Harvard University
Computer Science 121

Problem Set 4

Due Friday, October 15, 2010 at 1:20 PM.
Submit a single PDF (lastname+ps4.pdf) of your solutions to cs121+ps4@seas.harvard.edu
Late problem sets may be turned in until Monday, October 18, 2010 at 1:20 PM with a 20% penalty.
See syllabus for collaboration policy.

SOLUTIONS

PROBLEM 1 (4+4+4 points)

Are the following languages context-free? Prove or disprove. *When giving a formal construction, no correctness proof is required: just an explanation.*
(A) $L = \{a^n b^m : n < m < 3n\}$
(B) $\{a^{n^2} : n \in \mathbb{N}\}$ over $\Sigma = \{a\}$
(C) $\{a^i b^j c^k : i, j, k \in \mathbb{N},\ \text{and if } i = 1 \text{ then } j \geq k\}$ over $\Sigma = \{a, b, c\}$

(A) Yes. L is generated by the grammar $S \to aSb \mid aSbb \mid aSbbb \mid abb$.

To see this, note that the recursive rules preserve $n \leq m \leq 3n$, and the only terminating rule makes the inequalities strict.

For the other direction, note that for a string with $m+1$ $a's$, we can use the first rule m times to get $m+1$ a's and $m+2$ b's. Then we can swap in rules $2, 3$ to increase b's by 1 up to $3m+2$ when only rule 3 is used.

Some simple inductions can fill in the details.

(B) No. We proceed with a proof by contradiction using the pumping lemma for context-free languages.

Assume L is context-free, and let p be the pumping length. Let $w = a^{p^2} \in L$, and let $w = uvxyz$ such that $|vy| > 0$ and $|vxy| \leq p$. Then $|uv^2xy^2z| = p^2 + |vy|$. But $p^2 < p^2 + |vy| \leq p^2 + |vxy| \leq p^2 + p < p^2 + 2p + 1 = (p+1)^2$. Since $|uv^2xy^2z|$ is strictly between two consecutive squares, we know that uv^2xy^2z is not in L. This contradicts the pumping lemma, so L must not be context-free.

(C) Yes. Note that $L = \{a^i b^j c^k : i \neq 1\} \cup \{a^i b^j c^k : j \geq k\}$. The former is regular - a regular expression for it is $(\epsilon \cup a^2 a^*) b^* c^*)$ - and thus it is context-free. The latter is generated by the grammar $S \to aS \mid T,\ \ T \to bT \mid bTc \mid \epsilon$. Since context-free languages are closed under union, L is then context-free.

PROBLEM 2 (6 points)

Draw the state diagram for a PDA for the language of all strings with twice as many as as bs over the alphabet $\{a, b\}$. Use the state diagram notation for PDAs given in Sipser.

(A)

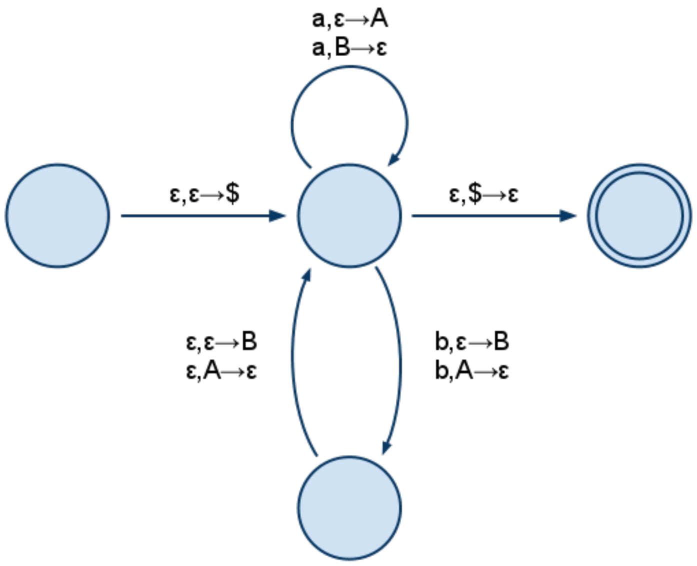

Figure 1: PDA for problem 2.

When the PDA reads an a, it pushes A or pops B. When it reads a B, it pushes two B's or pops two A's or one of each. If the stack is empty, there are twice as many A's.

PROBLEM 3 (2+8 points)

(A) Give an English language description of the language generated by the following grammar:

$$S \to aB \mid bA \mid \varepsilon$$

$$A \to aS \mid bAA$$

$$B \to bS \mid aBB$$

(B) Now, prove that the grammar does indeed produce the language that you describe.

(A) $L = \{w \in \{a, b\}^* : w$ has an equal number of a's and b's$\}$.

(B) First, we define the CFGs $G_X = (V, \Sigma, R, X)$ for all $X \in \{S, B, A\}$ - i.e. the same grammar with a different start state.

Now, define the following 3 statements.

- $P_1(n)$: for $|w| \leq n$, w has the same number of as and bs if and only if $w \in L(G_S)$
- $P_2(n)$: for $|w| \leq n$, w has one more a than bs if and only if $w \in L(G_A)$
- $P_3(n)$: for $|w| \leq n$, w has one more b than as if and only if $w \in L(G_B)$

Note that $P_i(n)$ is a true/false statement, and that we are ultimately trying to prove $P_1(n)$ is true for all n.

We will prove by induction on n that all three $P_i(n)$ hold for all n.

Base case: $n = 0$.

The only string of length 0 is ε. By inspection of the grammar, we see that $\varepsilon \in L(G_S)$, and that $\varepsilon \notin L(G_A)$ and $\varepsilon \notin L(G_B)$. This shows $P_i(0)$ for $i = 1, 2, 3$.

Inductive case:

Using strong induction, we assume that $P_i(k)$ is true for all $k \leq n$, and need to show that $P_i(n+1)$ holds.

Proof for $P_1(n+1)$: if $|w| = n+1$, $w \in L(G_s)$, we need to show that w must have the same number of as and bs. Since $w \neq \epsilon$, it must have been derived from S using either the aB or the bA rules. In the case of aB, w can be written as ax, $x \in B$, and $|x| = n$. By the induction hypothesis, x has one more b than a, so $w = ax$ has equal numbers of as and bs. The bA case is symmetric.

Now assume that $|w| = n+1$, and w has equal numbers of as and bs. We need to show $w \in L(G_s)$. Again, since we know that $|w| > 0$, w either starts with a or b. If it starts with a, then it can be written in the form ax, where $|x| = n$, and x has one more b than a. By the induction hypothesis, $x \in L(G_B)$, and so w can be derived from S: $S \Rightarrow aB \Rightarrow^* ax = w$. The case where w starts with b is symmetric.

Proof for $P_2(n+1)$: if $|w| = n+1$, $w \in L(G_A)$, we need to show that w must have one more a than b. If w starts with a, it is either just a (trivial), or was derived from A using the rule $A \to aS$. If so, we can use our induction hypothesis (specifically $P_1(n)$) to get that w has one more a than b. If w starts with b, it must have been derived via the rule $A \to bAA$. This means that w can be written as bxy, where $|x| \leq n$ and $|y| \leq n$, and $x, y \in L(G_A)$. Now again we can apply our induction hypothesis to get that w has one more a than b.

The other direction is that if $|w| = n+1$, and w has one more a than b, we need to show that $w \in L(G_A)$. If w starts with a, we have an argument very similar to the one above–it can be written ax, where x has equal numbers of as and bs, and so by the induction hypothesis $x \in L(G_S)$, meaning that w can be derived using the rule $A \to aS$, so $w \in L(G_A)$, as desired.

If w starts with b, we can write it as bx, where x has two more as than bs. If we can show that any such x can be written as $x = yz$, where y and z have one more a than b, then we get that byz can be derived from A via $A \to bAA$, which shows that $w \in L(G_A)$. Now we give the $x = yz$ argument:

Lemma: If a string x has two more as than bs, it can be written as $x = yz$, where both y and z have one more a than b.

Proof: Define a function $d_n(w)$, which returns the difference between the number of as and the number of bs in the first n symbols of a string w. Then we have that $d_0(x) = 0$, and that $d_{|x|}(x) = 2$. Note that d always goes either up or down by one for every extra symbol: $|d_n(x) - d_{n+1}(x)| = 1$ for all $n \leq |x|$. Thus, there must be some k such that $d_k(x) = 1$. This means that the first k symbols of x have one more a than b, implying that the remaining symbols of x also have one more a than b, which is what we needed to show.

Proof for $P_3(n+1)$: This case is symmetric to $P_2(n+1)$.

We're done! The above proves that $P_i(n)$ holds for all n, which means that $P_1(n)$ holds for all n, which means that $L(G)$ is in fact the language of strings with equal numbers of as and bs.

Note: Induction, like other mathematical tools, is used to make the intuitive formal and precise. Please proofread your proofs. If each statement doesn't follow from the ones before it, or if there is ambiguity, revise your proof. For example, "Because S must behave in the same way recursively, we know that aS results in a string with one more a than b" is not a mathematically precise statement. We make it precise using induction.

PROBLEM 4 (12 points)

Unlike the regular languages, the class of context-free languages is not closed under complement. In Example 2.38, Sipser proves that the language $L = \{ww : w \in \{a,b\}^*\}$ is not context-free. Here, you get to look at the complement of L:

Show that the language $\overline{L} = \overline{\{ww : w \in \{a,b\}^*\}}$ is context-free by giving a grammar that generates it. In a sentence or two, justify the correctness of your grammar.

(Hint: argue first that $\overline{L}$ is the set of all strings of the form $xaybz$ or $xbyaz$, where $|x|+|z| = |y|$, along with all strings of odd length.)

(A)

Before we dive into our grammar, we wish to understand intuitively what strings the language in question contains (as the hint suggests!). On its most basic level, the language contains all strings that *cannot* be formed by taking a string and concatenating it with itself. This means, first of all, that all odd-length strings are in this language. More subtly, let us consider the structure of an even-length string in this language, breaking each half of the string into a 'pre' section, a single symbol, and a 'post' section. That is, we can write any even length string in the language as $x\sigma yu\gamma v$. We observe that as long as σ and γ are in the same positions relative to their halves of the string and $\sigma \neq \gamma$, then the string will be in our language. That is, all even length strings in the language must either be of the form $xay'u'bv$ or $xby'u'av$ with $m = |x| = |u'|$ and $n = |y'| = |v|$. Note that this *is not* the format that the hint suggests, and that writing a grammar based on this format would be very difficult! What we want to do is redivide the substring $y'u'$ into yu such that $|x| = |y|$ and $|v| = |u|$ rather than what we have already ($|x| = |u'|$ and $|y'| = |v|$).

So now consider the length of the entire string expressed in terms of the lengths of these parts: $m+1+n+m+1+n$. But this is of course equivalent to $m+1+m+n+1+n$, which means that the even-length strings in our language can also be written as $xayubv$ or $xbyuav$ where $|x| = |y|$ and $|u| = |v|$, as the hint suggests. We can now build a CFG to generate this language:

Let $G = (V, \Sigma, R, S)$ with:

$$\begin{aligned}
V &= \{S, O, A, B, L, a, b\} \\
\Sigma &= \{a, b\} \\
R &= \{S \to O \mid AB \mid BA, \\
&\quad O \to OLL \mid L, \\
&\quad L \to a \mid b, \\
&\quad A \to LAL \mid a, \\
&\quad B \to LBL \mid b\}
\end{aligned}$$

PROBLEM 5 (Challenge!! 3 points)

Show that every context-free language over a unary alphabet is regular.

(A) We don't have solutions to this typed up yet.