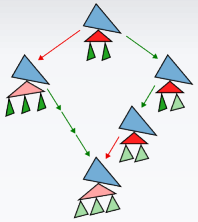
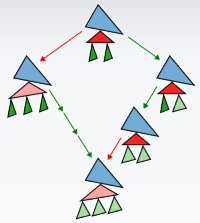
Восходящий разбор

Теория формальных языков
2021 г.



Детерминированные КС-языки

Язык L обладает префикс-свойством (prefix-free), если $\forall w(w \in L \Rightarrow \forall v(v \neq \varepsilon \Rightarrow wv \notin L))$.

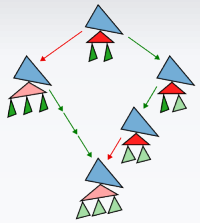


Детерминированные КС-языки

Язык L обладает префикс-свойством (prefix-free), если $\forall w(w \in L \Rightarrow \forall v(v \neq \varepsilon \Rightarrow wv \notin L))$.

Детерминированные языки с префикс-свойством — языки, распознаваемые DPDA с допуском по пустому стеку.

Рассмотрим язык a^+. Предположим, он распознаётся DPDA с допуском по пустому стеку. Тогда на элементе a стек уже обязательно пуст. А значит, работа DPDA не может быть продолжена, и элемент aa не может быть им распознан.

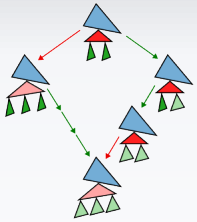


Детерминированные КС-языки

Язык L обладает префикс-свойством (prefix-free), если $\forall w(w \in L \Rightarrow \forall v(v \neq \varepsilon \Rightarrow wv \notin L))$.

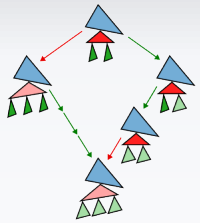
Детерминированные языки с префикс-свойством — языки, распознаваемые DPDA с допуском по пустому стеку.

Рассмотрим язык L, $w_1, w_1w_2 \in L$, $w_2 \neq \varepsilon$.
Предположим, он распознаётся DPDA с допуском по пустому стеку. Тогда на элементе w_1 стек уже обязательно пуст. А значит, работа DPDA не может быть продолжена, и элемент w_1w_2 не может быть им распознан.



Эндмаркеры

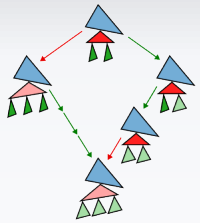
Рассмотрим язык $a^+\$$ (алфавит терминалов $\Sigma = \{a, \$\}$). В этом языке ни одно слово не является префиксом другого.



Эндмаркеры

Рассмотрим язык $\{w\$ \mid w \in \mathrm{L}\}$ (алфавит терминалов $\Sigma = \Sigma_{\mathrm{L}} \cup \{\$\}$, $\$ \notin \Sigma_{\mathrm{L}}$). Независимо от L, в этом языке ни одно слово не является префиксом другого.

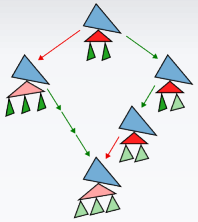
- Хорошие новости: любой детерминированный КС-язык легко преобразовать в язык, распознаваемый DPDA с допуском по пустому стеку.



Эндмаркеры

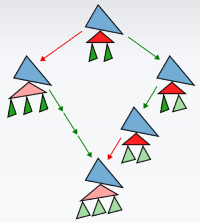
Рассмотрим язык (алфавит терминалов). этом языке ни одно слово не является префиксом другого.

- Хорошие новости: любой детерминированный КС-язык легко преобразовать в язык, распознаваемый DPDA с допуском по пустому стеку.
- Плохие новости: существенно неоднозначные контекстно-свободные языки с префикс-свойством. Стандартный пример: $\{a^n b^n c^m d\} \cup \{a^m b^n c^n d\}$.



Языки нередуцируемых префиксов

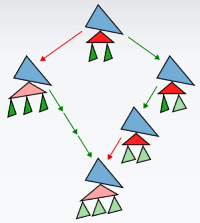
Определим понятие свёртки — перехода справа налево в правиле переписывания $A \to \alpha$. Что можно сказать о всех возможных префиксах сентенциальных форм, порождаемых грамматикой G, к которым нельзя применить ни одну свёртку?



Языки нередуцируемых префиксов

Определим понятие свёртки — перехода справа налево в правиле переписывания $A \to \alpha$. Что можно сказать о всех возможных префиксах сентенциальных форм, порождаемых грамматикой G, к которым нельзя применить ни одну свёртку?

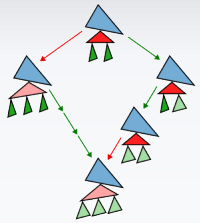
Такие с.ф. образуют регулярный язык. Идея обоснования: в распознающем их PDA из стека ничего не читается, т.е. PDA учитывает только символы сент. формы и свои состояния.



Автомат нередуцируемых префиксов

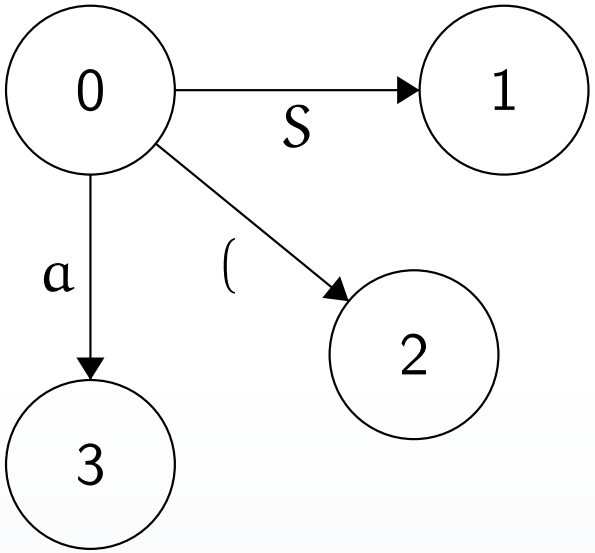
Описание конструкции

- Отмеченная позиция в правиле: $\bullet$. В правиле с правой частью $\xi_1 \dots \xi_n$ есть $n + 1$ таких позиций.
- Правило $A \to \alpha \bullet B\beta$ и правило $B \to \bullet\gamma$ — одно и то же множество переходов по символу, не приводящих к редукции $\Rightarrow$ в одном состоянии.
- При чтении элемента правой части сдвигаем $\bullet$ вправо на позицию.

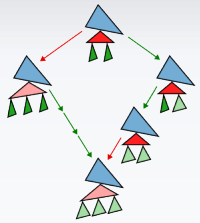


Автомат нередуцируемых префиксов

$$S' \to S \quad S \to a \quad S \to (S)$$

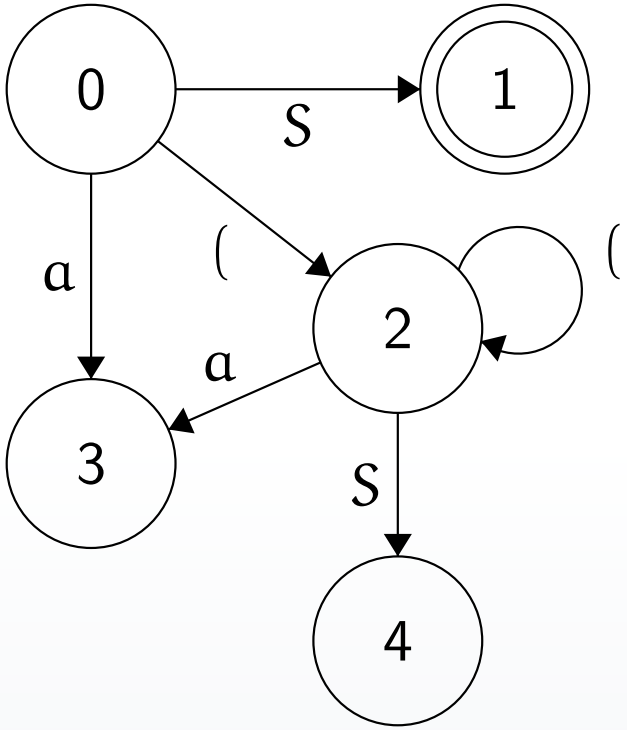


0	$S' \to \bullet S$
	$S \to \bullet(S)$
	$S \to \bullet a$
1	$S' \to S\bullet$
2	$S \to (\bullet S)$
3	$S \to a\bullet$

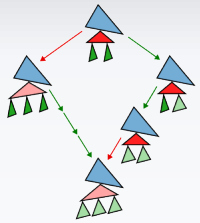


Автомат нередуцируемых префиксов

$$S' \to S \quad S \to a \quad S \to (S)$$

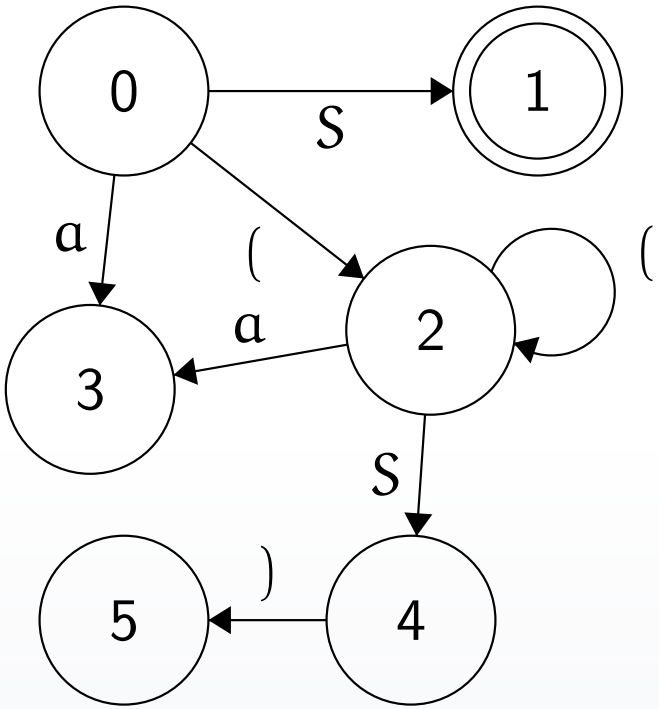


0	$S' \to \bullet S$
	$S \to \bullet(S)$
	$S \to \bullet a$
1	$S' \to S\bullet$
2	$S \to (\bullet S)$
	$S \to \bullet(S)$
	$S \to \bullet a$
3	$S \to a\bullet$
4	$S \to (S\bullet)$

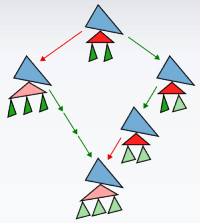


Автомат нередуцируемых префиксов

$$S' \to S \quad S \to a \quad S \to (S)$$



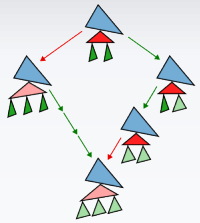
0	$S' \to \bullet S$
	$S \to \bullet(S)$
	$S \to \bullet a$
1	$S' \to S\bullet$
2	$S \to (\bullet S)$
	$S \to \bullet(S)$
	$S \to \bullet a$
3	$S \to a\bullet$
4	$S \to (S\bullet)$
5	$S \to (S)\bullet$



Типы состояний автомата

1. Финальное (свёртка в S').
2. Не финальное, но свёртка.
3. Сдвиг по символу сентенциальной формы.

Что хранить в стеке PDA, построенного по такому автомату?

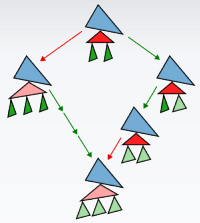


Типы состояний автомата

1. Финальное (свёртка в S').
2. Не финальное, но свёртка.
3. Сдвиг по символу сентенциальной формы.

Что хранить в стеке PDA, построенного по такому автомату?

- Хранить сами сентенциальные формы плохо — проблема с извлечением нескольких подряд символов.

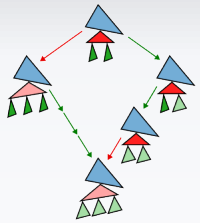


Типы состояний автомата

1. Финальное (свёртка в S').
2. Не финальное, но свёртка.
3. Сдвиг по символу сентенциальной формы.

Что хранить в стеке PDA, построенного по такому автомату?

- Хранить сами сентенциальные формы плохо — проблема с извлечением нескольких подряд символов.
- Логично хранить последовательности последних символов с.ф., которые могут привести к разным свёрткам, закодированными одним символом стека.

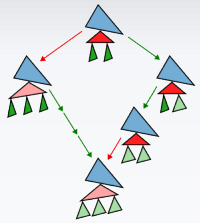


Типы состояний автомата

1. Финальное (свёртка в S').
2. Не финальное, но свёртка.
3. Сдвиг по символу сентенциальной формы.

Что хранить в стеке PDA, построенного по такому автомату?

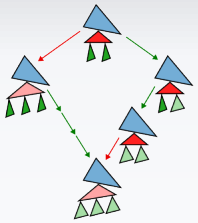
- Хранить сами сентенциальные формы плохо — проблема с извлечением нескольких подряд символов.
- Логично хранить последовательности последних символов с.ф., которые могут привести к разным свёрткам, закодированными одним символом стека.
- А это — в точности состояния автомата.



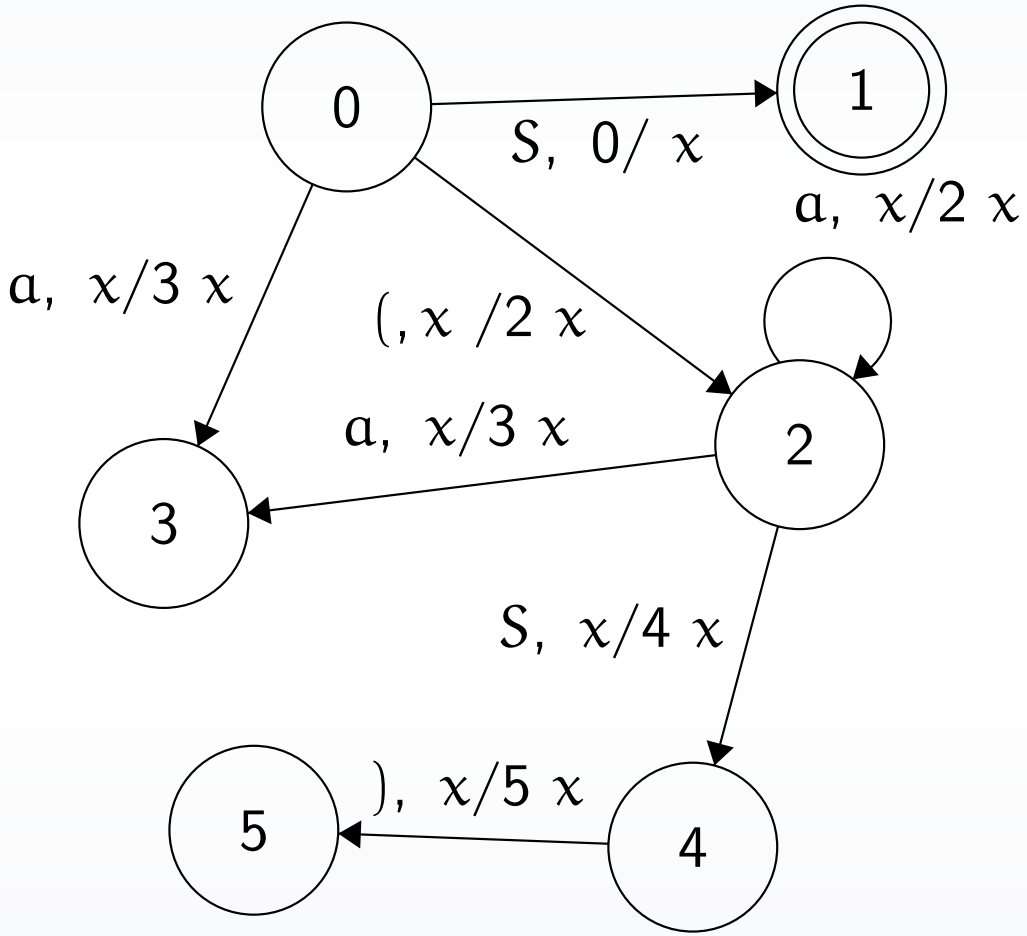
PDA по LR(0)-автомату

Общая конструкция

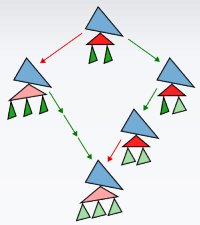
- При каждом сдвиге кладём в стек номер состояния, в которое приходим в конечном автомате.
- При каждой свёртке извлекаем из стека n символов, где n — длина правой части β правила $A \to \beta$, после чего переходим в состояние с номером $n + 1$-ого символа в стеке по символу A.



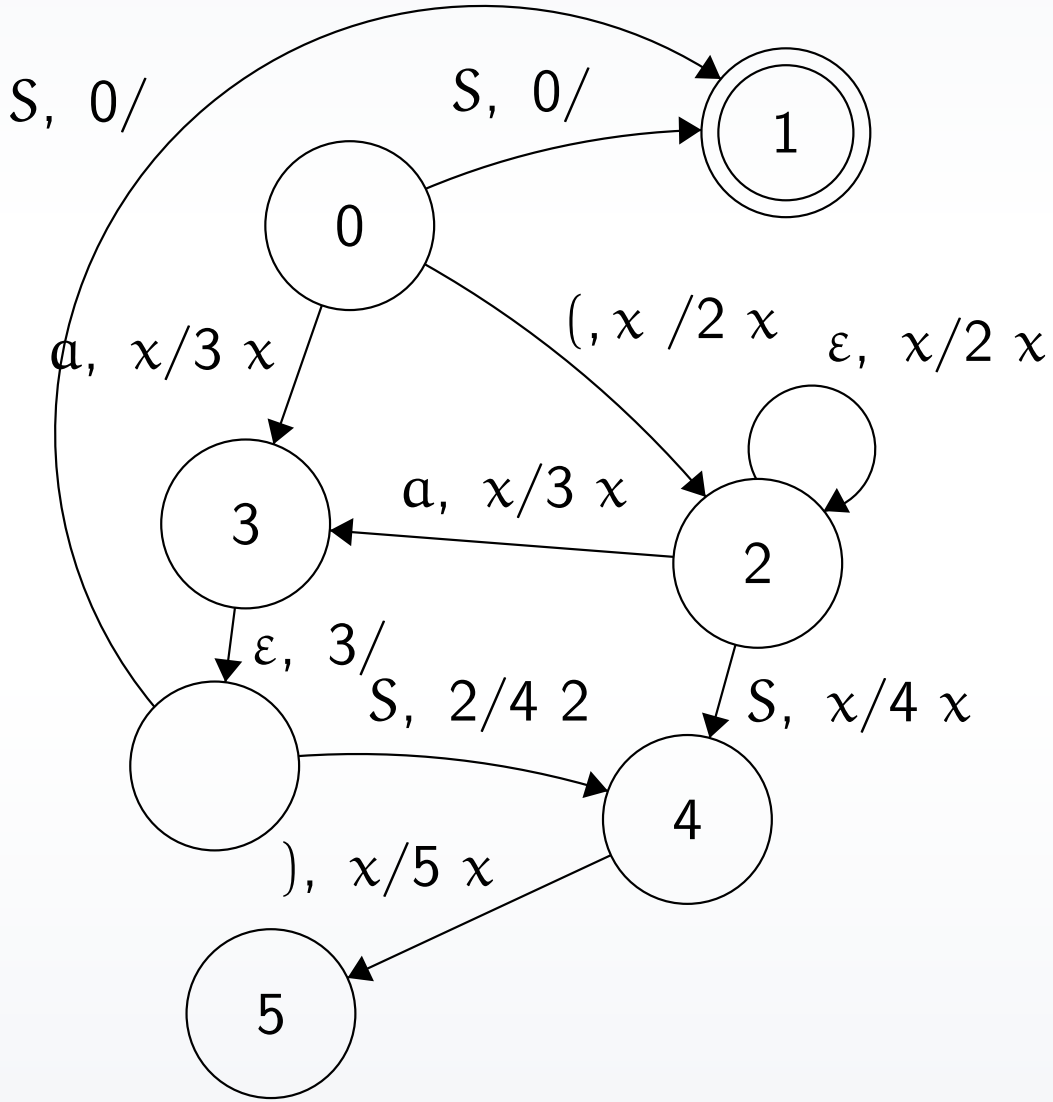
Пример построения PDA



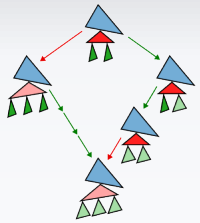
0	$S' \to \bullet S$ $S \to \bullet(S)$ $S \to \bullet a$	
1	$S' \to S\bullet$	$S \to S'$
2	$S \to (\bullet S)$ $S \to \bullet(S)$ $S \to \bullet a$	
3	$S \to a\bullet$	$a \to S$
4	$S \to (S\bullet)$	
5	$S \to (S)\bullet$	$(S) \to S$



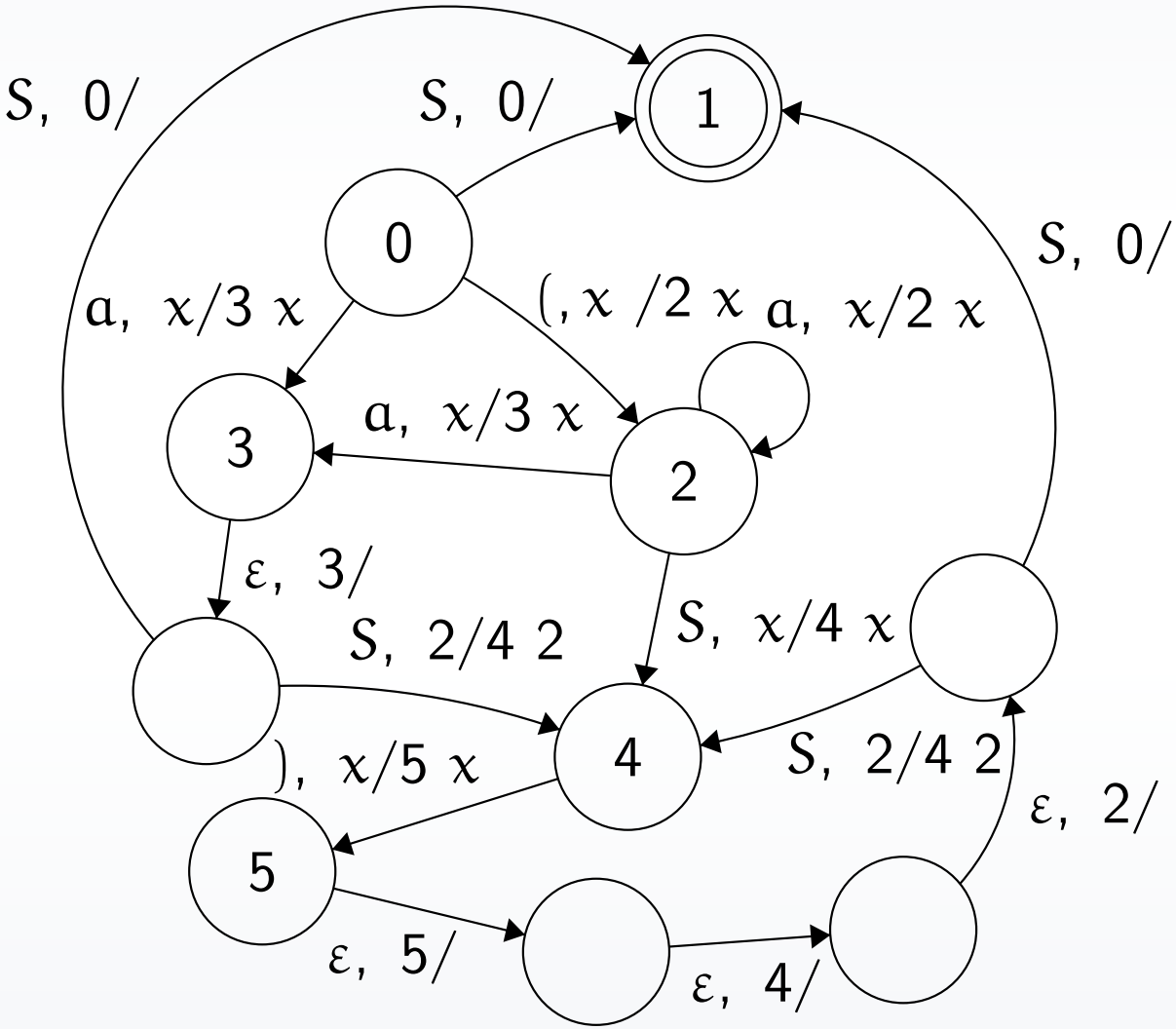
Пример построения PDA



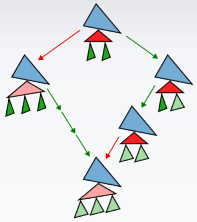
0	$S' \to \bullet S$ $S \to \bullet(S)$ $S \to \bullet a$	
1	$S' \to S\bullet$	$S \to S'$
2	$S \to (\bullet S)$ $S \to \bullet(S)$ $S \to \bullet a$	
3	$S \to a\bullet$	$a \to S$
4	$S \to (S\bullet)$	
5	$S \to (S)\bullet$	$(S) \to S$



Пример построения PDA

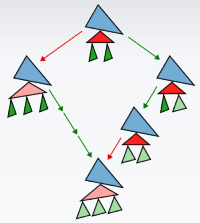


0	$S' \to \bullet S$	
	$S \to \bullet(S)$	
	$S \to \bullet a$	
1	$S' \to S\bullet$	$S \to S'$
2	$S \to (\bullet S)$	
	$S \to \bullet(S)$	
	$S \to \bullet a$	
3	$S \to a\bullet$	$a \to S$
4	$S \to (S\bullet)$	
5	$S \to (S)\bullet$	$(S) \to S$



PDA или DPDA?

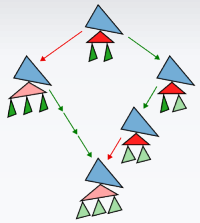
- Если есть ε-переходы, то нет никаких других.
- Если есть ε-переход, то он единственный из данного состояния.



PDA или DPDA?

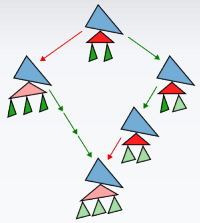
- ~~Если есть ε-переходы, то нет никаких других.~~ Если делается свёртка, то нельзя сделать сдвиг.
- ~~Если есть ε-переход, то он единственный из данного состояния.~~ Если делается свёртка одного типа, то нельзя сделать свёртку другого типа.
- Допуск — по пустому стеку $\Rightarrow$ DPDA для языков с префикс-свойством.

- DPDA с допуском по пустому стеку распознают те же языки, что и LR(0)-разбор.
- В конструкции LR(0)-автомата часто навязывается эндмаркер $\Rightarrow$ изначальная грамматика может описывать не LR(0)-язык!



Отказ от эндмаркера и SLR

- Используем **ту же конструкцию** автомата.
- Разрешим при возможности сделать свёртку вида $\beta \to A$ заглянуть в множество $\mathrm{FOLLOW}(A)$, чтобы понять, какую свёртку делать (и делать ли).

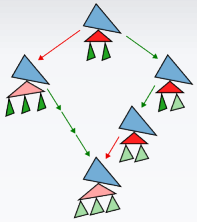


Отказ от эндмаркера и SLR

- Используем **ту же конструкцию** автомата.
- Разрешим при возможности сделать свёртку вида $\beta \to A$ заглянуть в множество $\mathrm{FOLLOW}(A)$, чтобы понять, какую свёртку делать (и делать ли).

$$
\begin{array}{lll}
S' \to \mathrm{E} & \mathrm{E} \to \mathrm{E} + \mathrm{T} & \mathrm{E} \to \mathrm{T} \\
\mathrm{E} \to \mathrm{V} = \mathrm{E} & \mathrm{T} \to (\mathrm{E}) & \mathrm{T} \to \mathrm{id} \\
 & \mathrm{V} \to \mathrm{id} &
\end{array}
$$

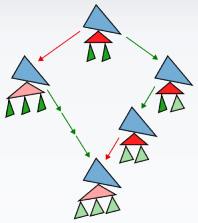
Здесь есть конфликт свёрток для S' (по $\mathrm{V} \to \mathrm{id}\bullet$ и $\mathrm{T} \to \mathrm{id}\bullet$), но $\mathrm{FOLLOW}_1(\mathrm{V}) \cap \mathrm{FOLLOW}_1(\mathrm{T}) = \emptyset \Rightarrow$ эта грамматика — $\mathrm{SLR}(1)$.



Коллапс линейных парсеров

Теорема

Для всякого языка из класса DCFL существует распознающая его $\mathrm{SLR}(1)$-грамматика.



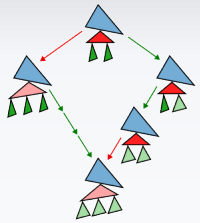
Теоретический коллапс линейных парсеров

Теорема

Для всякого языка из класса DCFL существует распознающая его $\mathrm{SLR}(1)$-грамматика.

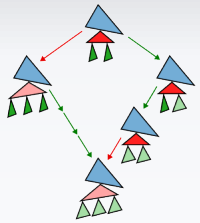
Следует из теоремы:

Для всякого языка из класса DCFL существует распознающая его $\mathrm{LR}(\mathrm{k})$-грамматика.



$\mathrm{LR}(k)$-распознаватели

Грамматика G — $\mathrm{LR}(k)$, тогда и только тогда, когда для всех пар сентенциальных форм xy, xy', порождаемых правосторонним разбором, где $y, y' \in \Sigma^+$, таких что xy допускает правую свёртку в префиксе y по правилу ξ_1, а xy' — свёртку где угодно по правилу ξ_2, и первые k символов y и y' совпадают, $\xi_1 = \xi_2$.

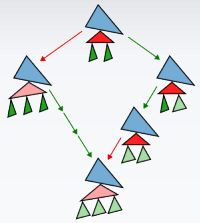


$\mathrm{LR}(k)$-распознаватели

Грамматика G — $\mathrm{LR}(k)$, тогда и только тогда, когда для всех пар сентенциальных форм xy, xy', порождаемых правосторонним разбором, где $y, y' \in \Sigma^+$, таких что xy допускает правую свёртку в префиксе y по правилу ξ_1, а xy' — свёртку где угодно по правилу ξ_2, и первые k символов y и y' совпадают, $\xi_1 = \xi_2$.

$$\begin{array}{lll} S' \to S & S \to L = R; & S \to R; \\ L \to \mathrm{id} & L \to *R & R \to L \end{array}$$

Поскольку $= \in \mathrm{FOLLOW}_1(R)$, возникает конфликт вида сдвиг–свёртка при попытке анализа с.ф. L. Но lookahead у L, порождённой посредством $S \to L = R$, и посредством $S \to R; \to L;$, будет разный.

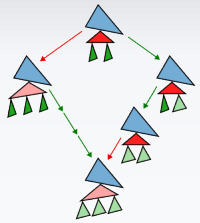


$\mathrm{LR}(k) \to \mathrm{LR}(1)$, Mickunas–Lancaster–Shneider

$$
\begin{array}{lll}
S' \to S & S \to Abb & S \to Bbc \\
A \to aA & A \to a & B \to aB \\
 & B \to a &
\end{array}
$$

Не $\mathrm{LR}(1)$, из-за свёрток $A \to a$, $B \to a$. Используем трансформацию присоединения правого контекста:

$$
\begin{array}{lll}
S' \to S & S \to [Ab]b & S \to [Bb]c \\
[Ab] \to a[Ab] & [Ab] \to ab & [Bb] \to a[Bb] \\
 & [Bb] \to ab &
\end{array}
$$



$\mathrm{LR}(k) \to \mathrm{LR}(1)$, Mickunas–Lancaster–Shneider

$$S' \to S \quad S \to bSS \quad S \to a$$
$$S \to aac$$

Не $\mathrm{LR}(1)$, конфликт свёртки на префиксе ba с контекстом a.

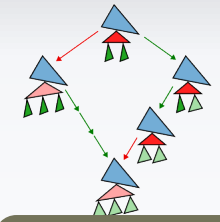
Используем трансформацию уточнения правого контекста:

$S \to bSa[a/S]$ $\quad S \to bSb[b/S]$ $\quad S \to a$ $\quad S \to aac$

$[a/S] \to \varepsilon$ $\quad [a/S] \to ac$ $\quad [b/S] \to Sa[a/S]$ $\quad [b/S] \to Sb[b/S]$

Теперь присоединим правые контексты:

$$\begin{array}{llll}
S \to b[Sa][a/S] & \mid b[Sb][b/S] & \mid a & \mid aac \\
[Sa] \to b[Sa][[a/S]a] & \mid b[Sb][[b/S]a] & \mid aa & \mid aaca \\
[Sb] \to b[Sa][[a/S]b] & \mid b[Sb][[b/S]b] & \mid ab & \mid aacb \\
[a/S] \to \varepsilon & \mid ac & & \\
[[a/S]a] \to a & \mid aca & & \\
[[a/S]b] \to b & \mid acb & & \\
[[b/S]a] \to [Sa][[a/S]a] & \mid [Sb][[b/S]a] & & \\
[[b/S]b] \to [Sa][[a/S]b] & \mid [Sb][[b/S]b] & &
\end{array}$$

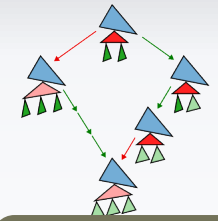


Лемма о накачке для DCFL

Теорема (S. Yu)

Пусть L — DCFL. Тогда существует такая длина накачки p, что для всех пар слов $w, w' \in L$, таких что $w = xy$ & $w' = xz$, $|x| > p$ и первые буквы y, z совпадают, выполнено одно из двух:

1. существует накачка только префикса x (в привычном смысле);
2. существует разбиение $x = x_1x_2x_3$, $y = y_1y_2y_3$, $z = z_1z_2z_3$ такое, что $|x_2x_3| \leqslant p$, $|x_2| > 0$, и $\forall i(x_1x_2^ix_3y_1y_2^iy_3 \in L \;\&\; x_1x_2^ix_3z_1z_2^iz_3 \in L)$.



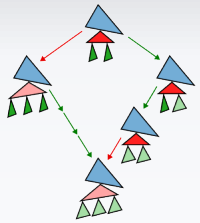
Лемма о накачке для DCFL

Теорема (S. Yu)

Пусть L — DCFL. Тогда существует такая длина накачки p, что для всех пар слов $w, w' \in L$, таких что $w = xy$ & $w' = xz$, $|x| > p$ и первые буквы y, z совпадают, выполнено одно из двух:

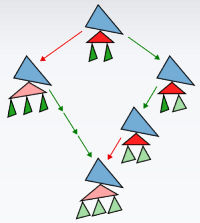
1. существует накачка только префикса x (в привычном смысле);
2. существует разбиение $x = x_1x_2x_3$, $y = y_1y_2y_3$, $z = z_1z_2z_3$ такое, что $|x_2x_3| \leqslant p$, $|x_2| > 0$, и $\forall i(x_1x_2^ix_3y_1y_2^iy_3 \in L \;\&\; x_1x_2^ix_3z_1z_2^iz_3 \in L)$.

Рассмотрим язык $\{a^nb^n\} \cup \{a^nb^{2n}\}$, положим $x = a^nb^{n-1}$, $y = b$, $z = b^{2n-1}$, где $n - 1 > p$. Тогда в случае 2 придётся накачивать в x только b, а в случае 1 нет подходящей накачки.



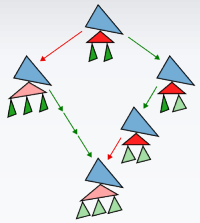
Замыкания DCFL

- Замкнуты относительно дополнения (смена конечных состояний в DPDA).
- Замкнуты относительно пересечения с регулярным языком.
- Не замкнуты относительно объединения (см. $\{a^n b^n\} \cup \{a^n b^{2n}\}$).
- Не замкнуты относительно пересечения.



Замыкания DCFL

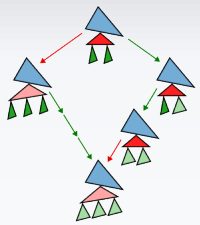
- Замкнуты относительно дополнения (смена конечных состояний в DPDA).
- Замкнуты относительно пересечения с регулярным языком.
- Не замкнуты относительно объединения (см. $\{a^n b^n\} \cup \{a^n b^{2n}\}$).
- Не замкнуты относительно пересечения.
- Не замкнуты относительно гомоморфизмов. См. $\{ca^n b^n\} \cup \{a^n b^{2n}\}$.
- Не замкнуты относительно конкатенации. См. $L_1 = \{ca^n b^n\} \cup \{a^n b^{2n}\}$, $L_2 = c^*$.

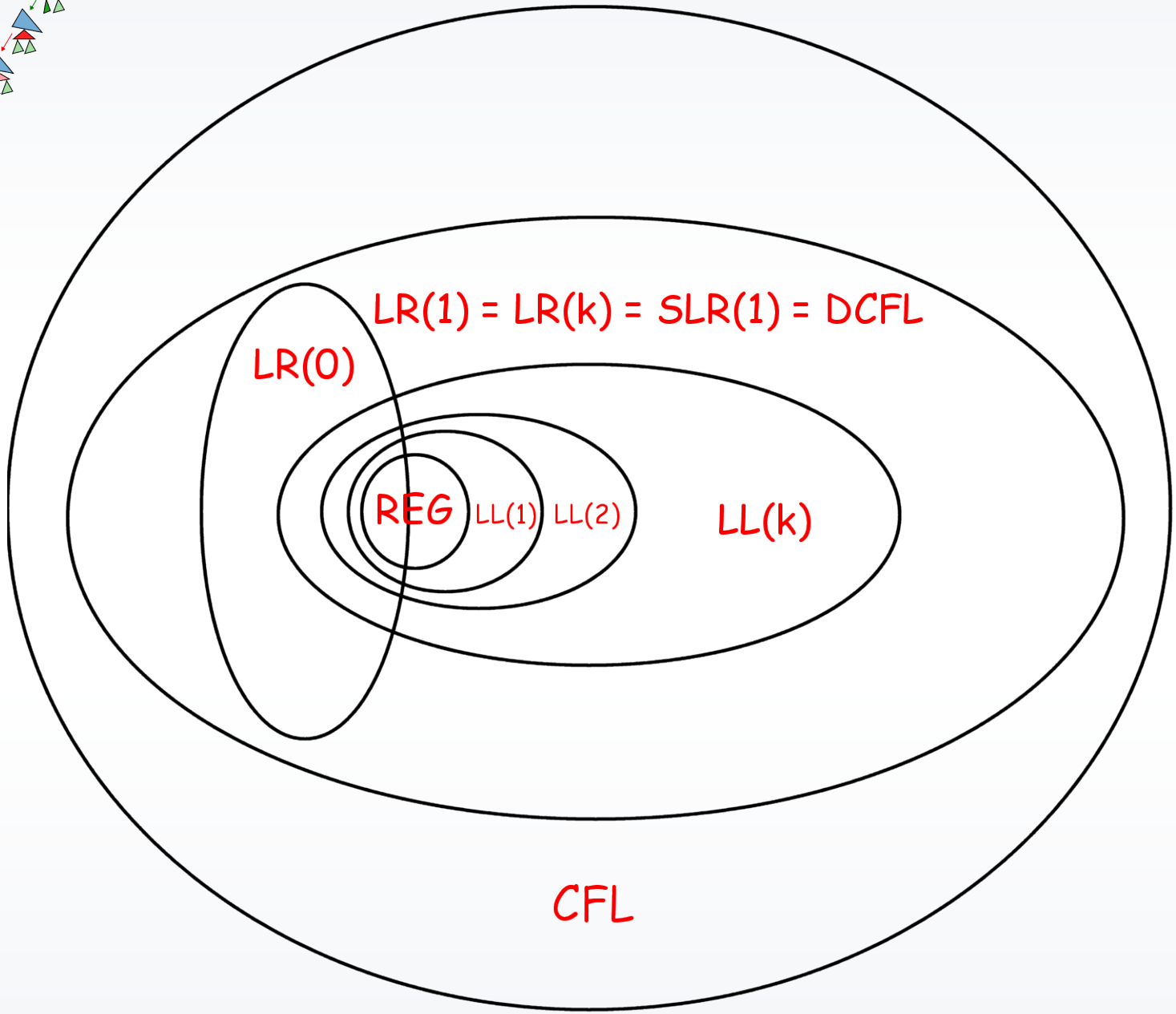


Иерархия Хомского revisited

Утверждения ниже касаются только языков (не грамматик)!

- $\mathrm{RegL} \subset \mathrm{CFL}$;
- $\mathrm{RegL} \subset \mathrm{DCFL}$;
- $\mathrm{DCFL} \subset \mathrm{CFL}$;
- $\mathrm{RegL} \subset \mathrm{LL}(1)$;
- $\mathrm{LR}(0)$ не сравним с RegL;
- $\mathrm{LR}(0)$ не сравним с $\mathrm{LL}(k)$;
- $\mathrm{LL}(k) \subset \mathrm{LL}(k+1)$;
- $\mathrm{LL}(k) \subset \mathrm{LR}(1)$;
- $\mathrm{LR}(k) = \mathrm{SLR}(1) = \mathrm{DCFL}$.




LR(1) = LR(k) = SLR(1) = DCFL
LR(0)
REG
LL(1)
LL(2)
LL(k)
CFL