Top Down Parsing

- p206 exercise 4.2.1: [24 pts.] Consider the context-free grammar:

G[S]: S→SS+|SS*|a

and the token String : aa+a*

- [5 pts.](1) Rewrite this grammar to left factor and eliminate left recursion.
- [5 pts.](2) Compute First Set and Follow Set for the rewritten grammar.
- [2 pts.](3) Is it LL(1) grammar, why?
- [7 pts.](4) Construct the LL(1) parsing table.
- [5 pts.](5) Match the input string aa+a* based on the parsing table and LL(1) algorithm.

Top Down Parsing

- (1) Rewrite this grammar to left factor and eliminate left recursion.
- 左因子和左递归
  - 含有左因子和左递归文法一定不是LL(1)文法
  - 通过提取左因子和消除左递归改写文法(rewrite)

Grammar： S→SS+|SS*|a

1）提取左因子：

S →SSO|a

O →+|*

需要进一步消除左递归

Top Down Parsing

- 左因子和左递归
  - 含有左因子和左递归文法一定不是LL(1)文法
  - 通过提取左因子和消除左递归改写文法(rewrite)

1）提取左因子：

S →SSO|a

O →+|*

2）消除左递归：

S →aS’

S’ → SOS’| ε

O →+|*

Top Down Parsing

- (2) Compute First Set and Follow Set for the rewritten grammar.
  - 计算nullable nonterminals
  - 计算所有nonterminal的First Set
  - 计算所有nonterminal的Follow Set

S →aS’
S’ → SOS’| **ε**　　S’为Nullable
O →+|*

- Algorithm for computing First(A) for each grammar symbol A ($A \in V_T \cup V_N$)
  - For all $a \in V_T$, then First(a)={ a };
  - For all $A \in V_N$,if $A \Rightarrow^* \varepsilon$, then First(A)={ε} else First(A)={ } ;
  - For each production $A \rightarrow X1 \ldots Xj \ldots Xn$, First(A)=First(A)∪**SectionFirst(X1…Xj…Xn)**;
  - Repeat step 3 until there is no change to any First set;

SectionFirst($X_1 \ldots X_j \ldots X_n$)

SectionFirst($X_1 \ldots X_j \ldots X_n$)

$= (First(X_1) - \{\varepsilon\}) \cup (First(X_2) - \{\varepsilon\}) \cup \ldots \cup (First(X_j) - \{\varepsilon\}) \cup First(X_{j+1})$

X_{j+1} is the first symbol that is **not nullable** in the right-hand of production

- if X_1 is not nullable, then SectionFirst($X_1 \ldots X_j \ldots X_n$) =First($X_1$)
- if X_1 is nullable, then SectionFirst($X_1 \ldots X_j \ldots X_n$) =First($X_1$) -{ε}, and continue to see X_2. Stop until X_i is not nullable .
- if $X_1 \ldots X_n$ are all nullable, then SectionFirst($X_1 \ldots X_n$)= $(First(X_1) - \{\varepsilon\}) \cup (First(X_2) - \{\varepsilon\}) \cup \ldots \cup (First(Xn) - \{\varepsilon\}) \cup \{\varepsilon\}$

Top Down Parsing

- (2) Compute First Set and Follow Set for the rewritten grammar.
  - 计算所有nonterminal的First Set

S →aS’
S’ → SOS’| **ε**
O →+|*

First(S) ={ a }
First(S’) ={ **ε** , a }
First(O)={ +, * }

Compute FOLLOW(A) for every nonterminal A

1. S is the start symbol, Follow(S)={$};for all $A \in V_N$, and $A \neq S$,Follow(A)={ };
2. For each production $B \rightarrow \alpha A \gamma$, for each A that is a nonterminal do

   $Follow(A)=Follow(A) \cup (First(\gamma) - \{\varepsilon\})$

   if $\varepsilon \in First(\gamma)$ then add Follow(B) to Follow(A)
3. Repeat 2,until there is no change to any follow set

Top Down Parsing

- (2) Compute First Set and Follow Set for the rewritten grammar.
  - 计算所有nonterminal的Follow Set

S →aS’

S’ → SOS’ | **ε**

O →+|*

First(S) ={ a }

First(S’) ={ **ε** , a }

First(O)={ +, * }

	Origin	Pass 1	Pass 2
S	$	$, +, *	$, +, *
S’		$	$, +, *
O		a, $	a, $, +, *

Follow(S) ={$, +, * }

Follow(S’) ={$, +, * }

Follow(O)={a, $, +, * }

Top Down Parsing

- (3) Is it LL(1) grammar, why?
  - 针对同一nonterminal的多个产生式，计算其每一个产生式右部文法符号串的First Set，根据计算结果判定是否满足LL(1)的判定条件1
  - 如果**ε** ∈First(nonterminal)，则根据该nonterminal的Follow Set判定是否满足LL(1)的判定条件2

S →aS’

S’ → SOS’ | **ε**

O →+|*

First(S) ={ a }

First(S’) ={ **ε** , a }

First(O)={ +, * }

First(SOS’)=First(S) ={a}

First(**ε**) ={ **ε** }

First(SOS’) ∩ First(**ε**) = **Φ**

First(+)= {+}

First(*) ={*}

First(+) ∩ First(*) = **Φ**

满足LL(1)的判定条件1

Top Down Parsing

- (3) Is it LL(1) grammar, why?
  - 针对同一nonterminal的多个产生式，计算其每一个产生式右部的First Set，根据计算结果判定是否满足LL(1)的判定条件1
  - 如果**ε** ∈First(nonterminal)，则根据该nonterminal的Follow Set判定是否满足LL(1)的判定条件2

S →aS'

S' → SOS' | **ε**

O →+|*

First(S) ={ a }

First(S') ={ **ε** , a }

First(O)={ +, * }

Follow(S')={**$**, +, *}

First(S') ∩ Follow(S')={ **ε** , a } ∩ {**$**, +, *}= **Φ**

满足LL(1)的判定条件2 ,因此该文法是LL(1)文法!

Top Down Parsing

- 4、构建LL(1)分析表
  - Repeat the following two steps for each nonterminal A and production choice A→ α
    - For each token 'a' in First(α), add A→ α to the entry M[A,a]
    - if ε is in First(α), for each element 'a' of Follow(A) (token or $), add A→ α to M[A,a]

Top Down Parsing

S → aS’

S’ → SOS’ | ε

O → + | *

First(S) ={ a }
First(S’) ={ ε , a }
First(O)={ +, * }

First(aS’) ={a}
First(SOS’)=First(S)={ a}
First(ε)= {ε}
First(+) ={+}
First(*) = {*}

Follow(S’)={$, +, *}

	a	+	*	$
S	S→aS’			
S’	S’ → SOS’	S’ → ε	S’ → ε	S’ → ε
O		O →+	O →*	

	a	+	*	$
S	S→aS’			
S’	S’ → SOS’	S’ → ε	S’ → ε	S’ → ε
O		O →+	O →*	

- LL(1)分析过程模拟

step	stack	input	action
1	$S	aa+a*$	S→aS’
2	$S’a	aa+a*$	match
3	$S’	a+a*$	S’ → SOS’
4	$S’OS	a+a*$	S→aS’
5	$S’OS’a	a+a*$	match
6	$S’OS’	+a*$	S’ → ε
7	$S’O	+a*$	O →+
8	$S’+	+a*$	match

	a	+	*	$
S	S→aS’			
S’	S’ → SOS’	S’ → ε	S’ → ε	S’ → ε
O		O →+	O →*	

step	stack	input	action
9	$S’	a*$	S’ → SOS’
10	$S’OS	a*$	S→aS’
11	$S’OS’a	a*$	match
12	$S’OS’	*$	S’ → ε
13	$S’O	*$	O →*
14	$S’*	*$	match
15	$S’	$	S’ → ε
16	$	$	**accept**