Miscellaneous Problems on Syntax

* 1. A language is said to be *context-free* just if it is expressible by a context free grammar. For each of the following statements about languages over the alphabet $\{0, 1\}$, determine if it is true or false.

(a) Every word of a context-free language is of finite length.

(b) The language $\emptyset$ is context free.

(c) The language of all even length strings is context free.

(d) Every finite subset of $\{0, 1\}^*$ is context free.

Solution

(a) True

(b) True

(c) True

(d) True

* 2. For each of the following statements regarding context-free grammars and context-free languages over some alphabet Σ, determine if it is true or false.

(a) In a context-free grammar, there is exactly one rule for each nonterminal.

(b) The language Σ^* is context free.

(c) No context-free language can include the empty word.

(d) In a context-free grammar, there are always more terminal symbols than non-terminal symbols.

Solution

(a) False

(b) True

(c) False

(d) False

* 3. Consider the following grammar, with start symbol S:

$$\begin{aligned} S &\longrightarrow 0\ T\ 0 \mid 1\ T\ 1 \\ T &\longrightarrow 0\ T \mid 1\ T \mid 0 \mid 1 \end{aligned}$$

For each of the following words, give a derivation to show that it is in the language of this grammar.

(a) 0110

(b) 00110

(c) 11001

Solution

(a) $S \to 0T0 \to 01T0 \to 0110$

(b) $S \to 0T0 \to 00T0 \to 001T0 \to 00110$

(c) $S \to 1T1 \to 11T1 \to 110T1 \to 11001$

* 4. For each of the following CFG over $\{a, b, c\}$, with start symbol S, give (I) one word that is in the language and (II) one word that is not.

Both words should be over the alphabet $\{a, b, c\}$. Label the two words with (I) and (II) so that it is clear which is claimed to be in and which is claimed to be not in.

(a)

$$\begin{array}{rcl} S & \longrightarrow & XXX \\ X & \longrightarrow & a \mid b \end{array}$$

(b)

$$\begin{array}{rcl} S & \longrightarrow & T\ S \mid \epsilon \\ T & \longrightarrow & A\ b\ A\ b\ c \\ A & \longrightarrow & a\ A \mid \epsilon \end{array}$$

(c)

$$\begin{array}{rcl} S & \longrightarrow & AC \mid BC \\ A & \longrightarrow & a \mid a\ A \\ B & \longrightarrow & b \mid b\ B \\ C & \longrightarrow & c \mid c\ C \end{array}$$

(d)

$$S \longrightarrow a\ S\ a \mid b\ S\ \mid c$$

Solution

Lots of answers are possible, for example:

(a) (I) aaa, (II) ϵ

(b) (I) bbc, (II) c

(c) (I) c, (II) ϵ

* 5. Consider the grammar for Lisp, given below with start symbol S.

$$\begin{array}{lcl} S & \longrightarrow & A \mid (\ E\ ) \\ E & \longrightarrow & S\ E \mid \epsilon \\ A & \longrightarrow & \textsf{id} \mid \textsf{num} \end{array}$$

This grammar is over the 4 terminal symbols:

( ) id num

(a) Compute the nullable, first and follow maps for this grammar.

(b) Construct the parse table for this grammar.

(c) Is this grammar LL(1)?

Solution

(a)

Nonterminal	Nullable(-)	First(-)	Follow(-)
S	no	(, id, num	(,), id, num
E	yes	(, id, num	)
A	no	id, num	(,), id, num

(b)

Nonterminal	(	)	id	num
S	$S \longrightarrow (E)$		$S \longrightarrow A$	$S \longrightarrow A$
E	$E \longrightarrow SE$	$E \longrightarrow \epsilon$	$E \longrightarrow SE$	$E \longrightarrow SE$
A			$A \longrightarrow$ id	$A \longrightarrow$ num

(c) Yes.

** 6. For each of the following languages over $\{0,1\}$, construct a CFG to express it.

(a) $\{uv^n \mid u \in \{0\}^*, v = 11, n \in \mathbb{N}\}$

(b) $\{w \mid w \text{ starts with } 1\}$

(c) $\{0u1v0 \mid u \text{ is } v \text{ reversed}\}$

Solution

(a)

$$\begin{array}{lcl} S & \longrightarrow & U\ V \\ U & \longrightarrow & 0\ U \mid \epsilon \\ V & \longrightarrow & 11\ V \mid \epsilon \end{array}$$

(b)

$$S \longrightarrow S\ 0 \mid S\ 1 \mid 1$$

(c)

$$\begin{array}{lcl} S & \longrightarrow & 0\ T\ 0 \\ T & \longrightarrow & 0\ T\ 0 \mid 1\ T\ 1 \mid 1 \end{array}$$

** 7. For each of the following languages over $\{a, b\}$, construct a CFG to express it.

(a) $\{w \mid \text{in } w \text{ every 'a' is followed immediately by a 'b'}\}$

(b) $\{w \mid \text{the number of occurrences of 'a' in } w \text{ is a multiple of 3}\}$

(c) $\{w \mid w \text{ does } \textit{not} \text{ contain substring "ab"}\}$

Solution

(a)

$$S \longrightarrow b\ S \mid a\ b\ S \mid \epsilon$$

(b)

$$\begin{array}{lcl} S & \longrightarrow & b\ S \mid a\ T \mid \epsilon \\ T & \longrightarrow & b\ T \mid a\ U \\ U & \longrightarrow & b\ U \mid a\ S \end{array}$$

(c) If a word does not contain the substring ab then it must consist of some number of b (possibly 0) followed by some number of a (possibly 0).

$$\begin{array}{lcl} S & \longrightarrow & B\ A \\ A & \longrightarrow & a\ A \mid \epsilon \\ B & \longrightarrow & b\ B \mid \epsilon \end{array}$$

** 8. Design CFG to express the following sets of strings over the alphabet of ASCII characters. Note: (a) you will find it convenient use some abbreviation (like $\cdots$) to help present the expressions compactly and (b) this would not be given as an exam question without specifying the shape of the strings in each part more precisely.

(a) Valid Bristol University usernames (two lowercase letters followed by five digits)

(b) Valid 24 hour clock times in format HH:MM

(c) Valid IPv4 addresses written in decimal

Solution

(a)

$$\begin{array}{rcl} S & \longrightarrow & LLDDDDD \\ L & \longrightarrow & a \mid b \mid \cdots \mid z \\ D & \longrightarrow & 0 \mid 1 \mid \cdots \mid 9 \end{array}$$

(b)

$$\begin{array}{rcl} S & \longrightarrow & H:M \\ H & \longrightarrow & 0D \mid 1D \mid 20 \mid 21 \mid 22 \mid 23 \\ D & \longrightarrow & 0 \mid 1 \mid \cdots \mid 9 \\ M & \longrightarrow & 0D \mid 1D \mid 2D \mid 3D \mid 4D \mid 5D \end{array}$$

(c)

$$\begin{array}{rcl} S & \longrightarrow & X.X.X.X \\ X & \longrightarrow & D \mid DD \mid 0DD \mid 1DD \mid 2ED \mid 25E \mid 255 \\ E & \longrightarrow & 0 \mid 1 \mid 2 \mid 3 \mid 4 \\ D & \longrightarrow & 0 \mid 1 \mid \cdots \mid 9 \end{array}$$

** 9. Construct a context-free grammar to recognise Haskell floating point literals, e.g. 2.99, 23.09e+34, 0.12e−200, 1.4e1.

A general description is as follows. A *decimal literal* is a non-empty sequence of digits (0–9). A *floating point literal* is either:

- a decimal literal followed by a decimal point followed by a decimal literal, optionally followed by an exponent
- or, a decimal literal followed by an exponent.

An exponent is the character e; optionally followed by the character $+$ or the character $-$; followed in all cases by a decimal literal.

Solution

$$\begin{array}{rcl} S & \longrightarrow & L.L \mid L.LE \mid LE \\ D & \longrightarrow & 0 \mid 1 \mid \cdots \mid 9 \\ L & \longrightarrow & DL \mid D \\ E & \longrightarrow & eL \mid e+L \mid e-L \end{array}$$

** 10. For each of the following, give an equivalent grammar which is LL(1).

(a)

$$\begin{array}{rcl} S & \longrightarrow & S \wedge S \mid G \Rightarrow \textsf{prop} \\ G & \longrightarrow & G \wedge G \mid \textsf{prop} \end{array}$$

(b)

$$S \longrightarrow \textsf{int} \mid \textsf{string} \mid S \Rightarrow S \mid S \times S \mid (\ S\ )$$

Solution

(a) After dealing with left recursion in S and left factoring the result (but leaving G as is) we get a grammar like:

$$\begin{array}{rcl} S & \longrightarrow & D\ T \\ T & \longrightarrow & \wedge\ D\ T \mid \epsilon \\ D & \longrightarrow & G \Rightarrow \textsf{prop} \\ G & \longrightarrow & G \wedge G \mid \textsf{prop} \end{array}$$

Next, we need to deal with left recursion and then left factor the in G part:

$$\begin{array}{rcl} S & \longrightarrow & D\ T \\ T & \longrightarrow & \wedge\ D\ T \mid \epsilon \\ D & \longrightarrow & G \Rightarrow \textsf{prop} \\ G & \longrightarrow & \textsf{prop}\ H \\ H & \longrightarrow & \wedge\ \textsf{prop}\ H \mid \epsilon \end{array}$$

(b) First we factor out the base types int and string and the parenthesized form into a new nonterminal for clarity and then remove left recursion and then left factor - S derives sequences of A separated by ⇒ and ×.

$$\begin{array}{rcl} S & \longrightarrow & A\ T \\ T & \longrightarrow & \Rightarrow\ A\ T \mid \times\ A\ T \mid \epsilon \\ A & \longrightarrow & \textsf{int} \mid \textsf{string} \mid (\ S\ ) \end{array}$$

** 11. Consider the following grammar, with start symbol *DeclList*:

$$\begin{array}{rcl} \mathit{DeclList} & \longrightarrow & \mathit{DeclList}\ ;\ \mathit{Decl} \mid \epsilon \\ \mathit{Decl} & \longrightarrow & \mathit{IdList}\ :\ \mathit{Type} \\ \mathit{IdList} & \longrightarrow & \mathit{IdList}\ ,\ \textsf{id} \mid \textsf{id} \\ \mathit{Type} & \longrightarrow & \textsf{ty} \mid \mathit{Type}\ \textsf{tymod} \end{array}$$

This grammar is over the six terminal symbols:

; : , id ty tymod

(a) Give an equivalent grammar which is LL(1).

(b) Demonstrate that your grammar is LL(1) by constructing the parse table.

Solution

(a)

$$
\begin{array}{rrcl}
(1) & \mathit{DeclList} & \longrightarrow & \mathit{Decl}\ ;\ \mathit{DeclList} \\
(2) & & | & \epsilon \\
(3) & \mathit{Decl} & \longrightarrow & \mathit{IdList}\ :\ \mathit{Type} \\
(4) & \mathit{IdList} & \longrightarrow & \mathsf{id}\ \mathit{IdListRest} \\
(5) & \mathit{IdListRest} & \longrightarrow & ,\ \mathsf{id}\ \mathit{IdListRest} \\
(6) & & | & \epsilon \\
(7) & \mathit{Type} & \longrightarrow & \mathsf{ty}\ \mathit{TypeRest} \\
(8) & \mathit{TypeRest} & \longrightarrow & \mathsf{tymod}\ \mathit{TypeRest} \\
(9) & & | & \epsilon
\end{array}
$$

(b) The nullable, first and follow maps:

Nonterminal	Nullable(-)	First(-)	Follow(-)
DeclList	✓	id	
Decl		id	;
IdList		id	:
IdListRest	✓	,	:
Type		ty	;
TypeRest	✓	tymod	;

The parse table is:

Nonterminal	;	:	id	,	ty	tymod
DeclList			1			
Decl			3			
IdList			4			
IdListRest		6		5		
Type					7	
TypeRest	9					8

*** 12. Construct a CFG expressing the language of bit strings (strings over $\{0,1\}$) that represent numbers written in binary that are divisible by three. For example, 10010 should be derivable because it represents the decimal number 18 written in binary and this number is divisible by 3. However, 101 should not be derivable, because this is the binary representation of the number 5, which is not divisible by 3.

Solution

$$
\begin{array}{rcl}
S & \longrightarrow & 0A \mid 1B \\
A & \longrightarrow & 0A \mid 1B \mid \epsilon \\
B & \longrightarrow & 0C \mid 1A \\
C & \longrightarrow & 0B \mid 1C
\end{array}
$$

The idea of this grammar is as follows. Imagine a derivation starting from S. Each sentential form in the derivation, except the first and the last, has shape uX, for some non-empty string u over $\{0,1\}$ and some non-terminal $X \in \{A,B,C\}$. The non-terminal expresses exactly whether the string u is a bitstring which has remainder 0 (A), 1 (B) or 2 (C) after dividing by 3. For example:

$$S \to 1B \to 11A \to 111B$$

The bitstring 1 indeed has remainder 1 when divided by 3, corresponding to nonterminal B. The bitstring 11, representing the number 3 in binary, has remainder 0 when divided by 3, corresponding to nonterminal A. The bitstring 111, representing the number 7, has remainder 1 when divided by 3, corresponding to nonterminal B.

The grammar is designed with this scheme in mind. For example, when sentential form is of some shape uB, this means that the word u has remainder 1 after dividing by 3. Therefore, if we extend the word by adding a 0 on the end, then we know, by simple modular arithmetic, that the remainder will now be $2 * 1 + 0 = 2$. Hence, we can replace B by $0C$ - we extend the word with a 0 and record that the word $u0$, whatever it is, must now be remainder 2 after dividing it by 3. Similarly, if from uB we choose to extend the word with a 1, then the new remainder will be $2 * 1 + 1 = 3$, i.e. remainder 0. Therefore, the other option when replacing B is to replace by $0A$, i.e. to create the sentential form $u0C$, which correctly records that it is a word with remainder 0 after dividing by 3. Since we only want to derive words that are divisible by 3, i.e. that have remainder 0, we only allow the removal of that single nonterminal X when the remainder is 0, i.e. when $X = A$.