Языки, автоматы и регулярные выражения. Лекция 1.

Александр Чуклин

Февраль 2010 года

Литература

Ахо А., Ульман Д. *Теория Синтаксического Анализа, Перевода и Компиляции* М.: Мир, 1978, Гл. 0, 2.

Хопкрофт Д., Мотвани Р., Ульман Д. *Введение в теорию автоматов, языков и вычислений.* М.,: Вильямс, 2002.

Ахо А., Сети Р., Ульман Д. *Компиляторы. Принципы, технологии, инструменты. (DRAGON BOOK)* Москва, Санкт-Петербург, Киев, 2003.

Пентус А.Е., Пентус М.Р. *Теория формальных языков. Учебное пособие.* Москва, 2004.

Слова, языки, операции над языками

Обозначения. Пусть Σ— некоторый конечный алфавит. Тогда Σ^*— множество всех слов в Σ. Пустое слово обозначается ε. Языком называется любое подмножество Σ^*. Пустой язык (не содержащий *никаких* слов) обозначается $\emptyset$.

С языками можно производить любые теоретико-множественные операции. Кроме того, вводится операция *конкатенации (или соединения).* По определению, если $L_1, L_2 \subseteq \Sigma^*$, то их соединение $L_1 \cdot L_2$ состоит из всех слов вида uv, $u \in L_1, v \in L_2$. *Итерация* языка L обозначается L^* и, по определению, равна: $L^* \stackrel{def}{=} \varepsilon + L + L \cdot L + L \cdot L \cdot L + \ldots$. Можно также ввести операции *правого* и *левого* деления. *Правое частное* языков определяется так: $L_1 \swarrow L_2 \stackrel{\text{def}}{=} \{x \,|\, \exists y \in L_2 : xy \in L_1\}$, т.е. множество префиксов слов из L_1, чьи суффиксы принадлежат L_2. *Левое частное* языков определяется так: $L_1 \searrow L_2 \stackrel{\text{def}}{=} \{y \,|\, \exists x \in L_2 : xy \in L_1\}$, т.е. множество суффиксов слов из L_1, чьи префиксы принадлежат L_2.

Декартово произведение (языков) обозначается $\times$.

Мощность множества L обозначается $|L|$.

Множество всех подмножеств множества L обозначается 2^L.

Метасимвол ? обозначает любой символ из Σ.

Число букв a в слове x обозначается $|x|_a$.

Регулярные языки и регулярные выражения.

Мы переходим к изучению *регулярных* или *рациональных* языков — базисного понятия теории формальных языков. Одновременно с регулярными языками вводятся и *регулярные выражения* — некоторый способ кодировки регулярных языков.

Определение 1 (Рег. выражения и рег. языки) *Пусть алфавит Σ не содержит символы $\emptyset, \varepsilon, \circ, +, (,), *$.*

Регулярными выражениями и обозначаемые ими регулярными языками в алфавите Σ называются следующие рекурсивные обекты:

1. *$\emptyset, \varepsilon$, а также буквы алфавита $a \in \Sigma$ — регулярные выражения, обозначающие, соответственно, регулярные языки. $\{\emptyset\}, \{\varepsilon\}$, и $\{a\}$.*
2. *Если $\mathcal{A}$ и $\mathcal{B}$ — регулярные выражения, обозначающие регулярные языки A и B, то регулярными будут также выражения $(\mathcal{A} \circ \mathcal{B})$, $(\mathcal{A} + \mathcal{B})$ и $(\mathcal{A}^*)$, которые обозначают, соответственно, регулярные языки $A \circ B$, $A \cup B$ и A^*.*
3. *Никаких других регулярных выражений и регулярных языков нет.*

Из определения видно, что любому регулярному выражению R однозначно сопоставляется корневое ориентированное дерево $T(R)$, в котором все дуги ориентированы по направлению к корню, а полустепень входа каждой вершины v не более 2. При этом

- если $indeg(v) = 0$ (т.е. v — лист), то v помечена либо буквой алфавита Σ, либо одним из символов $\emptyset, \varepsilon$;
- если $indeg(v) = 1$, то v помечена символом $*$;
- если $indeg(v) = 2$, то v помечена одним из символов $\circ, +$.

Неформально дерево $T(R)$ задает поток слов в алфавите Σ, так что каждой вершине приписывается некоторый язык. Поток инициализируется в листьях, а затем в каждой внутренней вершине v пришедшие туда по дугам потоки преобразуются в соответствии с пометкой v, знак $+$ (или $|$) интерпретируется как объединение, знак $\circ$ (или пустое место)— как конкатенация и знак $*$ как *итерация.* Напомню, последняя операция над языком $L \subset \Sigma^*$ определяется так: $L^* \stackrel{def}{=} \varepsilon + L + L^2 + \ldots$.

Корню дерева отвечает регулярный язык $\tilde{R}$.

По определению, регулярное выражение R *задает* или *обозначает* язык $\tilde{R}$.

Например, регулярное выражение $((a+b)^*)$ задает множество всех слов из a и b; регулярное выражение $(a \circ (a \circ a)^*)$ задает множество всех цепочек из a нечетной длины[1] . Как правило, можно использовать регулярные выражения для обозначения языков ими задаваемых. Но в специальных случаях (если существенно различие между выражением и множеством) язык, заданный регулярным выражением R, обозначается $L(R)$.

Определение 2 (Равенство регулярных выражений) *Два регулярных выражения равны $(=)$, если равны задаваемые ими регулярные языки.*

Исходя из определения, просто проверить следующие свойства регулярных выражений.

[1] В дальнейшем в записи знак $\circ$, а также некоторые пары скобок будут опускаться, а вместо символа $+$ часто будет использоваться $|$.

Пусть α, β, γ– регулярные выражения. Тогда

$$\begin{array}{rclcrcl}
\alpha+\beta & = & \beta+\alpha; & & \alpha+\alpha & = & \alpha; \\
(\alpha+\beta)+\gamma & = & \alpha+(\beta+\gamma) & & \alpha+\emptyset & = & \alpha \\
\alpha(\beta+\gamma) & = & \alpha\beta+\alpha\gamma & & (\alpha+\beta)\gamma & = & \alpha\gamma+\beta\gamma \\
\alpha\varepsilon & = & \alpha & & \emptyset\alpha & = & \emptyset \\
\alpha+\emptyset & = & \alpha & & \alpha^* & = & \alpha+\alpha^* \\
\emptyset^* & = & \varepsilon & & (\alpha^*)^* & = & \alpha^*
\end{array}$$

Таким образом, регулярные языки ведут себя относительно операций объединения и конкатенации почти как обычные числа (конкатенация, правда, некоммутативна); роль нуля играет $\emptyset$, а роль единицы — ε.

Применение регулярных выражений

Регулярные выражения (regexp'ы) являются основным средством для поиска образцов (шаблонов) в тексте. На практике регулярное выражение компилируются в детерминированный конечный автомат (ДКА)[2], который затем моделируется для получения программы. Мы рассмотрим два основных приложения регулярных выражений: *лексический анализатор* и *поиск в тексте*.

Для начала посмотрим как выглядят РВ в реальной жизни на примере системы UNIX [3]. Следует отметить, что РВ в UNIX содержит много расширений, которые позволяют распознать нерегулярные языки. Этого вопроса мы касаться не будем, но рассмотрим другие отличия:

1. Символ . (точка) обозначает "любой символ".
2. Последовательность $[a_1a_2...a_k]$ означает $(a_1|a_2|...|a_k)$.
3. $[0-9]$ означает то же, что и $[0123456789]$.
4. Оператор ? значит "ни одного или один раз"$(R? = \varepsilon+R)$.
5. Оператор + значит "один или несколько раз"$(R+ = RR^*)$.

Есть ещё некоторый набор обозначений, используемый в первую очередь для упрощения записи регулярного выражения (представьте, если бы мы пользовались стандартными обозначениями в реальной жизни).

Поиск в тексте по шаблону реализован во многих программах и библиотеках. Примером может служить команда *grep* в UNIX-подобных операционных системах.Подробную справку по этой команде и регулярным выражениям в Linux можно получить, набрав в терминале команду *man grep*.

Лексический анализатор — компонент компилятора, который сканирует исходную программу и распознаёт все *лексемы*, т.е. подцепочки последовательных символов, логически составляющих единое целое. В качестве примера рассмотрим генератор лексического анализатора *lex*, применяемый в компиляторах. Он получает на вход список регулярных выражений в стиле UNIX, за каждым из которых следует код, указывающий, что должен делать анализатор при обнаружении экземпляра лексемы. Затем lex преобразует РВ в ДКА для того, чтобы генерировать эффективную функцию, разбивающую исходную программу на лексемы. Ниже приведён пример входных данных для программы lex:

```
else                    {возвращает (ELSE);}
[A-Za-z][A-Za-z0-9]     {код для помещения
                        найденного идентификатора
                        в таблицу символов;
                        возвращает (ID);}
>=                      {возвращает (GE);}
=                       {возвращает (EQ);}
...
```

[2] об этом в следующей лекции

[3] Заметим, что regexp'ы **в командной строке** UNIX и DOS существенно отличаются от указанных.

Языки, автоматы и регулярные выражения. Лекция 2.

Александр Чуклин

Февраль 2010 года

Регулярные языки. Конечные автоматы.

Напомним определение регулярного языка:

Определение (Рег. выражения и рег. языки) *Пусть алфавит Σ не содержит символы $\emptyset, \varepsilon, \circ, +, (,), *$. Регулярными выражениями и обозначаемые ими регулярными языками в алфавите Σ называются следующие рекурсивные объекты:*

i. $\emptyset, \varepsilon$, а также буквы алфавита $a \in \Sigma$ — регулярные выражения, обозначающие, соответственно, регулярные языки. $\{\emptyset\}, \{\varepsilon\}$, и $\{a\}$.

ii. Если $\mathcal{A}$ и $\mathcal{B}$ — регулярные выражения, обозначающие регулярные языки A и B, то регулярными будут также выражения $(\mathcal{A} \circ \mathcal{B})$, $(\mathcal{A}+\mathcal{B})$ и $(\mathcal{A}^*)$, которые обозначают, соответственно, регулярные языки $A \circ B$, $A \cup B$ и A^*.

iii. Никаких других регулярных выражений и регулярных языков нет.

Определим еще один важнейший персонаж.

Определение (Конечный автомат) *(Недетерминированным конечным) автоматом называется пятерка $A = (Q, \Sigma, \delta, q_0, F)$, где*
(i) Q — конечное множество состояний *управляющего устройства (УУ); (ii) Σ — входной алфавит;*

δ — отображение $Q \times \Sigma$ в множество всех подмножеств 2^Q, называемое функцией переходов УУ; (iii) q_0 — начальное состояние УУ; (iv) $F \subseteq Q$ — множество финальных состояний.

На вход УУ подается слово $w = a_1 a_2 \dots a_n$, записанное на *входной ленте*. Автомат обрабатывает его слева направо по *тактам*. Если автомат находится в текущем состоянии q и читает текущий символ a входного слова w, то такт состоит из (недетерминированной) смены состояния по формуле $q_{new} \in \delta(q, a)$ и перехода к следующему символу a_{i+1} на входной ленте.

Упомянем также две модификации определения КА. В *КА с ε-тактами* состояние может изменяться *без чтения входного символа*, т.е. $\delta(\cdot, \cdot) : Q \times (\Sigma \cup \varepsilon) \to 2^Q$.

Назовем *конфигурацией* пару $(q, w) \in Q \times \Sigma^*$. На множестве конфигураций введем соответствующее тактам работы автомата бинарное отношение $\vdash$: для всех $q' \in \delta(q, a)$ и для всех $w \in \Sigma^*$ $(q, aw) \vdash (q', w)$. Рефлексивное и транзитивное замыкание отношения $\vdash$ обозначим $\vdash^*$

Определение (Автоматный язык) *Автомат A* принимает (распознает, допускает) *слово $w \in \Sigma^*$, если $(q_0, w) \vdash^* (q, \varepsilon)$, $q \in F$.*

Автоматный язык $L(A)$ состоит из всех слов, принимаемых автоматом A, т.е. $L(A) = \{w | w \in \Sigma^*, (q_0, w) \vdash^* (q, \varepsilon)$ для некоторого $q \in F\}$.

Автоматы A_1 и A_2 называются эквивалентными, если принимаемые ими языки совпадают $L(A_1) = L(A_2)$

КА удобно изображать графически посредством т.н. *диаграммы Мура*. Построим на множестве состояний Q, как на вершинах, ориентированный помеченный *граф (переходов)* Γ (возможно имеющий петли и кратные дуги). Правило построения дуг такое: для любой тройки (q, a, q') такой, что $q' \in \delta(q, a)$, проводится ориентированная дуга от q к q' и помечается символом a. Примеры диаграмм приведены в задаче N 1 и доп. задачах NN 1, 4.

Сопоставим любому ориентированному пути в графе Γ слово на дугах. Тогда предыдущее определение говорит, что слово w принимается автоматом A, если в графе $\Gamma(A)$ существует ориентированный путь, начинающийся в вершине q_0 и заканчивающийся в одной финальных вершин, который сопоставлен[1] w. Таким образом, КА можно использовать для представления путей в помеченном орграфе и, как мы увидим, многие алгоритмы в теории автоматов имеют прямые графические аналогии.

Вопрос 1 *Постройте автоматы и их диаграммы, принимающие следующие языки в алфавите $\Sigma = \{a, b\}$*

(i) $L_{\neg(bb)}$ состоит из всех слов, в которых нет двух соседних букв b.

(ii) $L_{a?b}$ состоит из всех слов, содержащих подслово $a?b$ (напомню, что метасимвол ? обозначает любой символ из Σ).

Определение (Детерминированный автомат) *Автомат A называется* детерминированным, *если для любых $q \in Q, a \in \Sigma$ множество $\delta(q, a)$ содержит не более одного состояния. Если множество $\delta(q, a)$ всегда содержит точно одно состояние, то A называется* полным. *(В графе переходов детерминированного автомата A не может быть двух кратных дуг, помеченных одинаковой буквой. У полного автомата дополнительно возможен переход из любого состояния по любому входному символу.)*

[1] Подчеркнем, что финальные вершины не являются *поглощающими*. Требуется только, чтобы ориентированный путь, отвечающий допустимому слову, начинался в q_0 и *заканчивался* в какой-то финальной вершине.

Определение (ПДКА) *Детерменированный конечный автомат называется полным (ПДКА), если из любого состояния есть переход по каждому входному символу.*

Вопрос 2 *Как по произвольному ДКА построить полный ДКА?*

Теорема 1 (Эквивалентности) *Следующие классы языков совпадают:*

i. регулярные языки;

ii. языки, принимаемые НКА;

iii. языки, принимаемые ДКА;

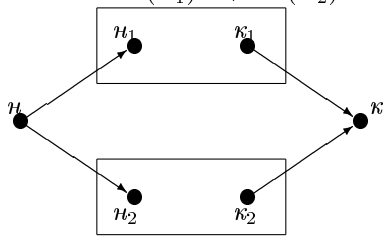
Доказательство 1 *теоремы проводится путем моделирования одного класса посредством другого*[2].

i → *ii*

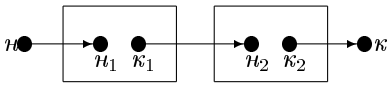
Поскольку любой регулярный язык задается регулярным выражением, достаточно промоделировать индуктивное построение регулярных выражений с помощью недетерминированных конечных автоматов.

Легко построить автоматы, принимащие, соответственно, пустой язык $\emptyset$, пустое слово ε и отдельные буквы входного алфавита. Теперь для доказательства достаточно научится моделировать операции сложения $+$, конкатенации $\circ$ и итерации $*$. Схематически проиллюстрируем как, имея конечные автоматы A_1 и A_2, принимающие, соответственно, языки $L(A_1)$ и $L(A_2)$ (начальные и конечные состояния которых помечены буквами H и K, соответственно), построить конечные автоматы $A_+, A_\circ, A_*$.

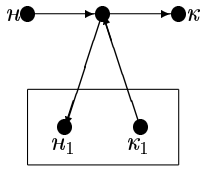
i. Конечный автомат A_+, принимащий язык $L(A_1) + L(A_2)$. . Для этого



ii. Конечный автомат $A_\circ$, принимащий язык $L(A_1) \circ L(A_2)$.



iii. Конечный автомат A_*, принимащий язык $L(A_1)^*$.



Заполните детали доказательства самостоятельно. Помните, что рассматриваемые нами конечные автоматы не имеют *переходов по пустому символу!*

Построим регулярное выражение, задающее язык, принимаемый конечным автоматом A. Перенумеруем состояния от 1 до k. Пусть 1– начальное, а k– финальное состояния. Обозначим $D_{i,j,T} (1 \le i,j \le k, 0 \le T \le k)$– множество всех слов, которые отвечают всем путям в графе $\Gamma(A)$, стартующим из состояния i, заканчивающимся в состоянии j и использующим в качестве промежуточных только состояния из множества $\{1,2,\dots,T\}$. По определению, $L(A) = D_{1,k,k}$.

Покажем теперь индукцией по T регулярность множеств $D_{i,j,T}$ при всех i,j. База индукции для $T = 0$ очевидна, т.к. промежуточные состояния запрещены и допустимы только прямые переходы, т.е. буквы. Индукционный переход вытекает из следующего соотношения

$$D_{i,j,T+1} = D_{i,j,T} + D_{i,T+1,T} D^*_{T+1,T+1,T} D_{T+1,j,T}$$

(достаточно отметить моменты, когда путь попадает в состояние $T+1$)[3].

Сведение **iii** → **ii** *очевидно.*

ii → **iii** **(subset construction)**

Пусть $A = (Q, \Sigma, \delta, q_0, F)$– недетерминированный автомат. Построим детерминированный автомат $\tilde{A} = (\tilde{Q}, \Sigma, \tilde{\delta}, \tilde{q}_0, \tilde{F})$ по следующим правилам:

i. $\tilde{Q} = 2^Q$;

ii. для любого $P \in 2^Q$ и $a \in \Sigma$ положим $\tilde{\delta}(P,a) = P'$, где $P' = \cup_{q \in P} \delta(q,a)$;

iii. $\tilde{q}_0 = \{q_0\} \in 2^Q$;

iv. $F' \in \tilde{F}$ тогда и только тогда, когда $F' \cap F \neq \emptyset$.

Покажем, что $L(A) \subseteq L(\tilde{A})$. Рассмотрим произвольное слово $w = a_1 a_2 \dots a_n \in L(A)$. По определению автомата A, ему отвечает некоторая принимающая последовательность конфигураций $(q_0, w) \vdash (q_1, a_2 a_3 \dots a_n) \vdash \dots \vdash (q_n, \varepsilon)$, $q_n \in F$. Теперь по построению автомата $\tilde{A}$ существует последовательность конфигураций для $\tilde{A}$ $(q_0, w) \vdash (Q_1, a_2 a_3 \dots a_n) \vdash \dots \vdash (Q_n, \varepsilon)$. Поскольку $q_n \in Q_n$, то $Q_n \in \tilde{F}$ и, следовательно, это будет принимающая последовательность конфигураций для $\tilde{A}$ и $w \in L(\tilde{A})$.

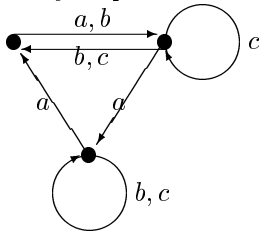
Теперь покажем, что $L(\tilde{A}) \subseteq L(A)$. Рассмотрим произвольное слово $w = a_1 a_2 \dots a_n \in L(\tilde{A})$. Следовательно,

[2] Более подробное изложение можно посмотреть, например, в книгах из списка литературы.

[3] Приведенное рассуждение аналогично обоснованию алгоритма Флойда, для вычисления длины кратчайшего пути во взвешенном графе без отрицательных циклов.

существуют принимающая последовательность конфигураций автомата $\tilde{A} : (\{q_0\}, w) \vdash (Q_1, a_2a_3 \dots a_n) \vdash \dots \vdash (Q_n, \varepsilon)$, $Q_n \in \tilde{F}$ *и состояние* $q_n \in Q_n \cup F$. *По построению функции переходов* $\tilde{\delta}$, *существует состояние* $q_{n-1} : \delta(q_{n-1}, a_n) = q_n$. *Аналогично существует состояние* $q_{n-2} \in Q : \delta(q_{n-2}, a_{n-2}) = q_{n-1}$ *и т.д. Таким образом, существует принимающая последовательность конфигураций для слова* w *в автомате* A *и* $w \in L(A)$.

Вопрос 3 *Доказательство предыдущего утверждения построено так, что платой за устранение индетерминизма является, вообще говоря, экспоненциальное увеличение "памяти"– числа состояний. Возникает вопрос, можно ли понизить эту верхнюю оценку? Оказывается, что нет. На рисунке приведен пример недетерминированного автомата в трехбуквенном входном алфавите* $\Sigma = \{a, b, c\}$, *имеющего* $n = 3$ *состояния и такого, что любой эквивалентный ему детерминированный автомат имеет не менее* $2^n = 8$ *состояний. Докажите это утверждение.*



Обобщите этот пример для $n > 3$.

Ответ 1 *Заметим, что примеры автоматов подобного рода существуют и для двухбуквенного входного алфавита (см. задачи) Для однобуквенного алфавита оценка несколько ниже* $2^{\sqrt{n \log n}}$

Из теоремы эквивалентности немедленно вытекает следующее важнейшее утверждение.

Теорема 2 *Если* L*– регулярный язык, то регулярным будет и дополнительный язык* $\bar{L} = \Sigma^* \setminus L$.

Доказательство 2 *Пусть* $A = (Q, \Sigma, \delta, q_0, F)$*– полный детерминированный автомат, для которого* $L = L(A)$. *Тогда автомат* $\tilde{A} = (Q, \Sigma, \delta, q_0, Q \setminus F)$ *определяет язык* $\overline{L(A)}$.

Из замкнутости относительно дополнения следует, что регулярным является не только объединение, но и пересечение регулярных языков L_1 и L_2 (поскольку $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$). Поэтому регулярные языки образуют *булеву алгебру* (т.е. ведут себя по отношению к теоретико-множественным операциям как подмножества некоторого множества). Более того, верна следующая теорема :

Теорема 3 (Клини) *Семейство регулярных множеств в алфавите* Σ *есть наименьшее семейство подмножеств множества* Σ^*, *содержащих конечные множества и замкнутое относительно объединения* $\cup$, *конкатенации* $\circ$ *и итерации* $*$

Вопрос 4 *Докажите эту теорему.*

Лемма о разрастании (pumping lemma)
Ей удобно пользоваться для доказательства НЕрегулярности языка

Теорема 4 (Лемма о разрастании (ЛР)) *Пусть* L*– регулярное множество. Существует такая константа* C, *что при* $w \in L$, $|w| \geq C$ *слово* w *представимо в виде* $w = xyz$, *причем* $|xy| \leq C$ *и* $0 < |y|$, *и для всех* $i \geq 0$, $xy^iz \in L$.

Приведен **сильный вариант** *ЛР, часто бывает достаточен* **слабый вариант**, *где требуют выполнения неравенства* $0 < |y| \leq C$.

В качестве примера использования леммы приведем стандартное доказательство нерегулярности языка $L = \{0^n1^n \mid n = 0, 1 \dots\}$.

Предположим, что L — регулярный. Тогда он удовлетворяет ЛР с некоторой константой, скажем, p. Возьмем слово $w = 0^p1^p \in L$. Его длина больше, чем p и, следовательно, существует такое разбиение $w = xyz$, что $0 < |y| \leq C$ и для всех $i \geq 0$, $xy^iz \in L$. Возможны только следующие случаи **i.** $y \in 0^+$, тогда для слова $xyyz$ нарушается баланс нулей и единиц. **ii.** $y \in 1^+$, опять для слова $xyyz$ нарушается баланс. **iii.** y содержит нули и единицы, тогда в слове z нарушится порядок следования нулей и единиц. Замечу, что в данном случае нам повезло и мы обнаружили все противоречия в слове $xyyz$.

Поскольку ЛР не является критерием регулярности, то её нельзя использовать для проверки регулярности L

Сформулируем **критерий регулярности** в терминах, аналогичных ЛР.

Теорема 5 (Jaffe (1962)) *Язык* $L \subset \Sigma^*$ *регулярен тогда и только тогда, когда существует такая константа* $C \geq 0$, *что для всех* $y \in \Sigma^*$, $|y| = C$ *существует такое разбиение* $y = uvw$, $v \neq \varepsilon$, *что* **для всех** $z \in \Sigma^*$ **и** $i \geq 0$ **выполняется** $yz \in L \Leftrightarrow uv^iwz \in L$.