

LR -методы анализа

$LR(0)$

В. С. Полозов

Кафедра системного программирования СПбГУ



Теория автоматов и формальных языков

Идея

Задача - найти основу в сентенциальной форме наиболее эффективно для наибольшего числа грамматик.

- Для поиска подстроки эффективно использовать КА. Но очевидный автомат ищет самую левую сворачиваемую подстроку - что может не быть основой.
- Идея: запустим автомат от стартового правила и в любой позиции будем рассматривать только те правые части, которые выводятся из стартового (как и в Эрли)
- Такой автомат остановится на правом конце основы и принимающее состояние будет соответствовать правилу свёртки.

Грамматика для примера

На практике, размер таких конечных автоматов достаточно велик, поэтому будем рассматривать упрощённую грамматику:

$$S_s \rightarrow E \#$$

$$E \rightarrow E - T$$

$$E \rightarrow T$$

$$T \rightarrow n$$

$$T \rightarrow (E)$$

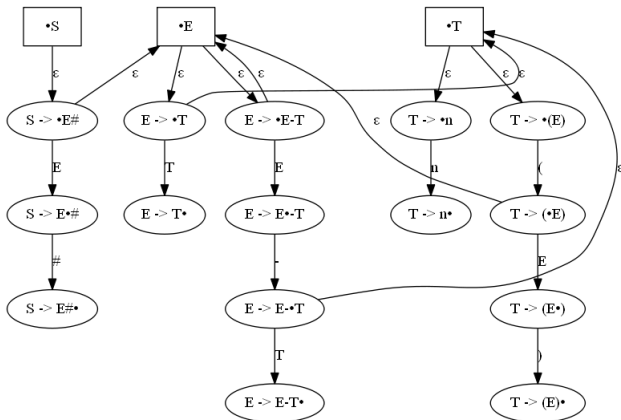
где $\#$ - конец ввода, S_s - стартовый символ.

Построение автомата

Построим *нисходящий ограничивающий конечный автомат распознавания основы*, недетерминированный:

- Множество состояний соединенных переходами
- Переходы помечены символом или ε
- Состояние соответствует *ситуации* ($A \rightarrow \alpha \bullet \beta$)
- Другой тип состояний - *станции* вида $\bullet X$. Только ε -переходы

Построим НКА для возможных ситуаций:



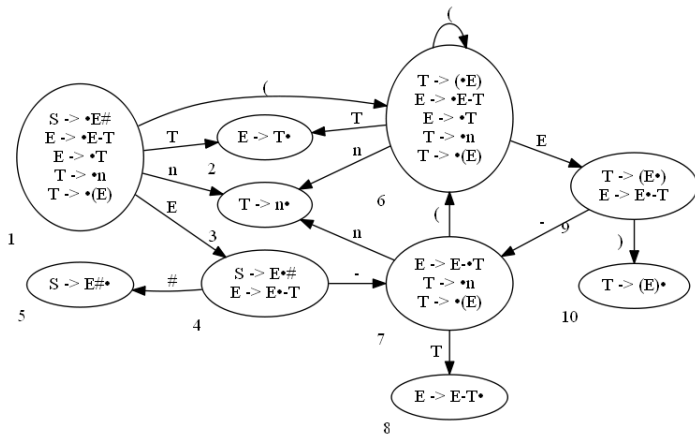
И рассмотрим пример работы со входом: $E - n - n$

Пример работы автомата

Пример работы построенного НКА на входе: $E - n - n$



Построим детерминированный автомат на основе НКА:



Это - LR(0)-автомат

ДКА-пример

Теперь можно ввести следующее обозначение для сентенциальной формы с состояниями:

$$\textcircled{1} E \textcircled{4} - \textcircled{7} n \textcircled{3} \quad - n$$

Что соответствует пути $\textcircled{1} \textcircled{4} \textcircled{7} \textcircled{3}$ в построенном ДКА. Состояние $\textcircled{3}$ - принимающее, соответствует редукции $T \rightarrow n$. После редукции получаем новую сентенциальную форму:

$$\textcircled{1} E \textcircled{4} - \textcircled{7} T \textcircled{8} \quad - n$$

Из чего делаем два вывода:

- Новая свёртка $E \rightarrow E - T$ из состояния $\textcircled{8}$
- Пересчитывать все состояния не надо

Такой тип редукции назовём *GOTO*

Таблица LR(0) анализатора

Представим построенный автомат в виде таблицы:

	n	-	()	#	E	T	reduced by
1	s3	e	s6	e	e	s4	s2	
2								$E \rightarrow T$
3								$T \rightarrow n$
4	e	s7	e	e	s5			
5								$S \rightarrow E\#$
6	s3	e	s6	e	e	s9	s2	
7	s3	e	s6	e	e		s8	
8								$E \rightarrow E - T$
9	e	s7	e	s10	e			
10								$T \rightarrow (E)$

Таблица LR(0) автомата

В таблице:

- $s\langle state \rangle$ - сдвинуться и перейти в состояние $\langle state \rangle$
- e - ошибка.
- пустое место - невозможное состояние, в независимости от входа.

Замечания:

- Свёртка, как правило, ассоциирована не с правилом грамматики, а с процедурой обработки.
- Таблица сильно разрежена и избыточна. По некоторым исследованиям можно сократить до 15

LR(0)-грамматики

Итак, нам удалось построить анализатор работающий за линейное время распознающий любую грамматику:

- По любой грамматике можно построить НКА
- По НКА можно построить ДКА
- ДКА работает за линейное время.

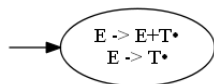
LR(0)-конфликты

Возможные конфликты:



сдвиг/свёртка (по +)

(shift/reduce)



свёртка/свёртка

(reduce/reduce)

LR(0)-грамматики

Определение

Состояние $LR(0)$ -автомата, содержащее конфликт, называется *неадекватным*.

Определение

Грамматика, по которой строится детерминированный $LR(0)$ -автомат, не содержащий неадекватных состояний, называется *$LR(0)$ -грамматикой*.

Вопросы