第五章

2．对下面的文法G：

E🡪TE/

E/🡪+E|ε

T🡪FT/

T/🡪T|ε

F🡪PF/

F/🡪\*F/|ε

P🡪(E)|a|b|^

1. 计算这个文法的每个非终结符的FIRST集和FOLLOW集。
2. 证明这个方法是LL(1)的。
3. 构造它的预测分析表。
4. 构造它的递归下降分析程序。

解：  
（1）计算这个文法的每个非终结符的FIRST集和FOLLOW集。

FIRST集合有：

FIRST(E)=FIRST(T)=FIRST(F)=FIRST(P)={(,a,b,^};

FIRST(E/)={+,ε}

FIRST(T)=FIRST(F)=FIRST(P)={(,a,b,^};

FIRST(T/)=FIRST(T)∪{ε}={(,a,b,^,ε};

FIRST(F)=FIRST(P)={(,a,b,^};

FIRST(F/)=FIRST(P)={\*,ε};

FIRST(P)={(,a,b,^};

FOLLOW集合有：

FOLLOW(E)={),#};

FOLLOW(E/)=FOLLOW(E)={),#};

FOLLOW(T)=FIRST(E/)∪FOLLOW(E)={+,),#};//不包含ε

FOLLOW(T/)=FOLLOW(T)=FIRST(E/)∪FOLLOW(E)={+,),#};

FOLLOW(F)=FIRST(T/)∪FOLLOW(T)={(,a,b,^,+,),#};//不包含ε

FOLLOW(F/)=FOLLOW(F)=FIRST(T/)∪FOLLOW(T)={(,a,b,^,+,),#};

FOLLOW(P)=FIRST(F/)∪FOLLOW(F)={\*,(,a,b,^,+,),#};//不包含ε

(2)证明这个方法是LL(1)的。

各产生式的SELECT集合有：

SELECT(E🡪TE/)=FIRST(T)={(,a,b,^};

SELECT(E/🡪+E)={+};

SELECT(E/🡪ε)=FOLLOW(E/)={),#}

SELECT(T🡪FT/)=FIRST(F)={(,a,b,^};

SELECT(T/🡪T)=FIRST(T)={(,a,b,^};

SELECT(T/🡪ε)=FOLLOW(T/)={+,),#};

SELECT(F🡪PF/)=FIRST(P)={(,a,b,^};

SELECT(F/🡪\*F/)={\*};

SELECT(F/🡪ε)=FOLLOW(F/)={(,a,b,^,+,),#};

SELECT(P🡪(E))={(}

SELECT(P🡪a)={a}

SELECT(P🡪b)={b}

SELECT(P🡪^)={^}

可见，相同左部产生式的SELECT集的交集均为空，所以文法G[E]是LL(1)文法。

(3)构造它的预测分析表。

文法G[E]的预测分析表如下：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | + | \* | ( | ) | a | b | ^ | # |
| E |  |  | 🡪TE/ |  | 🡪TE/ | 🡪TE/ | 🡪TE/ |  |
| E/ | 🡪+E |  |  | 🡪ε |  |  |  | 🡪ε |
| T |  |  | 🡪FT/ |  | 🡪FT/ | 🡪FT/ | 🡪FT/ |  |
| T/ | 🡪ε |  | 🡪T | 🡪ε | 🡪T | 🡪T | 🡪T | 🡪ε |
| F |  |  | 🡪PF/ |  | 🡪PF/ | 🡪PF/ | 🡪PF/ |  |
| F/ | 🡪ε | 🡪\*F/ | 🡪ε | 🡪ε | 🡪ε | 🡪ε | 🡪ε | 🡪ε |
| P |  |  | 🡪(E) |  | 🡪a | 🡪b | 🡪^ |  |

(4)构造它的递归下降分析程序。

对每个非终结符写出不带回溯的递归子程序如下：

char CH;//存放当前的输入符号

void P\_E()//非终结符E的子程序

{

if(IsIn(CH,FIRST\_TEP)) //FIRST\_TEP为T🡪TE/ 的右部的FIRST集合，产生式E🡪TE/

{

P\_T();

P\_EP();

}

else ERR;  
}

void P\_EP()//非终结符E/的子程序

{

if(CH==’+’) //产生式E/🡪+E

{

READ(CH);

P\_E();

}

else//产生式E/🡪ε

{

if(IsIn(CH,FOLLOW\_EP)) //FOLLOW\_EP为E/的FOLLOW集合

return ;

else ERR;  
}

}

void P\_T()//非终结符T的子程序

{

if(IsIn(CH,FIRST\_FTP)) //FIRST\_TEP为T🡪FT/ 的右部的FIRST集合，产生式T🡪FT/

{

P\_F();

P\_TP();

}

else ERR;  
}

void P\_TP()//非终结符T/的子程序

{

if(IsIn(CH,FIRST\_T)) //FIRST\_T为产生式T/🡪T的右部的FIRST集合，产生式T/🡪T

{

P\_T();

}

else//产生式T/🡪ε

{

if(IsIn(CH,FOLLOW\_TP)) //FOLLOW\_TP为T/的FOLLOW集合

return ;

else ERR;  
}

}

void P\_F()//非终结符F的子程序

{

if(IsIn(CH,FIRST\_PFP)) //FIRST\_PFP为F🡪PF/ 的右部的FIRST集合，产生式F🡪PF/

{

P\_P();

P\_FP();

}

else ERR;  
}

void P\_FP()//非终结符F/的子程序

{

if(CH==’\*’) //产生式F/🡪\*F/

{

READ(CH);

P\_FP();

}

else//产生式F/🡪ε

{

if(IsIn(CH,FOLLOW\_FP)) //FOLLOW\_FP为F/的FOLLOW集合

return ;

else ERR;  
}

}

void P\_P()//非终结符P的子程序

{

if(CH==’(‘)

{

READ(CH);

P\_E();

if(CH==’)’) READCH(CH);

else

ERR;

}

else if(CH==’a’) READ(CH);

else if(CH==’b’) READ(CH);

else if(CH==’^’) READ(CH);

else ERR;

}

4证明下述文法不是LL（1）的。

S->C$ C->bA|aB A->a|aC|bAA B->b|bC|aBB 你能否构造一等价的文法，使其是LL（1）的，并给出判断过程。

【解】因为SELECT(A->a)∩SELECT(A->aC)≠Ф,根据LL（1）文法的判定条件：

(1)文法不含左递归

(2)对于文法U的任意两个不同的规则有: Select(U→α) ∩ Select(U→)=Φ一个文法若满足以上条件，称该文法G为LL(1)文法。得出该文法不是LL（1）文法。 该文法含公共因子，消除后的文法为：

S->C$

C-> bA|aB

A->aA'|bAA

A’->C|ε

1. >bB'| aBB
2. B’->C|ε

【证明】因为SELECT(C-> bA) ∩SELECT(C->aB)= Φ SELECT(A->Aa) ∩SELECT(A->bAA) = Φ SELECT(A’->C) ∩SELECT(A’->ε)=(FIRST(C)-{ ε})∩FOLLOW(A’) ≠Ф 因此消除公共因子后得到文法也不是LL（1）文法。