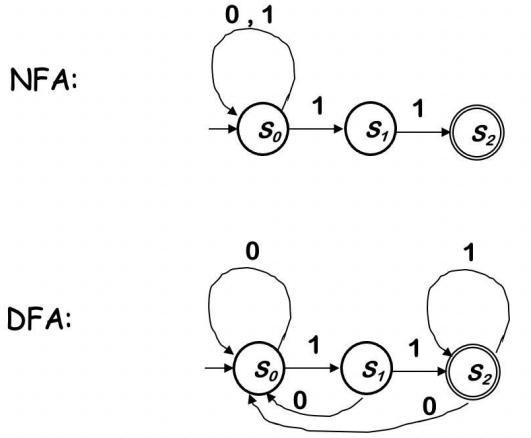
CS314 Spring 2023

Homework 2

Due Friday, February 11, 11:59pm
SAMPLE SOLUTION

1 Problem — Finite State Automaton (FSA)

1. Specify the state transition graph of (1) a NFA (which is not a DFA as well) without ϵ transitions and (2) a DFA that recognizes the following language: "All strings of 0's and 1's that end with 11"



2. In addition to the state transition graphs (diagram), give the state transition table and

the formal specifiction of an automaton as the quadrupel <S, s, F, T> for both, your NFA and DFA. Do not include "error" states.

State transition tables

NFA:

	0	1
s_0	s_0	s_0, s_1
s_1		s_2
s_2		

DFA:

	0	1
s_0	s_0	s_1
s_1	s_0	s_2
s_2	s_0	s_2

Formal specicification

NFA:

S = $\{s_0, s_1, s_2\}$

s = s_0

F = $\{s_2\}$

T: $S \times \Sigma \to S$

T(s_0, 0) = s_0 , T(s_0, 1) = s_0 , T(s_0, 1) = s_1 , T(s_1, 1) = s_2

DFA:

S = $\{s_0, s_1, s_2\}$

s = s_0

F = $\{s_2\}$

T: $S \times \Sigma \to S$

T(s_0, 0) = s_0 , T(s_0, 1) = s_1 , T(s_1, 0) = s_0 ,

T(s_1, 1) = s_2 , T(s_2, 0) = s_0 , T(s_2, 1) = s_2

2 Problem — Regular and Context-Free Languages

Are the following languages context-free or not? If yes, specify a context-free grammar in BNF notation that generates the language. If not, give an informal argument.

All of these languages are context free. Sample sets of rules are given, but other rules may also work.

1. { $a^n b^m c^o$ | m > 0, n $\geq$ 0, o > 0} , with alphabet Σ = {a, b, c}

   $< S >::= < A >< B >< C >$

   $< A >::= a < A > \mid \epsilon$

   $< B >::= b < B > \mid b$

   $< C >::= c < C > \mid c$

   This is a regular language specified by regular expression

   $a^* b^+ c^+$

2. { $a^n b^n c^n$ | n > 0} , with alphabet Σ = {a, b, c}

   This is not a context-free language.

   Informal argument: In order to make sure that there are equal numbers of a's and b's, the PDA first pushes all a's, and then pops one a for each encountered b. At the end of this matching process, the stack is empty. Therefore, there is no way to ensure that there will be the same number of c's as there are a's and b's.

3. { $0^{2n} 1^{4n}$ | n $\geq$ 0 }, with alphabet Σ = {0, 1}

   $< S >::= 00 < S > bbbb \mid \epsilon$

4. { wcw^R | w $\in \Sigma^*$ and w^R is w in reverse }, with alphabet Σ ={a, b, c}

   $< S >::= a < S > a \mid b < S > b \mid$ c

5. { $a^n b^m c^m d^n$ | n $\geq$ 0, m $\geq$ 0 }, with alphabet Σ = {a, b, c, d}

   $< S >::= a < S > d \mid < A > \mid \epsilon$

   $< A >::= b < A > c \mid \epsilon$

6. { $a^n b^m c^n d^m$ | n $\geq$ 0, m $\geq$ 0 }, with alphabet Σ = {a, b, c, d}

   This is not a context-free language.

   Informal argument: In order to make sure that there are equal numbers of a's and c's, the PDA first pushes all a's, and then pops one a for each encountered c. To do so, it will need to skip over all b's, making it impossible to match b's and d's.

7. { $a^n a^n b^n b^n$ | n $\geq$ 0 }, with alphabet Σ = {a, b}

   $< S >::= aa < S > bb \mid \epsilon$

8. { w | w has more than 3 symbols}, with alphabet Σ={a, b}

   $< S >::= < A >< A >< A >< A >< B >$

   $< A >::= a \mid b$

   $< B >::= a < B > \mid b < B > \mid \epsilon$

This is a regular language specified by regular expression
$(a|b)(a|b)(a|b)(a|b)(a|b)^*$

3 Problem — Derivation, Parse Tree, Ambiguity, Precedence & Associativity

A language that is a subset of the language of propositional logic may be defined as follows:

<start> ::= <expr>
<expr> :: = <expr> ∨ <expr> |
<expr> ∧ <expr> |
<expr> ↔ <expr> |
<const> | <var>
<const> :: = true | false
<var> :: = a | b | c

1. Give a leftmost and a rightmost derivation for the sentence
   a ∧ true ∧ b ↔ false ∨ true .

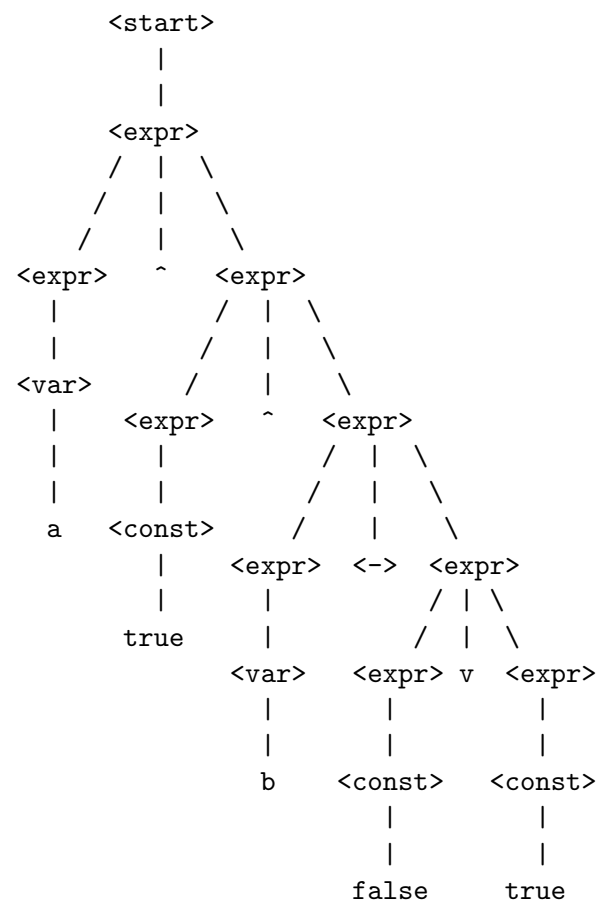
   There are several possible leftmost and rightmost derivations since the grammar is ambiguous (see part 4).

   <start>
   $\Rightarrow_{LM}$ <expr>
   $\Rightarrow_{LM}$ <expr> ∧ <expr>
   $\Rightarrow_{LM}$ <var> ∧ <expr>
   $\Rightarrow_{LM}$ a ∧ <expr>
   $\Rightarrow_{LM}$ a ∧ <expr> ∧ <expr>
   $\Rightarrow_{LM}$ a ∧ <const> ∧ <expr>
   $\Rightarrow_{LM}$ a ∧ true ∧ <expr>
   $\Rightarrow_{LM}$ a ∧ true ∧ <expr> ↔ <expr>
   $\Rightarrow_{LM}$ a ∧ true ∧ <var> ↔ <expr>
   $\Rightarrow_{LM}$ a ∧ true ∧ b ↔ <expr>
   $\Rightarrow_{LM}$ a ∧ truee ∧ b ↔ <expr> ∨ <expr>
   $\Rightarrow_{LM}$ a ∧ true ∧ b ↔ <const> ∨ <expr>
   $\Rightarrow_{LM}$ a ∧ true ∧ b ↔ false ∨ <expr>
   $\Rightarrow_{LM}$ a ∧ true ∧ b ↔ false ∨ <const>
   $\Rightarrow_{LM}$ a ∧ true ∧ b ↔ false ∨ true

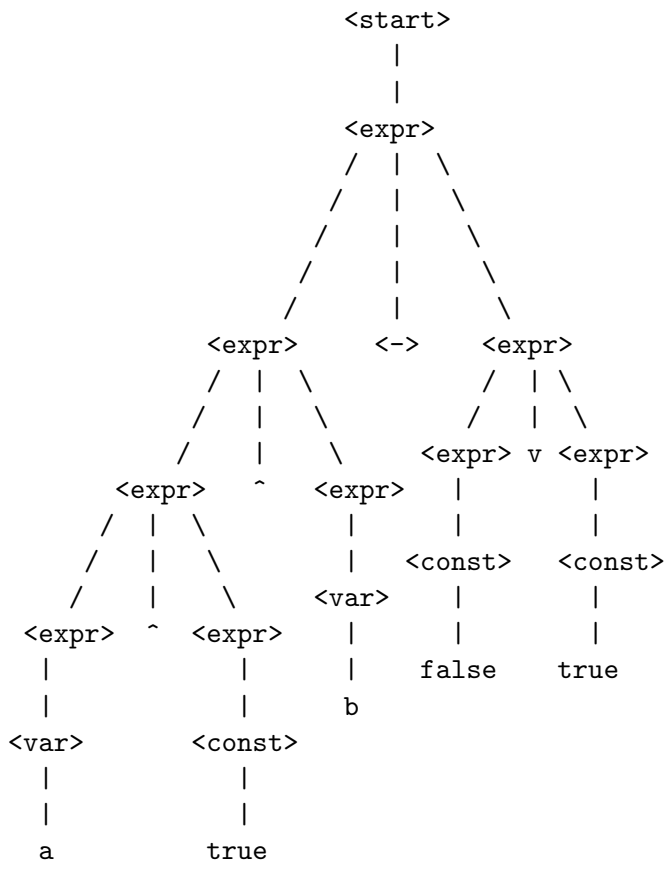
<start>
$\Rightarrow_{RM}$ <expr>
$\Rightarrow_{RM}$ <expr> $\leftrightarrow$ <expr>
$\Rightarrow_{RM}$ <expr> $\leftrightarrow$ <expr> $\vee$ <expr>
$\Rightarrow_{RM}$ <expr> $\leftrightarrow$ <expr> $\vee$ <const>
$\Rightarrow_{RM}$ <expr> $\leftrightarrow$ <expr> $\vee$ true
$\Rightarrow_{RM}$ <expr> $\leftrightarrow$ <const> $\vee$ true
$\Rightarrow_{RM}$ <expr> $\leftrightarrow$ false $\vee$ true
$\Rightarrow_{RM}$ <expr> $\wedge$ <expr> $\leftrightarrow$ false $\vee$ true
$\Rightarrow_{RM}$ <expr> $\wedge$ <var> $\leftrightarrow$ false $\vee$ true
$\Rightarrow_{RM}$ <expr> $\wedge$ b $\leftrightarrow$ false $\vee$ true
$\Rightarrow_{RM}$ <expr> $\wedge$ <expr> $\wedge$ b $\leftrightarrow$ false $\vee$ true
$\Rightarrow_{RM}$ <expr> $\wedge$ <const> $\wedge$ b $\leftrightarrow$ false $\vee$ true
$\Rightarrow_{RM}$ <expr> $\wedge$ true $\wedge$ b $\leftrightarrow$ false $\vee$ true
$\Rightarrow_{RM}$ <var> $\wedge$ true $\wedge$ b $\leftrightarrow$ false $\vee$ true
$\Rightarrow_{RM}$ a $\wedge$ true $\wedge$ b $\leftrightarrow$ false $\vee$ true

2. Show the corresponding parse trees for the derivations

Parse tree for leftmost derivation given in previous answer:

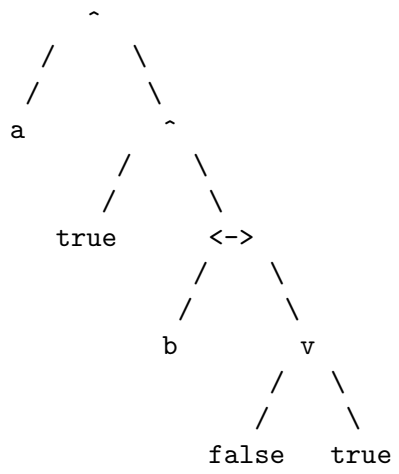


Parse tree for rightmost derivation given in previous answer:

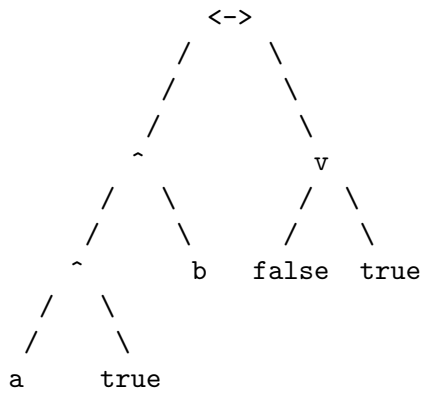


3. Give the corresponding abstract syntax tree (AST).

   AST for leftmost derivation given in previous answer:



   AST for rightmost derivation given in previous answer:



4. Show that the above grammar is ambiguous.

   Since, as shown above, the sentence "a $\wedge$ true $\wedge$ b $\leftrightarrow$ false $\vee$ true" has multiple possible parse trees, the grammar is ambiguous. Many other sentences with their distinct parse trees will also demonstrate this.

5. Give an unambiguous grammar for the same language that enforces the following precedence and associativity:

   - $\wedge$ has highest precedence (binds strongest), followed by $\vee$, and then $\leftrightarrow$
   - $\wedge$ is left associative, and $\leftrightarrow$ and $\vee$ are right associative

   $< start >::=< expr >$

   $< expr >::=< orexpr > \leftrightarrow < expr > \mid < orexpr >$

   $< orexpr >::=< andexpr > \vee < orexpr > \mid < andexpr >$

   $< andexpr >::=< andexpr > \wedge < term > \mid < term >$

   $< term >::=< const > \mid < var >$

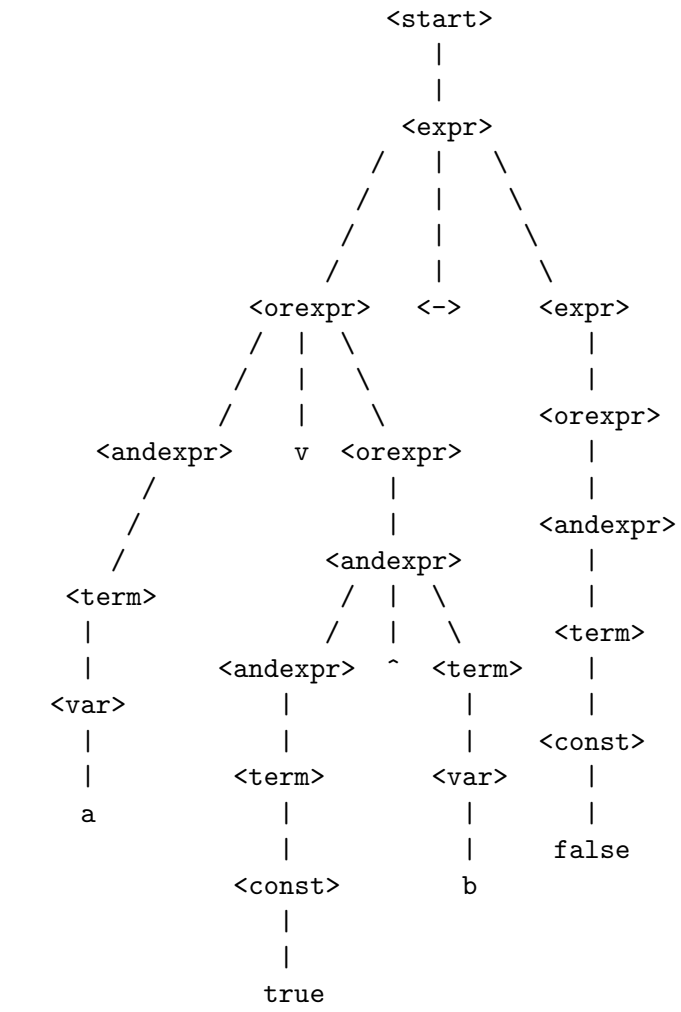
   $< const >::= true \mid false$

   $< var >::= a \mid b \mid c$

6. Give the parse tree and AST for your new, unambiguous grammar for the sentence

   $\mathsf{a} \vee \mathsf{true} \wedge \mathsf{b} \leftrightarrow \mathsf{false}$.

Parse tree:



AST: