Теория автоматов и формальных языков

Контекстно-свободные языки

Лектор: Екатерина Вербицкая

Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»

4 октября 2016г.

В предыдущей серии

- Регулярные выражения, регулярные грамматики и конечные автоматы задают класс регулярных языков
- Класс регулярных языков замкнут относительно теоретико-множественных операций, конкатенации, итерации, гомоморфизма цепочек
- Определение принадлежности слова языку осуществляется за $O(n)$ операций
- Однако класс регулярных языков достаточно узок, ни один используемый в промышленности язык программирования не является регулярным
  - Лемма о накачке для доказательства нерегулярности языка
  - Язык правильных скобочных последовательностей, язык палиндромов не являются регулярными

Контекстно-свободная грамматика

Четверка $\langle V_T, V_N, P, S \rangle$

- V_T — алфавит терминальных символов (терминалов)
- V_N — алфавит нетерминальных символов (нетерминалов)
  - $V_T \cap V_N = \emptyset$
  - $V ::= V_T \cup V_N$
- P — конечное множество правил вида $A \to \alpha$
  - $A \in V_N$
  - $\alpha \in V^*$
- S — начальный нетерминал грамматики, $S \in V_N$

Пример: арифметические выражения

$$\begin{aligned} E &\to E + E \,|\, E * E \,|\, N \\ N &\to 0 \,|\, 1 \,|\, \ldots \,|\, 9 \end{aligned}$$

Вывод в грамматике

- **Отношение выводимости**:
  $\forall \alpha, \gamma, \delta \in V^*, A \in V_N : A \to \alpha \in P. \gamma A \delta \Rightarrow \gamma \alpha \delta$
- **Вывод** — транизитивное, рефлексивное замыкание отношения выводимости $(\overset{*}{\Rightarrow}, \overset{+}{\Rightarrow}, \overset{k}{\Rightarrow})$
- **Левосторонний (правосторонний) вывод** — на каждом шаге заменяем самый левый (правый) нетерминал
  - Если не специфицируется, подразумевается левосторонний вывод
- По сути, правила грамматики рассматриваются как правила переписывания

Пример вывода

Построим левосторонний вывод цепочки $2 + 3 * 4$ в грамматике $\langle \{0, 1, \ldots, 9, +, *\}, \{E, N\}, P, E \rangle$

$$\begin{array}{lcl} E & \to & E + N \,|\, E * N \,|\, N \\ N & \to & 0 \,|\, 1 \,|\, \ldots \,|\, 9 \end{array}$$

$$E \Rightarrow E * N \Rightarrow E + N * N \Rightarrow N + N * N \Rightarrow 2 + N * N \overset{2}{\Rightarrow} 2 + 3 * 4$$

Существование левостороннего вывода

Теорема

Если для цепочки ω существует некоторый вывод $S \overset{*}{\Rightarrow} \omega$, то существует и левосторонний вывод для этой цепочки $S \underset{l}{\overset{*}{\Rightarrow}} \omega$

Доказательство.

Докажем более общее утверждение: если существует $A \overset{*}{\Rightarrow} \omega$, то существует $A \underset{l}{\overset{*}{\Rightarrow}} \omega$, где $A \in V_N$.

Доказываем по индукции по длине вывода k

$k = 1 : A \Rightarrow \omega$ — тривиально.

$k \to k+1 : \lhd A \Rightarrow \alpha \overset{*}{\Rightarrow} \omega$.

Обозначим $\alpha = B_1 B_2 \dots B_m \overset{*}{\Rightarrow} \omega_1 \omega_2 \dots \omega_m = \omega; \forall i . B_i \overset{l_i}{\Rightarrow} \omega_i, l_i \leq n$

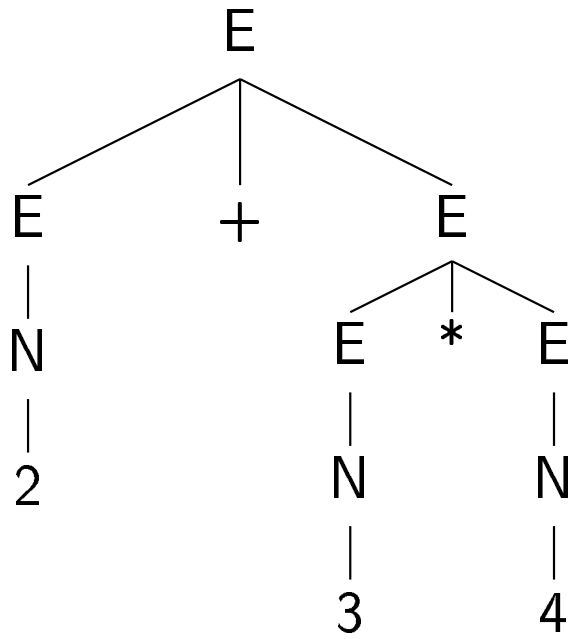
По индукционному предположению $\forall i.\ B_i \underset{l}{\overset{*}{\Rightarrow}} \omega_i$

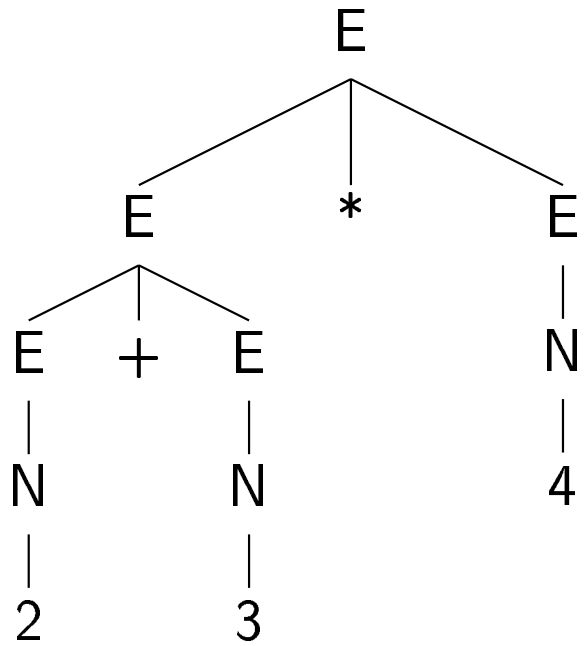
$\Mapsto$: $A \Rightarrow B_1 B_2 \dots B_m \underset{l}{\overset{*}{\Rightarrow}} \omega_1 B_2 \dots B_m \underset{l}{\overset{*}{\Rightarrow}} \omega$ — левосторонний вывод $\square$

Единственность вывода

Не всегда (левосторонний) вывод единственен: 2 вывода строки $2 + 3 * 4$

$$
\begin{aligned}
E &\rightarrow E + E \,|\, E * E \,|\, N \\
N &\rightarrow 0 \,|\, 1 \,|\, \ldots \,|\, 9
\end{aligned}
$$





Однозначность грамматики

- Грамматика называется **однозначной**, если для *любого* слова языка существует *единственный* (левосторонний) вывод
- Грамматика называется **неоднозначной**, если *существует* слово языка, такое что для него *существует несколько* (левосторонних) выводов

- По однозначной грамматике можно тривиальным образом построить неоднозначную: продублировать правило
  - $S \to A; A \to a$
  - $S \to A|B; A \to a; B \to a$
- Не существует общего алгоритма преобразования неоднозначной грамматики в однозначную

Примеры однозначной и неоднозначной грамматики

- Неоднозначная грамматика

$$\begin{aligned} E &\rightarrow E + E \,|\, E * E \,|\, N \\ N &\rightarrow 0 \,|\, 1 \,|\, \ldots \,|\, 9 \end{aligned}$$

- Однозначная грамматика

$$\begin{aligned} E &\rightarrow E + N \,|\, E * N \,|\, N \\ N &\rightarrow 0 \,|\, 1 \,|\, \ldots \,|\, 9 \end{aligned}$$

Проверка однозначности грамматики — неразрешимая задача

- Проверка однозначности грамматик сводится к задаче соответствий Поста
- Задача соответствий Поста: Даны списки $A = (a_1, \ldots, a_n)$ и $B = (b_1, \ldots, b_n)$, где $\forall i.\, a_i \in \Sigma^*$ и $b_i \in \Sigma^*$.Существует ли непустая последовательность $(i_1, \ldots, i_k)$, удовлетворяющая условию $a_{i_1} \ldots a_{i_k} = b_{i_1} \ldots b_{i_k}$, где $\forall j.\, 1 \leq i_j \leq n$

Контекстно-свободный язык

- Язык называется **контекстно-свободным**, если для него *существует* контекстно-свободная грамматика
- Язык, задаваемый КС грамматикой $\langle V_T, V_N, P, S \rangle$: $\{\omega \in V_T^* | S \stackrel{*}{\Rightarrow} \omega\}$
- КС язык называется **существенно неоднозначным**, если для него не существует однозначной грамматики

Пустота КС языка

Теорема

Существует алгоритм, определяющий, является ли язык, порождаемый КС грамматикой, пустым

Доказательство.

Для доказательства потребуется следующая лемма □

Лемма

Теорема

Если в данной грамматике выводится некоторая цепочка, то существует цепочка, дерево вывода которой не содержит ветвей длиннее m, где m — количество нетерминалов грамматики

Доказательство.

Рассмотрим дерево вывода цепочки ω. Если в нем есть 2 узла, соответствующих одному нетерминалу A, обозначим их n_1 и n_2. Предположим, n_1 расположен ближе к корню дерева, чем n_2; $A_{n_1} \stackrel{*}{\Rightarrow} \alpha\omega_1\beta; A_{n_2} \stackrel{*}{\Rightarrow} \gamma\omega_2\delta$. При этом ω_2 является подцепочкой ω_1. Заменим в изначальном дереве узел n_1 на n_2. Полученное дерево является деревом вывода $\alpha\omega_2\delta$. Повторяем процесс замены одинаковых нетерминалов до тех пор, пока в дереве не останутся только уникальные нетерминалы.
В полученном дереве не может быть ветвей длины большей, чем m.
По постороению оно является деревом вывода. □

Алгоритм проверки пустоты КС языка

Доказательство.

Строим коллекцию деревьев, представляющих вывод в грамматике.

1. Инициализируем коллекцию деревом из одного узла S
2. Добавляем в коллекцию дерево, полученное применением единственного правила грамматики из какого-нибудь дерева из коллекции, если его в нем еще нет, и самая длинная ветвь не длиннее m
3. Если после окончания построения коллекции в ней существует дерево, являющееся деревом вывода некоторой цепочки терминалов, значит, язык не пуст

Упрощение КС грамматики: удаление непродуктивных нетерминалов

Продуктивный нетерминал: нетерминал, для которого существует цепочка терминалов, выводимая из него ($\exists \omega \in V_T^*. A \overset{*}{\Rightarrow} \omega$)
Непродуктивный нетерминал: нетерминал, не являющийся продуктивным

Упрощение КС грамматики: удаление непродуктивных нетерминалов

Теорема

Для любой КС грамматики $G = \langle V_T, V_N, P, S \rangle : L(G) \neq \varnothing$, *можно постороить эквивалентную грамматику, каждый нетерминал которой продуктивен*

Доказательство.

Удаляем из грамматики все нетерминалы $A : L(A) = \varnothing$, а также правила, использующие их. Полученную грамматику обозначаем G_1. Докажем, что $L(G) = L(G_1)$. Очевидно, $L(G_1) \subseteq L(G)$.
Докажем от противного, что $L(G) \subseteq L(G_1)$. Предположим, что $\exists \omega \in L(G)$, но $\omega \notin L(G_1)$. Тогда $S \overset{*}{\Rightarrow} \alpha_1 A \alpha_2 \overset{*}{\Rightarrow} \omega$, где $A \in V_N \setminus V_{N_1}$, но тогда $\exists \gamma \in V_T^*. A \overset{*}{\Rightarrow} \gamma$. Противоречие $\square$

Упрощение КС грамматики: приведение

Теорема

Для любой КС грамматики, порождающей непустой язык, можно постороить эквивалентную, для каждого нетерминала A которой существует вывод вида $S \overset{*}{\Rightarrow} \omega_1 A \omega_3 \overset{*}{\Rightarrow} \omega_1 \omega_2 \omega_3, \omega_i \in V_T^*$

Доказательство.

Будем рассматривать грамматику без непродуктивных нетерминалов $G_1 = \langle V_{N_1}, V_T, P_1, S \rangle$.

Верно: если существует $S \overset{*}{\Rightarrow} \alpha_1 A \alpha_3, \alpha_i \in V^*$, то $S \overset{*}{\Rightarrow} \alpha_1 A \alpha_3 \overset{*}{\Rightarrow} \omega_1 A \omega_3 \overset{*}{\Rightarrow} \omega_1 \omega_2 \omega_3, \omega_i \in V_T^*$

Строим множество нетерминалов, встречающихся в выводах: добавляем сначала S, потом добавляем нетерминалы, встречающиеся в правой части правил для нетерминалов из множества. Завершаем процесс, когда больше ничего не добавить. Обозначаем полученное множество V_{N_2}, удаляем все правила грамматики, содержащие нетерминалы из $V_{N_1} \setminus V_{N_2}$

Упрощение КС грамматики: приведение

Доказательство.

Получили грамматику $G_2 = \langle V_{N_2}, V_T, P_2, S \rangle$.
Докажем: $L(G_2) = L(G_1)$
$L(G_2) \subseteq L(G_1)$, так как $P_2 \subseteq P_1$
Докажем: $L(G_1) \subseteq L(G_2)$. Пусть $S \underset{G_1}{\overset{*}{\Longrightarrow}} \omega$. Все нетерминалы, встречающиеся в этом выводе содержатся в V_{N_2}, соответственно используются только правила из $P_2 \Rightarrow S \underset{G_2}{\overset{*}{\Longrightarrow}} \omega$
Так как все нетерминалы V_{N_2} продуктивны, то
$S \overset{*}{\Rightarrow} \omega_1 A \omega_3 \overset{*}{\Rightarrow} \omega_1 \omega_2 \omega_3, \omega_i \in V_T^*$

□

Грамматика G_2 называется **приведенной**, ее нетерминалы — **достижимыми**
Недостижимые и непродуктивные нетерминалы называются **бесполезными**

Упрощение КС грамматики: удаление цепных правил

Правило называется **цепным**, если оно имеет вид $A \to B; A, B \in V_N$.

Теорема

Для любой КС грамматики $G = \langle V_N, V_T, P, S \rangle$ можно построить эквивалентную, не содержащую цепных правил

Доказательство.

Строим новое множество правил P_1. Включаем в него все нецепные правила P. Затем добавляем в P_1 правила вида $A \overset{*}{\Rightarrow} \alpha$, если $A \overset{*}{\Rightarrow} B$, где $A, B \in V_N$ и $B \to \alpha$ — нецепное правило из P.

Замечание: достаточно проверять только цепные выводы длины меньшей, чем V_N

Обозначем полученную грамматику за $G_1 = \langle V_N, V_T, P_1, S \rangle$, докажем $L(G_1) = L(G)$ □

Упрощение КС грамматики: удаление цепных правил

Доказательство.

Очевидно $L(G_1) \subseteq L(G)$

Покажем $L(G) \subseteq L(G_1)$. Пусть $\omega \in L(G)$. Рассмотрим левосторонний вывод $S \underset{G}{\Rightarrow} \alpha_0 \underset{G}{\Rightarrow} \alpha_1 \underset{G}{\Rightarrow} \ldots \underset{G}{\Rightarrow} \alpha_n = \omega$.

Предположим $\alpha_i \underset{G}{\Rightarrow} \alpha_{i+1}$ — первый шаг, выполняемый посредством цепного правило в выводе; $\forall k \in [i..j]. \alpha_k \underset{G}{\Rightarrow} \alpha_{k+1}$ — посредством цепного правила; $\alpha_j \underset{G}{\Rightarrow} \alpha_{j+1}$ — посредством нецепного правила

Тогда $|\alpha_i| = |alpha_{i+1}| = \cdots = |alpha_j|$, и на каждом шаге заменяется один и тот же нетерминал. Тогда $\alpha_i \underset{G_1}{\Rightarrow} \alpha_{j+1}$ посредством правила из $P_1 \setminus P \Rightarrow \omega \in L(G_1)$ □

Нормальная форма Хомского

КС грамматика находится в **нормальной форме Хомского**, если все ее правила имеют вид $A \to BC$, или $A \to a$, где $A, B, C \in V_N, a \in V_T$

Теорема

Для любой КС грамматики можно построить эквивалентную в нормальной форме Хомского

1. Удаляем цепные правила. Теперь $\forall A \to B. B \in V_T$
2. Заменяем правило $A \to B_1 B_2 \dots B_n$ на $A \to C_1 C_2 \dots C_n$, где $C_i = B_i$, если $B_i \in V_N$, или $C_i \to B_i$, если $B_i \in V_T$
3. Заменяем правило $A \to C_1 C_2 \dots C_n$ на множество правил $A \to C_1 D1, D1 \to C_2 D_2, \dots, D_{n-3} \to C_{n-2} D_{n-2}, D_{n-2} \to C_{n-1} C_n$

Полученная грамматика находится в НФХ и эквивалентна данной

Пример приведения в НФХ

$G = \langle \{S, A, B\}, \{a, b\}, P, S \rangle$, где P:

$$
\begin{array}{lcllll}
S & \to & bA & | & aB & \\
A & \to & a & | & aS & | \quad bAA \\
B & \to & b & | & bS & | \quad aBB
\end{array}
$$