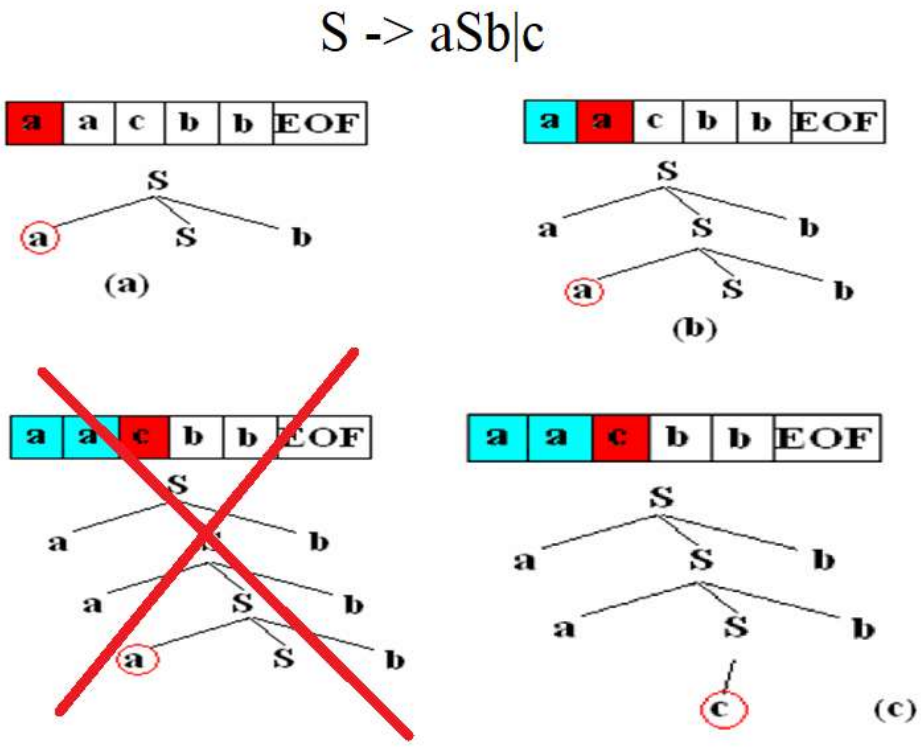
Bài 7
Phân tích cú pháp tiền định

ONE LOVE. ONE FUTURE.

Quay lại giải thuật phân tích cú pháp trên xuống quay lui

- Liệu có thể chọn vế phải của sản xuất với vế trái là nút hoạt động?
- Điều đó phụ thuộc dạng của văn phạm



Phân tích tiền định

- Tư tưởng chính của giải thuật phân tích cú pháp trên xuống **quay lui**
  - Bắt đầu từ gốc, phát triển xuống các nút cấp dưới
  - Chọn một sản xuất và thử xem có phù hợp với xâu vào không
  - Quay lui nếu lựa chọn dẫn đến ký hiệu được sinh bởi văn phạm không phù hợp ký hiệu đang xét
- Có thể tránh được **quay lui**?
  - Cho sản xuất $A \rightarrow \alpha \mid \beta$ bộ phân tích cú pháp cần chọn giữa α và β
- Làm thế nào?
  - Cho ký hiệu không kết thúc A và ký hiệu xem trước t, sản xuất nào của A chắc chắn sinh ra một xâu bắt đầu bởi t?

Bộ phân tích tiền định

- Bộ phân tích cú pháp có thể "đoán trước" sản xuất nào sẽ được dùng khi nút hoạt động là ký hiệu không kết thúc
  - Bằng cách xem trước 1 hoặc nhiều hơn các token
  - Không quay lui
- Bộ phân tích cú pháp tiền định hoạt động trên các văn phạm LL(k)
  - L bên trái là "left-to-right" : hướng đọc file nguồn
  - L bên phải là "leftmost derivation": phân tích trên xuống, trả ra phân tích trái
  - k là số token phải xem trước để "đoán" vế phải được chọn
- Phương pháp phân tích LL(1) được dùng trong thực tế

Bộ phân tích cú pháp tiền định cho văn phạm LL(1)

- Sử dụng *bảng phân tích M* và *stack* (nguyên mẫu của stack D2 trong bộ phân tích trên xuống quay lui).
- Để thực hiện giải thuật này, văn phạm phải thỏa mãn điều kiện
  - *Không có 2 vế phải của cùng một vế trái sinh ra các xâu có tiền tố giống nhau. Ta chỉ xét văn phạm LL(1)*
- Đặc biệt cũng giống như các phương pháp phân tích trên xuống khác, văn phạm cần không đệ quy trái để tránh lặp vô hạn

Đệ quy trái

Văn phạm G là *đệ quy trái* nếu nó chứa một ký hiệu không kết thúc A với suy dẫn:

$$A \Rightarrow^* A\alpha \qquad \text{với } \alpha \text{ là xâu bất kỳ}$$

Đệ quy trái có thể xuất hiện trong một bước suy dẫn (*đệ quy trái trực tiếp*) hoặc nhiều bước suy dẫn (*đệ quy trái gián tiếp*)

Tuy nhiên, ta đã chứng minh được là có thể khử đệ quy trái trong văn phạm bằng cách thay thế các sản xuất đệ quy trái bằng các sản xuất tương đương không đệ quy trái.

Đệ quy trái trực tiếp

$A \rightarrow A\ \alpha \mid \beta$ β không đệ quy trái

$\Downarrow$ Loại các sản xuất đệ quy trái trực tiếp

$A \rightarrow \beta\ A'$

$A' \rightarrow \alpha\ A' \mid \varepsilon$ Nhận được văn phạm tương đương

Tổng quát,

$A \rightarrow A\ \alpha_1 \mid ... \mid A\ \alpha_m \mid \beta_1 \mid ... \mid \beta_n$ $\beta_1 ... \beta_n$ không đệ quy trái

$\Downarrow$ Loại bỏ đệ quy trái trực tiếp

$A \rightarrow \beta_1\ A' \mid ... \mid \beta_n\ A'$

$A' \rightarrow \alpha_1\ A' \mid ... \mid \alpha_m\ A' \mid \varepsilon$ nhận được văn phạm tương đương

Đệ quy trái gián tiếp

- Một văn phạm không chứa đệ quy trái trực tiếp nhưng có thể vẫn đệ quy trái.
- Nếu chỉ khử đệ quy trái trực tiếp, văn phạm nhận được có thể vẫn đệ quy trái.

$S \rightarrow Aa \mid b$

$A \rightarrow Sc \mid d$ văn phạm này không đệ quy trái trực tiếp nhưng chứa đệ quy trái gián tiếp

$\underline{S} \Rightarrow Aa \Rightarrow \underline{S}ca$ hay

$\underline{A} \Rightarrow Sc \Rightarrow \underline{A}ac$ đệ quy trái

- Tất nhiên phải khử hết các trường hợp đệ quy trái, nhưng giải thuật khử đệ quy trai gián tiếp không thuộc phạm vi môn học.

Giải thuật khử đệ quy trái

- Đánh số vế trái của sản xuất: $A_1 \ldots A_n$
- **for** i **from** 1 **to** n **do** {
  - **for** j **from** 1 **to** i-1 **do** {

    thay thế mỗi sản xuất

    $A_i \rightarrow A_j \gamma$

    bởi

    $A_i \rightarrow \alpha_1 \gamma \mid \ldots \mid \alpha_k \gamma$

    với $A_j \rightarrow \alpha_1 \mid \ldots \mid \alpha_k$ là các sản xuất không đệ quy trái của A_i

  }

  - Loại bỏ các sản xuất đệ quy trái của A_i}

Ví dụ về khử đệ quy trái

$E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow id \mid (E)$

$\Downarrow$ Khử đệ quy trái trực tiếp

$E \rightarrow T\ E'$

$E' \rightarrow +T\ E' \mid \varepsilon$

$T \rightarrow F\ T'$

$T' \rightarrow *F\ T' \mid \varepsilon$

$F \rightarrow id \mid (E)$

Nhân tử trái

- Với văn phạm

  $E \rightarrow T + E \mid T$

  $T \rightarrow int \mid int * T \mid (E)$

- Làm sao đoán đúng sản xuất?
  - T có hai sản xuất bắt đầu bằng int ->thử xét trường hợp LL(k)
  - E thậm chí còn khó hơn vì hai vế phải đều bắt đầu bằng ký hiệu không kết thúc T -> Cần thực hiện kỹ thuật nhân tử trái (left-factoring)

Nhân tử trái

Tổng quát,

$A \rightarrow \alpha\beta_1 \mid \alpha\beta_2$ α không rỗng và ký hiệu đầu của β_1 và β_2 (nếu có) là khác nhau.

Khi xử lý α ta không biết nên mở rộng

A thành $\alpha\beta_1$ hay

A thành $\alpha\beta_2$

Nhưng nếu ta chuyển đổi văn phạm thành

$A \rightarrow \alpha A'$

$A' \rightarrow \beta_1 \mid \beta_2$ ta chỉ có một lựa chọn cho A, còn Á cũng có thể xử lý tương tự

Nhân tử trái

- Muốn phân tích tiền định, văn phạm cần *nhân tử hóa*
- Cần biến đổi văn phạm thành văn phạm mới thích hợp để phân tích tiền định

if_stmt → if expr then stmt else stmt |
if expr then stmt

Giải thuật nhân tử trái

- Với mỗi ký hiệu không kết thúc A có từ hai vế phải trở lên có chung tiền tố

$$A \rightarrow \alpha\beta_1 \mid \ldots \mid \alpha\beta_n \mid \gamma_1 \mid \ldots \mid \gamma_m$$

chuyển đổi thành

$$A \rightarrow \alpha A' \mid \gamma_1 \mid \ldots \mid \gamma_m$$
$$A' \rightarrow \beta_1 \mid \ldots \mid \beta_n$$

Ví dụ nhân tử trái

S → if E then S | if E then S else S

có thể viết thành

S → if E then S S'

S' → else S | ε

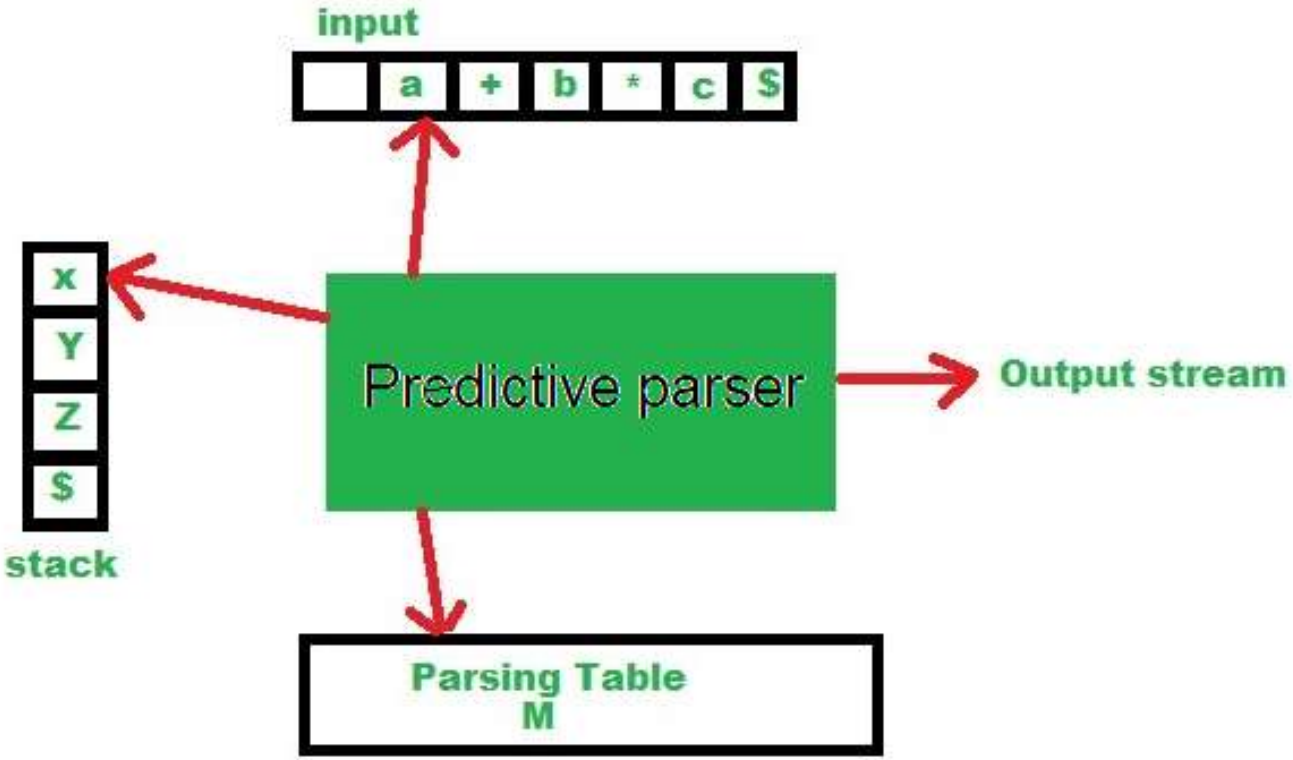
Trong KPL

```
IfSt ::= KW_IF Condition KW_THEN Statement ElseSt

ElseSt ::= KW_ELSE Statement
ElseSt ::= ε
```

- Với các lệnh if lồng nhau, nhánh else thuộc vào if gần nó nhất.

Bộ phân tích cú pháp tiền định



Bộ phân tích cú pháp tiền định

- Với các khái niệm
  - FIRST
  - FOLLOW
- Ta có thể xây dựng bộ phân tích cú pháp mà không đòi hỏi quay lui
- Chỉ có thể xây dựng bộ phân tích cú pháp như vậy cho những văn phạm đặc biệt, đó là loại văn pham LL(k). Trong thực tế chỉ xây dựng cho loại văn phạm LL(1)
- Loại văn phạm như vậy bao gồm văn phạm một số ngôn ngữ lập trình đơn giản, chẳng hạn KPL,PL/0, PASCAL-S
- Thành phần quan trọng nhất của bộ phân tích cú pháp tiền định là bảng phân tích cú pháp.

Bảng phân tích cú pháp

- Với đầu vào của giải thuật: văn phạm G và xâu w, tương tự bộ phân tích cú pháp quay lui, các hành động diễn ra như sau
- Căn cứ
  - Ký hiệu đang xét
  - Ký hiệu đang ở đỉnh stack
- Quyết định duy nhất một hành động
  - Thay thế ký hiệu không kết thúc
  - Chuyển con trỏ sang ký hiệu tiếp
  - Chấp nhận xâu
  - Thông báo lỗi
- Vì dựa trên 2 yếu tố để quyết định hành động, hành động được mô tả dưới dạng bảng, các cột thể hiện ký hiệu đang xét, đầu các dòng thể hiện ký hiệu đỉnh stack. Bảng có tên BẢNG PHÂN TÍCH CÚ PHÁP

Bảng phân tích cú pháp tiền định M

- *M*[*X*, *a*] chỉ ra sản xuất cần áp dụng nếu đỉnh stack là ký hiệu không kết thúc *X* và ký hiệu (từ tố) *a*
- Hành động cần làm là thay *X* trên đỉnh stack bằng vế phải được chỉ ra trong *M[X, a]* .
- Ký hiệu $ biểu diễn EOF. *S* là ký hiệu đầu.
- *M[a,a]* cho phép lấy đi ký hiệu đỉnh stack và đầu đọc chuyển sang ký hiệu tiếp theo trên xâu vào.
- *M[#,$]* stack rỗng, đọc hết xâu, xâu đúng cú pháp

Bảng phân tích cú pháp của văn phạm S→ aSb | c

- Văn phạm đã cho thỏa mãn điều kiện LL(1) nên có thể lập bảng phân tích cú pháp chỉ dựa trên ký hiệu hiện hành và ký hiệu trên đỉnh stack.

	a	b	c	$
S	S → aSb	Error	S → c	Error
a	Push	Error	Error	Error
b	Error	Push	Error	Error
c	Error	Error	Push	Error
#	Error	Error	Error	Accept

Bảng phân tích cú pháp. Lỗi.

- Các ô màu vàng chỉ ra các tình huống gặp lỗi.
  - Nếu gặp ô có giá trị [S,b]
  - Thông báo lỗi có thể là “Không thể suy dẫn ra xâu bắt đầu bởi b từ ký hiệu không kết thúc S”.

Giải thuật xây dựng bảng phân tích

1. Với mỗi sản xuất $A \rightarrow \alpha$ của văn phạm G, thực hiện các bước 2 và 3.
2. Với mỗi ký hiệu kết thúc $a \in FIRST(\alpha)$, thêm $A \rightarrow \alpha$ vào M[A,a].
3. Nếu ε thuộc $FIRST(\alpha)$, thêm $A \rightarrow \alpha$ vào M[A,b] với mỗi b thuộc FOLLOW(A). Nếu ε thuộc $FIRST(\alpha)$, và $ thuộc FOLLOW(A), thì thêm $A \rightarrow \alpha$ vào M[A,$]
4. Các ô M[a, a] với a là ký hiệu kết thúc, thêm hành động "Push" (đầy)
5. M[#,$] = "Accept" (nhận)
6. Các ô còn lại đánh dấu là "Error" (lỗi).

Ví dụ phân tích xâu aacbb với bộ phân tích cú pháp LL(1)

Stack	Input	Action
S#	aacbb$	$S \rightarrow aSb$
aSb#	aacbb$	push
Sb#	acbb$	$S \rightarrow aSb$
aSbb#	acbb$	push
Sbb#	cbb$	$S \rightarrow c$
cbb#	cbb$	push
bb#	bb$	push
b#	b$	push
#	$	ACCEPT

Các khái niệm liên quan: FIRST VÀ FOLLOW

- Tính FIRST(X): Có thể coi như ký hiệu kết thúc đầu tiên được sinh từ X
  - Nếu X là ký hiệu kết thúc FIRST(X)={X}
  - Nếu $X \rightarrow \varepsilon$ là một sản xuất thì thêm ε vào FIRST(X)
  - Nếu X là ký hiệu không kết thúc và $X \rightarrow Y_1 Y_2 \ldots Y_n$ là một sản xuất ,
    - Thêm FIRST(Y_1) vào FIRST(X)
    - Thêm FIRST(Y_{i+1}) vào FIRST(X) nếu FIRST(Y_1),. . . FIRST(Y_j) chứa ε
- Tính FIRST(α) tương tự bước thứ ba trong tính FIRST(X)

Tính FOLLOW

- FOLLOW(S) chứa ε (EOF)
- Với các sản xuất dạng $A \rightarrow \alpha B \beta$, mọi ký hiệu trong FIRST(β) trừ ε tham gia vào FOLLOW(B)
- Với các sản xuất dạng $A \rightarrow \alpha B$ hoặc $A \rightarrow \alpha B \beta$ trong đó FIRST(β) chứa ε, FOLLOW(B) chứa mọi ký hiệu của FOLLOW(A) và \$ (hoặc ε)

Bộ phân tích cú pháp tiền định

Vào: Văn phạm phi ngữ cảnh LL(1) G
Xâu w

Các thành phần cơ bản

- Stack
- Bảng phân tích
- Băng vào
- Chương trình phân tích

Hoạt động của bộ phân tích cú pháp

- Nếu stack còn lại # (đáy), đầu đọc chỉ $ (EOF), dừng và đoán nhận xâu.
- Nếu X=a (ký hiệu kết thúc đang xét trên xâu vào) và không là $, xóa X trên đỉnh stack , chuyển đầu đọc sang ô kế tiếp (hành động push).
- Nếu X là ký hiệu không kết thúc, bộ PTCP tra bảng phân tích cú pháp M, tìm ô M[X,a], thay thế ký hiệu đỉnh stack (X) bằng vế phải sản xuất trong ô (nếu có). Nếu là ô rỗng -> Error, gọi thủ tục thông báo lỗi.

Ví dụ phân tích với văn phạm chứa ε – sản xuất

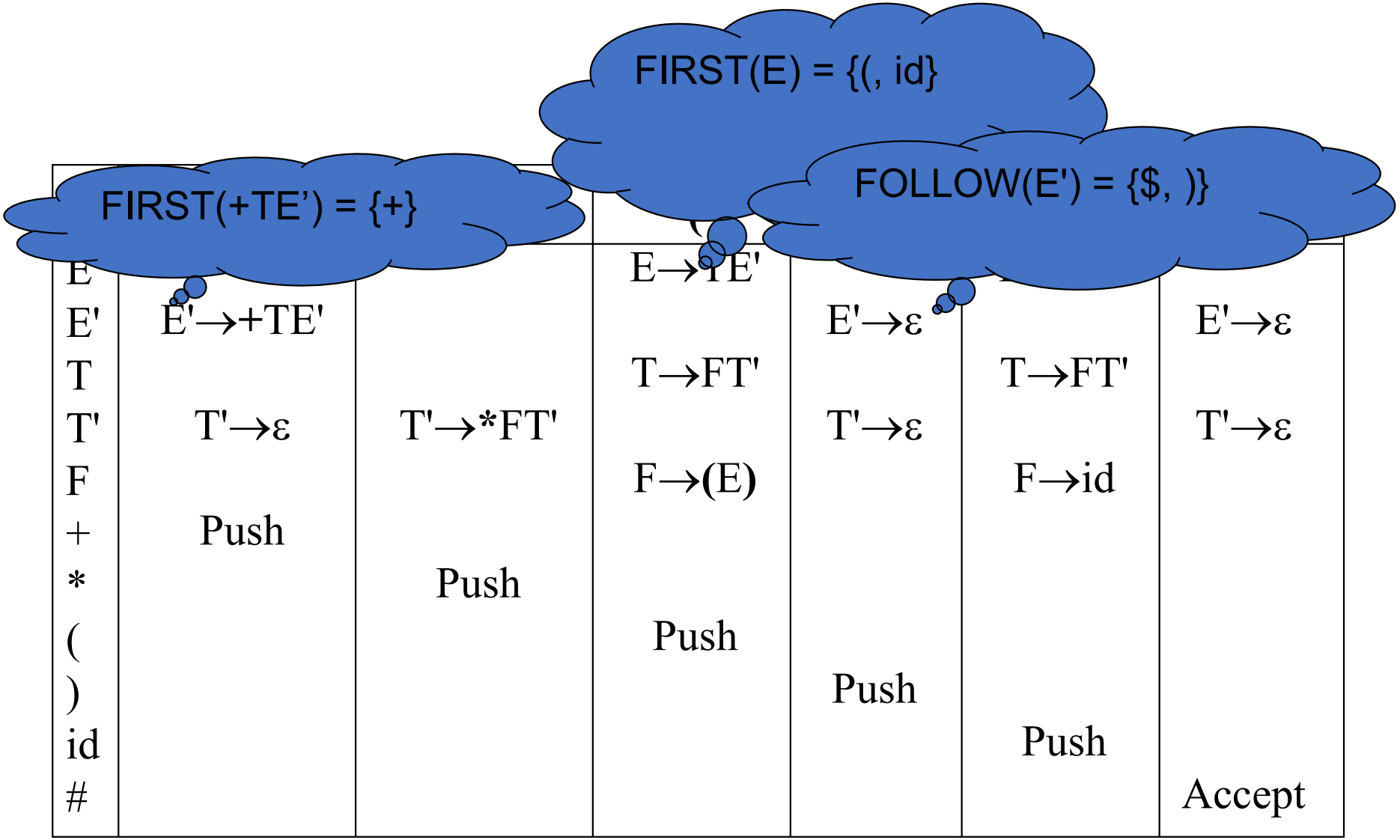
- Văn phạm:

$$E \rightarrow TE'$$
$$E' \rightarrow +TE' \mid \varepsilon$$
$$T \rightarrow FT'$$
$$T' \rightarrow *FT' \mid \varepsilon$$
$$F \rightarrow (E) \mid id$$

FIRST(TE') = {(, id}
FIRST(+TE') = {+}
FOLLOW(E') = {$,)}
FIRST(*FT') = {*}
FOLLOW(T') = {+, $,)}
FIRST((T)) = {(, id}
FIRST(id) = {id}

Văn phạm này LL(1)
có thể xây dựng bộ phân tích tiền định

Bảng phân tích



Phân tích xâu vào id*id sử dụng bảng phân tích và stack

Bước	Stack	Xâu vào	Hành động kế tiếp
1	E#	id*id$	E→TE'
2	TE'#	id*id$	T→FT'
3	FT'E'#	id*id$	F→id
4	idT'E'#	id*id$	push (đẩy id)
5	T'E'#	*id$	T'→*FT'
6	*FT'E'#	*id$	push (đẩy *)
7	FT'E'#	id$	F→id
8	idT'E'#	id$	push (đẩy id)
9	T'E'#	$	T'→ε
10	E'#	$	E' →ε
11	#	$	nhận