AUTOMATI, FORMALNI JEZICI I JEZIČNI PROCESORI I

Deterministički konačni automat

$dka = (Q, \Sigma, \delta, q_0, F)$

Q - konačan skup stanja

Σ - konačan skup ulaznih znakova

$q_0 \in Q$ - početno stanje

$F \subseteq Q$ - skup prihvatljivih stanja

δ - funkcija prijelaza: $Q x \Sigma \to Q$

$\hat{\delta}: Q x \Sigma^* \to Q$

Σ^* = skup svih mogućih nizova ulaznih znakova uključujući i prazan niz

1.) $\hat{\delta}(q, \varepsilon) = q$

ε = prazan niz

2.) $\hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a)$

$w \in \Sigma^*, \quad a \in \Sigma$

Nedeterministički konačni automat

$nka = (Q, \Sigma, \delta, q_0, F)$

Q - konačan skup stanja

Σ - konačan skup ulaznih znakova

$q_0 \in Q$ - početno stanje

$F \subseteq Q$ - skup prihvatljivih stanja

δ - funkcija prijelaza: $Q x \Sigma \to 2^Q$

$\hat{\delta}: Q x \Sigma^* \to 2^Q$

1.) $\hat{\delta}(q, \varepsilon) = \{q\}$

2.) $\hat{\delta}(q, wa)$ = P = {p | za neko stanje r iz $\hat{\delta}(q, w)$, p je iz $\delta(r, a)$ }

$w \in \Sigma^*, \quad a \in \Sigma, \quad P \subseteq Q$

Konstrukcija DKA iz NKA

NKA: $M(Q, \Sigma, \delta, q_0, F)$

DKA: $M'(Q', \Sigma', \delta', q_0', F')$

1.) skup stanja DKA je $Q' = 2^Q$, tj. skup svih podskupova stanja NKA Q označenih kao $[p_0, p_1, \ldots, p_j] \in Q'$, gdje su $p_k \in Q$

2.) skup prihvatljivih stanja DKA, F' je skup svih stanja $[p_0, p_1, \ldots, p_j]$ gdje je barem jedan $p_k \in F$

3.) početno stanje DKA je $q_0' = [q_0]$

4.) funkcija prijelaza DKA je $\hat{\delta}'([p_0, p_1, \ldots, p_k], a) = [r_0, r_1, \ldots, r_j]$ ako i samo ako je $\delta(\{p_0, p_1, \ldots, p_k\}, a) = \{r_0, r_1, \ldots, r_j\}$, gdje je $a \in \Sigma$.

Nedeterministički konačni automat s ε prijelazima

$\varepsilon - \text{nka} = (Q, \Sigma, \delta, q_0, F)$

Q - konačan skup stanja

Σ - konačan skup ulaznih znakova

$q_0 \in Q$ - početno stanje

$F \subseteq Q$ - skup prihvatljivih stanja

δ - funkcija prijelaza: $Qx(\Sigma \cup \{\varepsilon\}) \to 2^Q$

$\hat{\delta}: Qx\Sigma^* \to 2^Q$

$\varepsilon - \text{CLOSURE} = \varepsilon - C$

1.) $\varepsilon - C(R) = \bigcup_{q \in R} \varepsilon - C(q)$

$R \subseteq Q$

2.) $\hat{\delta}(q, \varepsilon) = \varepsilon - C\{q_0\}$

3.) $\hat{\delta}(q, wa) = \varepsilon - C(P)$, gdje je P = {p | za neko stanje r iz $\hat{\delta}(q, w)$, p je iz $\delta(r, a)$ }

$w \in \Sigma^*, \quad a \in \Sigma$

4.) $\delta(R, a) = \bigcup_{q \in R} \delta(q, a)$

$R \subseteq Q, \quad a \in \Sigma$

5.) $\hat{\delta}(R, w) = \bigcup_{q \in R} \hat{\delta}(q, w)$

$R \subseteq Q, \quad q \in \Sigma^*$

Konstrukcija NKA iz ε-NKA

1.) Q = Q'

2.) $q_0' = q_0$

3.) $F' = F \cup q_0$, ako $\varepsilon - C(q_0)$ sadrži barem jedno stanje iz F; inače F'=F

4.) $\delta'(q, a) = \hat{\delta}(q, a), \quad a \in \Sigma, \quad q \in Q$

Mooreov automat

$\text{MoDka} = (Q, \Sigma, \Delta, \delta, \lambda, q_0)$

Q - konačan skup stanja

Σ - konačan skup ulaznih znakova

$q_0 \in Q$ - početno stanje

δ - funkcija prijelaza: $Qx\Sigma \to Q$

Δ - konačan skup izlaznih znakova

λ - funkcija izlaza: $Q \to \Delta$

- za prazan niz automat daje $\lambda(q_0)$

Mealyev automat

$\text{MeDka}(Q, \Sigma, \Delta, \delta, \lambda, q_0)$

Q - konačan skup stanja

Σ - konačan skup ulaznih znakova

$q_0 \in Q$ - početno stanje

δ - funkcija prijelaza: $Q x \Sigma \to Q$

Δ - konačan skup izlaznih znakova

λ - funkcija izlaza: $Q x \Sigma \to \Delta$

- za prazan niz automat ne daje nikakav izlaz

Konstrukcija Mealyevog iz Mooreovog automata

1.) $\lambda'(q,a) = \lambda(\delta(q,a))$

$\forall q \in Q \wedge \forall a \in \Sigma$

Konstrukcija Mooreovog iz Mealyevog automata

1.) $Q' = Q x \Delta$, stanje $[q,b] \in Q'$

$q \in Q, \quad b \in \Delta$

2.) $q_0' = [q_0, b_0]$, gdje je b_0 proizvoljan element iz skupa Δ

3.) $\delta'([q,b],a) = [\delta(q,a), \lambda(q,a)]$

$q \in Q, \quad b \in \Delta, \quad a \in \Sigma$

4.) $\lambda'([q,b]) = b$

$q \in Q, \quad b \in \Delta$

Rekurzivna pravila za regularne izraze:

1.) $\varnothing$ je regularni izraz i označava jezik $L(\varnothing) = \{\ \}$

2.) ε je regularni izraz i označava jezik $L(\varepsilon) = \{\varepsilon\}$

3.) za $\forall a \in \Sigma$, *a* je regularni izraz i označava jezik $L(a) = \{a\}$

4.) Neka su *r* i *s* regularni izrazi koji označavaju jezike $L(r)$ i $L(s)$. Tada je:

a) $(r)+(s)$ regularni izraz koji označava jezik $L((r)+(s)) = L(r) \cup L(s)$

b) $(r)(s)$ regularni izraz koji označava jezik $L((r)(s)) = L(r)L(s)$

c) $(r)^*$ regularni izraz koji označava jezik $L((r)^*) = L(r)^*$

Presjek regularnih jezika

Neka su zadana dva DKA: DKA $M_1 = (Q_1, \Sigma, \delta_1, F_1)$ i DKA $M_2 = (Q_2, \Sigma, \delta_2, F_2)$. Tada se može izgraditi DKA $M = (Q, \Sigma, \delta_0, F)$ koji prihvaća regularni jezik $L(M') = L(M_1) \cap L(M_2)$ na sljedeći način:

1.) $Q = Q_1 \, x \, Q_2$, stanje $[q,p] \in Q$, gdje je $q \in Q_1$ i $p \in Q_2$

2.) $q_0 = [q_1, q_2]$

3.) $F = F_1 \, x \, F_2$, stanje $[q,p] \in F$, gdje je $q \in F_1$ i $p \in F_2$

4.) $\delta([q,p],a) = [\delta_1(q,a), \delta_2(p,a)]$, za $\forall q \in Q_1$, $\forall p \in Q_2$ i $\forall a \in \Sigma$

Kontekstno neovisna gramatika je uređena četvorka G = (V,T,P,S)

V = konačan skup nezavisnih znakova

T = konačan skup završnih znakova $V \cap T = 0$

P = konačan skup pravila produkcija oblika $A \to \alpha$, gdje je A nezavršni znak, a α je niz znakova iz skupa $(V \cap T)^*$

S = početni znak

Definicija generativnog stabla

Stablo je generativno za gramatiku G ako su:

1.) čvorovi stabla označeni znakovima iz $V \cup T \cup \{\varepsilon\}$

2.) korijen stabla označen je s početnim nezavršnim znakom S

3.) ako je unutrašnji čvor označen s *A*, tada *A* mora biti nezavršni znak, $A \in V$

4.) Neka su čvorovi $n_1, n_2, \ldots, n_k$ čvorovi sinovi od čvora *n*. Neka je čvor *n* označen sa *A*, a čvorovi $n_1, n_2, \ldots, n_k$ s $X_1, X_2, \ldots, X_k$

Tada $A \to X_1, X_2, \ldots, X_k$ mora biti produkcija iz skupa *P*.

5.) ako je čvor *n* označen s *ε*, tada je čvor *n* list i on je jedini neposredni sljedbenik jednog od unutarnjeg čvora.

Konstrukcija gramatike za regularni jezik zadan DKA

1.) za skup završnih znakova gramatike, T uzima se skup ulaznih znakova DKA, Σ, tj. **T=Σ**

2.) za skup nezavršnih znakova gramatike, V uzima se skup stanja DKA, Q, tj. **V=Q**

3.) za početni nezavršni znak gramatike, S uzima se početno stanje, q_0, tj. **S=q_0**

4.) ako je $\delta(A, a) = B$ prijelaz DKA iz stanja *A* u stanje *B* za ulazni znak *a*, tada se uvodi produkcija $A \to aB$ gdje su *A* i *B* nezavršni znakovi gramatike, a *a* je završni znak

5.) ako je *A* prihvatljivo stanje DKA, tj. $A \in F$, tada se uvodi produkcija $A \to \varepsilon$ gdje je *A* nezavršni znak gramatike, a ε je prazan niz.

Konstrukcija NKA za regularni jezik zadan gramatikom

1.) Za skup ulaznih znakova NKA, Σ, uzima se skup završnih znakova gramatike, T, tj. $\Sigma = T$.

2.) Za skup stanja NKA, Q, uzima se skup nezavršnih znakova gramatike, V, tj. $Q = V$

3.) Za početno stanje NKA, q_0, uzima se početni nezavršni znak gramatike, S, tj. $q_0 = S$

4.) Ako je $A \to aB$ produkcija gramatike, tada se uvodi prijelaz iz stanja *A* u stanje *B* za ulazni znak *a*: $\delta(A, a) = \delta(A, a) \cup \{B\}$

5.) Ako gramatika ima produkciju $A \to \varepsilon$, gdje je *A* nezavršni znak gramatike, a *ε* prazan niz, tada je *A* prihvatljivo stanje, $A \in F$

Konstrukcija NKA iz desno-linearne gramatike

1.) Za sve produkcije oblika $A \to w$, gdje je *w* neprazan niz nezavršnih znakova, uvodi se jedan novi nezavršni znak, npr. *[ε]*, i dodaje se jedna nova produkcija: $[\varepsilon] \to \varepsilon$, te se sve produkcije oblika $A \to w$ zamjenjuju novim produkcijama oblika: $A \to w[\varepsilon]$

2.) Sve produkcije oblika $A \to a_1 \ldots a_n B$, $n > 1$ zamjenjuju se produkcijama oblika

$$A \to a_1[a_2 \ldots a_n B]$$
$$[a_2 \ldots a_n B] \to a_2[a_3 \ldots a_n B]$$
$$\vdots$$
$$[a_{n-1} a_n B] \to a_{n-1}[a_n B]$$
$$[a_n B] \to a_n B$$

gdje su $[a_i \ldots a_n B]$ novi nezavršni znakovi, za $1 < i \le n$

3.) Ako je nezavršni znak *B* jedini znak na desnoj strani produkcije $A \to B$ tada se prvo izbace sve produkcije koje imaju istu lijevu i desnu stranu ($B \to B$), te, ako ostanu produkcije koje imaju različite lijeve i desne strane ($A \to B$), one se zamjenjuju s $A \to y$ za sve kombinacije nezavršnih znakova *A* i desnih strana produkcija *y* za koje vrijedi: $A \to B$ i $B \to y$.

Konstrukcija NKA iz lijevo-linearne gramatike

1.) Neka je $G'(V, T, P', S)$ gramatika čiji skup produkcija *P'* čine produkcije iz skupa *P* gramatike *G* s tim da su desne strane produkcija napisane obrnutim redoslijedom:

$P' = \{A \to \alpha \mid A \to \alpha^R \text{ je iz skupa } P\}$. Dobivena gramatika *G'* je desno-linearna i ona generira nizove završnih znakova koji su obrnuto napisani nizovi znakova koje generira gramatika *G*, tj. $L(G') = L(G)^R$

2.) Sad se konstruira NKA M koji prihvaća jezik $L(M) = L(G') = L(G)^R$ na temelju desno-linearne gramatike *G'*

3.) Da se dobije jezik $L(G)$ potrebno je izgraditi NKA M' na temelju NKA M takav da prihvaća jezik $L(M') = L(M)^R = L(G')^R = L(G)$, tj. NKA M' prihvaća upravo jezik $L(M') = L(G)$ kojeg generira lijevo-linearna gramatika G.

Konstrukcija lijevo-linearne gramatike iz NKA

1.) Prvo se konstruira NKA M takav da M prihvaća jezik $L(M) = L^R$ kao i kod konstrukcije NKA iz lijevo-linearne gramatike

2.) Za izgrađeni NKA M izgradi se desno-linearna gramatika *G* tako da vrijedi $L(G) = L(M) = L^R$

3.) S desne strane produkcije se napišu obrnutim redoslijedom tako da se dobije lijevo-linearna gramatika *G'* za koju vrijedi $L(G') = L(G)^R = L(M)^R = L$

Izbacivanje mrtvih znakova

Neka je $G = (V, T, P, S)$ kontekstno-neovisna gramatika i neka je jezik neprazan, tj. $L(G) \neq \varnothing$. Tada se može izgraditi ekvivalentna gramatika $G' = (V', T', P', S)$ takva da nema mrtvih nezavršnih znakova, odnosno da za svaki $A \in V'$ postoji niz *w* završnih znakova tako da vrijedi $A \overset{*}{\Rightarrow} w$

Algoritam za pronalaženje živih znakova:

1.) Stavi u listu živih znakova sve nezavršne znakove koji se pojavljuju na lijevim stranama produkcija čija desna strana nema niti jedan nezavršni znak.

2.) Ukoliko postoji produkcija čiji su svi nezavršni znakovi na desnoj strani živi znakovi, tada se nezavršni znak s lijeve strane produkcije dodaje u listu živih znakova.

3.) Ukoliko nije više moguće proširiti listu živih znakova, tada su svi znakovi u listi živi, dok su svi ostali mrtvi.

Izbacivanje nedohvatljivih stanja

Neka je $G = (V, T, P, S)$ kontekstno-neovisna gramatika. Tada se može izgraditi ekvivalentna gramatika $G' = (V', T', P', S)$ takva da nema nedohvatljivih znakova, odnosno da za svaki $X \in V' \cup T'$ postoje $\alpha, \beta \in (V' \cup T')$ takvih da vrijedi $S \overset{*}{\Rightarrow} \alpha X \beta$

Algoritam za traženje dohvatljivih znakova:

1.) Stavi u listu dohvatljivih znakova početni nezavršni znak gramatike

2.) Ako postoji produkcija čija lijeva strana se nalazi u listi dohvatljivih znakova tada se nezavršni i završni znakovi na desnoj strani dodaju u listu dohvatljivih znakova.

3.) Ukoliko se lista dohvatljivih znakova ne može proširiti, tada su u listi svo dohvatljivi znakovi. Svi ostali znakovi su nedohvatljivi.

Izbacivanje beskorisnih znakova

Primjenom algoritma za izbacivanje mrtvih znakova, a zatim algoritma za izbacivanje nedohvatljivih znakova, mogu se iz gramatike izbaciti svi beskorisni znakovi. Obrnuto, nužno ne vrijedi.

Izbacivanje ε-produkcija

1.) Prvo se utvrdi koji nezavršni znakovi generiraju prazan niz, odnosno traže se svi nezavršni znakovi za koje vrijedi $A \overset{*}{\Rightarrow} \varepsilon$. Neka se ti nezavršni znakovi nazovu *prazni znakovi*. Prazni znakovi se mogu pronaći na slijedeći način:

Prvo se u listu praznih znakova stave svi nezavršni znakovi za koje postoje produkcije oblika $A \to \varepsilon$. U slijedećem koraku se lista praznih znakova nadopuni s nezavršnim znakom *B* ukoliko postoji

produkcija $B \to \alpha$ i svi znakovi u α su u skupu praznih znakova. Algoritam se nastavlja sve dok se lista praznih znakova može proširiti novim nezavršnim znakovima.
2.) Zamijene se sve produkcije koje na desnoj strani imaju barem jedan prazni nezavršni znak sa skupom produkcija. Ukoliko jedna produkcija ima *k* praznih znakova, tada se umjesto jedne produkcije uvodi skup od 2^k produkcija. Novo izgrađene produkcije imaju sve moguće kombinacije brisanja i ostavljanja praznih znakova.
3.) Iz skupa produkcija izbace se sve ε-produkcije nastale u prethodnom koraku.

Izbacivanje jediničnih produkcija
1.) U skup produkcija *P'* gramatike *G'* prvo se stave sve produkcije gramatike *G* koje nisu jedinične
2.) Neka se postupkom generiranja iz nezavršnog znaka *A* dobije nezavršni znak *B*, tj. $A \overset{*}{\Rightarrow} B$, gdje su *A* i *B* nezavršni znakovi gramatike *G*. Tada se skup produkcija *P'* proširi produkcijama $A \to \alpha$ i to za sve produkcije $B \to \alpha$ koje nisu jedinične. Dovoljno je promatrati samo nizove $A \overset{*}{\Rightarrow} B$ $A \overset{*}{\Rightarrow} B_1 \overset{*}{\Rightarrow} B_2 \overset{*}{\Rightarrow} \ldots \overset{*}{\Rightarrow} B_m \overset{*}{\Rightarrow} B$ u kojima se ne pojavljuje niti jedan nezavršni znak po dva puta.

Chomskyev normalni oblik produkcija
Sve produkcije oblika: $A \to BC$ ili $A \to a$.
1.) Prvo se u skup produkcija *P'* prepišu sve produkcije koje već imaju Chomskyev normalni oblik. Isto tako prepiše se skup znakova u skup *T'* i skup nezavršnih znakova u skup *V'*
2.) Neka je produkcija gramatike *G* oblika: $A \to X_1 X_2 \ldots X_m$, $A \in V$, $X_i \in T$ ili $X_i \in V$, $m \geq 2$, $1 \leq i \leq m$. Znak X_i može biti završni ili nezavršni. Neka je X_i završni znak *a*, $a \in T$. Sada se proširi skup nezavršnih znakova i uvede novi nezavršni znak C_a, $C_a \in V'$. Za novi nezavršni znak C_a uvede se u skup *P'* produkcija: $C_a \to a$ koja je u Chomskyevom normalnom obliku. Sada se svi završni znakovi *a* na desnoj strani produkcija zamijene nezavršnim znakom C_a. Proces uvođenja novih nezavršnih znakova i novih produkcija nastavlja se za sve završne znakove na desnoj strani produkcije $A \to X_1 X_2 \ldots X_m$. Nakon zamjene svih završnih znakova s novim nezavršnim znakovima, produkcija $A \to X_1 X_2 \ldots X_m$ imat će na desnoj strani samo nezavršne znakove gramatike *G'*. Proces se nastavlja za sve produkcije oblika $A \to X_1 X_2 \ldots X_m$
3.) Sad imamo sve produkcije oblika $A \to a$ ili $A \to B_1 B_2 \ldots B_m$, $m \geq 2$. Produkcije oblika $A \to B_1 B_2 \ldots B_m$, za $m \geq 3$, odnosno one koje imaju više od tri znaka na desnoj strani zamijene se novim produkcijama tako da se uvedu novi nezavršni znakovi D_1, D_2,…,D_{m-1}, a zatim se produkcija $A \to B_1 B_2 \ldots B_m$ zamijeni skupom produkcija: { $A \to B_1 D_1$, $D_1 \to B_2 B_2$, $D_2 \to B_3 B_3$, …, $D_{m-3} \to B_{m-2} B_{m-2}$, $D_{m-2} \to B_{m-1} B_m$ }

Potisni automat

Potisni automat (PA) je uređena sedmorka: $PA = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

Q - konačni skup stanja
Σ - konačni skup ulaznih znakova
Γ - konačan skup znakova stoga
$q_0 \in Q$ - početno stanje
$Z_0 \in \Gamma$ - početni znak stoga
$F \subseteq Q$ - skup prihvatljivih stanja
δ - funkcija prijelaza koja pridružuje trojki $Qx(\Sigma \cup \{\varepsilon\})x\Gamma$ konačni podskup skupa svih mogućih parova $Qx\Gamma^*$

Funkcija prijelaza PA

$\delta(q, a, Z) = \{(p_1, \gamma_1), (p_2, \gamma_2), \ldots, (p_m, \gamma_m)\}$ q - stanje PA ($q \in Q$)

a - ulazni znak ($a \in \Sigma$)
Z - znak stoga ($Z \in \Gamma$)
p - stanje u koje prelazi PA ($p \in Q$)
γ_i - nizovi znakova stoga ($\gamma \in \Gamma$)

Konfiguracija PA

1.) trenutno stanje
2.) neispitani dio ulaznih znakova
3.) znakovi na stogu

Prihvaćanje jezika zadanim PA

Kaže se da PA prihvaća jezik L(M) *prihvatljivim stanjem* ako je: $L(M) = \{ w \mid (q_0, w, Z_0) \vdash^* (p, \varepsilon, \gamma)$, za neko stanje $p \in F$ i $\gamma \in \Gamma \}$ PA prihvaća jezik N(M) *praznim stogom* ako je $N(M) = \{ w \mid (q_0, w, Z_0) \vdash^* (p, \varepsilon, \varepsilon)$ za neko stanje $p \in Q \}$ uz $F = \varnothing$

Deterministički PA

PA je deterministički PA (DPA) ako je:

1.) Za svako stanje $q \in Q$, za svaki znak stoga $Z \in \Gamma$ i svaki ulazni znak $a \in \Sigma$, mora vrijediti da je $\delta(q, a, Z)$ prazan skup ako je $\delta(q, \varepsilon, Z)$ neprazan skup.

2.) Za svako stanje $q \in Q$, za svaki znak stoga $Z \in \Gamma$ i svaki ulazni znak $a \in (\Sigma \cup \{\varepsilon\})$ skup $\delta(q, a, Z)$ ima najviše jedan element.

*NDA prihvaća širu klasu jezika od DPA

Konstrukcija PA koji prihvaća praznim stogom iz PA koji prihvaća prihvatljivim stanjem

Neka je $M_2 = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ PA koji prihvaća jezik L(M_2) prihvatljivim stanjem. PA M_1 koji prihvaća praznim stogom, a ekvivalentan je s PA M_2 konstruira se kao: $M_1 = (Q \cup \{q_0', q_e\}, \Sigma, \Gamma \cup \{X_0\}, \delta', q_0', X_0', \varnothing)$ gdje se funkcija prijelaza sefinira kao:

1.) $\delta'(q_0', \varepsilon, X_0) = \{(q_0, Z, X_0)\}$. Na početku rada PA M_1 bez učitanog znaka na ulazu prelazi iz svoje početne konfiguracije u početnu konfiguraciju PA M_2. PA M_1 stavlja znak X_0 na dno stoga da se izbjegne prihvaćanje niza u slučaju kad M_2 obriše cjelokupni stog a ne uđe u jedno od prihvatljivih stanja.

2.) Skup $\delta'(q, a, Z)$ sadrži sve elemente iz skupa $\delta(q, a, Z)$, i to sve *q* iz *Q*, za sve *a* iz $\Sigma \cup \{\varepsilon\}$, te za sve *Z* iz *Γ*. Ovim prijelazima omogućuje se da PA M_1 simulira rad PA M_2.

3.) Skup $\delta'(q, \varepsilon, Z)$ sadrži (q_e, ε) i to za svaki *q* iz skupa *F*, te za svaki ulazni znak *Z* iz $\Gamma \cup \{X_0\}$. Ukoliko M_2 uđe u jedno od prihvatljivih stanja, tada se skup prijelaza proširuje mogućnosti prijelaza u stanje q_e. Istodobno se s vrha stoga skine jedan znak

4.) Skup $\delta'(q_e, \varepsilon, Z)$ sadrži (q_e, ε), i to za svaki znak *Z* iz $\Gamma \cup \{X_0\}$. Ovim prijelazom omogućuje se brisanje čitavog stoga znak po znak, uključujući i znak X_0. Time je omogućeno da M_1 prihvati niz praznim stogom svaki put kad M_2 prihvati niz prihvatljivim stanjem.

Konstrukcija PA koji prihvaća prihvatljivim stanjem iz PA koji prihvaća praznim stogom

Neka je $M_1 = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \varnothing)$ PA koji prihvaća lezik N(M_1) praznim stogom. PA M_2 koji prihvaća prihvatljivim stanjem, a ekvivalentan je s PA M_1konstruira se kao: $M_2 = (Q \cup \{q_0', q_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta', X_0, \{q_f\})$gdje se funkcija prijelaza sefinira kao:

1.) $\delta'(q_0', \varepsilon, X_0) = \{(q_0, Z_0, X_0)\}$. Na početku rada PA M_2 prelazi iz svoje početne konfiguracije u početnu konfiguraciju PA M_1 i stavlja znak X_0 na dno stoga. Kada se X_0 pojavi na vrhu stoga, to je znak PA M_2 da je PA M_1 obrisao svoj čitav stog.

2.) Skup $\delta'(q, a, Z)$ sadrži sve elemente iz skupa $\delta(q, a, Z)$, i to za sve *q* iz *Q*, za sve *a* iz $\Sigma \cup \{\varepsilon\}$, te za sve *Z* iz *Γ*. Ovim prijezima omogućuje se da PA M_2 simulira rad PA M_1.

3.) Skup $\delta'(q,\varepsilon,X_0)$ sadrži (q_f,ε) i to za sve *q* iz *Q*. Ukoliko se na vrhu stoga pojavi X_0, tada je to znak da je PA M_1 ispraznio čitav svoj stog. Tada PA M_2 prijeđe u prihvatljivo stanje q_f kako bi PA M_2 prihvatio niz prihvatljivim stanjem svaki puta kad M_1 prihvati niz praznim stogom.

Konstrukcija PA koji prihvaća praznim stogom jezik zadan kontekstno-neovisnom gramatikom

Za svaki kontekstno-neovisni jezik *L* može se izgraditi gramatika $G=(V,T,P,S)$ takva da produkcije *P* imaju Greibachov oblik, odnosno sve produkcije su oblika $A\to a\beta$, gdje je $A\in V$, $a\in T$, $\beta\in V^*$ i β može biti prazan niz. Neka se konstruira PA $M=(\{q\},T,V,\delta,q,S,\varnothing)$ takav da:

a) PA M ima samo jedno stanje *q* koje je ujedno i početno stanje
b) Skup ulaznih znakova Σ jednak je skupu završnih znakova *T*, tj. $\Sigma=T$
c) Skup znakova stoga Γ jednak je skupu nezavršnih znakova gramatike, tj. $\Gamma=V$
d) Početni znak stoga je početni nezavršni znak gramatike S
e) Skup prihvatljivih stanja je prazan skup, $F=\varnothing$
f) PA M prihvaća praznim stogom.

Funkcija prijelaza zadaje se na slijedeći način: $\delta(q,a,A)$ sadrži (q,γ) ako i samo ako postoji produkcija $A\to a\gamma$.

Konstrukcija kontekstno-neovisne gramatike za jezik koji se prihvaća praznim stogom zadanog PA

Neka se za zadani PA $M=(Q,\Sigma,\Gamma,\delta,q_0,Z_0,F)$ konstruira kontekstno-neovisna gramatika $G=(V,T,P,S)$ gdje je skup nezavršnih znakova *V* skup stanja označenih kao $[q,A,p]$ gdje su *p* i *q* stanja iz skupa stanja *Q* PA M, a *A* je znak stoga iz skupa Γ. Tom skupu dodaje se i početni nezavršni znak *S*. Skup produkcija izgradi se na slijedeći način:

1.) Uvode se produkcije $S\to[q_0,Z_0,q]$ za sva stanja *q* iz *Q*, gdje je q_0 početno stanje PA, a Z_0 početni znak stoga.

2.) Ako skup $\delta(q,a,A)$ sadrži $(q_1,B_1,B_2,\ldots,B_m)$ tada se uvode nove produkcije $[q,A,q_{m+1}]\to a[q_1,B_1,q_2][q_2,B_2,q_3]---[q_m,B_m,q_{m+1}]$ za svako stanje $q_1, q_2, \ldots, q_{m+1}$ iz skupa *Q*, za svaki *a* iz skupa $\Sigma\cup\{\varepsilon\}$ i za svaki A, $B_1, \ldots, B_m$ iz skupa Γ. Ako je m=0, tada se uvodi nova produkcija $[q,A,q_1]\to a$

Unija dva kontekstno-neovisna jezika

Gramatika $G=(V_3,T_3,P_3,S_3)$ koja generira jezik $L(G_1)\cup L(G_2)$ konstruira se na slijedeći način:

1.) $V_3=V_1\cup V_2\cup\{S_3\}$, gdje je $S_3\notin V_1$ i $S_3\notin V_2$, $V_1\cap V_2=\varnothing$
2.) $T_3=T_1\cup T_2$
3.) skupu produkcija $P_3=P_1\cup P_2$ dodaju se produkcije: $S_3\to S_1\,|\,S_2$

Konkatenacija dva kontekstno-neovisna jezika

Gramatika $G_4=(V_4,T_4,P_4,S_4)$ koja generira jezik $L(G_4)=L(G_1)L(G_2)$ konstruira se na sljedeći način:

1.) $V_4=V_1\cup V_2\cup\{S_4\}$, gdje je $S_4\notin V_1$ i $S_4\notin V_2$, $V_1\cap V_2=\varnothing$
2.) $T_4=T_1\cup T_2$
3.) skupu produkcija $P_4=P_1\cup P_2$ doda se produkcija $S_4\to S_1S_2$

Kleenov operator

Neka je gramatika $G_1 = (V_1, T_1, P_1, S_1)$ generira jezik $L(G_1)$. Gramatika $G_5 = (V_5, T_5, P_5, S_5)$ koja generira jezik $L(G_5) = L(G_1)^*$ konstruira se na sljedeći način:

1.) $V_5 = V_1 \cup \{S_5\}$, gdje je $S_5 \notin V_1$

2.) $T_5 = T_1$

3.) skupu produkcija $P_5 = P_1$ dodaju se produkcije: $S_5 \to S_1 S_5 \mid \varepsilon$

Supstitucija

Pretpostavimo da gramatika $G = (V, T, P, S)$ generira kontekstno-neovisan jezik $L(G)$. Neka se svaki znak a_i jezika $L(G)$ zamijeni jednim od nizova iz kontekstno-neovisnog jezika $L(G_i)$, gdje je $1 \le i \le k$. Broj *k* je broj znakova u abecedi jezika *L*. Označimo gramatiku koja generira jezik $L(G_i)$ s $G_i = (V_i, T_i, P_i, S_i)$. Nastali jezik *L'* je kontekstno-neovisan jezik za koji se može konstruirati gramatika $G' = (V', T', P', S')$ na sljedeći način:

1.) $V' = V \cup V_1 \cup V_1 - - - \cup V_k$, $V \cap V_i = \varnothing$, $V_i \cap V_j = \varnothing$, za $1 \le i \le k, 1 \le j \le k$, $i \ne j$, gdje se skup nezavršnih znakova gradi tako da se u njega stave svi nezavršni znakovi gramatike *G* i nezavršni znakovi svih gramatika G_i

2.) $T' = T \cup T_1 - - - T_k$,

gdje skup završnih znakova čine samo završni znakovi gramatika G_i

3.) $S' = S$,

početni nezavršni znak *S'* je početni nezavršni znak *S* gramatike *G*

4.) Skupu produkcija $P' = P_1 \cup P_1 - - - P_k$ dodaju se produkcije gramatike *G* koje se prethodno preurede na slijedeći način:

u svim produkcijama $A \to \alpha$ gramatike *G* svaki završni znak a_i zamjenjuje se početnim nezavršnim znakom S_i gramatike G_i

Presjek kontekstno-neovisnog jezika i regularnog jezika

Presjek kontekstno-neovisnog jezika i regularnog jezika je kontekstno-neovisni jezik.

Pretpostavimo da je L_1 kontekstno-neovisnan jezik, pa prema tome postoji PA $M_1 = (Q_1, \Sigma, \Gamma, \delta_1, q_0, Z_1, F_1)$ koji prihvaća jezik L_1. Neka regularni jezik L_2 prihvaća DKA $M_2 = (Q_2, \Sigma, \delta_2, p_0, F_2)$. Može se izgraditi PA $M' = (Q', \Sigma, \Gamma, \delta', q'_0, Z_0, F')$ koji prihvaća jezik $L = L_1 \cap L_2$:

1.) $Q' = Q_2 x Q_1$

2.) $q'_0 = [p_0, q_0]$

3.) $F' = F_2 x F_1$

4.) skup $\delta'([p,q], a, X)$ sadrži $([p', q'], \gamma)$ ako i samo ako DKA M_2 ima funkciju prijelaza $\delta_2(p, a) = p'$ i ako i samo ako PA M_1 u skupu $\delta'(q, a, X)$ sadrži (q', γ). Ukoliko *a* je *ε*, tada je $p' = p$

Turingov stroj

Turingov stroj je uređena sedmorka: $TS = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$

Q - konačan skup stanja

Γ - konačan skup znakova trake

$B \in \Gamma$ - znak kojim se označava prazna ćelija

$\Sigma \subseteq (\Gamma - \{B\})$ - konačan skup ulaznih znakova

$q_0 \in Q$ - početno stanje

$F \subseteq Q$ - skup prihvatljivih stanja

δ - funkcija prijelaza $\delta : Q \times \Gamma \to Q \times \Gamma \{L, R\}$, gdje *L* i *R* označavaju pomak glave u lijevo i desno. Funkcija δ može biti nedefinirana za pojedine argumente

Turingov stroj s beskonačnom trakom

Turingov stroj s beskonačnom trakom označava se s $TS\ M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ kao i osnovni model. Za razliku od osnovnog modela TS-a, ulazna traka je beskonačna na lijevu stranu isto kao i na desnu stranu.

Konstrukcija osnovnog modela TS-a iz TS-a s dvostranom beskonačnom trakom

Neka je $TS\ M_2 = (Q_2, \Sigma_2, \Gamma_2, \delta_2, q_2, B, F_2)$ Turingov stroj s dvostranom beskonačnom trakom. Može se konstruirati osnovni model stroja $TS\ M_1 = (Q_1, \Sigma_1, \Gamma_1, \delta_1, q_1, B, F_1)$ sa dva traga koji prihvaća isti jezik na sljedeći način:

1.) Ulazni znakovi iz Σ_1 sadrže prazninu u donjem tragu i ulazni znak u gornjem tragu, $[a, B] \in \Sigma_1$, gdje je $a \in \Sigma_2$

2.) Praznina se označava kao $B_1 = [B, B]$

3.) Stanja Q_1 pohranjuju pored podatka u kojem stanju se nalazi TS M_2 i podatak da li TS M_1 radi na gornjem ili donjem tragu. Ukoliko TS M_1 koristi gornji trag, tada će stanje biti $[q, U]$, gdje je $q \in Q_2$. Ukoliko TS M_1 koristi gornji trag, tada će stanje biti $[q, D]$, gdje je $q \in Q_2$. U skupu stanja Q_1 je također i početno stanje q_1.

4.) Znakovi trake iz skupa Γ_1 su oblika $[X, Y]$, gdje su *X* i *Y* iz skupa Γ_2. Znak *Y* može biti **ç**.

5.) $F_1 = \{[q, U], [q, D] \mid q \in F_2\}$

6.) Funkcija δ_1 definira se kao:

a) Za svaki ulazni znak $a \in (\Sigma_2 \cup \{B\})$ uvodi se: $\delta_1(q_1, [a, B]) = ([q, U], [X, c], R)$ ako je $\delta_2(q_2, a) = (q, X, R)$

b) Za svaki ulazni znak $a \in (\Sigma_2 \cup \{B\})$ uvodi se: $\delta_1(q_1, [a, B]) = ([q, D], [X, c], R)$ ako je $\delta_2(q_2, a) = (q, X, L)$

c) Za svaki znak trake $[X, Y] \in \Gamma_1$ gdje je $Y \neq c$ i *A* je *L* ili *R* uvodi se: $\delta_1([q, U], [X, Y]) = ([p, U], [Z, Y], A)$ ako je $\delta_2(q, X) = (p, Z, A)$

d) Za svaki znak trake $[X, Y] \in \Gamma_1$ gdje je $Y \neq c$ uvodi se: $\delta_1([q, D], [X, Y]) = ([p, D], [X, Z], A)$ ako je $\delta_2(q, Y) = (p, Z, \bar{A})$

e) $\delta_1([q, U], [X, c]) = \delta_1([q, D], [X, c]) = ([p, C], [Y, c], R)$ ako je $\delta_2(q, X) = (p, Y, A)$, gdje *C* je *U* ukoliko *A* je *R*, odnosno *C* je *D* ukoliko *A* je *L*.

TS s višestrukim trakama

Model TS-a s višestrukim beskonačnim trakama ima *k* glava za čitanje i pisanje i *k* dvostruko beskonačnih traka. Upravljačka jedinica donosi odluku na temelju dvije grupe parametara:

a) stanje upravljačke jedinice
b) *k* pročitanih znakova sa svih *k* traka

Jednim prijelazom TS može:

1) promijeniti stanje
2) upisati različite znakove na svih *k* traka
3) pomaknuti svaku od *k* glava nezavisno ulijevo ili udesno

Na jednoj traci se zapisuje niz koji se ispituje. Zbog toga se ta traka označava kao ulazna traka, dok se sve ostale trake označavaju kao radne trake.

Generiranje jezika pomoću TS-a

Za zadani TS M_1 koji generira jezik $G(M_1) = L$ može se izgraditi TS M_2 koji prihvaća jezik $L(M_2) = L$ na sljedeći način: Konstruira se TS M_2 tako da ima jednu traku više od TS M_1. TS M_2 simulira rad TS M_1 tako da koristi sve trake osim ulazne trake na kojoj je zapisan niz koji se ispituje da li je iz jezika $L(M_2)$. Svaki put kad tokom simulacije TS M_1 ispiše graničnik #, tada TS M_2 uspoređuje upravo generirani niz s nizom na ulaznoj traci. Ukoliko su dva niza ista, tada TS M_2 stane i prihvati niz na ulaznoj traci. Ukoliko dva niza nisu ista, tada TS M_2 nastavlja simulirati rad TS M_1.

Za zadani TS M_2 koji prihvaća jezik $L = L(M_2)$ može se izgraditi TS M_1 koji generira jezik $G(M_1) = L$.

⇨ Jednostavni postupak konstrukcije može se opisati na slijedeći način: U prvom koraku ispisuju se jedan po jedan niz $w_1, w_2, w_3, \ldots$ nad zadanom abecedom Σ na jednu od radnih traka. Određenim redoslijedom ispisuju se svi mogući nizovi iz skupa Σ^* (potrebno je uočiti da se nizovi ne ispisuju na izlaznu traku). Nakon što se ispiše jedan niz w_i, TS M_1 simulira rad TS M_2 s nizom w_i na ulazu. Ukoliko TS M_2 prihvaća niz w_i, tada TS M_1 generira taj niz na izlaznu traku. TS M_1 će generirati niz w_i na izlaznu traku ako i samo ako TS M_2 prihvaća niz w_i. Prema tome, ukoliko bi TS M_2 stao za svaki niz na ulazu, tada bi TS M_1 na izlaznoj traci generirao sve nizove iz jezika $L(M_2)$, odnosno $G(M_1) = L(M_2)$

⇨ Poboljšani postupak može se opisati na slijedeći način: Neka se prvo generira par (i, j), a zatim simulira rad TS M_2 za i-ti niz i to pomoću najviše *j* pomaka glave. Neka je redoslijed generiranja parova (i, j) određen prema rastućoj sumi $i + j$. Parovi koji imaju istu sumu generiraju se prema rastućem broju *i*. Parovi se generiraju slijedećim redoslijedom: $(1,1)$, $(1,2)$, $(2,1)$, $(2,2)$, … Neka TS M_1 izvodi sljedeći postupak: Prvo se generira par (i, j) prethodno opisanim redoslijedom. Zatim se generira niz w_i na jednu od radnih traka i simulira se rad TS M_2 za niz w_i i to u *j* koraka. Ukoliko TS M_2 prihvaća niz w_i u točno *j* pomaka glave, tada TS M_1 generira niz w_i na izlaznu traku. Na ovaj način se sprečava da TS M_1 neprestano ispituje jedan niz ukoliko je ispitivani jezik rekurzivno prebrojiv, a nije rekurzivan (tada se može desiti da TS M_2 nikad ne stane).

Nedeterministički TS

Nedeterministički TS je TS s jednom trakom koja je beskonačna samo u desno. Za razliku od funkcije prijelaza δ deterministčkog TS-a koja je za svaki par stanja i ulaznog znaka jednoznačna, funkcija δ nedeterminističkog TS-a daje mogućnost konačnog broja izbora:

$$\delta(q, X) = \{(p_1, Z_1, D_1), (p_2, Z_2, D_2), \ldots, (p_k, Z_k, D_k)\}.$$

Niz se prihvaća ukoliko barem jedan slijed prijelaza vodi NTS u jedno od prihvatljivih stanja. Niz se ne prihvaća ukoliko niti jedan slijed prijelaza ne vodi NTS u niti jedno od prihvatljivih stanja.

Konstrukcija deterministčkog TS-a iz nedeterminističkog TS-a

Izgradi se deterministički TS M_2 koji ima tri trake. Prva traka je ulazna i na njoj je zapisan niz koji se ispituje. Na drugoj traci TS M_2 generira slijed brojeva $n_1, n_2, \ldots, n_m$ po slijedećem pravilu: TS M_2

prvo generira one sljedove brojeva $n_1, n_2, \ldots, n_m$ koji su kraći. Ukoliko dva slijeda brojeva imaju istu dužinu, tada je redoslijed generiranja određen njihovom numeričkom vrijednošću.
Za svaki generirani slijed brojeva $n_1, n_2, \ldots, n_m$ TS M_2 kopira niz s ulazne trake na treću traku. TS M_2 simulira rad TS M_1 tako da koristi treću traku kao ulaznu, dok niz zapisan na drugoj traci određuje koju funkciju prijelaza TS M_1 će se primijeniti.
Ukoliko TS M_1 prihvati niz, tada ga prihvaća i TS M_2. Ukoliko postoji slijed prijelaza koji vodi TS M_1 u jedno od prihvatljivih stanja, tada će TS M_2 na drugoj traci sigurno generirati slijed brojeva $n_1, n_2, \ldots, n_m$ koji odgovara slijedu prijelaza koji vodi u jedno od prihvatljivih stanja.

TS s višedimenzionalnim poljem
Osnovni model TS-a proširi se tako da se umjesto jednodimenzionalne trake uvede k-dimenzionalno polje ćelija koje je beskonačno u svih *2k* smjerova, gdje je *k* konačan broj. Na temelju stanja i učitanog znaka, TS odlučuje o promjeni stanja, ispisuje novi znak u ćeliju na kojoj je pozicionirana glava i pomaku glave. Glava se može pomaknuti uzduž svake osi od *k* osi u pozitivnom ili negativnom smjeru (ukupno *2k* različitih načina pomicanja glave). Na početku rada TS-a ulazni niz je zapisan uzduž jedne od *k* osi i glava je pozicionirana na krajnje lijevi znak ulaznog niza.
U bilo kojem trenutku rada TS-a, samo konačan broj redaka u bilo kojoj dimenziji može imati neprazne ćelije i svaki od tih redaka može imati konačan broj neprazih ćelija.

Simulacija TS M_2 s dvodimenzionalnim poljem s TS M_1 s jednodimenzionalnom trakom
Neka se svi znakovi dvodimenzionalnog polja različiti od praznina zaokruže četverokutom. Sadržaji svih ćelija u četverokutu se pohrane na jednodimenzionalnu traku red po red tako da pojedini redovi čine blokove odvojene znakom *****. Početak prvog bloka i kraj zadnjeg bloka neka se označe sa dva znaka *****. Jednodimenzionalna traka ima dva traga, gdje se drugi trag koristi za označavanje pozicije glave.
U opisu simulacije posebno se promatraju četiri pomaka glave TS M_2:

Horizontalni pomak unutar četverokuta: TS M_1 miče oznaku pozicije glave lijevo i desno za jednu ćeliju ovisno da li TS M_2 miče glavu lijevo ili desno.

Vertikalni pomak unutar četverokuta: TS M_1 koristi posebnu traku kao brojilo. Pomoću brojila utvrđuje se pozicija oznake glave unutar bloka. Ukoliko je pomak na dolje (gore), tada se oznaka pozicije glave miče u desno (lijevo) preko ćelija u trenutnom bloku do oznake *****, a zatim preko ćelija desnog (lijevog) bloka na istu poziciju unutar bloka kao i prije.

Vertikalni pomak izvan četverokuta: TS M_1 nadodaje jedan blok praznina na krajnju desno ili lijevu stranu. Broj praznina u bloku određuje se pomoću posebne trake koja se koristi kao brojilo. Nakon toga TS M_1 simulira vertikalni pomak unutar četverokuta.

Horizontalni pomak izvan četverokuta: Najprije TS M_1 proširuje svaki od blokova jednom prazninom na lijevoj ili desnoj strani bloka, a zatim simulira horizontalni pomak unutar četverokuta.

TS s više glava za čitanje i pisanje
To je prošireni model Turingovog stroja koji ima *k* glava za čitanje i pisanje, te jednu traku. Glave se mogu nezavisno micati ulijevo ili udesno. TS donosi odluku na temelju stanja i na temelju *k* učitanih znakova.

Konstrukcija osnovnog modela TS M_1 iz TS M_2 s više glava za čitanje i pisanje
Konstruira se TS M_1 s jednom glavom za čitanje i pisanje, te s jednom trakom koja ima *k+1* tragova. Jedan trag se koristi za pohranjivanje sadržaja trake TS M_2 dok ostalih *k* tragova služi za označavanje pozicija *k* glava za čitanje i pisanje TS M_2. Za simulaciju TS M_2 koristi se algoritam sličan onom kod simulacije Turingovog stroja s višestrukim trakama.

Neizravan TS
Neizravan TS je TS s višestrukim trakama čija ulazna traka se može samo čitati. Niz na ulaznoj traci je ograđen posebnim oznakama (graničnicima) **c** i **$**. Glava za čitanje i pisanje ulazne trake ne može prepisati graničnike **c** i **$** drugim znakovima, niti TS može pomaknuti glavu izvan područja omeđenog tim graničnicima.

Simulacija računala zasnovanog na memoriji sa slobodnim pristupom Turingovim strojem

TS ima više traka. Jedna traka se koristi za pohranu adresa i sadržaja memorijskih lokacija. Adrese i sadržaji memorijskih lokacija pohranjuju se na ovaj način: $\#0*v_0\#1*v_1\#10*v_2"\#\ldots\#i*v_i$, gdje je v_i sadržaj memorijske lokacije čija je adresa i. To je momorijska traka.

Druga traka se koristi za pohranu sadržaja aritmetičkih registara (registarska traka). Može se organizirati kao i memorijska, s tim da adresni dio selektira određeni registar. Budući da ima konačni broj registara i njihov broj se ne mijenja tokom rada RAM-a, tada se adresni dio može izostaviti.

Treća traka se koristi kao programsko brojilo.

Četvrta traka koristi se kao memorijski adresni registar.

Način izvođenja naredbe za svaki tip može biti definiran pomoću funkcije prijelaza TS. Nakon što se naredba dekodira, jedna od komponenti stanja TS može poprimiti vrijednost tipa naredbe. Drugi način je da se izgradi posebna traka na kojoj je zapisan "mikrokod" za svaku naredbu.

Potisni automat kao ograničeni model Turingovog stroja

PA je ekvivalentan nedeterminističkom TS koji ima dvije trake: jednu ulaznu i jednu stogovnu radnu traku.

Ulazna traka može samo čitati i glava za čitanje može se micati samo udesno.

Stogovna radna traka: Funkcije prijelaza na njoj su ograničene funkcije stoga. Prilikom pomaka glave ulijevo u ćeliju se uvijek upisuje praznina.

Stroj s više stogova

Deterministički stroj s dva stoga je deterministički TS s jednom ulaznom trakom i dvije stogovne radne trake. Ulazna traka može samo čitati, dok stogovna traka radi isto što i kod prethodnog ograničenog modela Turingovog stroja.

Stroj s brojilima

Model determinističkog stroja s dva stoga može se i dalje ograničiti tako da stroj umjesto dvije dtogovne radne trake ima četiri brojila. Jedno brojilo gradi se pomoću stogovne radne trake. Stogovne radne trake imaju samo dva znaka: Z i prazninu B. Znak Z koristi se kao dno stoga. Dno stoga se može nalaziti samo na krajnje lijevoj ćeliji. Cjelobrojna vrijednost i pohranjuje se tako da se glava radne trake pomakne udesno za i ćelija od znaka Z. Brojilo se može povećati ili smanjiti za jedan pomakom glave udesno ili ulijevo. Ukoliko glava čita znak Z, tada je vrijednost brojila 0.

TS s jednom trakom može se simulirati pomoću stroja s četiri brojila.

Gramatika neograničenih produkcija

Gramatika s neograničenim produkcijama označava se kao uređena četvorka $G=(V,T,P,S)$. Relacija $\Rightarrow$ definira se kao: $\gamma\alpha\delta\Rightarrow\gamma\beta\delta$ ukoliko je $\alpha\to\beta$ produkcija gramatike G. Neka je relacija $\overset{*}{\Rightarrow}$ refleksivno i tranzitivno okruženje relacije $\Rightarrow$.

Gramatika $G=(V,T,P,S)$ generira jezik: $L(G)=\left\{w \mid w\in T^* \wedge S\overset{*}{\Rightarrow} w\right\}$

Konstrukcija TS za jezik zadan gramatikom s neograničenim produkcijama

Ako je $L(G)$ jezik kojeg generira gramatika $G=(V,T,P,S)$ s neograničenim produkcijama, tada je $L(G)$ rekurzivno prebrojiv jezik i za njega se može izgraditi nedeterministički TS M s dvije trake. Neka se s w označi niz koji se ispituje da li je iz $L(G)$. Niz w se upiše na prvu traku. Na drugu traku se na početku upiše početni nezavršni znak gramatike S. Tokom rada TS na drugoj traci će se nalaziti međunizovi α koje generira gramatika G.

Rad TS M može se opisati na sljedeći način:

1.) TS nedeterministički izabere mjesto i u nizu α koji se nalazi zapisan na drugoj traci, gdje je $1\le i\le|\alpha|$. Mjesta se sustavno izaberu počev od krajnje lijevog mjesta pa pomicanjem udesno za jedno mjesto.

2.) TS nedeterministički izabire produkciju $\beta\to\gamma$ gramatike G

3.) Ukoliko se niz β pojavi počev od mjesta *i*, tada se β zamjenjuje sa γ u nizu α. Ovisno o dužinama niza $|\gamma|$ i $|\beta|$, moraju se na odgovarajući način pomaknuti znakovi niza α

4.) Sada se uspoređuje novonastali niz α na traci 2 s nizom *w* zapisanim na prvoj traci. Ukoliko niz α odgovara nizu *w*, tada se niz prihvaća. Ukoliko nizovi nisu isti, tada se nastavlja rad TS od koraka (1)

Konstrukcija gramatike za jezik zadan TS

Ako je $L = L(M)$ rekurzivno prebrojiv jezik kojeg prihvaća TS M, tada postoji gramatika $G = (V, T, P, S)$ s neograničenim produkcijama koja generira jezik $L(G) = L(M)$ i postoji $TS\ M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ koji prihvaća jezik *L*.

Produkcije gramatike mogu se podijeliti u četiri skupine.

U prvoj skupini su produkcije koje "nedeterministički" generiraju niz nezavršnih znakova čije dvije komponente predstavljaju dvije kopije niza završnih znakova $a_1 a_2 - - - a_n$. Pored nezavršnih znakova koji predstavljaju dvije kopije niza koji se ispituje, gramatika generira nezavršni znak koji označava početno stanje q_0 TS M. Time je definirana početna konfiguracija TS M:

1.) $A_1 \rightarrow q_0 A_2$

2.) $A_2 \rightarrow [a, a] A_2$, za svaki $a \in \Sigma$, gdje su A_1 i A_2 nezavršni znakovi gramatike. A_1 je ujedno i početni nezavršni znak.

U drugoj skupini su produkcije koje "nadodaju" potreban broj praznih ćelija koje koristi TS M prilikom ispitivanja niza $a_1 a_2 - - - a_n$:

3.) $A_2 \rightarrow A_3$

4.) $A_3 \rightarrow [\varepsilon, B] A_3$

5.) $A_3 \rightarrow \varepsilon$

Slijedeća skupina produkcija simulira rad TS M. Za svaki ulazni znak $a \in \Sigma \cup \{\varepsilon\}$, stanje $q \in Q$ i za svaki ulazni znak trake $X \in \Gamma$ za koje postoji prijelaz $\delta(q, X) = (p, Y, R)$ uvodi se produkcija:

6.) $q[a, X] \rightarrow [a, Y] p$, gdje je $p \in Q$ i $Y \in \Gamma$.

Ukoliko postoji prijelaz $\delta(q, X) = (p, Y, L)$, uvodi se produkcija:

7.) $[b, Z] q [a, X] \rightarrow p [b, Z] [a, Y]$, gdje su $a, b \in \Sigma \cup \{\varepsilon\}$, $p, q \in Q$ i $X, Y, Z \in \Gamma$.

Gramatika *G* generira niz $a_1 a_2 - - - a_n$ onda i samo onda ako TS M prihvaća taj isti niz. TS M prihvaća niz $a_1 a_2 - - - a_n$ ako i samo ako TS M iz konfiguracije $a_0 a_1 a_2 - - - a_n$ primjenom funkcije prijelaza uđe u jedno od prihvatljivih stanja. Zbog toga se uvode sljedeće produkcije za sva stanja *q* iz skupa *F*, za svaki $a \in \Sigma \cup \{\varepsilon\}$ i za svaki znak $X \in \Gamma$:

8.) $[a, X] q \rightarrow qaq$

9.) $q[a, X] \rightarrow qaq$

10.) $q \rightarrow \varepsilon$

Kontekstno-ovisna gramatika

Neka se oblik produkcija $\alpha \rightarrow \beta$ ograniči tako da broj znakova u nizu β ne bude manji od broja znakova u α, odnosno neka je $|\alpha| \leq |\beta|$. Ukoliko sve produkcije gramatike *G* zadovoljavaju dano ograničenje, tada se kaže da je gramatika *G* Kontekstno-ovisna gramatika.

Linearno ograničen automat (LOA)

LOA je ograničeni NTS sa sljedećim svojstvima:

1.) Skup ulaznih znakova Σ sadrži dva posebna znaka: **c** i **$**. Znak **c** je lijevi graničnik na ulaznoj traci, a znak **$** je desni graničnik.

2.) LOA ne može pomaknuti glavu za čitanje i pisanje lijevo od znaka *c*, niti može pisati preko znaka *c*. Isto tako LOA ne može pomaknuti glavu za čitanje i pisanje desno od znaka *$*, niti može pisati preko njega.

LOA se zadaje kao uređena osmorka: $LOA = (Q, \Sigma, \Gamma, \delta, q_0, c, S, F)$.

Konstrukcija LOA za jezik zadan kontekstno-ovisnom gramatikom

Konstrukcija LOA je slična konstrukciji TS za kontekstno-neovisan jezik, osim što se umjesto druge trake koristi drugi trag ulazne trake. Na početku se u gornjem tragu LOA nalazi *cw$*. U donji trag, točno ispod prvog znaka niza *w*, zapiše se početni nezavršni znak *S*.

Ukoliko je niz *w* prazan niz *ε*, tada LOA odmah stanje i ne prihvaća prazan niz. Ukoliko niz *w* nije prazan, tada LOA nedeterministički izabire mjesto u generiranom međunizu i nedeterministički izabire produkciju koja će se primijeniti. Budući da su desne strane produkcija kontekstno-ovisne gramatike iste dužine ili duže od lijevih strana, tada se jedino može dogoditi da je novo generirani međuniz iste dužine ili duži od prethodnog međuniza. Ukoliko se dogodi da generirani međuniz postane duži od niza *w*, tada se zaustavlja daljnje generiranje međunizova bez ulaska LOA u prihvatljivo stanje. LOA ne prihvaća niti jedan niz *w* za koji postoji međuniz α takav da $S \overset{*}{\vdash} \alpha$ i $|\alpha| > |w|$.

Konstrukcija kontekstno-ovisne gramatike za jezik zadan LOA

Slijedeće produkcije generiraju niz nezavršnih znakova $[a_1, a_1][a_2, a_2] - - - [a_n, a_n]$:

1.) $A_1 \rightarrow [a, q_0 c a] A_2$

2.) $A_1 \rightarrow [a, q_0 c a S]$

3.) $A_2 \rightarrow [a, a] A_2$, za svaki $a \in \Sigma$

4.) $A_2 \rightarrow [a, aS]$, za svaki $a \in (\Sigma - \{c, S\})$

Za svaki od prijelaza LOA M uvodi se jedna od produkcija:

5.) $[b, qX][a, Z] \rightarrow [b, Y][a, pZ]$

$[b, qcX] \rightarrow [b, cpX]$

$[b, cqX][a, Z] \rightarrow [b, cY][a, pZ]$

$[b, qX][a, ZS] \rightarrow [b, Y][a, pZS]$

$[b, qXS] \rightarrow [b, YpS]$

6.) $[b, Z][a, qX] \rightarrow [b, pZ][a, Y]$

$[b, ZqS] \rightarrow [b, pZS]$

$[b, Z][a, qXS] \rightarrow [b, pZ][a, YS]$

$[b, cZ][a, qX] \rightarrow [b, cpZ][a, Y]$

$[b, cqX] \rightarrow [b, pcY]$

7.) $[a, qcXS] \rightarrow [a, cpXS]$

$[a, cqXS] \rightarrow [a, cYpS]$

$[a, cXqS] \rightarrow [a, cpXS]$

$[a, cqXS] \rightarrow [a, pcYS]$

Ukoliko je $q \in F$, tada se uvode produkcije koje pretvaraju nezavršne znakove u završne:

8.) $[a, \alpha q \beta] \rightarrow a$ za sve $a \in \{\Sigma - \{c, S\}\}$ i sve moguće nizove α i β.

Pretvorba nezavršnog znaka u završni dozvoljava se isto tako ukoliko nezavršni znak ima za susjeda završni znak:

9.) $[a, \alpha] b \rightarrow ab$

10.) $b[a, \alpha] \rightarrow ba$