МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра ИиСП

КУРСОВАЯ РАБОТА

По дисциплине: «Операционные системы»

НАПРАВЛЕНИЯ ПОДГОТОВКИ

09.03.04 - ПРОГРАММНАЯ ИНЖЕНЕРИЯ

Выполнил: студент 3 курса

специальности «ПС»

группы «ПС-31»

Садовин Алексей Альбертович

Подпись:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил:

Нехорошкова Людмила Георгиевна

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Йошкар-Ола

2022 год

Содержание

[Введение 3](#_Toc100161395)

[Основная часть 4](#_Toc100161396)

[Минимизация автоматов Мили и Мура 5](#_Toc100161397)

[Перевод недетерминированного конечного автомата в детерминированный 7](#_Toc100161398)

[Перевод регулярного выражения в детерминированный конечный автомат 8](#_Toc100161399)

[Лексический анализатор 9](#_Toc100161400)

[Сквозной пример 10](#_Toc100161401)

[Заключение 11](#_Toc100161402)

# Введение

Целью курсового проекта является закрепление материала, изученного на курсе по дисциплине “Операционные системы”. Данный курсовой проект затрагивает теорию автоматов, которая является разделом дискретной математики, изучающий абстрактные автоматы — вычислительные машины, представленные в виде математических моделей — и задачи, которые они могут решать. Теория автоматов наиболее тесно связана с теорией алгоритмов: автомат преобразует дискретную информацию по шагам в дискретные моменты времени и формирует результат по шагам заданного алгоритма.

Теория автоматов лежит в основе всех цифровых технологий и программного обеспечения, так, например, компьютер является частным случаем практической реализации конечного автомата. Часть математического аппарата теории автоматов напрямую применяется при разработке лексических и синтаксических анализаторов для формальных языков, в том числе языков программирования, а также при построении компиляторов и разработке самих языков программирования, описания аппаратуры, а также разметки. Другое важнейшее применение теории автоматов — математически строгое нахождение разрешимости и сложности задач.

# Основная часть

Конечный автомат (КА) — математическая абстракция, модель дискретного устройства, имеющего один вход, один выход и в каждый момент времени находящегося в одном состоянии из множества возможных. Является частным случаем абстрактного дискретного автомата, число возможных внутренних состояний которого конечно. При работе на вход КА поступают последовательно входные воздействия, а на выходе КА формирует выходные сигналы. Обычно под входными воздействиями принимают подачу на вход автомата символов одного алфавита, а на выход КА в процессе работы выдаёт символы в общем случае другого, возможно даже не пересекающегося с входным, алфавита.

Различают детерминированные КА — автоматы, в которых следующее состояние однозначно определяется текущим состоянием и выход зависит только от текущего состояния и текущего входа, и недетерминированные КА, следующее состояние у которых в общем случае неопределённо и, соответственно, не определён выходной сигнал.

Теория конечных автоматов практически широко используется, например, в синтаксических и лексических анализаторах, тестировании программного обеспечения на основе моделей.

Для изучения и закрепления материала данного курса был выполнены следующие лабораторные работы:

* Минимизация автоматов Мили и Мура
* Детерминирование конечного автомата
* Перевод регулярного выражения в детерминированный конечный автомат
* Лексический анализатор

Программы для лабораторных работ разрабатывались на языке программирования С++.

## Минимизация автоматов Мили и Мура

Задача минимизации автомата сводится к поиску его минимальной формы. Для произвольного конечного автомата может быть построен эквивалентный ему конечный автомат с наименьшим числом состояний.

**Минимальный автомат** — это автомат, имеющий наименьшее возможное количество состояний и реализующий заданную функцию выходов. Задача минимизации автомата сводится к поиску его минимальной формы. Для произвольного конечного автомата может быть построен эквивалентный ему конечный автомат с наименьшим числом состояний.

**Автомат Мили** — конечный автомат, выходная последовательность которого (в отличие от автомата Мура) зависит от состояния автомата и входных сигналов. Это означает, что в графе состояний каждому ребру соответствует некоторое значение (выходной символ). В вершины графа автомата Мили записываются выходящие сигналы, а дугам графа приписывают условие перехода из одного состояния в другое, а также входящие сигналы.

**Автомат Мура** — конечный автомат, выходное значение сигнала в котором зависит лишь от текущего состояния данного автомата, и не зависит напрямую, в отличие от автомата Мили, от входных значений.

Алгоритм минимизации состоит в последовательном построении разбиения множества состояний на одно-, два-, три- и так далее эквивалентные. Если текущее разбиение совпадает с предыдущим, то полученные классы эквивалентности и определяют минимальный автомат.

Примеры входных данных:

|  |  |
| --- | --- |
| Автомат Мили | Автомат Мура |
| Mi  8  3  2  S4 S5 S3 S5 S7 S1 S5 S3 Y1 Y2 Y2 Y1 Y1 Y1 Y1 Y2 S2 S1 S5 S8 S2 S3 S3 S5 Y2 Y1 Y1 Y2 Y2 Y2 Y2 Y1 S5 S4 S4 S4 S1 S4 S7 S6 Y1 Y2 Y2 Y2 Y1 Y2 Y2 Y2 | Mr  5  2  3  Y1 Y1 Y3 Y2 Y3  S2 S5 S5 S3 S3 S4 S2 S2 S1 S1 |

Структура входных данных по строчно:

1. Тип автомата
2. Количество начальных состояний
3. Количество входных символов
4. Количество выходных символов

Далее идут сами входные данные.

Примеры выходных данных:

|  |  |
| --- | --- |
| Автомат Мили | Автомат Мура |
| S1 S2 S3 S4  X1 [S4, Y1] [S1, Y2] [S3, Y2] [S1, Y1]  X2 [S2, Y2] [S1, Y1] [S1, Y1] [S3, Y2]  X3 [S1, Y1] [S4, Y2] [S4, Y2] [S4, Y2] | Y1 Y1 Y3 Y3 Y2  S1 S2 S3 S5 S4  X1 [S2] [S5] [S5] [S3] [S3]  X2 [S4] [S2] [S2] [S1] [S1] |

## Перевод недетерминированного конечного автомата в детерминированный

Детерминированный конечный автомат, известный также как детерминированный конечный распознаватель — это конечный автомат, принимающий или отклоняющий заданную строку символов путём прохождения через последовательность состояний, определённых строкой и имеет единственную последовательность состояний во время работы.

ДКА распознаёт в точности множество регулярных языков, которые, среди прочего, полезны для лексического анализа и сопоставления с образцом. ДКА могут быть построены из недетерминированного конечного автомата.

Программа может детерминировать как леволинейную, так и праволинейную грамматику. Для распознавания грамматики первой строкой в файле указывается тип грамматики, а второй строкой идет количество состояний.

Пример входного файла:

|  |  |
| --- | --- |
| Праволиненейная грамматика | Леволинейная грамматика |
| * R S:aS|aB|cS|c B:bB|aB|a | * L S:Ab|b A:Sc|c|Aa |

Пример входного файла:

H -> bS | cA

S -> cA

A -> aA | bS

## Перевод регулярного выражения в детерминированный конечный автомат

Регулярные выражения — используемый в компьютерных программах, работающих с текстом, формальный язык поиска и осуществления манипуляций с подстроками в тексте, основанный на использовании метасимволов. Для поиска используется строка-образец, состоящая из символов и метасимволов и задающая правило поиска. Для манипуляций с текстом дополнительно задаётся строка замены, которая также может содержать в себе специальные символы.

Чтобы преобразовать регулярное выражение в ДКА, нужно:

* Преобразовать регулярное выражение в НКА с ε-переходами
* Устранить ε-переходы
* Построить по НКА эквивалентный ДКА

Пример входных данных:

(xy\*|ab|(x|a\*))(x|y\*)

Пример выходного файла для этого регулярного выражения:

0 -> a1 | x2 | y3

1 -> a4 | b5 | x6 | y3

2 -> x6 | y7

3 -> y3

4 -> a4 | x6 | y3

5 -> x6 | y3

7 -> x6 | y7

## Лексический анализатор

**Лексический анализ** — процесс аналитического разбора входной последовательности символов на распознанные группы — лексемы — с целью получения на выходе идентифицированных последовательностей, называемых «токенами» (подобно группировке букв в словах).

Мною было принято решение написать лексический анализатор для языка С++. На вход программе выдаётся исходный код программы на языке C++. На выход программа выдает список токенов и их значений, а так же позицию в файле.

Пример выходного файла:

TOKEN VALUE LINE POS

INT int 0 0

IDENTIFIER n 0 4

OPEN\_PARENTHESIS ( 0 8

CLOSE\_PARENTHESIS ) 0 9

OPEN\_CURLY\_BRACKET { 0 10

FLOAT float 1 4

IDENTIFIER f 1 10

ASSIGN = 1 12

DOT . 1 14

NUMBER 0f 1 15

SEMI\_COLON ; 1 17

## Сквозной пример

Ориентируясь на выполненные лабораторные работы, можно привести сквозной пример преобразование регулярного выражения в детерминированный автомат.

Регулярное выражение:

(xy\*|ab|(x|a\*))(x|y\*)

Преобразуем его в детерминированный автомат:

0 -> a1 | x2 | y3

1 -> a4 | b5 | x6 | y3

2 -> x6 | y7

3 -> y3

4 -> a4 | x6 | y3

5 -> x6 | y3

7 -> x6 | y7

# Заключение

Целью курсового проекта является закрепление материала, изученного на курсе по дисциплине “Операционные системы”.

В процессе курсовой работы были написаны программы для взаимодействия и преобразования разных типов грамматик, конечных автоматов, использование регулярных выражений. Изучены такие темы программирования как конечные автоматы, недетерминированные и детерминированные автоматы, леволинейная и праволинейная грамматика, перевод автомата Мили в Мура и наоборот, минимизация автоматов, перевод недетерминированного автомата в детерминированный, перевод регулярного выражения в детерминированный конечный автомат, лексический анализатор для языка С++.

Данная практическая работа оказалась очень ценной и необходимой для развития навыков программирования.

Поставленная цель достигнута.