1. Giới thiệu bài toán  
    1.1 Vấn đề bài toán…………………………………2  
    1.2 NFAε là gì………………………………………..2
2. Mục tiêu bài toán…………………………………………5
3. Phương pháp thực hiện…………………………………6  
    3.1 Xây dựng NFAε………………………………….6  
    3.2 Kiểm tra NFAε……………………………………7
4. Thiết kế cài đặt……………………………………………8
5. Kết quả đạt được………………………………………….10
6. Hướng phát triển………………………………………….10
7. **Giới thiệu bài toán**

Bài toán xây dựng NFAε và kiểm tra 1 chuỗi có thuộc NFAε đã cho hay không?

Giới thiệu NFAε:

NFAε là một loại máy trạng thái hữu hạn không đồng nhất (NFA) với khả năng chuyển trạng thái bằng cách sử dụng epsilon-điều kiện, tức là có thể chuyển sang một trạng thái mới mà không cần xem xét ký tự đầu vào.

Một NFAε bao gồm :

Tập trạng thái Q: Q = {q0, q1, q2, ..., qn}, trong đó q0 là trạng thái bắt đầu và q1, q2, ..., qn là các trạng thái khác.

Bộ ký tự đầu vào Σ: Σ = {a1, a2, ..., am}, trong đó m là số lượng ký tự khác nhau trong đầu vào.

Hàm chuyển trạng thái δ: δ: Q x (Σ ∪ {ε}) → P(Q), trong đó P(Q) là tập hợp các tập con của Q. Với δ(q, a) là tập hợp các trạng thái mà NFAε có thể đi tới từ trạng thái q với ký tự đầu vào a hoặc ε.

Trạng thái bắt đầu q0: q0 ∈ Q.

Tập trạng thái kết thúc F: F ⊆ Q.

Tuy nhiên, NFAε có thể có nhiều hơn một trạng thái bắt đầu và nhiều hơn một trạng thái kết thúc.

Hàm chuyển trạng thái của NFAε có thể có các trường hợp sau:

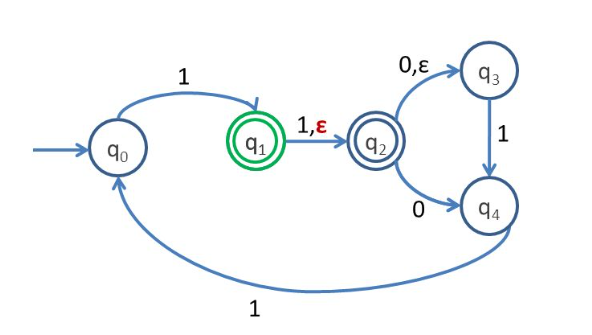
Chuyển từ một trạng thái đến một trạng thái khác dựa trên một ký tự đầu vào cụ thể.

Chuyển từ một trạng thái đến một trạng thái khác dựa trên epsilon-điều kiện mà không cần xem xét ký tự đầu vào.

Có thể kết hợp cả hai trường hợp trên để chuyển từ một trạng thái đến trạng thái mới.

Một NFAε có thể được chuyển đổi thành một NFA tương đương bằng cách loại bỏ tất cả các epsilon-điều kiện, nhưng NFAε có thể giúp rút ngắn và đơn giản hóa quá trình chuyển đổi này.

Đầu ra của 1 NFAε một ngôn ngữ, đó là tập hợp các xâu được chấp nhận bởi NFAε đó. Một xâu được chấp nhận bởi NFAε nếu tồn tại ít nhất một đường đi từ trạng thái ban đầu đến trạng thái kết thúc sao cho các chuyển trạng thái được thực hiện trên xâu đó có thể được thực hiện bằng cách sử dụng các chuyển trạng thái epsilon (ε), tức là các chuyển trạng thái không cần đọc bất kỳ ký tự nào của xâu.

  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
 *Hình 1*

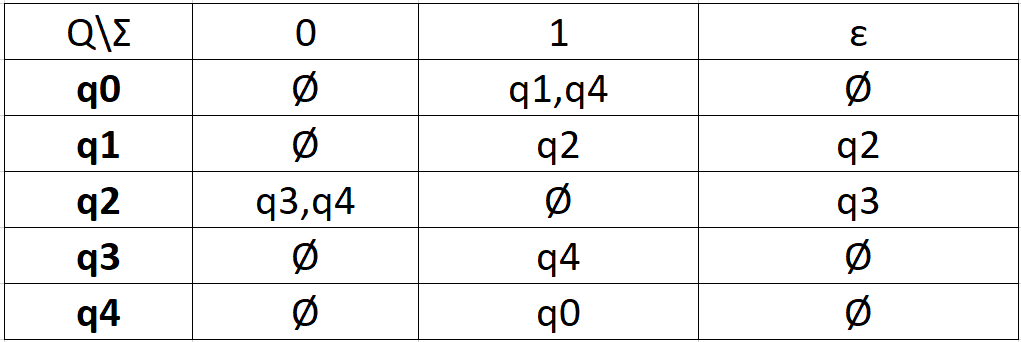
Tập trạng thái Q: {q0, q1,q2 ,q3 , q4}.

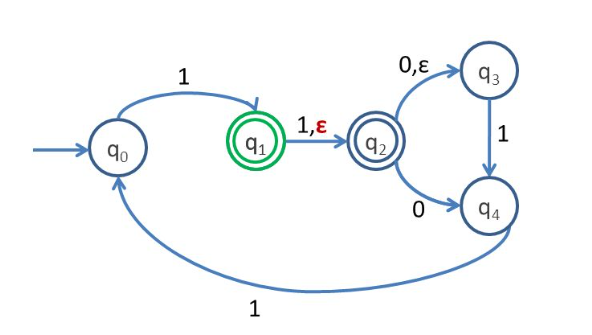
Bộ ký tự đầu vào Σ: {1,0, ε}

Trạng thái bắt đầu Q0: {q0}

Trạng thái kết thúc F:{q1,q2}

Hàm chuyển trạng thái δ:

Từ hình trên ở trạng thái bắt đầu khi q0 nhận giá trị 1 thì có thể dịch chuyển sang q1 và q4, tương tự khi q1 nhận giá trị 1 hoặc không nhận giá trị nào(kí hiệu là ε) thì sẽ sang q2. Tương tự như vậy cho các q khác  
Một chuỗi sẽ được chấp nhận bởi NFAε nếu tồn tại một đường đi từ trạng thái bắt đầu đến một trạng thái kết thúc sao cho khi đọc các ký tự trong chuỗi đó và di chuyển qua các trạng thái theo hàm chuyển trạng thái(δ) thì kết thúc ở trạng thái kết thúc. Các ký tự trong chuỗi được đọc từ trái sang phải và NFAε có thể sử dụng epsilon-điều kiện để chuyển đổi trạng thái.  
  
Ví dụ 1:



Từ trạng thái bắt đầu, chuỗi ‘11011’ có thuộc NFAε ở hình trên hay không?

q0 nhận giá trị 1 ở chuỗi ‘11011’ sẽ đi qua q1, chuỗi sẽ còn ’1011’.

sau đó từ q1 nhận giá trị 1 ở chuỗi ‘1011’ sẽ đi qua q2, chuỗi sẽ còn ‘011’.

q2 nhận giá trị 0 từ chuỗi ‘011’ sẽ đi qua q4 và chuỗi sẽ còn ‘11’.

q4 nhận giá trị 1 sẽ đi qua q0 chuỗi còn ‘1’.

q0 nhận giá trị cuối cùng của chuỗi là 1 sẽ đi đến q1.

Mà q1 là trạng thái kết thúc nên chuỗi này được chấp nhận

Ví dụ 2: chuỗi ‘10’ có thuôc NFAε ở hình trên hay không?

q0 nhận giá trị 1 sẽ đi qua q1 chuỗi sẽ còn ‘0’. Khác với ví dụ 1 từ q1 chúng ta có thể sang q2 mà không cần giá trị nào do có kí tự ε. Do đó q2 sẽ nhận giá trị là 0 và sang q3 hoặc q4. Ở q3 và q4 không phải là trạng thái kết thúc cho nên chuỗi này sẽ không được chấp nhận.

**2. Mục tiêu bài toán**

Viết 1 demo chương trình xây dựng lên NFAε dựa trên thông tin lí thuyết ở trên và sau đó nhập vào 1 chuỗi bất kì và kiểm tra chuỗi đó có được chấp nhận hay không chấp nhận.

Với bài toán này thì đầu vào sẽ là 1 chuỗi và đoạn chương trình code sẽ xử lí và cho kết quả trả về là chấp nhận hoặc không chấp nhận

**OUTPUT**

**NOT ARGEE**

**ARGEE**

**INPUT**

**NFAε**

**3. Phương pháp thực hiện**

**3.1Xây dựng NFAε.**

Để xây dựng NFAε, ta cần thực hiện các bước sau:

1. Xác định tập trạng thái của NFAε.
2. Bắt đầu với trạng thái ban đầu của NFAε, đánh dấu trạng thái đó là trạng thái đầu tiên của tập trạng thái.
3. Duyệt qua tất cả các trạng thái tiếp theo mà có thể đạt được từ trạng thái hiện tại bằng một epsilon transition (chuyển tiếp với epsilon), và thêm tất cả các trạng thái đó vào tập trạng thái.
4. Duyệt qua tất cả các trạng thái mà có thể đạt được từ trạng thái hiện tại bằng một chuyển tiếp với ký tự vào, và thêm tất cả các trạng thái đó vào tập trạng thái.
5. Lặp lại bước 2 và 3 cho đến khi không có thêm trạng thái mới được thêm vào tập trạng thái nữa.
6. Tập trạng thái mới được xác định sẽ chứa tất cả các trạng thái có thể đạt được từ trạng thái ban đầu của NFAε bằng một chuỗi chuyển tiếp với epsilon hoặc ký tự vào.
7. Tập trạng thái mới được đánh dấu là trạng thái cuối cùng nếu bất kỳ trạng thái nào trong tập trạng thái mới là trạng thái kết thúc của NFAε.
8. Xác định bảng chuyển trạng thái (*transition table)* cho NFAε, trong đó mỗi dòng tương ứng với một trạng thái và mỗi cột tương ứng với một ký tự vào (hoặc ε). Giá trị của mỗi ô trong bảng là một tập các trạng thái mà NFAε có thể chuyển đến từ trạng thái tương ứng khi nhận vào ký tự hoặc ε.
9. Xác định trạng thái ban đầu của NFAε.

Để xác định trạng thái ban đầu của một NFAε, ta cần xem xét đến các trạng thái mà NFAε có thể ở được từ trạng thái rỗng (epsilon state) mà không cần đọc vào bất kỳ ký tự nào. Các trạng thái này được gọi là trạng thái ban đầu của NFAε.

Cụ thể, ta có thể thực hiện các bước sau để xác định trạng thái ban đầu của một NFAε:

1. Tìm tất cả các trạng thái mà NFAε có thể chuyển đến từ trạng thái rỗng bằng cách không đọc vào bất kỳ ký tự nào (epsilon state).
2. Từ các trạng thái này, ta tìm ra các trạng thái mà NFAε có thể đến được bằng cách đọc vào một ký tự bất kỳ.
3. Nếu các trạng thái thu được ở bước 2 không có trạng thái nào mà chuyển đến chúng bằng epsilon state, thì chúng là các trạng thái ban đầu của NFAε.

Ví dụ: Cho trước một NFAε với các trạng thái là {q0, q1, q2, q3}, bảng chuyển trạng thái như sau:

|  |  |  |  |
| --- | --- | --- | --- |
|  | a | b | ε |
| q0 | q1 | q3 | q2 |
| q1 |  | q2 |  |
| q2 | q0 |  | q1 |
| q3 | q2 | q2 |  |

Để xác định trạng thái ban đầu của NFAε này, ta có thể thực hiện các bước sau:

1. Tìm các trạng thái mà NFAε có thể đến được từ trạng thái rỗng (epsilon state): q0 và q2.
2. Từ các trạng thái q0 và q2, ta tìm ra các trạng thái mà NFAε có thể đến được bằng cách đọc vào một ký tự bất kỳ: q1 và q3.
3. Các trạng thái thu được ở bước 2 đều có trạng thái ban đầu là trạng thái rỗng (epsilon state), nên chúng đều không phải là trạng thái ban đầu của NFAε. Do đó, trong trường hợp này, NFAε không có trạng thái ban đầu nào.
4. Xác định tập trạng thái kết thúc của NFAε.

Để xác định tập trạng thái kết thúc của NFAε, ta cần thực hiện các bước sau:

1. Xác định tập trạng thái kết thúc của các trạng thái đơn (non-epsilon) trong NFAε.
2. Với mỗi trạng thái chứa epsilon-transition, ta cần tìm tất cả các trạng thái đơn (non-epsilon) mà nó có thể đến được bằng cách đi qua epsilon-transition.
3. Gộp tất cả các trạng thái đơn (non-epsilon) thu được ở bước 1 và 2 để tạo thành tập trạng thái kết thúc của NFAε.

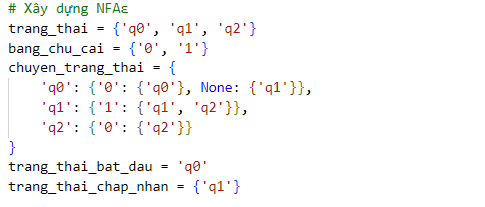
**3.2 Kiểm tra NFAε**

Sau khi xây dựng NFAε, để kiểm tra một chuỗi có thuộc NFAε đã cho hay không, ta thực hiện các bước sau:

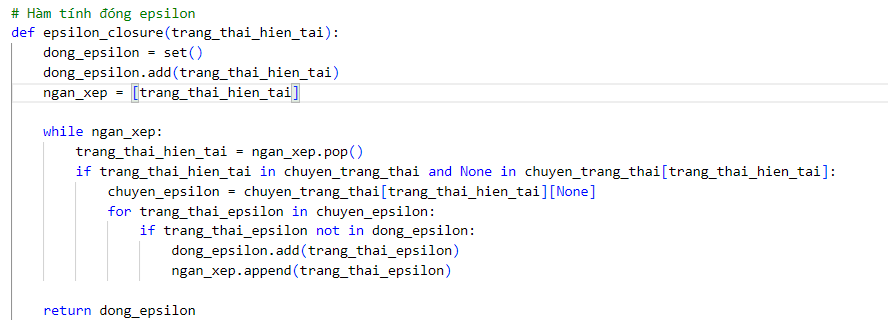
1. Khởi tạo một tập trạng thái hiện tại bằng tập trạng thái ban đầu của NFAε.
2. Duyệt qua từng ký tự trong chuỗi đầu vào và áp dụng bảng chuyển trạng thái để cập nhật tập trạng thái hiện tại.
3. Nếu tập trạng thái hiện tại rỗng, tức là NFAε không có đường đi nào tương ứng với chuỗi đầu vào, thì kết luận rằng chuỗi đầu vào không thuộc NFAε.
4. Sau khi duyệt qua hết chuỗi đầu vào, nếu tập trạng thái hiện tại có chứa ít nhất một trạng thái kết thúc của NFAε, thì kết luận rằng chuỗi đầu vào thuộc NFAε. Ngược lại, chuỗi đầu vào không thuộc NFAε.

**4. Thiết kế cài đặt:**

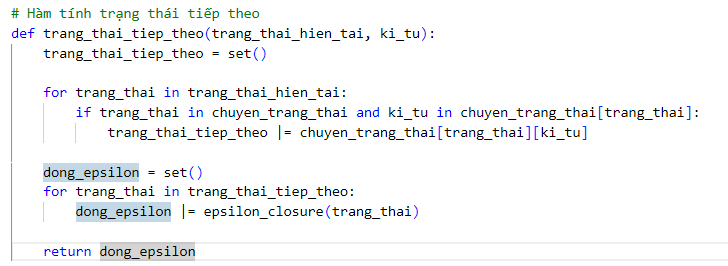
4.1 Xây dựng NFAε bằng cách định nghĩa tập trạng thái, bảng chữ cái, chuyển trạng thái và trạng thái bắt đầu, trạng thái chấp nhận.



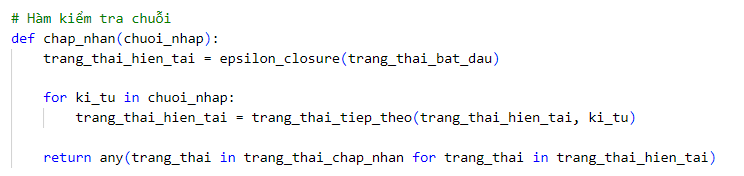
4.2 Tạo hàm epsilon\_closure(trạng thái hiện tại) để tính đóng epsilon của trạng thái hiện tại. Hàm này sử dụng thuật toán tìm kiếm theo chiều sâu để tìm các trạng thái liên thông với trạng thái hiện tại bằng epsilon chuyển đổi.



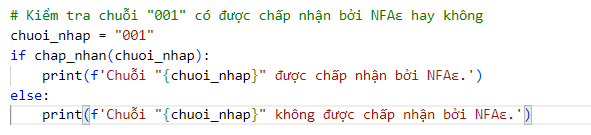
4.3Tạo hàm trang\_thai\_tiep\_theo(trạng thái hiện tại, kí tự) để tính tập các trạng thái tiếp theo khi đưa vào một kí tự. Hàm này sẽ lấy tất cả các trạng thái trong tập trạng thái hiện tại và tìm tất cả các trạng thái có thể được đạt đến bằng cách chuyển đổi bằng kí tự đầu vào.



4.4 Tạo hàm chap\_nhan(chuỗi nhập) để kiểm tra xem một chuỗi đầu vào có được chấp nhận bởi NFAε hay không. Hàm này sẽ tính đóng epsilon của trạng thái bắt đầu, sau đó áp dụng hàm trang\_thái\_tiếp\_theo() cho tất cả các kí tự trong chuỗi đầu vào, đồng thời tính đóng epsilon của mỗi trạng thái tiếp theo. Nếu một trong các trạng thái tiếp theo có thể đạt được trạng thái chấp nhận, hàm sẽ trả về True, ngược lại sẽ trả về False.



4.5 Cuối cùng, kiểm tra một chuỗi đầu vào bằng cách gọi hàm chap\_nhan() với chuỗi đó. Nếu chuỗi được chấp nhận bởi NFAε, hàm sẽ in ra thông báo "Chuỗi được chấp nhận bởi NFAε", ngược lại sẽ in ra thông báo "Chuỗi không được chấp nhận bởi NFAε".



**5. Kết quả đạt được**

Kết quả đạt được của bài toán viết demo xây dựng NFAε và kiểm tra một chuỗi có thuộc NFAε đã cho không là đưa ra kết luận xem chuỗi đó có thuộc ngôn ngữ được định nghĩa bởi NFAε hay không. Nếu chuỗi đó thuộc ngôn ngữ của NFAε, chương trình sẽ in ra thông báo cho biết chuỗi đó được chấp nhận bởi NFAε, ngược lại, nếu chuỗi không thuộc ngôn ngữ của NFAε, chương trình sẽ thông báo rằng chuỗi không được chấp nhận. Kết quả đạt được của bài toán là một công cụ hỗ trợ kiểm tra tính đúng đắn của NFAε và đưa ra quyết định về việc một chuỗi có thuộc ngôn ngữ được định nghĩa bởi NFAε hay không.

**6. Để phát triển và cải tiến thuật toán này, có thể thực hiện các bước sau:**

1. Tối ưu hóa thuật toán: Có thể tối ưu hóa thuật toán bằng cách sử dụng các kỹ thuật tối ưu hóa và giải thuật tìm kiếm hiệu quả hơn.
2. Mở rộng hỗ trợ cho các loại ngôn ngữ khác: Hiện tại, thuật toán này chỉ hỗ trợ cho các ngôn ngữ đơn giản chỉ gồm các ký tự '0' và '1'. Tuy nhiên, có thể mở rộng thuật toán để hỗ trợ cho các loại ngôn ngữ khác.
3. Áp dụng vào các ứng dụng thực tế: Thuật toán này có thể được áp dụng vào các ứng dụng thực tế như kiểm tra tính hợp lệ của địa chỉ email, mật khẩu, địa chỉ IP,...
4. Hỗ trợ đa luồng: Nếu cần xử lý đa nhiệm, có thể phát triển thuật toán để hỗ trợ đa luồng, tăng tốc độ xử lý dữ liệu.