TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT VĨNH LONG

KHOA CÔNG NGHỆ THÔNG TIN

--ଓ📖ଌ--

Tiểu Luận

Xây dựng bộ công cụ cho phép vẽ các đồ thị chuyển trạng thái (các đối tượng trong đồ thị có thể thay đổi vị trí được), nhập các bảng truyền của các DFA, NFA. Lưu trữ và hiển thị lại

Sinh viên thực hiện: Đàm Ngọc Đại -18004016

Lớp: 1CTT18A1

Khóa: K43

Học kỳ 2: 2019 – 2020

Giảng viên hướng dẫn: Ths. Nguyễn Thị Hồng Yến

Vĩnh Long – năm 2020

Mục lục

NHẬN XÉT ĐÁNH GIÁ ĐIỂM CỦA NGƯỜI HƯỚNG DẪN

- Ý thức thực hiện:

……..……

…….……

……..…

- Nội dung thực hiện:

……..……

…….……

……..…

- Hình thức trình bày:

……..……

…….……

……..

- Tổng hợp kết quả:

……..……

…….……

……..…

Vĩnh Long, ngày……tháng……năm 2020

Giảng viên hướng dẫn

Phần 1: Giới thiệu về ÔTÔMÁT HỮU HẠN (FA: Finite Automata)

Ôtômát hữu hạn FA là một mô hình tính toán của hệ thống với sự mô tả bởi các input và output. Tại mỗi thời điểm, hệ thống có thể được xác định ở một trong số hữu hạn các cấu hình nội bộ gọi là các trạng thái (states). Mỗi trạng thái của hệ thống thể hiện sự tóm tắt các thông tin liên quan đến những input đã chuyển qua và xác định các phép chuyển kế tiếp trên dãy input tiếp theo. Trong khoa học máy tính, ta có thể tìm thấy nhiều ví dụ về hệ thống trạng thái hữu hạn, và lý thuyết về ôtômát hữu hạn là một công cụ thiết kế hữu ích cho các hệ thống này. Chẳng hạn, một hệ chuyển mạch như bộ điều khiển (Control Unit) trong máy tính. Một chuyển mạch thì bao gồm một số hữu hạn các cổng (gate) input, mỗi cổng có 2 giá trị 0 hoặc 1. Các giá trị đầu vào này sẽ xác định 2 mức điện thế khác nhau ở cổng output. Mỗi trạng thái của một mạng chuyển mạch với n cổng bất kỳ sẽ là một trường hợp trong 2n phép gán của 0 và 1 đối với các cổng khác nhau. Các chuyển mạch thì được thiết kế theo cách này, vì thế chúng có thể được xem như hệ thống trạng thái hữu hạn. Các chương trình sử dụng thông thường, chẳng hạn trình soạn thảo văn bản hay bộ phân tích từ vựng trong trình biên dịch máy tính cũng được thiết kế như các hệ thống trạng thái hữu hạn. Ví dụ bộ phân tích từ vựng sẽ quét qua tất cả các dòng ký tự của chương trình máy tính để tìm nhóm các chuỗi ký tự tương ứng với một tên biến, hằng số, từ khóa, …Trong quá trình xử lý này, bộ phân tích từ vựng cần phải nhớ một số hữu hạn thông tin như các ký tự bắt đầu hình thành những chuỗi từ khóa. Lý thuyết về ôtômát hữu hạn thường được dùng đến nhiều cho việc thiết kế các công cụ xử lý chuỗi hiệu quả. Máy tính cũng có thể được xem như một hệ thống trạng thái hữu hạn. Trạng thái hiện thời của bộ xử lý trung tâm, bộ nhớ trong và các thiết bị lưu trữ phụ ở mỗi thời điểm bất kỳ là một trong những số rất lớn và hữu hạn của số trạng thái. Bộ não con người cũng là một hệ thống trạng thái hữu hạn, vì số các tế bào thần kinh hay gọi là neurons là số có giới hạn, nhiều nhất có thể là 235. Lý do quan trọng nhất cho việc nghiên cứu các hệ thống trạng thái hữu hạn là tính tự nhiên của khái niệm và khả năng ứng dụng đa dạng trong nhiều lĩnh vực thực tế. Ôtômát hữu hạn (FA) được chia thành 2 loại: đơn định (DFA) và không đơn định (NFA). Cả hai loại ôtômát hữu hạn đều có khả năng nhận dạng chính xác tập chính quy. Ôtômát hữu hạn đơn định có khả năng nhận dạng ngôn ngữ dễ dàng hơn ôtômát hữu hạn không đơn định, nhưng thay vào đó thông thường kích thước của nó lại lớn hơn so với ôtômát hữu hạn không đơn định tương đương.

1.1 Khái niệm về otomat hữu hạn đơn định (DFA)

1.1.1 Định nghĩa otomat hữu hạn đơn định (Deterministic Finite Automata-DFA)

Là một bộ năm: $A = < Q, \Sigma, \delta, q0, F >$ trong đó: . Q là một tập hữu hạn khác rỗng, được gọi là tập các trạng thái (set of states); . Σ là một bảng chữ cái, được gọi là bảng chữ vào (input alphabet); . δ: $D \rightarrow Q$, là một ánh xạ từ $D \subseteq Q \times \Sigma$ vào Q, được gọi là hàm chuyển trạng thái (transition function); . $q0 \in Q$, được gọi là trạng thái khởi đầu (starting state); . $F \subseteq Q$ được gọi là tập trạng thái kết thúc (final states). Nếu $D = Q \times \Sigma$, tức là hàm hai biến $\delta(q, a)$ xác định với mọi cặp giá trị (q, a), với $q \in Q$, $a \in \Sigma$ thì ta nói A là otomat đầy đủ, trái lại, nếu có những cặp giá trị (q, a) mà tại đó hàm chuyển $\delta(q, a)$ không xác định, thì otomat A gọi là không đầy đủ.

Sau này ta sẽ thấy rằng mọi otomat hữu hạn đều đưa về được otomat hữu hạn đầy đủ tương đương. Hoạt động của otomat hữu hạn đơn định A = với một xâu vào ω = a1a2… an có thể được mô tả như sau: Khi bắt đầu làm việc, otomat ở trạng thái khởi đầu q0 và đầu đọc đang nhìn vào ô có ký hiệu a1. Tiếp theo, dưới tác động của ký hiệu vào a1, otomat chuyển từ trạng thái q0 về trạng thái mới q1 xác định bởi hàm chuyển, tức là δ(q0, a1) = q1∈ Q, và đầu đọc chuyển sang phải một ô, tức là nhìn vào ô có ký hiệu a2. Sau đó otomat A có thể lại tiếp tục chuyển từ trạng thái q1 nhờ hàm chuyển δ về trạng thái mới q2 = δ(q1, a2). Quá trình đó sẽ tiếp tục cho tới khi gặp một trong các tình huống sau:

a). Otomat A đọc hết xâu vào ω và chuyển sang trạng thái qn, tức là A ở trạng thái qn-1, đọc ký hiệu cuối cùng an và có δ(qn-1,an) = qn , khi đó nếu qn∈ F, ta nói rằng A đoán nhận xâu ω, trái lại nếu qn∉ F thì ta nói otomat A không đoán nhận xâu ω.

b). Hoặc khi otomat A ở trạng thái qi nào đó, đọc đến ký hiệu aj của xâu vào, (j ≤ n) và hàm chuyển δ(qi , aj) không xác định, khi đó otomat A dừng, ta cũng nói A không đoán nhận xâu ω.

Chú ý rằng nếu A là otomat đầy đủ thì tình huống a). luôn xảy ra, tức là xâu ω luôn được đọc hết, còn tình huống b). chỉ có thể xảy ra khi A là otomat không đầy đủ.

1.1.2 Ngôn ngữ được chấp nhận bởi DFA

Một chuỗi w được chấp nhập bởi ôtômát hữu hạn M (Q, Σ, δ, q_0, F) nếu δ(q_0, w) = p với p **c** F. Ngôn ngữ được chấp nhận bởi M, ký hiệu L(M) là tập hợp:

$$L(M) = \{ w \mid \delta(q_0, w) \in F \}$$

Ví dụ 1.1.2: Xây dựng DFA đoán nhận ngôn ngữ:

L = {w | w chỉ chứa ký hiệu 0, 1 và luôn kết thúc bởi ký hiệu 1}

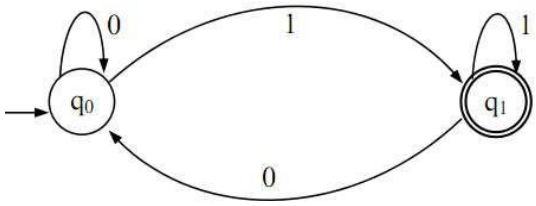
DFA phải phân biệt 2 khả năng:

1) Khi đọc ký hiệu 1, thì ngay sau đó có thể kết thúc
2) Khi đọc ký hiệu 0, thì ngay sau đó không được kết thúc, mà phải đọc tiếp các ký hiệu khác cho đến khi gặp ký hiệu 1

Như thế, DFA sẽ có ít nhất 2 trạng thái tương ứng với hai khả năng trên.

Ta xây dựng DFA như sau:

M = ({q_0, q_1},{0, 1}, d, q_0,{q_1}), trong đó hàm d như sau:



Hay chúng ta có thể biểu diễn hàm d bởi bảng dịch chuyển như sau:

δ	Inputs	
Trạng thái	0	1
q_0	q_0	q_1
q_1	q_0	q_1

Bây giờ, chúng ta sử dụng hàm dịch chuyển mở rộng để đoán nhận chuỗi w = 101101 của ngôn ngữ L.

d(q_0, e) = q_0

d(q_0, 1) = d(d(q_0, e), 1) = d(q_0, 1) = q_1

d(q_0, 10) = d(d(q_0, 1), 0) = d(q_1, 0) = q_0

d(q_0, 101) = d(d(q_0, 10), 0) = d(q_0, 1) = q_1

d(q_0, 1011) = d(d(q_0, 101), 1) = d(q_1, 1) = q_1

d(q_0, 10110) = d(d(q_0, 1011), 0) = d(q_1, 0) = q_0

d(q_0, 101101) = d(d(q_0, 10110), 1) = d(q_0, 1) = q_1

Như thế, chuỗi vào w = 101101 được đoán nhận bởi DFA.

Chúng ta có thể hiểu quá trình đoán nhận đơn giản như sau:

$q_0$101101 g $q_1$01101 g $q_0$1101 g $q_1$101 g $q_1$01 g $q_0$1 g q_1 c F

Nhận xét: Đối với một chuỗi vào w, DFA chỉ cho một quá trình đoán nhận duy nhất.

Giải thuật mô phỏng hoạt động của một DFA

```
. Input: Chuỗi nhập x kết thúc bởi $

. Output: Câu trả lời "YES" nếu DFA chấp nhận chuỗi x
và "NO" nếu ngược lại.

. Giải thuật:

     q:= q0 ;

     c:= nextchar ; {c là ký hiệu nhập được đọc tiếp
     theo}

     While q <> $ do
```

Nhận xét: Một cách tổng quát, ta thấy tập Q của DFA thể hiện các trạng thái lưu trữ của ôtômát trong quá trình đoán nhận ngôn ngữ, và như vậy khả năng lưu trữ của ôtômát là hữu hạn. Mặt khác, hàm chuyển δ là hàm toàn phần và đơn trị, cho nên các bước chuyển của ôtômát luôn luôn được xác định một cách duy nhất. Chính vì hai đặc điểm này mà DFA mô tả như trên được gọi là ôtômát hữu hạn đơn định.

1.2 Khái niệm về otomat hữu hạn không đơn định

Một ôtômát hữu hạn không đơn định có khả năng ở trong nhiều trạng thái khác nhau ở tại một thời điểm. Sự khác nhau cơ bản giữa DFA và NFA là định nghĩa của hàm dịch chuyển. Đối với NFA, hàm dịch chuyển nhận một trạng thái và một ký hiệu nhập và trả về một tập hợp (có thể rỗng) các trạng thái (thay vì trả về đúng 1 trạng thái đối với DFA)

1.2.1 Định nghĩa otomat hưu hạn không đơn định (Nondeterministic Finite Automata-NFA)

Một cách hình thức ta định nghĩa ôtômát hữu hạn không đơn định NFA là một bộ gồm 5 thành phần $(Q, \Sigma, \delta, q_0, F)$ trong đó Q, Σ, q_0 và F có ý nghĩa như trong DFA, nhưng δ là hàm chuyển ánh xạ từ $Q \times \Sigma \rightarrow 2^Q$.

Khái niệm δ(q, a) là tập hợp tất cả các trạng thái p sao cho có phép chuyển trên nhãn a từ trạng thái q tới p.

Hàm chuyển trạng thái mở rộng: Để thuận tiện trong việc mô tả hoạt động ôtômát trên chuỗi, ta mở rộng hàm chuyển δ ánh xạ từ $Q \times \Sigma^* \rightarrow 2^Q$ như sau:

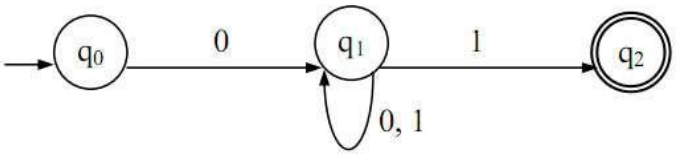
1. $\delta(q, \varepsilon) = \{q\}$
2. $\delta(q, wa) = \{ p \mid$ có một trạng thái r trong $\delta(q, w)$ mà p thuộc $\delta(r, a)\}$

   $= \delta(\delta(q, w), a)$
3. $\delta(P, w) = \cup_{q \in P} \delta(q, w)$, "P ⊂ Q

1.2.2 Ngôn ngữ được chấp nhận bởi NFA

Ngôn ngữ L(M), với M là ôtômát hữu hạn không đơn định NFA $(Q, \Sigma, \delta, q_0, F)$ là tập hợp: $L(M) = \{w \mid \delta(q_0, w)$ có chứa một trạng thái trong F }

bắt đầu bởi ký hiệu 0 và kết thúc bởi ký hiệu 1. NFA được xây dựng như trong hình .



Giả sử chuỗi vào là w = 0101101, quá trình đoán nhận chuỗi w như sau:

$q_0 0101101 \vdash q_1 101101 \vdash q_1 01101 \vdash q_1 1101 \vdash q_1 101 \vdash q_1 01 \vdash q_1 1 \vdash q_2 \in F$.

Tuy nhiên, chúng ta có thể dễ dàng nhận thấy rằng, quá trình đoán nhận trên chỉ là một trong những quá trình đoán nhận

Nhận xét: Đối với một chuỗi vào w, NFA có thể có nhiều quá trình đoán nhận khác nhau, trong khi DFA thì chỉ có một quá trình đoán nhận.

1.3 SỰ TƯƠNG ĐƯƠNG GIỮA ÔTÔMÁT HỮU HẠN VÀ BIỂU THỨC CHÍNH QUY

Trong chương trước ta đã thấy rằng với mọi ngôn ngữ chính quy đều tồn tại một văn phạm chính quy sinh ra nó, và ngược lại ngôn ngữ sinh bởi văn phạm chính quy là ngôn ngữ chính quy.

Trong phần này, ta sẽ thấy có một sự liên hệ tương tự như vậy giữa ôtômát hữu hạn và ngôn ngữ chính quy

***Định lý 1.3.1*: Nếu r là biểu thức chính quy thì tồn tại một NFA với ε-dịch chuyển chấp nhận L(r).**

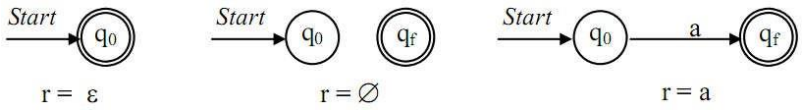
Chứng minh

Ta sẽ chứng minh quy nạp theo số phép toán của biểu thức chính quy r rằng có tồn tại một NFA M với ε-dịch chuyển có một trạng thái kết thúc và không có các phép chuyển khỏi trạng thái này chấp nhận biểu thức chính quy r: L(M) = L(r).

- **r không có phép toán**

Vậy r phải là $\varnothing$, ε hoặc a (với $a \in \Sigma$).

Các NFA dưới đây thoả mãn điều kiện:



- **r có chứa các phép toán**

Giả sử định lý đúng với r có ít hơn i phép toán, $i \geq 1$.

Xét r có i phép toán. Có 3 trường hợp:

1) r = r1 + r2

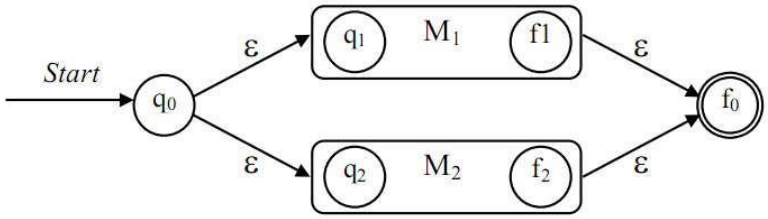
Cả hai biểu thức chính quy r_1, r_2 có ít hơn i phép toán, vậy ta có 2 ôtômát hữu hạn

NFA là M_1 (Q_1, Σ_1, δ_1, q_1, {f_1}) và M_2 (Q_2, Σ_2, δ_2, q_2, {f_2}) sao cho $L(M_1) = L(r_1)$ và $L(M_2) = L(r_2)$. Vì các trạng thái có thể thay đổi tên nên ta giả sử hai tập trạng thái Q_1 và Q_2 là rời nhau. Đặt q_0 là trạng thái bắt đầu mới và {f_0} là tập trạng thái kết thúc mới, ta xây dựng *NFA M* (Q_1 Q_2 {q_0, f_0}, Σ_1 Σ_2, δ, q_0, {f_0}), trong đó δ được xác định như sau:

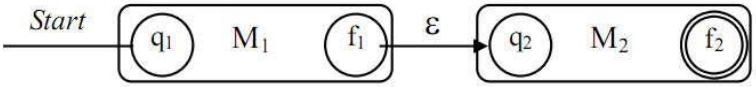
- $\delta(q_0, \varepsilon) = \{q_1,$
- $\delta(q, a) = \delta_1(q, a)$ với q $Q_1 - \{f_1\}$ và a Σ_1 $\{\varepsilon\}$
- $\delta(q, a) = \delta_2(q, a)$ với q $Q_2 - \{f_2\}$ và a Σ_2 $\{\varepsilon\}$
- $\delta(f_1, \varepsilon) = \delta(f_2, \varepsilon) = \{f_0\}$

Chú ý do giả thiết quy nạp là không có phép chuyển nào ra khỏi f_1, f_2 trong M_1, M_2. Vì vậy tất cả các phép chuyển của M_1 và M_2 đều có trong M. Cách xây dựng M chỉ ra trong hình a. Bất kỳ đường đi nào trong sơ đồ chuyển của M từ q_0 tới f_0 phải bắt đầu bằng cách đi tới q_1 hoặc q_2 bằng nhãn ε. Nếu đường đi qua q_1 thì nó theo một đường đi nào đó trong M_1 tới f_1 rồi sau đó tới f_0 bằng nhãn ε. Tương tự trong trường hợp đường đi qua q_2. Có một đường đi từ q_0 đến f_0 nhãn x khi và chỉ khi có đường đi nhãn x trong M_1 từ q_1 đến f_1 hoặc có đường đi nhãn x trong M_2 từ q_2 đến f_2.

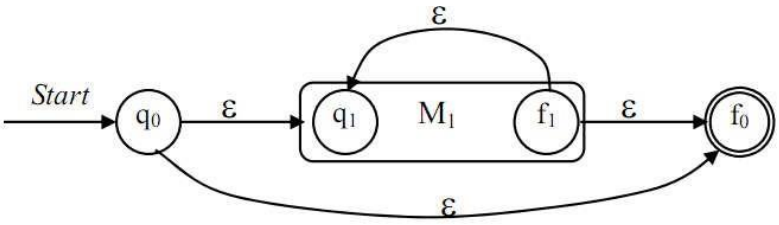
Vậy $L(M) = L(M_1) \cup L(M_2)$



Hình a. Phép hợp



Hình b. Phép kết hợp



Hình c. Phép bao đóng

1) r = r1 r2

Đặt M_1 và M_2 là các ôtômát NFA như trong trường hợp trên và ta xây dựng một ôtômát *M (Q, Σ, δ, {q_1}, {f_2})*, trong đó δ được xác định như sau:

- $\delta(q, a) = \delta_1(q, a)$ *với* $q \in Q_1 - \{f_1\}$ *và* $a \in \Sigma_1 \cup \{\varepsilon\}$
- $\delta(f_1, \varepsilon) = \{q_2\}$
- $\delta(q, a) = \delta_2(q, a)$ *với* $q \in Q_2$ *và* $a \in \Sigma_2 \cup \{\varepsilon\}$

Cách xây dựng M chỉ ra trong hình b của hình 3.9. Mỗi đường đi trong M từ q_1 tới f_2 là đường đi có nhãn x từ q_1 tới f_1 sau đó là một cung từ f_1 tới q_2 nhãn ε và tiếp đến là đường đi từ q_2 tới f_2.

Vậy $L(M) = \{xy \mid x \in L(M_1) \text{ và } y \in L(M_2)\}$ hay $L(M) = L(M_1)\,L(M_2)$.

2) $r = r^*$

Đặt $M_1 (Q_1, \Sigma_1, \delta_1, q_1, \{f_1\})$ và $L(M_1) = r_1$.
Xây dựng $M (Q_1 \cup \{q_0, f_0\}\ \Sigma_1, \delta, q_0, \{f_0\})$, trong đó δ được cho:

- $\delta(q_0, \varepsilon) = \delta(f_1, \varepsilon) = \{q_1, f_0\}$

- $\delta(q, a) = \delta_1(q, a)$ *với* $q \in Q_1 - \{f_1\}$ *và* $a \in \Sigma_1 \cup \{\varepsilon\}$

Cách xây dựng M được chỉ ra trong hình c của hinhg 3.9. Mỗi đường đi từ q_0 tới f_0 gồm: hoặc đường đi từ q_0 tới f_0 bằng nhãn ε; hoặc đường đi từ q_0 tới q_1 bằng nhãn ε và sau đó là đường đi từ q_1 tới f_1 trên chuỗi thuộc L(M), rồi đến f_0 bằng nhãn ε. Như vậy có đường đi từ q_0 tới f_0 nhãn là x nếu và chỉ nếu ta có thể viết $x = x_1 x_2 \ldots x_j$ với $j \geq 0$ (trường hợp j = 0 khi x = ε) $\forall x_i \in L(M_1)$. Vậy $L(M) = L(M_1)^*$.

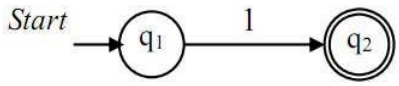
Ví dụ 3.14: Xây dựng NFAε chấp nhận lớp ngôn ngữ được ký hiệu bởi biểu thức chính quy $r = 01^* + 1$.

Ta thấy $L(r) = \{1, 0, 01, 011, 0111, 01111, 011111, \ldots\}$ là tập ngôn ngữ chứa các b_i đơn 0, 1 và các chuỗi bit nhị phân bắt đầu bằng bit 0, theo sau là một chuỗi bit 1 với độ dài tuỳ ý.

Theo quy luật thứ tự ưu tiên, biểu thức $01^* + 1$ thực chất là $(0(1^*)) + 1$, vì vậy nó có dạng $r_1 + r_2$ với $r_1 = 01^*$ và $r_2 = 1$.

Ta sẽ lần lượt xây dựng các NFA cho các biểu thức chính quy con, sau đó dựa vào các quy tắc kết hợp để xây dựng NFA cho toàn bộ biểu thức chính quy đã cho.

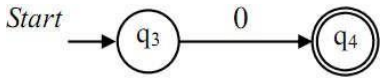
- NFA cho r = 1 dễ dàng để xây dựng:



- NFA cho $r_1 = 01^*$

  Ta tách $r_1 = r_3\, r_4$, trong đó $r_3 = 0$ và $r_4 = 1^*$

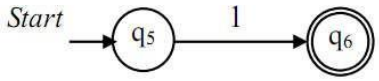
  + NFA cho $r_3 = 0$:



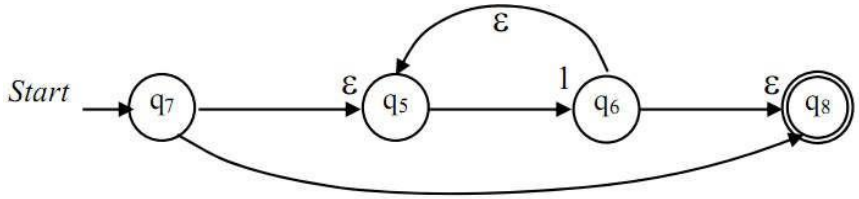
  + NFA $r_4 = 1^*$:

    Ta viết $r_4 = r_5^*$, trong đó $r_5 = 1$.

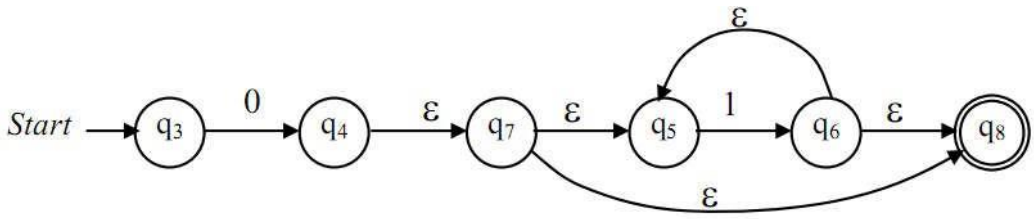
    NFA cho $r_5 = 1$



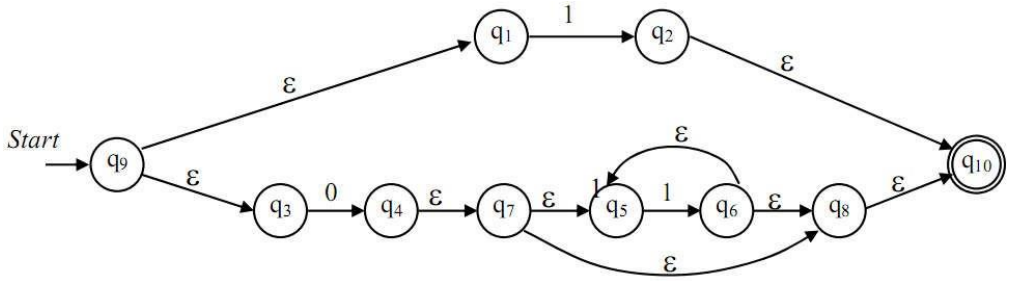
Theo quy tắc 3) ta xây dựng được NFA cho $r_4 = r_5^* = 1^*$ như sau:



Theo quy tắc 2) ta xây dựng được NFA cho $r_1 = r_3\, r_4 = 01^*$ như sau:



Cuối cùng, theo quy tắc 1) ta xây dựng NFA cho $r = r_1 + r_2 = 01^* + 1$ như sau:



Phần chứng minh của Định lý 3.3 trên cũng chính là cơ sở của giải thuật chuyển đổi một biểu thức chính quy thành ôtômát hữu hạn. Một điểm cần lưu ý là thứ tự ưu tiên của các phép toán được sử dụng trong biểu thức chính quy, điều này rất quan trọng cho quá trình tách biểu thức chính quy thành các biểu thức con trong những trường hợp viết biểu thức chính quy ở dạng tắt (không có dấu ngoặc).

Bây giờ, ta cần chứng tỏ rằng mọi tập hợp được chấp nhận bởi một ôtômát hữu hạn thì cũng được ký hiệu bởi một số biểu thức chính quy.

***Định lý 1.3.2*: Nếu L được chấp nhận bởi một DFA, thì L được ký hiệu bởi một biểu thức chính quy.**

Chứng minh

Đặt L là tập hợp được chấp nhận bởi DFA M ($\{q_1, q_2, ..., q_n\}$, Σ, δ, q_1, F). Đặt R^k_{ij} là tập hợp tất cả các chuỗi x sao cho $\delta(q_i, x) = q_j$ và nếu $\delta(q_i, y) = q_l$, với y là tiền tố bất kỳ của x, khác x hoặc ε, thì $l \leq k$. Tức là R^k_{ij} là tập hợp tất cả các chuỗi làm cho ôtômát đi từ trạng thái qi tới qj không đi ngang qua trạng thái nào (được đánh số) lớn hơn k. (Chú ý, khái niệm "đi ngang qua một trạng thái" có nghĩa là có phép chuyển vào và ra khỏi trạng thái đó, nên i hoặc j đều có thể lớn hơn k). Vì chỉ có n trạng thái nên R^n_{ij} sẽ là tập hợp tất cả các chuỗi làm ôtômát đi từ q_i tới q_j.

Ta định nghĩa R^k_{ij} một cách đệ quy như sau:

$$R^0_{ij} = \begin{cases} R^k_{ij} = R^{k-1}_{ik} (R^{k-1}_{kk})^* R^{k-1}_{kj} \cup R^{k-1}_{ij} \quad (1) \\ \{ a \mid \delta(q_i, a) = q_j \} \cup \{\varepsilon\} \text{ nếu } i \end{cases}$$

Một cách hình thức, R^k_{ij} định nghĩa như trên là các chuỗi nhập hay nguyên nhân đưa M từ q_i tới q_j không đi ngang qua trạng thái cao hơn q_k, nghĩa là xảy ra hoặc một trong hai trường hợp sau:

1) Nằm trong R^{k-1}_{ij} (để không bao giờ đi ngang qua một trạng thái nào cao như q_k).
2) Bao gồm một chuỗi trong R^{k-1}_{ik} (chuỗi làm M chuyển đến q_k), theo sau bởi không hoặc nhiều chuỗi trong R^{k-1}_{kk} (chuỗi làm M chuyển từ q_k trở về q_k mà không ngang qua q_k hoặc một trạng thái nào cao hơn) và cuối cùng là một chuỗi trong R^{k-1}_{kj} (chuỗi làm M chuyển từ q_k đến q_j).

Ta sẽ chỉ ra rằng với mỗi i, j và k tồn tại biểu thức chính quy r^k_{ij} ký hiệu cho ngôn ngữ R^k_{ij}. Ta quy nạp theo k như sau:

- *k = 0: khi đó R^0_{ij} là tập hợp hữu hạn các chuỗi có một ký hiệu hoặc ε. Vậy r^0_{ij} có thể viết dưới dạng $a_1 + a_2 + ... + a_p$ (hoặc $a_1 + a_2 + ... + a_p + \varepsilon$ nếu i = j). Trong đó $\{a_1, a_2, ..., a=\}$ là tập hợp tất cả các ký hiệu a sao cho $\delta(q_i, a) = q_j$. Nếu không có ký hiệu a nào như thế thì ∅ (hoặc ε khi i = j) ký hiệu cho r^0_{ij}.*

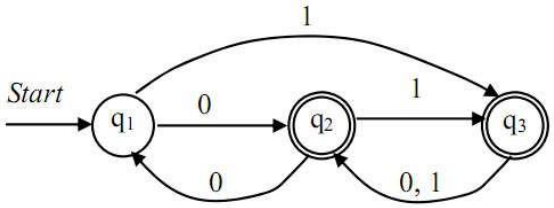
- *Công thức (1) cho R^k_{ij} chỉ liên quan đến các phép toán trên biểu thức chính quy: hợp, nối kết, và bao đóng. Hơn nữa theo giả thiết quy nạp, với mỗi l, k và m tồn tại biểu thức chính quy r^{k-1}_{lm} sao cho $L(r^{k-1}_{lm}) = R^{k-1}_{lm}$. Vậy đối với r^k_{ij} ta có thể chọn biểu thức chính quy:*

$$(r^{k-1}_{ik})(r^{k-1}_{kk})^*(r^{k-1}_{kj}) + r^{k-1}_{ij}$$

Cuối cùng ta có nhận xét rằng $L(M) = \bigcup_{q_j \in F} R^n_{1j}$ và R^n_{1j} ký hiệu cho tất cả các nhãn của tất cả các đường đi từ q_1 đến q_j.

Vậy L(M) được ký hiệu bởi biểu thức chính quy $r = r^n_{1j1} + r^n_{1j2} + \ldots + r^n_{1jp}$, trong đó tập $F = \{q_{j1}, q_{j2}, \ldots, q_{jp}\}$

<u>*Ví dụ 1.1*</u>: Viết biểu thức chính quy ký hiệu cho ngôn ngữ được chấp nhận bởi DFA sau:


Start
q1
q2
q3
1
0
1
0
0, 1

Gọi DFA được chỉ định trong hình 3.11 là M ({q_1, q_2, q_3}, {0,1}, d, q_1, {q_2, q_3}). Ta thấy, tập hợp tất cả các chuỗi được chấp nhận bởi DFA trên là các chuỗi làm cho ôtômát chuyển từ trạng thái bắt đầu q_1 đến một trong hai trạng thái kết thúc q_2, q_3 và không chuyển qua số tối đa là 3 (k = 3) trạng thái của ôtômát. Vậy ta cần viết biểu thức chính quy ký hiệu cho tập hợp này như sau:

$$r = r^3_{12} + r^3_{13}$$

Theo công thức đã được chứng minh trong định lý, ta có thể tính được các giá trị r^k_{ij} với i, j là chỉ số các trạng thái từ 1 đến 3 và với k = 0, 1 và 2 như bảng sau:

	k = 0	**k = 1**	**k = 2**
r^k_{11}	e	e	$(00)^*$
r^k_{12}	0	0	$0(00)^*$
r^k_{13}	1	1	0^*1
r^k_{21}	0	0	$0(00)^*$
r^k_{22}	e	e + 00	$(00)^*$
r^k_{23}	1	1 + 01	0^*1
r^k_{31}	Ø	Ø	$(0 + 1)(00)^*0$
r^k_{32}	0 + 1	0 + 1	$(0 + 1)(00)^*$
r^k_{33}	e	e	e + $(0 + 1)0^*1$

Bằng cách dùng các công thức tương đương như (r + s)t = rt + st và $(e + r)^* = r^*$ để đơn giản các biểu thức, chẳng hạn khi tính biểu thức:

$$r^1_{22} = r^0_{21}(r^0_{11})^* r^0_{12} + r^0_{22} = \mathbf{0(\varepsilon)^*0 + \varepsilon = 00 + \varepsilon}$$

Tương tự khi đơn giản biểu thức:

$$r^2_{13} = r^1_{12}(r^1_{22})^* r^1_{23} + r^1_{13} = \mathbf{0(00 + \varepsilon)^*(1 + 01) + 1}$$

ta nhận thấy $\mathbf{(00 + \varepsilon)^*}$ tương đương với $(00)^*$ và **(1 + 01)** thì tương đương với (e + 0)1 nên ta rút gọn:

$$r^2_{13} = \mathbf{0(00)^*(\varepsilon + 0)*1}$$

Mặt khác, chú ý rằng $\mathbf{(00)^*(\varepsilon + 0)}$ thì tương đương với $\mathbf{0^*}$, vì thế **0(00) (ε + 0)1+1** cũng bằng **00*1 + 1** hay cuối cùng là **0*1**.

Để hoàn thành việc xây dựng biểu thức chính quy cho M, ta cần tính r^3_{12} và r^3_{13}. Ta viết:

$$r^3_{12} = r^2_{13}(r^2_{33})^* r^2_{32} + r^2_{12}$$
$$= 0^*1(\varepsilon + (0 + 1)0^*1)^*(0 + 1)(00)^* + 0(00)^*$$
$$\mathbf{= 0^*1((0 + 1)0^*1)^*(0 + 1)(00)^* + 0(00)^*}$$

và
$$r^3_{13} = r^2_{13}(r^2_{33})^* r^2_{33} + r^2_{13}$$
$$= 0^*1(\varepsilon + (0 + 1)0^*1)^*(\varepsilon + (0 + 1))0^*1) + 0^*1$$
$$\mathbf{= 0^*1((0 + 1)0^*1)^*}$$

Vậy biểu thức chính quy có dạng:

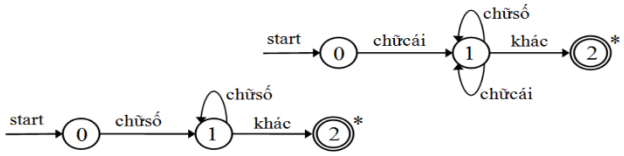
$$r = r^3_{12} + r^3_{13} = \mathbf{0^*1((0+1)0^*1)^*(\varepsilon + (0+1)(00)^*) + 0(00)^*}$$

Phần 2: Các Đồ Thị Chuyển Trạng Thái (transition diagram)

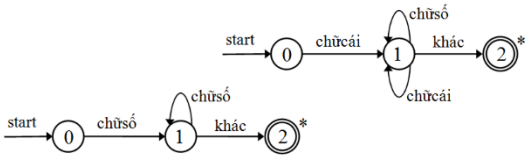
2.1 Định nghĩa đồ thị chuyển

Đồ thị chuyển là phương pháp thường sử dụng để mô tả một cách trực quan sơ đồ hoạt động của các automat hữu hạn

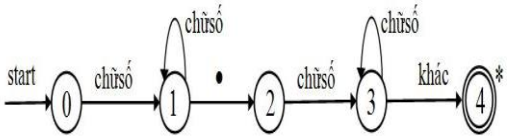
Đồ thị chuyển biểu diễn tên



Đồ thị biểu diễn số nguyên dương



Đồ thị biểu diễn số nguyên tố



2.2 Các kí hiệu của đồ thị chuyển

Trạng thái: vẽ bởi vòng tròn, kí hiệu ghi bên trong là "tên" (số hiệu) của trạng thái đó

Trạng thái kết thúc: vòng tròn kép

Trạng thái kết thúc có đánh dấu sao (*): ký tự cuối cùng không thuộc vào từ tố được đoán nhận

Bước chuyển: vẽ bởi mũi tên nối tới trạng thái sẽ chuyển đến, kí hiệu ghi bên cạnh là "nhãn" của bước chuyển

Nhãn ghi các kí tự hoặc loại kí tự cho phép thực hiện bước chuyển

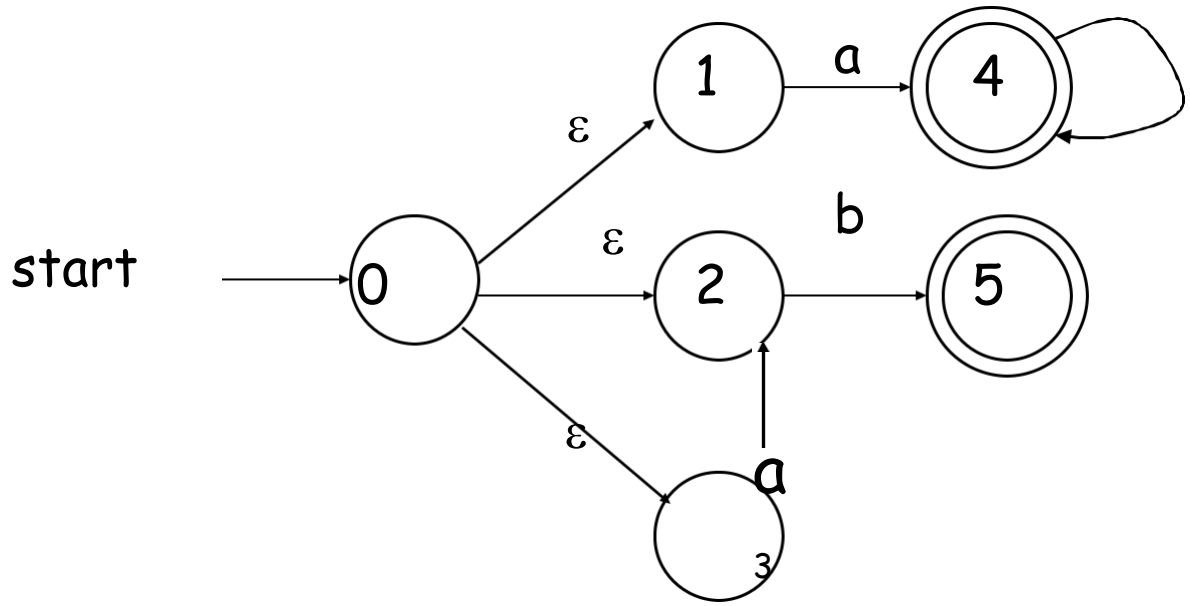
Nhãn "start": xác định trạng thái bắt đầu của automat.

2.3 Đồ thị chuyển của một NFA

Xét ngôn ngữ chính quy L = aa* | b | ab

Ta có thể xây dựng đồ thị chuyển nhận biết L có các đặc trưng của NFA:
Từ một trạng thái có thể có nhiều bước chuyển tương tự

Chứa kí hiệu ε ở nhãn



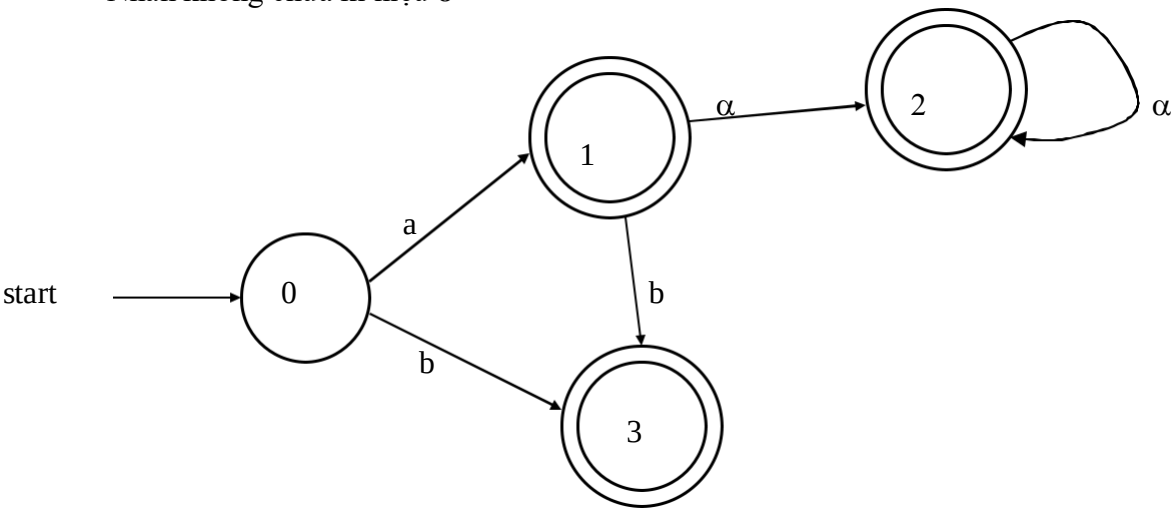
2.4 Đồ thị chuyển của một DFA

Cũng vẫn với ngôn ngữ L = aa* | b | ab

Ta có thể xây dựng đồ thị chuyển nhận biết L có các đặc trưng của DFA:

Từ một trạng thái không thể có các bước chuyển tương tự nhau (nhãn giống nhau)

Nhãn không chứa kí hiệu ε



2.5 Chuyển đổi từ NFA sang DFA

Chuyển đổi từ NFA sang DFA gồm hai bài toán:

Loại bỏ các bước chuyển chấp nhận kí hiệu đầu vào ε

Loại bỏ các trạng thái đa định

Input: một NFA (gọi là N)

Output: một DFA (gọi là D) đoán nhận cùng ngôn ngữ với N. Xây dựng D, gồm 2 bước:
Xây dựng tập trạng thái của D

Xây dựng các hàm chuyển move(s,a) đơn trị

Ý tưởng: quan sát hoạt động của N, một trạng thái của D là tập các trạng thái của N, một bước chuyển của D là một bước chuyển của tập trạng thái của N