German University in Cairo Department of Computer Science Assoc. Prof. Haythem O. Ismail

CSEN 1003 Compiler, Spring Term 2020 Practice Assignment 5

Discussion: 1.03.20 - 04.03.20

Exercise 5-1

Consider the following CFG:

$$\begin{array}{ccc} S & \rightarrow & \mathsf{0}T\mathsf{1}S \mid \varepsilon \\ T & \rightarrow & \mathsf{0}T\mathsf{1} \mid \varepsilon \end{array}$$

a) Compute the FIRST and FOLLOW sets.

Solution:

$$\operatorname{first}(S) = \{\mathbf{0}, \varepsilon\}$$

$$\mathrm{first}(T) = \{\mathbf{0}, \varepsilon\}$$

$$FOLLOW(S) = \{\$\}$$

$$FOLLOW(T) = \{1\}$$

b) Compute the parsing table.

Solution:

	Input Symbol				
Non-terminal	0	1	\$		
S	0 <i>T</i> 1 <i>S</i>		ε		
T	0 <i>T</i> 1	arepsilon			

c) Prove that this grammar is LL(1).

Solution:

The grammar is LL(1) because every entry in parsing table contains at most one rule.

d) What language does this grammar recognize?

Solution:

$$\{0^n 1^n | n \ge 1\}^*$$

Exercise 5-2

Consider the following CFG:

$$S \rightarrow SAB \mid SBC \mid \varepsilon$$

$$A \rightarrow aAa \mid \varepsilon$$

$$B \rightarrow bB \mid \varepsilon$$

$$C \rightarrow cC \mid \varepsilon$$

a) Compute FIRST and FOLLOW sets for each non-terminal.

Solution:

Non-terminal	FIRST	FOLLOW
S	$\{\mathtt{a},\mathtt{b},\mathtt{c},arepsilon\}$	$\{\$, a, b, c\}$
A	$\{\mathtt{a},arepsilon\}$	$\{\$, a, b, c\}$
B	$\{\mathtt{b},arepsilon\}$	$\{\$, a, b, c\}$
C	$\{\mathtt{c},arepsilon\}$	$\{\$, a, b, c\}$

b) Build the parsing table.

Solution:

	Input Symbol				
Non-terminal	a	b	С	\$	
	SAB	SAB	SAB	SAB	
S	SBC	SBC	SBC	SBC	
	ε	ε	ε	ε	
A	a A a	ε	ε	ε	
A	ε				
В	ε	${\mathtt b} B$	ε	ε	
D		ε			
C	ε	ε	c C	ε	
			ε		

c) From the parsing table, show why the grammar is not LL(1).

Solution:

The grammar is not LL(1) because some entries of the parsing table contain more than one rule.

Exercise 5-3

Construct a parsing table for the following grammar. Is the grammar LL(1)?

$$S \rightarrow AB$$

$$A \rightarrow \operatorname{id} A \mid \operatorname{num}$$

$$B \rightarrow CA$$

$$C \rightarrow \mathbf{0}C \mid \mathbf{1}$$

Solution:

	Input Symbol					
Non-terminal	id	num	0	1	\$	
S	AB	AB				
A	id A	num				
B			CA	CA		
C			0 C	1		

The grammar is LL(1) because every position in the parsing table contains at most one rule.

Exercise 5-4

Consider the following CFG:

$$egin{array}{lll} S &
ightarrow & (L) \mid {\tt a} \\ L &
ightarrow & L, S \mid S \end{array}$$

a) Eliminate left recursion.

Solution:

b) Compute FIRST and FOLLOW sets for each non-terminal.

Solution:

$$\begin{aligned} & \text{FIRST}(S) = \{(\tt,a\} \\ & \text{FIRST}(L) = \{(\tt,a\} \\ & \text{FIRST}(L') = \{\tt,\varepsilon\} \\ & \text{FOLLOW}(S) = \{\tt,\tt,s\} \} \\ & \text{FOLLOW}(L) = \{\tt)\} \\ & \text{FOLLOW}(L') = \{\tt)\} \end{aligned}$$

c) Build the parsing table.

Solution:

	Input Symbol				
Non-terminal	a	,	()	\$
S	a		(<i>L</i>)		
L	aL'		(L)L'		
L'		, SL'		ε	

d) Trace the operations of a predictive top-down parser on the string (a,(a,a)).

3

Exercise 5-5

Consider the following CFG:

$$S \rightarrow SS + \mid SS * \mid$$
 a

a) Eliminate left recursion and left factor the grammar.

Solution:

$$\begin{array}{ccc} S & \rightarrow & \mathrm{a}S' \\ S' & \rightarrow & SS'' \mid \varepsilon \\ S'' & \rightarrow & +S' \mid *S' \end{array}$$

b) Compute first and follow sets for each non-terminal.

Solution:

$$\begin{split} &\operatorname{FIRST}(S) = \{\mathtt{a}\} \\ &\operatorname{FIRST}(S') = \operatorname{FIRST}(S) \cup \{\varepsilon\} = \{\mathtt{a}, \varepsilon\} \\ &\operatorname{FIRST}(S'') = \{+, *\} \\ &\operatorname{FOLLOW}(S) = \operatorname{FIRST}(S'') \cup \{\$\} = \{+, *, \$\} \\ &\operatorname{FOLLOW}(S') = \operatorname{FOLLOW}(S) = \{+, *, \$\} \\ &\operatorname{FOLLOW}(S'') = Follow(S') = \{+, *, \$\} \end{split}$$

c) Build the parsing table.

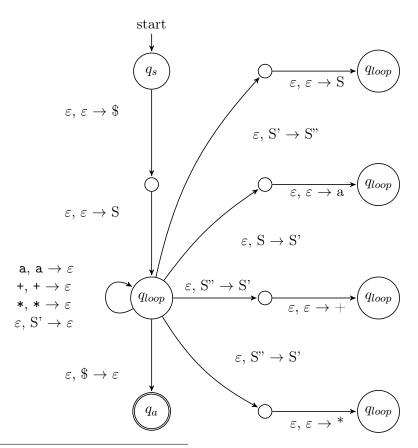
Solution:

	Input Symbol				
Non-terminal	a	+	*	\$	
S	aS' SS"				
S'	SS''	ε	ε	ε	
S''		+S'	*S'		

1

d) Draw the corresponding PDA for the above grammar.

Solution:



¹Note that all the states labelled q_{loop} are the same state. They are only drawn as separate states to avoid clutter.

e) Trace the operations of a predictive top-down parser on the string aa+.						