CS 314
Assignment 3

1 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, try to give an informal argument (proving a language is not context-free requires Pumping Lemma, which is not covered in our class, but if you can learn Pumping Lemma by yourself and prove it correctly, it will be extra-credit).

(a) $\{ a^m c^o b^m \mid$ m > 0, o > 0$\}$, with alphabet $\Sigma = \{$a, b, c$\}$

(b) $\{ a^n b^{3n} \mid$ n ≥ 0 $\}$, with alphabet $\Sigma = \{$a, b$\}$

(c) $\{ ww^R \mid$ w $\in \Sigma^*$ and w^R is w in reverse $\}$, with alphabet $\Sigma = \{$a, b$\}$

(d) $\{$ w | w has no more than 3 symbols$\}$, with alphabet Σ=$\{$a, b$\}$

a) <token1> ::= a <token 2> b | ε
<token2> ::= c | ε

b) <token1> ::= a <token1> bbb | ε

c) <w> ::= a <w> a | b <w> b | ε

d) <token1> ::= aaa|aab|aac|aba|abb|abc|aca|acb|acc|
baa|bab|bac|bba|bbb|bbc|bca|bcb|bcc|
caa|cab|cac|cba|cbb|cbc|cca|ccb|ccc

2 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> **OR** <expr> |
<expr> **AND** <expr> |
<expr> → <expr> |
<const> | <var>

<const> :: = TRUE | FALSE

<var> :: = a | b | c | . . . | z

(a) Give a leftmost and a rightmost derivation for the sentence

a **OR** FALSE **AND** b → FALSE .

(b) Give the corresponding parse trees for the derivations.

(c) Show that the above grammar is ambiguous.

(d) Give an unambiguous grammar for the same language that enforces the following precedence and associativity:

- **AND** has highest precedence (binds strongest), followed by **OR**, and then →
- **AND** and **OR** are left associative, and → is right associative

(e) Give the parse tree for your new, unambiguous grammar for the sentence

a **OR** TRUE **AND** b → true **OR** FALSE .

a) Leftmost Derivation

Rule	Sentence
	<start>
1	<expr>
2	<expr> OR <expr>
2	<var> OR <expr>
4	a OR <expr>
2	a OR <expr> AND <expr>
2	a OR <const> AND <expr>
3	a OR FALSE AND <expr>
2	a OR FALSE AND <expr> → <expr>
2	a OR FALSE AND <var> → <expr>
4	a OR FALSE AND b → <expr>
2	a OR FALSE AND b → <const>
3	a OR FALSE AND b → FALSE

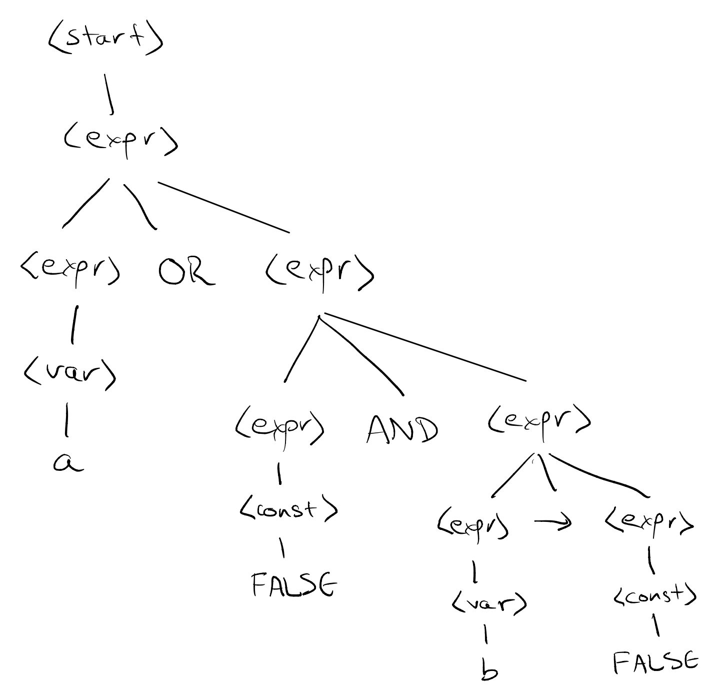
Rules

1) <start> ::= <expr>
2) <expr> :: = <expr> **OR** <expr> |
<expr> **AND** <expr> |
<expr> → <expr> |
<const> | <var>
3) <const> :: = TRUE | FALSE
4) <var> :: = a | b | c | . . . | z

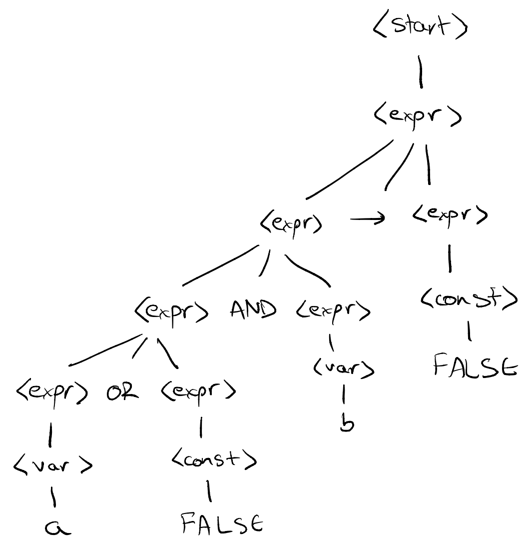
Rightmost Derivation

Rule	Sentence
	<start>
1	<expr>
2	<expr> → <expr>
2	<expr> → <const>
3	<expr> → FALSE
2	<expr> AND <expr> → FALSE
2	<expr> AND <var> → FALSE
4	<expr> AND b → FALSE
2	<expr> OR <expr> AND b → FALSE
2	<expr> OR <const> AND b → FALSE
3	<expr> OR FALSE AND b → FALSE
2	<var> OR FALSE AND b → FALSE
4	a OR FALSE AND b → FALSE

b) Leftmost Parse Tree



Rightmost Parse Tree



c) There are multiple valid parse trees for the same expression (as shown in part b) so this language is ambiguous.

d)

OLD Ambiguous Grammar

<start> ::= <expr>
<expr> :: = <expr> **OR** <expr> |
<expr> **AND** <expr> |
<expr> → <expr> |
<const> | <var>
<const> :: = TRUE | FALSE
<var> :: = a | b | c | . . . | z

New Unambiguous Grammar

<start> ::= <expr>
<expr> ::= <AO> → <expr> | <AO>
<AO> ::= <AO> OR <A> | <A>
<A> ::= <A> AND <CV> | <CV>
<CV> ::= <const> | <var>
<const> ::= TRUE | FALSE
<var> ::= a | b | ... | z

e) Parse Tree For:
a **OR** TRUE **AND** b → true **OR** FALSE