Programski prevodioci (vežbe)

"Top-down" sintaksna anailza,

LL(1) gramatike

Algoritmi za sintaksnu analizu

- “Top-down” algoritmi – algoritmi koji sintaksno stablo kreiraju “sa vrha ka dnu”, tj. ovi algoritmi pokušavaju da startni simbol gramatike preslikaju u kod koji analiziraju.
- “Bottom-up” algoritmi – algoritmi koji sintaksno stablo kreiraju “sa dna ka vrhu”, tj. ovi algoritmi pokušavaju da ulazni kod redukuju na startni simbol gramatike.

Opšti “top-down” algoritam za sintaksnu analizu

1. U izvedenu sekvencu upisati startni simbol, proglasiti ga za tekući u izvedenoj sekvenci i pročitati prvi simbol iz ulaznog koda.
2. Ukoliko je tekući simbol u izvedenoj sekvenci neterminalni simbol, zameniti ga desnom stranom prve smene na čijoj je levoj strani taj neterminalni simbol.
3. Ukoliko je tekući simbol u izvedenoj sekvenci terminalni simbol jednak tekućem ulaznom simbolu, prihvati ga (preći na analizu sledećeg simbola).
4. Ukoliko je tekući simbol u izvedenoj sekvenci terminalni simbol različit od tekućeg ulaznog simbola, poništiti dejstvo poslednje primenjene smene. Ukoliko postoji još koja smena za preslikavanje istog neterminalnog simbola, pokušati sa primenom sledeće smene, u suprotnom vratiti se još jedan korak nazad.
5. Ukoliko vraćanjem dođemo do startnog simbola i ne postoji više smena za njegovo preslikavanje, ulazni kod sadrži sintaksnu grešku.
6. Ukoliko nakon prihvatanja poslednjeg ulaznog simbola, ni u izvedenoj sekvenci nema neobrađenih simbola, kod je sintaksno korektan.

Primer

- Metodom “top-down” proveriti da li reč ‘abbc’ pripada jeziku koji je definisan gramatikom zadatom sledećim skupom smena:

  $A \rightarrow \mathbf{a}\ B \mid \mathbf{a}\ C$

  $B \rightarrow \mathbf{b}\ B \mid \mathbf{b}$

  $C \rightarrow \mathbf{b}\ C \mid \mathbf{c}$

Uslov primenljivosti “top-down” algoritma za sintaksnu analizu

- Bilo koji “top-down” algoritam za sintaksnu analizu je primenljiv ukoliko gramatika ne sadrži levo-rekurzivne smene (kako direktne, tako i indirektne).

Pravila za eliminaciju levo-rekurzivnih smena

- Eliminisanje direktnih levih rekurzija:
  - Skup smena oblika:

    $X \rightarrow X\alpha \mid \beta$

    se može zameniti sledećim skupom smena:

    $X \rightarrow \beta\, X'$

    $X' \rightarrow \alpha X' \mid \varepsilon$

Pravila za eliminaciju levo-rekurzivnih smena

Jezik koji je zadat osnovnim skupom smena

$X \to X\alpha \mid \beta$ dobijamo kao:

$X \to X\alpha \to X\alpha\alpha \to \ldots \to X\alpha\alpha\ldots\alpha \to \beta\alpha\alpha\ldots\alpha$

Jezik koji je zadat transformisanim skupom smena

$X \to \beta X'$

$X' \to \alpha X' \mid \varepsilon$ dobijamo kao:

$X \to \beta X' \to \beta\alpha X' \to \beta\alpha\alpha X' \to \ldots \to \beta\alpha\alpha\ldots\alpha X' \to \beta\alpha\alpha\ldots\alpha$

Osnovni i transformisani skup smena su ekvivalentni

Pravila za eliminaciju levo-rekurzivnih smena

- Indirektne levo-rekurzivne smene:

  U gramatici postoji indirektna leva rekurzija ukoliko postoji izvodjenje oblika:

  $$X \Rightarrow X\alpha$$

- Primer indirektno levo-rekurzivnog skupa smena:

  $$X \rightarrow Y\alpha$$

  $$Y \rightarrow X\beta \mid \gamma$$

- Eliminacija indirektnih levih rekurzija:

  Sve skupove smena oblika:

  $$X \rightarrow Y\alpha$$

  $$Y \rightarrow X\beta_1 \mid \beta_2 \ldots \mid \beta_n$$

  zameniti smenama:

  $$X \rightarrow X\beta_1\alpha \mid \beta_2\alpha \ldots \mid \beta_n\alpha$$

  a zatim se osloboditi direktnih levo-rekurzivnih smena.

Primer 1:

Blok naredbi u jeziku μPascal definisan je sledećim skupom smena:

Blok → **begin** *NizNar* **end**

NizNar → *NizNar* **;** *Naredba* | *Naredba*

Naredba → *Dodela* | *Blok*

Dodela → **ID** := *Izraz*

Izraz → *Izraz* + **CONST** | **CONST**

Metodom "top-down" proveriti da li je blok:

```
begin
   ID := CONST
end
```

sintaksno korektno napisan.

Korak 1: Eliminacija levo-rekurzivnih smena

1. Skup smena:

   $NizNar \rightarrow NizNar$ **;** $Naredba \mid Naredba$

   zamenjujemo smenama:

   $NizNar \rightarrow Naredba\ NizNar'$

   $NizNar' \rightarrow$ **;** $Naredba\ NizNar' \mid \varepsilon$

2. Skup smena:

   $Izraz \rightarrow Izraz$ **+ CONST | CONST**

   zamenjujemo smenama:

   $Izraz \rightarrow$ **CONST** $Izraz'$

   $Izraz' \rightarrow$ **+ CONST** $Izraz' \mid \varepsilon$

Transformisani skup smena

(1) *Blok* → **begin** *NizNar* **end**

(2) *NizNar* → *Naredba NizNar'*

(3) *NizNar'* → **;** *Naredba NizNar'*

(4) *NizNar'* → **ε**

(5) *Naredba* → *Dodela*

(6) *Naredba* → *Blok*

(7) *Dodela* → **ID** := *Izraz*

(8) *Izraz* → **CONST** *Izraz'*

(9) *Izraz'* → + **CONST** *Izraz'*

(10) *Izraz'* → **ε**

Korak 2: Analiza

LL(1) gramatike

- LL(1) gramatike omogućavaju “top-down” postupak sintaksne analize bez vraćanja.
- Osnovna ideja je da se ne pokušava uvek sa primenom prve smene za preslikavanje tekućeg neterminalnog simbola, nego se “pogleda” sledeći simbol iz ulaznog niza i na osnovu toga odluči koja će se smena primeniti.
- Poreklo imena LL(1)
  - Ulazni niz se analizira s Leva na desno.
  - Nalazi se Levo izvođenje sintaksnog stabla.
  - Pri odluci koja će se smena primeniti, “pogleda” se (1) naredni simbol iz ulaznog niza.

FIRST i FOLLOW funkcije

- Za formalnu definiciju LL(1) gramatika potrebno je definisati funkcije FIRST i FOLLOW.
- Ukoliko postoji smena oblika $X \rightarrow \alpha$, definiše se funkcija FIRST(α) koja sadrži sve terminalne simbole koji mogu da se nađu na početku reči izvedenih iz niza α, tj.

  $x \in \text{FIRST}(\alpha) \Leftrightarrow \alpha \Rightarrow x\beta$
- Za svaki neterminalni simbol gramatike X definiše se skup FOLLOW(X) koji sadrži sve terminalne simbole koji u razvoju mogu da se nađu iza simbola X, tj.

  $x \in \text{FOLLOW}(X) \Leftrightarrow S \Rightarrow \alpha X x\beta$,

  gde je S startni simbol gramatike.

Odredjivanje FOLLOW funkcije

- Po definiciji FOLLOW funkcija startnog simbola gramatike sadrži granični simbol #.
- Za određivanje FOLLOW funkcije neterminalnog simbola X posmatraju se desne strane smena u kojima se posmatrani simbol pojavljuje. Simbol X na desnoj strani smene može da se nađe u jednom od sledećih konteksta:
  - $Z \to \alpha\, X\, x\beta \wedge x \in \mathrm{V_T} \Rightarrow x \in \mathrm{FOLLOW}(X)$
  - $Z \to \alpha\, XY\beta \wedge Y \in \mathrm{V_N} \Rightarrow \mathrm{FIRST}(Y) \subset \mathrm{FOLLOW}(X)$

    (Napomena: ukoliko postoji izvodjenje $Y \Rightarrow \varepsilon$, simbol Y se preskače i gleda se nastavak smene)
  - $Z \to \alpha\, X \Rightarrow \mathrm{FOLLOW}(Z) \subset \mathrm{FOLLOW}(X)$

Formalna definicija LL(1) gramatike

- Gramatika G jeste LL(1) gramatika ako i samo ako za svaki skup smena koje na levoj strani imaju isti neterminalni simbol važi:

$$X \rightarrow \alpha_1 \mid \alpha_2 \mid \ldots \mid \alpha_n$$

1. FIRST(α_i) $\cap$ FIRST(α_j) = $\varnothing$, za sve parove $i \neq j$
2. Samo jedan od nizova $\alpha_1, \alpha_2, \ldots, \alpha_n$ se može preslikati u prazan niz ε.
3. Ukoliko se niz α_k preslikava u prazan niz, onda

FIRST(α_i) $\cap$ FOLLOW(X)= $\varnothing$, za $i \in [1,n]$.

Leva faktorizacija

- Iz uslova 1. sledi da gramatika ne može biti tipa LL(1) ukoliko sadrži smene koje imaju isti neterminalni simbol na levoj i isti prvi simbol na desnoj strani, tj. LL(1) gramatika ne sadrži smene oblika:

  $X \rightarrow Y\alpha_1 \mid Y\alpha_2$

- Ukoliko u gramatici postoje smene navedenog oblika, gramatika se transformiše u LL(1) gramatiku tako što se prikazani skup smena zamenjuje smenama:

  $X \rightarrow Y\,X'$

  $X' \rightarrow \alpha_1 \mid \alpha_2$

- Navedena transformacija se naziva levom faktorizacijom.

Postupak LL(1) sintaksne analize

1. U postupku se koriste 1 radni magacin koji čuva trenutno stanje u razvoju i niz koji pamti primenjene smene. Na početku analize se u radni magacin upisuje granični simbol i startni simbol gramatike i pročita se prvi simbol iz ulaznog niza.
2. Na osnovu simbola na vrhu radnog magacina i izdvojenog ulaznog simbola, određuje se akcija koju treba izvršiti. Sve moguće akcije zapamćene su u LL(1) sintaksnoj tabeli.
3. Korak 2 se ponavlja dok pročitana akcija iz sintaksne tabele ne bude “PRIHVATI” (acc) ili “GREŠKA” (err).

LL(1) sintaksna tabela

- LL(1) sintaksna tabela ima onoliko vrsta koliko je ukupno neterminalnih i terminalnih simbola gramatike (računajući i granični simbol kao fiktivni terminalni simbol) i onoliko kolona koliko je terminalnih simbola u gramatici.
- Elementi LL(1) sintaksne tabele su:

$$M(A,a)=\begin{cases} pop, & \text{ako je } A=a\,(a\in V_T) \\ acc, & \text{ako je } A=\text{ i } a= \\ (\alpha,i) & \text{ako je } i\text{-ta smena oblika } A\to\alpha \text{ i} \\ & (a\in FIRST(\alpha)\vee(\varepsilon\in FIRST(\alpha)\wedge a\in FOLLOW(A))) \\ err & \text{u ostalim slučajevima} \end{cases}$$

Primer 2:

- Metodom LL(1) sintaksne anaize proveriti da li je blok definisan u primeru 1 sintaksno ispravno zapisan.

Korak 1: provera da li gramatika jeste LL(1) gramatika

1. Oslobađanje od levo-rekurzivnih smena. Nakon ove transformacije gramatika postaje:

(1) *Blok* → **begin** *NizNar* **end**
(2) *NizNar* → *Naredba NizNar'*
(3) *NizNar'* → **;** *Naredba NizNar'*
(4) *NizNar'* → **ε**
(5) *Naredba* → *Dodela*
(6) *Naredba* → *Blok*
(7) *Dodela* → **ID** := *Izraz*
(8) *Izraz* → **CONST** *Izraz'*
(9) *Izraz'* → **+ CONST** *Izraz'*
(10) *Izraz'* → **ε**

Korak 1: provera da li gramatika jeste LL(1) gramatika

2. Analiza skupova smena koje na levoj strani imaju isti neterminalni simbol.

   I par takvih smena:

   (3) *NizNar'* → **;** *Naredba NizNar'*

   (4) *NizNar'* → **ε**

- FIRST(**;** *Naredba NizNar'*) = {;}

FIRST(**ε**) = {**ε**}

FIRST(**;** *Naredba NizNar'*) ∩ FIRST(**ε**) = ∅

- Samo jedna od navedenih smena se preslikava u prazan niz.
- FOLLOW(*NizNar'*) = FOLLOW(*NizNar*) = { end }

FIRST(**;** *Naredba NizNar'*) ∩ FOLLOW(*NizNar'*) = ∅

Korak 1: provera da li gramatika jeste LL(1) gramatika

Il par takvih smena:

$$(5) \quad Naredba \rightarrow Dodela$$

$$(6) \quad Naredba \rightarrow Blok$$

- FIRST($Dodela$) = { ID }

  FIRST($Blok$) = { begin }

  FIRST($Dodela$) $\cap$ FIRST($Blok$) = $\varnothing$
- Ni jedna od navedenih smena se ne preslikava u prazan niz.

Korak 1: Provera da li gramatika jeste LL(1) gramatika

III par takvih smena:

(9) $Izraz' \rightarrow + \mathbf{CONST}\ Izraz'$

(10) $Izraz' \rightarrow \varepsilon$

- FIRST(+ **CONST** *Izraz'*) = { + }
  FIRST(ε) = {ε }
  FIRST(+ **CONST** *Izraz'*) ∩ FIRST(ε) = ∅
- Samo jedna od navedenih smena se preslikava u prazan niz.
- FOLLOW(*Izraz'*) = { **;**, **end** }
  FIRST(+ **CONST** *Izraz'*) ∩ FOLLOW(*Izraz'*) = ∅

Korak 2: Kreiranje LL(1) sintaksne tabele

Korak 2: Sintaksna analiza

Korak 4: Kreiranje sintaksnog stabla