METODY TRANSLACJI 2004

Folie z wykładu

notatka nr 3

Gramatyki LL(1)

Gramatyka nazywa się LL(1) gdy z

$$S \stackrel{*}{\Rightarrow}_l wA\alpha \Rightarrow_l w\beta\alpha \stackrel{*}{\Rightarrow}_l wx, \qquad S \stackrel{*}{\Rightarrow}_l wA\alpha \Rightarrow_l w\gamma\alpha \stackrel{*}{\Rightarrow}_l wy$$

oraz

$$\text{FIRST}(x) = \text{FIRST}(y)$$

wynika, że $\beta = \gamma$.

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

Przykład

następująca gramatyka jest LL(1):
$S \rightarrow aAS|b$
$A \rightarrow a|bSA$

Testowanie na własność LL(1)

G jest LL(1), gdy dla

$$S \stackrel{*}{\Rightarrow}_l wA\alpha, \text{ i dowolnych produkcji } A \rightarrow \beta, A \rightarrow \gamma$$

mamy

$$\text{FIRST}(\beta\alpha) \cap \text{FIRST}(\gamma\alpha) = \emptyset$$

Zalety własności: FIRST można efektywnie obliczyć!

Dowód własności

1) niech $c \in \text{FIRST}(\beta\alpha) \cap \text{FIRST}(\gamma\alpha)$, $\beta \neq \gamma$.

wtedy potrafimy zbudować wyprowadzenia z $\beta\alpha$ i $\gamma\alpha$ uzyskując c na pierwszym miejscu

otrzymamy

$$S \stackrel{*}{\Rightarrow}_l wA\alpha \Rightarrow_l w\beta\alpha \stackrel{*}{\Rightarrow}_l wcx, \qquad S \stackrel{*}{\Rightarrow}_l wA\alpha \Rightarrow_l w\gamma\alpha \stackrel{*}{\Rightarrow}_l wcy$$

ponieważ $\text{FIRST}(cy) = c = \text{FIRST}(cx)$, z definicji LL(1) mielibyśmy $\beta = \gamma$ zatem gramatyka nie jest LL(1)

Dowód własności

2) niech G nie będzie LL(1), tj.: jeśli

$$S \stackrel{*}{\Rightarrow}_l wA\alpha \Rightarrow_l w\beta\alpha \stackrel{*}{\Rightarrow}_l wcx, \qquad S \stackrel{*}{\Rightarrow}_l wA\alpha \Rightarrow_l w\gamma\alpha \stackrel{*}{\Rightarrow}_l wcy$$

wtedy $c \in \text{FIRST}(\beta\alpha) \cap \text{FIRST}(\gamma\alpha)$. (oczywiste!)

Test na LL(1)

G jest LL(1) wtedy i tylko wtedy, gdy dla każdego A i każdej produkcji $A \to \beta, A \to \gamma$:

$$\text{FIRST}(\beta\ \text{FOLLOW}(A)) \cap \text{FIRST}(\gamma\ \text{FOLLOW}(A)) = \emptyset$$

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

Oczywiście:
Jeśli $S \stackrel{*}{\Rightarrow}_l wA\alpha \Rightarrow_l w\beta\alpha \stackrel{*}{\Rightarrow}_l wx$
to FIRST(x) należą do FIRST$(\beta\ \text{FOLLOW}(A))$.

Gdyby $S \stackrel{*}{\Rightarrow}_l wA\alpha \Rightarrow_l w\gamma\alpha \stackrel{*}{\Rightarrow}_l wy$ oraz $\text{FIRST}(x) = \text{FIRST}(y) = c$, to mielibyśmy $c \in \text{FIRST}(\beta\ \text{FOLLOW}(A)) \cap \text{FIRST}(\gamma\ \text{FOLLOW}(A))$.

Ale jest to niemożliwe.

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

zakładamy, że

$$c \in \text{FIRST}(\beta\ \text{FOLLOW}(A)) \cap \text{FIRST}(\gamma\ \text{FOLLOW}(A))$$

Przypadek 1: $c \in \text{FIRST}(\beta) \cap \text{FIRST}(\gamma)$
wtedy $S \stackrel{*}{\Rightarrow}_l vA\alpha \Rightarrow_l v\beta\alpha \stackrel{*}{\Rightarrow}_l vcx$ dla pewnego x,
$S \stackrel{*}{\Rightarrow}_l vA\alpha \Rightarrow_l v\gamma\alpha \stackrel{*}{\Rightarrow}_l vcy$ dla pewnego y,
$\text{FIRST}(cy) = c = \text{FIRST}(cx)$. Równocześnie $\beta \neq \gamma$, więc G nie jest LL(1)!

Dowód: $(\Rightarrow)$

Przypadek 2: $c \notin \text{FIRST}(\beta), c \notin \text{FIRST}(\gamma)$, ale $c \in \text{FOLLOW}(A)$, $\varepsilon \in \text{FIRST}(\beta), \varepsilon \in \text{FIRST}(\gamma)$;
wtedy: $S \stackrel{*}{\Rightarrow}_l vA\alpha \Rightarrow_l v\beta\alpha \stackrel{*}{\Rightarrow}_l v\alpha \stackrel{*}{\Rightarrow}_l vcx$.
Także $S \stackrel{*}{\Rightarrow}_l vA\alpha \Rightarrow_l v\gamma\alpha \stackrel{*}{\Rightarrow}_l v\alpha \stackrel{*}{\Rightarrow}_l vcx$.
Wbrew definicji LL(1).

Dowód: $(\Rightarrow)$

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

Tabele parsowania dla LL(1)

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

Gramatyka spoza LL(1)

Gramatyka G:
$S \to \varepsilon | abA, \quad A \to Saa | b$.

G nie jest LL(1):
$S \Rightarrow abA \Rightarrow abSaa \Rightarrow ababAaa \stackrel{*}{\Rightarrow}_l ab\underline{a}...$
$S \Rightarrow abA \Rightarrow abSaa \Rightarrow ab\underline{a}a$

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

FIRST_k

$\text{FIRST}_k(\alpha)$ zdefiniowane tak jak $\text{FIRST}(\alpha)$:

- $\exists \alpha \stackrel{*}{\Rightarrow}_l w$, w składa się z terminali, $|w| \geq k$ i x jest prefixem w długości k, lub:
- $\exists \alpha \stackrel{*}{\Rightarrow}_l w$, w składa się z terminali, $|w| < k$ i $x = w$.

Gramatyka LL(k)

gramatyka jest LL(k) $\Leftrightarrow$
z

$$S \stackrel{*}{\Rightarrow}_l wA\alpha \Rightarrow_l w\beta\alpha \stackrel{*}{\Rightarrow}_l wx \qquad S \stackrel{*}{\Rightarrow}_l wA\alpha \Rightarrow_l w\gamma\alpha \stackrel{*}{\Rightarrow}_l wy$$

i

$$\text{FIRST}_k(x) = \text{FIRST}_k(y)$$

wynika, że $\beta = \gamma$.

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

Gramatyka nie-LL(k) dla dowolnego k

gramatyka
$S \to A|B, \quad A \to aAb|0, \quad B \to aAbb|1$
definiuje $\{a^n 0 b^n : n \geq 0\} \cup \{a^n 0 b^{2n} : n \geq 0\}$.

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 $S \Rightarrow B$).

Podstawowa własność

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

$$\text{FIRST}_k(\beta\alpha) \cap \text{FIRST}_k(\gamma\alpha) = \emptyset$$

Brak analogii do LL(1)

następująca własność nie jest równoważna LL(k)!

jeśli $A \to \beta$, $A \to \gamma$, $\beta \neq \gamma$, to
$\text{FIRST}_k(\beta\ \text{FOLLOW}_k(A)) \cap \text{FIRST}_k(\gamma\ \text{FOLLOW}_k(A)) = \emptyset$

przykład:
$S \to aAaa|bAba, \quad A \to b|\varepsilon$
widać z definicji, że gramatyka jest LL(2).
Ale: $\text{FOLLOW}_2(A) = \{aa, ba\}$,
$\text{FIRST}_2(b\ \text{FOLLOW}_2(A)) \cap \text{FIRST}_2(\varepsilon\ \text{FOLLOW}_2(A)) = \{ba\}$

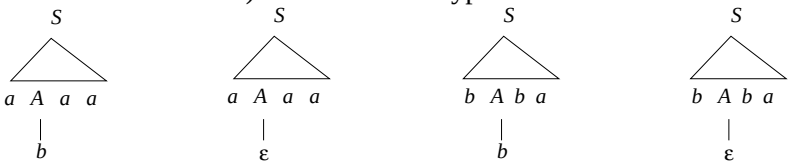
Konstrukcja parsera LL(k)

- G - gramatyka LL(k), $wx \in L_G$
- konstruujemy lewostronne wyprowadzenie dla wx, zakładamy, że mamy $S \overset{*}{\Rightarrow}_l w\alpha$, takie że
  α zaczyna się nieterminalem, $\alpha \overset{*}{\Rightarrow}_l x$
- z def: z w i k następnych znaków x można określić następny krok wyprowadzenia
- problem: w nie można przechować na stosie (tam jest α)

Konstrukcja parsera LL(k) - problemy

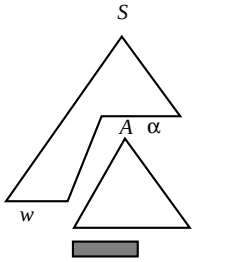
Informacje o w mogą być niezbędne. Przykład:
(1) $S \to aAaa$
(2) $S \to bAba$
(3) $A \to b$
(4) $A \to \varepsilon$.
input $abaa$.

pierwszy krok: $S \Rightarrow aAaa$,
aby określić następny krok nie wystarcza A i następne 2 znaki inputu, tj. ba.
niesprzeczna z tą informacją także $A \to \varepsilon$!
(choć $S \Rightarrow aAaa \Rightarrow abaa$). Oto możliwe wyprowadzenia:



Sterowanie parsingiem

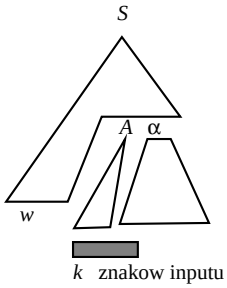
- po pierwszym kroku trzeba wiedzieć, że z A będzie można tylko uzyskać aa
- przy wyborze następnego kroku dla $S \stackrel{*}{\Rightarrow}_l A\alpha$:



1. pierwsze k znaków otrzymanych z A musi się zgadzać z inputem,

Idea konstrukcji parsera

- 2. kiedy otrzymujemy y z A i $|y| < k$, to y musi być uzupełniony przez



odpowiednie znaki wyprowadzalne z α,

Idea konstrukcji parsera

- „ciągi wyprowadzalne z α“ to tylko aa. Istotnie:
  1. produkcja $A \to \varepsilon$ nie pasuje do drugiego kroku:
     jako prefix długości k dostalibyśmy $\mathrm{FIRST}_2(\varepsilon\ aa) = aa$. Następne 2 znaki inputu to ba!
  2. produkcja $A \to b$ pasuje:
     $\mathrm{FIRST}_2(b\ aa) = ba$.

Idea konstrukcji parsera

- mamy: $S \stackrel{*}{\Rightarrow}_l wA\alpha$, prefix w został przeczytany.
- $A\alpha$ na stosie, ale z dodatkowymi informacjami.
- każdy nieterminal B jest zastępowany przez LL(k)-tabelę $T_{B,L}$, gdzie L jest zbiorem ciągów długości k, które mogą zostać wyprowadzone **za** ciągiem wyprowadzonym z B.

Notacja

$L_1 \oplus_k L_2 =$ to

$$\{w : \exists x \in L_1 \exists y \in L_2 \quad (w = xy \text{ oraz } |xy| \leq k) \text{ lub } (w = \text{FIRST}_k(xy))\}$$

Tabele LL(k)

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

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

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, x_i -ciąg terminali. Wtedy:

$$Y_i = \mathrm{FIRST}_k(x_i B_{i+1} \ldots B_m x_m \oplus L)$$

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

Konstrukcja tabel LL(k)

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

jeśli $T \in I$ 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 I.

Parser LL(k)

Na początku $T_{S,\{\varepsilon\}}$ na stosie.
Krok parsera:

- $M[T_{A,L}, u]$ zdefiniowany jak następuje:
  niech $T_{A,L}(u) = (A \to x_0 B_1 x_1 B_2 x_2 \ldots B_m x_m, \langle Y_1, \ldots, Y_m \rangle)$,
  wtedy usuń $T_{A,L}$ ze stosu, zapisz $x_0 T_{B_1,Y_1}, x_1, \ldots, T_{B_m,Y_m}, x_m$ 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, wpw.

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.