CSE 105 Homework 2 Solutions

Problem 1. Show that $L = \{x_1\#x_2\#\ldots\#x_k \mid k \geq 1$, each $x_i \in \{a.b\}^\star$, and for some i and j, $x_i = x_j^r\}$ is context-free.

Solution: In order to prove L is a context-free language, we will provide a context-free grammar that generates L. To do so, we can think of any string in L as containing 5 parts $AWBW^rC$, where W^r is the reverse of W. A can be any string in $\{a, b, \#\}^\star$ ending with $\#$ or just an empty string, ϵ. Similarly, C can be any string in $\{a, b, \#\}^\star$ starting with $\#$ or just ϵ. B can be any string starting and ending with $\#$ or a or b or ϵ, where the latter three cases deal with the case where $i = j(x_i = x_i^r)$. Based on this idea, the following grammar will generate all strings in L.

$$
\begin{aligned}
S &\rightarrow A\,S'\,C \\
A &\rightarrow U\# \mid \epsilon \\
C &\rightarrow \#U \mid \epsilon \\
U &\rightarrow aU \mid bU \mid \#U \mid \epsilon \\
S' &\rightarrow aS'a \mid bS'b \mid S'' \\
S'' &\rightarrow \#U\# \mid \# \mid a \mid b \mid \epsilon
\end{aligned}
$$

Problem 2. Let $L = \{x\#y \mid x, y \in \{0, 1\}^\star$ and $x \neq y\}$. Show that L is a context-free language (CFL).

Solution: In order to show that L is a CFL, it's sufficient to provide a CFG that generates it. To do so, we can use the fact that the set of CFLs is closed under union and separate our solution into two cases: when $|x| \neq |y|$ and when $|x| = |y|$.

In order to deal with the first case where $|x| \neq |y|$, we can first create a variable T that generates an arbitrary string of equal length sides (where a variable X generates a single arbitrary symbol) and, then, create another variable U that generates an arbitrary string of size at least 1 and contatenate this string to either the beginning ($|x| > |y|$) or the end ($|x| < |y|$) of the string generated by X.

In order to deal with the second case where $|x| = |y|$, at least one symbol on the left must mismatch the corresponding symbol on the right. However, if we try to only generate strings of equal length sides, we won't be able to do it because the language

$$\{x\#y \mid x, y \in \{0, 1\}^\star \text{ and } x \neq y \text{ and } |x| = |y|\}$$

itself is not a CFL (it's a good exercise to prove that!). Therefore, we should make use of a little trick and also allow the generation of strings of unequal length sides. Because all such strings also belong to the language, the final result will still be correct. Then, we first do the case where a 0 on the left mismatches a 1 on the right. We can do so by using two variables A and B. Variable A generates an equal number of symbols to either side of itself and then becomes a 0, followed by an arbitrary string, followed by a $\#$. Variable B generates a 1 followed by an arbitrary string. Because the distance between the symbol 1 and the $\#$ is the same of that between the 0 and the beginning of the string, we are the x cannot be equal to y. A production $S_2 \rightarrow AB$ would then allow derivations of the form

$$S_2 \Rightarrow AB \overset{\star}{\Rightarrow} X^iAX^iB \overset{\star}{\Rightarrow} X^i0(\text{any})\#X^i1(\text{any}).$$

The case where a 1 on the left mismatches a 0 on the right can be generated in a similar fashion. Then, the entire grammar is:

$$
\begin{array}{lcll}
S & \rightarrow & S_1 \,| S_2 \,| S_3 & \\
S_1 & \rightarrow & TU \,| UT & \text{case 1: unequal length sides} \\
T & \rightarrow & XTX \,| \# & \\
U & \rightarrow & XU \,| X & \\
X & \rightarrow & 0 \,| 1 & \\
S_2 & \rightarrow & AB & \text{case 2: 0 on left mismatches 1 on right} \\
A & \rightarrow & XAX \,| 0E\# & \\
E & \rightarrow & XE \,| \epsilon & \\
B & \rightarrow & 1E & \\
S_3 & \rightarrow & PQ & \text{case 3: 1 on left mismatches 0 on right} \\
P & \rightarrow & XPX \,| 1E\# & \\
Q & \rightarrow & 0E &
\end{array}
$$

Problem 3. Construct a PDA for the language of all non-palindromes over $\{a, b\}$.

Solution: We can use the PDA for the language of all palindromes over $\{a, b\}$, given in the sample solutions, to create a PDA for this language. To change the PDA accepting all palindromes into one that accepts all non-palidromes, we simply insist in the new machine that there is *at least one inconsistency* between the first and second half of input string w. So the new PDA can essentially be the same, except when we are popping symbols off the stack and matching them with inputs, we must make sure that there is at least one a where there should have been a b or vice versa. Figure 1 shows the PDA.

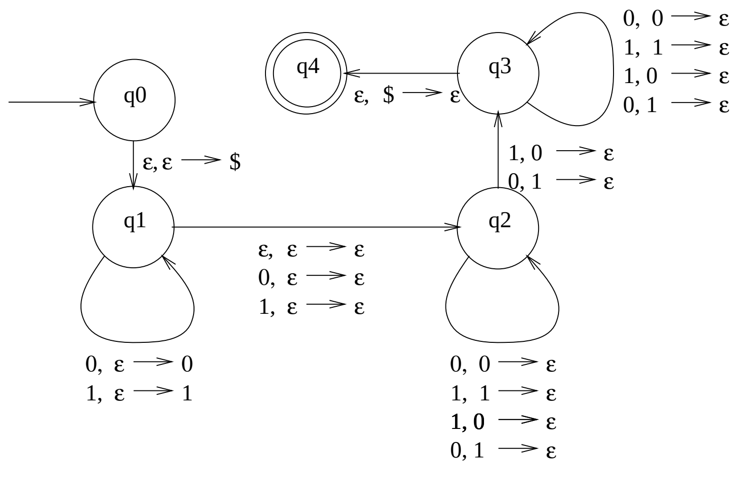

Figure 1: *PDA for the language of all non-palindromes over* $\{a, b\}$

We first mark the bottom of the stack with a \$, push the first half of the string (excepting the middle symbol if the string has odd length) onto the stack in q_1, guess nondeterministically where the middle of the string is and switch to state q_2. If $w^{\mathcal{R}} \neq w$, then at some point there will be a mismatch between what is on the stack and in the input; when this is true, the machine can take the transition from q_2 to q_3. Otherwise, any match or mismatch of inputs symbols to symbols on the stack is allowed. Finally, the machine accepts when 1) the input is exhausted and 2) the stack is empty.

Problem 4. Use the pumping lemma to show that the following language is not context free:

$$L = \{x_1\#x_2\#...\#x_k | k \geq 2, \text{each } x_i \in \{a,b\}^*, \text{and for some } i \neq j, x_i = x_j\}$$

Solution: Assume that L is CFL. Let p be the pumping length given by the pumping lemma and let $s = a^p b^p \# a^p b^p$. Obviously, $s \in L$. Because the length of s is grater than p we can split s into $uvxyz$ satisfying the following conditions:

1. for each $i \geq 0$, $uv^ixy^iz \in L$
2. $|vy| \geq 0$
3. $|vxy| \leq p$

For convenience, we will write s as $L\#R$, where L and R stand for the strings to the left and to the right of $\#$, respectively.

First of all, note that vy cannot contain $\#$, since uv^0xy^0z would contain no $\#$ which would contradict the fact that $k \geq 2$ (from the definition of L).

Then there are the following three cases for vy:

- vy contains more symbols from L than from R. In this case, $uv^2xy^2z = L'\#R'$, where $|L'| > |R'|$.
- vy contains more symbols from R than from L. In this case, $uv^2xy^2z = L'\#R'$, where $|L'| < |R'|$.
- vy contains equal number of symbols from both L and R. In this case, because of conditions 2 and 3 of the pumping lemma, $v = b^j$ and $w = a^j$ for some j such that $1 \leq j \leq p/2$. Therefore $uv^2xy^2z = a^pb^{p+j}\#a^{p+j}b^p$ which is not in the language.

Because we have found a word which is in L and which cannot be "pumped" the assumption that L is CFL must be false.