**BÀI THỰC HÀNH 6**

**Môn: Chương trình dịch**

Nội dung: Phân tích cú pháp Bottom-up với các phương pháp

**I. PTCP Bottom-up:**

**Hướng dẫn:**-

* Văn phạm lớp LR có khả năng mô tả mạnh hơn văn phạm lớp LL, có thể mô tả văn phạm đệ quy trái (có trong hầu hết các ngôn ngữ lập trình)
  + Văn phạm đệ quy trái không thể dùng để phân tích từ trên xuống: lặp vô hạn
  + **S** ⇒ **S** + E ⇒ **S** + E + E ⇒ S + E + E + E
* Dễ dàng mô tả các ngôn ngữ lập trình thông thường
* Bộ phân tích cú pháp *gạt – thu gọn (Shift-Reduce)*
  + Xây dựng cây suy dẫn phải; Tự động xây dựng bộ phân tích cú pháp ; Phát hiện lỗi ngay khi xuất hiện; Cho phép phục hồi khi lỗi xảy ra
* Dựa trên Suy dẫn phải
  + Cây suy dẫn được xây dựng ngược lại: Bắt đầu từ kí hiệu kết thúc; Kết thúc tại kí hiệu bắt đầu
  + Ví dụ: G: S 🡪 S+E | E; E 🡪 số | (S). Xâu: (1+2+(3+4))+5

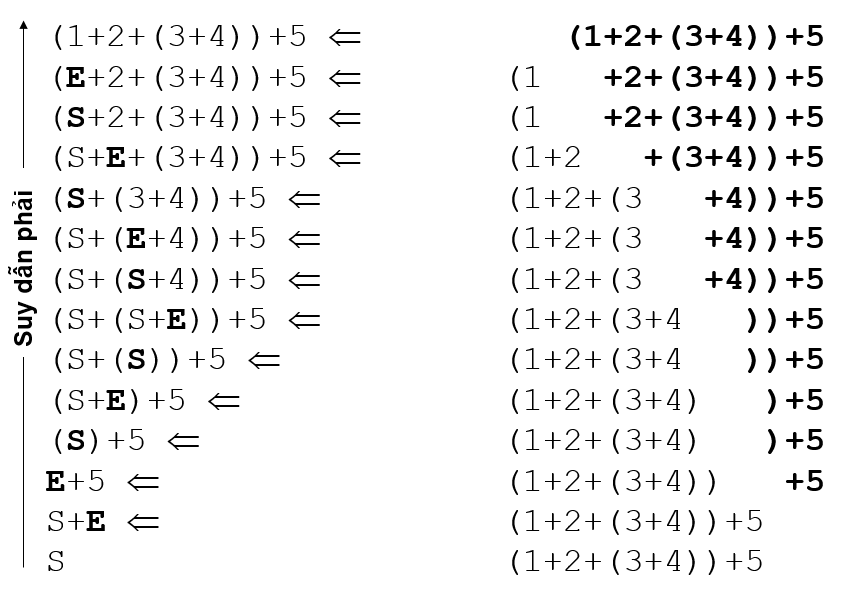
(1+2+(3+4))+5 ⇐ (**E**+2+(3+4))+5 ⇐

(**S**+2+(3+4))+5 ⇐ (S+**E**+(3+4))+5 ⇐

(**S**+(3+4))+5 ⇐ (S+(**E**+4))+5 ⇐ (S+(**S**+4))+5 ⇐

(S+(S+**E**))+5 ⇐ (S+(**S**))+5 ⇐ (S+**E**)+5 ⇐

(**S**)+5 ⇐ **E**+5 ⇐ **S**+5 ⇐ S+**E** ⇐ **S**



S

S

+

E

E

(

S

)

S

+

E

**S**

**+**

**E**

**E**

**1**

**2**

(

S

)

S

+

E

E

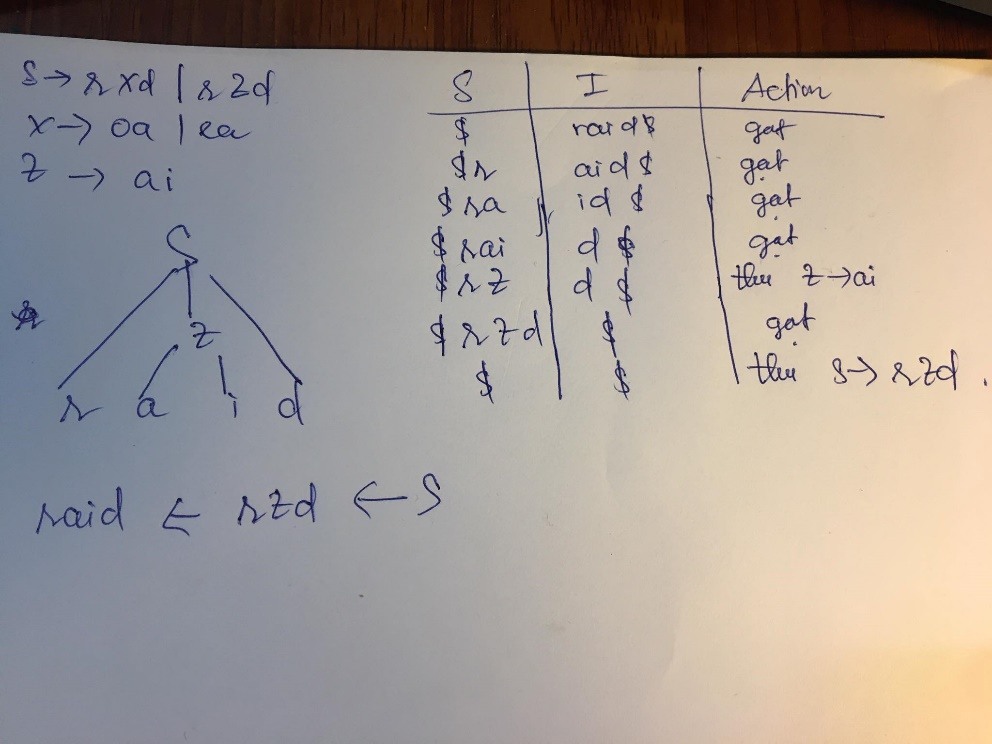
3

4

5

1. Chỉ ra quá trình thực hiện phân tích bottom-up của chuỗi **raid** thuộc văn phạm G có tập luật:  
S → r X d | r Z d  
X → o a | e a  
Z → a i

HD:



2. Chỉ ra quá trình thực hiện phân tích bottom-up của chuỗi **road** thuộc văn phạm G có tập luật:  
S → r X d | r Z d  
X → o a | e a  
Z → a i

3. Chỉ ra quá trình thực hiện phân tích bottom-up của chuỗi **read** thuộc văn phạm G có tập luật:  
S → r X d | r Z d  
X → o a | e a  
Z → a i

4. Chỉ ra quá trình thực hiện phân tích bottom-up của chuỗi **((x+y)=(y+x))** thuộc văn phạm G có tập luật:  
S → B  
B → R | ( B )  
R → E = E  
E → x | y | ( E + E )

5. Chỉ ra quá trình thực hiện phân tích bottom-up của chuỗi **((aopb)=(bopa))** thuộc văn phạm G có tập luật:  
S → A  
A → B | (A )  
B → E = E  
E → ( E **op** E )|a | b

6. Chỉ ra quá trình thực hiện phân tích bottom-up của chuỗi **((x+x)= (y+y))** thuộc văn phạm G có tập luật:  
S → T  
T → R | ( F )  
R→ F = F  
F → x | y | ( F + F )

7. Có thể áp dụng thuật toán phân tích bottom-up cho chuỗi **(a+a)\*a** thuộc văn phạm G dưới đây hay không? Chỉ ra quá trình thực hiện nếu có thể  
E → E + T | T  
T → T \* F | F  
F → ( E ) | a

8. Có thể áp dụng thuật toán phân tích bottom-up cho chuỗi **(b-b)\*b** thuộc văn phạm G dưới đây hay không? Chỉ ra quá trình thực hiện nếu có thể  
E → E + T | T| E-T  
T → T \* F |   
F → ( E ) | b

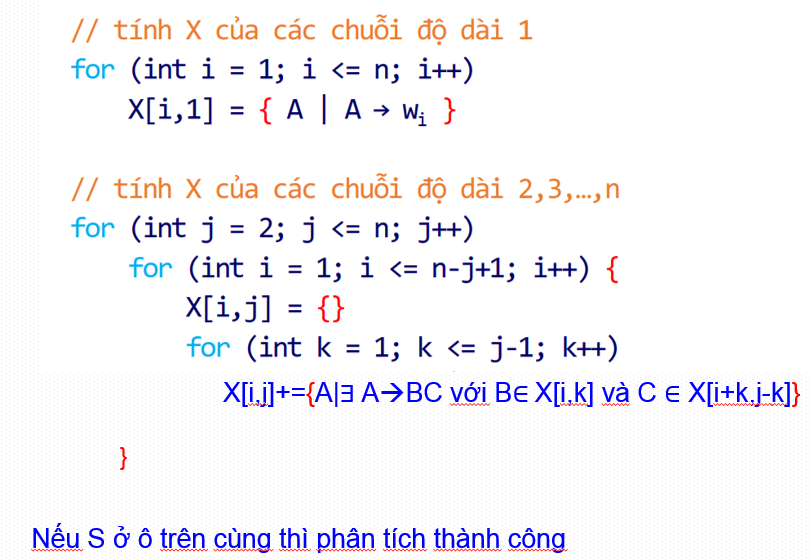
9. Có thể áp dụng thuật toán phân tích bottom-up cho chuỗi **(aopa)opa** thuộc văn phạm G dưới đây hay không? Chỉ ra quá trình thực hiện nếu có thể  
E → E op T | T| EopT  
T → T op F |   
F → ( E ) | a

10. Tương tự câu trên, chỉ ra quá trình phân tích bottom-up của chuỗi **true and not false** với tập luật:  
E → E and T | T  
T → T or F | F  
F → not F | ( E ) | true | false

11. Chỉ ra quá trình thực hiện phân tích bottom-up của chuỗi **abbcbd** thuộc văn phạm G có tập luật:  
S → a A | b A  
A → c A | b A | d

12. Chỉ ra quá trình thực hiện phân tích bottom-up của chuỗi **aaab** thuộc văn phạm G có tập luật:  
S → A B  
A → a A |   
B → b | b B

**II. Trình bày lại thuật toán CYK (Lưu ý đây là thuật toán Bottom-up nên phải suy luận từ dưới lên)**

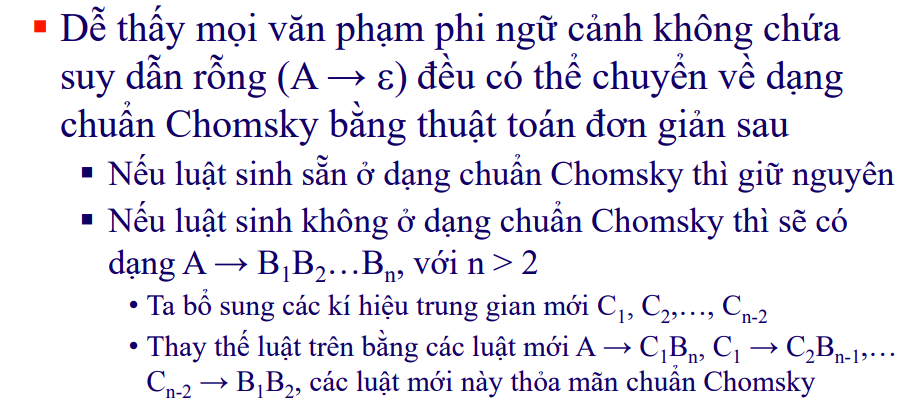


**Hướng dẫn:**

-B1: Kiểm tra VP có phải chuẩn CNF của Chomsky ko? Ko thì chuẩn hóa:

+ VP CNF: các luật sinh có dạng A🡪BC; A🡪 a

+ Chuẩn hóa về CNF:



1. Cho văn phạm G có tập luật:  
S → r X d | r Z d|d X r  
X → o a | e a|r|b  
Z → a i

Thuật toán CYK có phân tích thành công xâu w = board không? Chỉ ra?

**HD:**

**-** Chuyển VP về dạng chuẩn của Chomsky:

S🡪EF|EG

E🡪r|d

F🡪XE

G🡪Z H

H🡪d

X🡪AB|r|b

A🡪o|e

B🡪a

Z🡪CD

C🡪a

D🡪i

Bảng mô phỏng thuật toán CYK:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 |
| 5 | [1,5]=[1,1][2,4]+[1,2][3,3]+[1,3][4,2]+[1,4][5,1]=X,E,H |  |  |  |  |
| 4 | [1,4]=[1,1][2,3]+[1,2][3,1]=XF,B,C | [2][4]=[2,1][3,3]+[2,2][4,2]+[2,3][5,1]: A,X,FE,FH:# |  |  |  |
| 3 | [1,3]=[1,1][2,2]+[1,2][3,1]=XX,B,C:# | [2,3]=[2,1][3,2]+[2,2][4,1]=A,XE,XX:F | [3,3]=[3,1][4,2]+[3,2][5,1]=B,C,E,H:# |  |  |
| 2 | [1,2]=[1,1]+[2,1]=XA<--# | [2,2]=[2,1][3,1]=AB,AC <--**X** | [3,2]=[3,1]+[4,1]=BE|BX|CE|CX:# | [4,2]=[4,1][5,1]=EE|EH|XE|XH<--# | # |
| 1 | X | A | B,C | E,X | E,H |
| j | **b** | **o** | **a** | **r** | **d** |
| i |  |  |  |  |  |

Vì ô đầu tiên ko có trạng thái khởi tạo S nên xâu board không được sinh ra bởi VP trên.

2. Cho văn phạm G có tập luật:  
S → r X d | r Z d  
X → o a | e a|r  
Z → a i

Thuật toán CYK có phân tích thành công xâu w = roar không? Chỉ ra?

3. Cho văn phạm G có tập luật:  
S → r X d | r Z d  
X → o a | e a  
Z → a i

Thuật toán CYK có phân tích thành công xâu w = roar không? Chỉ ra?

4. Cho văn phạm G:  
S→AA | AS | b  
A→SA | AS | a  
Thuật toán CYK có phân tích thành công xâu w = aaabbb không? Chỉ ra?

5. Cho văn phạm G:  
S → AB | XB  
T → AB | XB  
X → AT  
A → a  
B → b  
Thuật toán CYK có phân tích thành công xâu w = ababab không? Chỉ ra?

6. Sử dụng thuật toán CYK để chỉ ra cây phân tích cho chuỗi **((3+1)= (1+3))** thuộc văn phạm G có tập luật:  
S → T  
T → R | ( F )  
R→ F = F  
F → số | ( F + F ) Với số là token của các số.

7. Sử dụng thuật toán CYK để chỉ ra cây phân tích cho chuỗi **(5+7)\*3** thuộc văn phạm G  
E → E + T | T  
T → T \* F | F  
F → ( E ) | số

8. Sử dụng thuật toán CYK để chỉ ra cây phân tích cho chuỗi **(10-2)\*2** thuộc văn phạm G  
E → E + T | T| E - T  
T → T \* F | F  
F → ( E ) | số

9. Sử dụng thuật toán CYK để chỉ ra cây phân tích cho chuỗi **(5+7)\*4** thuộc văn phạm G  
E → E op T | T  
T → T op F | F  
F → ( E ) | stn

Biết op là token toán tử với các từ vị {+,-,\*,/}, stn là token của các số tự nhiên.

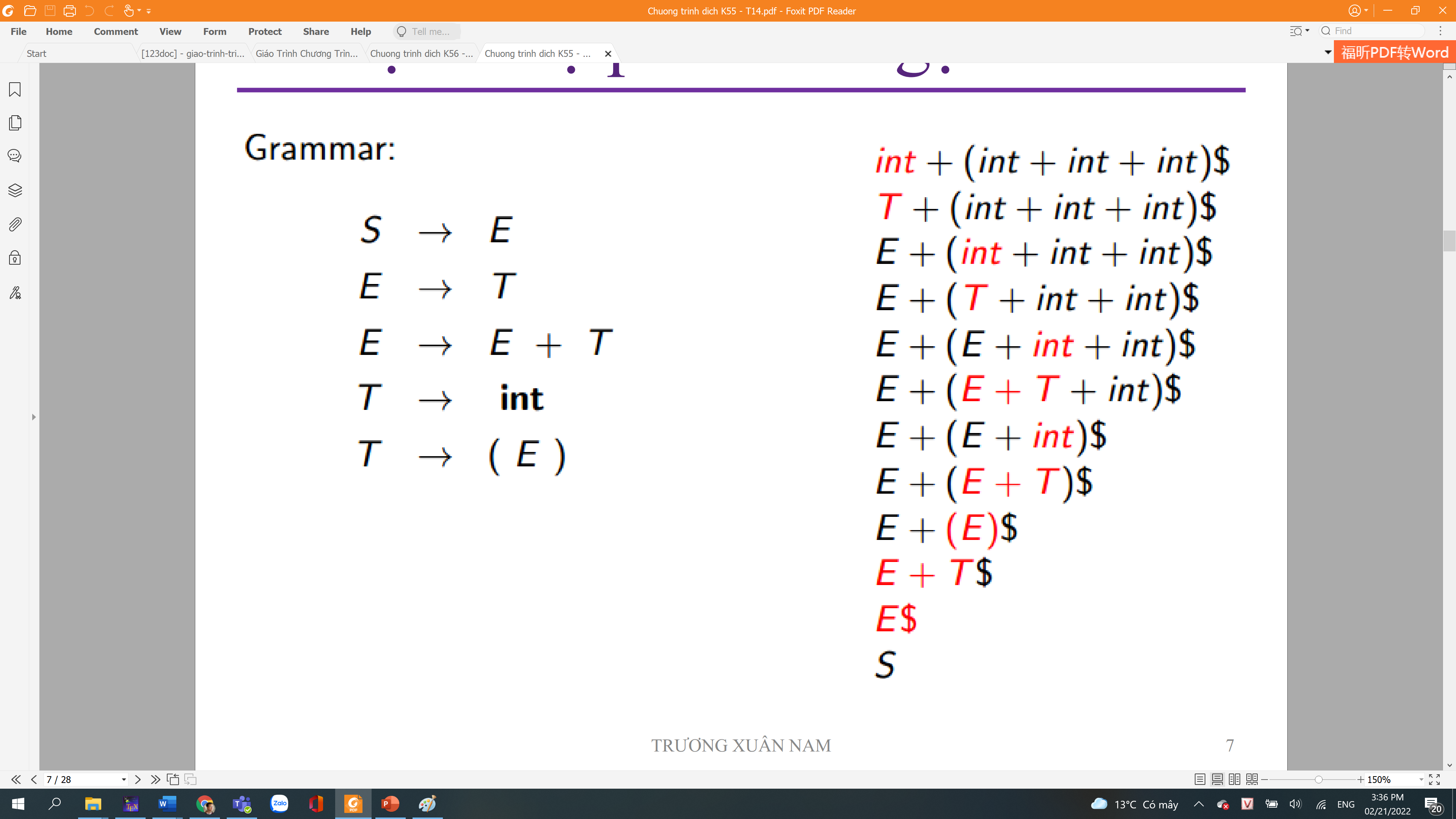
10. Chỉ ra cây phân tích của chuỗi **true and not false** sinh bởi thuật toán CYK với tập luật văn phạm G  
E → E and T | T  
T → T or F | F  
F → not F | ( E ) | true | false

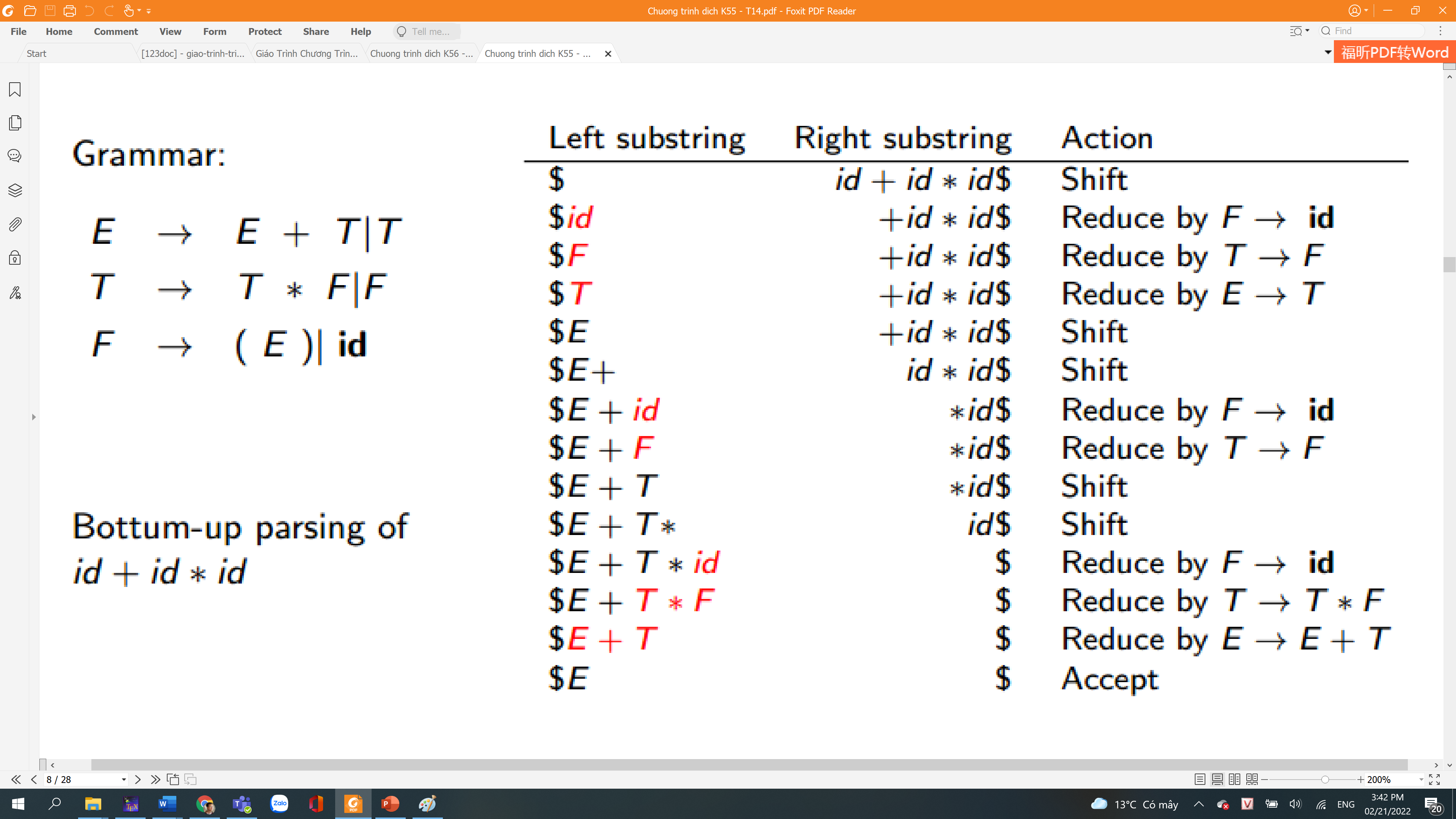
11. Cho văn phạm G:  
S→AA | AS | b  
A→SA | AS | a  
Chỉ ra quá trình thực hiện thuật toán CYK với w = abaab

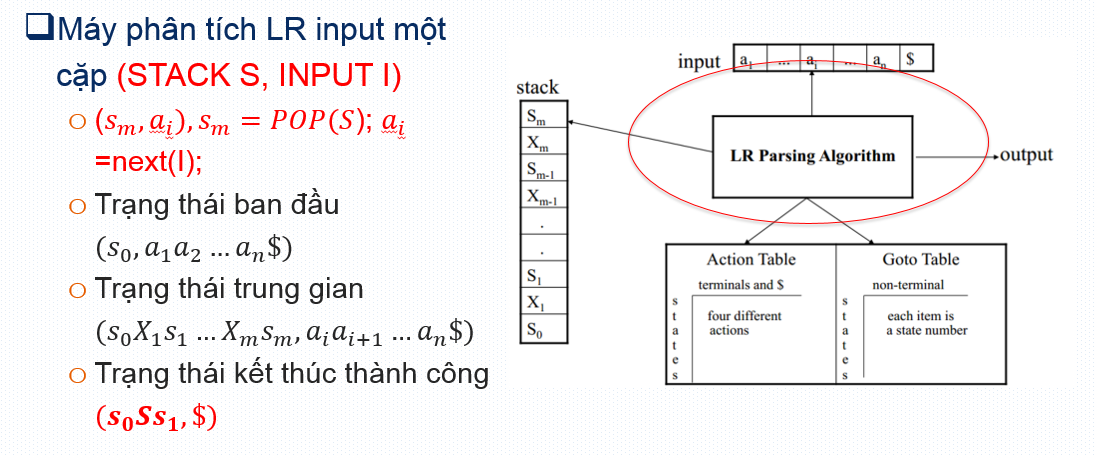
12. Cho văn phạm G:  
S → AB | XB  
T → AB | XB  
X → AT  
A → a  
B → b  
Chỉ ra quá trình thực hiện thuật toán CYK với w = aaabbb

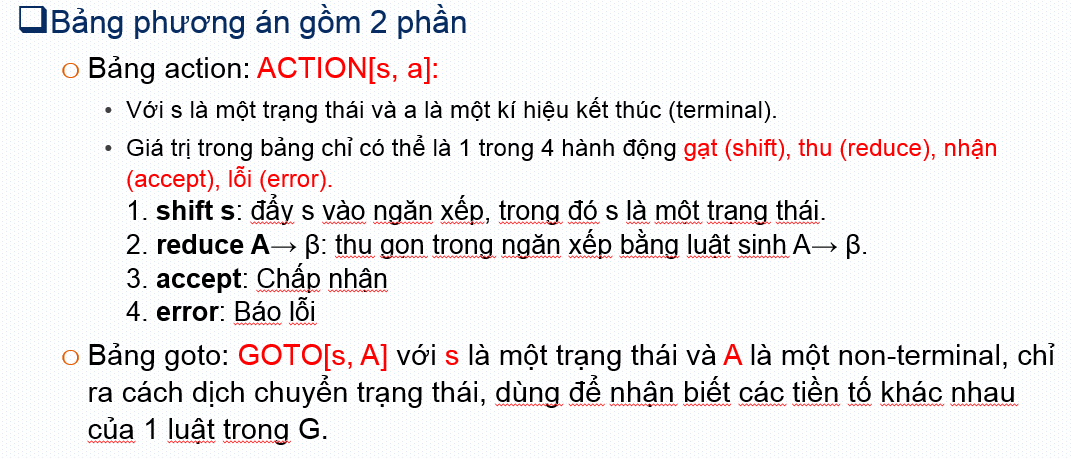
**II. LR**

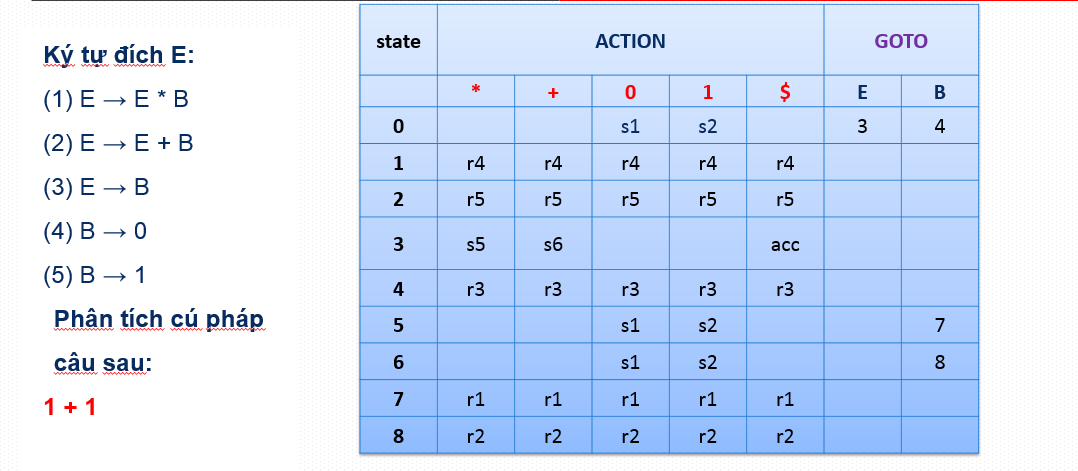
* **Chiến lược phân tích bottom-up (dưới lên) với bộ phân tích Gạt-Thu: Vấn đề: xây dựng bảng phương án như thế nào**
* **Khi nào thì gạt - shift**
* **Khi nào thì thu - reduce**
* **Còn hoạt động nào khác?**
* **Có trạng thái bị tranh chấp?**
* **Ví dụ:**

****

****

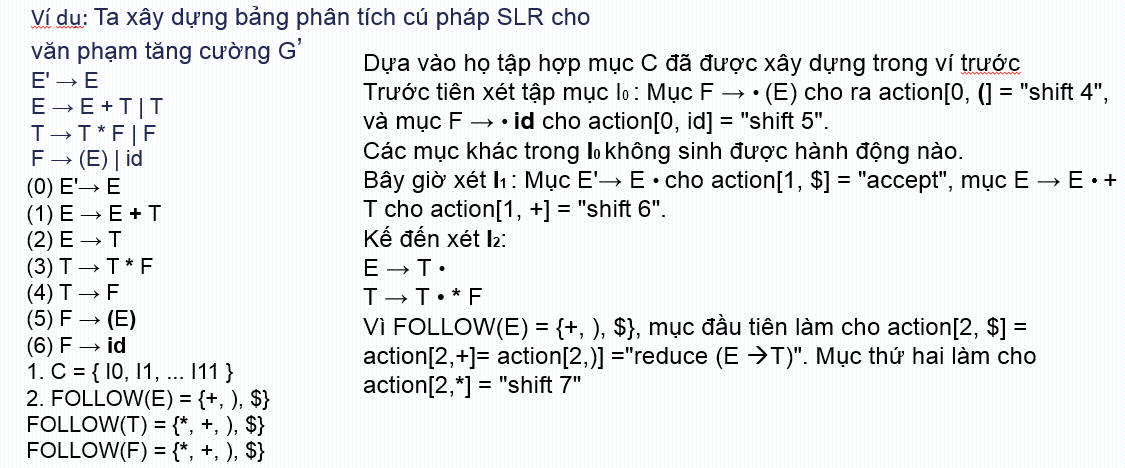




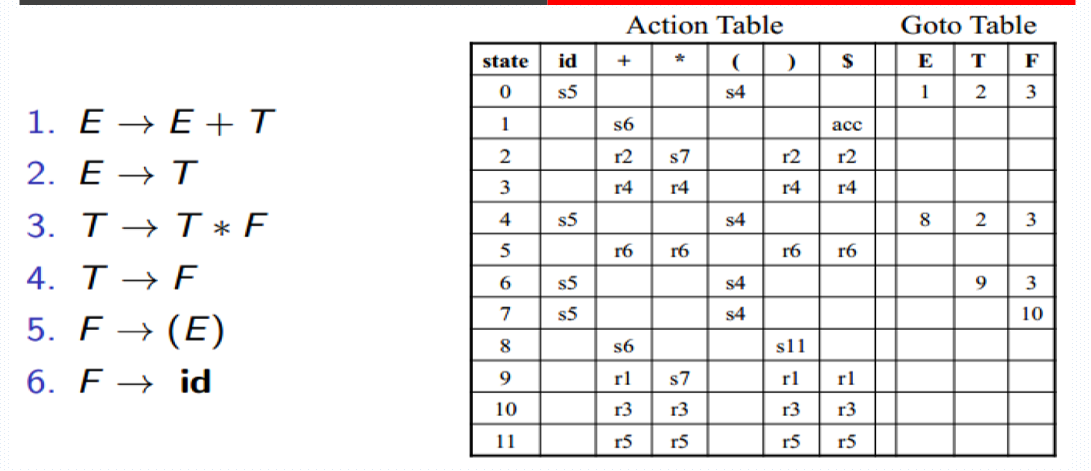


|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **State** | **Input stream** | **Output stream** | **Stack** | **Next action** |
| **0** | **1+1$** |  | **[0]** | **Shift 2** |
| **2** | **+1$** |  | **[0,2]** | **Reduce 5** |
| **4** | **+1$** | **5** | **[0,4]** | **Reduce 3** |
| **3** | **+1$** | **5,3** | **[0,3]** | **Shift 6** |
| **6** | **1$** | **5,3** | **[0,3,6]** | **Shift 2** |
| **2** | **$** | **5,3** | **[0,3,6,2]** | **Reduce 5** |
| **8** | **$** | **5,3,5** | **[0,3,6,8]** | **Reduce 2** |
| **3** | **$** | **5,3,5,2** | **[0,3]** | **Accept** |

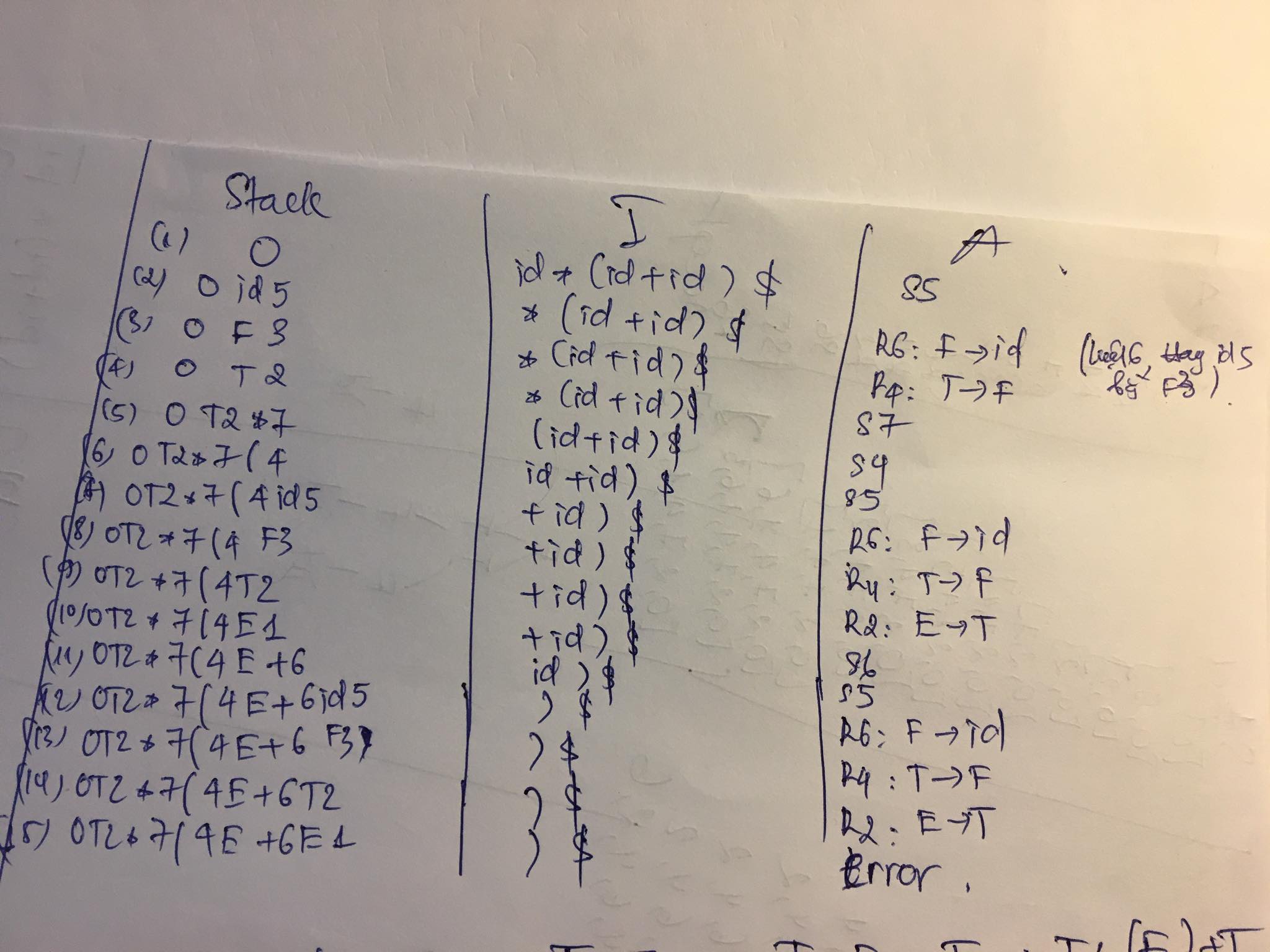
* Xây dựng bảng phân tích cú pháp Phương pháp SLR
  + **Mục (Item):** Cho một văn phạm G, mục LR(0) văn phạm là một luật sinh của G với một dấu chấm mục tại vị trí nào đó trong vế phải.
  + Ví dụ: Luật sinh A → XYZ có 4 mục như sau :  
    A → •XYZ  
    A → X• YZ  
    A → XY• Z  
    A → XYZ•  
    Luật sinh A → ε chỉ tạo ra một mục A → •
* **Phép toán bao đóng (Closure)**. Giả sử I là một tập các mục của văn phạm G thì bao đóng closure(I) là tập các mục được xây dựng từ I theo qui tắc sau:
  + Tất cả các mục của I được thêm cho closure(I).
  + Nếu và là một luật sinh thì thêm vào nếu nó chưa có trong đó.
  + Lặp lại bước này cho đến khi không thể thêm vào closure(I) được nữa.
  + Ví dụ Xét văn phạm tăng cường của biểu thức:  
    E' → E  
    E → E + T | T  
    T → T \* F | F  
    F → (E) | id
  + Nếu I là tập hợp chỉ gồm văn phạm {E'→ • E } thì closure(I) bao gồm:  
    E' → • E (Luật 1)  
    E → • E + T (Luật 2)  
    E → • T (Luật 2)  
    T → • T \* F (Luật 2)  
    T → • F (Luật 2)  
    F → • (E) (Luật 2)  
    F → • id (Luật 2)
* Cách tính goto(I, X):
  + Tạo một tập I' = ∅.
  + Nếu A → α•Xβ ∈ I thì đưa A→ αX•β vào I', tiếp tục quá trình này cho đến khi xét hết tập I.
  + Goto(I, X) = closure(I')
  + **Ví dụ:** Giả sử I = { E' → E•, E → E • + T }. Tính goto (I, +) ?
  + Ta có I' = { E→ E + • T }
  + ( goto (I, +) = closure(I') bao gồm các mục :
  + E → E + • T (Luật 1)
  + T → • T \* F (Luật 2)
  + T → • F (Luật 2)
  + F → • (E) (Luật 2)
  + F → • id (Luật 2)
* Thuật toán xây dựng bảng phân tích SLR   
  **Input**: Một văn phạm tăng cường G'  
  **Output**: Bảng phân tích SLR với hàm action và goto  
  Phương pháp:  
  1. Xây dựng C = { I0, I1, ..., In }, họ tập hợp các mục LR(0) của G'.  
  2. Trạng thái i được xây dựng từ Ii. Các action tương ứng trạng thái i được xác định như sau:  
   2.1. Nếu A → α • aβ ∈ Ii và goto (Ii, a) = Ij thì action[i, a] = "shift j". Ở đây a là ký hiệu kết thúc.  
   2.2. Nếu A → α• ∈ Ii thì action[i, a] = "reduce (A → α)", ∀a ∈ FOLLOW(A). Ở đây A không phải là S’  
   2.3. Nếu S' → S• ∈ Ii thì action[i, $] = "accept".  
  Nếu một action đụng độ được sinh ra bởi các luật trên, ta nói văn phạm không phải là SLR(1). Giải thuật sinh ra bộ phân tích cú pháp sẽ thất bại trong trường hợp này.  
  3. Với mọi ký hiệu chưa kết thúc A, nếu goto (Ii,A) = Ij thì goto [i, A] = j  
  4. Tất cả các ô không xác định được bởi 2 và 3 đều là “error”  
  5. Trạng thái khởi đầu của bộ phân tích cú pháp được xây dựng từ tập các mục chứa S’→ • S



Bài tập: Cho Văn phạm G và bảng phân tích LR sau:



1. Hãy mô tả các bước hoạt động của Stack dựa vào bảng và văn phạm trên, phân tích xâu: **id \* (id + id)**

****

2. Hãy mô tả các bước hoạt động của Stack dựa vào bảng và văn phạm trên, phân tích xâu: **id \* id + id\*id**

3. Hãy mô tả các bước hoạt động của Stack dựa vào bảng và văn phạm trên, phân tích xâu: **id + id \* id+id**

4. Hãy mô tả các bước hoạt động của Stack dựa vào bảng và văn phạm trên, phân tích xâu: (**id + id) \* (id+id)**

5. Văn phạm G: S🡪EF|id; E🡪SE|id. Hãy:

Xây dựng các item LR(0) cho Văn phạm

Xây dựng bảng PTCP bằng SLR

6. Cho văn phạm:

E🡪E+T|T

T🡪TF|F

F🡪F\*|a|b

Xây dựng bộ các item LR(0) cho văn phạm này

Xây dựng bảng PTCP bằng SLR

7. Cho văn phạm tăng cường:

S' -> S

S -> 0 A

A -> 0 A 1

A -> 1

Xây dựng bộ các item LR(0) cho văn phạm này

Xây dựng bảng PTCP bằng SLR

8. Cho văn phạm tăng cường:

0) S' -> S

1) S -> a B

2) B -> a B A B

3) B -> ε

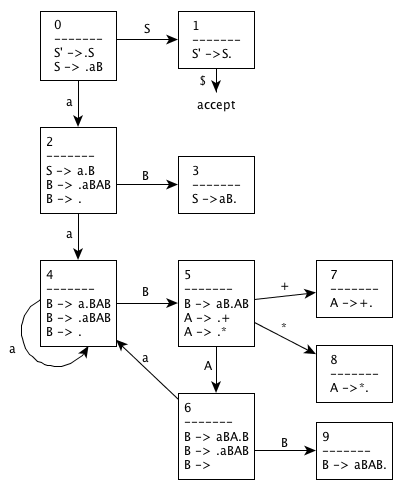
4) A -> +

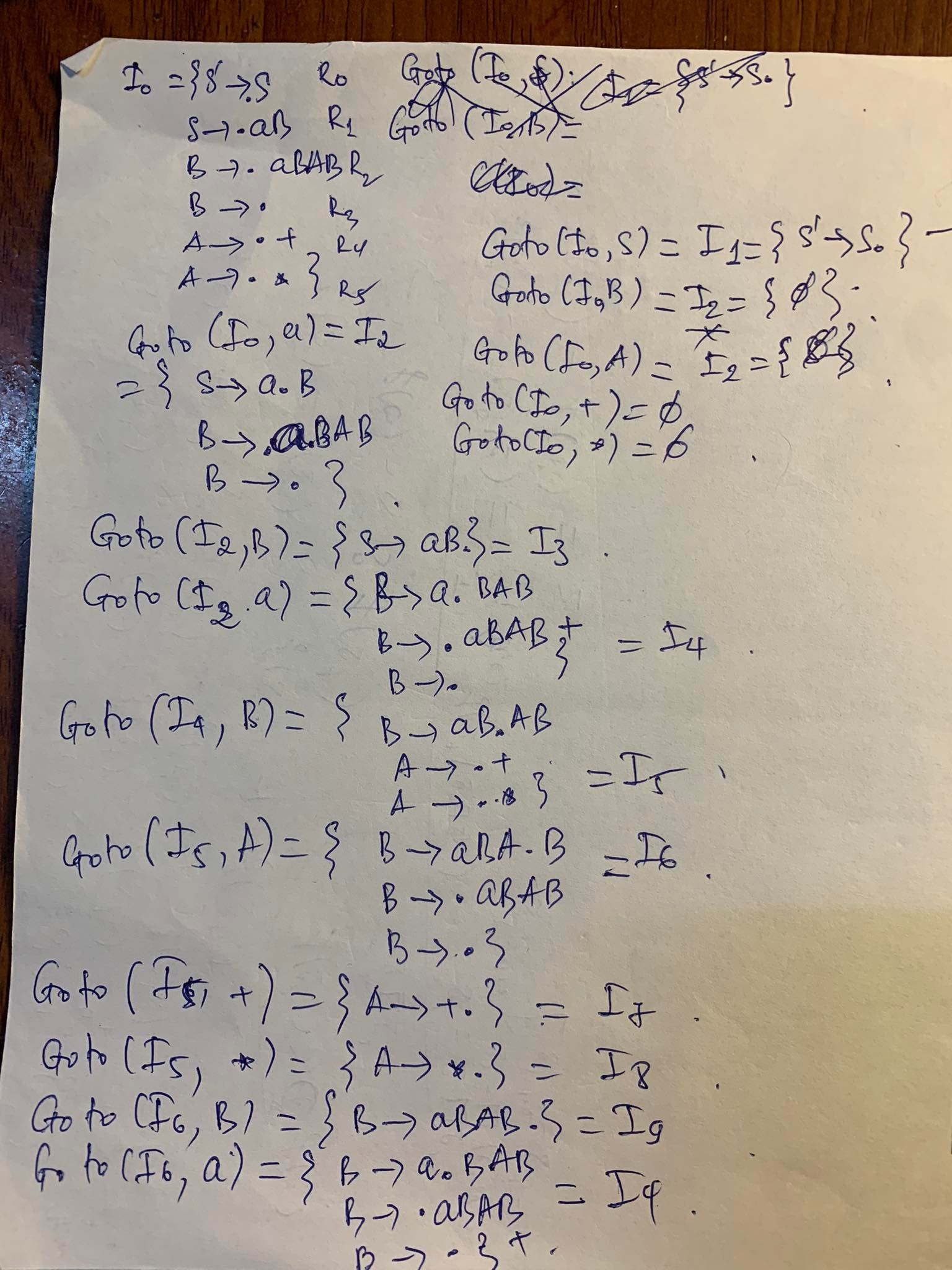
5) A -> \*

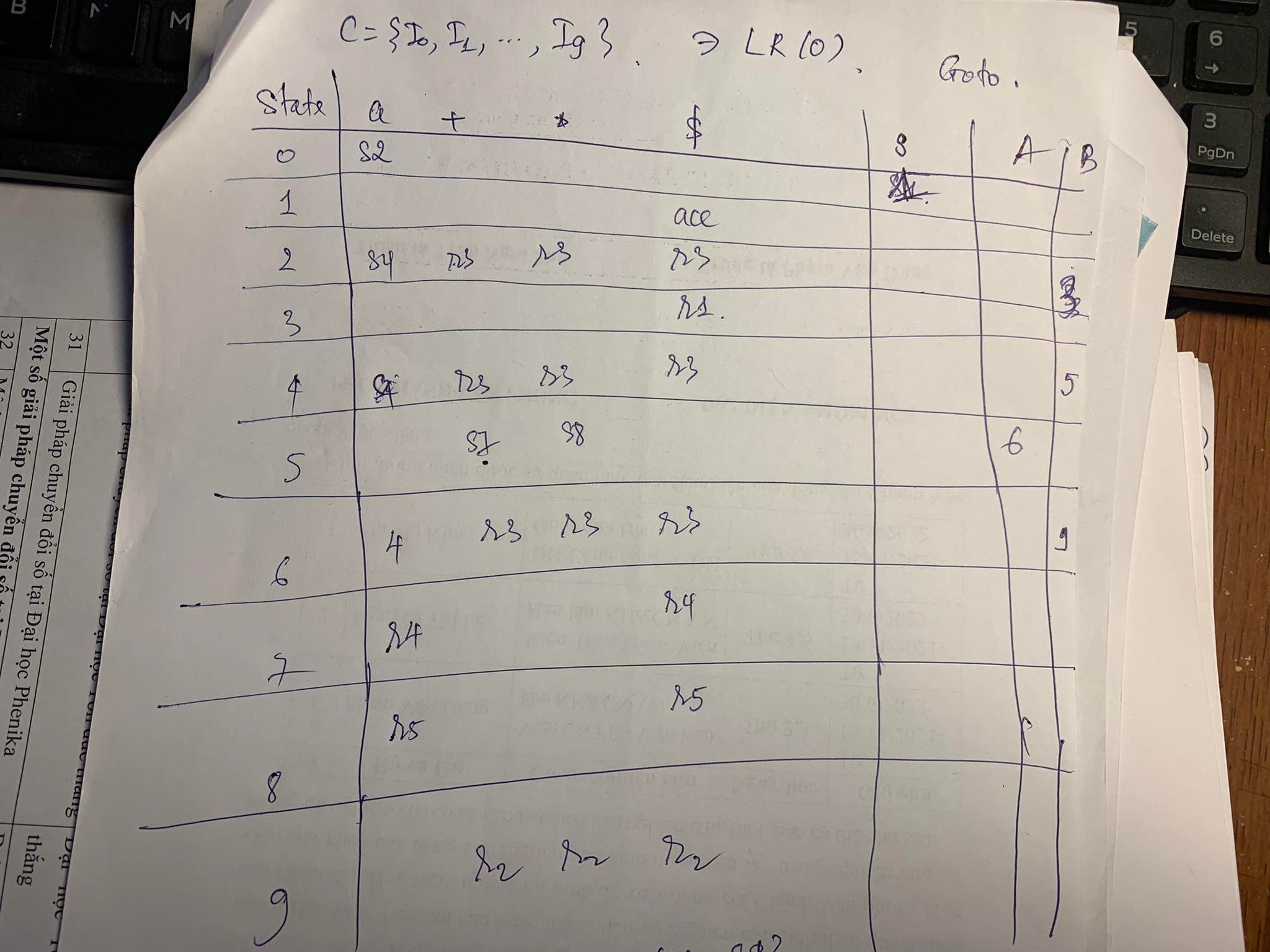
Xây dựng bộ các item LR(0) cho văn phạm này

Xây dựng bảng PTCP bằng SLR

**Tham khảo:**

****

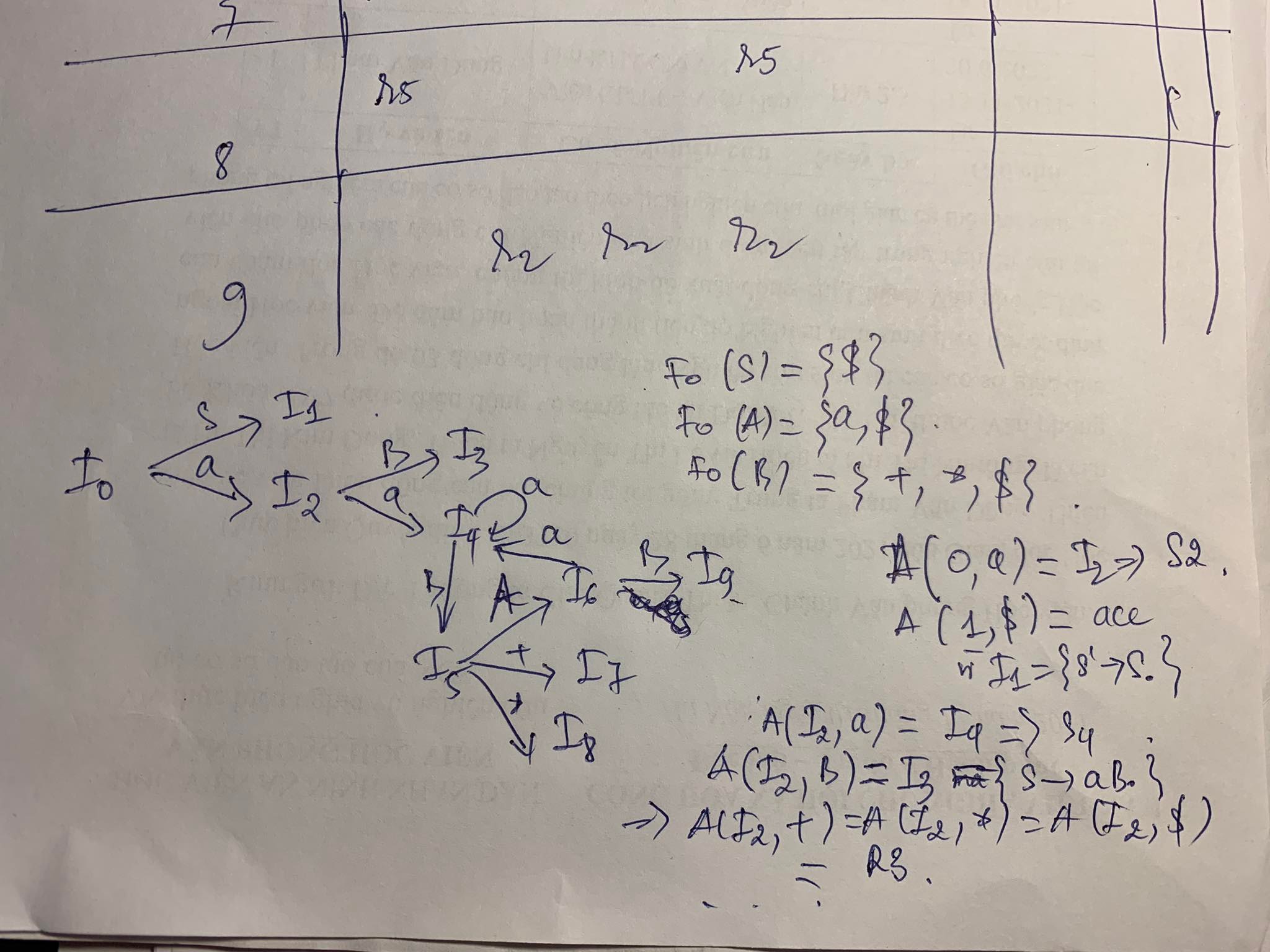
****



Giải thích:

- Đánh dấu các luật trong I0: ro,r1,…,r5

- Theo Goto vẽ sơ đồ chuyển I0🡪I1🡪…🡪I9



Tính FOLLOW(S), FOLLOW(A), FOLLOW(B). Áp dụng quy tắc tính trong bảng Action như sau:

A(0,a): Vì Goto(I0,a)=I2 nên ô này ghi s2 (Shift 2)

Tương tự A(1,$)=acc vì I1={S’🡪S.}; A(2,a)=s4;

A(2,B)=r3 vì Goto(I2,B)={S🡪aB.} nên thêm vào mọi cột có nhãn nằm trong Fo(B) luật thu r3 (B->.) ; Các trường hợp khác suy luận tương tự. 9. Cho văn phạm tăng cường:

0) S' -> S

1) S -> A

2) A -> (S) S A

3) A -> ε

Xây dựng bộ các item LR(0) cho văn phạm này

Xây dựng bảng PTCP bằng SLR

10. Cho văn phạm G: S->SS+|SS\*|a. Cho bảng trạng thái sau:

|  |
| --- |
|  |
| **State** | **ACTION** | | | | **GOTO** | | |
|  | **+** | **\*** | **$** | **S** | **A** | **B** |
| 0 | s2 |  |  |  | s1 |  |  |
| 1 |  |  |  | acc |  |  |  |
| 2 | s4 | r3 | r3 | r3 |  |  | s3 |
| 3 |  |  |  | r1 |  |  |  |
| 4 | s4 | r3 | r3 | r3 |  |  | s5 |
| 5 |  | s7 | s8 |  |  | s6 |  |
| 6 | s4 | r3 | r3 | r3 |  |  | s9 |
| 7 | r4 |  |  | r4 |  |  |  |
| 8 | r5 |  |  | r5 |  |  |  |
| 9 |  | r2 | r2 | r2 |  |  |  |

- Hãy xây dựng văn phạm tăng cường G’ cho G.

- Đưa ra các Action khi xử lý đầu vào aa\*a+

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.

12. Cho văn phạm sau:

S->SA|A

A->a

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