МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

КУРГАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

ВЫЧИСЛИТЕЛЬНЫЕ СТРУКТУРЫ

Методические указания
к выполнению лабораторных работ
по курсу «Теория вычислительных процессов и структур»
для студентов специальности 220400

Курган 2004

Кафедра: «Программное обеспечение вычислительной техники и автоматизированных систем»

Дисциплина: «Теория вычислительных процессов и структур»

Составил: доцент, канд. техн. наук Кокин А.Г.

Утверждены на заседании кафедры 20 октября 2004 г.

Рекомендованы методическим советом университета
«____» ______________ 2004 г.

ЛАБОРАТОРНАЯ РАБОТА №1

КОНЕЧНЫЕ АВТОМАТЫ

Задание

1. Разработать программу-редактор для представления полных и частичных конечных автоматов любой размерности.
2. Реализовать программу функционирования конечных автоматов.
3. Создать программу минимизации полных и частичных автоматов с помощью алгоритма Мили.
4. Реализовать проверку эквивалентности исходных и минимизированных конечных автоматов путем подачи на вход и получения на выходе определенных цепочек символов.

Последовательность выполнения лабораторной работы

1. Задать полный и частичный автомат таблицей переходов в редакторе.
2. Минимизировать автомат по алгоритму Мили.
3. Осуществить проверку эквивалентности полученного минимального автомата по сравнению с исходным путем подачи на вход обеих автоматов одинаковых цепочек символов.

Основные определения

Конечным автоматом называется система $S=\{A, Q, V, \delta, \lambda\}$, где $A=\{a_1,\ldots,a_m\}$ – входной алфавит, $Q =\{q_1,\ldots,q_n\}$ – алфавит состояний, $V = \{v_1,\ldots,v_k\}$ – выходной алфавит, δ - функция переходов, λ - функция выходов. Выделяется также начальное состояние q_1.
Функции δ и λ задаются с помощью таблицы, называемой таблицей переходов состояний автомата.
Например, таблица задает функции переходов и выходов для автомата с алфавитами: $A = \{a_1, a_2, a_3\}$, $Q = \{q_1, q_2, q_3\}$, $V = \{v_1, v_2\}$ имеет вид:

q_i	a_1	a_2	a_3
q_1	q_3, v_1	q_3, v_2	q_2, v_1
q_2	q_3, v_1	q_1, v_1	q_2, v_2
q_3	q_2, v_1	q_3, v_1	q_3, v_2

Зафиксируем в автомате S начальное состояние q_1 и каждому входному слову $\alpha = a_1, \ldots, a_к$ поставим в соответствие слово выходного алфавита $\omega = \lambda(q_1,a_1)$ $\lambda(q_1,a_1,a_2)$ $\lambda(q_1,a_1,\ldots,a_k)$. Это соответствие, отображающее входное слово в выходное, называется автоматным отображением. Другими словами, если на вход конечного автомата в начальном состоянии q_1 подавать символы входного алфавита, то на выходе появятся символы выходного алфавита.

Состояние q_j называется достижимым из состояния q_1, если существует входное слово α, такое, что $\delta(q_1,\alpha) = q_j$.

Полные и частичные автоматы

Полные автоматы. Полным называется автомат, в котором функции переходов и выходов полностью определены. Автоматы S и T называются неотличимыми, если для любого состояния q автомата S найдется неотличимое от него состояние r автомата T и наоборот. Отношение неотличимости межу состояниями и автоматами является отношением эквивалентности. Переход от автомата S к эквивалентному автомату T называется эквивалентным преобразованием автомата S.

Частичные автоматы. Автомат S называется частичным, если хотя бы одна из его функций не полностью определена (в автоматной таблице некоторые клетки не заполнены).

Рассмотрим автомат, заданный таблицей:

q_i	a_1	a_2	a_3
1	2, 0	- , -	3, -
2	- , -	1, -	3, 0
3	2, 1	1, -	3, 0

Для частичных автоматов вводятся понятия покрытия и совместимости.

Рассмотрим состояния q_2 и q_3. Область определения для q_2 содержится в области определения для q_3. Можно сказать, что q_3 делает больше, чем q_2. Поэтому, если вычеркнуть строку 2, а переходы в состояние q_2 заменить на переходы в q_3, то получим автомат S'. Это приводит к понятию покрытия для состояний и автоматов – состояние q_3 покрывает состояние q_2, автомат S' покрывает автомат S.

Рассмотрим состояния q_1 и q_2. можно представить состояние, которое покрывает q_1 и q_2. Это строка 4: 2, 0; 1, -; 3, 0.

Состояние q автомата q' совместимы, если существует состояние p, покрывающее q и q'.

Минимизация автоматов

Для полных автоматов: автомат, эквивалентный заданному и имеющий наименьшее возможное число состояний, называется минимальным.

Для частичных автоматов: понятия покрытия и совместимости дают общий план минимизации частичных автоматов – находятся совместные состояния и заменяются покрывающим состоянием. Однако отношение совместимости не транзитивно (q_1q_2 и q_2q_3 совместимы, а пара q_1q_3 – нет) и следовательно не является отношением эквивалентности, поэтому классы совместимости могут пересекаться.

Для минимизации конечных автоматов используется алгоритм Мили.

Алгоритм Мили (табличный). Составляется треугольная таблица, клетки которой соответствуют различным неупорядоченным парам состояний q_iq_j. Если для состояний q_i и q_j существует входной символ, приводящий к разным значениям выхода, то соответствующая клетка таблицы зачеркивается.
В остальные клетки записываются все пары состояний, отличные от q_iq_j, в которые автомат может перейти из q_iq_j.
Далее в таблице зачеркиваются клетки, в которых присутствуют пары состояний, соответствующие вычеркнутым клеткам.
Рассмотрим пример минимизации полного автомата:

q	a1	a2	a3
1	4,1	6,0	1,0
2	2,0	2,0	6,1
3	4,1	2,0	3,0
4	4,1	3,0	3,0
5	6,0	6,0	5,1
6	5,0	6,0	2,1

Шаг 1. Первое состояние по значениям функции выхода сравнивается с остальными и находятся состояния, отличные от первого. Это 2, 5 и 6 состояния. Аналогично последовательно сравниваются 2, 3, 4 и 5 состояния с остальными и находятся отличающиеся состояния. Для 2 состояния это 3 и 4 состояния; для 3 состояния – 5, 6; для 4 – 5, 6.
Зачеркиваются клетки таблицы с найденными парами состояний:

q2	X	//////////	///////////	////////////	///////////
q3		X	///////////	////////////	///////////
q4		X		////////////	///////////
q5	X		X	X	///////////
q6	X		X	X	
	q1	q2	q3	q4	q5

Шаг 2. Заполняются пустые клетки таблицы. Для клетки пары состояний q_1q_3 по исходной таблице определяются пары состояний, в которые может перейти автомат. Это q_2q_6. Для клетки q_1q_4 – q_3q_6, q_1q_3. И так далее. Получаем заполненную таблицу:

q2	X	//////////	////////////	////////////	////////////
q3	q2q6	X	///////////	////////////	////////////
q4	q3q6, q1q3	X	q2q3	////////////	////////////
q5	X	q2q6,q5q6	X	X	///////////
q6	X	q2q5	X	X	q2q5
	q1	q2	q3	q4	q5

Шаг 3. Зачеркиваются клетки, в которых присутствуют пары зачеркнутых клеток. Получаем:

q2	X	//////////	////////////	////////////	////////////
q3	q2q6	X	///////////	////////////	////////////
q4	X	X	X	////////////	////////////
q5	X	q2q6,q5q6	X	X	///////////
q6	X	q2q5	X	X	q2q5
	q1	q2	q3	q4	q5

Не вычеркнутые клетки результирующей таблицы соответствуют парам эквивалентных состояний: q_1q_3, q_2q_5, q_2q_6, q_5q_6. Они образуют вместе с состоянием q_4 три класса: (q_1q_3), (q_4), $(q_2q_5q_6)$.

Минимальный автомат будет иметь вид:

q	A_1	a_2	a_3
1 (1,3)	2, 1	3, 0	1, 0
2 (4)	3, 0	3, 0	3, 1
3 (2,5,6)	2, 1	1, 0	1, 0

ЛАБОРАТОРНАЯ РАБОТА №2

КОНЕЧНЫЕ РАСПОЗНАВАТЕЛИ

Задание

1. Разработать программу-редактор для представления детерминированных и недетерминированных конечных распознавателей различной размерности.
2. Создать программу построения недетерминированных конечных распознавателей по заданным цепочкам символов.
3. Реализовать программу преобразования недетерминированных распознавателей в детерминированные.
4. Создать программу минимизации детерминированных конечных распознавателей.
5. Реализовать проверку эквивалентности недетерминированных, детерминированных и минимизированных конечных распознавателей путем подачи на вход и распознавания исходных цепочек символов.

Последовательность выполнения лабораторной работы

1. На основе исходных цепочек символов построить недетерминированный распознаватель.
2. Преобразовать недетерминированный распознаватель в детерминированный.
3. Минимизировать детерминированный распознаватель.

4. Осуществить проверку эквивалентности полученного минимального автомата-распознавателя по сравнению с исходным путем подачи на вход и распознавания исходных цепочек символов.

Детерминированные конечные распознаватели

Основные определения

Конечный распознаватель – это модель устройства с конечным числом состояний, которое отличает допустимые цепочки символов от недопустимых.
Конечный распознаватель задается: S={A, Q, δ, F, q}, где $A=\{a_1,\ldots,a_m\}$ – входной алфавит, $Q=\{q_1,\ldots,q_n\}$ – алфавит состояний, δ - функция переходов, F – подмножество состояний, выделяемых в качестве допускающих, q – начальное состояние.
Например, автомат

	x	y	z	
1	1	3	4	1
2	2	1	3	0
3	2	4	4	1
4	3	3	3	0

имеет входное множество A={x, y, z}, множество состояний Q = {1, 2, 3, 4}, начальное состояние $q_1 = 1$, допускающие состояния F= {1, 3}.
Входная цепочка xyzz допускается автоматом, т.к. $1 \xrightarrow{x} 1 \xrightarrow{y} 3 \xrightarrow{z} 4 \xrightarrow{z} 3$.
Цепочка zyx отвергается, потому что $1 \xrightarrow{z} 4 \xrightarrow{y} 3 \xrightarrow{x} 2$.

Минимизация конечных распознавателей

Состояния q и r эквивалентны, если выполняются два условия: условие подобия – состояния q и r должны быть либо допускающими, либо отвергающими; условие преемственности – для всех входных символов состояния q и r должны переходить в эквивалентные состояния.
Автомат можно превратить в эквивалентный ему минимальный автомат, убирая недостижимые состояния и объединяя эквивалентные состояния.
Метод разбиения. Он заключается в разбиении множества состояний на непересекающиеся подмножества (блоки) такие, что неэквивалентные состояния попадают в разные блоки.
Например, для автомата-распознавателя

	a	b	
1	6	3	0
2	7	3	0
3	1	5	0
4	4	6	0
5	7	3	1
6	4	1	1
7	4	2	1

все состояния разбиваются на два блока: допускающие и отвергающие.

$$P_1 =(\{1, 2, 3, 4\},\{5, 6, 7\}).$$

Рассмотрим блок {1, 2, 3, 4} при входном символе a. Состояния 3 и 4 переходят в состояния 1 и 4 этого же блока, тогда как состояния 1 и 2 переходят в состояния 6 и 7 второго блока. Это нарушает условие преемственности, поэтому позволяет произвести новое разбиение.

$$P_2 ==(\{1, 2\},\{ 3, 4\},\{5, 6, 7\}).$$

Повторяем процесс разбиения пока это возможно. Разбиваем блок { 3, 4} и блок {5, 6, 7}. Окончательно получаем

$$P = (\{1, 2\},\{ 3\},\{ 4\},\{5\},\{6, 7\}).$$

Состояния внутри каждого блока эквивалентны. Минимальный эквивалентный распознаватель имеет вид:

		a	b	
{1, 2}	1	5	2	0
{ 3}	2	1	4	0
{ 4}	3	3	5	0
{5}	4	5	2	1
{6, 7}	5	3	1	1

Недетерминированные конечные распознаватели

Основные определения

Недетерминированный конечный распознаватель отличается от детерминированного тем, что значениями его функции переходов могут быть множества состояний.

$S=\{A, Q, \delta, F, Q_0\}$, где $A=\{a_1,\ldots,a_m\}$ – входной алфавит, $Q =\{q_1,\ldots,q_n\}$ – алфавит состояний, δ - функция переходов, F – подмножество состояний, выделяемых в качестве допускающих, Q_0 – подмножество начальных состояний.

Автомат допускает входную цепочку, если она позволяет связать одно из его начальных состояний с одним из допускающих.

Например, недетерминированный автомат

	0	1	
→ A	A,B	C	0
→ B	B	C	1
C	-	A,C	1

имеет $A =\{0, 1\}$, $Q = \{A, B, C\}$, $F = \{B, C\}$, $Q_0 =\{A, B\}$.

Цепочка 11 допускается автоматом, т.к. $B \xrightarrow{1} C \xrightarrow{1} C$. Переход $\delta(C,0)$ является переходом в пустое множество, это означает, что дальнейшие переходы невозможны.

Построение недетерминированного распознавателя

Возможности автомата, состоящие в определении допустимого множества входных цепочек, используются для построения автоматов распознавателей. Например, построить автомат, распознающий цепочки: *фактор, кактус*. Для простоты алфавит состояний обозначим как входной алфавит.

	ф	а	к	т	о	р	у	с	
→Ф	А								
А		К							
→К			Т,А						
Т				О,У					
О					Р				
Р						Z			
У							С		
С								Z	
Z									1

$Ф \xrightarrow{ф} А \xrightarrow{а} К \xrightarrow{к} Т \xrightarrow{т} О \xrightarrow{о} Р \xrightarrow{р} Z$ - допускается.

$К \xrightarrow{к} А \xrightarrow{а} К \xrightarrow{к} Т \xrightarrow{т} У \xrightarrow{у} С \xrightarrow{с} Z$ - допускается.

Преобразование недетерминированного распознавателя

Для каждого недетерминированного конечного распознавателя существует детерминированный конечный распознаватель, который допускает в точности те же входные цепочки, что и недетерминированный.

Процедура задается пятью шагами. Пусть Ан – недетерминированный автомат, а Ад – детерминированный, который нужно построить.

Шаг 1. Пометить первую строку таблицы переходов Ад множеством начальных состояний Ан.

Шаг 2. Для этого множества состояний по Ан определить множества переходов, куда может перейти автомат Ад по каждому входу.

Шаг 3. Из этих множеств состояний создать для Ад новые строки состояний и пометить их этими множествами.

Шаг 4. Для каждой новой строки состояний Ад определить множества переходов по Ан (шаг 2).

Шаг 5. Пометить строки состояний как допускающие, если они содержат хотя бы одно допускающее состояние автомата Ан.

Например, имеется недетерминированный автомат.

	0	1	
→ A	A,B	C	0
→ B	B	C	1
C	-	A,C	1

Построить эквивалентный ему детерминированный автомат.

Шаг 1.

	0	1
{A,B}		

Шаг 2.

	0	1
{A,B}	{A,B}	{C}

Шаг 3.

	0	1
{A,B}	{A,B}	{C}
{C}		

Шаг 4.

	0	1
{A,B}	{A,B}	{C}
{C}	{ }	{A,C}
{ }	{ }	{ }
{A,C}	{A,B}	{A,C}

Шаг 5.

	0	1	
1	1	2	1
2	3	4	1
3	3	3	0
4	1	4	1

ЛАБОРАТОРНАЯ РАБОТА №3

ФОРМАЛЬНЫЕ ЯЗЫКИ И ГРАММАТИКИ

Задание

1. Разработать программу-редактор для представления контекстно-свободной грамматики и программу вывода из грамматики цепочек символов.
2. Реализовать приведение контекстно-свободной грамматики к определенному виду.
3. Создать программу преобразования полученной грамматики в автоматную грамматику.
4. Разработать программу создания недетерминированного конечного распознавателя по автоматной грамматике.

5. Реализовать проверку эквивалентности полученного минимального автомата-распознавателя по сравнению с исходной грамматикой.

Последовательность выполнения лабораторной работы

1. Исходную контекстно-свободную (КС) грамматику привести к соответствующему виду: удалить непродуктивные и недостижимые нетерминальные символы. Вывести несколько терминальных цепочек и сравнить их с цепочками, выведенными в исходной грамматике.
2. Преобразовать полученную грамматику в автоматную (праволинейную, регулярную) грамматику.
3. Построить недетерминированный конечный распознаватель.
4. Преобразовать недетерминированный конечный распознаватель в детерминированный автомат и минимизировать его.
5. Осуществить проверку эквивалентности полученного минимального автомата-распознавателя по сравнению с исходной грамматикой путем подачи на вход распознавателя исходных выведенных из грамматики цепочек символов.

Языки и грамматики

.

Формальный язык L в алфавите V – это произвольное подмножество $L \subseteq V^*$.
Формальная порождающая грамматика G – это формальная система G = (V, W, I, P), где V – основной, терминальный алфавит, W – вспомогательный, нетерминальный алфавит, I – начальный символ грамматики, P – конечное множество правил вывода.
Обозначения. Терминальный алфавит – (a, b, c). Нетерминальный алфавит – (A, B, C). Цепочки символов – (α, β). Множество цепочек символов V*.
Классификация грамматик. Общепринятой классификацией грамматик является иерархия Хомского:
- типа 0–грамматика произвольного вида без ограничений на правила вывода;
- типа 1 – контекстная грамматика, все правила которой имеют вид:

$$\alpha A\beta \to \alpha\omega\beta,$$

где α,β,ω - цепочки в алфавите $(V \cup W)$, α, β - контекст правила;
- типа 2 – контекстно-свободная грамматика (КС), все правила которой имеют вид: $A \to \alpha$, где $\alpha \in (V \cup W)$;
- типа 3 – праволинейная, регулярная (автоматная) грамматика, все правила которой имеют вид: $A \to \alpha B$ либо $A \to \alpha$.

Вывод в грамматике. Рассмотрим КС-грамматику и вывод из нее терминальных цепочек символов:

1. $I \to aABc$
2. $I \to e$
3. $A \to cIB$

Вывод:

$I \to a\,A\,B\,c$ (3)

$\to a\,c\,I\,B\,c$ (2)

4. $A \to Ab$ $\quad\quad \to a\ c\ B\ c$ (5)
5. $B \to bB$ $\quad\quad \to a\ c\ b\ B\ c$ (6)
6. $B \to a$ $\quad\quad \to a\ c\ b\ a\ c$ - терминальная цепочка.

Приведение КС – грамматик

Нетерминальный символ называется существенным, если он является достижимым ($I \to^* \alpha A \beta$) и продуктивным ($A \to^* \gamma$), в противном случае он называется несущественным или бесполезным.

Удаление бесплодных нетерминалов

Нетерминальный символ является продуктивным, если из него выводится терминальная цепочка, иначе он является бесплодным.

Шаг 1. Составить список нетерминалов в правилах, правая часть которых не содержит нетерминальных символов.

Шаг 2. Если правая часть правила содержит нетерминалы, занесенные в список. Занести в него и нетерминалы левой части. Полученный список содержит только продуктивные нетерминальные символы.

Например, в грамматике

1.	$I \to aABI$	
2.	$I \to bcACd$	$I \to bcACd$
3.	$A \to bAB$	
4.	$A \to cIA$	$A \to cIA$
5.	$A \to cCC$	$A \to cCC$
6.	$B \to bAB$	
7.	$B \to cIB$	
8.	$C \to cI$	$C \to cI$
9.	$C \to c$	$C \to c$

на первом шаге в список можно занести нетерминал C из правила 9, на втором шаге – нетерминал A из правила 5 и I из правила 2. Продуктивными нетерминалами являются C, A, I, а нетерминал B – бесплодным.

Удаление недостижимых нетерминалов

Если нетерминал в левой части правила является достижимым, то достижимы и все нетерминалы правой части.

Шаг 1. Образовать список из начального нетерминала.

Шаг 2. если левая часть правила содержит нетерминалы, включенные в список, то включить в него и нетерминалы правой части.

Например, в грамматике

1.	$I \to aAI$	$I \to aAI$
2.	$I \to bcACd$	$I \to bcACd$
3.	$A \to bA$	$A \to bA$

4.	$A \rightarrow cIA$	$A \rightarrow cIA$
5.	$A \rightarrow cCC$	$A \rightarrow cCC$
6.	$B \rightarrow bAB$	
7.	$B \rightarrow cIB$	
8.	$C \rightarrow cI$	$C \rightarrow cI$
9.	$C \rightarrow c$	$C \rightarrow c$

на первом шаге в список заносится нетерминал I, на втором шаге – нетерминал A из правила 1 и C из правила 2. Недостижимым является нетерминал B.

Преобразование грамматик

Любое регулярное множество можно описать с помощью КС–грамматики.
На основании КС-грамматики возможно построение конечного автомата-распознавателя путем преобразования КС-грамматики в автоматную грамматику. Затем входным множеством сделать терминальное множество грамматики; множеством состояний автомата взять нетерминальное множество грамматики; начальное состояние – начальный нетерминал; ввести в автомат переход из состояния A в состояние B по входу x, если в грамматике имеется правило $A \rightarrow xB$; сделать состояние A допускающим, если в грамматике есть правило $A \rightarrow e$. Результатом этого построения является недетерминированный автомат с одним начальным состоянием.
Например, дана грамматика

1. $I \rightarrow aA$
2. $I \rightarrow bc$
3. $I \rightarrow A$
4. $A \rightarrow abbI$
5. $A \rightarrow cA$
6. $A \rightarrow e$

Правило 4 не имеет надлежащего вида, так как в нем три терминальных символов вместо одного. Заменим правило 4 тремя правилами:

$$A \rightarrow a<bbI>$$
$$<bbI> \rightarrow b<bI>$$
$$<bI> \rightarrow b<I>$$

Правило 2 в правой части не имеет нетерминал, поэтому заменим его двумя правилами:

$$I \rightarrow bc<E>$$
$$E \rightarrow e,$$

которые в окончательном виде имеют вид:

$$I \rightarrow b<cE>$$
$$<cE> \rightarrow cE$$
$$E \rightarrow e,$$

Правило 3 заменим для всех правил, содержащих A в левой части:

$$I \rightarrow a<bbI>$$
$$I \rightarrow cA$$
$$I \rightarrow e$$

В результате получим грамматику:

$$I \rightarrow aA$$
$$I \rightarrow b<cE>$$
$$<cE> \rightarrow cE$$
$$E \rightarrow e$$
$$I \rightarrow a<bbI>$$
$$I \rightarrow cA$$
$$I \rightarrow e$$
$$A \rightarrow a<bbI>$$
$$<bbI> \rightarrow b<bI>$$
$$<bI> \rightarrow b<I>$$
$$A \rightarrow cA$$
$$A \rightarrow e$$

Недетерминированный конечный распознаватель имеет вид:

	a	b	c	
I	A, bbI	cE	A	1
CE			E	0
E				1
A	bbI		A	1
bbI		bI		0
bI		I		0

СПИСОК ЛИТЕРАТУРЫ

1. 519.9 Л89 Кузнецов О.П., Адельсон-Вельский Г.М. Дискретная математика для инженера. 2 изд., - М.: Энергоиздат, 1988 -480 с.

2. 681.32 Ш78 Шоломов Л.А. Основы теории дискретных логических и вычислительных устройств. - М: Наука, 1989. - 400 с.

3. 681.3.06 Л91 Льюис Ф., Розенкранц Д., Стирнз Р. Теоретические основы проектирования компиляторов. - М.: Мир, 1979. – 564 с.

4. Минский М. Вычисления и автоматы. - М.: Мир, 1971.-368 с.

5. Гросс М., Лантен Л. Теория формальных грамматик.- М.:Мир,1971. - 294 с.

Кокин Александр Георгиевич

ВЫЧИСЛИТЕЛЬНЫЕ СТРУКТУРЫ

Методические указания
к выполнению лабораторных работ
по курсу «Теория вычислительных процессов и структур»
для студентов специальности 220400

Редактор Н.М. Кокина

Подписано в печать	Формат 60x84 1/16	Бумага тип. N1
Плоская печать	Усл.-печ.л. 1,5	Уч.-изд.л. 1,5
Заказ	Тираж 50	Цена свободная

Издательство Курганского государственного университета,
640669, г. Курган, ул. Гоголя, 25,
Курганский государственный университет, ризограф.