LR-методы анализа
$LR(0)$

В. С. Полозов

Кафедра системого 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:

$$
\begin{aligned}
S_s &\rightarrow E\ \# \\
E &\rightarrow E - T \\
E &\rightarrow T \\
T &\rightarrow n \\
T &\rightarrow (\ E\ )
\end{aligned}
$$

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

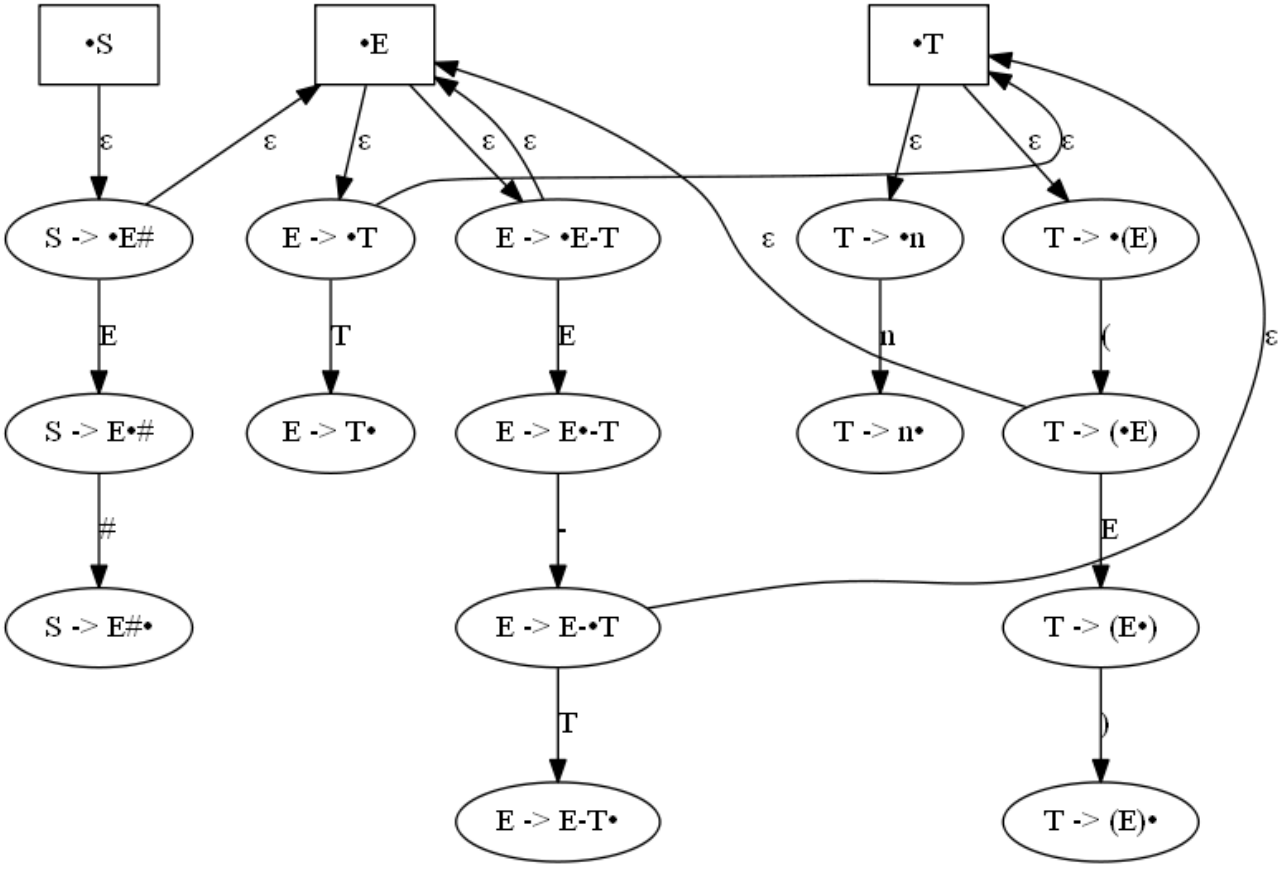
Построение автомата

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

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

НКА

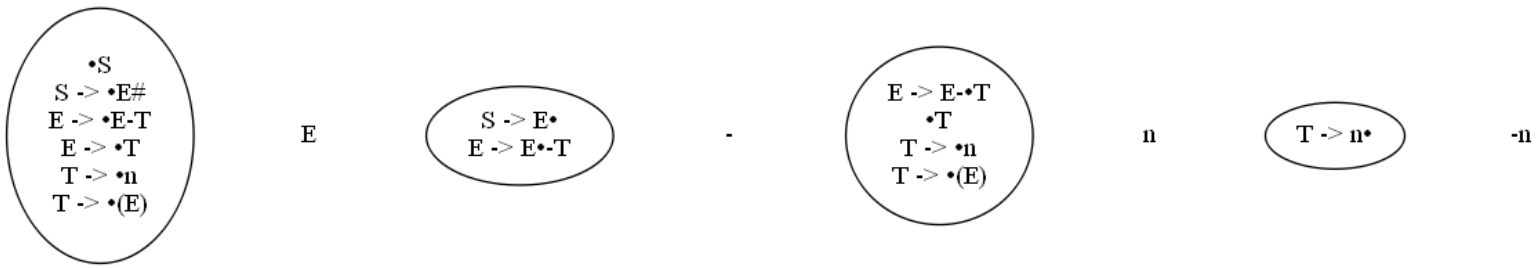
Построим НКА для возможных ситуаций:



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

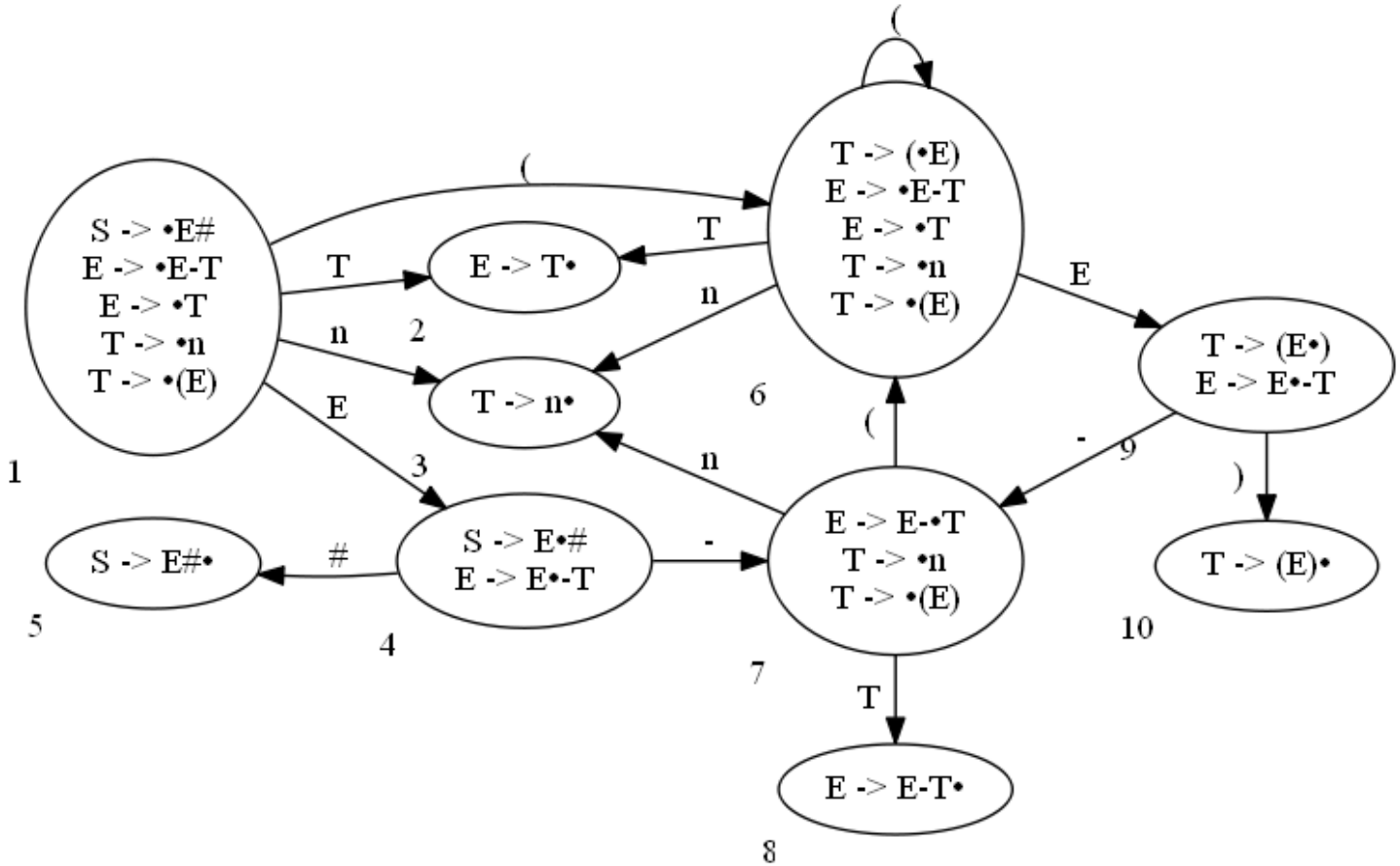
Пример работы автомата

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



ДКА

Построим детерминированный автомат на основе НКА:



Это - *LR(0)-автомат*

ДКА-пример

Теперь можно ввести следующее обозначение для сентенциальной формы с состояниями:

$$①\ \mathrm{E}\ ④\ \text{-}\ ⑦\ \mathrm{n}\ ③ \qquad \text{- n}$$

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

$$①\ \mathrm{E}\ ④\ \text{-}\ ⑦\ \mathrm{T}\ ⑧ \qquad \text{- n}$$

Из чего делаем два вывода:

- Новая свёртка $E \to E - T$ из состояния 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								$E \to T$
3								$T \to n$
4	e	s7	e	e	s5			
5								$S \to E\#$
6	s3	e	s6	e	e	s9	s2	
7	s3	e	s6	e	e		s8	
8								$E \to E - T$
9	e	s7	e	s10	e			
10								$T \to (E)$

Таблица LR(0) автомата

В таблице:

- s<state> - сдвинуться и перейти в состояние <state>
- 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

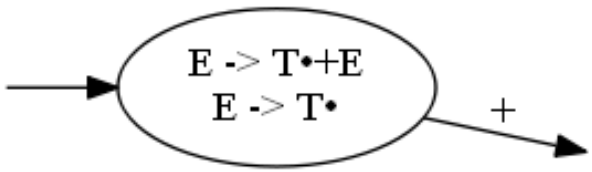
LR(0)-грамматики

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.

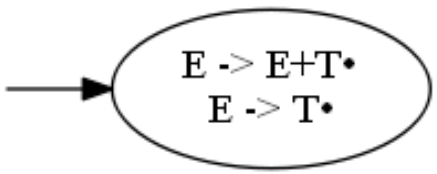
LR(0)-конфликты

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