CMSC 330 Exam 1: March 1, 2000

Show all work necessary to justify your answers! All questions will be given equal weight.

1. Given the following grammar G, with start symbol S

$$S \rightarrow aB \mid bA$$

$$A \rightarrow a \mid aS \mid bAA$$

$$B \rightarrow b \mid bS \mid aBB$$

For the string $aababb$, show

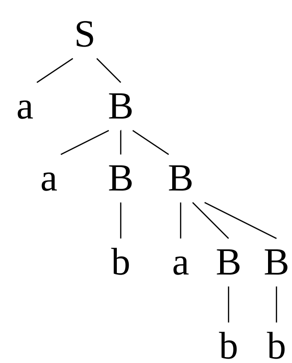
(a) a leftmost derivation,

$$S \implies aB \implies aaBB \implies aabB \implies aabaBB \implies aababB \implies aababb$$

(b) a rightmost derivation,

$$S \implies aB \implies aaBB \implies aaBaBB \implies aaBaBb \implies aaBabb \implies aababb$$

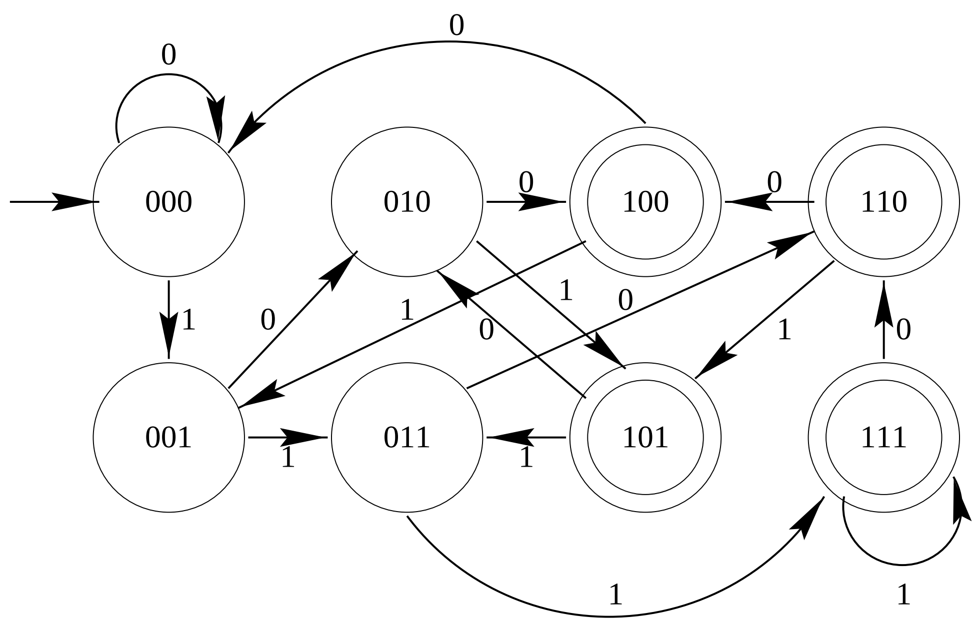
(c) a parse tree.



2. Produce a deterministic finite automaton (DFA) that recognizes the language:
L $= \{\ w \mid w \in \{0,1\}^*$ and the third to last symbol of w is a 1 $\}$

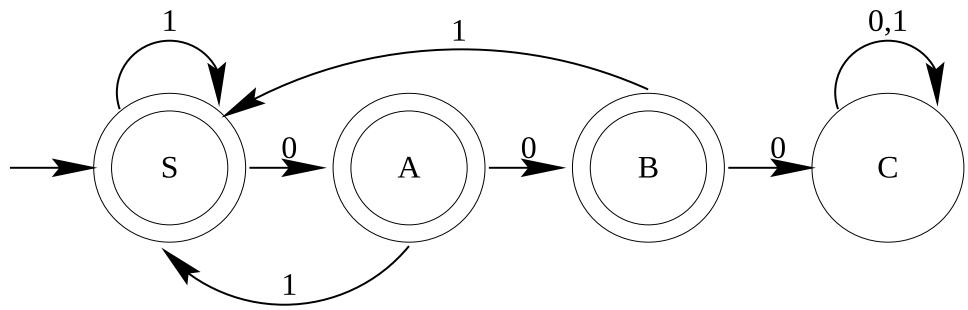
Note that L contains no strings of length less than 3.

The state labels correspond to the last 3 symbols read from the input string, with the first digit being the third to last symbol, the second digit the second to last, and the third digit the last input symbol read.



3. Give an unambiguous grammar for the language:
L $= \{\ w \mid w \in \{0,1\}^*$ and w doesn't have three consecutive 0's $\}$

From this DFA:



S' is the start symbol.

$$S' \longrightarrow S \mid \epsilon$$

$$\begin{aligned} S &\longrightarrow 0A \mid 0 \mid 1S \mid 1 \\ A &\longrightarrow 0B \mid 0 \mid 1S \mid 1 \\ B &\longrightarrow 0C \mid 1S \mid 1 \\ C &\longrightarrow 0C \mid 1C \end{aligned}$$

And, since the productions from the non-terminal C can never produce a string of only terminals, all productions with C in them can be omitted, leaving the grammar:

$$S' \longrightarrow S \mid \epsilon$$

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

4. Show a regular expression that generates all strings in the language:
L $= \{\ w \mid w \in \{0,1\}^*$ and neither 00 nor 11 is a substring of $w\ \}$.

Justify that your regular expression is correct.

$$(1 \mid \epsilon)(01)^*\ (0 \mid \epsilon)$$

or,

$$(0 \mid \epsilon)(10)^*\ (1 \mid \epsilon)$$

Either r.e. allows a string to start or end with either a 0 or 1, and does not allow a pair of consecutive 0's or a pair of consecutive 1's.

5. Write an unambiguous context free grammar for Fortran arithmetic expressions with operands represented by <id> and operators $+, -, *, /, **$. All operators are infix binary operators, and $+$ and $-$ are also prefix unary operators . Unary operators have higher precedence than binary operators, and for the binary operators $**$ has highest precedence, $*$ and $/$ have next highest precedence, and $+$ and $-$ have the lowest precedence. For the binary operators, $+, -, *, /$ have left associativity and $**$ has right associativity. Parentheses are used to override precedence and associativity (i.e. if E is an arithmetic expression, then so is (E)).

   Justify that your grammar is unambiguous.

   The start symbol is E.

$$E \longrightarrow E + T \mid E - T \mid T$$

$$T \longrightarrow T * P \mid T/P \mid P$$

$$P \longrightarrow Q * *P \mid Q$$

$$Q \longrightarrow +Q \mid -Q \mid R$$

$$R \longrightarrow < id > \mid (E)$$

   The grammar is unambiguous because it enforces precedence and associativity of the operators properly. This means that there is guaranteed to be only one parse tree (equivalently, only one leftmost derivation) for any legal Fortran arithmetic expression.