Formalne metode

u softverskom inženjerstvu

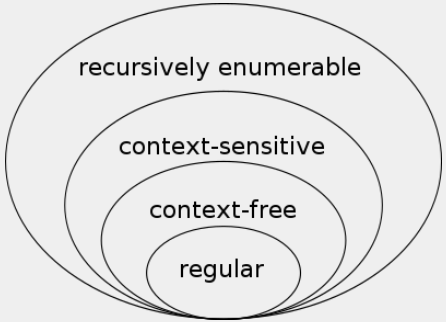
10 potisni automati

ETFBL 24-25

Dunja Vrbaški

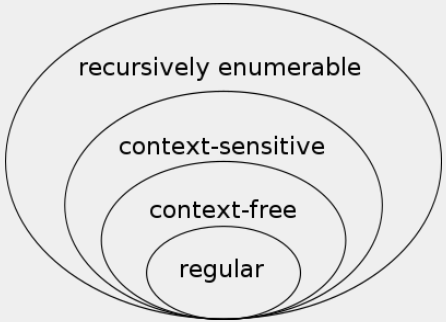
- regularna gramatika (regularni izrazi) generiše reči regularanog jezika
- konačni automat prepoznaje reči regularnog jezika

- CFG gramatika generiše reči CFG jezika
- potisni automat prepoznaje reči CFG jezika
- mora imati neku memoriju

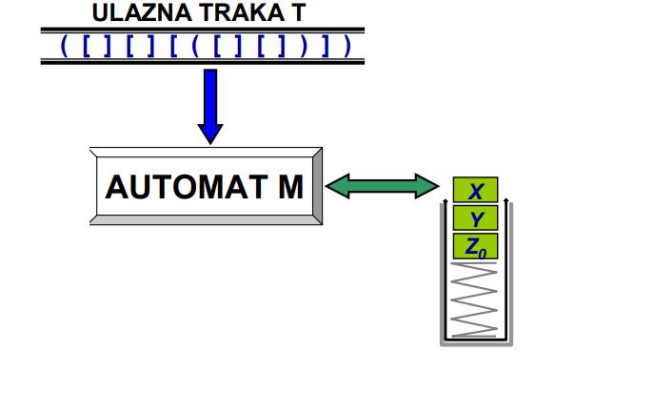


Potisni automat

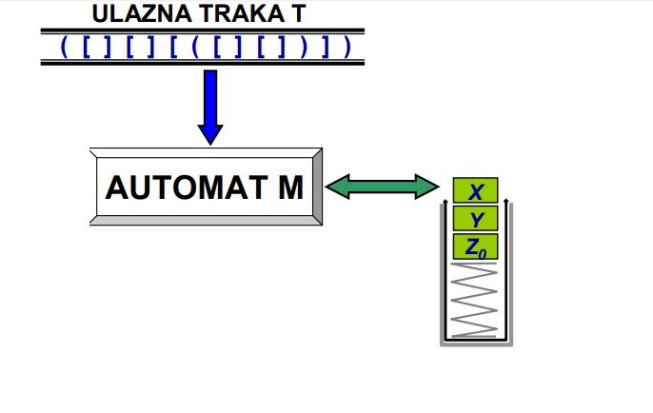
→ konačni automat + stek
← jednostavna tjuringova mašina



eng. Pushdown Automata



NKA + stek
prelaz: trenutno stanje, ulazni simbol + vrh steka



Početak:

- na traci se nalazi ulazni string
- glava trake se nalazi na početku trake
- automat je u početnom stanju
- stek je prazan
  (može imati samo jedan specijalan simbol)

Korak:
Zavisno od trenutnog stanja automata, simbola na koji pokazuje glava trake i vrha steka

- automat prelazi u novo stanje
- glava se pomera ili ne
- vrh steka se menja
  - pop ili
  - pop + novi elementi na steku (koji može sadržati i sklonjeni)

Kraj:
?

Potisni automat

A = (Σ, Γ, S, σ, $s_{0,}$, F)

- Σ - alfabet trake
- Γ - alfabet steka (sadrži specijalni, inicijalni, simbol, Z_0 ili \$)
- S - skup stanja
- σ - funkcija prelaza, σ: S x (Σ U ε) x $\Gamma \rightarrow$ S x {N, R} x Γ^*
- s_0 - inicijalno stanje
- F - skup finalnih stanja

postoje različite definicije

σ: S x (Σ U ε) x $\Gamma \rightarrow$ S x {N, R} x Γ^*

σ(s, a, A) = (p, x, w)

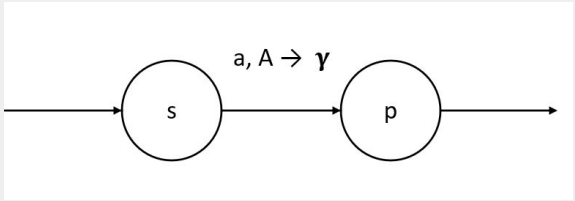
- automat se prebacuje iz stanja s u stanje p
- na traci je simbol a
- glava trake:
  x = R pomera se za jedan simbol
  x = N ne pomera se
- na steku se simbol A menja stringom w (reč nad azbukom Γ)

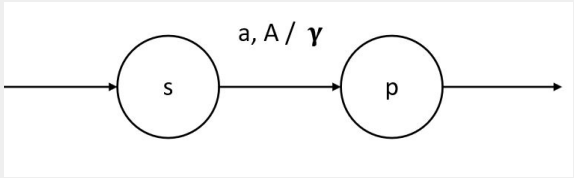
Prihvatanje reči:
- finalno stanje
- stek je prazan

(pojasnjenje u nastavku)

Notacije

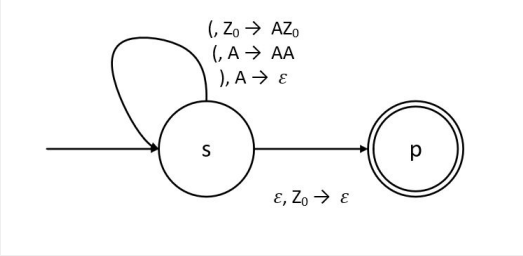
Korak:
$(s, ax, A\gamma) \vdash (p, x, \gamma)$, $a \in \Sigma \cup \varepsilon$, $x \in \Sigma^*$, $\gamma \in \Gamma^*$





PDA za balansirane zagrade

()((()))()



A = (Σ, Γ, S, σ, s_0, F)

- Σ = { (,) }
- Γ = { Z_0, A}
- S = {s, p}
- σ: S x (Σ U ε) x Γ → S x {N, R} x Γ^*
- s_0 = {s}
- F = {p}

(s, (w, Z_0) ⊢ (s, w, AZ_0)
(s, (w, A) ⊢ (s, w, AA)

(s,)w, A) ⊢ (s, w, ε)
(s,)w, Z_0) odbacivanje - *stek je prazan pre kraja (više desnih)*

(s, ε, Z_0) ⊢ (p, ε, ε) - prihvatanje
(s, ε, A) odbacivanje - *string je gotov (više levih)*

Zadatak

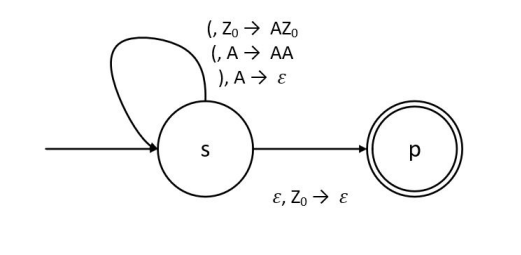
Kako bi izgledao PDA za različite vrste zagrada?

Deterministički vs nedeterministički

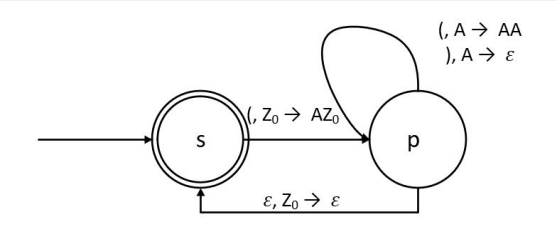
σ: S x (Σ U ε) x $\Gamma \rightarrow$ S x {N, R} x Γ^*

PDA je deterministički ako uvek postoji najviše jedan prelaz za svaku kombinaciju ulaznog simbola i trenutnog simbola na vrhu steka, u suprotnom je nedeterministički.

Iako imamo e-prelaz, ne dešava se nedeterminizam.
Jedino je mogući i najviše je jedan takav.



U stanju s ako je ulazni simbol “(“ i na steku je samo Z_0 imamo dva prelaza.
→ Nije deterministički



Prelazi su jedinstveni
→ Jeste deterministički

Za svaki regularan jezik se može konstruisati DFA (DFA <-> NDFA).

DFA i NDFA su iste moći

Postoje CFG jezici za koje se ne može konstruisati deterministički PDA.

NPDA su moćniji od DPDA

Rekli smo, prihvatanje reči:

- finalno stanje
- stek je prazan

$L_1(M) = \{w \in \Sigma^* \mid (s_0, w, Z_0) \vdash (f, \varepsilon, \gamma), f \in F, \gamma \in \Gamma^*\}$

prihvata završnim stanjem

$L_2(M) = \{w \in \Sigma^* \mid (s_0, w, Z_0) \vdash (p, \varepsilon, \varepsilon), p \in S\}$

prihvata praznim stekom

Za svaki automat M koji prihvata završnim stanjem se može konstruisati automat koji prihvata praznim stekom i obrnuto.

PDA za 0^n1^n

?

PDA za 0^n1^n

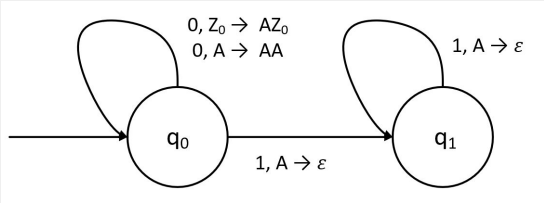
- dva stanja (za nule i jedinice)

- svaka pročitana 0
  - automat: isto stanje (inicijalno)
  - glava: pomera se
  - stek: odgovarajući simbol

- svaka pročitana 1
  - automat: isto stanje (stanje za 1)
  - glava: pomera se
  - stek: pop

- prelaz sa 0 na 1

$q_0 0\$ \to q_0 R\$S$	push S onto the stack
$q_0 0S \to q_0 RSS$	push S onto the stack
$q_0 1\$ \to q_0 N\$$	first symbol in the input is 1; loop forever
$q_0 1S \to q_1 R\epsilon$	first 1 is encountered
$q_0 \square\$ \to q_0 N\epsilon$	input string is empty; accept
$q_0 \square S \to q_0 NS$	input only consists of 0s; loop forever
$q_1 0\$ \to q_1 N\$$	0 to the right of 1; loop forever
$q_1 0S \to q_1 NS$	0 to the right of 1; loop forever
$q_1 1\$ \to q_1 N\$$	too many 1s; loop forever
$q_1 1S \to q_1 R\epsilon$	pop top symbol from the stack
$q_1 \square\$ \to q_1 N\epsilon$	accept
$q_1 \square S \to q_1 NS$	too many 0s; loop forever



prihvatanje praznim stekom

Rekli smo:

- CFG jezici se prepoznaju PDA automatima

Tvrđenje:
Jezik je kontekstno nezavisan akko postoji PDA koji prihvata taj jezik.

Treba pokazati:

- za proizvoljnu CFG postoji odgovarajući PDA
- za proizvoljni PDA postoji odgovarajuća CFG

CFG → PDA
Ideja: parsiranje

Posebne forme CFG gramatika

- dokazi
- automatizacija parsiranja

Pojednostavljenje gramatike:

- odbacivanje beskorisnih simbola
- odbacivanje e-produkcija
- odbacivanje jediničnih produkcija