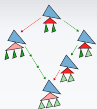




Лабораторная работа 4

Эмпирическая часть: «паническая атака»

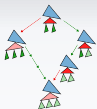
Добиться того, чтобы сообщение о синтаксической ошибке в программе на любом языке программирования, использованном вами при выполнении лабораторных работ по ТФЯ, указывало не на ту строчку, в которой ошибка действительно есть. Описать в отчёте свои предположения, почему так произошло.



Лабораторная работа 4

Практическая часть: автолекс (чётный вариант)

- 1 Найти в данной грамматике все заведомо регулярные нетерминалы.
- 2 Все терминалы в правых частях правил, определяющих прочие нетерминалы, обернуть в охранные нетерминалы и также объявить регулярными (с конечным языком).
- 3 Для каждого регулярного нетерминала T найти $\text{PRECEDE}(T)$ и $\text{FOLLOW}(T)$. Если ни один терминал из множества $\text{PRECEDE}(T) \cup \text{FOLLOW}(T)$ не входит в язык нетерминала T , объявить нетерминал T токеном, иначе объявить о конфликте языков.
- 4 Для каждого токена породить регулярное выражение, которое его распознаёт.



Лабораторная работа 4

Практическая часть: LR(0)-подгонка методом Микунаса–Ланкастера–Шнейдера (нечётный вариант)

- 1 Устранить в грамматике ε -правила.
- 2 Сгенерировать автомат LR(0)-переходов.
- 3 Если обнаружился shift–reduce или reduce–reduce конфликт в правилах $A_1 \rightarrow \gamma$, $A_2 \rightarrow \delta$, тогда во всех возможных правилах переписывания, содержащих A_1 и A_2 в правых частях, произвести для A_1 и A_2 присоединение правого контекста. Вернуться к пункту 2.
- 4 Если в предыдущем пункте в некотором правиле не удалось сделать присоединение правого контекста к A_1 либо A_2 , тогда произвести в нём уточнение правого контекста. Затем перейти к пункту 1.
- 5 Если автомат LR(0)-переходов не содержит конфликтов, либо уточнение правого контекста происходило более, чем N раз, сообщить об успехе / неудаче подгонки и вывести полученную грамматику.



Синтаксис входных данных

Чёрным обозначены элементы метаязыка, красным — элементы языка входных данных. Чтение данных осуществляется из файла. Расстановка пробелов произвольна, могут встречаться табуляции, новая строка может начинаться с `\n` или с `\r\n`. Начальный нетерминал — `[S]` (что существенно только для нечётного варианта, у чётного может его не быть).

У нечётного варианта первой строкой (перед грамматикой) во входном потоке должна стоять директива

\$EXTRACTIONS = N, где N — положительное число.

Синтаксис входных данных CFG:

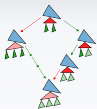
<code><grammar></code>	<code>::=</code>	<code><rule>⁺</code>
<code><rule></code>	<code>::=</code>	<code><nterm> ::= <term>⁺</code>
<code><term></code>	<code>::=</code>	<code><nterm> [a-z] [0-9] _ * + = () \$; :</code>
<code><nterm></code>	<code>::=</code>	<code>[[A-z]⁺]</code>



Множества LAST и PRECEDE

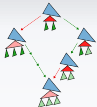
Язык таких множеств — язык терминалов плюс (в случае PRECEDE) специальный символ $\hat{}$ начала строки.

Определения и алгоритмы для нахождения $LAST(A)$ и $PRECEDE(A)$ полностью аналогичны таковым для $FIRST_1$ и $FOLLOW_1$, с той разницей, что ищется последний, а не первый, порождающийся A терминал, и возможные предшествующие A (а не следующие за A) терминалы языка.



Поиск заведомо регулярных нетерминалов

- Стоит учесть, что теперь в грамматике могут быть цепные правила (вида $A \rightarrow B$). Однако синтаксис грамматики гарантирует, что ε -правил в ней нет.
- Кроме подмножества регулярных нетерминалов, описываемых праволинейными правилами, могут появиться ещё подмножества, описываемые левوليнейными правилами. Чтобы преобразовать их в регулярные выражения по л.р. номер 2, придётся переделать в праволинейную форму.



Присоединение правого контекста

- Пусть нужно присоединить правые контексты к нетерминалу A . Для всех правил вида $C \rightarrow \gamma_1 A t \gamma_2$, где t — терминал, порождаем нетерминал $[At]$ и заменяем им часть At данного правила.
- Для всех правил вида $A \rightarrow \delta$ добавляем правило $[At] \rightarrow \delta t$.
- Данное преобразование не может быть применено к правилу вида $C \rightarrow \gamma_1 A B \gamma_2$. Поэтому, если нужно присоединять контекст в таком правиле, необходимо воспользоваться алгоритмом уточнения правого контекста.



Уточнение правого контекста

- Пусть нужно уточнить правый контекст у A по правилу $C \rightarrow \gamma_1 A B \gamma_2$. По условию, ε -правил нет. Поэтому $\text{FIRST}(B)$ не содержит ε .
- Для каждого элемента $c \in \text{FIRST}(B)$ строим нетерминал $[c/B]$ и правило $C \rightarrow \gamma_1 A c [c/B] \gamma_2$.
- Для всех правил вида $B \rightarrow c \delta$ строим правила вида $[c/B] \rightarrow \delta$.
- Для всех правил вида $B \rightarrow D \delta$ таких, что $c \in \text{FIRST}(D)$, строим правила вида $[c/B] \rightarrow [c/D] \delta$. Рекурсивно замыкаем процедуру (до неподвижной точки).
- В полученной грамматике могут появиться ε -правила (кодировку для ε можно выбрать произвольно). Поэтому их придётся в дальнейшем устранить.



Доп. баллы +1

- Чётные — случаи, когда язык нетерминала получается итеративным приписыванием левостороннего регулярного языка к правостороннему и наоборот, также должны обрабатываться как регулярные. Например, в следующей грамматике:
 $[S] \rightarrow [A], [A] \rightarrow [B], [A] \rightarrow a[A], [B] \rightarrow [B]b, [B] \rightarrow d$
нетерминал $[S]$ должен также распознаваться как регулярный.
- Нечётные — правый контекст можно доставать из $\text{FOLLOW}(A)$. Т.е. если нужно уточнить контекст A по правилу $C \rightarrow \Phi A$, тогда ищем все правила $C' \rightarrow \Psi_1 C \Psi_2$, которые порождают C , получаем правила $C' \rightarrow \Psi_1 \Phi A \Psi_2$ и действуем с ними так же, как при обычном уточнении правого контекста.