**I.11**

S → a A | b A

A → c A | b A | d

Phân tích chuỗi abbcbd theo thuật toán bottom-up

abbcbd <- abbcbA <- abbcA <- abbA <- abA <- aA <- S

**II.11**

11. Cho văn phạm G:

S→AA | AS | b

A→SA | AS | a

Chỉ ra quá trình thực hiện thuật toán CYK với w = abaab

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S |  |  |  |  |
| S,A | A,S |  |  |  |
| S,A | A,S | S,A |  |  |
| S,A | A | S | S,A |  |
| A | S | A | A | S |
| a | b | a | a | b |

**III.11**

Ngữ pháp LL(1) và SLR là gì? Cho văn phạm sau:

S->AaAb|BbBa

A->ε

B->ε

Đó là LL(1) hay SLR? Giải thích. Nếu là ngữ pháp nào, hãy xây dựng bảng phân tích tương ứng.

Ngôn ngữ LL(1) là loại ngôn ngữ được sử dụng trong việc phân tích cú pháp top-down trong lập trình. LL(1) có nghĩa là nó đọc các ký tự từ trái qua phải ,thực hiện các phép thay thế từ trên xuống dưới và chỉ cần xem trước một ký tự để quyết định phép thay thế tiếp theo. Điều này giúp LL(1) được xử lý nhanh chóng và được sử dụng trong nhiều trình biên dịch và bộ dịch.

Ngôn ngữ SLR là một trong những loại ngôn ngữ được sử dụng trong việc phân tích cú pháp bottom-up (từ dưới lên) trong lập trình. SLR được gọi là đơn giản vì nó chỉ sử dụng một tập hợp nhỏ các bảng trạng thái và các phép chuyển trạng thái .SLR có thể xử lý được nhiều loại ngôn ngữ và được sử dụng rộng rãi trong các trình biên dịch và bộ dịch.

Văn phạm trên thuộc SLR vì

Xây dựng văn phạm tăng cường

S’-> S

S->AaAb|BbBa

A->ε

B->ε

Có I = {S’ -> •S} thì closure(I) bao gồm:

I0:

S’ -> •S

S -> •AaAb

S -> •BbBa

A -> •ε

B -> •ε

Goto(I0,S)

I1 : S’ -> S•

Goto(I0,A)

I2: S -> A•aAb

Goto(I0,B)

I3: S -> B•bBa

Goto(I0,ε)

I4: A -> ε•

B -> ε•

Goto(I2,a)

I5: S-> Aa•Ab

A -> •ε

Goto(I3,b)

I6: S -> Bb•Ba

B -> •ε

Goto(I5,A)

I7: S -> AaA•b

Goto(I6,B)

I8: S -> BbB•a

Goto(I7,b)

I9: S -> AaAb•

Goto(I8,a)

I10: S-> BbBa•

Action Table Goto table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **State** | a | b | ε | $ | A | B | S |
| 0 |  |  | s4 |  |  |  |  |
| 1 |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |