**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**

**ĐẠI HỌC QUỐC GIA HÀ NỘI**

**BÁO CÁO VỀ BÀI TẬP XÂY DỰNG CÔNG CỤ TÁCH TỪ VÀ PHÂN TÍCH, CÚ PHÁP CHO FILE VC CƠ BẢN**

|  |  |
| --- | --- |
| Người thực hiện: | Nguyễn Mạnh Hưng  Nguyễn Xuân Nam |
| GV hướng dẫn: | Nguyễn Văn Vinh  Nguyễn Anh Quân |

**MỤC LỤC**

Kiến thức cần nhớ 4

Triển khai thuật toán 5

* Khởi tạo regex 5
* Thuật toán search regex cho tách từ 5
* Khởi tạo văn phạm 6
* Thuật toán check syntax cho code dùng văn phạm 6
* Những điều thuật check syntax chưa giải quyết được 6

**I. KIẾN THỨC CẦN NHỚ:**

**1. Automat hữu hạn đơn định**

1.1. Định nghĩa:

+ Là automat có dạng:

M = (Z, Q, , QO, F)

+ Trong đó:

Z: Tập kí tự vào

Q: Tập hữu hạn các trạng thái

QO: Trạng thái bắt đầu (QO ⊂ Q)

F: Tập trạng thái kết thúc (F ⊂ Q

: Hàm chuyển trạng thái

Có dạng: (q,a) = với q, p Q

a Z

1.2 Biểu diễn automat:

1.2.1. Dùng bảng chuyển trạng thái:

- Chỉ số hàng i: trạng thái vào qi Q

- Chỉ số cột j: kí tự nhận aj Z

- Chỉ số tại ô, i, j: trạng thái ra p thỏa mãn p = (qi,aj)

1.2.2. Dùng hình vẽ:

- Các trạng thái được đặt trong các vòng tròn, riêng trạng thái qO có dấu > biểu thị trạng thái bắt đầu

- Trạng thái kết thúc được đặt trong tấu tròn kép

- Các mũi tên chứa ký tự biểu thị sự chuyển trạng thái từ trạng thái này sang trạng thái kia theo hàm

1.3 . Hoạt động của automat:

- Đọc các xâu từ trái sang phải bắt đầu automat từ trạng thái qO

- Mỗi lần đọc hàm chuyển trạng thái dựa vào kí tự theo , có thể đọc xong hoặc không đọc xong xâu

- Nếu đọc vào trạng thái kết thúc thì vào trạng thái không kết thúc hay không đọc được thì xâu đoán nhận là xâu đúng cú pháp đọc vào trạng thái không kết thúc hay không đọc được xâu thì xâu đoán nhận không đúng cú pháp

**2. Phương pháp phân tích từ vựng:**

2.1. Phương pháp mô phỏng

- Mỗi trạng thái biểu thị 1 đoạn chương trình

- 2 trạng thái chuyển tiếp biểu thị hai chương trình nối tiếp nhau.

Lệnh rẽ nhánh:

Lệnh lặp:

2.2. Phương pháp dùng bảng

**II. TRIỂN KHAI THUẬT TOÁN (BÀI TẬP LỚN)**

**1. Xây dựng regex**

Chúng tôi xây dựng biểu thức chính quy cho tất cả các loại từ tố có trong code như kiểu dữ liệu, hàm class, struct, mảng, các từ khóa, toán tử, …

VD: Với từ tố biểu thị biến, như chúng ta đã biết biến được biểu diễn dưới dạng 1 xâu gồm số và chữ cái hoặc dấu ‘\_’ sao cho ký tự đầu tiên chỉ là   
chữ cái hoặc dấu ‘\_’ do đó, biểu thức chính quy cho chúng sẽ là:   
(\_|[a-zA-Z])(\_|[a-zA-Z0-9])+

- Ở đây chúng tôi sử dụng thư viện re của python là thư viện hỗ trợ xây dựng và quản lý các biểu thức chính quy. Với lệch re. Compile (r/ biểu thức) thì python sẽ hiểu và biên dịch các biểu thức chính quy. Re cũng có nhiều hàm giúp xử lý và tách các token dựa trên biểu thức chính quy đã tạo trong 1 văn bản cho sẵn, cái mà chúng tôi sẽ nói ở phần tiếp theo.

**2. Thuật toán tách từ số sử dụng regex đã tạo từ trước:**

- Cùng nói qua về tính chất của một ngôn ngữ lập trình, đơn biểu là C ++. Đây là ngôn ngữ không có tính nhập nhằng, Do vậy, tôi đã nghĩ ra ý tưởng đó là quyết đoạn code C++ từ trái sang phải sử dụng biểu thức chính quy.Điều này là hoàn toàn khả dĩ cho việc tách từ. Ý tưởng này chúng tôi đã thể hiện thông qua hàm scan.

- Cùng nói qua về các tham số trong số.

Thứ nhất là current, đây là tham số biểu thị index hiện tại của văn bản (vị trí ký tự mà tôi bắt đầu thực hiện việc quét. Điều này có nghĩa là đoạn xâu từ   
0 – current đã được quét ra từ đó.

Tiếp đến là dict, đây là một loại dictionary lưu các regex mà tôi đã tạo, mỗi lần quét tôi sẽ duyệt tất cả các phần tử trong dict để tìm ra regex cần tìm, chi tiết sẽ nói sau.

Thứ 3 là list\_token sẽ lưu các từ số đã tách được từ trái sang phải của văn bản.

- Quá trình chạy thuật toán:

Đầu tiên tôi sẽ khởi tạo current = 0, tức là bắt đầu quét từ bên trái văn bản.

Tôi sẽ duyệt văn bản từ vị trí current, sử dụng hàm re.search(). Hàm re.search() sẽ trả về xâu đầu tiên mà nó nhận được trong đoạn văn bản nó quét mà thỏa mãn một regex đưa ra. Trong đoạn code tôi đã dùng vòng lặp duyệt tất cả phần tử trong dict, mỗi lần duyệt tôi cho hàm re. search. Giả sử tôi duyệt đến phần tử int\_lit, tôi sẽ cho chạy dict (int\_lit). search nó sẽ tìm xâu đầu tiên mà khớp với regex của int\_lit (ở đây là biểu thức chính quy biểu diễn số). Do đó tại hàm if dict (category).search() != None nghĩa là tôi sẽ đưa vào một mảng list\_valid\_type (là mảng lưu giữ phần tử tạm thời các chỉ số đầu tiền trên của các xâu thỏa mãn một regex nào đó trong dict (xâu đấy nếu có là xâu đầu tiên mà regex quét được trong văn bản, hay mỗi regex sau khi quét thu được nhiều nhất 1 ứng viên). Hàm start() sẽ lấy chỉ số đầu tiên của các xâu đó.

Tôi cộng thêm tham số current vì tôi muốn lấy chỉ số cần tìm so với cả văn bản chứ không phải chỉ số so với trong xâu quét.

Như tôi đã nói, C++ là một ngôn ngữ không nhập nhằng vì vậy trong các ứng viên được đưa vào mảng list\_valid\_type tôi có thể tìm được xâu biên trái (hay xâu đầu tiên) của phân văn bản được quét để tách ra thành từ tố mới bằng cách chọn ra giá trị bé nhất trong mảng. Nó chính là vị trí bắt đầu của xâu biên trái cần tìm.

Khi đã có chỉ số đầu tiên, bằng 2 vòng lặp và biến stop (biến tìm vị trí cuối cùng của từ tố đầu tiên) thì tôi đã xác định được xâu biên trái (từ tố) và tách ra để push vào list\_token.

Xâu từ current đến vị trí đầu tiên của xâu biên trái (index\_must\_find) sẽ là một xâu không tìm được bất kỳ một biểu thức chính quy nào phù hợp. Nếu có   
(current < index – must-find) thì kết quả (list\_token) sẽ gần xâu này với định danh là “cannot be defined”.

**3. Khởi tạo văn phạm:**

Tôi khởi tạo tất cả các luật suy diễn có thể xảy ra trong code, với các token viết hoa là các trạng thái chưa kết thúc, token viết thường là các trạng thái kết thúc, với trạng thái bắt đầu là global.

**4. Thuật toán kiểm tra syntax (cú pháp) dựa trên văn phạm.**

Tôi sử dụng phương pháp vét cạn kinh điển check cú pháp theo thứ tự top-dourn bằng cách khởi tạo stack = Global và buffer là xâu tập hợp tất cả token tách được.

Đây là một thuật toán dạng quay lui với độ phức tạp dạng mũ. Do vậy nếu code C++ bị lỗi thì nó sẽ chạy khá lâu vì tất cả các khả năng đều được quét.

**5. Những điều mà văn phạm chưa giải quyết được**

a, Kiểm tra dữ liệu các biến:

Do tất cả các biến tôi đều gán là definition nên văn phạm của tôi chưa thể check được một biến đã được khai báo hay chưa lệnh gán có xung đột về kiểu dữ liệu biến hay không.

b, Vị trí khai báo hàm:

Văn phạm toàn bộ các vị trí có thể viết 1 lệnh mới tôi cho tự động chúng là trạng thái Global nên chưa thể kiểm tra được một hàm đã khai báo đúng phạm vi hay chưa.

**6. Đầu ra của code:**

Phần đầu sẽ in ra các dòng, mỗi dòng gồm 2 phần nhỏ, phần trước là từ vừa tách và vị trí bắt đầu của từ đó trong văn bản, phần sau sẽ in ra dịnh danh của từ đó (int\_list, string\_lit)

- Phần cuối sẽ in ra true/false, biểu thị đoạn văn bản (code C++ ) có viết đúng cú pháp hay không.