

Зміст

10 Синтаксичний аналіз на $LL(1)$-граматиках	1
10.1 Синтаксичний аналіз на основі $LL(1)$ -граматик	1
10.1.1 Приклад	2
10.1.2 Алгоритм	2
10.1.3 Майже $LL(1)$ -граматики	3
10.2 Контрольні запитання	4

10 Синтаксичний аналіз на $LL(1)$ -граматиках

10.1 Синтаксичний аналіз на основі $LL(1)$ -граматик

Згідно визначення $LL(1)$ -граматики, граматика G буде $LL(1)$ граматикою тоді і тільки тоді, коли для кожного A -правила вигляду $A \mapsto \omega_1 \mid \omega_2 \mid \dots \mid \omega_p$ виконуються умови

- $\text{First}_1(\omega_i) \cap \text{First}_1(\omega_j) = \emptyset$ для довільних $i \neq j$.
- якщо $\omega_i \Rightarrow^* \varepsilon$ для якогось i , то $\text{First}_1(\omega_j) \cap \text{Follow}_1(A) = \emptyset$ для усіх $j \neq i$.

Таблиця $M(a, b)$ де $a \in (N \cup \Sigma \cup \{\varepsilon\})$, $b \in (\Sigma \cup \{\varepsilon\})$ керування $LL(1)$ -синтаксичним аналізатором визначається наступним чином:

1. $M(A, b)$ — номер правила вигляду $A \mapsto \omega_i$ такого, що $\{b\} = \text{First}_1(\omega_i \cdot \text{Follow}_1(A))$.
2. $M(a, a)$ містить інструкцію **pop** для аналізатора яка позначає необхідність перенести символ a з пам'яті аналізатору у поле результату.
3. $M(\varepsilon, \varepsilon)$ містить інструкцію **accept** для аналізатора яка позначає що опрацьоване слово необхідно допустити (повернути **true** абощо).
4. В інших випадках $M(a, b)$ невизначено, чи радше містить інструкцію **reject** для аналізатора яка позначає що опрацьоване слово необхідно недопустити (повернути **false** абощо).

10.1.1 Приклад

Розглянемо вже добре відому нам граматичку зі схемою

$$\begin{aligned} S &\mapsto BA, \\ A &\mapsto +BA \mid \varepsilon, \\ B &\mapsto DC, \\ C &\mapsto \times DC \mid \varepsilon, \\ D &\mapsto (S) \mid a. \end{aligned}$$

і пронумеруємо її правила таким чином:

$$S \mapsto BA, \tag{1}$$

$$A \mapsto +BA, \tag{2}$$

$$A \mapsto \varepsilon, \tag{3}$$

$$B \mapsto DC, \tag{4}$$

$$C \mapsto \times DC, \tag{5}$$

$$C \mapsto \varepsilon, \tag{6}$$

$$D \mapsto (S), \tag{7}$$

$$D \mapsto a. \tag{8}$$

Нагадаємо що для цієї граматички $\text{First}_1(S) = \text{First}_1(B) = \text{First}_1(D) = \{ (, a), \text{First}_1(A) = \{ +, \varepsilon \}, \text{First}_1(A) = \{ \times, \varepsilon \}, \text{ а також } \text{Follow}_1(S) = \text{Follow}_1(A) = \{ \varepsilon,) \}, \text{Follow}_1(B) = \text{Follow}_1(C) = \{ +, \varepsilon,) \}, \text{Follow}_1(D) = \{ +, \times, \varepsilon,) \}.$

Знайдемо множини $\text{First}_1(\omega_i \cdot \text{Follow}_1(A))$ як $\text{First}_1(\omega_i) \oplus_1 \text{Follow}_1(A)$) використовуючи результати минулої лекції.

При побудові таблиці $M(a, b)$ керування $LL(1)$ -синтаксичним аналізатором достатньо лише побудувати першу її частину, тобто ту яка з $N \times (\Sigma \cup \{ \varepsilon \})$, оскільки решта таблиці визначається стандартно:

	a	$($	$)$	$+$	\times	ε
S	1	1				
A			3	2		3
B	4	4				
C			6	6	5	6
D	8	7				

10.1.2 Алгоритм

Побудуємо $LL(1)$ -синтаксичний аналізатор на основі таблиці керування $M(a, b)$:

1. Прочитаємо поточну лексему з вхідного файлу, у стек магазинного автомата занесемо аксіому S .
2. Загальний крок роботи:
 - Якщо на вершині стека знаходиться нетермінал A_i , то активізувати рядок таблиці, позначений A_i . Елемент $M(A_i, \langle \text{поточна лексема} \rangle)$ визначає номер правила, права частина якого заміняє A_i на вершині стека.
 - Якщо на вершині стека лексема $a_i = \langle \text{поточна лексема} \rangle$, то з вершини стека зняти a_i та прочитати нову поточну лексему.
 - Якщо стек порожній та досягли кінця вхідного файлу, то вхідна програма синтаксично вірна.
 - В інших випадках — синтаксична помилка.

10.1.3 Майже $LL(1)$ -граматики

У деяких випадках досить складно (а інколи й принципово неможливо побудувати $LL(1)$ -граматику для реальної мови програмування. При цьому $LL(1)$ -властивість задовольняється майже для всіх правил — лише декілька правил створюють конфлікт, але для цих правил задовольняється **сильна** $LL(2)$ -властивість. Тоді таблиця $M(a, b)$ визначається в такий спосіб:

- $M(A, b) = \langle \text{номер правила} \rangle$ вигляду $A_i \mapsto \omega_i$, такого, що $b \in \text{First}_1(\omega_i \cdot \text{Follow}_1(A))$
- $M(A, b) = \langle \text{ім'я допоміжної програми} \rangle$ за умови, що

$$b \in \text{First}_1(\omega_i \cdot \text{Follow}_1(A)) \cap b \in \text{First}_1(\omega_j \cdot \text{Follow}_1(A)), \quad i \neq j$$

Програма, яка виконує додатковий аналіз вхідного ланцюжка, повинна:

- прочитати додатково одну лексему;
- на основі двох вхідних лексем вибрати необхідне правило або сигналізувати про синтаксичну помилку;
- у випадку, коли правило вибрано, необхідно повернути додатково прочитану лексему у вхідний файл.

Звичайно, необхідно модифікувати алгоритм $LL(1)$ -синтаксичного аналізатора.

При цьому підпрограма аналізу конфліктної ситуації повинна додатково прочитати нову вхідну лексему, далі скориставшись контекстом з двох лексем, визначити номер правила, яке замість нетермінала на вершині стека та повернути додатково прочитану лексему у вхідний файл.

10.2 Контрольні запитання

1. Які дві умови повинна задовольняти граматика щоб бути $LL(1)$ -граматикою?
2. Що таке таблиця керування синтаксичного аналізатора на основі $LL(1)$ -граматики?
3. Який автомат і яка таблиця використовуються в алгоритмі роботи $LL(1)$ -синтаксичного аналізатора?
4. Опишіть алгоритм роботи $LL(1)$ -синтаксичного аналізатора.
5. Як необхідно модифікувати таблицю керування для сильної $LL(2)$ -граматики яка є майже $LL(1)$ -граматикою?
6. Як необхідно модифікувати алгоритм роботи синтаксичного аналізатора для сильної $LL(2)$ -граматики яка є майже $LL(1)$ -граматикою?