جلسه پنجم و ششم

مجموعه Follow(x) (مجموعه دنباله(x))

Follow یک None Terminal (X) :

مجموعه ترمینال هائی است که در شکل های جمله ای بلافاصله سمت راست **x** می آیند(اولین سمت راست آن)

$$\alpha \to BXa\gamma \to follow(X) = \{a\}$$

قواعد بدست آوردن مجموعه $follow(\alpha)$

- اگر α، $start\ \ symbol$ باشد آنگاه $ عضوی از $follow(\alpha)$ می باشد.
- اگر مولدی به صورت $B \to A\alpha$ داشته باشیم، آنگاه هر چیزی در $first(\alpha)$ به جز ε به مجموعه $follow(A)$ اضافه می شود.
- اگر مولدی به صورت $B \to \alpha A$ داشته باشیم، آنگاه هر چیزی در $follow(B)$ به مجموعه $follow(A)$ اضافه می شود.
- اگر مولدی به صورت $A \to \alpha B\beta$ وجود داشته باشد، آنگاه هر چیزی در $first(\beta)$ به جز ε به مجموعه $follow(B)$ اضافه می شود.
- مولدی به صورت $A \to \alpha B\beta$ که $first(\beta)$ حاوی ε باشد آنگاه هر چیزی در مجموعه $follow(A)$ به $follow(B)$ اضافه می شود.

مثال.FIRST() و Follow() هر کدام را حساب کنید.

1) $E \to TE'$
2) $E' \to +TE' \mid \varepsilon$
3) $T \to FT'$
4) $T' \to *FT' \mid \varepsilon$
5) $F \to (E) \mid id$

پاسخ

$first(T') = \{ *, \varepsilon \}$
$first(F) = \{ (, id \}$
$first(E') = \{ +, \varepsilon \}$
$first(T) = \{ (, id \}$
$first(E) = \{ (, id \}$

$follow(E) = \{ \$,) \}$
$follow(E') = \{ \$,) \}$
$follow(T) = \{ +, ,\$,) \}$
$follow(T') = \{ +, ,\$,) \}$
$follow(F) = \{ *, +, \$,) \}$

مثال.FIRST() و Follow() هر کدام را حساب کنید.

1) $S \to ABD$
2) $A \to a \mid BSB$
3) $B \to b \mid D$
4) $D \to d \mid \varepsilon$

پاسخ

$first(S) = \{ a, b, d \}$
$first(A) = \{ a, b, d \}$
$first(B) = \{ b, d\ \varepsilon \}$
$first(D) = \{ d, \varepsilon \}$

$follow\ (S) = \{ \$, b, d \}$
$follow\ (A) = \{ \$, b, d \}$
$follow\ (B) = \{ a, b, d, \$ \}$
$follow\ (D) = \{ a, b, d, \$ \}$

مثال.FIRST() و Follow() هر کدام را حساب کنید.

1) $S \to BCd \mid \varepsilon$
2) $A \to AaSb \mid SbC \mid \varepsilon$
3) $B \to b \mid \varepsilon$
4) $C \to c \mid B$

پاسخ

first(S) = { ε , b,c,d }
first(A) = { ε , b , c,d }
first(B) = { b , ε}
first(C) = { c,b,ε}

follow (S) = { b ,$}
follow (A) = { a }
follow (B) = { a , b , c , d }
follow (C) = { a , d }

گرامر زیر که در آن S , E و F سمبل های غیر ترمینال و $+$, $-$,$*$ سمبل های ترمینال هستند را در نظر بگیرید مجموعه $First(E)$ و $Follow(E)$ را بیابید.

$$S \to +E-$$
$$E \to F \mid *$$
$$F \to +E* \mid \lambda$$

پاسخ $first(E)=\{+,*\}$ $follow(E)=\{-,*\}$

☐ ترمینال ها **folow** ندارند.

گرامر بالا به پایین پیشگوکننده:

☐ مبهم نباشد

☐ بازگشتی چپ نداشته باشد.

☐ فاکتورگیری از چپ بر روی آن اعمال شده باشد.

بازگشتی چپ(left Rotation)

۱- بازگشتی چپ آشکار(بدیهی) $A \to A\alpha$

۲- بازگشتی چپ ضمنی $A \to B\alpha \mid c$, $B \to Ad \mid Bc \mid d$ این بازگشتی چپ برای **A** ضمنی است $A \to B\alpha \to Ad\alpha$

حذف بازگشتی چپ بدیهی:

چنانکه قانونی به شکل $A \to A\alpha_1 \mid A\alpha_2 \mid \ldots \mid Aa_m \mid \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$ داشته باشیم به طوریکه β_i ها با A شروع نشوند و هیچکدام از α_i ها نبایستی ε باشند می توان مشکل بازگشتی چپ را با جایگزینی دو قانون زیر حل کرد

$$A \to \beta_1 A' \mid \beta_2 A' \mid \ldots \mid \beta_n A'$$
$$A' \to \alpha_1 A' \mid \alpha_2 A' \mid \cdots \mid \alpha_m A' \mid \varepsilon$$

مثال. بازگشتی گرامر زیر را حذف کنید.

$$E \to EAc \mid Ea \mid d \mid Bc \mid Ed$$

پاسخ
$$E \to dE' \mid BcE'$$
$$E' \to AcE' \mid aE' \mid dE' \mid \varepsilon$$

حذف بازگشتی چپ ضمنی:

☐ گرامر**G** قاعده اپسیلون نداشته باشد.

☐ دورهای $A \xrightarrow{+} A$ در گرامر**G** نباشند

الگوریتم

$for\ i=1\ to\ n\ do$
$\quad for\ j=1\ to\ (i-1)\ do$

- به جاي هر قانون به شكل كلي $A_i \to A_j\beta$ به طوريكه $A_j \to \alpha_1 \mid \alpha_2 \mid \alpha_3 \mid \ldots \mid \alpha_k$ قرار بده
$A_i \to \alpha_1\beta \mid \alpha_2\beta \mid \ldots \mid \alpha_k\beta$
- حال اگر A_i داراي بازگشتي چپ اشكار است، آن را حذف كن.

مثال. بازگشتی چپ را از گرامر زیر حذف کنید.

$$S \to Aa \mid b$$
$$A \to Ac \mid Sd$$

برای حذف بازگشتی ضمنی چپ داریم

$$A \to Ac \mid Aad \mid bd$$
$$S \to Aa \mid b$$

حال با حذف بازگشتی چپ آشکار داریم

$$S \to Aa \mid b$$
$$A \to bdA'$$
$$A' \to cA' \mid adA' \mid \varepsilon$$

فاکتور چپ.

یک گرامر دارای فاکتور چپ است اگر در سمت چپ حداقل دو تا از تولیدات آن یک عنصر مشترک باشد. مثلا $A \to a\beta_1 \mid a\beta_2$ دارای فاکتور چپ α می باشد به روشنی می توان دید که از این نوع گرامر ها نمی توان در پارسر های بالا به پایین استفاده کرد چون در هنگام پارس کردن نمی توان تعیین کرد که کدام یک از این قوانین تولید ما را به جواب می رساند. برای رفع مشکل بالا می توان از α فاکتور گرفت.

□ در حالت کلی اگر داشته باشیم $A \to a\beta_1 \mid a\beta_2 \mid \ldots \mid a\beta_m \mid \delta_1 \mid \delta_n \mid \ldots \mid \delta_n$ آنگاه با فاکتور گیری از α می توان فاکتور چپ را به صورت زیر حذف نمود

$$A \to \alpha A' \mid \delta_1 \mid \delta_2 \mid \ldots \mid \delta_n$$
$$A' \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_m$$

مثال گرامر زیر را در نظر بگیرید

$$S \to iEtS$$
$$S \to iEtSeS \mid a$$
$$E \to b$$

پس از فاکتور گیری داریم

$$S \to iEtSS' \mid a$$
$$S' \to eS \mid \varepsilon$$
$$E \to b$$

مثال گرامر زیر را در نظر بگیرید

$$A \to abcE \mid abcdf \mid abcdT \mid abcdmT'$$

فاکتور گیری مرحله اول

$$A \to abcB'$$
$$B' \to E \mid df \mid dT \mid dmT'$$

فاکتور گیری مرحله دوم

$$A \to abcB'$$
$$B' \to E \mid dZ'$$
$$Z' \to f \mid T \mid mT'$$

ساخت جدول تجزیه(parse Table)

گرامر ورودی جدول تجزیه بایستی خواص زیر را داشته باشد.

- گرامر مبهم نباشد
- بازگشتی چپ نداشته باشد
- فاکتور گیری از چپ روی آن اعمال شده باشد.

جدول تجزیه:

- ماتریس دو بعدی است
- سطر ها غیر پایانه ها را نشان می دهد
- ستون ها پایانه ها را نشان می دهد.

برای تشکیل جدول پارسینگ به صورت زیر عمل می کنیم

۱- برای هر قانون به شکل $A \to \alpha$ مراحل زیر را تکرار کنید

a) به ازای $a \in first(\alpha)$ شماره قاعده $A \to \alpha$ را در محل $M[A,a]$ قرار می دهیم.

b) اگر $\varepsilon \in first(\alpha)$ آنگاه به ازای هر $B \in follow(A)$ شماره قاعده $A \to \alpha$ را در محل $M[A,b]$ قرار می دهیم.

c) اگر $\varepsilon \in first(\alpha)$ و $\$ \in follow(A)$ آنگاه در محل $M[A,\$]$ شماره قاعده $A \to \varepsilon$ را قرار می دهیم.

مثال.

1) $E \to TE'$

$2-3)\ E' \to +TE' \mid \varepsilon$

4) $T \to FT'$

$5-6)\ T' \to *FT' \mid \varepsilon$

$7-8) F \to (E) \mid id$

$first\ (E) = \{\ (\ ,\ id\ \}$

$first\ (T) = \{\ (\ ,\ id\ \}$

$first\ (E') = \{\ +\ ,\ \varepsilon\ \}$

$first\ (F) = \{\ (\ ,\ id\ \}$

$first\ (T') = \{\ *\ ,\ \varepsilon\ \}$

$follow\ (E) = \{\ \$\ ,\ )\ \}$

$follow\ (E') = \{\ \$\ ,\ )\ \}$

$follow\ (T) = \{\ +\ ,\ ,\$,\ )\ \}$

$follow\ (T') = \{\ +\ ,\ ,\$,\ )\ \}$

$follow\ (F) = \{\ *\ ,\ +\ ,\ \$\ ,\ )\ \}$

	+	*	(	)	id	$
E	E	E	1	E	1	E
E'	2	E	E	3	E	3
T	E	E	4	E	4	E
T'	6	5	E	6	E	6
F	E	E	7	E	8	E

مثال. برای گرامر زیر جدول پارسینگ تشکیل دهید.

(1,2,3) $S \to A;\mid for(A;C;A)S \mid B$

(4,5) $A \to V = E \mid \varepsilon$

(6,7) $C \to E \mid \varepsilon$

(8) $E \to V$

(9) $V \to idX$

(10,11) $X \to 0V \mid \varepsilon$

(12) $B \to \{L\}$

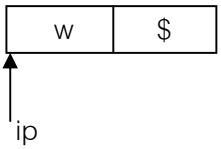
(13,14} $L \to SL\varepsilon \mid \varepsilon$

	;	for	(	)	=	id	0	{	}	$
S	1	2	E	E	E	1	E	3	E	E
A	5	E	E	5	E	4	E	E	E	E
C	7	E	E	E	E	6	E	E	E	E
E		E	E	E	E	8	E	E	E	E
V		E	E	E	E	9	E	E	E	E
X	11	E	E	11	11	E	10	E	E	E
B		E	E	E	E	E	12	E	E	E
L	13	13	E	E	E	13	E	13	14	E

	S	A	C	E	V	X	B	L
$first()$	id ; for {	$id\ \varepsilon$	$id\ \varepsilon$	id	id	$0\ \varepsilon$	{	id ; for { ε
$follow()$	$\$ id$; for {	;)	;	;)	= ;)	= ;)	$\$ id$; for {	}

الگوریتم تجزیه پیشگو کننده:LL(1)

- به انتهای پشته ورودی $ اضافه کنید.
- Ip را طوری مقدار دهی کنید که به ابتدای رشته w اشاره کند.
- به پشته $ و startsy را اضافه کنید(push)
- رشته ورودی توکن ها هستند
- در پشته، ترمینال، non ترمینال و $ قرار دارد.



با استفاده از جدول تجزیه و همچنین یک پشته به راحتی می توان عمل تجزیه بالا به پایین ر انجام داد.**روش کار به این صورت است** در ابتدا به انتهای رشته ورودی علامت $ که نشان دهنده خاتمه فایل است افزوده می گردد از سوی دیگر علامت سرترم گرامر نیز به همراه $ در داخل پشته قرار می گیرد. عمل تجزیه زمانی با موفقیت به پایان می رسد که در ورودی $ و در پشته تنها علامت $ باقی مانده باشد.

پس از قرار دادن سرترم گرامر و $ در پشته از ورودی یک علامت خوانده می شود، اگر فرض کنیم که علامت موجود در ورودی a و عنصر بالای پشته x باشد حالات زیر ممکن است اتفاق بیفتد

۱- اگر $a = X = \$$ باشد عمل تجزیه با موفقیت به پایان رسیده است

۲- اگر $a = X \neq \$$ باشد، X از بالای پشته حذف می شود و عنصر بعدی از ورودی خوانده می شود

۳- اگر X یک ترمینال باشد و $a \neq x$ خطای نحوی رخ میدهد

۴- اگر X یک ترم میانی(غیر ترمینال) باشد برنامه به محل $M[X,a]$ در جدول مراجعه می کند در این جا دو حالت امکان دارد

الف) قاعده ای به صورت $X \to UVW$ در خانه $M[X,a]$ وجود دارد در این حالت عنصر X از پشته pop شده و UVW به درون پشته push می شود به طوریکه عنصر U در بالای پشته قرار می گیرد.

ب) خانه $M[X,a]$ خالی است در این حالت یک خطای نحوی رخ میدهد.

توضیحات بالا را به صورت الگوریتم زیر نیز می توان نوشت.

```
set ip to point to the first symbol of W$;
repeat
    let X be the top stack symbpl and α the symbol pointed to by ip;
    if X is a terminal or $ then
        if X = α then
            pop X from the stack and Advance ip;
        else error();
    else  /* X is nontermina l */
      if M[X,a] = X → Y1Y2...Yk then begin
            pop X from the stack;
            push Yk,Yk-1,...Y1 on to the stack, with Y1 on top;
            output the production X → Y1Y2...Yk
      end
      else error
Until X = $  /* stack is empty */
```

عمل تجزیه برای جمله $id + id * id$ در زیر تشریح شده است(با توجه به جدول تجزیه صفحه قبل).

پشته	ورودی	قانون استفاده شده
$E	$id + id * id$	
$E'T	$id + id * id$	$E \to TE'$
$E'T'F	$id + id * id$	$T \to FT'$
$E'T'id	$id + id * id$	$F \to id$
$E'T'	$+ id * id$	
$E'	$+ id * id$	$T' \to \varepsilon$
$E'T+	$+ id * id$	$E' \to +TE'$
$E'T	$id * id$	
$E'T'F	$id * id$	$T \to FT'$
$E'T'id	$id * id$	$F \to id$
$E'T'	$* id$	
$E'T'F*	$* id$	$T' \to *FT'$
$E'T'F	id	
$E'T'id	id	$F \to id$
$E'T'	$	
$E'	$	$T' \to \varepsilon$
$	$	$E' \to \varepsilon$

گذر از خطا

خطا

۱- بالای پشته پایانه باشد و با ورودی فعلی تطابق نداشته باشد

۲- بالای پشته غیر پایانه ولی محل $M[X, token]$ خالی $(null)$ باشد.

دو روش برای اصلاح خطا های نحوی در روش $LL(1)$ وجود دارد.

- *panic Mode*
- *pharse level*

در روش *panic Mode* اگر پارسر با مراجعه به یک خانه خالی جدول تجزیه یک خطای نحوی بیابد، آنقدر از ورودی حذف می کند تا به یکی از اعضای مجموعه ای موسوم به مجموعه *synchornizing* برسد در روش *panic Mode* با ازای هر غیر پایانه در گرامر یک مجموعه *synchornizing* در نظر گرفته میشود. کارائی روش *panic Mode* نیز بستگی به انتخاب مناسب مجموعه *synchornizing* دارد. این

مجموعه باید به گونه ای انتخاب شود که عمل تجزیه بتواند بدون حذف قسمت زیادی از ورودی به کار خود ادامه دهد. یک انتخاب مناسب در نظر گرفتن مجموعه *follow* هر غیر پایانه ای به عنوان مجموعه *synchornizing* آن غیر پایانه است. با این وجود در نظر گرفتن مجموعه *follow* تنها برای *synchornizing* کافی نیست. برای این که حذف کمتری در برنامه صورت بگیرد می توان نماد های بیشتری را به این مجموعه افزود، مثلا می توان مجموعه *first* غیر پایانه ها را نیز به مجموعه *synchornizing* آنها افزود.

توسعه الگوریتم تجزیه جهت اعمال گذر از خطا

```
tos = 0;
stack[tos++] = $;
stack[tos++] = start symbol;
token = get token();
do{ // X is top of the stack
if X is a terminal or X = $;
    if X = token{
        pop X;
        token = get token();}
    else
       pop X;
else // X is a non-terminal;
if M[X,token] = NoT NULL && X ≠ synch;
{
pop X;
push Y_k, Y_{k-1}, ..., Y_1
}
else if X = synch;
    if X is not sign N.T in stack;
    pop X;
else
    token = get token();
else
token = get token();
} while(Token ≠ $);
```