ТЕОРИЯ АВТОМАТОВ И ФОРМАЛЬНЫХ ЯЗЫКОВ (ТАиФЯ)

Семестр	Текущая аттестация	Промежуточная аттестация
3 (осень 2024)	Лабораторные работы (6 работ)	Экзамен

Автомат — устройство, которое работает по программе и выполняет поставленные перед ним задачи без непосредственного участия человека.

В дисциплине ТАиФЯ будем изучать абстрактные автоматы, которые могут вычислять.

Вычислять можно функции.

Декартовым произведением $A_1 \times A_2 \times \ldots \times A_n$ *множеств* A_1, A_2 … и A_n называется множество $\{(a_1, a_2, \ldots, a_n) \mid a_1 \in A_1, a_2 \in A_2, \quad , a_n \in A_n\}$, т. е.

$A_1 \times A_2 \times \ldots \times A_n = \{(a_1, a_2, \ldots, a_n) \mid a_1 \in A_1, a_2 \in A_2, \quad , a_n \in A_n\}$,

а A^n обозначает $A \times A \times \ldots \times A$ (n раз).

Функцией, отображающей множество X во множество Y (F: $X \rightarrow Y$), называется множество $F \subseteq X \times Y$, в котором не найдется хотя бы двух пар, в которых первые элементы равны.

Множество X называется *областью отправления*,

а множество Y — *областью прибытия*.

Множество $\{x \mid (x, y) \in F\}$ — *область определения* функции F (или множество значений ее аргумента);

множество $\{y \mid (x, y) \in F\}$ — *область значений* функции F.

Если область определения равна области отправления, то функция называется *полностью (всюду) определенной*, иначе — *частичной*.

Функцию F: $X \rightarrow Y$ называют *n-местной функцией* над множеством A,

если $Y = A$ и $X = A^n$.

Предикатом называют функцию, областью значений которой является множество $\{0, 1\}$.

Алфавитом называют непустое конечное множество символов.

Например,

$V_1 = \{a, b\}$,

$V_2 = \{0, 1\}$,

$V_3 = \{a, +, 1, =\}$ – алфавиты.

Словом в алфавите V называется конечный объект, получаемый выписыванием одного за другим символов V,

например,

$a + 1 = 1 + a$ — слово в алфавите V_3,

101011 — слово в алфавите V_2,

abaab — слово в алфавите V_1.

Длина слова — число символов в нем, *пустое слово* не содержит ни одного символа.

Множество всех слов в алфавите V обозначается V^*.

n-местной словарной функцией над алфавитом V называют n-местную функцию над V^*, т. е. функцию из $V^* \times V^* \times \ldots \times V^*$ (n раз) в V^*.

Функция F называется *вычислимой*, если существует алгоритм нахождения ее значения $F(x)$ по значению аргумента x.

Для любой вычислимой функции существует автомат, который вычисляет её значение по значению аргумента — машина Тьюринга.

Машина Тьюринга — «сложный» автомат (например, может зациклиться или выполнять очень много действий в процессе вычислений, сложно или невозможно определить свойства машины Тьюринга), пригодный для вычисления любых вычислимых словарных функций.

В множестве всех вычислимых функций можно выделить классы функций, для вычисления которых можно использовать более простые автоматы — например конечные автоматы или автоматы с магазинной памятью.

Формальный язык — это множество *цепочек*. Множество цепочек может быть конечным или бесконечным.

Цепочка – это конечная последовательность символов, которая может заканчиваться *концевым маркером* (┤). *Пустая цепочка* (обозначается ε) не содержит ни одного символа.

Алфавит языка — это конечное множество символов, из которых могут быть образованы цепочки языка.

Одна из важных задач теории формальных языков является задача *распознавания* — определения, принадлежит ли заданная цепочка заданному языку.

Характеристической функцией множества M называется предикат
F_M: $V^* \to \{0, 1\}$,
всюду определенный на V^*:
$F_M(a) = 1$, если $a \in M$, и $F_M(a) = 0$, если $a \notin M$.

Задача распознавания сводится к вычислению характеристической функции множества, являющимся формальным языком.

Для вычисления характеристической функции формального языка (решения задачи распознавания) используются различные автоматы.

Основным применением теории автоматов и формальных языков является разработка трансляторов языков программирования и других программ обработки символьной информации и проектирование цифровых устройств.

Рекомендуемая литература

1. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции. – М.: Мир, 1978. – Т. 1, 612 с. Т.2, 487 с.
2. Ахо А., Сети Р., Ульман Дж. Компиляторы: принципы, технологиии и инструменты.: Пер. с англ. – М.: Издательский дом “Вильямс”, 2003. – 768 с.
3. Льюис Ф., Розенкранц Д., Стирнз Р. Теоретические основы проектирования компиляторов. – СПб.: Питер, 1997. – 625 с.
4. Хопкрофт, Д. Э. Введение в теорию автоматов, языков и вычислений : пер. с англ. / Д. Э. Хопкрофт, Р. Мотвани, Д. Д. Ульман. - 2-е изд. - М. : Вильямс, 2002. - 527 с.

1. Бикмуллина, И. И. Теория формальных грамматик и автоматов : учебное пособие / И. И. Бикмуллина, И. А. Барков. - Казань : КНИТУ-КАИ, 2021. - 272 с. - URL: https://e.lanbook.com/book/264845.

Формальный язык — это множество *цепочек*. Множество цепочек может быть конечным или бесконечным.

Цепочка – это конечная последовательность символов, которая может заканчиваться *концевым маркером (┤)*. *Пустая цепочка* (обозначается ε) не содержит ни одного символа.

Алфавит языка — это конечное множество символов, из которых могут быть образованы цепочки языка.

Способы задания языков

1. Формальный язык, представляющий собой конечное множество цепочек, может быть задан перечислением всех цепочек.

Пример.

L1={a, ab, ac}

2. Словесным описанием.

Пример.

Цепочки, содержащие не более двух символов, начинающиеся символом a , за которым может быть только символ b или символ c.

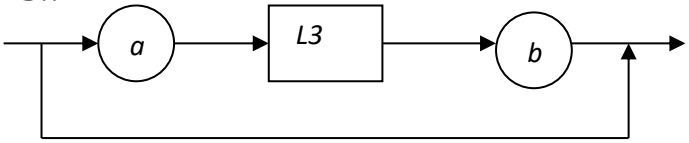
Цепочки, начинающиеся последовательностью символов a , за которой следует последовательность символов b .

3. Заданием характеристического свойства.

$L2=\{a^n b^m \mid n>0, m>0\}$

4. Синтаксической диаграммой.

L3::



5. Формулами Бэкуса-Наура (БНФ)

<Число> ::= <Цифра><Число>|<Цифра>

<Цифра> ::= 0|1|2|3|4|5|6|7|8|9

6. Расширенными формулами Бэкуса-Наура (БНФ)

$<\text{Число}> ::= \{<\text{Цифра}>\}^+$

<Цифра> ::= 0 – 9

6. Автоматом, распознающим цепочки языка.

7. Формальной грамматикой.

Операции над языками

1. *Включение L1 в L2* ($L1 \subseteq L2$ или $L2 \supseteq L1$) истинно, если каждая цепочка языка L1 принадлежит языку L2.

2. Языки L1 и L2 *равны* (L1=L2), если они состоят из одних и тех же цепочек, или если $L1 \subseteq L2$ и $L2 \subseteq L1$.

3. *Строгое включение L1 в L2* ($L1 \subset L2$ или $L2 \supset L1$) истинно, если $L1 \subseteq L2$ и $L1 \neq L2$.

4. *Объединение L1 и L2* ($L1 \cup L2$) есть язык, состоящий из всех тех и только тех цепочек, которые принадлежат L1 или L2, т.е.

$L1 \cup L2 = \{x \mid x \in L1 \text{ или } x \in L2\}$.

5. *Пересечение L1 и L2* ($L1 \cap L2$) есть язык, состоящий из всех тех и только тех цепочек, которые принадлежат каждому из языков L1 и L2, т.е.

$L1 \cap L2 = \{x \mid x \in L1 \text{ и } x \in L2\}$.

6. *Разность L1 и L2* (L1–L2) есть язык, состоящий из всех тех и только тех цепочек языка L1, которые не принадлежат языку L2, т.е.

$L1 - L2 = \{x \mid x \in L1 \text{ и } x \notin L2\}$.

7. *Конкатенация L1 и L2* (L1L2) есть язык, цепочки которого представляют собой цепочки языка L1, к которым приписаны цепочки языка L2, т.е.

$L1L2 = \{xy \mid x \in L1 \text{ и } y \in L2\}$.

8. *Итерация L* (L^*) определяется следующим образом:

1) $L^0 = \{\varepsilon\}$

2) $L^n = L\,L^{n-1}$, $n > 0$

3) $L^* = \cup\, L^n$, $n \geq 0$

9. *Позитивная итерация L* (L^+) — это язык, представляющий собой итерацию языка и несодержащий пустой цепочки, если $\varepsilon \notin L$, т.е.

$$L^+ = \cup\, L^n,\ n \geq 1 \quad \text{или} \quad L^+ = L\,L^* = L^*\,L$$

Универсальный язык U в алфавите *A* равен итерации алфавита ($U = A^*$) и представляет собой множество всех цепочек, которые можно составить из символов алфавита, включая ε.

10. *Дополнение L* до универсального языка U ($\overline{L}$) есть язык, состоящий из всех тех и только тех цепочек языка U, которые не принадлежат языку L, т.е.

$\overline{L} = U - L$

11. *Обращение языка L* (L^{-1}) — это язык, представляющий собой множество цепочек языка L , записанных в обратном порядке.

Формальные грамматики. Выводы

Формальная грамматика G =(N,T,P,S), где

N – конечное множество нетерминальных символов (нетерминалов),

T – конечное множество терминальных символов (терминалов),

P – конечное множество правил (продукций) вида $\alpha \to \beta$, где α и β - последовательности терминальных и нетерминальных символов,

S – начальный нетерминал.

Формальная грамматика G порождает цепочки языка L(G), поэтому она называется порождающей. Множество T терминальных символов грамматики представляет собой алфавит языка L(G). Если T состоит только из строчных букв, то в качестве элементов N будем использовать заглавные буквы, иначе элементы N (нетерминалы) будем заключать в угловые скобки.

Цепочка, выводимая в грамматике G = (N,T,P,S) , удовлетворяет следующим условиям:

1) S — выводимая цепочка;

2) если цепочка $\alpha\beta\gamma$ — выводимая, и $\beta \to \delta \in P$, то $\alpha\delta\gamma$ — выводимая (обозначается $\alpha\beta\gamma \Rightarrow \alpha\delta\gamma$, читается “$\alpha\delta\gamma$ непосредственно выводится из $\alpha\beta\gamma$”)

Если $\alpha_1 \Rightarrow \alpha_2 \Rightarrow \ldots \Rightarrow \alpha_n$, то α_n — цепочка, выводимая из α_1 , обозначается $\alpha_1 \Rightarrow {}^{*}\alpha_n$.

Терминальная цепочка – это цепочка, выводимая в грамматике G и не содержащая нетерминалов — принадлежит языку L(G).

Промежуточная цепочка (сентенциальная форма вывода) – это цепочка, выводимая в грамматике G и содержащая нетерминалы.

Классификация формальных грамматик по Хомскому

Формальные грамматики классифицируются по виду правил.

Тип 0. Грамматики общего вида. Без ограничений.

Тип 1. Контекстно-зависимые грамматики.

$\alpha A \beta \to \alpha \gamma \beta$ $\quad \alpha,\beta \in (T \cup N)^*, \gamma \in (T \cup N)^+$

Неукорачивающие грамматики.
$\alpha \to \beta$ $\quad |\alpha| \le |\beta|$

Тип 2. Контекстно-свободные грамматики.

$A \to \alpha$ $\quad \alpha \in (T \cup N)^*$

Тип 3. Регулярные грамматики.

Левосторонние		Правосторонние
$A \to B\alpha$	$\alpha \in T^*$	$A \to \alpha B$
$A \to \alpha$		$A \to \alpha$

Классификация формальных языков по Хомскому

Язык регулярный, если может быть описан регулярной грамматикой.

Язык контекстно-свободный, если может быть описан контекстно-свободной грамматикой, но не может быть описан регулярной грамматикой.

Язык контекстно-зависимый, если может быть описан контекстно-зависимой (или неукорачивающей) грамматикой, но не может быть описан контекстно-свободной или регулярной грамматикой.

Язык общего вида, если не может быть описан ни контекстно-ависимой грамматикой, ни контекстно-свободной, ни регулярной.

Примеры грамматик и языков различных типов

Тип 0. Язык общего вида $L=\{\alpha\alpha \mid \alpha\in\{a,b\}^*\}$

1. $S \to CD$
2. $C \to aCA$
3. $C \to bCB$
4. $AD \to aD$
5. $BD \to bD$
6. $Aa \to aA$
7. $Ab \to bA$
8. $Ba \to aB$
9. $Bb \to bB$
10. $C \to \varepsilon$
11. $D \to \varepsilon$

$$\underset{1}{\underline{S}} \Rightarrow \underset{2}{\underline{C}}D \Rightarrow a\underset{3}{\underline{C}}AD \Rightarrow abCB\underset{4}{\underline{AD}} \Rightarrow abC\underset{8}{\underline{Ba}}D \Rightarrow abCa\underset{5}{\underline{BD}} \Rightarrow$$

$$ab\underset{10}{\underline{C}}abD \Rightarrow abab\underset{11}{\underline{D}} \Rightarrow abab$$

Тип 1. Контекстно-зависимый язык $L=\{a^nb^nc^n \mid n>0\}$

Неукорачивающая грамматика

1. $S \to aSBC$
2. $S \to abC$
3. $CB \to BC$
4. $bB \to bb$
5. $bC \to bc$
6. $cC \to cc$

$$\underset{1}{\underline{S}} \Rightarrow a\underset{2}{\underline{S}}BC \Rightarrow aa\underset{5}{\underline{bC}}BC \Rightarrow aabcBC$$

$$\underset{1}{\underline{S}} \Rightarrow a\underset{2}{\underline{S}}BC \Rightarrow aab\underset{3}{\underline{CB}}C \Rightarrow aa\underset{4}{\underline{bB}}CC \Rightarrow aab\underset{5}{\underline{bC}}C \Rightarrow aabb\underset{6}{\underline{cC}} \Rightarrow aabbcc$$

Контекстно-зависимая грамматика

1. $S \to aSBC$
2. $S \to abC$
3.1. $CB \to CD$
3.2. $CD \to BD$
3.3. $BD \to BC$
4. $bB \to bb$
5. $bC \to bc$
6. $cC \to cc$

$$\underset{1}{\underline{S}} \Rightarrow a\underset{2}{\underline{S}}BC \Rightarrow aab\underset{3.1}{\underline{CB}}C \Rightarrow aab\underset{3.2}{\underline{CD}}C \Rightarrow aab\underset{3.3}{\underline{BD}}C \Rightarrow aa\underset{4}{\underline{bB}}CC \Rightarrow$$

$$aab\underset{5}{\underline{bC}}C \Rightarrow aabb\underset{6}{\underline{cC}} \Rightarrow aabbcc$$

Тип 2. Контекстно-свободный язык $L=\{a^n b^n \mid n\geq 0\}$

1. $S \to aSb$
2. $S \to \varepsilon$

$S \underset{1}{\Rightarrow} aSb \underset{1}{\Rightarrow} aaSbb \underset{1}{\Rightarrow} aaaSbbb \underset{2}{\Rightarrow} aaabbb$

Тип 3. Регулярный язык $L=\{a^n b^m \mid n\geq 0, m\geq 1\}$

Правосторонняя рамматика

1. $S \to aS$
2. $S \to bA$
3. $A \to bA$
4. $A \to \varepsilon$

$S \underset{2}{\Rightarrow} bA \underset{3}{\Rightarrow} bbA \underset{3}{\Rightarrow} bbbA \underset{4}{\Rightarrow} bbb$

$S \underset{1}{\Rightarrow} aS \underset{1}{\Rightarrow} aaS \underset{2}{\Rightarrow} aabA \underset{4}{\Rightarrow} aab$

Левосторонняя рамматика

1. $S \to Sb$
2. $S \to Ab$
3. $A \to Aa$
4. $A \to \varepsilon$

$S \underset{1}{\Rightarrow} Sb \underset{1}{\Rightarrow} Sbb \underset{2}{\Rightarrow} Abbb \underset{4}{\Rightarrow} bbb$

$S \underset{2}{\Rightarrow} Ab \underset{3}{\Rightarrow} Aab \underset{3}{\Rightarrow} Aaab \underset{4}{\Rightarrow} aab$

Контекстно-свободные грамматики (КС-грамматики)

КС-грамматика – это класс формальных грамматик, правила которых имеет вид $B \to \alpha$, где $B \in N$, т.е. левая часть каждого правила представляет собой только один нетерминал, а на правую часть правил ограничений нет. Правила грамматики будем нумеровать. Пример КС-грамматики G1:

1. $S \to aABc$
2. $S \to \varepsilon$
3. $A \to cSB$
4. $A \to Ab$
5. $B \to bB$
6. $B \to a$

Вывод в КС-грамматике начинается с начального нетерминала. В процессе вывода один из нетерминалов промежуточной цепочки заменяется правой частью правила, левая часть которого представляет собой заменяемый нетерминал. Будем говорить, что выполняется замена нетерминала по правилу. При записи вывода под заменяемыми нетерминалами будем записывать номер правила, по которому выполняется замена.

Пример. Вывод терминальной цепочки в грамматике G1:

$S \Rightarrow aABc \Rightarrow aAbBc \Rightarrow acSBbBc \Rightarrow acSabBc \Rightarrow acabBc \Rightarrow acabac$

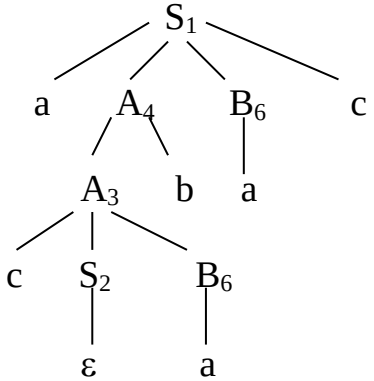
1 4 3 6 2 6

Вывод можно представить *деревом вывода* . Дерево вывода удовлетворяет следующим условиям:

- каждая вершина дерева обозначается терминалом или нетерминалом;
- корнь дерева обозначается начальным нетерминалом;
- каждый лист дерева обозначается терминалом или символом пустой цепочки ε;
- если некоторая вершина дерева обозначена нетерминалом A, то её сыновья обозначены символами правой части правила $A \to \alpha$, применяемого при выводе.

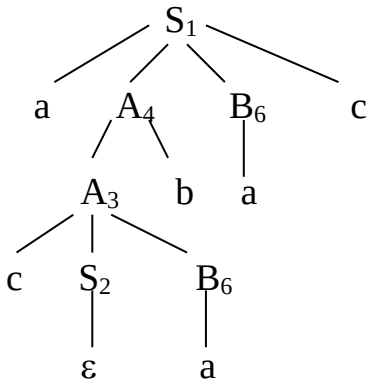
Если прочитать листья дерева вывода слева направо, то получим терминальную цепочку.

Пример. Дерево вывода цепочки acabac в грамматике G1:



Дерево вывода может быть представлено в *линейной скобочной форме* (ЛСФ). Записываем начальный нетерминал (корень дерева), за которым в скобках перечисляем сыновей. Если сын представляет собой нетерминал, то за ним в скобках перечисляем его сыновей и т.д.

Дерево вывода цепочки acabac в грамматике G1:



ЛСФ дерева вывода цепочки acabac в грамматике G1:
S(aA(A(cS()B(b))b)B(a)c).

ЛСФ можно получить, выполнив обход дерева вывода в глубину, расставляя скобки. Если в ЛСФ оставить только терминалы, то получим выводимую цепочку.

В дереве вывода отражено, какие правила применялись и к каким нетерминалам, но порядок применения правил скрыт, поэтому может быть много выводов, соответствующих одному и тому же дереву вывода.

Пример. Приведённому выше дереву соответствует другой вывод:

$S \Rightarrow aABc \Rightarrow aAbBc \Rightarrow acSBbBc \Rightarrow acBbBc \Rightarrow acabBc \Rightarrow acabac$

1 4 3 2 6 6

Вывод, на каждом шаге которого правило вывода применяется к самому левому нетерминалу промежуточной цепочки, называется *левым* (или *левосторонним*).

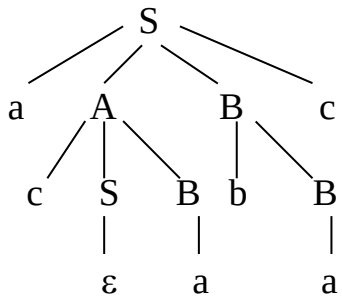
Вывод, на каждом шаге которого правило вывода применяется к самому правому нетерминалу промежуточной цепочки, называется *правым* (или *правосторонним*).

Для каждого дерева существует единственный левый и единственный правый вывод (могут совпадать).

Выводы, соответствующие одному и тому же дереву, называются *эквивалентными.* Эквивалентные выводы отличаются только последовательностью применения правил.

Неоднозначные грамматики и языки

Если одной терминальной цепочке соответствуют несколько деревьев вывода, то грамматика называется *неоднозначной*. Грамматика G1 – неоднозначная, т.к. цепочка acabac имеет другое дерево вывода:



Вывод, соответствующий этому дереву:

$S \Rightarrow aABc \Rightarrow aAbBc \Rightarrow acSBbBc \Rightarrow acSabBc \Rightarrow acabBc \Rightarrow acabac$

1 5 3 6 2 6

Если цепочка имеет два различных левых (правых) вывода, то деревья этих выводов будут различными.

Грамматика будет неоднозначной, если хотя бы одна цепочка имеет два различных левых (правых) вывода.

Проблема определения неодназначности грамматики алгоритмически неразрешима, однако существуют определённого вида правила грамматики, по наличию которых в множестве правил можно утверждать, что грамматика является неоднозначной. Эти правила имеют следующий вид:

1. $A \to AA \qquad A \to \alpha$
2. $A \to A\alpha A \qquad A \to \beta$
3. $A \to \alpha A \qquad A \to A\beta \qquad A \to \chi$
4. $A \to \alpha A \qquad A \to \alpha A \beta A \qquad A \to \chi$

1. $A \to AA$
2. $A \to \alpha$

Цепочки, выводимые в этой грамматике:
α
$\alpha\alpha$
$\alpha\alpha\alpha$ — цепочка имеет два левых вывода, грамматика неоднозначна
$\alpha\alpha\ldots\alpha$

Эта грамматика порождает язык $L=\{\alpha^n \mid n>0\}$.

Язык $L=\{\alpha^n \mid n>0\}$ определяется грамматикой:
1. $A \to \alpha A$
2. $A \to \alpha$

Эта грамматика однозначна.

---

1. $A \to A\alpha A$
2. $A \to \beta$

Цепочки, выводимые в этой грамматике:
β
$\beta\alpha\beta$
$\beta\alpha\beta\alpha\beta$ — цепочка имеет два левых вывода, грамматика неоднозначна
$\beta\alpha\beta\alpha\beta\alpha\beta$

Эта грамматика порождает язык $L=\{\beta(\alpha\beta)^n \mid n\geq 0\}$.

Язык $L=\{\beta(\alpha\beta)^n \mid n\geq 0\}$ определяется грамматикой:
1. $A \to \alpha B$
2. $B \to \alpha\beta B$
3. $B \to \varepsilon$

Эта грамматика однозначна.

Язык однозначный, если может быть описан однозначной грамматикой. Рассмотренные выше языки $L=\{\alpha^n \mid n>0\}$и $L=\{\beta(\alpha\beta)^n \mid n\geq 0\}$ однозначные.

Если язык не может быть описан однозначной грамматикой, то он неоднозначный.

Пример неоднозначного языка:

$L=\{a^nb^nc^md^m \mid n>0, m>0\} \cup \{a^nb^mc^md^n \mid n>0, m>0\}$

Грамматика этого языка:

1. $S \to AB$
2. $S \to C$
3. $A \to aAb$
4. $A \to ab$
5. $B \to cBd$
6. $B \to cd$
7. $C \to aCd$
8. $C \to aDd$
9. $D \to bDc$
10. $D \to bc$