Attach a fully filled-in cover sheet to the front of your printed homework. Your name should not appear anywhere except on the cover sheet; each individual page of the homework should include your banner ID instead.

While collaboration is encouraged in this class, please remember not to take away notes from any collaboration sessions. Also please list the names and logins of any of your collaborators at the beginning of each homework.

Please monitor Piazza, as we will post clarifications of questions there. You should hand in your solutions by 12:55 to the CSCI 51 bin on the second floor of the CIT. Late homeworks are not accepted.

The following is a warmup question, which will not be graded:

For each of the languages below, give a CFG that generates the language.

a) \{xy \mid x, y \text{ are palindromes over the alphabet } \Sigma = \{0, 1\}\}

For the purposes of this problem, \(\varepsilon\) is a palindrome.

b) \{01^{n+1}01^n \mid n, m \geq 0\}

c) \{a^i b^j c^k \mid i = j \text{ or } j = k \text{ where } i, j, k \geq 0\}

The following questions are lab problems.

Lab Problem 1

Let \(\Sigma = \{(,)\}\). A string \(w \in \Sigma^*\) is called matching if the parentheses are paired up correctly. Formally, we define the language \(L\) of matching strings recursively by:

\[
\begin{align*}
\epsilon & \in L; \\
\text{if } w \in L, \text{ then } (w) & \in L; \\
\text{if } v \in L \text{ and } w \in L, \text{ then } vw & \in L.
\end{align*}
\]
For example, 

“()”, “((()))”, and “(((())())”) are matching while “(()”, “)”, and “(())) (“ are not.

a. Prove that $L$ is not regular.

b. The following CFG $G$ recognizes the language $L$:

$$A \rightarrow AA \mid (A) \mid \varepsilon$$

Show that $G$ is ambiguous, i.e. that the same string can yield two distinct parse trees.

c. Here is another CFG $H$ that is ambiguous:

$$S \rightarrow S+S \mid S\cdot S \mid z$$

Give an unambiguous grammar $H'$ such that $L(H) = L(H')$.

d. Sketch a proof that $H'$ is unambiguous.

**Lab Problem 2**

In this problem, we will show that the language $L = \{a^ib^jc^k \mid i = j \text{ or } j = k\}$ is inherently ambiguous. Although we don’t use the pumping lemma for context-free languages directly, its proof is helpful for doing this problem.

a. Show that $L$ is context-free.

b. Let $G$ be any context-free grammar for $L$, and let $n$ be the number of variables in $G$, and let $d$ be the maximum number of symbols on the right hand side of a rule in $G$. Let $p = d^{n+1}$. Let $\tau$ be a minimal parse tree for the string $s = a^pb^pc^m$ for $m = p! + p$. Argue that there is a leaf at position $i$ in the string, $1 \leq i \leq p$ such that the depth of this leaf is at least $n+1$. That is, there is a leaf labeled ‘a’ with a path of length at least $n+1$ to the root.

c. Argue that it follows from part (b) that there must exist a variable $A$ in $G$ such that $A \Rightarrow^* vAy$ for $v = a^j, y = b^j$ for some $j > 0$, and that this variable $A$ must appear somewhere in $\tau$. (Recall the $x \Rightarrow^* y$ notation means that either $x = y$ or there is a sequence $x_1, \ldots, x_k$ such that $x \Rightarrow x_1 \Rightarrow x_2 \ldots \Rightarrow x_k \Rightarrow y$.)
d. Using $\tau$ and the fact that it can be “pumped” using $A \Rightarrow^* vAy$, prove that there exists a parse tree $\sigma$ for the string $a^m b^m c^m$ that contains the variable $A$.

e. Let $\tau'$ be a minimal parse tree for the string $s' = a^m b^p c^p$. Analogously to parts (b), (c) and (d), show that there must exist a variable $C$ in $G$ such that $C \Rightarrow^* uCw$ for $u = b^\ell$, $w = c^\ell$ for some $\ell > 0$, and that this variable $C$ must appear somewhere in $\tau'$. Using $\tau'$ and the fact that it can be “pumped” using $C \Rightarrow^* uCw$, prove that there exists a parse tree $\sigma'$ for the string $a^m b^m c^m$ that contains the variable $C$.

f. Show that $\sigma$ and $\sigma'$ are distinct parse trees for the same string. Hint: Can $\tau$ contain $C$?

**Lab Problem 3**

Consider the alphabet $\Sigma = \{\#, 0, 1, [\ 0 \ 0], [\ 0 \ 1], [\ 1 \ 0], [\ 1 \ 1]\}$.

Let $x, y$ be integers with binary representations $x_n \ldots x_0$ and $y_n \ldots y_0$ respectively. $n$ is chosen such that either $x_n = 1$ or $y_n = 1$, i.e. at least one of $x$ and $y$ has no trailing 0’s.

We use the notation $\begin{bmatrix} x \\ y \end{bmatrix}$ to denote the string $\begin{bmatrix} x_0 \\ y_0 \\ \vdots \\ x_n \\ y_n \end{bmatrix}$.

Now consider the language:

$$L = \left\{ s_1 \# s_2 \; | \; s_1 = \begin{bmatrix} x \\ y \end{bmatrix}, s_2 \text{ is the binary representation of } x + y \text{ using the least number of bits} \right\}$$

In this language, the numbers represented by $s_1 = \begin{bmatrix} x \\ y \end{bmatrix}$ have their least significant bits first, and the number $s_2$ has the most significant bits first.

To help understand the definition of the language, consider the following examples:

- $\# \in L$
- $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \#1000 \in L$ because $101 + 011 = 1000$
CSCI 0510 Due: Oct 8, 2015

• $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ #100 $\notin L$ because $x$ and $y$ both begin with extraneous 0s.

• $\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ #01000 $\notin L$ because $s_2$ begins with extraneous 0s.

a. Prove that $L$ is not regular.

b. Design a CFG for $L$. 