Формальные грамматики, весна 2019 г.
Лекция 12: Преобразование LL(k)-грамматик к нормальному виду. Ограничения LL(k)-грамматик*

Александр Охотин

13 мая 2019 г.

Содержание

1 Нормальные виды для LL-грамматик

1.1 Удаление пустых правил

Обычная процедура удаления пустых правил может перевести LL-грамматику в не-LL.

Пример 1. *Следующая LL(1)-граматика задаёт язык* $a^*d\{bc, b, c, \varepsilon\}$.

$$S \to ABC$$
$$A \to aA \mid d$$
$$B \to b \mid \varepsilon$$
$$C \to c \mid \varepsilon$$

Обычное преобразование, удаляющее пустые правила, даёт следующую грамматику, которая не LL(k) ни для какого k.

$$S \to ABC \mid AB \mid AC \mid A$$
$$A \to aA \mid d$$
$$B \to b$$
$$C \to c$$

Действительно, на входе $w = a^{k-1}d$, *LL(k)-анализатор не сможет определить, какое из четырёх правил для* S *использовать.*

*Краткое содержание лекций, прочитанных студентам СПбГУ, обучающимся по программе «математика», в весеннем семестре 2018–2019 учебного года. Страница курса: `http://users.math-cs.spbu.ru/~okhotin/teaching/fg_2019/`.

Есть особое построение для удаления пустых правил из LL-грамматик, которое, однако, увеличивает k на единицу.

Теорема 1 (Курки-Суонио [1969]; Розенкранц и Стирнс [1970])**.** *Для всякой LL(k)-грамматики существует и может быть эффективно построена LL(k + 1)-грамматика без пустых правил, задающая тот же язык.*

Первый шаг преобразования: обеспечить, чтобы никакое правило не начиналось с нетерминального символа, задающего пустую строку.

Лемма 1 (Розенкранц и Стирнс [1970])**.** *Для всякой обыкновенной грамматики $G = (\Sigma, N, R, S)$, существует другая грамматика $G' = (\Sigma, N \cup N', R', S')$, где $N' = \{ A' \mid A \in N \}$, удовлетворяющая следующим условиям.*

1. *Всякий нетерминальный символ $A \in N$ задаёт в G' тот же язык, что и в G.*
2. *Всякий нетерминальный символ $A' \in N$ в G' задаёт тот же язык, что и в G, с исключённой пустой строкой: $L_{G'}(A') = L_G(A') \setminus \{\varepsilon\}$. В частности, $L(G') = L(G) \setminus \{\varepsilon\}$.*
3. *Никакое правило из R' не начинается с нетерминального символа из N.*
4. *Если G — LL(k), то и G' — тоже LL(k).*

Доказательство. Пусть $N_0 = \{ A \mid A \in N,\ \varepsilon \in L_G(A) \}$ — нетерминальные символы в G, задающие пустую строку.

Пусть $A \to \alpha$ — произвольное правило из R, и пусть $B_1 \dots B_m$ — самый длинный префикс α, состоящий из нетерминальных символов из N_0.

$$A \to B_1 \dots B_m X_1 \dots X_n \quad (m, n \geqslant 0,\ B_1, \dots, B_m \in N_0,\ X_1, \dots, X_n \in (\Sigma \cup N)^*,\ X_1 \notin N_0)$$

Если LL-анализатор использует это правило в своём вычислении на какой-то входной строке, он обработает символы $B_1, \dots, B_m, X_1, \dots, X_n$ слева направо, прочитывая ε для нуля или более первых символов $B_1, \dots, B_{i-1}$. Со временем он или прочитает непустую строку для некого B_i или для X_1 (если $i - 1 = m$), или же прочитает входную строку для каждого символа этого правила (возможно только если $i - 1 = m$ и $n = 0$).

Покуда непустая строка не прочитывается для некоторого B_i или для X_1, указатель на входную строку не передвигается, и анализатор принимает все свои решения на основе одних и тех же следующих k символов. Все эти решения можно предсказать, глядя на A и на эти следующие k символов, и в новой грамматике, их итоговое действие можно повторить в отдельном правиле, созданном для этого случая. Соответственно, в R' будут следующие правила, соответствующие различному выбору самого левого символа, из которого получается непустая строка.

$$A \to B'_i B_{i+1} \dots B_m X_1 \dots X_n \qquad (\text{для } i \in \{1, \dots, m\}) \tag{1a}$$
$$A \to X_1 \dots X_n \tag{1b}$$

Если $n = 0$, то последнее правило задаёт пустую строку. Правила для второго нетерминального символа A' почти такие же, за исключением того, что пустая строка нигде не задаётся.

$$A' \to B'_i B_{i+1} \dots B_m X_1 \dots X_n \qquad (\text{для } i \in \{1, \dots, m\}) \tag{1c}$$
$$A' \to X_1 \dots X_n \qquad (\text{если } n > 0) \tag{1d}$$

Доказательство правильности построения — что нетерминальные символы новой грамматики задают заявленные языки — проводится обычными методами. В частности, из этого следует, что каждое правило в новой грамматике, полученное из некоторого правила первоначальной грамматики. задаёт подмножество языка, задаваемого исходным правилом.

Остаётся доказать, что свойство LL(k) сохраняется. Пусть G — LL(k). Выбор между двумя правилами для A в грамматике G' делается так. Если эти два правила получены из двух разных правил для A в G, то выбор между новыми правилами в G' производится точно так же, как между исходными. Если они получены из одного и того же правила в G, это правила $A \to B'_i B_{i+1} \dots B_m X_1 \dots X_n$ и $A \to B'_j B_{j+1} \dots B_m X_1 \dots X_n$, где $i < j$. Тогда анализатор для G' выбирает между ними по тому же принципу, по которому анализатор для G решает, получать ли из B'_i пустую строку или непустую. □

Получено, что правая часть каждого правила или пуста, или начинается не с обнуляемого нетерминального символа. Для всякого правила, пусть $X_1, \dots, X_\ell$, где $X_i \in N_1 \cup \Sigma$ и $\ell \geqslant 0$ — это все необнуляемые символы в его правой части, Между ними могут лежать любые строки, построенные из обнуляемых нетерминальных символов. Они обозначаются за $\theta_1, \dots, \theta_n \in N_0^*$. Тогда правило записывается так.

$$Y \to X_1 \theta_1 \dots X_\ell \theta_\ell$$

Главная мысль преобразования — рассматривать каждый обнуляемый кусок θ_i как добавление к предшествующему необнуляемому символу X_i, и представить их в виде одного «большого» нетерминального символа, обозначаемого через $[X_i\theta_i]$. Используя такое представление, решение о том, какие нетерминальные символы задают пустую подстроку, можно отложить до самого последнего момента — в отличие от обычного преобразования по удалению пустых правил, где это происходит при выборе правила.

Лемма 2 (Розенкранц и Стирнс [1970])**.** *Пусть $G = (\Sigma, N, R, S)$ — LL(k)-грамматика, не содержащая правил вида $A \to B\gamma$, где $\varepsilon \in L_G(B)$. Пусть обнуляемые нетерминальные символы обозначаются через $N_0 = \{ A \mid A \in N, \varepsilon \in L_G(A) \}$, и пусть $N_1 = \{ A \mid A \in N, \varepsilon \notin L_G(A) \}$ — все остальные. Тогда существует LL(k + 1)-грамматика $G' = (\Sigma, N', R', S')$, где N' — конечное множество нетерминальных символов вида $[X\theta]$, где $X \in \Sigma \cup N_1$ и $\theta \in N_0^*$, и $L_{G'}([X\theta]) = L_G(X\theta)$ для всякого $[X\theta] \in N'$. В частности, $\varepsilon \notin L_{G'}([X\theta])$ для всех $[X\theta] \in N'$.*

При этом множество N' не содержит ни одного элемента вида $[X\theta A\theta' A\theta'']$, который содержал бы повторяющиеся вхождения одного и того же нетерминального символа.

Правила грамматики G' строятся так. Если $[AE_1 \dots E_k] \in N'$ и в R есть правило $A \to X_1\theta_1 \dots X_\ell\theta_\ell$, где $X_1, \dots, X_\ell \in \Sigma \cup N_1$ и $\theta_1, \dots, \theta_\ell \in N_0^*$, то в новой грамматике есть соответствующее правило, разбитое на ℓ «больших» нетерминальных символов, где $E_1 \dots E_k$ дописаны к последнему из них.

$$[AE_1 \dots E_k] \to [X_1\theta_1] \dots [X_{\ell-1}\theta_{\ell-1}][X_\ell\theta_\ell E_1 \dots E_k] \tag{2}$$

Для нетерминального символа $[aE_1 \dots E_k]$ из N', начинающегося с символа $a \in \Sigma$, для всякого E_i и для всякого непустого правила $E_i \to X_1\theta_1 \dots X_\ell\theta_\ell \in R$, где $\ell \geqslant 1$, $X_1, \dots, X_\ell \in \Sigma \cup N_1$ и $\theta_1, \dots, \theta_\ell \in N_0^*$, в новой грамматике есть правило, в котором опускаются все предшествующие $E_1, \dots, E_{i-1}$ и подставляется правило для E_i.

$$[aE_1 \dots E_k] \to a[X_1\theta_1] \dots [X_{\ell-1}\theta_{\ell-1}][X_\ell\theta_\ell E_{i+1} \dots E_k] \tag{3}$$

Ещё одна разновидность правила для нетерминального символа из N', начинающегося с символа, получает пустую строку из всех $E_1, \dots, E_k$.

$$[aE_1 \dots E_k] \to a \tag{4}$$

Начальный символ новой грамматики — $S' = [S]$.

Чтобы понять, как именно построенная грамматика соотносится с исходной, удобнее всего сформулировать соответствие между деревьями разбора. Каждая вершина $[XE_1 \dots E_k]$ в дереве разбора некоторой строки в грамматике G' соответствует группе вершин $X, E_1, \dots, E_k$ в дереве разбора той же строки в грамматике G, поддеревья которых идут одно вслед за другим. Вершины группируются ради того, чтобы можно было исключить из дерева поддеревья разбора пустой строки.

Пример 2. *Для LL(1)-грамматики из примера 1 преобразование по лемме 2 даёт следующую LL(2)-грамматику.*

$$\begin{aligned}
[S] &\to [ABC] \\
[ABC] &\to a[ABC] \mid [dBC] && (\textit{подставить } A \to aA;\ \textit{подставить } A \to d) \\
[dBC] &\to d[bC] \mid d[c] \mid d && (\textit{подставить } B \to b;\ \textit{подставить } C \to c;\ \textit{опустить } B, C) \\
[bC] &\to b[c] \mid b \\
[c] &\to c
\end{aligned}$$

Набросок доказательства леммы 2. Нетрудно доказать, что $w \in L_{G'}([XE_1 \dots E_k])$ тогда и только тогда, когда $w \in L_G(XE_1 \dots E_k)$ — это делается простой индукцией по высоте дерева разбора.

Конечность числа нетерминальных символов, следует из последнего утверждения леммы об отсутствии нетерминальных символов вида $[X\theta A\theta' A\theta'']$ в N'. Если такие нетерминальные символы появятся, это значит, что в исходной грамматике возможно соседство деревьев разбора $A\theta' A$. Пусть $u \in L_G(A)$. Тогда можно вывести u из первого A, а из остальных — пустую строку; или же наоборот, из второго A – u, и пустую строку из остальных. Получится неоднозначность, поэтому G — не LL.

Условие LL$(k+1)$ на лекции и вовсе не доказывалось. □

Любопытно, что если данная грамматика не LL, то построение в лемме 2 в общем случае не работает. Действительно, конечность числа нетерминальных символов гарантирована только для LL-грамматики.

1.2 Нормальный вид Грейбах

Левая рекурсия в LL-грамматиках совершенно запрещена. Действительно, если есть последовательность правил $A_1 \to A_2\alpha_1$, $A_2 \to A_3\alpha_2$, …, $A_m \to A_0\alpha_m$, то у всех нетерминальных

символов $A_1, \ldots, A_m$ будет одно и то же множество $\textsc{First}_k$, и потому синтаксический анализатор должен будет применять эти правила по кругу, наращивая стек и не читая входных символов.

Нормальный вид Грейбах, в котором все правила имеют вид $A \to a\alpha$, где $a \in \Sigma$ и $\alpha \in (\Sigma \cup N)^*$, не только исключает возможность левой рекурсии, но и вообще удобен для обработки строки слева направо

Теорема 2 (Розенкранц и Стирнс [1970]). *Для всякой LL(k)-грамматики без пустых правил можно построить LL(k)-грамматику в н.в.Грейбах, задающую тот же язык.*

Доказательство. Сперва по теореме 1 удаляются пустые правила. Поскольку грамматика LL, левой рекурсии в ней уже не будет. Тогда достаточно применить конечное число подстановок. □

2 Ограничения LL-грамматик

2.1 Первый способ: подменяемая подстрока впереди

Пример 3 (Розенкранц и Стирнс [1970]). *Язык* $\{ a^n b^n \mid n \geqslant 0 \} \cup \{ a^n c^n \mid n \geqslant 0 \}$ *— не LL(k) ни для какого* k.

Доказательство. Предполагая обратное, пусть $G = (\Sigma, N, R, S)$ — LL(k)-грамматика без пустых правил, задающая этот язык. Для всякого $n \geqslant 0$, пусть $\alpha_n \in (\Sigma \cup N)^*$ — содержимое стека анализатора на входе $a^{n+k}b^{n+k}$ после прочтения символов a^n. На входе $a^{n+k}c^{n+k}$, у анализатора после прочтения a^n будет в стеке то же самое, поскольку он ещё не может видеть, b или c впереди.

Утверждение. $\alpha_m \neq \alpha_n$ *для всех* $m \neq n$.

Действительно, если $\alpha_m = \alpha_n$, то анализатор собьётся со счёту и примет строки $a^{m+k}b^{n+k}$ и $a^{n+k}b^{m+k}$.

Поскольку содержимое стека для разных n разное, его размер не ограничен никаким числом.

Утверждение. *Существует число* $n \geqslant 0$, *для которого* $|\alpha_n| \geqslant k + 2$.

Лемма доказывается через последнее утверждение. Пусть $\alpha_n = X_1 \ldots X_m$, где $X_i \in \Sigma \cup N$ и $m \geqslant k + 2$. Известно, что $a^k b^{n+k}, a^k c^{n+k} \in L_G(\alpha_n) = L_G(X_1 \ldots X_m)$, то есть, существуют разбиения $a^k b^{n+k} = x_1 \ldots x_m$ и $a^k c^{n+k} = y_1 \ldots y_m$ где $x_i, y_i \in L_G(X_i)$. Поскольку ни один из X_i не задаёт пустую строку, $x_i, y_i \neq \varepsilon$, и потому $x_m = b^\ell$ и $y_m = c^{\ell'}$, где $0 < \ell < n + k$ и $0 < \ell' < n + k$. Тогда строка $a^k b^{n+k-\ell} c^{\ell'}$ также лежит в $L_G(\alpha_n)$, в потому вся строка $a^{n+k}b^{n+k-\ell}c^{\ell'}$ задаётся грамматикой, противоречие. □

Этим же методом доказываются следующие примеры.

Пример 4 (Вуд). *Язык* $\{ a^n b a^n b \mid n \geqslant 0 \} \cup \{ a^n c a^n c \mid n \geqslant 0 \}$ *не LL(k) ни для какого* k.

Пример 5 (Битти). *Язык* $\{ a^n cb^n \mid n \geqslant 0 \} \cup \{ a^n db^{2n} \mid n \geqslant 0 \}$ *не LL(k) ни для какого* k.

Каждый из этих примеров даёт незамкнутость относительно объединения. Из примера 4 получается также незамкнутость относительно пересечения с регулярным языком — поскольку язык $\{ a^n sb^n t \mid n \geqslant 0,\ s,t \in \{c,d\} \}$ задаётся LL(1)-грамматикой, однако его пересечение с регулярным языком $a^*cb^*c \cup a^*db^*d$ — это язык из примера 4.

2.2 Второй способ: неожиданный конец

Пример 6 (Вуд [1971]). *Язык* $L = a^* \cup \{ a^n b^n \mid n \geqslant 0 \}$ *не LL(k) ни для каких* k.

Доказательство. Если он задаётся LL-грамматикой, то, стало быть, он задаётся некоторой LL(k)-грамматикой без пустых правил. С одной стороны, прочитав a^n, анализатор должен запомнить n в стеке, что требует неограниченного числа символов в стеке. С другой стороны, он должен всегда быть готов прочитать строку a^k и принять, и потому в стеке не может быть больше k символов. □

Отсюда — незамкнутость относительно объединения с регулярным языком.

Список литературы

[1969] R. Kurki-Suonio, “Notes on top-down languages”, *BIT Numerical Mathematics*, 9:3 (1969), 225–238.

[1970] D. J. Rosenkrantz, R. E. Stearns, “Properties of deterministic top-down grammars”, *Information and Control*, 17 (1970), 226–256.

[1971] D. Wood, “A further note on top-down deterministic languages”, *Computer Journal*, 14:4 (1971), 396–403.