Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

О Т Ч Е Т

по домашнему заданию

Название: Минимизация конечного автомата

Дисциплина: Прикладная теория цифровых автоматов

Студент	ИУ6-41Б		29.05.2022	Д.М. Голдышев
	(Группа)		(Подпись, дата)	(И.О. Фамилия)
Преподаватель				Ю.И. Бауман
			(Подпись, дата)	(И.О. Фамилия)

Москва, 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящей работе реализован алгоритм минимизации конечного автомата на языке Pascal.

Для реализации автомата существует 2 способа: аппаратный и программный. Программная реализация выполняется с помощью языков программирования высокого уровня. Аппаратная реализация подразумевает построение устройств с памятью для запоминания текущего состояния автомата, в роли которых обычно используются триггеры.

В данной работе использован программный способ реализации цифрового автомата, так как этот способ подразумевает вариативность реализации, возможность отладки и тестирования в процессе разработки программы.

Задание (вариант 4): минимизация конечного автомата.

Цель работы – закрепить навыки реализации конечных цифровых автоматов. Для реализации поставленной цели необходимо выполнить следующие задачи:

- Изучить задание в соответствии с вариантом;
- Описать автомат, соответствующий условию задачи;
- Изучить способы реализации автомата;
- Выбрать один из изученных способов;
- Программно реализовать описанный цифровой автомат;
- Протестировать работу программной реализации;

ОСНОВНАЯ ЧАСТЬ

Спецификация реализуемого автомата

Рассмотрим состояния автомата, входные и выходные сигналы.

1. Состояния автомата:
   - q0 – начальное состояние;
   - q1 – составлено разбиение π_k;
   - q2 – конечное состояние;
2. Входные сигналы:
   - a – входные данные некорректны (N < 1 или N > 10);
   - b – входные данные корректны (1 ≤ N ≤ 10);
   - c – разбиение π_k совпало с разбиением π_{k-1};
   - d – разбиение π_k не совпало с разбиением π_{k-1};
3. Выходные сигналы:
   - 0 – невозможно минимизировать автомат из-за некорректных введенных данных (N < 1 или N > 10);
   - 1 – составлено разбиение π_k;
   - 2 – процесс минимизации завершен;
   - 3 – можно составить следующие разбиение π_{k+1}.

Полученный цифровой автомат

Составим таблицу переходов, описывающую конечный автомат, составленный по условию задачи в результате проведенного анализа (см. таблицу 1).

Таблица 1 – таблица переходов конечного автомата

Состояние	**δ**				**λ**			
	a	**b**	**c**	**d**	**a**	**b**	**c**	**d**
q_0	q_0	q_1	–	–	0	1	–	–
q_1	–	–	q_2	q_0	–	–	2	3
q_2	–	–	–	–	–	–	–	–

Представим описанный автомат в виде графа переходов (см. рисунок 1).

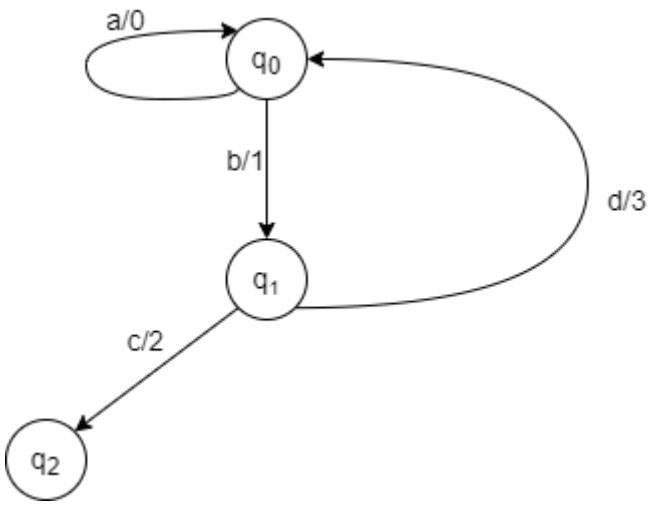

Рисунок 1 – Граф переходов цифрового автомата

Программная реализация цифрового автомата

Для реализации описанного цифрового автомата была разработана программа на высокоуровневом языке программирования Pascal (см. листинг 1).

Листинг 1 – Программа, реализующая цифровой автомат

```
program automat;
{$APPTYPE CONSOLE}
{$R *.res}

uses
  System.SysUtils;

Var
  input_atmt: array[1..10, 1..5] of integer;
  raz_mas: array[1..19] of integer;
  raz_cache: array[1..19] of integer;

  i, N, k, c, j, m, raz_count: integer;
  last_raz: boolean;
```

```
procedure input_auto();
begin
  writeln('Введите количество состояний исходного автомата (число от 1 до 10)');
  write('-> ');
  readln(N);

  while (N > 10) or (N < 1) do
  begin
    writeln('Ошибка! Автомат не может содержать заданное количество состояний.'
          + sLineBreak + 'Введите количество еще раз.');
    write('-> ');
    readln(N)
  end;

  k:= 1;
  for i:= 1 to N do
  begin
    raz_mas[k]:= i;
    k:= k + 2
  end;

  writeln(sLineBreak + 'Введите строки в следующем формате:' + sLineBreak +
  '   1 число: <переход автомата для сигнала a>' + sLineBreak +
  '   2 число: <переход автомата для сигнала b>' + sLineBreak +
  '   3 число: <выходной сигнал для сигнала a>' + sLineBreak +
  '   4 число: <выходной сигнал для сигнала b>');
  for i:= 1 to N do
```

```
  begin
    input_atmt[i, 1]:= i;
    write('-> ');
    readln(input_atmt[i, 2], input_atmt[i, 3], input_atmt[i, 4], input_atmt[i, 5]);
  end;

  writeln(sLineBreak + '~~~ Минимизация автомата ~~~');
  write('Разбиение pi0: < ');
  for i:= 1 to N do
    write('q', i, ' ');
  writeln('>');
end;

procedure next_raz_output();
begin
  inc(raz_count);
  write('Разбиение pi' + IntToStr(raz_count) + ': <');
  for i:= 1 to 19 do
  begin
    if raz_cache[i] <> 0 then write(' q', raz_cache[i])
    else if (i < 19) and (raz_cache[i + 1] <> 0) then write(' > <')
  end;
  writeln(' >');
end;

function raz_compare():boolean;
begin
  Result:= False;
  for i:= 1 to 19 do
    if raz_mas[i] <> raz_cache[i] then
```

```
    begin
      Result:= True;
      break;
    end;
end;

procedure razbieniye();
begin
  k:= 1;
    for i:= 1 to N do
    begin
      c:= 0;
      for j:= 1 to 19 do
        if i = raz_cache[j] then inc(c);
      if c = 0 then
      begin
        raz_cache[k]:= i;
        inc(k);
        m:= 0;
        for j:= i + 1 to N do
          if (input_atmt[i][4] = input_atmt[j][4]) and (input_atmt[i][5] =
input_atmt[j][5])
             and ((i < N) or ((i = N) and (i <> j))) then
          begin
            raz_cache[k]:= j;
            inc(k);
            inc(m);
          end;
      end;
      if (c = 0) or (m > 0) then
```

```
      begin
        raz_cache[k]:= 0;
        inc(k);
      end;
    end;
  next_raz_output();
  last_raz:= raz_compare();
  if last_raz then
    for i:= 1 to 19 do
    begin
      raz_mas[i]:= raz_cache[i];
      raz_cache[i]:= 0;
    end;
  for i:= 1 to N do begin
    m:= 1;
    for j:= 1 to 19 do
    begin
      if input_atmt[i][2] = raz_mas[j] then break;
      if raz_mas[j] = 0 then inc(m);
    end;
    input_atmt[i][4]:= m;
    m:= 1;
    for j:= 1 to 19 do
    begin
      if input_atmt[i][3] = raz_mas[j] then break;
      if raz_mas[j] = 0 then inc(m);
    end;
    input_atmt[i][5]:= m;
  end;
end;
```

```
begin
  writeln('--- Программа для минимизации автомата ---' + sLineBreak);
  raz_count:= 0;
  last_raz:= True;
  input_auto();
  while last_raz do
    razbieniye();
  writeln('~~~ Последнее разбиение совпало с полседним ~~~' + sLineBreak +
  sLineBreak + '        !!! Минимизация завершена !!!');
end.
```

Тестирование программы

В таблице 2 представлены наборы тестовых данных для полученной программной реализации цифрового автомата, а на рисунках 2-5 результаты работы программы при соответствующих тестовых данных.

Таблица 2 – Набор тестовых данных

№	Входной автомат				Ожидаемый результат	Полученный результат
1	N = 7				< q1 > < q2 > < q3 > < q4 > < q5 > < q6 > < q7 >	< q1 > < q2 > < q3 > < q4 > < q5 > < q6 > < q7 >
	1	7	1	0		
	2	6	0	1		
	3	5	1	0		
	4	3	1	0		
	5	2	0	1		
	6	1	0	1		
	6	7	1	0		
2	N = 7				< q1 > < q2 > < q3 > < q4 > < q5 > < q6 > < q7 >	< q1 > < q2 > < q3 > < q4 > < q5 > < q6 > < q7 >
	2	3	0	1		
	3	4	1	0		
	4	5	0	1		

<table>
<tr><td></td><td>5</td><td>6</td><td>1</td><td>0</td><td></td><td></td></tr>
<tr><td></td><td>6</td><td>7</td><td>0</td><td>1</td><td></td><td></td></tr>
<tr><td></td><td>7</td><td>1</td><td>1</td><td>0</td><td></td><td></td></tr>
<tr><td></td><td>1</td><td>2</td><td>0</td><td>1</td><td></td><td></td></tr>
<tr><td>3</td><td colspan="4">N = 7</td><td>< q1 q3 > < q2 > < q4 > < q5 > < q6 > < q7 ></td><td>< q1 q3 > < q2 > < q4 > < q5 > < q6 > < q7 ></td></tr>
<tr><td></td><td>3</td><td>6</td><td>0</td><td>1</td><td></td><td></td></tr>
<tr><td></td><td>6</td><td>2</td><td>0</td><td>1</td><td></td><td></td></tr>
<tr><td></td><td>1</td><td>6</td><td>0</td><td>1</td><td></td><td></td></tr>
<tr><td></td><td>2</td><td>3</td><td>1</td><td>0</td><td></td><td></td></tr>
<tr><td></td><td>4</td><td>1</td><td>0</td><td>1</td><td></td><td></td></tr>
<tr><td></td><td>3</td><td>1</td><td>1</td><td>0</td><td></td><td></td></tr>
<tr><td></td><td>4</td><td>6</td><td>0</td><td>1</td><td></td><td></td></tr>
<tr><td>4</td><td colspan="4">N = 15</td><td>Ошибка! Автомат не может содержать заданное количество состояний.</td><td>Ошибка! Автомат не может содержать заданное количество состояний.</td></tr>
<tr><td>5</td><td colspan="4">N = -5</td><td>Ошибка! Автомат не может содержать заданное количество состояний.</td><td>Ошибка! Автомат не может содержать заданное количество состояний.</td></tr>
</table>

```
--- Программа для минимизации автомата ---

Введите количество состояний исходного автомата (число от 1 до 10)
-> 7

Введите строки в следующем формате:
   1 число: <переход автомата для сигнала a>
   2 число: <переход автомата для сигнала b>
   3 число: <выходной сигнал для сигнала a>
   4 число: <выходной сигнал для сигнала b>
-> 1 7 1 0
-> 2 6 0 1
-> 3 5 1 0
-> 4 3 1 0
-> 5 2 0 1
-> 6 1 0 1
-> 6 7 1 0

~~~ Минимизация автомата ~~~
Разбиение pi0: < q1 q2 q3 q4 q5 q6 q7 >
Разбиение pi1: < q1 q3 q4 q7 > < q2 q5 q6 >
Разбиение pi2: < q1 q4 > < q2 q5 > < q3 > < q6 q7 >
Разбиение pi3: < q1 > < q2 > < q3 > < q4 > < q5 > < q6 > < q7 >
Разбиение pi4: < q1 > < q2 > < q3 > < q4 > < q5 > < q6 > < q7 >
~~~ Последнее разбиение совпало с полседним ~~~

        !!! Минимизация завершена !!!
```

Рисунок 2 – Ввод тестовых данных №1

```
--- Программа для минимизации автомата ---

Введите количество состояний исходного автомата (число от 1 до 10)
-> 7

Введите строки в следующем формате:
   1 число: <переход автомата для сигнала a>
   2 число: <переход автомата для сигнала b>
   3 число: <выходной сигнал для сигнала a>
   4 число: <выходной сигнал для сигнала b>
-> 2 3 0 1
-> 3 4 1 0
-> 4 5 0 1
-> 5 6 1 0
-> 6 7 0 1
-> 7 1 1 0
-> 1 2 0 1

~~~ Минимизация автомата ~~~
Разбиение pi0: < q1 q2 q3 q4 q5 q6 q7 >
Разбиение pi1: < q1 q3 q5 q7 > < q2 q4 q6 >
Разбиение pi2: < q1 q3 q5 > < q2 q4 q7 > < q6 >
Разбиение pi3: < q1 q3 q6 > < q2 q7 > < q4 > < q5 >
Разбиение pi4: < q1 q6 > < q2 > < q3 > < q4 > < q5 q7 >
Разбиение pi5: < q1 > < q2 > < q3 > < q4 q6 > < q5 > < q7 >
Разбиение pi6: < q1 > < q2 > < q3 > < q4 > < q5 > < q6 > < q7 >
Разбиение pi7: < q1 > < q2 > < q3 > < q4 > < q5 > < q6 > < q7 >
~~~ Последнее разбиение совпало с полседним ~~~

        !!! Минимизация завершена !!!
```

Рисунок 3 – Ввод тестовых данных №2

```
--- Программа для минимизации автомата ---

Введите количество состояний исходного автомата (число от 1 до 10)
-> 7

Введите строки в следующем формате:
   1 число: <переход автомата для сигнала a>
   2 число: <переход автомата для сигнала b>
   3 число: <выходной сигнал для сигнала a>
   4 число: <выходной сигнал для сигнала b>
-> 3 6 0 1
-> 6 2 0 1
-> 1 6 0 1
-> 2 3 1 0
-> 4 1 0 1
-> 3 1 1 0
-> 4 6 0 1

~~~ Минимизация автомата ~~~
Разбиение pi0: < q1 q2 q3 q4 q5 q6 q7 >
Разбиение pi1: < q1 q2 q3 q5 q7 > < q4 q6 >
Разбиение pi2: < q1 q3 > < q2 q5 > < q4 q6 > < q7 >
Разбиение pi3: < q1 q3 > < q2 > < q4 > < q5 > < q6 > < q7 >
Разбиение pi4: < q1 q3 > < q2 > < q4 > < q5 > < q6 > < q7 >
~~~ Последнее разбиение совпало с полседним ~~~

        !!! Минимизация завершена !!!
```

Рисунок 4 – Ввод тестовых данных №3

```
--- Программа для минимизации автомата ---

Введите количество состояний исходного автомата (число от 1 до 10)
-> 15
Ошибка! Автомат не может содержать заданное количество состояний.
Введите количество еще раз.
-> -5
Ошибка! Автомат не может содержать заданное количество состояний.
Введите количество еще раз.
->
```

Рисунок 5 – Ввод тестовых данных №4-5

ЗАКЛЮЧЕНИЕ

При выполнении домашнего задания изучен программный способ реализации конечного автомата. В ходе выполнения настоящего домашнего задания спроектирован и реализован цифровой автомат для минимизации конечного автомата с помощью высокоуровневого языка программирования Pascal. Отчет по проделанной работе оформлен с учетом требований ГОСТ 7.32.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Губарь А.М. Конспект лекций по дисциплине «Прикладная теория цифровых автоматов» – МГТУ им. Н.Э. Баумана, 2022;
2. Ожиганов А.А. Теория автоматов. Учебное пособие. – Санкт-Петербург, 2013.