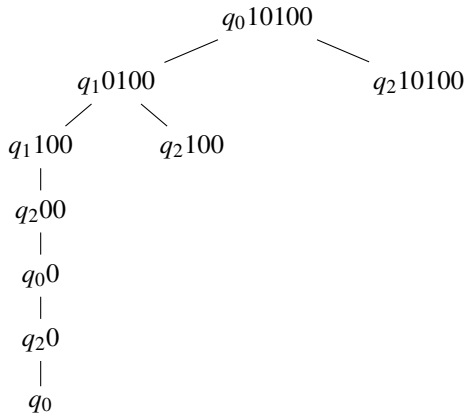
```
select  ⋮
        symbol=s_i choose goto i_{m_1} or ··· or goto i_{m_j} (ili STOP)
        ⋮
```

dozvoljeno izostaviti deo `symbol=`$s_i$. Drugim rečima, $\varepsilon$-NKA može promeniti stanje bez učitavnja simbola.

PRIMER 2.13. Jedan $\varepsilon$-NKA grafički je prikazan na sledećoj slici.




Izračunavanja do kojih može doći za ulaz 10100 prikazana su sledećim drvetom.




Definicija nedeterminističkog konačnog automata sa $\varepsilon$-prelazom razlikuje se od definicije NKA samo po tome što je domen funkcije tranzicije širi.

**Definicija 2.9.** *Nedeterministički konačni automat sa $\varepsilon$-prelazom je uredjena petorka $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$, gde je*

- *$Q$ konačan skup stanja,*
- *$q_0 \in Q$ početno stanje,*
- *$F \subseteq Q$ skup završnih stanja,*
- *$\Sigma$ ulazni alfabet,*
- *$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ funkcija tranzicije.*

**Definicija 2.10.** *Neka $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ nedeterministički konačni automat sa $\varepsilon$-prelazom.*

- *Računski korak automata* $\mathbb{M}$ *jeste binarna relacija* $\vdash_{\mathbb{M}}$ *na skupu svih konfiguracija* $Q \times \Sigma^*$ *definisana na sledeći način:*

$$qw \vdash_{\mathbb{M}} pv \Leftrightarrow \text{ za neki } s \in \Sigma \cup \{\varepsilon\}, w = sv \text{ i } p \in \delta(q,s).$$

- *Izračunavanje automata* $\mathbb{M}$ *je svaki konačan niz konfiguracija* $C_0, C_1, \ldots, C_n$, $n \geqslant 1$, *takav da je* $C_i \vdash_{\mathbb{M}} C_{i+1}$, *za svaki* $i$, $0 \leqslant i < n$.

- *Izračunavanje automata* $\mathbb{M}$ *za ulaz* $w$ *jeste svako izračunavanje* $C_0, C_1, \ldots, C_n$ *takvo da je* $C_0 = q_0 w$ *i*

  *i)* $C_n \in Q \times \{\varepsilon\}$ *ili*

  *ii)* $C_n = qsv$, *za neke* $q \in Q$, $s \in \Sigma \cup \{\varepsilon\}$, $v \in \Sigma^*$ *takve da* $\delta(q,s) = \emptyset$.

  *Tada se konfiguracije* $C_0$ *naziva početna konfiguracija, a* $C_n$ *završna konfiguracija.*

Na uobičajen način definišemo refleksivno i tranzitivno zatvorenje relacije $\vdash_{\mathbb{M}}$. Prihvatanje neke reči od strane $\varepsilon$-NKA, kao i jezik koji ovakav automat prihvata definišemo kao u slučaju NKA.

**Definicija 2.11.** *Neka* $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ *nedeterministički konačni automat sa* $\varepsilon$*-prelazom.*

- *Automat* $\mathbb{M}$ *prihvata reč* $w$, *ako postoji izračunavanje* $C_0, C_1, \ldots, C_n$ *automata* $\mathbb{M}$ *za ulaz* $w$, *takvo da je* $C_n \in F \times \{\varepsilon\}$, *odnosno ako postoji* $p \in F$ *tako da* $q_0 w \vdash^*_{\mathbb{M}} p$. *U suprotnom, automat* $\mathbb{M}$ *ne prihvata reč* $w$.

- *Jezik koji prihvata automat* $\mathbb{M}$ *jeste skup svih reči nad* $\Sigma$ *koje prihvata automat* $\mathbb{M}$*:*

$$\begin{aligned} L(\mathbb{M}) &= \{w \in \Sigma^* \mid \mathbb{M} \text{ prihvata reč } w\} \\ &= \{w \in \Sigma^* \mid q_0 w \vdash^*_{\mathbb{M}} p \text{ za neko } p \in F\}. \end{aligned}$$

Za svaki $\varepsilon$-NKA $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$, definišemo $\varepsilon$-zatvorenje podskupova od $Q$: ako je $P \subseteq Q$, $\varepsilon$-zatvorenje skupa $P$, u oznaci $\overline{P}$ jeste skup svih stanja u koja se može stići $\varepsilon$-strelicama polazeći od nekog stanja iz $P$. Dakle, važe sledeća svojstva:

- $P \subseteq \overline{P}$,

- ako $p \in P$, onda je $\delta(p, \varepsilon) \subseteq \overline{P}$.

Podskup $P$ od $Q$ nazivamo *regularnim* ako je $P = \overline{P}$. Nije teško uočiti da je za svako $P \subseteq Q$, skup $\overline{P}$ regularan. Zaista, ako $p \in \overline{\overline{P}}$, onda se u $p$ može stići $\varepsilon$-prelazima iz nekog stanja $q \in \overline{P}$, što dalje znači da se u $q$ može stići $\varepsilon$-prelazima iz nekog stanja $r \in P$, pa se samim tim u $p$ može stići $\varepsilon$-prelazima iz $r$, pa $p \in \overline{P}$. Takodje, unija regularnih skupova je regularan skup.

Zatvorenje funkcije tranzicije $\delta$ nekog $\varepsilon$-NKA definišemo oslanjajući se na $\varepsilon$-zatvorenje. Funkcija $\hat{\delta} : Q \times \Sigma^* \to \mathcal{P}(Q)$ definisana je sa:

- $\hat{\delta}(q,\varepsilon) = \overline{\{q\}}$,
- $\hat{\delta}(q,ws) = \bigcup_{r\in\hat{\delta}(q,w)} \overline{\delta(r,s)}$.

Drugim rečima, $\hat{\delta}(q,w)$ jeste skup svih stanja u koja se može stići nakon učitavanja reči $w$ polazeći iz stanja $q$. Primetimo da je za sve $q \in Q$ i $w \in \Sigma^*$, skup $\hat{\delta}(q,w)$ regularan. Jezik koji prihvata neki $\varepsilon$-NKA opisujemo i na sledeći način:

$$L(\mathbb{M}) = \{w \in \Sigma^* \mid \hat{\delta}(q,w) \cap F \neq \emptyset\}.$$

Ispostavlja se da se i svaki $\varepsilon$-NKA može simulirati nekim KA.

**Teorema 2.2.** *Za svaki $\varepsilon$-NKA $\mathbb{M}$ postoji KA $\mathbb{A}$ takav da je $L(\mathbb{M}) = L(\mathbb{A})$.*

DOKAZ. Neka je $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ neki $\varepsilon$-NKA. Neka je $\mathcal{R}$ skup svih regularnih podskupova od $Q$ i $\mathcal{F} = \{R \in \mathcal{R} \mid R \cap F \neq \emptyset\}$. Definišimo i funkciju $\Delta : \mathcal{R} \times \Sigma \to \mathcal{R}$ na sledeći način:

$$\Delta(R,s) = \bigcup_{r\in R} \overline{\delta(r,s)}.$$

Dokazaćemo da je $\mathbb{A} = (\mathcal{R}, \overline{\{q_0\}}, \mathcal{F}, \Sigma, \Delta)$ traženi automat. Dovoljno je dokazati da za svaku reč $w$ važi:

$$(\star) \qquad \hat{\Delta}(\overline{\{q_0\}}, w) = \hat{\delta}(q_0, w).$$

Dokaz izvodimo indukcijom po dužini reči $w$.

*Baza indukcije.* Neka je $|w| = 0$, tj. $w = \varepsilon$. Tada je $\hat{\delta}(q_0,\varepsilon) = \overline{\{q_0\}} = \hat{\Delta}(\overline{\{q_0\}}, \varepsilon)$.

*Indukcijski korak.* Pretpostavimo da je jednakost $(\star)$ tačna za sve reči dužine $n$, gde je $n \geqslant 0$. Neka je $|w| = n+1$. Tada je $w = vs$, pri čemu je $|v| = n$ i $s \in \Sigma$. Tada je

$$\begin{array}{rcll} \hat{\Delta}(\overline{\{q_0\}}, vs) & = & \Delta(\hat{\Delta}(\overline{\{q_0\}}, v), s) & \text{[def. zatvorenja } \hat{\Delta}\text{]} \\ & = & \Delta(\hat{\delta}(q_0, v), s) & \text{[IP]} \\ & = & \bigcup\limits_{p\in\hat{\delta}(q_0,v)} \overline{\delta(p,s)} & \text{[def. funkcije } \Delta\text{]} \\ & = & \hat{\delta}(q_0, vs) & \text{[def. zatvorenja } \hat{\delta}\text{]}, \end{array}$$

što je i trebalo dokazati. □

Popularna upotreba konačnih automata (deterministčkih i nedeteministiķih, sa $\varepsilon$-prolazom ili bez) je u analizi velikih količina teksta, traženju ključnih reči ili skupa reči. Jedan od najpopularnijih alata zasnovan na konačnim automatima je `grep` komanda UNIX operatvnog sistema. Ova komanda kao argument prihvata regularni izraz (o kojima će biti reči kasnije), od njega formira nedeteminististički konačni automat, a zatim dobijeni automat prevodi u determinističi. Ukoliko je potrebno dodatno se vrši redukcija stanja dobijenog automata. Iako se na prevodjenje automata iz nedetrminističkog u deterministištički gubi odredjeno vreme, za velike primere determinstički automat je dosta efikasniji, pa se ovaj dodatni posao u velikoj meri isplati.

## 2.4 Regularni jezici i njihove osobine

Narednom definicijom izdvajamo sve jezike koji mogu biti prihvaćeni nekim konačnim automatom.

**Definicija 2.12.** *Jezik $L$ nad alfabetom $\Sigma$ je regularan ako postoji konačan automat $\mathbb{M}$ sa ulaznim alfabetom $\Sigma$ koji prihvata jezik $L$, tj. takav da je $L(\mathbb{M}) = L$.*

PRIMER 2.14. Oslanjajući se na prethodne primere navodimo nekoliko regularnih jezika:
$\{w \in \{0,1\}^* \mid w \text{ se završava sa } 1\}$,
$\{w \in \{0,1\}^* \mid \text{num}(w) \text{ nije deljivo sa } 3\}$,
$\{w \in \{0,1\}^* \mid \text{num}(w) \equiv_3 1\}$,
$\{w \in \{a,b,c\}^* \mid w = a^k b^\ell c^m, k,\ell,m \geqslant 1\}$.

Da bismo dokazali da je neki jezik regularan treba konstruisati (nedeterministički) konačni automat (bez $\varepsilon$-prelaza ili sa ovakvim prelazima) koji ga prihvata, pri čemu se podrazumeva i dokaz da konstruisani automat zaista prihvata samo reči datog jezika.

Najjednostavniji regularni jezici nad nekim alfabetom $\Sigma$ jesu $\emptyset$, $\{\varepsilon\}$ i $\{s\}$, $s \in \Sigma$. Ove jezike nazivaćemo *osnovnim* regularnim jezicima alfabeta $\Sigma$.

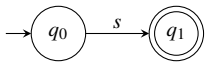
Jezik $L_\emptyset = \emptyset$ prihvata automat prikazan na narednoj slici.


Formalna definicija ovog automata je $\mathbb{M}_\emptyset = (\{q_0\}, q_0, \emptyset, \Sigma, \delta)$, pri čemu je $\delta : \{q_0\} \times \Sigma \to \mathcal{P}(\{q_0\})$ dato sa $\delta(q_0, s) = \emptyset$, za svako $s \in \Sigma$.

Jezik $L_\varepsilon = \{\varepsilon\}$ prihvata automat prikazan na narednoj slici.


Formalna definicija ovog automata je $\mathbb{M}_\varepsilon = (\{q_0\}, q_0, \{q_0\}, \Sigma, \delta)$, pri čemu je $\delta : \{q_0\} \times \Sigma \to \mathcal{P}(\{q_0\})$ dato sa $\delta(q_0, s) = \emptyset$, za svako $s \in \Sigma$.

Za svaki $s \in \Sigma$, jezik $L_s = \{s\}$ prihvata automat prikazan na narednoj slici.




Formalna definicija ovog automata je $\mathbb{M}_\varepsilon = (\{q_0, q_1\}, q_0, \{q_1\}, \Sigma, \delta)$, gde je $\delta : \{q_0, q_1\} \times \Sigma \to \mathcal{P}(\{q_0, q_1\})$ dato sa

$$\delta(q,x) = \begin{cases} \{q_1\}, & q = q_0, x = s, \\ \emptyset, & \text{inače.} \end{cases}$$

Skup svih regularnih jezika je zatvoren i za sve operacije nad jezicima koje smo uveli u prvoj glavi.

**Teorema 2.3.** *Skup svih regularnih jezika nad bilo kojim alfabetom $\Sigma$ zatvoren je za komplement, presek, uniju, nadovezivanje i Klinijevu zvezdicu.*

DOKAZ. [**Komplementiranje**] Neka je $L$ regularan jezik i $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ automat koji ga prihvata. Jednostavno je uočiti da $\mathbb{M}' = (Q, q_0, Q \setminus F, \Sigma, \delta)$ prihvata samo reči koje ne prihvata $\mathbb{M}$, tj. da je $L(\mathbb{M}') = L^{\complement}$.

[**Presek**] Neka su $L_1$ i $L_2$ dva regularna jezika nad istim alfabetom $\Sigma$, a $\mathbb{M}_1 = (Q_1, q_0^1, F_1, \Sigma, \delta_1)$ i $\mathbb{M}_2 = (Q_2, q_0^2, F_2, \Sigma, \delta_2)$ konačni automati takvi da je $L(\mathbb{M}_1) = L_1$ i $L(\mathbb{M}_2) = L_2$. Konstrisaćemo automat koji simulira rad automata $\mathbb{M}_1$ i $\mathbb{M}_2$ i prihvata reč samo u slučaju da to čine i $\mathbb{M}_1$ i $\mathbb{M}_2$. Neka je $\mathbf{M} = (Q_1 \times Q_2, (q_0^1, q_0^2), F_1 \times F_2, \Sigma, \delta)$, pri čemu je

$$\delta((q', q''), s) = (\delta_1(q', s), \delta_2(q'', s)),\ q' \in Q_1, q'' \in Q_2, s \in \Sigma.$$

Može se pokazati da je $L(\mathbb{M}) = L(\mathbb{M}_1) \cap L(\mathbb{M}_2) = L_1 \cap L_2$.

[**Unija**] Unija dva regularna jezika $L_1$ i $L_2$ nad istim alfabetom $\Sigma$, takodje je regularan jezik, jer je $L_1 \cup L_2 = \left(L_1^{\complement} \cap L_2^{\complement}\right)^{\complement}$, pa tvrdjenje sledi iz prethodna dva.

Naravno, tvrdjenje možemo i direktno dokazati konstrukcijom odgovarajućeg automata.

Za automate $M_1 = (Q_1, q_0^1, F_1, \Sigma, \delta_1)$ i $M_2 = (Q_2, q_0^2, F_2, \Sigma, \delta_2)$ takve da je $L_1 = L(M_1)$ i $L_2 = L(M_2)$ definišemo automat

$$M = (Q_1 \cup Q_2 \cup \{q_0\}, q_0, F_1 \cup F_2, \Sigma, \delta)$$

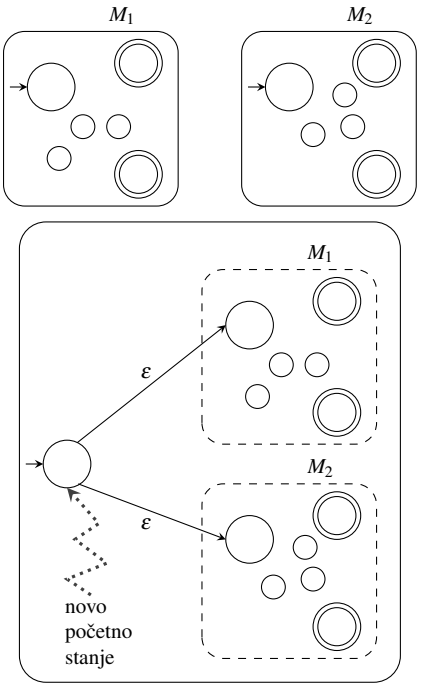
gde je

$$\delta : (Q_1 \cup Q_2 \cup \{q_0\}) \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q_1 \cup Q_2)$$

definisano sa:

$$\delta(q,s) = \begin{cases} \{q_0^1, q_0^2\}, & q = q_0,\ s = \varepsilon \\ \delta_1(q,s), & q \in Q_1,\ s \in \Sigma \\ \delta_2(q,s), & q \in Q_2,\ s \in \Sigma \\ \emptyset, & \text{inače} \end{cases}$$

takav da je $L(M) = L_1 \cup L_2$.

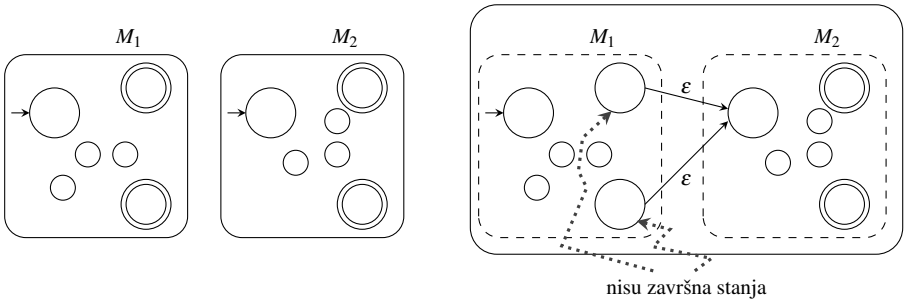



[**Nadovezivanje**] Neka su $L_1$ i $L_2$ dva regularna jezika nad istim alfabetom $\Sigma$, a $\mathbb{M}_1 = (Q_1, q_0^1, F_1, \Sigma, \delta_1)$ i $\mathbb{M}_2 = (Q_2, q_0^2, F_2, \Sigma, \delta_2)$ konačni automati takvi da je $L(\mathbb{M}_1) = L_1$ i $L(\mathbb{M}_2) = L_2$. Konstruisaćemo $\varepsilon$-NKA $\mathbb{M}$ koji prihvata $L_1 L_2$. Neka je $\mathbb{M} = (Q_1 \cup$

$Q_2, q_0^1, F_2, \Sigma, \delta)$, pri čemu je $\delta : (Q_1 \cup Q_2) \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q_1 \cup Q_2)$ definisano sa:

$$\delta(q,s) = \begin{cases} \{\delta_1(q,s)\}, & q \in Q_1, s \in \Sigma \\ \{\delta_2(q,s)\}, & q \in Q_2, s \in \Sigma \\ \{q_0^2\}, & q \in F_1, s = \varepsilon, \\ \emptyset, & \text{inače.} \end{cases}$$

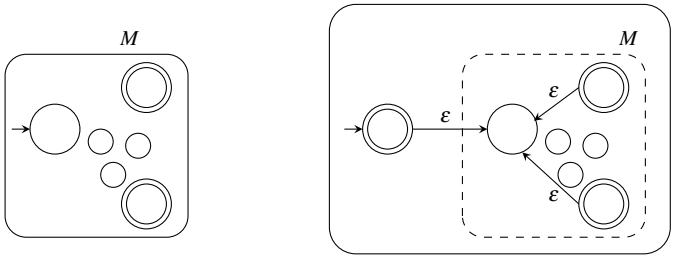
Tada je $L(\mathbb{M}) = L_1 L_2$. Naredna slika ilustruje konstrukciju automata $\mathbb{M}$.




[**Klinijeva zvezdica**] Neka je $L$ regularan jezik i $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ automat koji ga prihvata. Konstruisaćemo $\varepsilon$-NKA $\mathbb{M}'$ koji prihvata $L^*$. Da bismo definisali najavljeni automat, dodajemo jedno novo stanje $q^0$, $q^0 \notin Q$. Neka je $\mathbb{M}' = (Q \cup \{q^0\}, q^0, F \cup \{q^0\}, \Sigma, \delta)$, pri čemu je $\delta : (Q \cup \{q^0\}) \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q \cup \{q^0\})$ definisano sa:

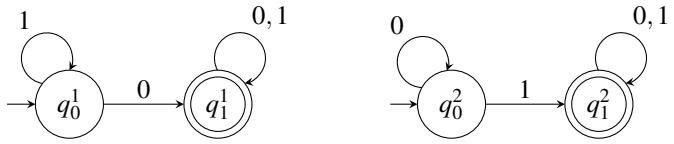
$$\delta(q,s) = \begin{cases} \{q_0\}, & q = q^0, s = \varepsilon \\ \{\delta(q,s)\}, & q \in Q, s \in \Sigma \\ \{q_0\}, & q \in F, s = \varepsilon, \\ \emptyset, & \text{inače.} \end{cases}$$

Tada je $L(\mathbb{M}') = L^*$. Naredna slika ilustruje konstrukciju automata $\mathbb{M}'$.




U prethodnoj teoremi pretpostavka je da su u slučaju operacija presek, unija i nadovezivanje dati jezici nad istim alfabetom $\Sigma$. U slučaju da je potrebno ove operacije primeniti na dva jezika nad različitim alfabtima, na primer jezik $L_1$ je nad alfabetom $\Sigma_1$, dok je jezik $L_2$ nad alfabetom $\Sigma_2$, tada se definiše unija ova dva alfabeta, tj. $\Sigma = \Sigma_1 \cup \Sigma_2$. Može se smatrati da reči jezika $L_1$, odnosno jezika $L_2$, sadrže simbole iz nekog podskupa alfabeta $\Sigma$. Primenom operacija preseka, unije i nadovezivanja u ovom slučaju dobija se rezultujući jezik nad alfabetom $\Sigma$.

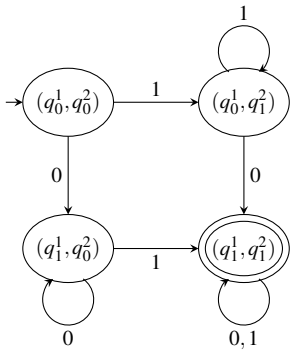
PRIMER 2.15. Na narednim slikama su prikazana dva automata. Levo je dat automat koji prihvata samo reči koje sadrže 0, a desno automat koji prihvata reči koje sadrže 1.




Sprovodeći odgovarajuću konstrukciju iz dokaza prethodne teoreme, konstruišemo presek jezika ova dva automata, tj. jezik koji prihvata samo reči koje sadrže i 0 i 1.

U rezultujućem automatu $M = (Q_1 \times Q_2, (q_0^1, q_0^2), \{(q_1^1, q_1^2)\}, \{0,1\}, \delta)$, funkcija tranzicije $\delta$ data je sledećom tablicom

| $\delta$ | 0 | 1 |
|---|---|---|
| $(q_0^1, q_0^2)$ | $(q_1^1, q_0^2)$ | $(q_0^1, q_1^2)$ |
| $(q_0^1, q_1^2)$ | $(q_1^1, q_1^2)$ | $(q_0^1, q_1^2)$ |
| $(q_1^1, q_0^2)$ | $(q_1^1, q_0^2)$ | $(q_1^1, q_1^2)$ |
| $(q_1^1, q_1^2)$ | $(q_1^1, q_1^2)$ | $(q_1^1, q_1^2)$ |




**Teorema 2.4.** *Svaki regularan jezik se može predstaviti primenom unije, nadovezivanja i Klinijeve zvezdice na osnovne regularne jezike.*

DOKAZ. Neka je $L$ regularan jezik i $\mathbb{M} = (\{q_1, \ldots, q_n\}, q_1, F, \Sigma, \delta)$ automat koji ga prihvata, $L = L(\mathbb{M})$.

Označimo sa $R_{ij}^k$, $i, j = 1, \ldots, n$, $k = 0, 1, \ldots, n$ skup svih reči koje se mogu pročitati na putu iz stanja $q_i$ u stanje $q_j$ koji ne prolazi kroz stanja $q_\ell$ za $\ell > k$. Definicija skupova $R_{ij}^k$ je induktivna: počinjemo sa $k = 0$ i završavamo sa $k = n$ (u ovom poslednjem slučaju nema nikakvih ograničenja).

$\boxed{k = 0}$ Pošto su sva stanja automata $\mathbb{M}$ numerisana brojevima većim od 1, zahtev za puteve izmedju od $q_i$ do $q_j$ znači da nema posrednika. Za $i \neq j$, takav put je zapravo ivica od stanja $q_i$ do stanja $q_j$, a za $i = j$ reč je o petlji oko čvora $q_i$.

Neka je $i \neq j$. Potražimo sve simbole $s \in \Sigma$ takve da postoji ivica iz $q_i$ u $q_j$ označena sa $s$.

- Ako ne postoji simbol $s \in \Sigma$ takav da je $\delta(q_i, s) = q_j$, onda je $R_{ij}^0 = \emptyset$.
- Ako su $s_{\ell_1}, \ldots, s_{\ell_m}$, za neko $m \geqslant 1$, medjusobno različiti simboli iz $\Sigma$, takvi da je $\delta(q_i, s_{\ell_t}) = q_j$, $t = 1, \ldots, m$, onda je $R_{ij}^0 = \{s_{\ell_1}\} \cup \cdots \cup \{s_{\ell_m}\}$.

Dakle, $R_{ij}^0 = \{s \in \Sigma \mid \delta(q_i, s) = q_j\}$.

Neka je $i = j$. Potražimo sve simbole $s \in \Sigma$ takve da postoji petlja oko $q_i$ označena sa $s$.

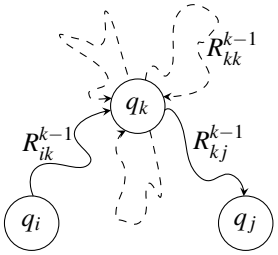
- Ako ne postoji simbol $s \in \Sigma$ takav da je $\delta(q_i, s) = q_i$, onda je $R_{ii}^0 = \{\varepsilon\}$.
- Ako su $s_{\ell_1}, \ldots, s_{\ell_m}$, za neko $m \geqslant 1$, medjusobno različiti simboli iz $\Sigma$, takvi da je $\delta(q_i, s_{\ell_t}) = q_i$, $t = 1, \ldots, m$, onda je $R_{ii}^0 = \{\varepsilon\} \cup \{s_{\ell_1}\} \cup \cdots \cup \{s_{\ell_m}\}$.

Dakle, $R_{ii}^0 = \{\varepsilon\} \cup \{s \in \Sigma \mid \delta(q_i, s) = q_i\}$.

$\boxed{k > 0}$ Za svaki put iz stanja $q_i$ u stanje $q_j$ koji ne prolazi kroz stanja $q_\ell$ za $\ell > k$ postoje dve mogućnosti.

**1. slučaj**. Ako učeni put ne prolazi kroz stanje $q_k$, onda on zapravo pripada $R_{ij}^{k-1}$.

**2. slučaj**. Ako uočeni put prolazi kroz stanje $q_k$ bar jednom, onda se on može razložiti na nekoliko puteva koji ne prolaze kroz stanja $q_\ell$, $\ell > k-1$, kao na narednoj slici.

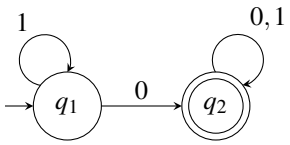



Dakle, $R_{ij}^k = R_{ij}^{k-1} \cup R_{ik}^{k-1}(R_{kk}^{k-1})^* R_{kj}^{k-1}$.

Na osnovu opisane konstrukcije sledi da se svi skupovi $R_{ij}^k$ mogu dobiti primenom unije, nadovezivanja i Klinijeve zvezdice na osnovne jezike. Isto važi i za skup $L$, jer je

$$L = L(\mathbb{M}) = \bigcup_{q_\ell \in F} R_{1\ell}^n.$$

PRIMER 2.16. Dokaz prethodne teoreme sadrži zapravo postupak kojim se jezik nekog konačnog automata može opisati polazeći od osnovnih jezika primenom unije, nadovezivanja i Klinijeve zvezdice. Postupak ilustrujemo na primeru jezika $L(\mathbb{M})$, gde je $\mathbb{M}$ automat prikazan na narednoj slici.




Odredimo najpre skupove $R_{ij}^0$, $1 \leqslant i, j \leqslant 2$:
$R_{11}^0 = \{\varepsilon, 1\} = \{\varepsilon\} \cup \{1\}$,

$R^0_{12} = \{0\}$,
$R^0_{21} = \emptyset$,
$R^0_{22} = \{\varepsilon, 0, 1\} = \{\varepsilon\} \cup \{0\} \cup \{1\}$.

Koristeći jednakost $R^1_{ij} = R^0_{ij} \cup R^0_{i1} \left(R^0_{11}\right)^* R^0_{1j}$, odredjujemo skupove $R^1_{ij}$:
$R^1_{11} = \{\varepsilon, 1\} \cup \{\varepsilon, 1\}\{\varepsilon, 1\}^*\{\varepsilon, 1\} = \{1\}^*$
$R^1_{12} = \{0\} \cup \{\varepsilon, 1\}\{\varepsilon, 1\}^*\{0\} = \{1\}^*\{0\}$
$R^1_{21} = \emptyset \cup \emptyset\{\varepsilon, 1\}^*\{\varepsilon, 1\} = \emptyset$
$R^1_{22} = \{\varepsilon, 0, 1\} \cup \emptyset\{\varepsilon, 1\}^*\{0\} = \{\varepsilon, 0, 1\}$.

Najzad, koristeći jednakost $R^2_{ij} = R^1_{ij} \cup R^1_{i2} \left(R^1_{22}\right)^* R^1_{2j}$, odredjujemo skupove $R^2_{ij}$. Primetimo da je $L(\mathbb{M}) = R^2_{12}$, pa je dovoljno odrediti samo $R^2_{12}$.
$R^2_{12} = \{1\}^*\{0\} \cup \{1\}^*\{0\}\{\varepsilon, 0, 1\}^*\{\varepsilon, 0, 1\} = \{1\}^*\{0\}\{0, 1\}^*$.

Dakle, $L(\mathbb{M}) = R^2_{12} = \{1\}^*\{0\}\{0, 1\}^*$. Ovakve opise regularnih jezika, koje skraćujemo izostavljanjem vitičastih zagrada, nazivamo *regularnim izrazima*, kojima je posvećeno naredno poglavlje.

Na osnovu prethodne dve teoreme sledi da se skup regularnih jezika nad $\Sigma$, u oznaci $\mathrm{Reg}(\Sigma)$, može induktivno definisati kao najmanji (u smislu inkluzije) skup takav da:

- $\emptyset, \{\varepsilon\}, \{s\} \in \mathrm{Reg}(\Sigma)$, za svako $s \in \Sigma$;
- ako $L_1, L_2 \in \mathrm{Reg}(\Sigma)$, onda $L_1 \cup L_2, L_1 L_2, L_1^* \in \mathrm{Reg}(\Sigma)$.

## 2.5 Regularni izrazi

Teorema 2.4 omogućava da na veoma elegantan način opisujemo regularne jezike. Reč je o regularnim izrazima.

**Regularni izrazi** nad alfabetom $\Sigma$ formiraju se na sledeći način:

- $\emptyset$, $\varepsilon$ i $s$, za svako $s \in \Sigma$, jesu regularni izrazi;
- ako su $\alpha$ i $\beta$ regularni izrazi, onda su to i $(\alpha \cup \beta)$ i $(\alpha\beta)$;
- ako je $\alpha$ regularan izraz, onda je to i $\alpha^*$.

Kao pri zapisivanju bilo kakvih algebarskih izraza, usvajamo neke prirodne konvencije o izostavljanju zagrada. Tako, izostavljamo spoljašnje zagrade, kao i one zagrade koje postaju suvišne nakon uvodjenja prioriteta medju operacijama koje učestvuju u gradjenju regularnih izraza: smatraćemo da je $*$ operacija najvećeg prioriteta, da je nadovezivanje manjeg prioriteta, i najzad da je unija najmanjeg prioriteta.

PRIMER 2.17. Navodimo nekoliko regularnih izraza nad $\Sigma_{\mathrm{bool}} = \{0, 1\}$:

$$\emptyset, \emptyset \cup 0, 01; 1^*0 \cup 10^*; (0^*1 \cup \varepsilon)^*10, \ldots$$

Svakom regularnom izrazu $\alpha$ nad $\Sigma$, na prirodan način pridružujemo jedan regularan jezik $L(\alpha) \subseteq \Sigma^*$:

- $L(\emptyset) = \emptyset$, $L(\varepsilon) = \{\varepsilon\}$, $L(s) = \{s\}$, $s \in \Sigma$;
- $L(\alpha \cup \beta) = L(\alpha) \cup L(\beta)$, $L(\alpha\beta) = L(\alpha)L(\beta)$,
- $L(\alpha^*) = L(\alpha)^*$.

PRIMER 2.18. Posmatrajmo neke regularne izraze nad $\Sigma_{\text{bool}} = \{0,1\}$ i odredimo odgovarajuće jezike.

$$\begin{aligned} L(01^* \cup 10^*) &= L(0)L(1)^* \cup L(1)L(0)^* = \{0\}\{1\}^* \cup \{1\}\{0\}^* \\ &= \{01^n \mid n \geqslant 0\} \cup \{10^n \mid n \geqslant 0\} \end{aligned}$$

$$L(0^*1^*) = L(0)^*L(1)^* = \{0\}^*\{1\}^* = \{0^m1^n \mid m,n \geqslant 0\}$$

$$L((01)^*) = (L(0)L(1))^* = \{01\}^* = \{(01)^n \mid n \geqslant 0\}$$

$$L((0 \cup 1)^*) = (L(0) \cup L(1))^* = \{0,1\}^* = \Sigma_{\text{bool}}^*$$

PRIMER 2.19. Kao što je već rečeno, regularni izrazi, u praksi, mogu da se koriste za prepoznavanje ključnih reči u tekstu. Na primer, nad alfabetom tastature $\Sigma_{\text{keyboard}}$) može da se definiše regularni izraza za prepoznavanje naziva promenljivih u programskom jeziku C.

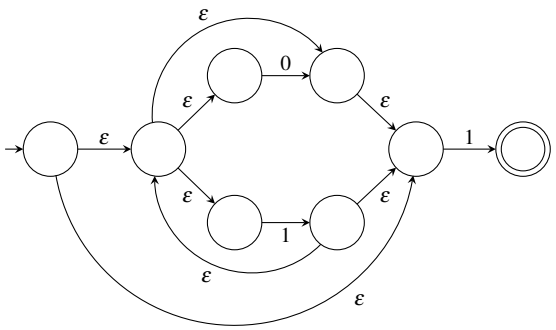
$$(\{\_\} \cup \varepsilon)\{\texttt{a},\texttt{b},\ldots,\texttt{z},\texttt{A},\texttt{B},\ldots,\texttt{Z}\}\{\_,\texttt{a},\texttt{b},\ldots,\texttt{z},\texttt{A},\texttt{B},\ldots,\texttt{Z},\texttt{0},\texttt{1},\ldots,\texttt{9}\}^*$$

Direktnom primenom teorema 2.3 i 2.4 sledi sledeće tvrdjenje.

**Teorema 2.5.** *Jezik $L \subseteq \Sigma^*$ je regularan akko postoji regularan izraz $\alpha$ nad $\Sigma$ takav da je $L = L(\alpha)$.*

Dokazi pomenutih teorema su konstruktivni: na osnovu dokaza teoreme 2.3 za svaki regularan izraz možemo konstruisati konačni automat koji prihvata jezik odredjen tim izrazom, dok dokaz teoreme 2.4 omogućava da se jezik koji prihvata neki konačan automat opiše regularnim izrazom.

PRIMER 2.20. Konstruišimo konačan automat koji prihvata jezik odredjen regularnim izrazom $(0 \cup 1)^*1$.



Postoji mnogo praktičniji postupak kojim se jezik konačnog automata opisuje regularnim izrazom. Taj postupak je zasnovan na sledećoj lemi.

**Lema 2.2.** *Neka su $K \subseteq \Sigma^+$ i $L \subseteq \Sigma^*$ regularni jezici. Tada jednačina $X = XK \cup L$ ima jedinstveno rešenje $X = LK^*$ koje je regularan jezik.*

DOKAZ. Jednostavno se proverava da je $LK^*$ jedno rešenje date jednačine:

$$(LK^*)K \cup L = L(K^*K) \cup L = LK^+ \cup L = L(K^+ \cup \{\varepsilon\}) = LK^*.$$

Da bismo dokazali da je ovo i jedinstveno rešenje, pretpostavimo da su $L_1$ i $L_2$ dva različita rešenja, i $w$ najkraća (po dužini) reč koja pripada simetričnoj razlici jezika $L_1$ i $L_2$. Bez gubljenja opštosti, neka je $w \in L_1 \setminus L_2$. Sada imamo da je

$$w \notin L_2 \overset{L_2 = L_2K \cup L}{\Longrightarrow} w \notin L \overset{L_1 = L_1K \cup L}{\Longrightarrow} w \in L_1K,$$

pa je $w = uv$, za neke $u \in L_1$ i $v \in K$. Kako je $K \subseteq \Sigma^+$, $|v| > 0$, pa je $|u| < |w|$. Medjutim, pošto $u \in L_1$, iz minimalnosti dužine reči $w$ sledi da $u \in L_2$. Na taj način dolazimo do kontradikcije:

$$w = uv \in L_2K \subseteq L_2.$$

□

Opišimo sada kako ova lema daje najavljeni algoritam. Posmatrajmo proizvoljan konačni automat $\mathbb{M} = (\{q_0, q_1, \ldots, q_n\}, q_0, F, \Sigma, \delta)$. Definišimo sledeće skupove:

$$L_i = L(\mathbb{M}_i), \text{ gde je } \mathbb{M}_i = (\{q_0, q_1, \ldots, q_n\}, q_0, \{q_i\}, \Sigma, \delta),\ 0 \leqslant i \leqslant n;$$

$$K_{ij} = \{s \in \Sigma \mid \delta(q_j, s) = q_i\},\ 0 \leqslant i, j \leqslant n.$$

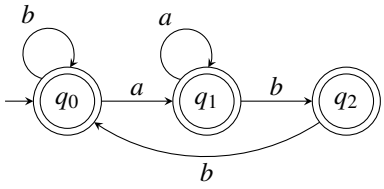
Ovi jezici su povezani sledećim jednakostima:

$$(\star) \qquad \begin{cases} L_0 = L_0K_{00} \cup L_1K_{01} \cup \cdots \cup L_nK_{0n} \cup \{\varepsilon\}, \\ L_1 = L_0K_{10} \cup L_1K_{11} \cup \cdots \cup L_nK_{1n}, \\ \vdots \\ L_n = L_0K_{n0} \cup L_1K_{n1} \cup \cdots \cup L_nK_{nn}. \end{cases}$$

U ovim jednakostima, jezici $L_i$ su nepoznati jezici, dok $K_{ij}$ mogu biti direktno odredjeni iz $\mathbb{M}$. Takodje, imamo da je $L(\mathbb{M}) = \bigcup_{q_i \in F} L_i$.

Prethodnu lemu primenjujemo pri rešavanju sistema $(\star)$. Najpre, na osnovu $K_{00} \subseteq \Sigma^+$ možemo naći $L_0$ u zavisnosti od $L_1, L_2, \ldots, L_n$ iz prve jednačine i zameniti ga u ostalim jednačinama. Na taj način dobijamo sistem sa manje nepoznatih, a jednačine i dalje zadovoljavaju uslove za koeficijente $K$. Dakle, procedura se može nastaviti, i na taj način se mogu naći ostale promenljive. Opisana procedura se može primeniti i na nedeterminističke automate.

PRIMER 2.21. Posmatrajmo automat prikazan na narednoj slici:




Tada dobijamo sistem

$$\begin{cases} L_0 = L_0b \cup L_1\emptyset \cup L_2b \cup \{\varepsilon\}, \\ L_1 = L_0a \cup L_1a \cup L_2\emptyset, \\ L_2 = L_0\emptyset \cup L_1b \cup L_2\emptyset \end{cases} \Leftrightarrow \begin{cases} L_0 = L_0b \cup L_2b \cup \{\varepsilon\}, \\ L_1 = L_0a \cup L_1a, \\ L_2 = L_1b \end{cases}$$

$$\overset{L_2=L_1b}{\Leftrightarrow} \begin{cases} L_0 = L_0b \cup L_1bb \cup \{\varepsilon\}, \\ L_1 = L_0a \cup L_1a, \end{cases}$$

$$\overset{L_1=L_0aa^*}{\Leftrightarrow} L_0 = L_0b \cup L_0aa^*bb \cup \{\varepsilon\} \Leftrightarrow L_0 = (b \cup a^+bb)^*.$$

$$\begin{cases} L_0 = (b \cup a^+bb)^* \\ L_1 = (b \cup a^+bb)^*a^+ \\ L_2 = (b \cup a^+bb)^*a^+b \end{cases} \Rightarrow L(\mathbb{M}) = (b \cup a^+bb)^*(\varepsilon \cup a^+(\varepsilon \cup b))$$

## 2.6 Jezici koji nisu regularni; Lema naduvavanja za regularne jezike

Za svaki alfabet $\Sigma$ postoji samo prebrojivo mnogo regularnih izraza, dok podskupova od $\Sigma^*$ ima neprebrojivo mnogo, odakle direktno sledi da postoje jezici nad $\Sigma$ koji nisu regularni (naravno, koristimo teoremu 2.5).

PRIMER 2.22. Dokažimo da jezik $L = \{0^n1^n \mid n \geqslant 1\}$ nije regularan.

Pretpostavimo suprotno, neka je $\mathbb{M} = (Q, q_0, F, \{0,1\}, \delta)$ automat takav da je $L = L(\mathbb{M})$. Neka je $|Q| = n$. Posmatrajmo reči

$$0, 0^2, 0^3, \ldots, 0^n, 0^{n+1}.$$

Prema Dirihleovom principu postoje $i, j$ takvi da je $1 \leqslant i < j \leqslant n+1$ i

$$\widehat{\delta}(q_0, 0^i) = \widehat{\delta}(q_0, 0^j).$$

Tada će za svaku reč $w \in \{0,1\}^*$ važiti

$$\widehat{\delta}(q_0, 0^iw) = \widehat{\delta}(q_0, 0^jw).$$

Specijalno, za $w = 1^i$ važi $\widehat{\delta}(q_0, 0^i1^i) = \widehat{\delta}(q_0, 0^j1^i)$. Medjutim, ovo je nemoguće, jer $\widehat{\delta}(q_0, 0^i1^i) \in F$, dok $\widehat{\delta}(q_0, 0^i1^j) \notin F$.

**Teorema 2.6.** *Za svaki regularan jezik $L$ nad nekim alfabetom $\Sigma$ postoji prirodan broj $n$ takav da se svaka reč $w$ iz $L$ dužine bar $n$ ($|w| \geqslant n$) može zapisati u obliku $w = xyz$, za neke $x, y, z \in \Sigma^*$ tako da važi:*

- $y \neq \varepsilon$;
- $|xy| \leqslant n$;
- $xy^k z \in L$, *za svako* $k \geqslant 0$.

NAPOMENA. Prethodna teorema tvrdi da se u svakoj dovoljno dugačkoj reči regularnog jezika može pronaći neprazna podreč, ne mnogo udaljena od početka uočene reči, koja se može izbrisati i naduvavati, a da novodobijena reč i dalje pripada jeziku.

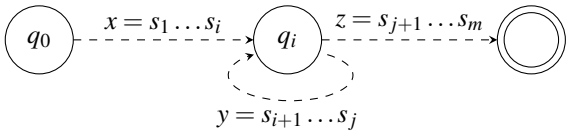
DOKAZ. Neka je $L$ regularan jezik i $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ automat koji ga prihvata, $L = L(\mathbb{M})$. Pokazaćemo da je $n = |Q|$ traženi prirodan broj.

Neka je $w \in L$ reč čija dužina nije manja od $n$; $w = s_1 s_2 \cdots s_m$, $m \geqslant n$. Za svako $i \in \{0, 1, \ldots, n\}$, neka je $q_i$ stanje u kome se nalazi automat nakon čitanja prvih $i$ simbola reči $w$:

$$q_1 = \delta(q_0, s_1),\ q_2 = \delta(q_1, s_2),\ \cdots,\ q_n = \delta(q_{n-1}, s_n).$$

Tada, prema Dirihleovom principu, medju stanjima $q_0$, $q_1$, $\ldots$, $q_n$ postoje dva ista stanja (jer je $n = |Q|$). Neka je $q_i = q_j$, za neke $0 \leqslant i < j \leqslant n$. Tada je $w = xyz$, pri čemu je:

- $x = s_1 \cdots s_i$ ili $x = \varepsilon$ ako je $i = 0$,
- $y = s_{i+1} \cdots s_j$,
- $z = s_{j+1} \cdots s_m$ ili $z = \varepsilon$ ako je $j = n$.




Iz $i < j$ sledi da je $y \neq \varepsilon$, iz $j \leqslant n$ da je $|xy| \leqslant n$. Najzad, iz $q_i s_{i+1} \cdots s_j \vdash_{\mathbb{M}}^* q_i$, sledi da $xy^k z \in L$, za svako $k \geqslant 0$. □

PRIMER 2.23. Dokažimo da jezici

$$L_1 = \{0^n 1^n \mid n \geqslant 0\} \text{ i } L_2 = \{w \in \Sigma_{\text{bool}}^* \mid |w|_0 = |w|_1\}$$

nisu regularni.

1) Pretpostavimo suprotno: neka je $L_1$ regularan. Označimo sa $n$ prirodan broj čije postojanje tvrdi lema naduvavanja. Posmatrajmo reč $w = 0^n1^n \in L_1$. Tada je $|w| = 2n \geqslant n$, pa postoji razlaganje $w = xyz$ takvo da je $y \neq \varepsilon$, $|xy| \leqslant n$ i $xy^kz \in L_1$, za bilo koje $k \geqslant 0$. Medjutim, iz

$$0^n1^n = xyz,\ y \neq \varepsilon \text{ i } |xy| \leqslant n$$

sledi da je $y = 0^t$, za neko $t \geqslant 1$, odakle dobijamo da

$$0^{n+kt}1^n \in L_1, \text{ za svako } k > 0,$$

što je kontradikcija. Dakle, $L_1$ nije regularan.

2) Pretpostavimo da je $L_2$ regularan. Tada je i $L_2 \cap L(0^*1^*)$ takodje regularan jezik, što nije moguće jer je $L_2 \cap L(0^*1^*) = L_1$.

PRIMER 2.24. Dokažimo da jezik $L = \{a^{n^2} \mid n \geqslant 1\}$ nije regularan.

Pretpostavimo suprotno. Neka je $n_L$ prirodan broj čije postojanje tvrdi lema naduvavanja. Posmatrajmo reč $w = a^{n_L^2}$. Tada je $|w| = n_L^2 \geqslant n_L$. Prema lemi naduvavanja, postoji razlaganje $w = xyz$, takvo da je

$$y \neq \varepsilon,\ |xy| \leqslant n_L, xy^kz \in L,\ k \geqslant 0.$$

Tada je

$$x = a^p,\ y = a^q,\ z = a^r,$$

pri čemu je

$$0 \leqslant p < n_L,\ 0 < q \leqslant n_L,\ 0 \leqslant r,\ p+q \leqslant n_L,\ p+q+r = n_L^2.$$

Dužina reči $xy^2q$ jednaka je $p+2q+r$ što nije potpun kvadrat:

$$n_L^2 = p+q+r < p+2q+r \leqslant n_L^2 + n_L < n_L^2 + 2n_L + 1 = (n_L+1)^2,$$

odakle sledi da $xy^2z \notin L$. Dobijena kontradikcija obara pretpostavku da je $L$ regularan jezik.

Na osnovu dokaza leme naduvavanja izvodimo neke korisne posledice.

**Posledica 2.1.** *Neka je* $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ *konačni automat.*

1) $L(\mathbb{M}) \neq \emptyset$ *akko postoji* $w \in L(\mathbb{M})$ *takva da je* $|w| < |Q|$;

2) $L(\mathbb{M})$ *je beskonačan akko postoji* $w \in L(\mathbb{M})$*takva da je* $|Q| \leqslant |w| < 2|Q|$.

DOKAZ. 1) ($\Leftarrow$) Očigledno.

($\Rightarrow$) Neka je $|Q| = n$ i $w$ najkraća reč u $L(\mathbb{M})$. Ako je $|w| < n$ tvrdjenje je dokazano. U suprotnom, ako je $|w| \geqslant n$, onda je $w = xyz$, pri čemu je $y \neq \varepsilon$ i $xz \in L(\mathbb{M})$, što je nemoguće, jer je $w$ minimalne dužine.

2) ($\Leftarrow$) Direktno iz leme naduvavanja.

($\Rightarrow$) Ako je $L(\mathbb{M})$ beskonačan jezik, on sadrži reč dužine bar $n = |Q|$. Neka je $w$ najkraća reč čija je dužina bar $n$. Ako je $|w| < 2n$ tvrdjenje je dokazano. U suprotnom, neka je $|w| \geqslant 2n$ prema lemi naduvavanja $w$ se može rastaviti $w = xyz$ tako da je $|xy| \leqslant n$, $y \neq \varepsilon$ i $xz \in L(\mathbb{M})$. Medjutim, tada je $|w| > |xz| \geqslant n$, što je nemoguće. □

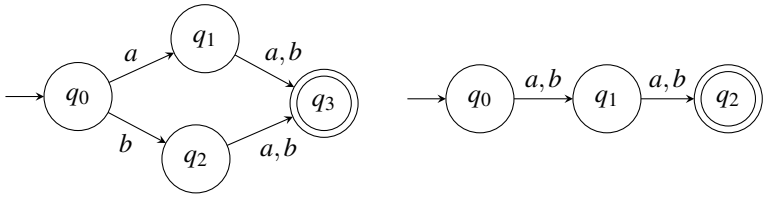
**2.1**. Dokazati da sledeći jezici nisu regularni:

1) $\{ww \mid w \in \{0,1\}^*\}$;

2) $\{w \in \{0,1\}^* \mid |w|_0 = |w|_1\}$.

## 2.7 Minimizacija konačih automata

PRIMER 2.25. Neka su dati automati




Jasno je da oni prihvataju isti jezik. Obzirom da prvi automat ima veći broj stanja, očigledno je da se može napraviti automat sa manjim brojem stanja, pa se samim tim postavlja pitanje postojanje automata sa minimalnim brojem stanja.

Za potrebe razmatranja pitanja minimalnog automata, najpre uvodimo sledeću definiciju:

**Definicija 2.13.** *Reči $x$ i $y$ su ekvivalentne u odnosu na jezik $L$, u oznaci $x \approx_L y$, akko* $\{z \mid xz \in L\} = \{z \mid yz \in L\}$

Za definisanje ekvivalencije dve reči nije neophodno da jezik $L$ bude regularan, ali pokazaće se da u slučaju da $L$ jeste regularan, relacija $\approx_L$ ima konačno mnogo klasa ekvivaencije.

Relacija $\approx_L$ može da se uvede i na sledeći način:

- Ako postoji $z$ tako da $xz \in L$ i $yz \notin L$ tada $x \not\approx_L y$
- Ako postoji $z$ tako da $xz \notin L$ i $yz \in L$ tada $x \not\approx_L y$

- U suprotnom $x \approx_L y$

PRIMER 2.26. Neka je $L = \{ab, ac, bb, bc\}$, tada je

| $x$ | $\{z \mid xz \in L\}$ |
|---|---|
| $a$ | $\{b, c\}$ |
| $b$ | $\{b, c\}$ |
| $c$ | $\emptyset$ |
| $ab$ | $\{\varepsilon\}$ |
| $ac$ | $\{\varepsilon\}$ |
| $\varepsilon$ | $L$ |
| $abc$ | $\emptyset$ |

To znači da je $a \approx_L b$ i $ab \approx_L ac$, ali $a \not\approx_L c$ i $b \not\approx_L c$, na primer. Ovo znači da postoje četiri klase ekvivalencije koje odgovaraju različitim vrednostima skupova $\{z \mid xz \in L\}$ za različite vrednosti $x$-a.

PRIMER 2.27. Neka je $L = \{w \in \{0,1\}^* \mid |w| \text{ je paran broj}\}$, tada je

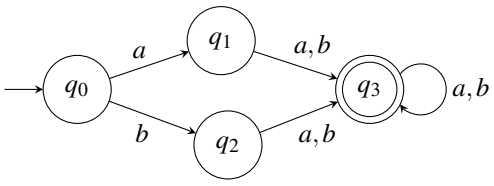
| $x$ | $\{z \mid xz \in L\}$ |
|---|---|
| 0 | reči neparne dužine |
| 1 | reči neparne dužine |
| 00 | reči parne dužine |
| 01 | reči parne dužine |
| 101 | reči neparne dužine |
| $\vdots$ | $\vdots$ |

To znači da je $0 \approx_L 1$, $00 \approx_L 01$, $0 \approx_L 101$, $0 \not\approx_L 00$, $1 \not\approx_L 01$ i tako dalje. Dakle, postoje dve klase ekvivalencije.

Kako svakom regularnom jeziku $L$ odgovara neki konačni automat $M$, nameće pitanje veze automata $M$ i klasa ekvivalencije relacie $\approx_L$. Iz tog razloga, uvodi se sleća definicija:

**Definicija 2.14.** *Dve reči $x, y \in \Sigma^*$ su ekvivalentne u odnosu na automat $M = (Q, q_0, F, \Sigma, \delta)$, u oznaci $x \sim_M y$, akko $q_0 x \vdash_{\mathbb{M}}^* t$ i $q_0 y \vdash_{\mathbb{M}}^* t$.*

PRIMER 2.28. Za automat



važi da reči $a \not\sim_M b$, pošto njihovim učitavanjem, polazeći od početnog stanja, automat završava u stanjima $q_1$ i $q_2$, respektivno. Medjutim važi da je $aa \sim_M ba$, isto kao i $aa \sim_M aaa$.

**Teorema 2.7.** *Za proizvoljni (deterministički) konačni automat $M = (Q, q_0, F, \Sigma, \delta)$ i proizvoljne reči $x, y \in \Sigma^*$, ako je $x \sim_M y$ tada je $x \approx_{L(M)} y$.*

DOKAZ. Neka je za automat $M = (Q, q_0, F, \Sigma, \delta)$ $x \sim_M y$. Potrebno je pokazati da važi $x \approx_M y$, odnosno da za svako $z$ treba da važi $xz \in L(M)$ akko $yz \in L(M)$.

Neka je

$$q_0 x \vdash^*_{\mathbb{M}} t_1 \qquad \text{i} \qquad q_0 y \vdash^*_{\mathbb{M}} t_2,$$

gde je $q_0$ početno stanje automata $M$. Na osnovu $x \sim_M y$, mora da važi da je $t_1 = t_2$.

Za neka stanja $u_1$ i $u_2$ važi

$$q_0 xz \vdash^*_{\mathbb{M}} u_1 \qquad \text{i} \qquad q_0 yz \vdash^*_{\mathbb{M}} u_2.$$

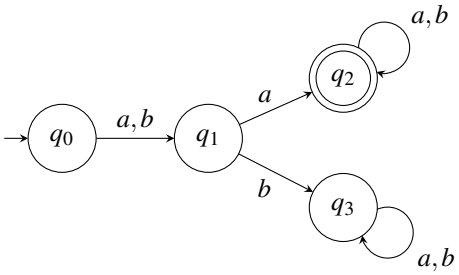
Odnosno,

$$q_0 xz \vdash^*_{\mathbb{M}} t_1 z \vdash^*_{\mathbb{M}} u_1 \qquad \text{i} \qquad q_0 yz \vdash^*_{\mathbb{M}} t_2 z \vdash^*_{\mathbb{M}} u_2.$$

Obzirom da je $t_1 = t_2$ i da je $M$ deterministički automat, to znači da je $u_1 = u_2$.

Iz prethodnog sledi da automat $M$ prihvata reč $xz$
akko je $u_1$ završno stanje, odnosno,
akko je $u_2$ završno stanje, odnosno,
akko prihvata reč $yz$.
To znači da $xz \in L(M)$ akko $yz \in L(M)$, tj. $x \approx_{L(M)} y$. □

PRIMER 2.29. Neka je dat automat




Važi da je $a \sim_M b$, pošto oba završavaju u istom stanju. Pitanje je da li važi $a \approx_{L(M)} b$, tj. da li važi da $az \in L(M)$ akko $bz \in L(M)$. Na primer, za $z = a$ važi da $aa \in L(M)$ kao i da $ba \in L(M)$. Takodje, za $z = b$, važi da $ab \notin L(M)$ i $bb \notin L(M)$. Isto važi sa svako $z$, tako da $az \in L(M)$ akko $bz \in L(M)$, odnosno $a \approx_{L(M)} b$.

Na osnovu prethodne teoreme može se zaključiti da

**Posledica 2.2.** *Svaki konačni automat $M$ koji prepoznaje jezik $L$ mora da ima najmanje onoliko stanja koliko ima klasa ekvivalencije relacije $\approx_L$.*

Ukoliko bi postojalo više klasa ekvivalencije nego stanja, tada bi moralo da postoji neko stanje $q$ automata $M$ i dve reči $x, y \in \Sigma^*$ takva da $x \not\approx_L y$, ali učitavanje obe reči $x$ i $y$ završava u stanju $q$, tj. $x \sim_M y$. Što je nemoguće na osnovu teoreme. Takodje,

**Posledica 2.3.** *Ako je jezik $L$ regularan, tada relacija $\approx_L$ ima konačno mnogo klasa ekvivalencije.*

Ako je jezik $L$ regularan, tada postoji konačni automat koji ga prihvata i koji ima konačno mnogo stanja. Broj stanja automata $M$ jednak je broju klasa ekvivalencije relacije $\sim_M$ (ili veći ako postoje nedostižna stanja), tako da je broj klasa ekvivalencije relacije $\sim_M$ konačno. Ovaj broj je jednak ili veći od broja klasa ekvivalencije relacije $\approx_L$, koji je takodje mora biti konačan.

**Teorema 2.8.** *[Myhill-Nerode teorema] Neka je $L \subseteq \Sigma^*$ regularan jezik. Tada postoji minimalni konačni automat $M$ koji ima broj stanja jednak broju klasa ekvivalencije relacije $\approx_L$.*

DOKAZ. Automat $M$ će biti konstruisan tako da svakom stanju automata odgovara jedna klasa ekvivalencije relacije $\approx_L$, na sledeći način:

Neka su $L_1, \ldots, L_r$ klase ekvivalencije relacije $\approx_L$. Ovi skupovi su disjunktni i njihova unija je jednaka skupu $\Sigma^*$. Bez gubljenja opštosti sa $L_1$ se može označiti skup takav da $\varepsilon \in L_1$. Definisaćemo $Q = \{q_1, \ldots, q_r\}$, za svaku klasu ekvivalencije po jedno stanje automata.

Primetimo da za svako $L_j$ i proizvoljno $a \in \Sigma$, za svako $x, y \in L_j$, važi da $xa \approx_L ya$, zato što za svako $z \in \Sigma^*$ važi $xaz \in L \Leftrightarrow yaz \in L$, pošto $x \approx_L y$.

Da bismo odredili $\delta(q_j, a)$, izaberimo proizvoljno $x \in L_j$ i odredimo $k$ tako da $xa \in L_k$ i tada je $\delta(q_j, a) = q_k$. Odgovor će se dobiti isti bez obzira na izabrano $x$.

Da bismo odredili završna stanja, primetimo da za svako $j$ važi ili $L_j \subset L$ ili $L_j \cap L = \emptyset$. Na osnovu toga $F = \{q_j \mid L_j \subseteq L\}$.

Indukcijom se lako pokazuje da $\hat{\delta}(q_1, x) = q_j \Leftrightarrow x \in L_j$. Ova činjenica, zajedno sa načinom izbora skupa $F$, obezbedjuje da je $L(M) = L$. □

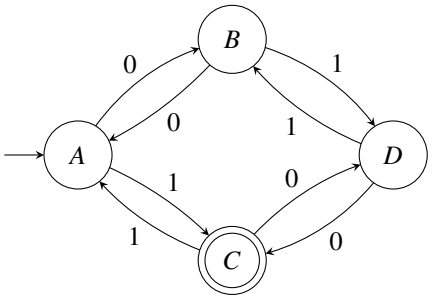
PRIMER 2.30. Koristeći ideju izloženu u dokazu teoreme konstruišimo automat koji prihvata jezik $L = \{w \in \Sigma^*_{bool} \mid |w|_0$ je paran broj i $|w|_1$ je neparan broj$\}$.

Uočimo, najpre klase ekvivalencije relacije $\approx_L$. Jasno je da postoje sledeće četiri klase ekvivalencije:

| $A$ | $B$ | $C$ | $D$ |
|---|---|---|---|
| $\|w\|_0$ parno | $\|w\|_0$ neparno | $\|w\|_0$ parno | $\|w\|_0$ neparno |
| $\|w\|_1$ parno | $\|w\|_1$ parno | $\|w\|_1$ neparno | $\|w\|_1$ neparno |

Pa će odgovarajuća stanja automata biti $Q = \{A, B, C, D\}$. Početno stanje mora biti ono gde odgovarajajućoj klasi ekvivalencije pripada $\varepsilon$, tj. $A$, a završno stanje je ono čija odgovarajuća klasa ekvivalencije pripada jeziku, odnosno $C$. Funkcija tranzicije će biti definisana na sledeći način:

| $\delta$ | 0 | 1 |
|---|---|---|
| $A$ | $B$ | $C$ |
| $B$ | $A$ | $D$ |
| $C$ | $D$ | $A$ |
| $D$ | $C$ | $B$ |




PRIMER 2.31. Konstruišimo automat koji prihvata jezik

$$L = \{w \in \Sigma^*_{\text{bool}} \mid |w|_0 = 2 \text{ i } (|w|_1 \geqslant 2 \text{ ili } |w|_1 = 0)\}.$$

Svaki konačni automat $\mathbb{M}$, takav da je $L(\mathbb{M}) = L$ treba da proveri da li je $|w|_0 = 2$, što znači da mora da treba da razlikuje sledeće slučajeve

$$|w|_0 = 0, |w|_0 = 1, |w|_0 = 2 \text{ i } |w|_0 \geqslant 3,$$

za svaku reč $w$. Istovremeno, $\mathbb{M}$ treba da broji i simbole 1 da bi mogao da razlikuje sledeće slučajeve

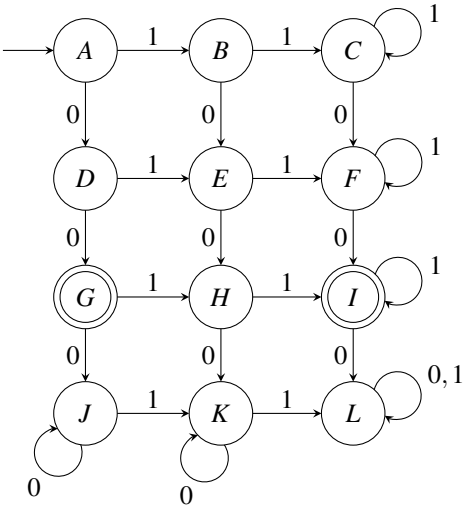
$$|w|_1 = 0, |w|_1 = 1, |w|_1 \geqslant 2.$$

Na osnovu ovih razmatranja naslućujemo da postoje sledeće klase ekvivalencije:

| A | B | C | D | E | F |
|---|---|---|---|---|---|
| $\|w\|_0 = 0$<br>$\|w\|_1 = 0$ | $\|w\|_0 = 0$<br>$\|w\|_1 = 1$ | $\|w\|_0 = 0$<br>$\|w\|_1 \geqslant 2$ | $\|w\|_0 = 1$<br>$\|w\|_1 = 0$ | $\|w\|_0 = 1$<br>$\|w\|_1 = 1$ | $\|w\|_0 = 1$<br>$\|w\|_1 \geqslant 2$ |
| G | H | I | J | K | L |
| $\|w\|_0 = 2$<br>$\|w\|_1 = 0$ | $\|w\|_0 = 2$<br>$\|w\|_1 = 1$ | $\|w\|_0 = 2$<br>$\|w\|_1 \geqslant 2$ | $\|w\|_0 \geqslant 3$<br>$\|w\|_1 = 0$ | $\|w\|_0 \geqslant 3$<br>$\|w\|_1 = 1$ | $\|w\|_0 \geqslant 3$<br>$\|w\|_1 \geqslant 2$ |

Odgovarajuća stanja automata će biti $Q = (\{A, B, \ldots, L\})$. Početno stanje mora biti $A$, s obrirom da $\varepsilon \in A$, dok su završna stanja $G$ i $H$, pošto odgovarajuće klase ekvivalencije pripadaju jeziku. Funkcija trazicije je u ovom slučaju definiše na sledeći način:

| $\delta$ | 0 | 1 |
|---|---|---|
| A | D | B |
| B | E | C |
| C | F | C |
| D | G | E |
| E | H | F |
| F | I | F |
| G | J | H |
| H | K | I |
| I | L | I |
| J | J | K |
| K | K | L |
| L | L | L |




Da bi definisali postupak po kome se već postojeći konačni automat transformiše u odgovarajući minimalni automat uvešćemo još jednu relaciju ekvivalencije:

**Definicija 2.15.** *Za svako $i \geqslant 0$ definišemo relaciju ekvivalencije $\equiv_i$ na skupu stanja $Q$ automata $M = (Q, q_0, F, \Sigma, \delta)$ tako da je $p \equiv_i q$ akko za svaku reč $z \in \Sigma^*$ i $|z| \leqslant i$, $\hat{\delta}(p,z) \in F \Leftrightarrow \hat{\delta}(q,z) \in F$.*

Ova definicija, zapravo kaže da ako važi $p \equiv_i q$ za svako $i$, tada izračunavanja koji prolaze kroz stanja $p$ i $q$ treba da se završe u istom stanju minimalnog konačnog automata za $L(M)$, a u suprotom ova izračunavanja treba da se završe u različitim stanjima minimalnog konačnog automata.

**Lema 2.3.** *Neka je $L = L(M)$. Neka su $p, q \in Q$ i neka je svako stanje automata $M = (Q, q_0, F, \Sigma, \delta)$ dostižno iz početnog stanja. Tada je $p \equiv_i q$ za svako $i \geqslant 0$ akko klase ekvivalencije relacije $\sim_M$ koje odgovaraju stanjima $p$ i $q$ odgovaraju istim klasama ekvivalencije relacije $\approx_L$.*

DOKAZ. Neka je $p \equiv_i q$ za svako $i \geqslant 0$. Tada za ma koje $z \in \Sigma^*$, ako $\hat{\delta}(q_0, x) = p$ i $\hat{\delta}(q_0, y) = q$, pošto je $p \equiv_i q$ za $i = |z|$, znamo da je $\hat{\delta}(q_0, xz) \in F \Leftrightarrow \hat{\delta}(q_0, yz) \in F$, što znači da je $xz \in L \Leftrightarrow yz \in L$. Odakle sledi da ma koje dve reči čijim učitavanjem se prolazi kroz stanja $p$ i $q$ su u istoj klasi ekvivalencije relacije $\approx_L$.

Neka je $p \not\equiv_i q$ za neko $i$. Tada postoji neko $z$, $|z| \leqslant i$ takvo da je samo jedan od $\hat{\delta}(p,z)$ i $\hat{\delta}(q,z)$ u $F$. Obzirom da su oba dostižna iz početnog stanja $q_0$, postoje reči $x$ i $y$ takve da $\hat{\delta}(q_0, x) = p$ i $\hat{\delta}(q_0, y) = q$. To znači da je samo jedan od $\hat{\delta}(q_0, xz)$ i $\hat{\delta}(q_0, yz)$ u $F$, pa samim tim je i samo jedna od reči $xz$ i $yz$ u $L$. To znači da klase ekvivalencije relacije $\sim_M$ koje odgovaraju stanjima $p$ i $q$ ne pripadaju istim klasama ekvivalencije relacije $\approx_L$. □

Na osnovu ovoga, za minimizaciju datog konačnog automata, treba odrediti klase ekvivalencije $\equiv_i$ za svako $i \geqslant 0$, koje će nam definisati ekvivalentna stanja i, samim

tim, suvišna stanja koja treba eliminisati. Algoritam na osnovu kog odredjujemo klase ekvivalencije za svako $\equiv_i$ ima sledeće korake:

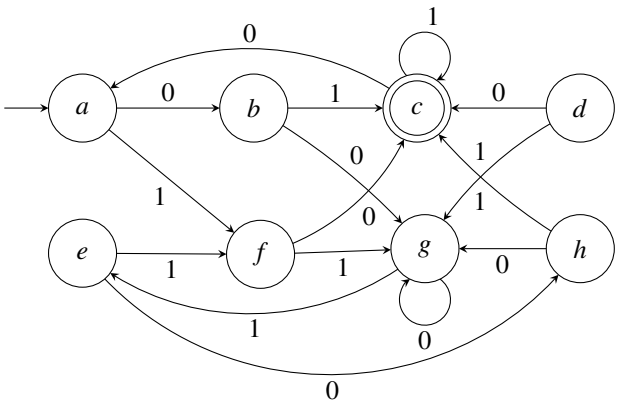
**I korak** Klase ekvivalencije za $\equiv_0$ je lako odrediti i to su skupovi $F$ i $Q \setminus F$.

**II korak** Za date klase ekvivalencije za $\equiv_i$ odredjuju se klase ekvivalencije za $\equiv_{i+1}$: Najpre, stanja koja su istoj klasi ekvivalencije za $\equiv_{i+1}$ moraju pripadati istoj klasi ekvivalencije i za $\equiv_i$, pošto uslov mora da bude ispunjen za sve reči $z$ dužine do $i$. Da bi proverili da li je to zadovoljeno i za reč $z'$ dužine do $i+1$ dovoljno je proveriti da li nakon učitavanja prvog karaktera reči $z'$ se stiže u stanja koja se ekvivalentna za sve reči do dužine $i$. Preciznije, $p \equiv_{i+1} q$ akko $p \equiv_i q$ i za svako $a \in \Sigma$, $\delta(p,a) \equiv_i \delta(q,a)$.

**III korak** Ukoliko se u nekom trenutku dobije da su klase ekvivalencije za $\equiv_{j+1}$ i $\equiv_j$ jednake, sa deljenjem klasa se staje sa zaključkom da nema novih klasa ni za naredne vrednosti $i$. Relacija $\equiv_j$ se zove fiksna tačka.

Za minimizaciju konačnog automata $M$, polazi se od klasa ekvivalencije relacije $\equiv_0$, zatim se izračunavaju $\equiv_{i+1}$ na osnovu $\equiv_i$, dok se ne stigne do fiksne tačke $\equiv_j$. Sva stanja koja se nalaze u istoj klasi eqvivalencije za $\equiv_j$ se sažimaju u jedno stanje.

PRIMER 2.32. Neka je dat automat na slici




Stanje $d$ nije dostižno iz početnog stanja tako može odmah biti uklonjeno. Ostale klase ekvivalencije se dobijaju na sledeći način:

$\equiv_0$: $\{a,b,e,f,g,h\}$, $\{c\}$ - U prvom skup su stanja koja nisu završna, a u drugom žavršna stanja.

Iz prvog skupa stanje $f$ ima 0-ivicu koja vodi ka stanju iz drugog skupa, dok ostala stanja iz prvog skupa imaju 0-ivice ka stanjima iz istog skupa. Stanja $b$ i $h$ imaju 1-ivice koje vode ka stanju iz drugog skupa, dok ostala stanja iz prvog skupa imaju 1-ivice ka stanjima iz istog skupa.

$\equiv_1$: $\{a,e,g\}, \{b,h\}, \{f\}, \{c\}$

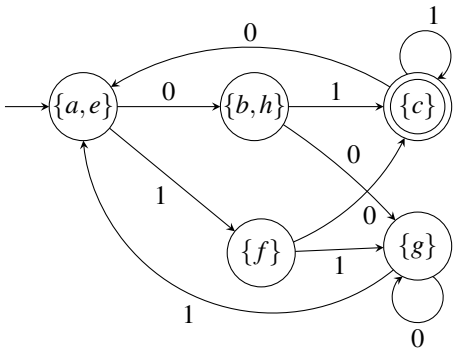
Stanja $a$ i $e$ iz prvog skupa imaju 0-ivice koje vode ka stanjima iz drugog skupa, dok stanje $g$ ima 0-ivicu koja vodi ka stanju iz prvog skupa, takodje, stanja $a$ i $e$ iz prvog skupa imaju 1-ivice koje vode ka stanjima iz trećeg skupa, dok stanje $g$ ima 1-ivicu koja vodi ka stanju iz prvog skupa. U drugom skupu oba stanja imaju 0-ivice ka stanjima iz prvog skupa i 1-ivice ka stanju iz četvrtog skupa.

$\equiv_2$: $\{a,e\}, \{g\}, \{b,h\}, \{f\}, \{c\}$

U prvom skupu oba stanja imaju 0-ivice ka stanju iz trećeg skupa i 1-ivice ka stanju iz četvrtog skupa. U trećem skupu oba stanja imaju 0-ivice ka stanju iz trećeg skupa, i 1-ivice ka stanju iz petog skupa

$\equiv_3$: $\{a,e\}, \{g\}, \{b,h\}, \{f\}, \{c\}$

Obzirom da su klase ekvivalencija relacija $\equiv_2$ i $\equiv_3$ iste, algoritam staje. Odgovarajući minimalni automat ima pet stanja i dat je sledećom slikom




Procedura minimizacije istog automata, koristeći ideju ekvivalentnih stanja, može biti sprovedena i na sledeći način:

Tablicom se predstave parovi stanja, a potom:

**I korak** U tablici se markiraju parovi u kojima je jedno stanje završno, a drugo ne i dobija se sledeća tablica:

| | | | | | | |
|---|---|---|---|---|---|---|
| b | | | | | | |
| c | × | × | | | | |
| e | | | × | | | |
| f | | | × | | | |
| g | | | × | | | |
| h | | | × | | | |
| | a | b | c | e | f | g |

**II korak** U prvom prolazu se ispituju svi nemarkirani parovi na sledeći način: par $(a,b)$ nije markiran i pri tome je $\delta(a,0)=b$ i $\delta(b,0)=g$ i par $(b,g)$ nije markiran, tako da za sada ne zahteva promenu, medjutim $\delta(a,1)=f$ i $\delta(b,1)=c$, a par $(c,f)$ jeste markiran, što znači da i par $(a,b)$ treba da bude markiran. Ispitivanjem svih nemarkiranih parova i markiranjem odgovarajućih dobija se sledeća tablica:

| | | | | | | |
|---|---|---|---|---|---|---|
| b | × | | | | | |
| c | × | × | | | | |
| e | | × | × | | | |
| f | × | × | × | × | | |
| g | | × | × | | × | |
| h | × | | × | × | × | × |
| | a | b | c | e | f | g |

U sledećoj iteraciji se ponovo ispituju preosteli nemarkirani parovi. Par $(a,g)$ će takodje biti markiran, obzirom da je $\delta(a,0)=b$ i $\delta(g,0)=0$, a par $(b,g)$ je markiran. Nakon ove iteracije dobija se sledeća tablica:

| | | | | | | |
|---|---|---|---|---|---|---|
| b | × | | | | | |
| c | × | × | | | | |
| e | | × | × | | | |
| f | × | × | × | × | | |
| g | × | × | × | × | × | |
| h | × | | × | × | × | × |
| | a | b | c | e | f | g |

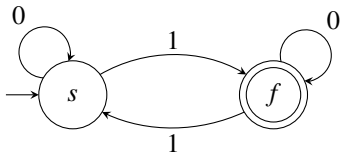
U narednoj iteraciji, u ponovnom ispitivamju nemarkiranih parova, neće biti novih markiranja, pa se ovde korak II završava.

**III korak** Nemarkirani parovi stanja dobijeni na kraju koraka II, predstavljaju ekvivalentna stanja koja se sažimaju u jedno, a će minimalni automat imati stanja $\{a,e\}$, $\{b,h\}$, $\{c\}$, $\{f\}$, $\{g\}$ (ista kao i primenom prethodne metode).

## 2.8 Pojam gramatike; Regularne gramatike

Konačne automate možemo shvatiti kao mehanizme (algoritme, postupke) za ispitivanje da li reč odgovarajućeg alfabeta pripada ili ne nekom regularnom jeziku. Medjutim, automat možemo posmatrati i kao skup pravila koji služi za generisanje reči koje prihvata. To ilustrujemo u narednom primeru.

PRIMER 2.33. Posmatrajmo konačni automat prikazan na slici ispod.




Posmatrajmo ivice datog automata kao pravila:

$$s \xrightarrow{(1)} 0s, \quad s \xrightarrow{(2)} 1f, \quad f \xrightarrow{(3)} 0f, \quad f \xrightarrow{(4)} 1s, \quad f \xrightarrow{(5)} \varepsilon.$$

Nije teško uočiti da automat prihvata reč 10011. Pokušajmo da izvedemo datu reč polazeći od slova $s$, koje označava početno stanje i koristeći navedena pravila:

$$s \xrightarrow{(2)} 1f \xrightarrow{(3)} 10f \xrightarrow{(3)} 100f \xrightarrow{(4)} 1001s \xrightarrow{(2)} 10011f \xrightarrow{(5)} 10011$$

Navodimo nekoliko opštih napomena o pravilima koje smo koristili:

- pravila primenjujemo tako što slovo sa leve strane zamenjujemo parom slova sa desne strane;
- pravilo $(5)$ je *pravilo brisanja*, koje smo dodali jer je $f$ završno stanje;
- u izvodjenju, oznake stanja ($s$ i $f$) predstavljaju *pomoćne simbole* koje menjamo – pa ih smatramo svojevrsnim promenljivama, dok simboli jezika (0 i 1) kada se jednom pojave u izvodjenju ostaju do kraja – pa ih nazivamo *terminalima*.

Prirodno se nameću i neka pitanja. Možemo li analogno izvesti svaku reč koju prihvata dati automat? Da li je moguće izvesti neku reč koju automat ne prihvata?

Odgovore na ova pitanja, u jednom opštem kontekstu, dajemo u ovom odeljku.

**Definicija 2.16.** ***Gramatika*** *je uredjena četvorka* $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$, *pri čemu je:*

- $\Gamma$ *konačan skup simbola koje nazivamo* ***promenljivama*** *(i uglavnom ih obeležavamo velikom slovima);*
- $\Sigma$ *konačan skup simbola, koje nazivamo* ***terminalima****, takav da je* $\Gamma \cap \Sigma = \emptyset$ *(pa za terminale ne koristimo velika slova);*
- $S \in \Gamma$ *je* ***početna promenljiva****;*
- $\mathscr{P} \subseteq V^* \times V^*$, *gde je* $V = \Gamma \cup \Sigma$ *(tzv. alfabet gramatike), jeste* ***skup pravila****, pri čemu svako pravilo* $(u, v) \in \mathscr{P}$ *označavamo* $u \to_{\mathbb{G}} v$ *(izostavljajući indeks* $\mathbb{G}$ *kada je jasno o kojoj gramatici je reč).*

Pravilo $u \to_{\mathbb{G}} v$ primenjujemo na reč $w$, koja sadrži $u$ kao podreč, tako što $u$ zamenimo sa $v$: ako je $w = w_1uw_2$, onda primenom navedenog pravila iz $w$, u gramatici $\mathbb{G}$, direktno izvodimo reč $w_1vw_2$ i pišemo $w_1uw_2 \to_{\mathbb{G}} w_1vw_2$. Izvodjenje je konačan niz direktnih izvodjenja. Kažemo da se iz $w$ izvodi u gramatici $\mathbb{G}$ reč $w'$, i pišemo $w \to^*_{\mathbb{G}} w'$, ako postoji konačan niz reči $w_0, w_1, \ldots, w_k$ (za neko $k \geqslant 0$) takav da je $w = w_0$, $w_k = w'$ i $w_i \to_{\mathbb{G}} w_{i+1}$, za $0 \leqslant i < k$.

**Definicija 2.17.** *Neka je $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ gramatika, jezik*

$$L(\mathbb{G}) = \{w \in \Sigma^* \mid S \to^*_{\mathbb{G}} w\}$$

*je **jezik gramatike** $\mathbb{G}$ ili **jezik generisan gramatikom** $\mathbb{G}$.*

PRIMER 2.34. (1) Neka je $\mathbb{G}_1 = (\{S,A\}, \{0,1\}, S, \mathscr{P}_1)$ gramatika čija su pravila:

$$\mathscr{P}_1: \quad S \to 0S,\ S \to A,\ A \to 1A,\ A \to 1.$$

Tada je $L(\mathbb{G}_1) = \{0^m 1^n \mid m \geqslant 0, n \geqslant 1\}$. Izvodjenje reči $0^2 1^3$ je dato sledećim nizom

$$S \to 0S \to 00S \to 00A \to 001A \to 0011A \to 00111.$$

(2) Neka je $\mathbb{G}_2 = (\{S\}, \{0,1\}, S, \mathscr{P}_2)$ gramatika čija su pravila:

$$\mathscr{P}_2: \quad S \to 0S1,\ S \to 01.$$

Tada je $L(\mathbb{G}_2) = \{0^n 1^n \mid n \geqslant 1\}$. Izvodjenje reči $0^3 1^3$ je dato sledećim nizom

$$S \to 0S1 \to 00S11 \to 000111$$

Primetimo da je $L(\mathbb{G}_1)$ regularan jezik, dok $L(\mathbb{G}_2)$ nije regularan. U nastavku ćemo opisati gramatike koje generišu isključivo regularne jezike.

**Definicija 2.18.** *Gramatika $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ je (desno) **regularna** ako su sva njena pravila obika: $S \to \varepsilon$, $X \to sY$, $X \to s$, za neke $X, Y \in \Gamma$ i neko $s \in \Sigma$. (Naravno, pravilo $S \to \varepsilon$ može, ali ne mora da pripada regularnoj gramatici.)*

**Teorema 2.9.** *Jezik $L \subseteq \Sigma^*$ je regularan akko postoji regularna gramatika $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ takva da je $L = L(\mathbb{G})$.*

DOKAZ. ($\Rightarrow$) Neka je $L$ regularan jezik i $\mathbb{M} = (Q, q_0, F, \Sigma, \delta)$ konačni automat takav da je $L = L(\mathbb{M})$. Definišimo gramatiku $\mathbb{G} = (Q, \Sigma, q_0, \mathscr{P})$, pri čemu je $\mathscr{P}$ najmanji, u smislu inkluzije, skup pravila koji zadovoljava sledeće uslove:

- ako $q_0 \in F$, onda $q_0 \to \varepsilon$ pripada $\mathscr{P}$;
- ako $\delta(q,s) = q'$ i $q' \notin F$, onda $q \to sq'$ pripada $\mathscr{P}$;

- ako $\delta(q,s) = q'$ i $q' \in F$, onda $q \to sq'$ i $q \to s$ pripadaju $\mathscr{P}$.

Indukcijom se dokazuje da za svako $w \in \Sigma^*$ važi:

$$\widehat{\delta}(q_0, w) \in F \text{ akko } q_0 \to_{\mathbb{G}}^* w,$$

odakle sledi da je $L = L(\mathbb{G})$.

($\Leftarrow$) Neka je $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ regularna gramatika. Dokažimo da je $L(\mathbb{G})$ regularan jezik, tj. konstruišimo automat $\mathbb{M}$ takav da je $L(\mathbb{G}) = L(\mathbb{M})$.

Neka je $Q = \Gamma \cup \{Z\}$, gde je $Z$ neki simbol koji ne pripada $\Gamma$ i

$$F = \begin{cases} \{Z\}, & \text{ako } S \to \varepsilon \text{ ne pripada } \mathscr{P}, \\ \{S, Z\}, & \text{ako } S \to \varepsilon \text{ pripada } \mathscr{P}. \end{cases}$$

Funkciju $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ definišemo na sledeći način:

- $\delta(Z, s) = \emptyset$, za sve $s \in \Sigma$,
- za $X \in \Gamma$ i $s \in \Sigma$,
  - ako $X \to s \notin \mathscr{P}$, onda $\delta(X, s) = \{Y \in \Gamma \mid X \to sY \in \mathscr{P}\}$;
  - ako $X \to s \in \mathscr{P}$, onda $\delta(X, s) = \{Y \in \Gamma \mid X \to sY \in \mathscr{P}\} \cup \{Z\}$.

Za nedeterministički automat $\mathbb{M} = (Q, S, F, \Sigma, \delta)$ i svaku reč $w \in \Sigma^*$ važi:

$$S \to_{\mathbb{G}}^* w \text{ akko } \widehat{\delta}(S, w) \in F,$$

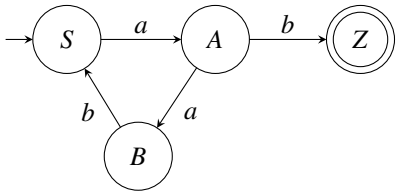
odakle sledi $L(\mathbb{G}) = L(\mathbb{M})$. □

PRIMER 2.35. Neka je data gramatika $G = (\{S, A, B\}, \{a, b\}, S, \mathscr{P})$ gde je

$$\mathscr{P} : S \to aA,\ A \to aB \mid b,\ B \to bS.$$

Koristeći dokaz prethodne teoreme automat koji prihvata jezik $L(G)$ je dat sa $M = (\{S, A, B, Z\}, S, \{Z\}, \{a, b\}, \delta)$ pri čemu je funkcija $\delta$ data sa

| $\delta$ | $a$ | $b$ |
|---|---|---|
| $S$ | $\{A\}$ | $\emptyset$ |
| $A$ | $\{B\}$ | $\{Z\}$ |
| $B$ | $\emptyset$ | $\{S\}$ |
| $Z$ | $\emptyset$ | $\emptyset$ |



# Glava 3

# Kontekstno slobodni jezici

Dalja pažnja će biti usmerena na klasu jezika koja je šira od klase regularnih jezika, na kontekstno slobodne jezike. Kontekstno slobodne gramatike predstavljaju prirodnu, rekurzivnu notaciju ovih jezika. One igraju glavnu ulogu u razvoju kompajlera još od 1960tih godina. Koristeći kontekstno slobodne gramatike postalo je lako definisati parsere, tj. funkcije koje proveravaju strukturu ulaza, čime se posao koji je bio veoma vremenski zahtevan, sveo na rutinski posao koji može da se uradi za kratko vreme.U skorije vreme, kontekstno slobodne gramatike se koriste za opise formata dokumenata (*documet-type definition*, DTD) koji se koriste u XML zajednici za razmenu informacije na vebu.

## 3.1 Kontekstno slobodne gramatike

**Definicija 3.1.** *Gramatika* $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ *je kontekstno-slobodna ako su sva njena pravila oblika* $X \to w$*, za neko* $X \in \Gamma$ *i neku reč* $w \in (\Gamma \cup \Sigma)^*$.

OZNAKE I TERMINI. 1. Za pravilo $X \to w$, kažemo da je $X$ *glava* ovog pravila, a $w$ telo *pravila*. Ukoliko u gramatici postoji više pravila sa istom glavom:

$$X \to w_1,\ X \to w_2, , \ldots, X \to w_k,$$

onda ta pravila kraće zapisujemo:

$$X \to w_1 \mid w_2 \mid \cdots \mid w_k.$$

2. Pravilo $X \to w$ je *linearno* ako je $w = uYv$, za neke $u, v \in \Sigma^*$ i $Y \in \Gamma$, tj. ako sa desne strane sadrži samo jednu promenljivu. Specijalno, pravila oblika $X \to Y$, $X, Y \in \Gamma$, nazivaju se *lančasta* pravila. Pravila oblika $X \to \varepsilon$, $X \in \Gamma$ nazivamo *pravilima brisanja*.

**Definicija 3.2.** *Jezik $L \subseteq \Sigma^*$ je kontekstno-slobodan ako postoji kontekstno slobodna gramatika $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ takva da je $L = L(\mathrm{G})$.*

Očigledno, svaki regularan jezik je kontekstno slobodan, a obrnuto nije tačno. (Dati komentare u vezi sa pravilima brisanja ...)

PRIMER 3.1. 1) Palindromi nad $\{0,1\}$

Neka je $\mathbb{G} = (\{S\}, \{0,1\}, S, \mathscr{P})$, gde je

$$\mathscr{P}: \quad S \to 0S0 \mid 1S1 \mid 0 \mid 1.$$

Nije teško uočiti da je $L(\mathbb{G})$ skup svih palindroma nad $\{0,1\}$ (bez prazne reči). Pravila date gramatike u potpunosti odslikavaju induktivnu sledeću definiciju skupa palindroma koji ćemo označiti sa $S$:

- 0 i 1 su palindromi, tj. $0, 1 \in S$ (što odgovara pravilima $S \to 0 \mid 1$);
- ako je $w$ palindrom, onda su to i $0w0$ i $1w1$, tj. ako $w \in S$, onda $0w0, 1w1 \in S$ (što odgovara pravilima $S \to 0S0 \mid 1S1$).

2) Dajkov jezik je generisan gramatikom $\mathbb{G} = (\{S\}, \{(,)\}, S, \mathscr{P})$, gde je

$$\mathscr{P}: \quad S \to SS \mid (S) \mid \varepsilon.$$

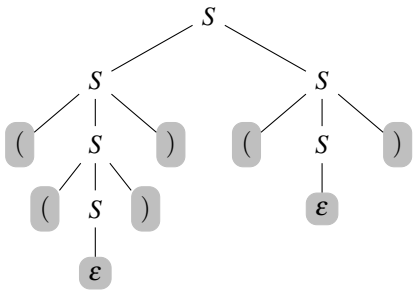
Izvedimo u ovoj gramatici reč $(())()$:

$$S \to SS \to (S)S \to ((S))S \to ((S))(S) \to (())(S) \to (())().$$

Naravno, do iste reči dolazimo ako neznatno izmenimo prethodno izvodjenje:

$$S \to SS \to S(S) \to (S)(S) \to ((S))(S) \to ((S))() \to (())().$$

Prirodno je ova dva izvodjenja reči $(())()$ smatrati istim, te je pogodno prikazivati ih u obliku tzv. *drveta* izvodjenja, koja ćemo kasnije preciznije opisati. Za sada naglašavamo samo naredno drvo reprezentuje oba izvodjenja (koja su strogo uzevši različita).

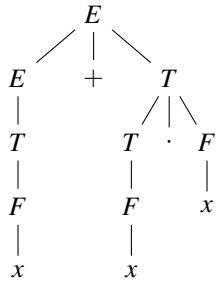


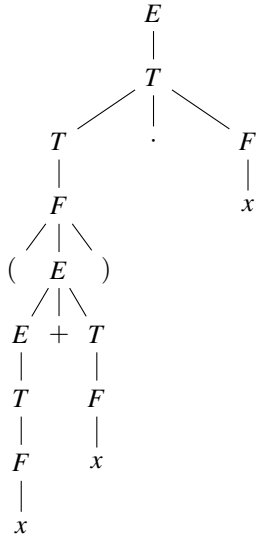
Nije teško uočiti da $L(\mathbb{G})$ sadrži sve korektno izgradjene nizove levih i desnih zagrada.

3) Neka je $\mathbb{G} = (\{E,T,F\},\{x,+,\cdot,(,)\},E,\mathscr{P})$, gde je

$$\mathscr{P}: \quad E \to E+T \mid T, \quad T \to T\cdot F \mid F, \quad F \to (E) \mid x.$$

Drvo desno predstavlja izvodjenje reči (izraza) $(x+x)\cdot x$. Drugo drvo izvodjenja daje $x+x\cdot x$.




Prikazivanje izvodjenja u kontekstno slobodnim gramatikama drvetom potpuno je prirodno budući da dinamiku izvodjenja odredjuju promenljive kojih može biti više od jedne sa desne strane pravila, a da pri tome nije važan redosled kojim ćemo svaku od njih, u nastavku izvodjenja, aktivirati. Uopšte, ako je $\mathbb{G} = (\Gamma,\Sigma,S,\mathscr{P})$ kontekstno slobodna gramatika i $X \in \Gamma$, $X$-**drvo izvodjenja** je drvo čiji su čvorovi označeni elementima iz $\Gamma \cup \Sigma \cup \{\varepsilon\}$ i važi:

- koren drveta je označen sa $X$,
- listovi (čvorovi koji nemaju naslednike) su označeni terminalima ili $\varepsilon$, a unutrašnji čvorovi (koji imaju naslednike, pa nisu listovi) promenljivama,
- ako je neki unutrašnji čvor označen promenljivom $Y$, a njegovi neposredni sledbenici simbolima $s_1,\ldots,s_k \in \Gamma\cup\Sigma$ (redom sleva nadesno), onda $Y \to s_1\cdots s_k \in \mathscr{P}$,
- ako je list označen sa $\varepsilon$ neposredni sledbenik nekog unutrašnjeg čvora $Y$, onda je on i jedini neposredni sledbenik tog čvora.

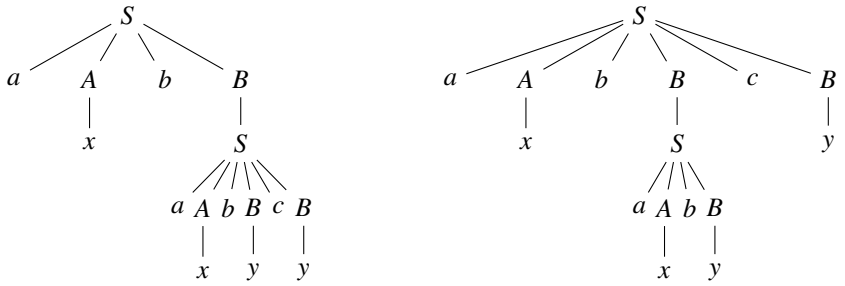
Rezultat nekog drveta izvodjenja jeste reč koja se dobija dopisivanjem sleva nadesno listova tog drveta. Primetimo da svaki unutrašnji čvor nekog drveta izvodjenja zajedno sa svim svojim sledbenicima obrazuje novo drvo izvodjenja.

Svakom izvodjenju u kontekstno slobodnoj gramatici odgovara jedinstveno drvo izvodjenja, dok obrnuto nije tačno, što je ilustrovano u primeru 3.1. Medjutim, ako se ograničimo samo na tzv. *leva izvodjenja*, dobijamo obostrano jednoznačnu korespondenciju izmedju izvodjenja i drveta. Leva izvodjenja su ona u čijem je svakom koraku $w \to_{\mathbb{G}} w'$ primenjeno pravilo čija je glava najlevlja promenljiva iz $w$. Može se dokazati da svaku kontekstnu slobodnu gramatiku $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ važi: $S \to^*_{\mathbb{G}} w$ akko postoji levo izvodjenje reči $w$ iz $S$. Naravno, do istih zaključaka bismo došli da smo posmatrali desna izvodjenja.

PRIMER 3.2. Neka je $\mathbb{G} = (\{S,A,B\}, \{a,b,c,x,y\}, S, \mathscr{P})$, gde je

$$\mathscr{P}: \quad S \to aAbB \mid aAbBcB, \quad A \to x, \quad B \to S \mid y.$$

U ovoj gramatici reč *axbaxbycy* možemo izvesti na dva načina kojima odgovaraju dva različita drveta izvodjenja.




**Definicija 3.3.** *Kontekstno slobodna gramatika $\mathbb{G}$ je* ***dvosmislena*** *ako postoji reč $w \in L(\mathbb{G})$ koja ima dva različita drveta izvodjenja u $\mathbb{G}$. U suprotnom, gramatika je* ***nedvosmislena****.*

**Definicija 3.4.** *Kontekstno slobodni jezik je* ***suštinski dvosmislen*** *ako je dvosmislena svaka gramatika koja ga generiše.*

PRIMER 3.3. Može se dokazati da je jezik

$$\{a^m b^m c^n d^n \mid m,n \geqslant 1\} \cup \{a^m b^n c^n d^m \mid m,n \geqslant 1\}$$

suštinski dvosmislen.

Iako regularni izrazi mogu da se koriste u nekim segmenitma kreiranja interpretera, kao u ranije navedenom primeru za prepoznavanje promenljivih, to nikako nije dovoljan alat. Već primeri uparivanja zagrada ili prepoznavanja aritmetičkih izraza prevazilaze mogućnosti regularnih izraza. Ovi problemi, kao što je pokazano, se mogu lako opisati gramatičkim pravilima kontekstno slobodne gramatike. Problem uparivanje vitičastih zagrada ({ i }) za ograničavanje blokova u programskom jeziku C ili uparivanje BEGIN i END komandi u programskom jeziku PASCAL je potuno analogan problemu uparivanja zagrada. Sličan je i sledeći primer koji proverava istpavnu upotrebu IF-ELSE konstrukcije

PRIMER 3.4. Gramatičko pravilo koje generiše ispravne nizove IF i ELSE reči (predstavljenih sa *i* i *e*) može se opisati na sledeći način:

$$S \to \varepsilon \mid SS \mid iS \mid iSeS.$$

Na osnovu ovog pravila mogu se dobiti nizovi *ieie*, *iie*, *iei* i slično, dok nižovi *ei* ili *ieeii* nisu korektni i ne mogu biti generisani datim pravlom.

Programi koji vrše proveru ispravne strukturanosti i dalju obradu ispravnih ulaznih podataka, zasnovani na konteksno slobodnim gramtikama zovu se **parseri**. Jedan od najpopularnijih generatora parsera je YACC. Njegova popularnost je posledica činjenica da je osnovna verzija YACC deo operativnog sistema UNIX i pri tome omogućava da se delovi programskog koda pišu u programskom jeziku C. Zahvaljujući popularnosti, razvijene su i verzije koje rade pod drugim operativnim sistemima i u različitim okruženima omogućavaju povezivanje i sa drugim programsim jezicima, pre svega sa C++ i C#.

Parseri, a samim tim i kontekstno slobodne gramatike su, takodje, vrlo zgodan alat za prepoznavanje i prihvatanje strukture definisane markap jezicima poput HTML i XML. Dokumenti koji sadže informacije opisane ovom vrstom jezika imaju precizno definisanu strukturu, za koju je jednostavno definisati gramatičko pravilo koje je prepoznaje ispravno opisani sadržaj i formira stablo izvodjenja koje će biti dalje procesirano.

## 3.2 Normalna forma Čomskog

U ovom odeljku, pokazaćemo da se svaka kontekstno slobodna gramatika može transformisati u gramatiku koja generiše isti jezik ali čija su pravila veoma jednostavnog oblika.

Neka je $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ kontekstno slobodna gramatika. Promenljiva $X \in \Gamma$ je

- **završavajuća** ako postoji reč $w \in \Sigma^*$ takva da $X \to^*_{\mathbb{G}} w$;
- **dostižna** ako postoje $u, v \in (\Gamma \cup \Sigma)^*$ takve da $S \to^*_{\mathbb{G}} uXv$.

**Definicija 3.5.** *Kontekstno slobodna gramatika je* ***redukovana*** *ako su sve njene promenljive završavajuće i dostižne.*

Pre nego što dokažemo da svaku kontekstno slobodnu gramatiku $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ može transformisati u ekvivalentnu redukovanu (teorema 3.1) gramatiku, primetimo da se za $\mathbb{G}$ mogu efektivno odrediti:

- skup $\mathbb{Z}$ svih završavajućih promenljivih i

- skup $\mathcal{D}$ svih dostižnih promenljivih.

Definišimo skupove $\mathcal{Z}_i$ i $\mathcal{D}_i$, $i \geqslant 0$, induktivno, na sledeći način:
$\mathcal{Z}_0 = \{X \in \Gamma \mid (\exists w \in \Sigma^*) X \to w \in \mathscr{P}\}$,
$\mathcal{Z}_{i+1} = \{X \in \Gamma \mid (\exists w \in (\mathcal{Z}_i \cup \Sigma)^*) X \to w \in \mathscr{P}\}$, $i \geqslant 0$;
$\mathcal{D}_0 = \{S\}$,
$\mathcal{D}_{i+1} = \mathcal{D}_i \cup \{X \in \Gamma \mid (\exists Y \in \mathcal{D}_i)(\exists u, v \in (\Gamma \cup \Sigma)^*) Y \to uXv \in \mathscr{P}\}$, $i \geqslant 0$.

Tada važe sledeće osobine:

| | |
|---|---|
| 1) $\mathcal{Z}_0 \subseteq \mathcal{Z}_1 \subseteq \mathcal{Z}_2 \subseteq \cdots$ | $\mathcal{D}_0 \subseteq \mathcal{D}_1 \subseteq \mathcal{D}_2 \subseteq \cdots$ |
| 2) $\mathcal{Z}_i = \mathcal{Z}_{i+1} \Rightarrow \mathcal{Z}_i = \mathcal{Z}_{i+1} = \mathcal{Z}_{i+2} = \cdots$ | $\mathcal{D}_i = \mathcal{D}_{i+1} \Rightarrow \mathcal{D}_i = \mathcal{D}_{i+1} = \mathcal{D}_{i+2} = \cdots$ |
| 3) $\mathcal{Z}_i = \mathcal{Z}_{i+1}$, za $i = \lvert\Gamma\rvert - 1$ | $\mathcal{D}_i = \mathcal{D}_{i+1}$, za $i = \lvert\Gamma\rvert - 1$ |
| 4) $\mathcal{Z} = \bigcup_{i \geqslant 0} \mathcal{Z}_i$ | $\mathcal{D} = \bigcup_{i \geqslant 0} \mathcal{D}_i$ |

odakle sledi da se $\mathcal{Z}$ i $\mathcal{D}$ mogu efektivno odrediti.

NAPOMENA. Problem $L(\mathbb{G}) \neq \emptyset$ je odlučiv, jer je ekvivalentan sa $S \in \mathcal{Z}$.

**Teorema 3.1.** *Za svaku kontekstno slobodnu gramatiku $\mathbb{G}$, takvu da je $L(\mathbb{G}) \neq \emptyset$, postoji redukovana kontekstno slobodna gramatika $\mathbb{G}^{\mathrm{r}}$ takva da je $L(\mathbb{G}) = L(\mathbb{G}^{\mathrm{r}})$.*

DOKAZ. Gramatiku $\mathbb{G}^{\mathrm{r}}$ konstruišemo u dve etape.

1. ETAPA. Odredimo najpre skup $\mathcal{Z}$ svih završavajućih promenljivih gramatike $\mathbb{G}$. Neka je $\mathbb{G}' = (\Gamma', \Sigma, S, \mathcal{P}')$ gramatika takva da je:

- $\Gamma' = \mathcal{Z}$ ($S \in \mathcal{Z}$, jer je $L(\mathbb{G}) \neq \emptyset$);
- $\mathcal{P}'$ dobijeno iz $\mathcal{P}$ izbacivanjem svih pravila u kojima se pojavljuju (sa bilo koje strane) promenljive iz $\Gamma \setminus \mathcal{Z}$.

Nije teško zaključiti da je $L(\mathbb{G}) = L(\mathbb{G}')$.

2. ETAPA. Odredimo najpre skup $\mathcal{D}'$ svih dostižnih promenljivih gramatike $\mathbb{G}'$. Neka je $\mathbb{G}^{\mathrm{r}} = (\Gamma^{\mathrm{r}}, \Sigma, S, \mathcal{P}^{\mathrm{r}})$ gramatika takva da je:

- $\Gamma^{\mathrm{r}} = \mathcal{D}'$ (primetimo da $S \in \Gamma^{\mathrm{r}}$);
- $\mathcal{P}^{\mathrm{r}}$ dobijeno iz $\mathcal{P}'$ izbacivanjem svih pravila iz $\mathscr{P}'$ u kojima se pojavljuju (sa bilo koje strane) promenljive iz $\Gamma' \setminus \mathcal{D}'$.

Tada je $L(\mathbb{G}^{\mathrm{r}}) = L(\mathbb{G}')$ i $\mathbb{G}^{\mathrm{r}}$ je tražena gramatika. □

NAPOMENA. Etape u prethodnom dokazu ne mogu zameniti mesta, a evo i zašto. Pretpostavimo da $\mathbb{G}$ ima samo jedno pravilo sa glavom $A$ i neka je to $A \to BC$, pri čemu je $A$ dostižna, $B$ završavajuća, a $C$ nije završavajuća promenljiva. Ako bismo prvo izbacili nedostižne promenljive, onda bi posle prve etape ostale promenljive $A$, $B$ i $C$, dok bi u drugoj etapi bili izbačeni $A$ i $C$, a $B$ bi ostalo. Nasuprot tome, ako prvo izbacimo nezavršavajuće promenljive, izbacićemo ih odmah sve tri.

**Definicija 3.6.** *Kontekstno slobodna gramatika* $(\Gamma, \Sigma, S, \mathscr{P})$ *je u* ***normalnoj formi Čomskog*** *ako su sva njena pravila oblika:*

- $X \to YZ$, $X, Y, Z \in \Gamma$,
- $X \to x$, $X \in \Gamma$, $x \in \Sigma$, *ili*
- $S \to \varepsilon$,

*i pri tome, ako se* $S \to \varepsilon$ *pojavljuje medju pravilima, onda se* $S$ *ne pojavljuje sa desne strane nijednog pravila.*

**Teorema 3.2.** *Za svaku kontekstno slobodnu gramatiku* $\mathbb{G}$ *postoji kontekstno slobodna gramatika* $\mathbb{G}^{\mathrm{CH}}$ *u normalnoj formi Čomskog takva da je* $L(\mathbb{G}) = L(\mathbb{G}^{\mathrm{CH}})$.

DOKAZ. Neka je $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ kontekstno slobodna gramatika. Pretpostavićemo da se $S$ ne pojavljuje u telu nijednog pravila iz $\mathscr{P}$. Ovu pretpostavku slobodno možemo uvesti, jer se promenljivama iz $\Gamma$ može dodati nova promenljiva $S'$, a skupu pravila $\mathscr{P}$ pravilo $S' \to S$ i da se na taj način dobije gramatika

$$(\Gamma \cup \{S'\}, \Sigma, S', \mathscr{P} \cup \{S' \to S\})$$

u kojoj se početna promenljiva ne pojavljuje u telima pravila, a koja generiše isti jezik kao i polazna gramatika $\mathbb{G}$.

Traženu gramatiku u normalnoj formi Čomskog konstruišemo u četiri etape.

1. ETAPA – eliminacija pravila brisanja.

Eliminišemo pravila $X \to \varepsilon$, gde $X$ nije početni simbol.

Za svako pravilo iz $\mathscr{P}$ na čijoj se desnoj strani pojavljuje promenljiva $X$ dodajemo pravila koja se dobijaju izostavljanjem promenljive $X$. Na primer, ako se $X$ pojavljuje samo jednom sa desne strane nekog pravila $Y \to vXw$, onda dodajemo pravilo $Y \to vw$; ako se $X$ pojavljuje dva puta sa desne strane nekog pravila $Y \to uXvXw$ dodajemo pravila $Y \to uvXw$, $Y \to uXvw$ i $Y \to uvw$; itd.

Ako ima pravila $Y \to X$ dodajemo pravilo $Y \to \varepsilon$, ukoliko ono nije prethodno eliminisano. Opisani postupak nastavljamo sve dok ne eliminišemo sva pravila brisanja.

2. ETAPA – uvodjenje novih promenljivih koje odgovaraju terminalima.

U ovoj etapi, za svaki terminal $s \in \Sigma$ dodajemo jednu novu promenljivu $X_s$, a pravila (dobijena u prethodnoj etapi) preradjujemo tako što u svakom pravilu u čijem se telu pojavljuje $s$, terminal $s$ zamenjujemo sa $X_s$. Najzad, ovako preradjenom skupu pravila dodajemo $X_s \to s$, za svaki $s \in \Sigma$.

3. ETAPA – skraćivanje pravila čija tela sadrže više od dve promenljive.

Svako pravilo oblika $X \to Y_1 Y_2 \cdots Y_k$, $k \geqslant 3$, zamenjujemo pravilima

$$X \to Y_1 Z_1,\ Z_1 \to Y_2 Z_2,\ \ldots,\ Z_{k-2} \to Y_{k-1} Y_k,$$

pri čemu su $Z_1, Z_2, \ldots, Z_{k-1}$ nove promenljive.

4. ETAPA – eliminacija lančastih pravila.

Eliminišemo pravila oblika $X \to Y$ i za svako pravilo oblika $Y \to w$ dodajemo pravilo $X \to w$, osim ako to pravilo nije lančasto pravilo koje je već eliminisano.

Ovako preradjena gramatika je u normalnoj formi Čomskog i generiše isti jezik kao polazna gramatika. □

Normalna forma Čomskog znatno pojednostavljuje ispitivanja svojstava kontekstno slobodnih jezika. Pored toga, prema prethodnoj teoremi, konteksno slobodni jezici su generisani veoma jednostavnim drvetima izvodjenja čiji svaki unutrašnji čvor ili ima tačno dva naslednika označena promenljivama ili ima samo jednog naslednika koji je označen terminalom.

PRIMER 3.5. Transformišimo u normalnu formu Čomskog kontekstno slobodnu gramatiku $(\{S,A,B\},\{x,y\},S,\mathscr{P})$, gde je

$$\mathscr{P}: \quad S \to ASA,\ S \to xB,\ A \to B,\ A \to S,\ B \to y,\ B \to \varepsilon.$$

Pošto se $S$ pojavljuje u telima nekih pravila, uvodimo novu početnu promenljivu $S'$ i posmatramo gramatiku čija su pravila:

$$\mathscr{P}': \quad S' \to S,\ S \to ASA,\ S \to xB,\ A \to B,\ A \to S,\ B \to y,\ B \to \varepsilon.$$

1. ETAPA – eliminacija pravila brisanja.

**1.1** – eliminacija pravila $B \to \varepsilon$ iz $\mathscr{P}'$:

$\mathscr{P}': \quad S' \to S,\ S \to ASA,\ S \to xB,\ A \to B,\ A \to S,\ B \to y,\ \underline{B \to \varepsilon}$

$\downarrow$

$\mathscr{P}'_1: \quad S' \to S,\ S \to ASA,\ S \to xB,\ A \to B,\ A \to S,\ B \to y,\ \underline{S \to x,\ A \to \varepsilon}$

**1.2** – eliminacija pravila $A \to \varepsilon$ iz $\mathscr{P}'_1$:

$\mathscr{P}'_1: \quad S' \to S,\ S \to ASA,\ S \to xB,\ A \to B,\ A \to S,\ B \to y,\ S \to x,\ \underline{A \to \varepsilon}$

$\downarrow$

$\mathscr{P}'_2: \quad S' \to S,\ S \to ASA,\ S \to xB,\ A \to B,\ A \to S,\ B \to y,\ S \to x,\ \underline{S \to S,\ S \to AS,\ S \to SA}$

2. ETAPA – uvodjenje novih promenljivih koje odgovaraju terminalima.

Uvodimo nove promenljive $X$ i $Y$ i transformišemo skup pravila $\mathscr{P}'_2$ na sledeći način:

$\mathscr{P}'_3: \quad S' \to S,\ S \to ASA,\ S \to XB,\ A \to B,\ A \to S,\ B \to Y,\ S \to X,\ S \to S,\ S \to AS,\ S \to SA$
$X \to x,\ Y \to y$

3. ETAPA – skraćivanje pravila čija tela sadrže više od dve promenljive.

$\mathscr{P}'_3: \quad S' \to S,\ \underline{S \to ASA},\ S \to XB,\ A \to B,\ A \to S,\ B \to Y,\ S \to X,\ S \to S,\ S \to AS,\ S \to SA$
$X \to x,\ Y \to y$

$$\downarrow$$
$$\mathscr{P}'_4: \left|\begin{array}{lll} S' \to S & S \to XB & X \to x \\ A \to B & S \to AS & Y \to y \\ A \to S & S \to SA & \\ B \to Y & \underline{S \to AZ} & \\ S \to X & \underline{Z \to SA} & \end{array}\right.$$

4. ETAPA – eliminacija lančastih pravila.

**4.1** – eliminacija pravila $S' \to S$ iz $\mathscr{P}'_4$:

$$\mathscr{P}'_5: \left|\begin{array}{lll} & & S \to XB \quad S' \to XB \quad X \to x \\ A \to B & & S \to AS \quad S' \to AS \quad Y \to y \\ A \to S & & S \to SA \quad S' \to SA \\ B \to Y & & S \to AZ \quad S' \to AZ \\ S \to X & S' \to X & Z \to SA \end{array}\right.$$

**4.2** – eliminacija pravila $A \to B$ iz $\mathscr{P}'_5$:

$$\mathscr{P}'_6: \left|\begin{array}{lll} & & S \to XB \quad X \to x \\ & & S \to AS \quad Y \to y \\ A \to S & & S \to SA \\ B \to Y & A \to Y & S \to AZ \\ S \to X & & Z \to SA \\ S' \to X & & S' \to XB \\ & & S' \to AS \\ & & S' \to SA \\ & & S' \to AZ \end{array}\right.$$

**4.3** – eliminacija pravila $A \to S$ iz $\mathscr{P}'_6$:

$$\mathscr{P}'_7: \left|\begin{array}{lll} & & S \to XB \quad A \to XB \quad X \to x \\ & & S \to AS \quad A \to AS \quad Y \to y \\ & & S \to SA \quad A \to SA \\ B \to Y & & S \to AZ \quad A \to AZ \\ S \to X & A \to X & Z \to SA \\ S' \to X & & S' \to XB \\ A \to Y & & S' \to AS \\ & & S' \to SA \\ & & S' \to AZ \end{array}\right.$$

**4.4** – eliminacija pravila $B \to Y$ iz $\mathscr{P}'_7$:

$\mathscr{P}'_8$ :

| | | | |
|---|---|---|---|
| | $S \to XB$ | $X \to x$ | |
| | $S \to AS$ | $Y \to y$ | $B \to y$ |
| | $S \to SA$ | | |
| | $S \to AZ$ | | |
| $S \to X$ | $Z \to SA$ | | |
| $S' \to X$ | $S' \to XB$ | | |
| $A \to Y$ | $S' \to AS$ | | |
| $A \to X$ | $S' \to SA$ | | |
| | $S' \to AZ$ | | |
| | $A \to XB$ | | |
| | $A \to AS$ | | |
| | $A \to SA$ | | |
| | $A \to AZ$ | | |

**4.5** – eliminacija pravila $S \to X$ iz $\mathscr{P}'_8$:

$\mathscr{P}'_9$ :

| | | | |
|---|---|---|---|
| | $S \to XB$ | $X \to x$ | $S \to x$ |
| | $S \to AS$ | $Y \to y$ | |
| | $S \to SA$ | $B \to y$ | |
| | $S \to AZ$ | | |
| | $Z \to SA$ | | |
| $S' \to X$ | $S' \to XB$ | | |
| $A \to Y$ | $S' \to AS$ | | |
| $A \to X$ | $S' \to SA$ | | |
| | $S' \to AZ$ | | |
| | $A \to XB$ | | |
| | $A \to AS$ | | |
| | $A \to SA$ | | |
| | $A \to AZ$ | | |

**4.6** – eliminacija pravila $S' \to X$ iz $\mathscr{P}'_9$:

$\mathscr{P}'_{10}$ :

| | | | |
|---|---|---|---|
| | $S \to XB$ | $X \to x$ | $S' \to x$ |
| | $S \to AS$ | $Y \to y$ | |
| | $S \to SA$ | $B \to y$ | |
| | $S \to AZ$ | $S \to x$ | |
| | $Z \to SA$ | | |
| | $S' \to XB$ | | |
| $A \to Y$ | $S' \to AS$ | | |
| $A \to X$ | $S' \to SA$ | | |
| | $S' \to AZ$ | | |
| | $A \to XB$ | | |
| | $A \to AS$ | | |
| | $A \to SA$ | | |
| | $A \to AZ$ | | |

**4.7** – eliminacija pravila $A \to Y$ iz $\mathscr{P}'_{10}$:

| $\mathscr{P}'_{11}$ : | | | | |
|---|---|---|---|---|
| | | $S \to XB$ | $X \to x$ | |
| | | $S \to AS$ | $Y \to y$ | $A \to y$ |
| | | $S \to SA$ | $B \to y$ | |
| | | $S \to AZ$ | $S \to x$ | |
| | | $Z \to SA$ | $S' \to x$ | |
| | | $S' \to XB$ | | |
| | | $S' \to AS$ | | |
| | $A \to X$ | $S' \to SA$ | | |
| | | $S' \to AZ$ | | |
| | | $A \to XB$ | | |
| | | $A \to AS$ | | |
| | | $A \to SA$ | | |
| | | $A \to AZ$ | | |

**4.8** – eliminacija pravila $A \to X$ iz $\mathscr{P}'_{11}$:

| $\mathscr{P}'_{12}$ : | | | |
|---|---|---|---|
| | $S \to XB$ | $X \to x$ | $A \to x$ |
| | $S \to AS$ | $Y \to y$ | |
| | $S \to SA$ | $B \to y$ | |
| | $S \to AZ$ | $S \to x$ | |
| | $Z \to SA$ | $S' \to x$ | |
| | $S' \to XB$ | $A \to y$ | |
| | $S' \to AS$ | | |
| | $S' \to SA$ | | |
| | $S' \to AZ$ | | |
| | $A \to XB$ | | |
| | $A \to AS$ | | |
| | $A \to SA$ | | |
| | $A \to AZ$ | | |

**Lema 3.1.** *Ako je $G$ kontekstno slobodna gramatika u normalnoj formi Čomskog, onda za svaku reč $w \in L(G)$ takvu da je $|w| \geqslant 1$ potrebno je tačno $2|w|-1$ koraka za bilo koje izvodjenje reči $w$ u gramatici $G$.*

## 3.3 Parsiranje u kontekstno slobodnim gramatikama

Za kontekstno slobodne gramatike $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ u normalnoj formi Čomskog postoji jednostavan algoritam kojim se rešava problem pripadanja: za zadatu reč $w \in \Sigma^*$ ispitati da li $w \in L(\mathbb{G})$ ili ne. Naravno, poželjno bi bilo da ukoliko $w \in L(\mathbb{G})$ algoritam proizvede i odgovarajuće drvo izvodjenja.

Prikazaćemo tzv. CYK-algoritam (Kuk-Janger-Kasami algoritam).

Ako je $w = \varepsilon$, treba samo proveriti da li $S \to \varepsilon \in \mathscr{P}$.

Neka je $w = s_1 \cdots s_n \in \Sigma^+$ i $w[i,j] = s_i \cdots s_j$, $1 \leqslant i \leqslant j \leqslant n$. Pomenuti algoritam je zasnovan na sledećem zapažanju: provera da li je $X \to^* w[i,j]$ svodi se na sledeće slučajeve:

- ako je $i = j$, treba proveriti da li $X \to s_i \in \mathscr{P}$;
- ako je $i < j$, prvi korak odgovarajućeg izvodjenja mora biti $X \to YZ$, za neke promenljive $Y$ i $Z$, što dalje znači da postoji $k$, $i \leqslant k < j$ takvo da $Y \to^* w[i,k]$ i $Z \to^* w[k+1,j]$.

CYK-algoritam

INPUT: $(\Gamma, \Sigma, S, \mathscr{P})$ – gramatika u normalnoj formi Čomskog;
$w = s_1 \cdots s_n$ – reč iz $\Sigma^*$
for $i := 1$ to $n$ do
$U[i,i] := \{X \in \Gamma \mid X \to s_i \in \mathscr{P}\}$
for $d := 1$ to $n-1$ do;
for $i := 1$ to $n-d$ do
$j := i+d$
$U[i,j] := \emptyset$
for $k := i$ to $j-1$ do
$U[i,j] := U[i,j] \cup \{X \in \Gamma \mid X \to YZ \in \mathscr{P}, Y \in U[i,k], Z \in U[k+1,j]\}$
OUTPUT: $U[i,j]$ ($U[i,j]$ sadrži sve promenljive $X$ takve da $X \to^* s_i \cdots s_j$)
if $S \in U[1,n]$ then
return TRUE
else
return FALSE

PRIMER 3.6. Ilustrujmo navedeni algoritam za ulaz: $\mathbb{G} = (\{S,A,B,C\}, \{a,b,c\}, S, \mathscr{P})$

$$\mathscr{P}: \quad S \to AB \mid b,\ A \to CB \mid AA \mid a,\ B \to AS \mid b,\ C \to BS \mid c$$

i $w = cabab \in L(\mathbb{G})$.

| $i \diagdown j$ | $\frac{c}{1}$ | $\frac{a}{2}$ | $\frac{b}{3}$ | $\frac{a}{4}$ | $\frac{b}{5}$ |
|---|---|---|---|---|---|
| $c \mid 1$ | | | | | |
| $a \mid 2$ | $\times$ | | | | |
| $b \mid 3$ | $\times$ | $\times$ | | | |
| $a \mid 4$ | $\times$ | $\times$ | $\times$ | | |
| $b \mid 5$ | $\times$ | $\times$ | $\times$ | $\times$ | |

Najpre popunjavamo dijagonalu tabele, tj. nalazimo skupove $U[i,i]$, $i = 1, \ldots, 5$:
$U[1,1] = \{X \in \{S,A,B,C\} \mid X \to c \in \mathscr{P}\} = \{C\}$,

$U[2,2] = U[4,4] = \{X \in \{S,A,B,C\} \mid X \to a \in \mathscr{P}\} = \{A\}$,
$U[3,3] = U[5,5] = \{X \in \{S,A,B,C\} \mid X \to b \in \mathscr{P}\} = \{S,B\}$.

| $i \diagdown j$ | $\underset{1}{c}$ | $\underset{2}{a}$ | $\underset{3}{b}$ | $\underset{4}{a}$ | $\underset{5}{b}$ |
|---|---|---|---|---|---|
| $c \mid 1$ | $C$ | | | | |
| $a \mid 2$ | $\times$ | $A$ | | | |
| $b \mid 3$ | $\times$ | $\times$ | $S,B$ | | |
| $a \mid 4$ | $\times$ | $\times$ | $\times$ | $A$ | |
| $b \mid 5$ | $\times$ | $\times$ | $\times$ | $\times$ | $S,B$ |

Zatim popunjavamo dijagonalu iznad upravo pounjene:

$$\begin{aligned} U[1,2] &= \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[1,1], Z \in U[2,2]\} \\ &= \{X \in \{S,A,B,C\} \mid X \to CA\} = \emptyset \end{aligned}$$

$$\begin{aligned} U[2,3] &= \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[2,2], Z \in U[3,3]\} \\ &= \{X \in \{S,A,B,C\} \mid X \to AS \text{ ili } X \to AB\} = \{S,B\} \end{aligned}$$

$$\begin{aligned} U[3,4] &= \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[3,3], Z \in U[4,4]\} \\ &= \{X \in \{S,A,B,C\} \mid X \to SA \text{ ili } X \to BA\} = \emptyset \end{aligned}$$

$$\begin{aligned} U[4,5] &= \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[4,4], Z \in U[5,5]\} \\ &= \{X \in \{S,A,B,C\} \mid X \to AS \text{ ili } X \to AB\} = \{S,B\} \end{aligned}$$

| $i \diagdown j$ | $\underset{1}{c}$ | $\underset{2}{a}$ | $\underset{3}{b}$ | $\underset{4}{a}$ | $\underset{5}{b}$ |
|---|---|---|---|---|---|
| $c \mid 1$ | $C$ | $\emptyset$ | | | |
| $a \mid 2$ | $\times$ | $A$ | $S,B$ | | |
| $b \mid 3$ | $\times$ | $\times$ | $S,B$ | $\emptyset$ | |
| $a \mid 4$ | $\times$ | $\times$ | $\times$ | $A$ | $S,B$ |
| $b \mid 5$ | $\times$ | $\times$ | $\times$ | $\times$ | $S,B$ |

$$\begin{aligned} U[1,3] &= \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[1,1], Z \in U[2,3]\} \cup \\ &\quad \cup \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[1,2], Z \in U[3,3]\} \\ &= \{X \in \{S,A,B,C\} \mid X \to CS \text{ ili } X \to CB\} = \{A\} \end{aligned}$$

$$\begin{aligned} U[2,4] &= \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[2,2], Z \in U[3,4]\} \cup \\ &\quad \cup \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[2,3], Z \in U[4,4]\} \\ &= \{X \in \{S,A,B,C\} \mid X \to SA \text{ ili } X \to BA\} = \emptyset \end{aligned}$$

$$\begin{aligned} U[3,5] &= \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[3,3], Z \in U[4,5]\} \cup \\ &\quad \cup \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[3,4], Z \in U[5,5]\} \\ &= \{X \in \{S,A,B,C\} \mid X \to SS \text{ ili } X \to SB \text{ ili } X \to BS \text{ ili } X \to BB\} = \{C\} \end{aligned}$$

| $i \diagdown j$ | $\frac{c}{1}$ | $\frac{a}{2}$ | $\frac{b}{3}$ | $\frac{a}{4}$ | $\frac{b}{5}$ |
|---|---|---|---|---|---|
| $c \mid 1$ | $C$ | $\emptyset$ | $A$ | | |
| $a \mid 2$ | $\times$ | $A$ | $S,B$ | $\emptyset$ | |
| $b \mid 3$ | $\times$ | $\times$ | $S,B$ | $\emptyset$ | $C$ |
| $a \mid 4$ | $\times$ | $\times$ | $\times$ | $A$ | $S,B$ |
| $b \mid 5$ | $\times$ | $\times$ | $\times$ | $\times$ | $S,B$ |

$$\begin{aligned} U[1,4] &= \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[1,1], Z \in U[2,4]\} \cup \\ &\quad \cup \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[1,2], Z \in U[3,4]\} \\ &\quad \cup \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[1,3], Z \in U[4,4]\} \\ &= \{X \in \{S,A,B,C\} \mid X \to AA\} = \{A\} \end{aligned}$$

$$\begin{aligned} U[2,5] &= \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[2,2], Z \in U[3,5]\} \cup \\ &\quad \cup \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[2,3], Z \in U[4,5]\} \\ &\quad \cup \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[2,4], Z \in U[5,5]\} \\ &= \{X \in \{S,A,B,C\} \mid X \to AC \text{ ili } X \to SS \text{ ili } X \to SB \text{ ili } X \to BS \text{ ili } X \to BB\} = \{C\} \end{aligned}$$

| $i \diagdown j$ | $\frac{c}{1}$ | $\frac{a}{2}$ | $\frac{b}{3}$ | $\frac{a}{4}$ | $\frac{b}{5}$ |
|---|---|---|---|---|---|
| $c \mid 1$ | $C$ | $\emptyset$ | $A$ | $A$ | |
| $a \mid 2$ | $\times$ | $A$ | $S,B$ | $\emptyset$ | $C$ |
| $b \mid 3$ | $\times$ | $\times$ | $S,B$ | $\emptyset$ | $C$ |
| $a \mid 4$ | $\times$ | $\times$ | $\times$ | $A$ | $S,B$ |
| $b \mid 5$ | $\times$ | $\times$ | $\times$ | $\times$ | $S,B$ |

Najzad,

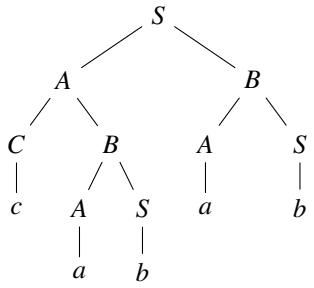
$$\begin{aligned} U[1,5] &= \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[1,1], Z \in U[2,5]\} \cup \\ &\quad \cup \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[1,2], Z \in U[3,5]\} \\ &\quad \cup \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[1,3], Z \in U[4,5]\} \\ &\quad \cup \{X \in \{S,A,B,C\} \mid X \to YZ, Y \in U[1,4], Z \in U[5,5]\} \\ &= \{X \in \{S,A,B,C\} \mid X \to CC \text{ ili } X \to AS \text{ ili } X \to AB\} = \{S,B\} \end{aligned}$$

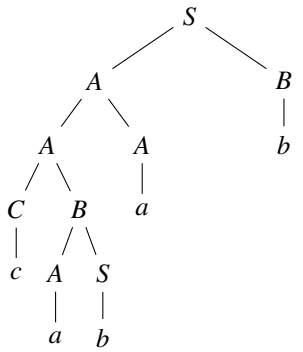
| $i \diagdown j$ | $\frac{c}{1}$ | $\frac{a}{2}$ | $\frac{b}{3}$ | $\frac{a}{4}$ | $\frac{b}{5}$ |
|---|---|---|---|---|---|
| $c \mid 1$ | $C$ | $\emptyset$ | $A$ | $A$ | $\mathbf{S}, B$ |
| $a \mid 2$ | $\times$ | $A$ | $S, B$ | $\emptyset$ | $C$ |
| $b \mid 3$ | $\times$ | $\times$ | $S, B$ | $\emptyset$ | $C$ |
| $a \mid 4$ | $\times$ | $\times$ | $\times$ | $A$ | $S, B$ |
| $b \mid 5$ | $\times$ | $\times$ | $\times$ | $\times$ | $S, B$ |

Dakle, $cabab \in L(\mathbb{G})$.

Prethodni algoritam se jednostavno može dopuniti, tako da se iz dobijene tabele može pročitati i odgovarajuće izvodjenje. Da bi se rekonstruisalo izvodjenje, potrebno je pored svake promenljive, upisane van glavne dijagonale, upisati njeno poreklo.

| $i \diagdown j$ | $\frac{c}{1}$ | $\frac{a}{2}$ | $\frac{b}{3}$ | $\frac{a}{4}$ | $\frac{b}{5}$ |
|---|---|---|---|---|---|
| $c \mid 1$ | $C$ | $\emptyset$ | $A : (C, B, 1)$ | $A : (A, A, 3)$ | $\mathbf{S} : (A, B, 3), B : (A, S, 3)$<br>$\mathbf{S} : (A, B, 4), B : (A, S, 4)$ |
| $a \mid 2$ | $\times$ | $A$ | $S : (A, B, 2)$<br>$B : (A, S, 2)$ | $\emptyset$ | $C : (B, S, 3)$ |
| $b \mid 3$ | $\times$ | $\times$ | $S, B$ | $\emptyset$ | $C : (B, S, 3)$ |
| $a \mid 4$ | $\times$ | $\times$ | $\times$ | $A$ | $S : (A, B, 4)$<br>$B : (A, S, 4)$ |
| $b \mid 5$ | $\times$ | $\times$ | $\times$ | $\times$ | $S, B$ |




Ako $S \in U[1, n]$, onda $w \in L(\mathbb{G})$, inače $w \notin L(\mathbb{G})$.

## 3.4 Svojstva kontekstno-slobodnih jezika

**Teorema 3.3.** *Skup kontekstno-slobodnih jezika nad istim alfabetom zatvoren je za uniju, nadovezivanje i Klinijevu zvezdicu.*

DOKAZ. Neka su $L_1$ i $L_2$ kontekstno-slobodni jezici nad istim alfabetom $\Sigma$ i $\mathbb{G}_i = (\Gamma_i, \Sigma, S_i, \mathscr{P}_i)$, $i = 1, 2$, kontekstno-slobodne gramatike koje ih generišu. Bez gubljenja opštosti možemo pretpostaviti da je $\Gamma_1 \cap \Gamma_2 = \emptyset$. Konstruišimo konteksno-slobodne gramatike koje redom generišu $L_1 \cup L_2$, $L_1 L_2$ i $L_1^*$. Neka je $S$ nova promenljiva, $S \notin \Gamma_1 \cup \Gamma_2$.

$\mathbb{G}_{\text{unija}} = (\Gamma_1 \cup \Gamma_2 \cup \{S\}, \Sigma, S, \mathscr{P}_1 \cup \mathscr{P}_2 \cup \{S \to S_1, S \to S_2\})$

$\mathbb{G}_{\text{nadovezivanje}} = (\Gamma_1 \cup \Gamma_2 \cup \{S\}, \Sigma, S, \mathscr{P}_1 \cup \mathscr{P}_2 \cup \{S \to S_1 S_2\})$

$\mathbb{G}_{\text{zvezdica}} = (\Gamma_1 \cup \{S\}, \Sigma, S, \mathscr{P}_1 \cup \{S \to SS_1, S \to S_1, S \to \varepsilon\})$ □

Posebno važna je varijanta leme napumpavanja za konteksno-slobodne jezike. Najpre ćemo dokazati jedno opštije tvrdjenje.

**Teorema 3.4.** [***Lema naduvavanja za konteksno-slobodne jezike***] *Za svaki konteksno-slobodan jezik $L$, postoji prirodan broj $n$ takav da se svaka reč $w$, čija je dužina bar $n$ ($|w| \geqslant n$), može rastaviti na pet delova $w = xuyvz$ takvih da važi:*

- $|uv| \geqslant 1$;
- $|uyv| \leqslant n$;
- $xu^k yv^k z \in L$, *za svako* $k \geqslant 0$.

DOKAZ. Neka je $L$ proizvoljan konteksno-slobodan jezik i $\mathbb{G} = (\Gamma, \Sigma, S, \mathscr{P})$ gramatika u normalnoj formi Čomskog koja ga generiše. Neka je $m = |\Gamma|$. Dokazaćemo da je $n = 2^{m+1}$ traženi prirodan broj. Neka je $w$ reč iz $L$ takva da je $|w| \geqslant 2^{m+1}$.

Posmatrajmo drvo izvodjenja reči $w$. Pošto je $\mathbb{G}$ u normalnoj formi Čomskog, svaki unutrašnji čvor drveta vodi ka nekoj podreči od $w$ koja je neprazna, tj. ovu reč proizvodi odgovarajuće poddrvo čiji je koren promenljiva koja odgovara uočenom unutrašnjem čvoru.

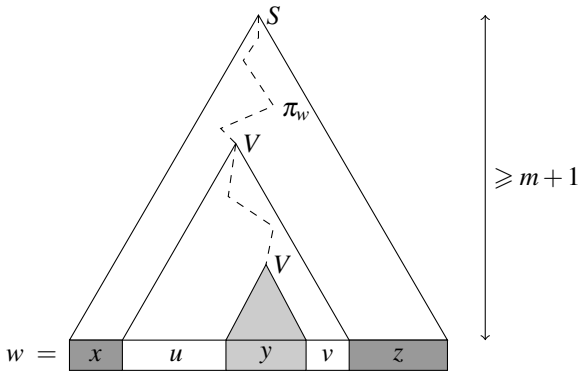
U drvetu izvodjenja reči $w$ izaberimo put $\pi_w$ na sledeći način:

- koren drveta izvodjenja, tj. početna promenljiva $S$, pripada putu $\pi_w$,
- ako je promenljiva $X$ na putu $\pi_w$ i ima dva naslednika, koji su promenljive, onda se na putu $\pi_w$ nalazi onaj naslednik koji vodi ka većemo broju terminala (simbola reči $w$); ukoliko naslednici vode ka istom broju terminala, onda se proizvoljno bira jedan od njih koji će biti na putu $\pi_w$;
- ako je promenljiva $X$ na putu $\pi_w$ i ima jednog naslednika, koji je terminal, onda se taj terminal nalazi na putu $\pi_w$.

Primetimo da je $\pi_w$ najduži put u drvetu izvodjenja reči $w$. Takodje, svaki unutrašnji čvor (neračunajući $S$) na putu $\pi_w$ vodi najmanje ka polovini terminala u odnosu na svog neposrednog prethodnika. Zato na putu $\pi_w$ mora postojati bar $m+1$ unutrašnjih čvorova. Ako bi $\pi_w$ imao $\leqslant m$ unutrašnjih čvorova, onda bi:

- poslednja promenljiva (prva odozdo) vodila ka jednom ($2^0$) terminalu;
- pretposlednja promenljiva (druga odozdo) vodila ka dva ($2^1$) terminala;
- treća promenljiva odozdo vodila ka najviše četiri ($2^2$) terminala;
- $\vdots$
- poslednja promenljiva odozdo, tj. $S$, vodila ka najviše $2^m$ terminala, što je nemoguće jer je $|w| \geqslant 2^{m+1}$.

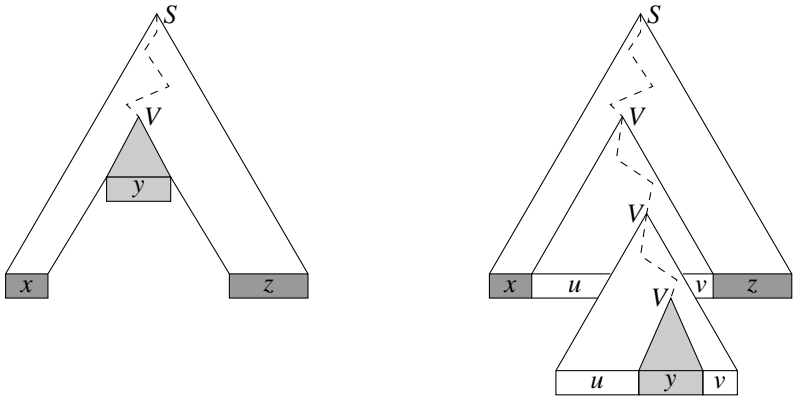
Izaberimo poslednjih $m+1$ unutrašnjih čvorova na putu $\pi_w$. Tada su bar dva čvora medju njima označena istom promenljivom, na primer $V$. Neka je $y$ podreč od $w$ koja potiče od nižeg pojavljivanja promenljive $V$, a $u$ i $v$ podreči od $w$ takve da $uyv$ potiče od višeg pojavljivanja promenljive $V$. Drvo izvodjenja reči $w$ se može predstaviti sledećom slikom.




Pošto je više pojavljivanje promenljive $V$ medju poslednjih $m+1$ čvorova, zaključujemo da reč $uyv$ ne može biti duža od $2^{m+1}$. Takodje, očigledno je da reči $u$ i $v$ ne mogu biti obe prazne. Reči $x$ i $y$ su izabrane tako da je $w = xuyvz$, iz $V \to^*_{\mathbb{G}} uyv$ i $V \to^*_{\mathbb{G}} y$ sledi:

$$S \to^*_{\mathbb{G}} xVz \to^*_{\mathbb{G}} xu^kVv^kz \to^*_{\mathbb{G}} xu^kyv^kz.$$

U skladu sa prethodnim, stabla izvodjenja reči $xyz, xu^2yv^2z \in L$ su prikazana narednom slikom.




□

PRIMER 3.7. Dokažimo da jezik $L = \{a^i b^i c^i \mid i \geqslant 1\}$ nije kontekstno-slobodan.

Pretpostavimo suprotno da $L$ jeste kontekstno-slobodan. Neka je $n$ prirodan broj čije postojanje tvrdi lema napumpavanja. Posmatrajmo reč $a^n b^n c^n \in L$. Tada se ova reč može rastaviti na pet delova $a^n b^n c^n = xuyvz$ takvih da je $|uv| \geqslant 1$, $|uyv| \leqslant n$ i $xu^k yv^k z \in L$, za svako $k \geqslant 0$.

Postoje dve mogućnosti.

1) $u$ i $v$ su oblika $a^+b^+$ ili $b^+c^+$: tada očigledno $xu^2yv^2z \notin a^*b^*c^*$.

2) $u, v \in a^* \cup b^* \cup c^*$: tada će se brisanjem $u$ i $v$ narušiti jednakost broja pojavljivanja slova $a$, $b$ i $c$, odn. biće ili $|xyz|_a \neq |xyz|_b$ ili $|xyz|_b \neq |xyz|_c$.

Izvedene kontradikcije obaraju pretpostavku da je $L$ kontekstno-slobodan.

**Teorema 3.5.** *Skup kontekstno-slobodnih jezika nad istim alfabetom nije zatvoren za presek i komplementiranje.*

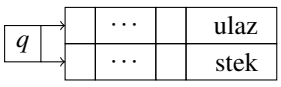
DOKAZ. Jezici $L_1 = \{a^i b^i c^j \mid i, j \geqslant 1\}$ i $L_2 = \{a^i b^j c^j \mid i, j \geqslant 1\}$ jesu kontekstno-slobodni. Jezik $L_1$ je generisan gramatikom čija su pravila $S \to AC$, $A \to aAb \mid \varepsilon$, $C \to cC \mid \varepsilon$. Analogno se mogu odrediti pravila koja generišu $L_2$. Medjutim, presek ova dva jezika

$$L_1 \cap L_2 = \{a^i b^i c^i \mid i \geqslant 1\}$$

nije kontekstno-slobodan kao što je dokazano u prethodnom primeru.

Ako bu skup kontekstno-slobodnih jezika bio zatvoren za komplementiranje, zbog zatvorenosti za uniju, morao bi biti zatvoren i za presek što nije tačno. □

## 3.5 Potisni automati (automati sa stekom)



**Definicija 3.7.** *Potisni automat je sedmorka* $\mathbb{A} = (Q, q_0, F, \Sigma, \Gamma, \$, \delta)$, *gde je*

- $Q$ *konačan skup stanja,*
- $q_0 \in Q$ *početno stanje,*
- $F \subseteq Q$ *skup završnih stanja,*
- $\Sigma$ *ulazni alfabet,*
- $\Gamma$ *alfabet steka,*
- $\$ \in \Gamma \setminus \Sigma$ *početni simbol steka,*
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \to \mathscr{P}(Q \times (\Gamma \cup \{\varepsilon\}))$.

*Početnu konfiguraciju* potisnog automata za ulaz $w = u_1 \dots u_k \in \Sigma^*$ označavaćemo sa

$$q_0 \left[ \begin{array}{c} u_1 \dots u_k \\ \$ \end{array} \right]$$

ukazujući na činjenicu da se automat nalazi u početnom stanju $q_0$, da je reč $w$ upisana na ulaznoj traci i da je na steku upisan samo znak \$. U svakom narednom trenutku, *konfiguraciju* čini trenutno stanje $q$, neki sufiks $u_i \dots u_k$ od $w$, tj. nepročitani deo ulaza $w$, upisan na ulaznoj traci ($u_j \in \Sigma$) i reč $s_1 \dots s_\ell$ iz $\Gamma^*$ upisana na steku ($s_j \in \Gamma$):

$$(*) \qquad q \left[ \begin{array}{c} u_i \dots u_k \\ s_1 \dots s_\ell \end{array} \right]$$

Računski korak je promena konfiguracije u skladu sa definicijom funkcije $\delta$. Tekuća konfiguracija $(*)$ se može promeniti izborom nekog elementa iz skupova $\delta(q, u_i, s_1)$, $\delta(q, \varepsilon, s_1)$, $\delta(q, u_i, \varepsilon)$ ili $\delta(q, \varepsilon, \varepsilon)$.

$$q \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s_1 s_2 \dots s_\ell \end{array} \right] \overset{(q', s') \in \delta(q, u_i, s_1)}{\longrightarrow} q' \left[ \begin{array}{c} u_{i+1} \dots u_k \\ s' s_2 \dots s_\ell \end{array} \right] \qquad q \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s_1 s_2 \dots s_\ell \end{array} \right] \overset{(q', \varepsilon) \in \delta(q, u_i, s_1)}{\longrightarrow} q' \left[ \begin{array}{c} u_{i+1} \dots u_k \\ s_2 \dots s_\ell \end{array} \right]$$

$$q \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s_1 s_2 \dots s_\ell \end{array} \right] \overset{(q', s') \in \delta(q, \varepsilon, s_1)}{\longrightarrow} q' \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s' s_2 \dots s_\ell \end{array} \right] \qquad q \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s_1 s_2 \dots s_\ell \end{array} \right] \overset{(q', \varepsilon) \in \delta(q, \varepsilon, s_1)}{\longrightarrow} q' \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s_2 \dots s_\ell \end{array} \right]$$

$$q \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s_1 s_2 \dots s_\ell \end{array} \right] \overset{(q', s') \in \delta(q, u_i, \varepsilon)}{\longrightarrow} q' \left[ \begin{array}{c} u_{i+1} \dots u_k \\ s' s_1 s_2 \dots s_\ell \end{array} \right] \qquad q \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s_1 s_2 \dots s_\ell \end{array} \right] \overset{(q', \varepsilon) \in \delta(q, u_i, \varepsilon)}{\longrightarrow} q' \left[ \begin{array}{c} u_{i+1} \dots u_k \\ s_1 s_2 \dots s_\ell \end{array} \right]$$

$$q \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s_1 s_2 \dots s_\ell \end{array} \right] \overset{(q', s') \in \delta(q, \varepsilon, \varepsilon)}{\longrightarrow} q' \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s' s_1 s_2 \dots s_\ell \end{array} \right] \qquad q \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s_1 s_2 \dots s_\ell \end{array} \right] \overset{(q', \varepsilon) \in \delta(q, \varepsilon, \varepsilon)}{\longrightarrow} q' \left[ \begin{array}{c} u_i u_{i+1} \dots u_k \\ s_1 s_2 \dots s_\ell \end{array} \right]$$

**Definicija 3.8.** *Potisni automat* $\mathbb{A}$ *prihvata reč* $w \in \Sigma^*$ *(završnim stanjima i praznim stekom) ukoliko postoji niz računskih koraka koji početnu konfiguraciju*

$$q_0 \left[ \begin{array}{c} w \\ \$ \end{array} \right]$$

*transformiše u konfiguraciju*

$$q \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix} \text{ za neko } q \in F.$$

*Jezik koji automat $\mathbb{A}$ prihvata (svojim završnim stanjima i praznim stekom) je:*

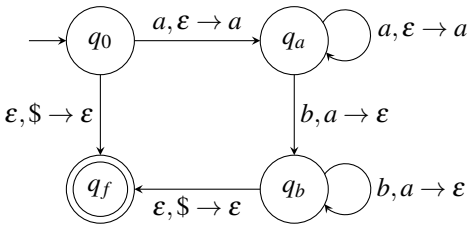
$$L(\mathbb{A}) = \left\{ w \in \Sigma^* \mid q_0 \begin{bmatrix} w \\ \$ \end{bmatrix} \vdash_{\mathbb{A}}^* q \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}, q \in F \right\}.$$

PRIMER 3.8. Konstruišimo potisni automat koji prihvata jezik

$$L = \{a^n b^n \mid n \geqslant 0\}.$$

Traženi potisni automat je $\mathbb{A} = (\{q_0, q_a, q_b, q_f\}, q_0, \{q_f\}, \{a,b\}, \{a,\$\}, \$, \delta)$, pri čemu je $\delta$ definisano sa:

$$\begin{aligned}
&\delta(q_0, \varepsilon, \$) = \{(q_f, \varepsilon)\} \\
&\delta(q_0, a, \varepsilon) = \{(q_a, a)\} \\
&\delta(q_a, a, \varepsilon) = \{(q_a, a)\} \\
&\delta(q_a, b, a) = \{(q_b, \varepsilon)\} \\
&\delta(q_b, b, a) = \{(q_b, \varepsilon)\} \\
&\delta(q_b, \varepsilon, \$) = \{(q_f, \varepsilon)\}
\end{aligned}$$




$$q_0 \begin{bmatrix} aabb \\ \$ \end{bmatrix} \vdash q_a \begin{bmatrix} abb \\ a\$ \end{bmatrix} \vdash q_a \begin{bmatrix} bb \\ aa\$ \end{bmatrix} \vdash q_b \begin{bmatrix} b \\ a\$ \end{bmatrix} \vdash q_b \begin{bmatrix} \varepsilon \\ \$ \end{bmatrix} \vdash q_f \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}$$

Dakle, $aabb \in L(\mathbb{A})$.

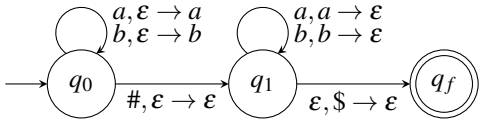
$q_0 \begin{bmatrix} \varepsilon \\ \$ \end{bmatrix} \vdash q_f \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}$, što znači da $\varepsilon \in L(\mathbb{A})$.

$q_0 \begin{bmatrix} aba \\ \$ \end{bmatrix} \vdash q_a \begin{bmatrix} ba \\ a\$ \end{bmatrix} \vdash q_b \begin{bmatrix} a \\ \$ \end{bmatrix}$, odnosno $aba \notin L(\mathbb{A})$.

PRIMER 3.9. $L = \{w\#w^R \mid w \in \{a,b\}^+\}$

$\mathbb{M} = (\{q_0, q_1, q_f\}, q_0, \{q_f\}, \{a,b,\#\}, \{a,b,\$\}, \$, \delta)$, pri čemu je $\delta$ definisano sa:

$$\begin{aligned}
&\delta(q_0, a, \varepsilon) = \{(q_0, a)\} \\
&\delta(q_0, b, \varepsilon) = \{(q_0, b)\} \\
&\delta(q_0, \#, \varepsilon) = \{(q_1, \varepsilon)\} \\
&\delta(q_1, a, a) = \{(q_1, \varepsilon)\} \\
&\delta(q_1, b, b) = \{(q_1, \varepsilon)\} \\
&\delta(q_1, \varepsilon, \$) = \{(q_f, \varepsilon)\}
\end{aligned}$$




$$\begin{aligned}
q_0 \begin{bmatrix} abb\#bba \\ \$ \end{bmatrix} &\vdash q_0 \begin{bmatrix} bb\#bba \\ a\$ \end{bmatrix} \vdash q_0 \begin{bmatrix} b\#bba \\ ba\$ \end{bmatrix} \vdash q_0 \begin{bmatrix} \#bba \\ bba\$ \end{bmatrix} \\
&\vdash q_1 \begin{bmatrix} bba \\ bba\$ \end{bmatrix} \vdash q_1 \begin{bmatrix} ba \\ ba\$ \end{bmatrix} \vdash q_1 \begin{bmatrix} a \\ a\$ \end{bmatrix} \vdash q_1 \begin{bmatrix} \varepsilon \\ \$ \end{bmatrix} \vdash q_f \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}
\end{aligned}$$

PRIMER 3.10. $L = \{ww^R \mid w \in \{a,b\}^+\}$

$\mathbb{M} = (\{q_0,q_1,q_f\},q_0,\{q_f\},\{a,b\},\{a,b,\$\},\$,\delta)$, pri čemu je $\delta$ definisano sa:

$$\begin{array}{l}\delta(q_0,a,\varepsilon) = \{(q_0,a)\}\\ \delta(q_0,b,\varepsilon) = \{(q_0,b)\}\\ \delta(q_0,\varepsilon,\varepsilon) = \{(q_1,\varepsilon)\}\\ \delta(q_1,a,a) = \{(q_1,\varepsilon)\}\\ \delta(q_1,b,b) = \{(q_1,\varepsilon)\}\\ \delta(q_1,\varepsilon,\$) = \{(q_f,\varepsilon)\}\end{array}$$

$$\begin{array}{rcl} q_0\begin{bmatrix} abbbba \\ \$ \end{bmatrix} & \vdash & q_0\begin{bmatrix} bbbba \\ a\$ \end{bmatrix} \vdash q_0\begin{bmatrix} bbba \\ ba\$ \end{bmatrix} \vdash q_0\begin{bmatrix} bba \\ bba\$ \end{bmatrix} \\ & \vdash & q_1\begin{bmatrix} ba \\ ba\$ \end{bmatrix} \vdash q_1\begin{bmatrix} a \\ a\$ \end{bmatrix} \vdash q_1\begin{bmatrix} \varepsilon \\ \$ \end{bmatrix} \vdash q_f\begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}\end{array}$$

Postoji još načina da se neki jezik pridruži potisnom automatu.

**Definicija 3.9.** *Jezik koji automat $\mathbb{A}$ prihvata svojim* ***završnim stanjima*** *je:*

$$L_f(\mathbb{A}) = \left\{ w \in \Sigma^* \mid q_0\begin{bmatrix} w \\ \$ \end{bmatrix} \vdash^*_{\mathbb{A}} q\begin{bmatrix} \varepsilon \\ \sigma \end{bmatrix}, q \in F, \sigma \in \Gamma^* \right\}$$

*Jezik koji prihvata automat $\mathbb{A}$* ***praznim stekom*** *je:*

$$L_e(\mathbb{A}) = \left\{ w \in \Sigma^* \mid q_0\begin{bmatrix} w \\ \$ \end{bmatrix} \vdash^*_{\mathbb{A}} q\begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}, q \in Q \right\}.$$

Ukoliko za neki potisni automat $\mathbb{A} = (Q,q_0,F,\Sigma,\Gamma,\$,\delta)$, posmatramo jezik $L_{\mathrm{e}}(\mathbb{A})$, onda uzimamo da je $F = \emptyset$, jer završna stanja nisu ni od kakvog značaja.

Označimo sa $\mathcal{L}_{\mathrm{fe}}(\Sigma)$, $\mathcal{L}_{\mathrm{e}}(\Sigma)$ i $\mathcal{L}_{\mathrm{f}}(\Sigma)$ skupove jezika nad $\Sigma$ za koje postoji potisni automat kojih ih redom prihvata

- praznim stekom i završnim stanjima,
- praznim stekom,
- završnim stanjima.

$$\begin{array}{lcll} \mathcal{L}_{\mathrm{fe}}(\Sigma) & \searrow & & \nearrow \quad L = L_{\mathrm{fe}}(\mathbb{A}). \\ \mathcal{L}_{\mathrm{e}}(\Sigma) & \to & \text{skup jezika } L \subseteq \Sigma^* \text{ za koje postoji p.a. } \mathbb{A} \text{ t.d.} & \to \quad L = L_{\mathrm{e}}(\mathbb{A}). \\ \mathcal{L}_{\mathrm{f}}(\Sigma) & \nearrow & & \searrow \quad L = L_{\mathrm{f}}(\mathbb{A}). \end{array}$$

**Teorema 3.6.** $\mathcal{L}_{\mathrm{fe}}(\Sigma) = \mathcal{L}_{\mathrm{e}}(\Sigma) = \mathcal{L}_{\mathrm{f}}(\Sigma)$

DOKAZ. Inkluzije $\mathcal{L}_{\mathrm{fe}}(\Sigma) \subseteq \mathcal{L}_{\mathrm{e}}(\Sigma)$ i $\mathcal{L}_{\mathrm{fe}}(\Sigma) \subseteq \mathcal{L}_{\mathrm{f}}(\Sigma)$ su očigledne. Dokazaćemo da važe i obratne inkluzije.

Dokažimo $\mathcal{L}_{\mathrm{e}}(\Sigma) \subseteq \mathcal{L}_{\mathrm{fe}}(\Sigma)$.

Neka $L \in \mathcal{L}_{\mathrm{e}}(\Sigma)$. Tada postoji potisni automat $\mathbb{A} = (Q, q_0, \emptyset, \Sigma, \Gamma, \$, \delta)$ takav da je $L = L_{\mathrm{e}}(\mathbb{A})$. Nije teško konstruisati potisni automat $\mathbb{A}'$ koji:

- simulira $\mathbb{A}$ i
- kada $\mathbb{A}$ isprazni svoj stek, $\mathbb{A}'$ ulazi u završno stanje.

$\mathbb{A}' = (Q \cup \{q_0', q_{\mathrm{f}}'\}, q_0', \{q_{\mathrm{f}}'\}, \Sigma, \Gamma \cup \{€\}, €, \delta')$, pri čemu je $\delta'$ odredjeno uslovima:

- $\delta'(q_0', \varepsilon, \varepsilon) = \{(q_0, \$)\}$,
- $\delta'(q, u, s) = \delta(q, u, s)$, $q \in Q$, $u \in \Sigma \cup \{\varepsilon\}$, $s \in \Gamma \cup \{\varepsilon\}$,
- $\delta'(q, \varepsilon, €) = \{(q_{\mathrm{f}}', \varepsilon)\}$, $q \in Q$.

$$q_0' \begin{bmatrix} w \\ € \end{bmatrix} \vdash q_0 \begin{bmatrix} w \\ \$€ \end{bmatrix} \vdash_{\mathbb{A}} q \begin{bmatrix} \varepsilon \\ € \end{bmatrix} \vdash q_{\mathrm{f}}' \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}$$

Dokažimo sada $\mathcal{L}_{\mathrm{f}}(\Sigma) \subseteq \mathcal{L}_{\mathrm{fe}}(\Sigma)$.

Neka $L \in \mathcal{L}_{\mathrm{f}}(\Sigma)$. Tada postoji potisni automat $\mathbb{A} = (Q, q_0, F, \Sigma, \Gamma, \$, \delta)$ takav da je $L = L_{\mathrm{f}}(\mathbb{A})$. Nije teško konstruisati potisni automat $\mathbb{A}'$ koji:

- simulira $\mathbb{A}$ i
- kada $\mathbb{A}$ udje u završno stanje, $\mathbb{A}'$ prazni svoj stek.

$\mathbb{A}' = (Q \cup \{q_0', q_{\mathrm{e}}'\}, q_0', \{q_{\mathrm{e}}'\}, \Sigma, \Gamma \cup \{€\}, €, \delta')$, pri čemu je $\delta'$ odredjeno uslovima:

- $\delta'(q_0', \varepsilon, \varepsilon) = \{(q_0, \$)\}$,
- $\delta'(q, u, s) = \delta(q, u, s)$, $q \in Q$, $u \in \Sigma \cup \{\varepsilon\}$, $s \in \Gamma \cup \{\varepsilon\}$,
- $\delta'(q, \varepsilon, s) = \{(q_{\mathrm{e}}', \varepsilon)\}$, $q \in F$, $s \in \Gamma \cup \{\varepsilon\}$,
- $\delta'(q_{\mathrm{e}}', \varepsilon, s) = \{(q_{\mathrm{e}}', \varepsilon)\}$, $s \in \Gamma \cup \{€\}$.

□

## 3.6 Kontekstno-slobodni jezici i potisni automati

Bez gubljenja opštosti možemo pretpostaviti da je funkcija tranzicije potisnih automata oblika $\overline{\delta} : Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \to \mathscr{P}_{\text{fin}}(Q \times \Gamma^*)$, tj. da je moguće upisati čitavu reč iz $\Gamma^*$ umesto prvog simbola steka ili čitavu reč dopisati na postojeću reč steka. Svaki potisni automat sa ovakvom funkcijom tranzicije se može transformisati u običan potisni automat dodavanjem novih stanja i odgovarajućom transformacijom funkcije tranzicije $\overline{\delta}$: ako

$$(q', s_1 \dots s_\ell) \in \overline{\delta}(q, u, s)$$

onda dodajemo nova stanja $q_1$, $q_2$, …, $q_{\ell-1}$ i pri definiciji funkcije $\delta$ uzimamo da:

$$\begin{aligned}
&(q_1, s_\ell) \in \delta(q, u, s),\ (q_1, s_\ell) \text{ zamenjuje } (q', s_1 \dots s_\ell)\\
&\delta(q_1, \varepsilon, \varepsilon) = \{(q_2, s_{\ell-1})\},\\
&\delta(q_2, \varepsilon, \varepsilon) = \{(q_3, s_{\ell-2})\},\\
&\vdots\\
&\delta(q_{\ell-1}, \varepsilon, \varepsilon) = \{(q', s_1)\}.
\end{aligned}$$

Štaviše, može se pretpostaviti i da su funkcije tranzicije oblika: $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to \mathscr{P}_{\text{fin}}(Q \times \Gamma^*)$

**Teorema 3.7.** *Za svaku kontekstno slobodnu gramatiku $G = (V, \Sigma, S, \mathscr{P})$ postoji potisni automat $\mathbb{A}$ takav da je $L(G) = L(\mathbb{A})$.*

*Dokaz.* $\mathbb{A} = (\{q_{\text{start}}, q_{\text{petlja}}, q_{\text{stop}}\}, q_{\text{start}}, \{q_{\text{stop}}\}, \Sigma, V \cup \Sigma \cup \{\$\}, \$, \overline{\delta})$
$\overline{\delta}(q_{\text{start}}, \varepsilon, \$) = \{(q_{\text{petlja}}, S\$)\}$
$\overline{\delta}(q_{\text{petlja}}, \varepsilon, X) = \{(q_{\text{petlja}}, w) \mid X \to w \in \mathscr{P}\}$
$\overline{\delta}(q_{\text{petlja}}, a, a) = \{(q_{\text{petlja}}, \varepsilon)\},\ a \in \Sigma$
$\overline{\delta}(q_{\text{petlja}}, \varepsilon, \$) = \{(q_{\text{stop}}, \varepsilon)\}$

PRIMER 3.11. Konstruišimo potisni automat koji prihvata reči jezika generisanog gramatikom $G = (\{S, A, B\}, \{a, b\}, S, \mathscr{P})$ čija su pravila:

$$\mathscr{P}: \quad S \to a \mid aAB,\ A \to aA \mid a,\ B \to bB \mid b.$$

$S \to aAB \to aaB \to aabB \to aabb$

$\mathbb{A} = (\{q_0, q_\text{p}, q_\text{f}\}, q_0, \{q_\text{f}\}, \{a, b\}, \{S, A, B, a, b, \$\}, \$, \delta)$, gde je $\delta$ definisano na sledeći način:
$\delta(q_0, \varepsilon, \$) = \{(q_p, S\$)\}$
$\delta(q_p, \varepsilon, S) = \{(q_\text{p}, a), (q_\text{p}, aAB)\}$
$\delta(q_\text{p}, \varepsilon, A) = \{(q_\text{p}, a), (q_\text{p}, aA)\}$
$\delta(q_\text{p}, \varepsilon, B) = \{(q_\text{p}, b), (q_\text{p}, bB)\}$

$\delta(q_\mathrm{p}, a, a) = \{(q_\mathrm{p}, \varepsilon)\}$
$\delta(q_\mathrm{p}, b, b) = \{(q_\mathrm{p}, \varepsilon)\}$
$\delta(q_\mathrm{p}, \varepsilon, \$) = \{(q_\mathrm{f}, \varepsilon)\}$.

$$\begin{aligned} q_0 \begin{bmatrix} aabb \\ \$ \end{bmatrix} &\vdash q_\mathrm{p} \begin{bmatrix} aabb \\ S\$ \end{bmatrix} \vdash q_\mathrm{p} \begin{bmatrix} aabb \\ aAB\$ \end{bmatrix} \vdash q_\mathrm{p} \begin{bmatrix} abb \\ AB\$ \end{bmatrix} \vdash q_\mathrm{p} \begin{bmatrix} abb \\ aB\$ \end{bmatrix} \\ &\vdash q_\mathrm{p} \begin{bmatrix} bb \\ B\$ \end{bmatrix} \vdash q_\mathrm{p} \begin{bmatrix} bb \\ bB\$ \end{bmatrix} \vdash q_\mathrm{p} \begin{bmatrix} b \\ B\$ \end{bmatrix} \vdash q_\mathrm{p} \begin{bmatrix} b \\ b\$ \end{bmatrix} \vdash q_\mathrm{p} \begin{bmatrix} \varepsilon \\ \$ \end{bmatrix} \\ &\vdash q_\mathrm{s} \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix} \end{aligned}$$

**Teorema 3.8.** *Svaki jezik koji prihvata potisni automat je kontekstno slobodan.*

DOKAZ. Neka je $L \subseteq \Sigma^*$ jezik koji praznim stekom prihvata potisni automat

$$\mathbb{A} = (Q, q_0, \emptyset, \Sigma, \Gamma, \$, \delta).$$

Dakle, $L = L_\mathrm{e}(\mathbb{A})$. Konstruisaćemo kontekstno-slobodnu gramatiku $\mathbb{G} = (\Gamma', \Sigma, S, \mathscr{P})$ takvu da je $L(\mathbb{G}) = L = L_\mathrm{e}(\mathbb{A})$.

Neka je $\Gamma' = \{S\} \cup (Q \times \Gamma \times Q)$. Pored početnog simbola $S$, za svaka dva stanja $p$ i $q$ i simbol steka $s \in \Gamma$ (potisnog automata $\mathbb{A}$) uvodimo po jednu promenljivu $(q, s, p)$ koja će generisati tačno one reči $w \in \Sigma^*$ za koje važi

$$q \begin{bmatrix} w \\ s \end{bmatrix} \vdash^* p \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}.$$

Primetimo da će tada za svako $\sigma \in \Gamma^*$ važiti

$$q \begin{bmatrix} w \\ s\sigma \end{bmatrix} \vdash^* p \begin{bmatrix} \varepsilon \\ \sigma \end{bmatrix}.$$

Ovo postižemo uvodjenjem odgovarajućeg skupa pravila $\mathscr{P}$.

Za ulaz $w \in \Sigma^*$, početna konfiguracija je

$$q_0 \begin{bmatrix} w \\ \$ \end{bmatrix},$$

a ako $w \in L_\mathrm{e}(\mathbb{A})$, završna konfiguracija je

$$p \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}, \text{ za neko } p \in Q$$

uvodimo sledeća pravila: $S \to (q_0, \$, p)$, $p \in Q$.

Ako $(p,\varepsilon) \in \delta(q,u,s)$, onda za $u \in \Sigma$ važi

$$q \begin{bmatrix} u \\ s \end{bmatrix} \vdash p \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix},$$

pa shodno tome uvodimo pravilo $(q,s,p) \to u$.

Ako $(p, s_1 s_2 \cdots s_m) \in \delta(q,u,s)$, za neko $m \geqslant 1$, onda za $w = uw'$ važi

$$q \begin{bmatrix} uw' \\ s \end{bmatrix} \vdash p \begin{bmatrix} w' \\ s_1 s_2 \cdots s_m \end{bmatrix}.$$

U slučajevima kada nakon ovog računskog koraka idemo ka završnoj konfiguraciji

$$q_m \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}, \text{ za neko } q_m \in Q$$

onda simboli $s_1, \ldots, s_m$ moraju biti obrisani u nekim koracima, tj. postoji niz stanja $q_1, \ldots, q_m$ i reč $w'$ se može zapisati u obliku $w_1 \cdots w_m$ tako da važi:

$$p \begin{bmatrix} w_1 \\ s_1 \end{bmatrix} \vdash q_1 \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}, q_1 \begin{bmatrix} w_2 \\ s_2 \end{bmatrix} \vdash q_2 \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}, \cdots, q_{m-1} \begin{bmatrix} w_m \\ s_m \end{bmatrix} \vdash q_m \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix},$$

pa dodajemo pravila

$$(q,s,q_m) \to u(p,s_1,q_1)(q_1,s_2,q_2)\cdots(q_{m-1},s_m,q_m),\ q_1,\ldots,q_m \in Q.$$

Može se podazati da za sve $w \in \Sigma^*$ važi

$$(q_0,\$,p) \to_{\mathbb{G}} w \text{ akko } q_0 \begin{bmatrix} w \\ \$ \end{bmatrix} \vdash p \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix}.$$

Odavde sledi da $L(\mathbb{G}) = L_{\mathrm{e}}(\mathbb{A})$. □

PRIMER 3.12. Neka je dat potisni automat $\mathbb{A} = (\{q_0,q_1\}, q_0, \emptyset, \{a,b\}, \{a,b,X,\$\}, \$, \delta)$, gde je $\delta$ definisano sa

(1) $\delta(q_0,a,X) = \{(q_1,X)\}$ (2) $\delta(q_0,b,\$) = \{(q_0,X\$)\}$ (3) $\delta(q_0,b,X) = \{(q_0,XX)\}$
(4) $\delta(q_0,\varepsilon,\$) = \{(q_0,\varepsilon)\}$ (5) $\delta(q_1,a,\$) = \{(q_0,\$)\}$ (6) $\delta(q_1,b,X) = \{(q_1,\varepsilon)\}$

Jezik koji automat prihvata je $L(\mathbb{A}) = \{\varepsilon\} \cup \{b^n a b^n a \mid n \geqslant 1\}$.

$$\begin{aligned} q_0 \begin{bmatrix} bbabba \\ \$ \end{bmatrix} &\overset{(2)}{\vdash} q_0 \begin{bmatrix} babba \\ X\$ \end{bmatrix} \overset{(3)}{\vdash} q_0 \begin{bmatrix} abba \\ XX\$ \end{bmatrix} \overset{(1)}{\vdash} q_1 \begin{bmatrix} bba \\ XX\$ \end{bmatrix} \\ &\overset{(6)}{\vdash} q_1 \begin{bmatrix} ba \\ X\$ \end{bmatrix} \overset{(6)}{\vdash} q_1 \begin{bmatrix} a \\ \$ \end{bmatrix} \overset{(5)}{\vdash} q_0 \begin{bmatrix} \varepsilon \\ \$ \end{bmatrix} \overset{(4)}{\vdash} q_0 \begin{bmatrix} \varepsilon \\ \varepsilon \end{bmatrix} \end{aligned}$$

Definišimo kontekstno-slobodnu gramatiku $\mathbb{G}$ takvu da je $L(\mathbb{G}) = L_e(\mathbb{A})$. Pravila gramatike uvodimo u skladu sa razmatranjima iz prethodne teoreme.

Najpre uvodimo pravila:
(0.1) $S \to (q_0, \$, q_0)$,
(0.2) $S \to (q_0, \$, q_1)$, a zatim na osnovu definicije funkcije $\delta$ dodajemo odgovarajuća pravila.

(1) $(q_1, X) \in \delta(q_0, a, X)$, pa odgovarajuća pravila dobijamo upisivanjem svih mogućih stanja na tačkom označena mesta sledeće sheme:

$$(q_0, X, \cdot) \to a(q_1, X, \cdot).$$

Na ovaj način dobijamo sledeća pravila:
(1.1) $(q_0, X, q_0) \to a(q_1, X, q_0)$,
(1.2) $(q_0, X, q_1) \to a(q_1, X, q_1)$.

(2) $(q_1, X\$) \in \delta(q_0, b, \$)$, pa odgovarajuća pravila dobijamo upisivanjem stanja na označena mesta sledeće sheme:

$$(q_0, \$, \cdot) \to b(q_0, X, *)(*, \$, \cdot),$$

pri čemu mesta označena istim simbolom menjamo istim stanjima. Na ovaj način dobijamo sledeća pravila:
(2.1) $(q_0, \$, q_0) \to b(q_0, X, q_0)(q_0, \$, q_0)$,
(2.2) $(q_0, \$, q_0) \to b(q_0, X, q_1)(q_1, \$, q_0)$,
(2.3) $(q_0, \$, q_1) \to b(q_0, X, q_0)(q_0, \$, q_1)$,
(2.4) $(q_0, \$, q_1) \to b(q_0, X, q_1)(q_1, \$, q_1)$.

(3) $(q_0, XX) \in \delta(q_0, b, X)$, pa kao u prethodnom slučaju dobijamo pravila na osnovu sheme $(q_0, X, \cdot) \to b(q_0, X, *)(*, X, \cdot)$, tj.
(3.1) $(q_0, X, q_0) \to b(q_0, X, q_0)(q_0, X, q_0)$,
(3.2) $(q_0, X, q_0) \to b(q_0, X, q_1)(q_1, X, q_0)$,
(3.3) $(q_0, X, q_1) \to b(q_0, X, q_0)(q_0, X, q_1)$,
(3.4) $(q_0, X, q_1) \to b(q_0, X, q_1)(q_1, X, q_1)$.

(4) $(q_0, \varepsilon) \in \delta(q_0, \varepsilon, \$)$, pa dobijamo pravilo:
(4.1) $(q_0, \$, q_0) \to \varepsilon$.

(5) $(q_1, \$) \in \delta(q_1, a, \$)$, pa iz $(q_0, \$, \cdot) \to a(q_0, \$, \cdot)$ dobijamo pravila:
(5.1) $(q_0, \$, q_0) \to a(q_0, \$, q_0)$,
(5.2) $(q_0, \$, q_1) \to a(q_0, \$, q_1)$.

(6) $(q_0, \varepsilon) \in \delta(q_1, b, X)$, pa dobijamo pravilo:
(6.1) $(q_1, X, q_1) \to b$.

$$
\begin{aligned}
S &\xrightarrow{(0.1)} (q_0,\$,q_0) \xrightarrow{(2.2)} b(q_0,X,q_1)(q_1,\$,q_0) \xrightarrow{(3.4)} bb(q_0,X,q_1)(q_1,X,q_1)(q_1,\$,q_0) \\
&\xrightarrow{(1.2)} bba(q_1,X,q_1)(q_1,X,q_1)(q_1,\$,q_0) \xrightarrow{(6.1)} bbab(q_1,X,q_1)(q_1,\$,q_0) \\
&\xrightarrow{(6.1)} bbabb(q_1,\$,q_0) \xrightarrow{(5.1)} bbabba(q_0,\$,q_0) \\
&\xrightarrow{(4.1)} bbabba
\end{aligned}
$$