Analiza zstępująca. Gramatyki typu $LL(k)$

Języki formalne i techniki translacji - Wykład 9

Maciek Gębala

6 grudnia 2022

Notatki

Analiza metodą zstępującą

- Mamy ciąg tokenów (terminali) $w \in T^*$ i gramatykę $G = (N, T, P, S)$.
- Chcemy sprawdzić czy $w \in L(G)$?
- Szukamy lewostronnego wyprowadzenia dla w.
- Budujemy drzewo wyprowadzenia dla w zaczynając od korzenia i tworząc wierzchołki w porządku preorder.

Notatki

Przykład

- Weźmy napis $w = cad$ i gramatykę

$$\begin{aligned} S &\to cAd \\ A &\to ab|a \end{aligned}$$

- Zaczynamy od symbolu początkowego, pierwszą literą w jest c więc bierzemy produkcję $S \to cAd$ i wyprowadzamy $S \Rightarrow cAd$
- Pierwsza litera cAd jest zgodna z pierwszą literą w więc przechodzimy do symboli A i a. Możemy teraz użyć produkcji $A \to ab$ i otrzymać ciąg $cabd$.
- Druga litera się zgadza więc sprawdzamy trzecią. Niestety mamy d i b czyli źle. Musimy wrócić do A i poszukać alternatywnego wyprowadzenia.
- Używamy produkcji $A \to a$ i tym razem jest dobrze.
- Uzyskaliśmy wyprowadzenie: $S \Rightarrow cAd \Rightarrow cad$.

Notatki

Przykład

- Weźmy gramatykę

$$\begin{aligned} S &\to aAd|aB \\ A &\to b|c \\ B &\to ccd|ddc \end{aligned}$$

- Weźmy słowo $w = accd$
- $S \Rightarrow aAd$
- $S \Rightarrow aAd \Rightarrow abd$ źle, $abd \neq accd$, nawracamy
- $S \Rightarrow aAd \Rightarrow acd$ źle, $acd \neq accd$, nawracamy
- $S \Rightarrow aB$
- $S \Rightarrow aB \Rightarrow accd$ OK

Notatki

Faktoryzacja lewostronna

Kiedy nie jest jasne na podstawie pierwszego symbolu, którą produkcję wybrać do rozwinięcia nieterminala, przekształcamy te produkcje tak aby decyzję podjąć później.

Przykład

- $instr \rightarrow$ *if* *wyr* *then* *instr* *else* $instr|$*if* *wyr* *then* *instr*
- $instr \rightarrow$ *if* *wyr* *then* *instr koniec*
  $koniec \rightarrow$ *else* $instr|\varepsilon$

Notatki

Algorytm lewostronnej faktoryzacji gramatyki

- Dla każdego nieterminala A znajdź najdłuższy prefiks α wspólny dla co najmniej dwóch prawych stron A-produkcji.
- Jeśli $\alpha \neq \varepsilon$ to zamień produkcje

$$A \rightarrow \alpha\beta_1|\ldots|\alpha\beta_n|\gamma_1|\ldots|\gamma_m$$

  na produkcje

$$\begin{array}{lcl} A & \rightarrow & \alpha B|\gamma_1|\ldots|\gamma_m \\ B & \rightarrow & \beta_1|\ldots|\beta_n \end{array}$$

  gdzie B jest nowym dodatkowym nieterminalem.
- Transformację powtarzamy dopóki zmienia ona gramatykę.

Notatki

Analizatory przewidujące

Często uważnie tworząc gramatykę, usuwając w niej lewostronną rekurencję i wykonując lewostronną faktoryzację możemy uzyskać gramatykę, która przy wyprowadzeniu nie potrzebuje nawrotów.

Wyprowadzenie w takiej gramatyce może być łatwo sprawdzane prostym, deterministycznym automatem ze stosem.

Notatki

Przykład

- Weźmy gramatykę

$$\begin{array}{lcl} E & \rightarrow & E+T|T \\ T & \rightarrow & T*F|F \\ F & \rightarrow & (E)|id \end{array}$$

- Eliminujemy lewostronną rekurencję

$$\begin{array}{lcl} E & \rightarrow & TG \\ G & \rightarrow & +TG|\varepsilon \\ T & \rightarrow & FV \\ V & \rightarrow & *FV|\varepsilon \\ F & \rightarrow & (E)|id \end{array}$$

- Gramatyka nie potrzebuje lewostronnej faktoryzacji.

Notatki

Działanie analizatora przewidującego na $id + id * id$

Stos	Wejście	Wyjście
$\$E$	$id + id * id\$$	
$\$GT$	$id + id * id\$$	$E \rightarrow TG$
$\$GVF$	$id + id * id\$$	$T \rightarrow FV$
$\$GV$	$+id * id\$$	$F \rightarrow id$
$\$G$	$+id * id\$$	$V \rightarrow \varepsilon$
$\$GT$	$id * id\$$	$G \rightarrow +TG$
$\$GVF$	$id * id\$$	$T \rightarrow FV$
$\$GV$	$*id\$$	$F \rightarrow id$
$\$GVF$	$id\$$	$V \rightarrow *FV$
$\$GV$	$\$$	$F \rightarrow id$
$\$G$	$\$$	$V \rightarrow \varepsilon$
$\$$	$\$$	$G \rightarrow \varepsilon$

Notatki

Zbiory $FIRST$ i $FOLLOW$

- $FIRST$ pomaga wybrać produkcję którą możemy użyć do wyprowadzania napisu.

$$FIRST(\alpha) = \{x \in T \ : \ \exists_\beta \ \alpha \Rightarrow^* x\beta\}$$

- $FOLLOW$ pomaga synchronizować symbole podczas odzyskiwania kontroli w trybie paniki.

$$FOLLOW(A) = \{x \in T \ : \ \exists_\alpha \exists_\beta \ S \Rightarrow^* \alpha A x \beta\}$$

- Na podstawie tych funkcji generujemy tablicę analizatora przewidującego która dla nieterminala A i terminala a wyznacza którą produkcję możemy użyć.

Notatki

Wyznaczanie $FIRST(X)$

Dla wszystkich symboli X z gramatyki G tworzymy zbiory $FIRST$ według następujących reguł

1. Jeśli X jest terminalem to $FIRST(X) = \{X\}$.
2. Jeśli $X \rightarrow \varepsilon$ jest produkcją to do $FIRST(X)$ dodajemy ε.
3. Jeśli X jest nieterminalem i $X \rightarrow Y_1 Y_2 \ldots Y_k$ to a dodajemy do $FIRST(X)$ jeżeli istnieje i takie, że $a \in FIRST(Y_i)$ oraz $\varepsilon \in FIRST(Y_j)$ dla każdego $j < i$. $\varepsilon \in FIRST(X)$ jeśli należy do wszystkich $FIRST(Y_i)$

Ponadto

- $FIRST(X\alpha) = FIRST(X)$ gdy $\varepsilon \notin FIRST(X)$
- $FIRST(X\alpha) = FIRST(X) \cup FIRST(\alpha)$ gdy $\varepsilon \in FIRST(X)$

Notatki

Wyznaczanie $FOLLOW(A)$

Dla wszystkich nieterminali A $FOLLOW(A)$ tworzymy według następujących reguł

1. Dla symbolu początkowego S do $FOLLOW(S)$ dodajemy \$.
2. Jeśli mamy produkcję $A \rightarrow \alpha B \beta$ to do $FOLLOW(B)$ dodajemy wszystkie symbole z $FIRST(\beta)$ poza ε.
3. Jeśli mamy produkcję $A \rightarrow \alpha B$ albo produkcję $A \rightarrow \alpha B \beta$, gdzie $\varepsilon \in FIRST(\beta)$ to do $FOLLOW(B)$ dodajemy wszystkie symbole z $FOLLOW(A)$.

Notatki

Przykład

- $FIRST(E) = FIRST(T) = FIRST(F) = \{(, id\}$
- $FIRST(G) = \{+, \varepsilon\}$
- $FIRST(V) = \{*, \varepsilon\}$
- $FOLLOW(E) = FOLLOW(G) = \{), \$\}$
- $FOLLOW(T) = FOLLOW(V) = \{+,), \$\}$
- $FOLLOW(F) = \{+, *,), \$\}$

Notatki

Budowa tablic analizatora przewidującego

Dla każdej produkcji $A \to \alpha$

1. dla każdego $a \in T$ jeśli $a \in FIRST(\alpha)$ to wpisz $A \to \alpha$ do $M[A, a]$.
2. jeśli $\varepsilon \in FIRST(\alpha)$ to dla każdego $b \in FOLLOW(A)$ wpisz $A \to \alpha$ do $M[A, b]$.

Notatki

Przykład

	id	$+$	$*$	(	)	\$
E	TG			TG		
G		$+TG$			ε	ε
T	FV			FV		
V		ε	$*FV$		ε	ε
F	id			(E)		

Notatki

Gramatyki $LL(1)$

- Pierwsze L – przeglądanie od lewej do prawej.
- Drugie L – tworzenie lewostronnego wyprowadzenia.
- 1 – używanie do podejmowania decyzji jednego symbolu w każdym kroku.
- Dla każdego nieterminala z dwóch różnych prawych stron produkcji nie da się wyprowadzić ciągów zaczynających się od tego samego terminala.

Niestety nie wszystkie gramatyki bezkontekstowe dają się sprowadzić do postaci $LL(1)$.

Notatki

Gramatyki $LL(1)$

Gramatyka jest typu $LL(1)$ gdy z (rozpatrujemy wyprowadzenia lewostronne)

$$S \Rightarrow^* wA\alpha \Rightarrow w\beta\alpha \Rightarrow^* wx, \qquad S \Rightarrow^* wA\alpha \Rightarrow w\gamma\alpha \Rightarrow^* wy$$

oraz

$$FIRST(x) = FIRST(y)$$

wynika, że $\beta = \gamma$.

Inaczej: jeśli $S \Rightarrow^* wA\alpha \Rightarrow w\beta\alpha \Rightarrow^* wx$, to aby odgadnąć następny krok w wyprowadzeniu należy poznać pierwszy znak x.

Notatki

Testowanie na własność $LL(1)$

G jest $LL(1)$, gdy dla $S \Rightarrow^* wA\alpha$ i dowolnych produkcji $A \rightarrow \beta$, $A \rightarrow \gamma$ mamy

$$FIRST(\beta\alpha) \cap FIRST(\gamma\alpha) = \emptyset.$$

Zalety własności: FIRST można efektywnie obliczyć!

Notatki

Dowód własności

1. Niech $c \in FIRST(\beta\alpha) \cap FIRST(\gamma\alpha)$, $\beta \neq \gamma$. Wtedy potrafimy zbudować wyprowadzenia z $\beta\alpha$ i $\gamma\alpha$ uzyskując c na pierwszym miejscu. Otrzymamy $S \Rightarrow^* wA\alpha \Rightarrow w\beta\alpha \Rightarrow^* wcx$ i $S \Rightarrow^* wA\alpha \Rightarrow w\gamma\alpha \Rightarrow^* wcy$, ponieważ $FIRST(cy) = c = FIRST(cx)$, z definicji $LL(1)$ mielibyśmy $\beta = \gamma$. Zatem gramatyka nie jest $LL(1)$.
2. Niech G nie będzie $LL(1)$, tj.: jeśli $S \Rightarrow^* wA\alpha \Rightarrow w\beta\alpha \Rightarrow^* wcx$ i $S \Rightarrow^* wA\alpha \Rightarrow w\gamma\alpha \Rightarrow^* wcy$, wtedy $c \in FIRST(\beta\alpha) \cap FIRST(\gamma\alpha)$. (oczywiste!)

Notatki

Test na $LL(1)$

G jest $LL(1)$ wtedy i tylko wtedy, gdy dla każdego A i każdej produkcji $A \rightarrow \beta|\gamma$: $FIRST(\beta\, FOLLOW(A)) \cap FIRST(\gamma\, FOLLOW(A)) = \emptyset$.

Dowód ($\Leftarrow$) - z pustości wynika $LL(1)$

Oczywiście:

- Jeśli $S \Rightarrow^* wA\alpha \Rightarrow w\beta\alpha \Rightarrow^* wx$ to $FIRST(x)$ należą do $FIRST(\beta\, FOLLOW(A))$.
- Gdyby $S \Rightarrow^* wA\alpha \Rightarrow w\gamma\alpha \Rightarrow^* wy$ oraz $FIRST(x) = FIRST(y) = c$, to mielibyśmy $c \in FIRST(\beta\, FOLLOW(A)) \cap FIRST(\gamma\, FOLLOW(A))$.
- Ale jest to niemożliwe.

Notatki

Test na $LL(1)$

Zakładamy, że $c \in FIRST(\beta\, FOLLOW(A)) \cap FIRST(\gamma\, FOLLOW(A))$.

Dowód: ($\Rightarrow$) z $LL(1)$ wynika pustość

1. $c \in FIRST(\beta) \cap FIRST(\gamma)$ wtedy $S \Rightarrow^* vA\alpha \Rightarrow v\beta\alpha \Rightarrow^* vcx$ dla pewnego x, $S \Rightarrow^* vA\alpha \Rightarrow v\gamma\alpha \Rightarrow^* vcy$ dla pewnego y, i $FIRST(cy) = c = FIRST(cx)$. Równocześnie $\beta \neq \gamma$, więc G nie jest $LL(1)$!
2. $c \notin FIRST(\beta)$, $c \notin FIRST(\gamma)$, ale $c \in FOLLOW(A)$; $\varepsilon \in FIRST(\beta)$, $\varepsilon \in FIRST(\gamma)$. Wtedy: $S \Rightarrow^* vA\alpha \Rightarrow v\beta\alpha \Rightarrow^* v\alpha \Rightarrow^* vcx$. Także $S \Rightarrow^* vA\alpha \Rightarrow v\gamma\alpha \Rightarrow^* v\alpha \Rightarrow^* vcx$. Wbrew definicji $LL(1)$.

Notatki

Test na $LL(1)$

Dowód: ($\Rightarrow$) z $LL(1)$ wynika pustość

3. $c \in FIRST(\gamma)$, $c \notin FIRST(\beta)$, ale $c \in FOLLOW(A)$ i $\varepsilon \in FIRST(\beta)$. Wtedy: $S \Rightarrow^* wA\alpha \Rightarrow w\beta\alpha \Rightarrow^* w\alpha \Rightarrow^* wcx$. Także: $S \Rightarrow^* wA\alpha \Rightarrow w\gamma\alpha \Rightarrow^* wcy\alpha \Rightarrow^* wcycx$. Ponieważ $FIRST(cx) = c = FIRST(cycx)$ oraz $\beta \neq \gamma$, mamy sprzeczność z $LL(1)$.

Notatki

Tabele parsowania dla $LL(1)$

- $M[A, a]$ zawiera $A \to \beta$, jeśli $a \in FIRST(\beta\, FOLLOW(A))$.
- Zgodnie z poprzednim twierdzeniem nie ma konfliktów dla $LL(1)$.
- Parsing z taką tabelą określa jednoznacznie jedyne możliwe wyprowadzenie lewostronne.

Notatki

Gramatyka spoza $LL(1)$

- Gramatyka G

$$\begin{array}{lcl} S & \to & \varepsilon | abA \\ A & \to & Saa | b \end{array}$$

- G nie jest $LL(1)$:

$$\begin{array}{lcl} S & \Rightarrow & abA \Rightarrow abSaa \Rightarrow ababAaa \Rightarrow^* ab\underline{a}... \\ S & \Rightarrow & abA \Rightarrow abSaa \Rightarrow ab\underline{a}a \end{array}$$

- Jeden znak nie wystarczy aby rozróżnić pomiędzy $S \to \varepsilon$ i $S \to abA$ dla drugiego kroku wyprowadzenia.

Notatki

$FIRST_k$

$FIRST_k(\alpha)$ zdefiniowane tak jak $FIRST(\alpha)$:

- $\exists \alpha \Rightarrow^* w$, w składa się z terminali, $|w| \geqslant k$ i x jest prefiksem w długości k, to $x \in FIRST_k(\alpha)$, lub
- $\exists \alpha \Rightarrow^* w$, w składa się z terminali, $|w| < k$ i $x = w$, to $x \in FIRST_k(\alpha)$.

Notatki

Gramatyka $LL(k)$

Gramatyka jest $LL(k) \Leftrightarrow$ z

$$\begin{aligned} S &\Rightarrow^* wA\alpha \Rightarrow w\beta\alpha \Rightarrow^* wx \\ S &\Rightarrow^* wA\alpha \Rightarrow w\gamma\alpha \Rightarrow^* wy \end{aligned}$$

i

$$FIRST_k(x) = FIRST_k(y)$$

wynika, że $\beta = \gamma$.

Inaczej: jeśli $S \Rightarrow^* wA\alpha \Rightarrow w\beta\alpha \Rightarrow^* wx$, to wystarcza poznać k znaków x aby określić następny krok wyprowadzenia $S \Rightarrow^* wA\alpha$.

Notatki

Gramatyka nie-$LL(k)$ dla dowolnego k

Przykład

Gramatyka

$$\begin{aligned} S &\to A|B \\ A &\to aAb|0 \\ B &\to aBbb|1 \end{aligned}$$

definiuje

$$\{a^n 0 b^n : n \geqslant 0\} \cup \{a^n 1 b^{2n} : n \geqslant 0\}.$$

Jest to język rozpoznawany deterministycznym automatem ze stosem, ale nie jest w żadnym $LL(k)$.
(Blok symboli a może być dowolnie długi i nie można się zdecydować na $S \Rightarrow A$ lub $S \Rightarrow B$.)

Notatki

Podstawowa własność

G jest $LL(k)$ wtedy i tylko wtedy, gdy dla dowolnych $S \Rightarrow^* wA\alpha$, oraz $A \to \beta$, $A \to \gamma$ mamy

$$FIRST_k(\beta\alpha) \cap FIRST_k(\gamma\alpha) = \emptyset$$

Następująca własność nie jest równoważna $LL(k)$!

Jeśli $A \to \beta$, $A \to \gamma$, $\beta \neq \gamma$, to
$FIRST_k(\beta\, FOLLOW_k(A)) \cap FIRST_k(\gamma\, FOLLOW_k(A)) = \emptyset$

Przykład

$S \to aAaa|bAba$ i $A \to b|\varepsilon$, to widać z definicji, że gramatyka jest $LL(2)$.
Ale: $FOLLOW_2(A) = \{aa, ba\}$,
$FIRST_2(b\, FOLLOW_2(A)) \cap FIRST_2(\varepsilon\, FOLLOW_2(A)) = \{ba\}$

Notatki

Konstrukcja parsera $LL(k)$

- G - gramatyka $LL(k)$, $wx \in L_G$
- Konstruujemy lewostronne wyprowadzenie dla wx, zakładamy, że mamy $S \Rightarrow^* w\alpha$, takie że α zaczyna się nieterminalem, $\alpha \Rightarrow^* x$.
- Z definicji, z w i k następnych znaków x można określić następny krok wyprowadzenia.
- Problem: w nie można przechować na stosie (tam jest α).

Notatki

Tabele $LL(k)$

Notacja

$L_1 \oplus_k L_2 = \{w : \exists_{x \in L_1} \exists_{y \in L_2} (w = xy \wedge |xy| \leqslant k) \vee (w = FIRST_k(xy))\}$

$T_{A,L}$ trzeba traktować jednocześnie jako funkcję. Niech u ma długość k. Wtedy

1. $T_{A,L}(u) = \texttt{error}$, jeśli dla żadnej produkcji $A \to \alpha$ nie zachodzi $u \in FIRST_k(\alpha) \oplus_k L$,
2. $T_{A,L}(u) = (A \to \alpha, \langle Y_1, \ldots, Y_m \rangle)$ jeśli dla dokładnie jednej produkcji $A \to \alpha$ mamy $u \in FIRST_k(\alpha) \oplus_k L$, ($\langle Y_1, \ldots, Y_m \rangle$ zdefiniowane poniżej)
3. $T_{A,L}(u) = \texttt{error}$ jeśli dla więcej niż jednej produkcji $A \to \alpha$ mamy $u \in FIRST_k(\alpha) \oplus_k L$, (dla języka $LL(k)$ nie powinno to zajść).

Notatki

Definicja $\langle Y_1, \ldots, Y_m \rangle$

Niech $\alpha = x_0 B_1 x_1 B_2 x_2 \ldots B_m x_m$, gdzie B_i jest nieterminalem, a x_i to ciąg terminali. Wtedy:

$$Y_i = FIRST_k(x_i B_{i+1} \ldots B_m x_m \oplus_k L).$$

Y_i mówi jakie ciągi wyprowadzane za B_i są dopuszczalne o ile skorzystamy z $A \to \alpha$.

Notatki

Konstrukcja tabel $LL(k)$

Konstruujemy tylko takie $T_{A,L}$, które okazują się niezbędne: dla sytuacji początkowej $\mathcal{I} = \{T_{S,\{\varepsilon\}}\}$ rozszerzamy dopóty, dopóki coś nowego się pojawia. Reguła:

jeśli $T \in \mathcal{I}$ *i* $T(u) = (A \to x_0 B_1 x_1 B_2 x_2 \ldots B_m x_m, \langle Y_1, \ldots, Y_m \rangle)$, *dołącz* T_{B_i, Y_i} *do* $\mathcal{I}$.

Notatki

Parser $LL(k)$

1. Na początku $T_{S,\{\varepsilon\}}$ na stosie.
2. Krok parsera:
   - $M[T_{A,L}, u]$ zdefiniowany jak następuje: niech $T_{A,L}(u) = (A \to x_0 B_1 x_1 B_2 x_2 \dots B_m x_m, \langle Y_1, \dots, Y_m \rangle)$, wtedy usuń $T_{A,L}$ ze stosu, zapisz $x_0 T_{B_1,Y_1}, x_1, \dots, T_{B_m,Y_m}, x_m$ na stosie, (głowica nie porusza się).
   - $M[a, av]$ = usuń a ze stosu i przesuń głowicę o 1 pozycję,
   - $M[\$, \varepsilon]$ = accept,
   - $M[X, u]$ = reject, w p.p.

Notatki

Podstawowa własność

Dla każdej $LL(k)$-gramatyki $LL(k)$-parser określa wyprowadzenie lewostronne gdy $w \in L_G$, lub daje komunikat błędu w przeciwnym wypadku.

Dowód długi i oczywisty.

Notatki

Notatki

Notatki