

Computer Science 414
Spring 2010

Mitderm 1 Practice Problems

We will go over solutions on Wednesday, March 10th

- Describe the difference between the languages described by the following 3 regular expressions:
 - a^*b^*
 - ab^*
 - $(ab)^*$
- Give a regular expression for C comments. In C, anything between `/*` and `*/` is a comment. In C, comments do not nest. So, `/* this is a comment */` and `/* This is also /* a legal C comment */` are both legal C comments, while `/* this is /* not */ a legal C comment */` and `/* this is also */ not a legal C comment */` are not legal C comments. (*HINT* – this is harder than it looks. Feel free to use the `[^c]` form, which matches any character except a c. Use escape character for characters with meaning – `*` for `*`, and so on.)
- Give a Context-Free Grammar for each of the following languages:
 - The set of all strings over $\{(,), [,]\}$ which form balanced parenthesis. That is, $()$, $()()$, $((()())())$, $[(())]$, and $(([]))$ are all in the language, but $)()$, $(()$ and $[]$ are not in the language.
 - The set of all strings over $\{\text{num}, +, -, *, /\}$ which are legal binary post-fix expressions. Thus num , $\text{num num} +$, $\text{num num num} * -$, and $\text{num num} - \text{num} *$ are all in the language, while $\text{num} *$, $\text{num} * \text{num}$, and $\text{num num num} -$ are not in the language. Is your CFG ambiguous?
- For the following CFGs, compute First and Follow sets for each non-terminal, and then create an LL(1) parse table
 - | | | |
|---------------|---|---|
| Terminals | = | {for, to, id, :=, num, print} |
| Non-Terminals | = | { S, E } |
| Rules | = | (1) $S \rightarrow \text{for id := } E \text{ to } E S$
(2) $S \rightarrow \text{print } E$
(3) $E \rightarrow \text{id}$
(4) $E \rightarrow \text{num}$ |
| Start Symbol | = | S |
 - | | | |
|---------------|---|---|
| Terminals | = | {a, b, c, \$} |
| Non-Terminals | = | { S, A, B } |
| Rules | = | (1) $S \rightarrow ABC\$$
(2) $A \rightarrow aA$
(3) $A \rightarrow \epsilon$
(4) $B \rightarrow bB$
(5) $B \rightarrow \epsilon$ |
| Start Symbol | = | S |

5. Given the following CFG and LR parse table

- Terminals = {num, *, +, (,), \$}
 Non-Terminals = { E' , E , T , F }
 Rules = (0) $E' \rightarrow E\$$
 (1) $E \rightarrow E + T$
 (2) $E \rightarrow T$
 (3) $T \rightarrow T * F$
 (4) $T \rightarrow F$
 (5) $F \rightarrow \text{num}$
 (4) $F \rightarrow (E)$
 Start Symbol = E'

	num	*	+	()	\$	E	T	F
1	s5			s6			g2	g3	g4
2		s8	s7			accept			
3		s8	r(2)		r(2)	r(2)			
4		r(4)	r(4)		r(4)	r(4)			
5		r(5)	r(5)		r(5)	r(5)			
6				s6			g11	g13	g4
7	s5			s6				g10	g4
8	s5			s6					g9
9		r(3)	r(3)		r(3)	r(3)			
10		s8	r(1)		r(1)	r(1)			
11			s7		s12				
12		r(6)	r(6)		r(6)	r(6)			

Which of the following strings are successfully parsed? If a string is not successfully parsed, give the contents of the stack (both states and symbols) when parsing fails.

- (a) num + num * num - num \$
- (b) num * num + + num \$
- (c) num + num * * num \$
- (d) num * (num + num) num \$
- (e) num + ((num * num)) \$

6. Create a set of LR(0) items and transitions for the following grammar. Show that the grammar is not LR(0). Give the SLR(1) parse table for the grammar.

- Terminals = {num, id, ., \$}
 Non-Terminals = { E' , E , V }
 Rules = (0) $E' \rightarrow E\$$
 (1) $E \rightarrow V$
 (2) $E \rightarrow \text{num}$
 (3) $V \rightarrow \text{id}$
 (4) $V \rightarrow V . \text{id}$
 Start Symbol = E'