

Методы нисходящего синтаксического анализа

Применяемый класс грамматик

- Контекстно-свободные грамматики
- Левая часть правила – нетерминальный символ, правая часть – произвольная строка

$$a \rightarrow v, \text{ где } a \text{ из } N, v \text{ из } \mathcal{A}^*$$

- Порождаемые языки – контекстно-свободные языки
- Механизм распознавания – автоматы с магазинной памятью

Нисходящий рекурсивный алгоритм

- Пусть имеется входная лента, содержащая строку символов в алфавите \mathcal{L} .
- Строка заканчивается специальным символом $>$.
- $\mathcal{N} = \{L, E, F, T, P, Q\}$
- $\mathcal{L} = \{x, +, *, (,), >\}$
- Правила вывода:
 1. $L \rightarrow EQ$
 2. $E \rightarrow TF$
 3. $E \rightarrow T$
 4. $F \rightarrow +E$
 5. $F \rightarrow *T$
 6. $T \rightarrow x$
 7. $T \rightarrow (EP$
 8. $P \rightarrow)$
 9. $Q \rightarrow >$
- Исходный символ – L
- Построить алгоритм нисходящего разбора данной строки

Основные множества, управляющие синтаксическим разбором

- для каждого нетерминального символа построим множества терминальных символов, которые могут за ними следовать
 - $\text{fin}(E) = \{ \text{) , } > \}$
 - $\text{fin}(F) = \{ \text{) , } > \}$
 - $\text{fin}(T) = \{ \text{) , } > \}$
- Для каждого правила вывода множество символов, с которых может начинаться выводимая из него строка
 1. $\text{beg}(L \rightarrow E >) = \{ \mathbf{x}, (\}$
 2. $\text{beg}(E \rightarrow T F) = \{ \mathbf{x}, (\}$ $\text{dir}(E \rightarrow TF) = \text{first}(F) = \{ +, * \}$
 3. $\text{beg}(E \rightarrow T) = \{ \mathbf{x}, (\}$ $\text{dir}(E \rightarrow T) = \text{fin}(T) = \{ \text{) , } > \}$
 4. $\text{beg}(F \rightarrow + E) = \{ + \}$
 5. $\text{beg}(F \rightarrow * T) = \{ * \}$
 6. $\text{beg}(T \rightarrow \mathbf{x}) = \{ \mathbf{x} \}$
 7. $\text{beg}(T \rightarrow (E P) = \{ (\}$
 8. $\text{beg}(P \rightarrow)) = \{ \text{) } \}$

Последовательность грамматического разбора LL(1)

```

L
E Q
T F Q
a F Q
a + E Q
a + T F Q
a + b F Q
a + b * T Q
a + b * ( E P Q
a + b * ( T F P Q
a + b * ( c F P Q
a + b * ( c + E P Q
a + b * ( c + T F P Q
a + b * ( c + d F P Q
a + b * ( c + d * T P Q
a + b * ( c + d * e P Q
a + b * ( c + d * e ) Q
a + b * ( c + d * e ) >
  
```

```

(1) L -> E >
(2) E -> T F
(6) T -> x
(4) F -> + E
(2) E -> T F
(6) T -> x
(5) F -> * T
(7) T -> ( E P
(2) E -> T F
(6) T -> x
(4) F -> + E
(2) E -> T F
(6) T -> x
(5) F -> * T
(6) T -> x
(8) P -> )
(9) Q-> >
  
```

```

a | + b*(c+d*e)>
a | + b*(c+d*e)>
+ | b*(c+d*e)>
b | *(c+d*e)>
b | *(c+d*e)>
* | (c+d*e)>
( | c+d*e)>
c | +d*e)>
c | +d*e)>
+ | d*e)>
d | *e)>
d | *e)>
* | e)>
e | )>
) | >
> |
  
```

1. $\text{beg}(L \rightarrow E \>) = \{x, (\}$
2. $\text{beg}(E \rightarrow T F) = \{x, (\}$
3. $\text{beg}(E \rightarrow T) = \{x, (\}$

$\text{dir}(E \rightarrow T F) = \{+, *\}$
 $\text{dir}(E \rightarrow T) = \{), >\}$

4. $\text{beg}(F \rightarrow + E) = \{+\}$
5. $\text{beg}(F \rightarrow * T) = \{*\}$
6. $\text{beg}(T \rightarrow x) = \{x\}$
7. $\text{beg}(T \rightarrow (E P) = \{((\}$
8. $\text{beg}(P \rightarrow)) = \{) \}$

Ограничения на КС-грамматику, чтобы она была LL(1)

- Грамматика не должна содержать правил вида: $A \rightarrow Bv$, $B \rightarrow Cv$, $C \rightarrow Aw$, то есть лево рекурсивных цепочек вывода
- Для любых двух правил с одинаковой левой частью должно выполняться одно из двух следующих правил:
 1. $\text{beg}(A \rightarrow v) \& \text{beg}(A \rightarrow w) == \text{null}$ то есть правила должны различаться по первому выводимому терминальному символу
 2. Если это не так, $\text{dir}(A \rightarrow v) \& \text{dir}(A \rightarrow w) == \text{null}$, то есть правила должны различаться по второму терминальному символу, выводимому из строк v и w

• [Пример](#)

Дополнительное ограничение

- В грамматику должны входить только правила следующего вида:
 - $N \rightarrow u$ где u из V^*
 - $N \rightarrow tu$ где t из \mathcal{L} и u из V^*

Например:

Из правила

$T \rightarrow (E)$

следует сделать 2 правила:

$T \rightarrow (ER$

$R \rightarrow)$

где R – новый нетерминальный символ

Порядок работы

- В стек помещаем аксиому
- На входную ленту – строку, подлежащую распознаванию
- Выбираем самый левый нетерминал N в стеке и самый левый символ x на входной ленте и по этой паре принимаем решение:
 - если существует правило, $N \rightarrow u$ однозначно определяемое парой N, x , то:
 - если $u = xv$, то заменяем в стеке N на xv и удаляем символ x из входной ленты
 - если u начинается с нетерминала или u – пустая строка, то только заменяем N на u
 - Если пара N, x не однозначно определяют правило замены, то привлекаем След символ
 - Если не находится правила по паре N, x то ошибка
- Критерий правильности распознавания:
 - входная лента пуста
 - в стеке – копия входной строки