TRƯỜNG ĐẠI HỌC KHOA HỌC
KHOA CÔNG NGHỆ THÔNG TIN

CHƯƠNG TRÌNH MÔ PHỎNG HOẠT ĐỘNG CỦA ÔTÔMAT HỮU HẠN ĐƠN ĐỊNH

THỰC TẬP VIẾT NIÊN LUẬN - 2021-2022.2.TIN3142.011
GIÁO VIÊN HƯỚNG DẪN: NGUYỄN THỊ BÍCH LỘC

SINH VIÊN THỰC HIỆN: NGUYỄN KIM THÀNH TÂN
MÃ SINH VIÊN: 19T1021229

HUẾ, THÁNG 8 NĂM 2022

MỤC LỤC

MỞ ĐẦU

Trong lí thuyết tính toán của ngành khoa học máy tính, những mô hình tính toán tương tự otomat hữu hạn đã được đề cập bởi Warren McCulloch and Walter Pitts vào năm 1943. Ứng dụng của Otomat hữu hạn được sử dụng rất nhiều trong các lĩnh vực như trong đơn vị điều khiển CPU, xử lí ngôn ngữ tự nhiên hay đơn giản là vận dụng trong thang máy hoặc tín hiệu đèn giao thông.

Để hiểu rõ hơn về cấu tạo, quy tắc hoạt động và mô phỏng hoạt động của nó, qua khuôn khổ bài báo cáo môn "Thực tập viết niên luận", em xin trình bày về "Chương trình mô phỏng hoạt động của ôtômát hữu hạn đơn định". Trong quá trình thực hiện bài báo cáo, nếu có sai xót, xin phép cô góp ý để bài báo cáo thêm hoàn thiện và chính xác. Em xin chân thành cảm ơn.

PHẦN BÀI LÀM

1. Cơ sở lí thuyết

1.1. Mở đầu

Một ôtômat hữu hạn là một mô hình tính toán thực sự hữu hạn. Mọi cái liên quan đến nó đều có kích thước hữu hạn cố định và không thể mở rộng trong suốt quá trình tính toán. Các loại ôtômat khác được nghiên cứu sau này có ít nhất một bộ nhớ vô hạn về tiềm năng. Sự phân biệt giữa các loại ôtômat khác nhau chủ yếu dựa trên việc thông tin có thể được đưa vào bộ nhớ như thế nào.

Một ôtômat hữu hạn làm việc theo thời gian rời rạc như tất cả các mô hình tính toán chủ yếu. Như vậy, ta có thể nói về thời điểm "kế tiếp" khi "đặc tả" hoạt động của một ôtômat hữu hạn.

Trường hợp đơn giản nhất là thiết bị không có bộ nhớ mà ở mỗi thời điểm, thông tin ra chỉ phụ thuộc vào thông tin vào lúc đó. Các thiết bị như vậy là mô hình của các mạch tổ hợp.

Tuy nhiên, nói chung, thông tin ra sản sinh bởi một ôtômat hữu hạn phụ thuộc vào cả thông tin vào hiện tại lẫn các thông tin vào trước đó. Như vậy ôtômat có khả năng (với một phạm vi nào đó) ghi nhớ các thông tin vào trong quá khứ của nó. Một cách chi tiết hơn, điều đó có nghĩa như sau.

Ôtômat có một số hữu hạn trạng thái bộ nhớ trong. Tại mỗi thời điểm i, nó ở một trong các trạng thái đó, chẳng hạn q_i. Trạng thái q_{i+1} ở thời điểm sau được xác định bởi q_i và thông tin vào a_i cho ở thời điểm i. Thông tin ra ở thời điểm i được xác định bởi trạng thái q_i (hay bởi cả a_i và q_i).

1.2. Định nghĩa

Một ôtômat hữu hạn đơn định hay một DFA (Deteministic Finite Automata) là một bộ năm

$$A = <Q, \Sigma, \delta, q_0, F>,$$

trong đó:

– Q là một tập hữu hạn khác rỗng, được gọi là tập các trạng thái;

– Σ là một bảng chữ, được gọi là bảng chữ vào;

– δ: $D \longrightarrow Q$, trong đó $D \subset Q \times \Sigma$, được gọi là ánh xạ chuyển;

– $q_0 \in Q$, được gọi là trạng thái đầu;

– $F \subset Q$, được gọi là tập các trạng thái kết thúc.

Trong trường hợp $D=Q \times \Sigma$, ta nói A là đầy đủ. Về sau ta sẽ thấy rằng mọi ôtômat hữu hạn đều đưa về được ôtômat hữu hạn đầy đủ tương đương.

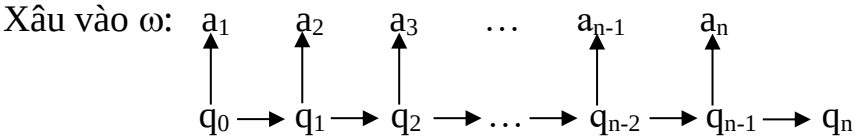
Hoạt động của ôtômat hữu hạn đơn định A = <Q, Σ, δ, q_0, F> khi cho xâu vào ω=a_1a_2… a_n có thể được mô tả như sau:

Khi bắt đầu làm việc, máy ở trạng thái đầu q_0 và đầu đọc đang nhìn vào ô có ký hiệu a_1. Tiếp theo máy chuyển từ trạng thái q_0 dưới tác động của ký hiệu vào a_1 về trạng thái mới $\delta(q_0, a_1)=q_1 \in Q$ và đầu đọc chuyển sang phải một ô, tức là nhìn vào ô có ký hiệu a_2. Sau đó ôtômat A có thể lại tiếp tục chuyển từ trạng thái q_1 nhờ ánh xạ chuyển δ về trạng thái mới $q_2=\delta(q_1, a_2)\in Q$. Quá trình đó sẽ tiếp tục cho tới khi gặp một trong các tình huống sau:

– Trong trường hợp ôtômat A đọc hết xâu vào ω và $\delta(q_{n-1},a_n)=q_n \in F$, ta nói rằng A đoán nhận ω.

– Trong trường hợp ôtômat A đọc hết xâu vào ω và $\delta(q_{n-1},a_n)=q_n \notin F$ hoặc tồn tại chỉ số j (j≤n) sao cho $\delta(q_{j-1},a_j)$ không xác định, ta nói rằng A không đoán nhận ω.

Q. Khi đó ôtômat dừng lại. Nếu $q_n \in F$ thì ta nói rằng ôtômat đã đoán nhận xâu ω.



1.3. Phương pháp biểu diễn ôtômat hữu hạn đơn định

Ánh xạ chuyển là một bộ phận quan trọng của một ôtômat hữu hạn đơn định. Nó có thể cho dưới dạng bảng chuyển hoặc cho dưới dạng đồ thị.

a. Phương pháp cho bảng chuyển:

Trạng thái	Ký hiệu vào			
	a_1	a_2	……..	a_n
q_1	$\delta(q_1,a_1)$	$\delta(q_1,a_2)$	……	$\delta(q_1,a_2)$
q_2	$\delta(q_2,a_1)$	$\delta(q_2,a_2)$	……	$\delta(q_2,a_2)$
q_3	$\delta(q_3,a_1)$	$\delta(q_3,a_2)$	……	$\delta(q_3,a_2)$
…	……...			
q_m	$\delta(q_m,a_1)$	$\delta(q_m,a_2)$	……	$\delta(q_m,a_2)$

trong đó dòng i cột j của bảng là ô trống nếu $(q_i,a_j)\notin D$, tức là $\delta(q_i,a_j)$ không xác định.

b. Phương pháp cho bằng đồ thị chuyển:

Cho ôtômat A = <Q, Σ, δ, q_0, F>. Ánh xạ chuyển δ có thể cho bằng một đa đồ thị có hướng, có khuyên G sau đây, được gọi là đồ thị chuyển của ôtômat A. Tập đỉnh của G là Q. Nếu a∈Σ và từ trạng thái q chuyển sang trạng thái p do đẳng thức δ(q, a)=p thì sẽ có một cung từ q tới p được gán nhãn a.

Đỉnh vào của đồ thị chuyển là đỉnh ứng với trạng thái ban đầu q_0. Các đỉnh sẽ được khoanh bởi các vòng tròn, tại đỉnh q_0 có mũi tên đi vào, riêng đỉnh với trạng thái kết thúc được khoanh bởi vòng tròn đậm.

Thí dụ 1: Cho hai ôtômat hữu hạn đơn định

$$A_1 = <\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\}>,$$

trong đó $\delta(q_0, a)=q_0$, $\delta(q_0, b)=q_1$, $\delta(q_1, a)=q_0$, $\delta(q_1, b)=q_2$, $\delta(q_2, a)=q_2$, $\delta(q_2, b)=q_2$ và

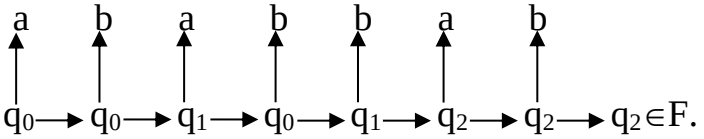
$$A_2 = <\{q_0, q_1, q_2, q_3\}, \{0, 1\}, \delta, q_0, \{q_0\}>,$$

trong đó $\delta(q_0, 0)=q_2$, $\delta(q_0, 1)=q_1$, $\delta(q_1, 0)=q_3$, $\delta(q_1, 1)=q_0$, $\delta(q_2, 0)=q_0$, $\delta(q_2, 1)=q_3$, $\delta(q_3, 0)=q_1$, $\delta(q_3, 1)=q_2$. Khi đó các bảng chuyển của A_1 và A_2 là:

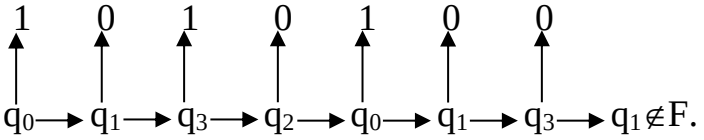
Trạng thái	Ký hiệu vào a	b
q_0	q_0	q_1
q_1	q_0	q_2
q_2	q_2	q_2

Trạng thái	Ký hiệu vào 0	1
q_0	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

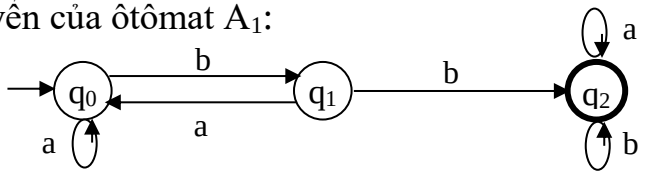
Dãy trạng thái của ôtômat A_1 khi cho xâu α=ababbab vào là:



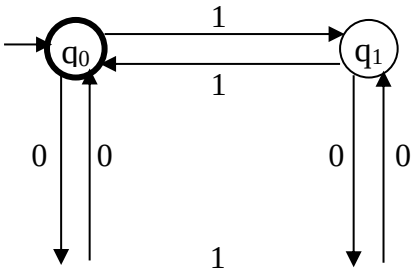
Dãy trạng thái của ôtômat A_2 khi cho xâu β=1010100 vào là:

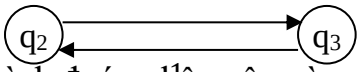


Đồ thị chuyển của ôtômat A_1:



Đồ thị chuyển của ôtômat A_2:





Ta có thể mô tả quá trình đoán nhận xâu vào của ôtômat hữu hạn đơn định đầy đủ A bằng thuật toán mô phỏng sau:

Đầu vào:

– Một xâu ω, kết thúc bởi ký hiệu hết File là eof.

– Một ôtômat hữu hạn đơn định đầy đủ A với trạng thái đầu q_0 và tập trạng thái kết thúc là F.

Đầu ra: "Đúng" nếu A đoán nhận xâu ω.

"Sai" nếu A không đoán nhận xâu ω.

Thuật toán:

```
Begin
    S:=q0;
    C:=ký hiệu tiếp theo;
    While C < > eof do
        begin
            S:=δ(S, C);
            C:=ký hiệu tiếp theo;
        end;
    if S in F return (True)
    else return (False);
End.
```

2. Chương trình mô phỏng hoạt động Ôtômat hữu hạn đơn định

2.1 Môi trường mô phỏng

- Trình soạn thảo: Visual Code
- Trình biên dịch: g++
- Trình hiển thị: Command Prompt
- Hệ điều hành: Windows
- Ngôn ngữ: C

2.2. Phát biểu bài toán

- Đầu vào là máy otomat A với tập các trạng thái, bảng chữ vào và tập các trạng thái kết thúc
  - Input:
    1. Số lượng các trạng thái (q_0, q_1,...., q_n)
    2. Trạng các trạng thái tiếp theo của trạng thái đó
    3. Cờ đánh dấu trạng thái đó có phải là trạng thái cuối cùng hay không
    4. Xâu đầu vào
  - Output:
    1. Kết quả mô phỏng hoạt động của otomat hữu hạn đơn định
    2. Kết quả đoán nhận xâu

Trạng thái	Kí hiệu vào		Trạng thái cuối
	0	1	
q_0*	q_0	q_1	0
q_1	q_0	q_2	0
q_2	q_2	q_2	1

*Bảng 1. Ví dụ input của chương trình (với * là đầu vào)*

2.3. Cấu trúc dữ liệu&thuật toán

- Dữ liệu ở đây là các danh sách liên kết các trạng thái để thuận tiện cho việc thực thi hoạt động của máy otomat, gồm 4 thành phần id_num, st_val, *link0 và link1;

```
struct node
{
  int id;
  int flag_end;
  struct node *link0;
  struct node *link1;
};

struct node *start, *q, *ptr;
int vst_arr[100], a[10];
```

- Khai báo kiểu cấu trúc node gồm 4 biến số:
  - id: kiểu dữ liệu int dùng để lưu id của trạng thái
  - flag_end: dùng để đánh dấu đánh trạng thái kết thúc (cờ trạng thái kết thúc với 0 – sai
    1 – đúng)
  - link0: con trỏ lưu địa chỉ ô nhớ của trạng thái tiếp theo nếu dữ liệu đầu vào của otomat là 0
  - link1: con trỏ lưu địa chỉ ô nhớ của trạng thái tiếp theo nếu dữ liệu đầu vào của otomat là 1
- Các con trỏ kiểu node:
  - start: con trỏ trỏ đến vị trí của trạng thái đầu vào trong tập trạng thái
  - q: con trỏ trỏ đến vị trí đầu tiên của tập trạng thái
  - ptr: con trỏ dùng để duyệt tập trạng thái khi nhận xâu đầu vào
- Các biến sử dụng:

```
int count, i, posi, j, loop;
  char n[100];
```

  - count: số lượng node sử dụng (số lượng trạng thái trong máy otomat)

- i, j, loop: dùng để sử dụng trong vòng lặp
- posi: dùng để gắn vị trí tiếp theo của các node

- Thuật toán:
  - Thuật toán sử dụng danh sách liên kết để duyệt đường đi của máy otomat dựa vào xâu đầu vào (qua việc sử dụng 2 biến con trỏ link0 và link 1)
  - Tóm tắt thuật toán như sau: (từ nay ta sẽ gọi các trạng thái là các *node*)
    1. Lấy số lượng node, và con trỏ trỏ tới các node tiếp theo tương ứng với đầu vào, đánh dấu cờ kết thúc
    2. Duyệt lần lượt các node từ trạng thái đầu
    3. Nếu đầu vào là 0, di chuyển con trỏ đến node của link 0, ngược lại di chuyển node của link1
    4. Tiếp tục đến khi hết xâu
    5. Nếu node hiện tại có flag_end = 1 thì đoán nhận sau, ngược lại thì không

2.4. Mã nguồn chương trình

Github: https://github.com/tanodd/dfa-in-c/

```
#include <stdio.h>
#include <stdlib.h>
#include <cstring>
#include <conio.h>
// #include <ctype.h>

/**
 * @id: id cua trang thai
 * @flag_end: flag trang thai cuoi -- 1 true -- o false
 * @link0: dia chi trang thai neu input la 0
 * @link1: dia chi trang thai neu input la 1
 */
struct node
{
  int id;
  int flag_end;
  struct node *link0;
  struct node *link1;
};

struct node *start, *q, *ptr;
int vst_arr[100], a[10];
int main()
{
  int count, i, posi, j, loop;
  char n[100];
  printf("\n\n");
  printf("\t-------------------------------------------------------
-----------------------------\n");
  printf("\t|\tCHUONG TRINH MO PHONG HOAT DONG CUA OTOMAT HUU HAN
DON DINH\t\t|\n");
  printf("\t|\tHo va ten: NGUYEN KIM THANH TAN -
19T1021229\t\t\t\t|\n");
  printf("\t|\tGiao vien huong dan: NGUYEN THI BICH LOC
\t\t\t\t|\n");
  printf("\t-------------------------------------------------------
-----------------------------\n");


  printf("\n");
  printf("\t\tHay nhap so luong cua tap trang thai Q\n\t\t");
  scanf("%d", &count);
```

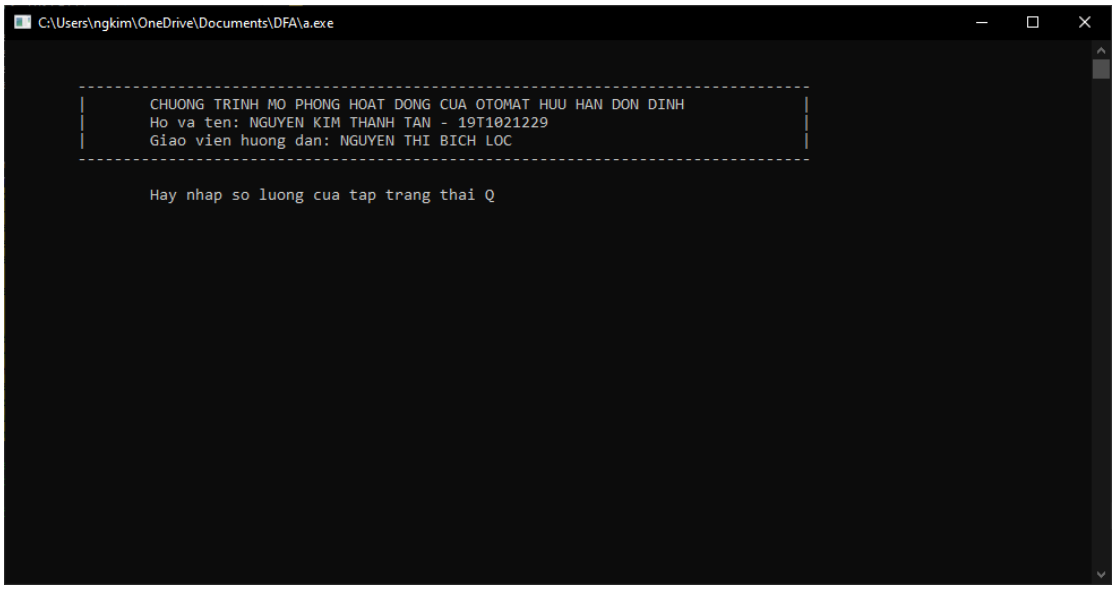
```
  q = (struct node *)malloc(sizeof(struct node) * count); // Cap
phat dong cho bo nho cho q

  for (i = 0; i < count; i++)
  {
    (q + i)->id = i;

    printf("\t\tTrang thai q%d\n", i);
    printf("\t\tTrang thai tiep theo neu dau vao la 0: \t\t");
    scanf("%d", &posi);
    (q + i)->link0 = (q + posi);

    printf("\t\tTrang thai tiep theo neu dau vao la 1: \t\t");
    scanf("%d", &posi);
    (q + i)->link1 = (q + posi);

    printf("\t\tTrang thai nay co thuoc trang thai ket
thuc?\n\t\t");
    scanf("%d", &(q + i)->flag_end);
  }

  printf("\t\tHay nhap ten trang thai dau:\n\t\t");
  scanf("%d", &posi);
  start = q + posi;
  printf("\t\t--------------------------------\n");



    ptr = start;
    printf("\t\tHay nhap xau dau vao \n\t\t");
    scanf("%s", n);
    posi = 0;
    system("cls");

    while (n[posi] != '\0')
    {
      a[posi] = (n[posi] - '0');
      posi++;
    }
    i = 0;


    // Display result
    printf("\t\t");
    for (loop = 0; loop < strlen(n); loop++)
```

```
    {
      printf("[%d]    ", a[loop]);
    }
    printf("\n");
    printf("\t\t");
    for (loop = 0; loop < strlen(n); loop++)
    {

      printf(" ^     ");
    }
    printf("\n");
    printf("\t\t");
    for (loop = 0; loop < strlen(n); loop++)
    {
      printf(" |     ");
    }
    printf("\n");
    printf("\t\t");
    do
    {
      vst_arr[i] = ptr->id;
      if (a[i] == 0)
      {
        ptr = ptr->link0;
      }
      else if (a[i] == 1)
      {
        ptr = ptr->link1;
      }
      else
      {
        system("cls");
        printf("\t\t dau vao sai!\n"); // Kiem tra chuoi dau vao
gia tri khac 0 hoac 1 hay khong
        return 0;
      }
      printf("[q%d]-->", vst_arr[i]);
      i++;
    } while (i < posi);
    //*********************************************
    printf("[q%d]", ptr->id);
    if (ptr->flag_end == 1)
      printf(" thuoc F");
    else
      printf(" khong thuoc F");
    printf("\n");
```

```
    printf("\n");
    if (ptr->flag_end == 1)
    {
      printf("\t\tDOAN NHAN xau %s\n", n);
      getch();
      return 1;
    }
    else
    {
      printf("\t\tKHONG doan nhan xau %s\n", n);
      getch();
      return 0;
    }

}
```

2.5. Kết quả thực hiện

- Màn hình console khi chạy chương trình

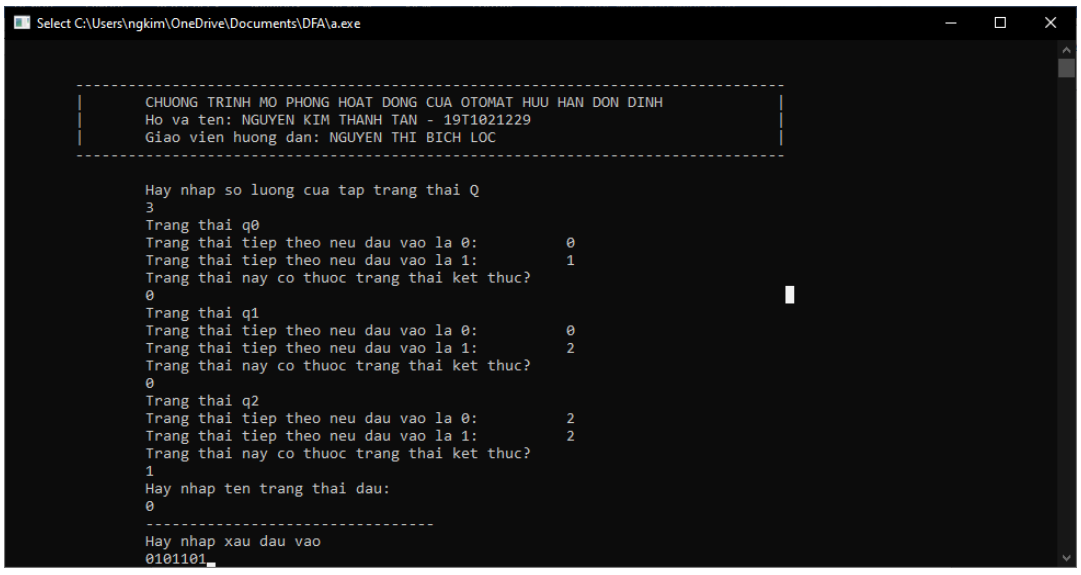


- Nhập đầu vào chương trình

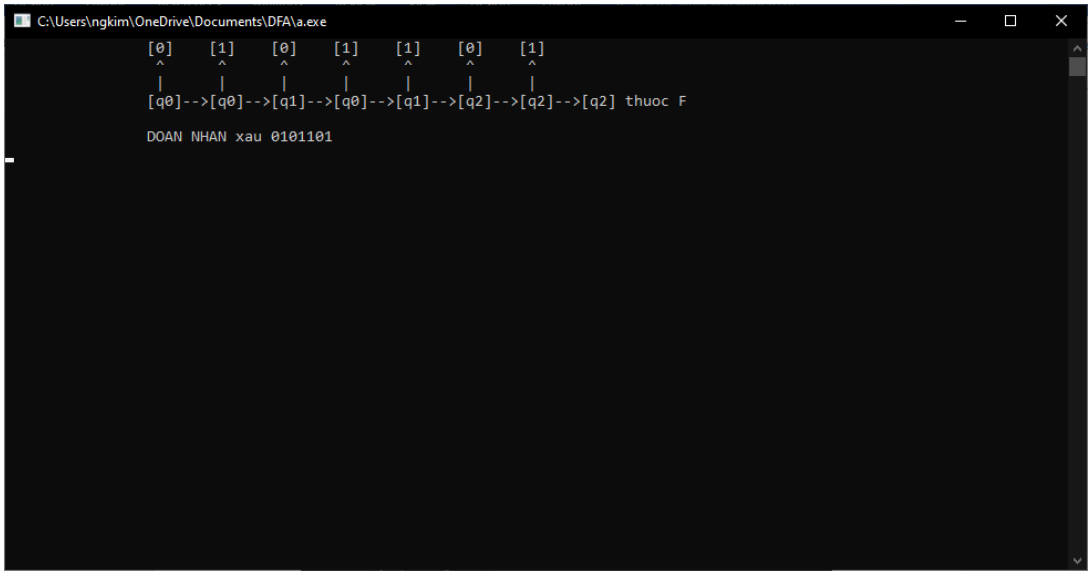
  Test 1:

Trạng thái	Kí hiệu vào		Trạng thái cuối
	0	1	
q0*	q0	q1	0
q1	q0	q2	0
q2	q2	q2	1

Chuỗi đầu vào: 0101101



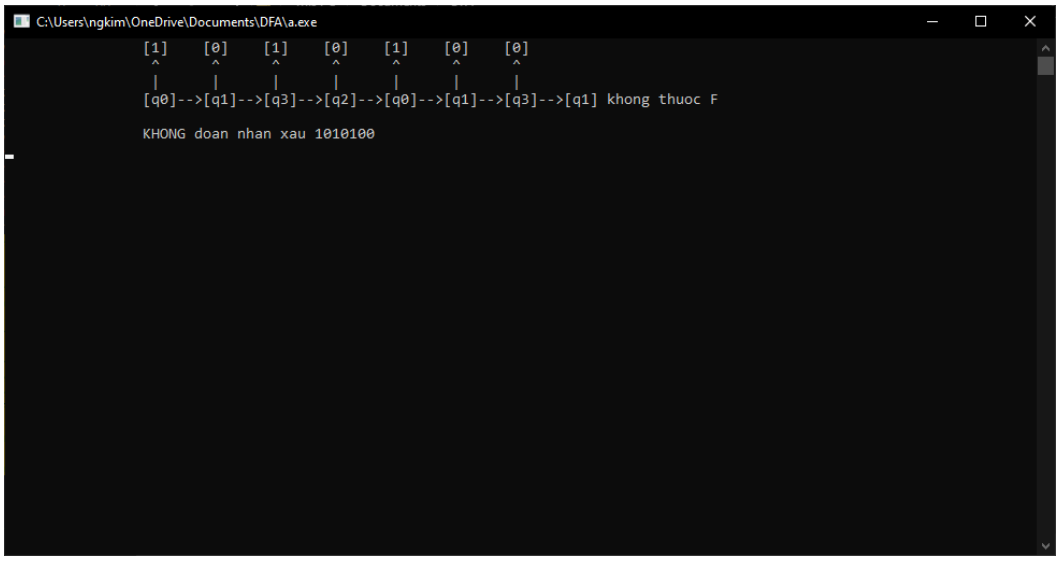
Kết quả:



Test 2: Chuỗi đầu vào: 1010100

Trạng thái	Kí hiệu vào		Trạng thái cuối
	0	1	
q0*	q1	q2	1
q1	q3	q0	0
q2	q0	q3	0
q3	q1	q2	0

Kết quả



2.6. Nhận xét đánh giá kết quả

Qua kết quả nhận được từ chương trình cho thấy chương trình đã mô phỏng được hoạt động của otomat hữu hạn đơn định. Vì đây là một máy otomat đơn giản nên chương trình cũng rất đơn giản, dễ hiểu và dễ thực thi

3. Kết luận và hướng phát triển

3.1. Kết luận

- Chương trình đã đáp ứng được yêu cầu của bài toán
- Đã học được kĩ năng trình bày và viết báo cáo

3.2. Hướng phát triển

- Sẽ nâng cấp chương trình để có thể mô phỏng nhiều loại otomat hơn

TÀI LIỆU THAM KHẢO

[1] Giáo trình OTOMAT và ngôn ngữ hình thức – PGS.TS Nguyễn Văn Định
[2] SimulateDFA - @github/subarno.
https://github.com/subarnop/SimulateDFA
[3] Applications of Deterministic Finite Automata - Eric Gribkof
https://www.cs.ucdavis.edu/~rogaway/classes/120/spring13/eric-dfa.pdf