Формальные грамматики

Основные понятия

Алфавит - это конечное множество символов.

Цепочка – последовательность символов.

Терминальная цепочка – последовательность терминальных символов.

Язык – множество терминальных цепочек.

Грамматика – это четвёрка $G = (N, T, P, S)$, где:

- N – множество нетерминальных символов (напр. $A, B, C...$);
- T – алфавит терминальных символов ($N \cap T = \emptyset$);
- P – конечное множество правил вывода
  $P = \{\alpha \to \beta \mid \alpha \in (N \cup T)^+;\ \beta \in (N \cup T)^*\}$;
- $S \in N$ – аксиома (или начальный символ) грамматики.

Язык, порождённый грамматикой – множество всех терминальных цепочек, выводимых из аксиомы грамматики. $L(G) = \{W \mid W \in T^*, S \Rightarrow^+ W\}$

Сентенциальная форма – последовательность символов (терминалов и нетерминалов), выводимых из начального символа (аксиомы).

Основа цепочки – это подцепочка сентенциальной формы, которая может быть сопоставлена правой части некоторого правила вывода, свертка по которому к левой части правила соответствует одному шагу в обращении правостороннего вывода.

Основа сентенциальной формы – позиция в сентенциальной форме, которая заменена в следующей сентенциальной форме. Грамматика называется **неоднозначной,** если для некоторой цепочки существует хотя бы два дерева вывода.

Основа правой сентенциальной формы z – это правило вывода $A \to v$ и позиция в z, в которой может быть найдена цепочка v такая, что в результате замены v на A получается предыдущая сентенциальная форма в правостороннем выводе z. Таким образом, если $S \Rightarrow_r avb$, то $A \to v$ в позиции, следующей за a - это основа цепочки avb. Подцепочка b справа от основы содержит только терминальные символы.

Символ x называется **недостижимым,** если он не входит ни в одну сентенциальную форму (не существует вывода $S \Rightarrow^* \alpha x \beta$).

Символ x называется **бесплодным,** если из него нельзя вывести ни одну терминальную цепочку (не существует вывода $x \Rightarrow w, w \in T^*$).

Бесполезный символ – это недостижимый или бесплодный символ.

Приведённая грамматика – это грамматика, не содержащая бесплодных и недостижимых символов.

Множество $FIRST(A)$ – множество терминальных символов, которыми начинаются цепочки, выводимые из A в грамматике G.

Множество $FOLLOW(A)$ – множество терминальных символов, которые следуют за цепочками, выводимыми из A в грамматике G.

Множество $FIRST1$ – множество всех терминальных символов, с которых может начинаться цепочка терминальных символов, выводимых из цели грамматики или ϵ, если $u \Rightarrow^* \epsilon$.

Лемма о разрастании Для любого КС-свободного языка L существуют целые числа l, k такие, что $\forall \alpha \in L, \mid \alpha \mid > l$ представима в виде $\alpha = uvwxy$, где:

- $\mid vwx \mid \leq k$;
- $vx \neq \epsilon$;
- $uv^i wx^i y \in L \; \forall i \geq 0$.

Лемма о накачке Для любого регулярого языка L существует натуральное число $p(L) \geq 1$ такое, что $\forall \alpha \in L, \mid w \mid \geq p$, представима в виде $\alpha = xyz$, где:

- $\mid y \mid \geq 1$;
- $\mid xy \mid \leq p$;
- $xy^i z \in L \; \forall i \geq 0$.

Иерархия Хомского

Если на грамматику $G = (N, T, P, S)$ не накладываются никакие ограничения, то её называют **грамматикой типа 0,** или **грамматикой без ограничений.**

Если:

- каждое правило грамматики, кроме $S \to \epsilon$, имеет вид $\alpha \to \beta, \mid \alpha \mid \leq \mid \beta \mid$;
- в том случае, когда $S \to \epsilon \in P$, символ S не встречается в правых частях правил; то грамматику называют **грамматикой типа 1,** или *неукорачивающей* грамматикой.

Если каждое правило $p \in P$ грамматики имеет вид $A \to \alpha, A \in N, \alpha \in (N \cup T)^*$, то такая грамматика называется **грамматикой типа 2** или *контекстно-свободной* грамматикой.

Если каждое правило $p \in P$ имеет вид $A \to xB$ или $A \to x$ (соответственно $A \to Bx$ или $A \to x$), где $A, B \in N, x \in T^*$, то такая грамматика называется **грамматикой типа 3** или **леволинейной** (соответственно, **праволинейной**) грамматикой.

Из определения иерархии следует, что $Z_0 \supseteq Z_1 \supseteq Z_2 \supseteq Z_3$.

Утверждение: $Z_0 \supset Z_1 \supset Z_2 \supset Z_3$.

Данное утверждение доказывается конструктивно (путём построения соответствующих грамматик).

Грамматика находится в **нормальной форме** Хомского, если правила вывода имеют вид:

- либо $A \to BC; A, B, C \in N$;

- либо $A \to a; a \in T^+$;
- либо $S \to \epsilon$, и в этом случае S не встречается в правых частях правил.

Выводы, однозначность, детерминированность и прочие вопросы

Определим на множестве $(N \cup T)^*$ грамматики $G = (N, T, P, S)$ бинарное отношение выводимости $\Rightarrow$ следующим образом: если $\delta \to \gamma \in P$, то $\alpha\delta\beta \Rightarrow \alpha\gamma\beta \forall \alpha, \beta \in (N \cup T)^*$.

- Если $\alpha_1 \Rightarrow \alpha_2$, то α_2 непосредственно выводима из α_1.
- Если $\alpha \Rightarrow^k \beta (k \geq 0)$, то существует последовательность шагов $\gamma_0 \Rightarrow \gamma_1 \Rightarrow \ldots \Rightarrow \Gamma_k$, где $\alpha = \gamma_0$ и $beta = \gamma_k$.

Последовательность цепочек $\gamma_0, \gamma_1, \ldots, \gamma_k$ в этом случае называется выводом β из α

КС-грамматика называется **однозначной** или **детерминированной,** если всякая выводимая терминальная цепочка имеет только одно дерево вывода (соотвественно только один левый и только один правый вывод).

КС-грамматика называется **неоднозначной,** если для некоторой цепочки существует хотя бы два различных дерева вывода.

Левосторонний вывод – такой вывод, на каждом шаге которого заменяется самый левый нетерминал.

Правосторонний вывод – такой вывод, на каждом шаге которого заменяется самый правый нетерминал.

Грамматика называется **леворекурсивной,** если в ней имеется нетерминал A такой, что существует вывод $A \Rightarrow^+ Aw$ для некоторой цепочки w.

Атрибутная грамматика – это четвёрка $AG = (G, A_S, A_I, R)$, где:

- $G = (N, T, P, S)$ – приведённая контекстно-свободная грамматика;
- A_S - конечное множество синтезируемых элементов;
- A_I –конечное множество наследуемых атрибутов, $A_S \cap A_I = \emptyset$;
- R – конечное множество семантических правил.

Машина Тьюринга и связанные определения

Недетерминированная машина Тьюринга – это шестёрка $M = (Q, \Gamma, T, F, q_0, \sigma)$, где:

- Q – конечное непустое множество состояний;
- Γ – конечный алфавит;
- T – входной алфавит, $T \subseteq \Gamma \cup \epsilon$;
- $q_0 \in Q$ – начальное состояние;
- $F \subseteq Q$ – множество конечных состояний;

- σ – функция переходов, определённая как отображение $\sigma : (Q \setminus F) \times \Gamma \to 2^{Q \times \Gamma \times \{L,R\}}$.

Детерминированная машина Тьюринга – это шестёрка $M = (Q, \Gamma, T, F, q_0, \sigma)$, где:

- Q – конечное непустое множество состояний;
- Γ – конечный алфавит;
- T – входной алфавит, $T \subseteq \Gamma \cup \epsilon$;
- $q_0 \in Q$ – начальное состояние;
- $F \subseteq Q$ – множество конечных состояний;
- σ – функция переходов, определённая как отображение $\sigma : (Q \setminus F) \times \Gamma \to Q \times \Gamma \times \{L, R\}$.

Детерминированная машина Тьюринга из данного состояния по данному символу может сделать не более одного перехода, недетерминированная же таким свойством не обладает.

Конфигурацией машины Тьюринга называется тройка (q, w, i), где:

- $q \in Q$ – состояние машины Тьюринга;
- $w \in \Gamma^*$ – вход, помещаемый на ленту машины Тьюринга ($w = a_1 \ldots a_n$);
- $i \in Z$ – положение головки машины Тьюринга.

Язык, **допускаемый** машиной Тьюринга – множество таких слов w, что машина Тьюринга, находясь в состоянии $(q_0, w, 1)$, может достигнуть через конечное число переходов состояния $q \in F$.

Линейно-ограниченный автомат – это недетерминированная машина Тьюринга, которая не может выходить за область входной строки.

Утверждения

- класс языков, допускаемых машиной Тьюринга, эквивалентен классу языков, порождаемых грамматиками типа 0;
- линейно-ограниченные автоматы распознают КЗ-языки (т.е. языки класса 1); языки класса 0 распознаются только машинами Тьюринга с неограниченной памятью.

Регулярные множества, выражения и грамматики

Регулярное множество в алфавите T определяется следующим образом:

- $\emptyset$ – регулярное множество в алфавите T;
- $\{a\}$ – регулярное множество в алфавите T для каждого $a \in T$;
- $\{\epsilon\}$ – регулярное множество в алфавите T;
- если P и Q – регулярные множества в алфавите T, то регулярны множества:
    - $P \cup T$ (объединение);

узел n	$firstpos(n)$
ϵ	$\emptyset$
$i \neq \epsilon$	$\{i\}$
$u \mid v$	$firstpos(u)\vert firstpos(v)$
$u.v$	$nullable(u)\ ?\ firstpos(u) \cup firstpos(v) : firstpos(u)$
v^*	$firstpos(v)$

Таблица 1: Построение $firstpos(n)$

- PQ (конкатенация, $\{pq \mid p \in P, q \in Q\}$;
- P^* (итерация, $P^* = \{\epsilon\} \cup P \cup PP \cup PPP \cup \ldots$);

• ничто другое не является регулярным множеством в алфавите T.

Регулярное выражение – это форма записи регулярного множества. Регулярное выражение и обозначаемое им регулярное множество в данном алфавите T определяются следующим образом:

- $\emptyset$ – обозначает множество $\emptyset$ в данном алфавите T;
- a - обозначает множество $\{a\}$;
- ϵ – обозначает множество $\{\epsilon\}$ в данном алфавите T;
- если регулярные выражения p и q обозначают регулярные множества P и Q в данном алфавите T соответственно, то регулярное выражение:
  - $(p \mid q)$ обозначает множество $P \cup Q$ в данном алфавите T;
  - (pq) обозначает множество PQ в данном алфавите T;
  - (p^*) обозначает множество P^* в данном алфавите T;
- ничто другое не является регулярным выражением в данном алфавите T.

Регулярные грамматики – это праволинейные и леволинейные грамматики.

Функция $nullable(n)$ для каждого узла n синтаксического дерева регулярных выражений даёт значение `true`, если поддеревья данного узла могут породить пустое слово, и `false` в противном случае.

Функция $firstpos(n)$ для каждого узла n синтаксического дерева регулярных выражений даёт множество позиций, которые соответствуют первым символам в цепочках, генерируемых подвыражением с вершиной n.

Функция $lastpos(n)$ для каждого узла n синтаксического дерева регулярных выражений даёт множество позиций, которым соответствуют последние символы в цепочках, генерируемых подвыражениями с вершиной n.

Функция $followpos(i)$ для позиции i есть множество позиций j таких, что существует некоторая строка $\ldots cd \ldots$, входящая в язык, описываемый регулярным выражением, такая, что позиция i соответствует вхождению c, а позиция j — вхождению d.

Утверждение: любой конечный автомат распознает регулярное множество цепочек символов входного алфавита. Верно и обратное – для любого регуляр-

узел n	$firstpos(n)$
ϵ	$\emptyset$
$i \neq \epsilon$	$\{i\}$
$u \mid v$	$lastpos(u) \vert lastpos(v)$
$u.v$	$nullable(v)\ ?\ lastpos(u) \cup lastpos(v) : lastpos(v)$
v^*	$lastpos(v)$

Таблица 2: Построение $lastpos(n)$

ного языка можно построить распознающий его конечный автомат.

Автоматы

Недетерминированный конечный автомат – это пятёрка $M = (Q, T, F, q_0, \sigma)$, где:

- Q – конечное непустое множество состояний;
- T – входной алфавит;
- $q_0 \in Q$ – начальное состояние;
- $F \subseteq Q$ – множество конечных состояний;
- σ – функция переходов, определённая как отображение $\sigma : Q \times (T \cup \{\epsilon\}) \to 2^Q$, такое, что $\sigma(q, a) = \{r : q \to_a r\}$.

Детерминированный конечный автомат – это пятёрка $M = (Q, T, F, q_0, \sigma)$, где:

- Q – конечное непустое множество состояний;
- T – конечный входной алфавит;
- $q_0 \in Q$ – начальное состояние;
- $F \subseteq Q$ – множество конечных состояний;
- σ – функция переходов, определённая как отображение $\sigma : Q \times T \to Q$

Недетерминированный конечный автомат является обобщением детерминированного.

Утверждение

Любой недетерминированный конечный автомат может быть преобразован в детерминированный так, чтобы их языки совпадали (такие автоматы называются эквивалентными).

В детерменированном автомате из одного состояния допускается не более одного перехода для каждого символа алфавита, в недетерменированном - произвольное количество. Кроме того, в НКА возможны ϵ-переходы.

Пусть $M = (Q, T, D, q_0, \sigma)$ – недетерминированный конечный автомат. **Конфигурацией автомата** M называется пара $(q, \omega) \in Q \times T^*$, где q – текущее состояние управляющего устройства, а ω – цепочка символов на входной ленте, состоящая из символов под головкой и всех символов справа от неё.

Два состояния q_i и q_j называются **эквивалентными** $(q_i \sim q_j)$, если $\forall z \in T^* \sigma(q_i, z) \in T \Leftrightarrow \sigma(q_j, z)\ inT$.

Два состояния q_i и q_j называются **различимыми,** если они не являются эквивалентными.

Автомат M **допускает цепочку** ω, если $M(q_0, \omega) \mid -^*(q, \omega), \forall q \in \sigma$.

Языком, допускаемым автоматом M, называется множество входных цепочек, допускаемых автоматом M, то есть множество $L(M) = \{\omega \mid \omega \in T^*$ и $(q_0, \omega) \mid -^*(q, \epsilon), \forall q \in \sigma\}$.

ϵ-замыкание множества состояний $R, R \subseteq Q$ – множество состояний НКА, достижимых из состояний, входящих в R, посредством только переходов по ϵ, то есть множество $S = U_{q \in R}\{p \mid (q, \epsilon) \mid -^*(p, \epsilon)\}$.

Расширенная функция переходов множества состояний $R, R \subseteq Q$ по a – множество состояний КА, в которые есть переход на входе a для состояний из R, то есть множество $S = U_{q \in R}\{p \mid p \in \sigma(q, a)\}$.

Автоматы с магазинной памятью

Недетерминированный МП-автомат – это семёрка $M = (Q, T, \Gamma, \sigma, q_0, Z_0, F)$, где:

- Q – конечное множество состояний, представляющее всевозможные состояния управляющего устройства; T – конечный входной алфавит; Γ – конечный алфавит магазинных символов; σ – отображение $Q \times (T \cup \{\epsilon\}) \times \Gamma \to Q \times \Gamma^*$, называемое функцией переходов; $q_0 \in Q$ – начальное состояние управляющего устройства; $Z_0 \in \Gamma$ – символ, находящийся в магазине в начальный момент (начальный символ магазина); $F \subseteq Q$ – множество заключительных состояний.

Детерминированный МП-автомат – это семёрка $M = (Q, T, \Gamma, \sigma, q_0, Z_0, F)$, где:

- Q – конечное множество состояний, представляющее всевозможные состояния управляющего устройства; T – конечный входной алфавит; Γ – конечный алфавит магазинных символов; σ – отображение $Q \times (T \cup \{\epsilon\}) \times \Gamma \to Q \times \Gamma^*$, называемое функцией переходов; $q_0 \in Q$ – начальное состояние управляющего устройства; $Z_0 \in \Gamma$ – символ, находящийся в магазине в начальный момент (начальный символ магазина); $F \subseteq Q$ – множество заключительных состояний.

Кроме того, должны выполняться следующие условия:

- множество $D(q, a, Z)$ содержит не более одного элемента $\forall q \in Q, a \in T \cup \{\epsilon\}, Z_0 \in \Gamma$;
- если $D(q, \epsilon, Z) \neq \emptyset$, то $D(q, a, Z) = \emptyset \forall a \in T$.

Конфигурацией МП-автомата называется тройка (q, w, u), где

- $q \in Q$ – текущее состояние магазинного устройства;
- $w \in T^*$ – непрочитанная часть входной цепочки; первый символ цепочки w находится под входной головкой; если $w = \epsilon$, то считается, что входная лента прочитана;
- $u \in \Gamma^*$ — содержимое магазина; самый левый символ цепочки u считается вершиной магазина; если $u = \epsilon$, то магазин считается пустым.

  Цепочка w **допускается** МП-автоматом, если $(q_0, w, Z_0) \mid -^* (q, \epsilon, u)$ для некоторых $q \in F$ и $u \in \Gamma^*$. Язык, **допускаемый** МП-автоматом – множество всех цепочек, допускаемых этим автоматом.

  В случае, когда цепочка w допускается МП-автоматом, если $(q_0, w, Z_0) \mid -^* (q, \epsilon, \epsilon)$ для некоторого $q \in Q$, говорят, что автомат допускает цепочку **опустошением магазина.**

  Утверждение: языки, порождаемые КС-грамматиками, и языки, допускаемые недетерминированными МП-автоматами, совпадают.

LL(k) и LR(k)

Контекстно-свободная грамматика называется $LL(1)$ **грамматикой** тогда и только тогда, когда выполняются следующие два условия:

- для каждого нетерминала, являющегося левой частью нескольких правил, $A \to \alpha_1|\alpha_2|\dots|\alpha_n$ необходимо, чтобы $FIRST(\alpha_i) \cap FIRST(\alpha_j) = \emptyset$, $i \neq j$;
- для каждого аннулирующего нетерминала $A \Rightarrow^* \$$ необходимо, чтобы $FIRST(A) \cap FOLLOW(A) = \emptyset$.

Грамматика, для которой таблицы анализа не имеют неоднозначно определённых входов, называется $LL(1)$.

Грамматика, для которой можно построить таблицу LR разбора, называется LR **грамматикой.**

Грамматика называется $LR(1)$, если из условий:

- $S' \Rightarrow_r uAw \Rightarrow_r uvw$;
- $S' \Rightarrow_r zBx \Rightarrow_r uvy$;
- $FIRST(v) = FIRST(y)$

следует, что $uAy = zBx$ (т.е. $u = z, A = B$ и $x = y$).

Конфигурация LR анализатора – это пара, первая компонента которой – содержимое стека, а вторая – непросмотренный вход: $(S_0X_1S_1X_2S_2\dots X_mS_m, A_iA_{i+1}\dots A_n\$)$. Эта конфигурация соответствует правой сентенциальной форме $X_1X_2\dots mA_iA_{i+1}\dots A_n$.

Замыкание множества $LR(1)$ ситуаций, допустимых для некоторого активного префикса z – это множество всех $LR(1)$ ситуаций, допустимых для этого префикса.

LR анализатор выполняет **4 типа действий:**

- `Shift`;
- `Reduce`;
- `Accept`;
- `Reject`.

При выполнении действия `Shift`, верхний символ входа переносится в магазин, а затем на верх магазина помещается состояние `action` (предыдущее состояние, взятый символ.

При выполнении действия `Reduce`, из стека убирается правая часть правила и добавляется левая и состояние `goto` (последнее состояние в магазине, символ из левой части правила).

В канонической системе множеств $LR(1)$ ситуаций может возникать **2 типа конфликтов:**

- `Shift - Reduce`;
- `Reduce - Reduce`.