Harvard University Extension School
Computer Science E-121

Problem Set 4

Due Due October 23, 2015 at 11:59 PM.
Submit your solutions electronically on the course website, located at https://canvas.harvard.edu/courses/4896/assignments. On the site, use the assignments tab to find the correct problem set, then with the "submit assignment" button, upload the PDF file of your solution.
LATE PROBLEM SETS WILL NOT BE ACCEPTED.
See the syllabus for the collaboration policy.

PROBLEM 1 (3+3+3 points, suggested length of 1/2 page)

For each of the following languages, state whether the language is context-free or not. If context-free, give a context-free grammar that generates the language (or construct a pushdown automaton that recognizes the language). If not context-free, write a short proof to justify.

(A) $\{a^n : n \text{ is a prime number}\}$.
(B) $\{a^n b^m : n, m \in \mathbb{N}, n \neq m \text{ and } n \neq 2m\}$.
(C) $\{a^n b^n c^n : n \in \mathbb{N}\} \cup \{(ab)^n (ca)^n (cb)^n : n \in \mathbb{N}\}$.

PROBLEM 2 (6 points, suggested length of 1/2 page)

Draw the state diagram of a PDA that recognizes the language $\{w : \text{the number of } a\text{'s in } w \text{ is greater than two times the number of } b\text{'s in } w\}$. Use Sipser's notation to label the transitions. (That is, use labels in the form $x, y \to z$ to mean that when you read an x from your string and pop a y from the stack, follow the transition and push a z onto the stack.) Explain in a few sentences why your construction is correct (no need to prove formally).

PROBLEM 3 (6 points, suggested length of 1/2 page)

A pushdown automaton (PDA) is made by equipping an NFA with a stack. In this problem, we consider an NFA equipped with a queue – a Queue Automaton (QA). The queue supports the following two operations:

1. *Pop.* Read the leftmost symbol of the queue and remove it from the queue.
2. *Push.* Write a symbol to the rightmost end of the queue.

In a Queue Automaton, a transition $(q, \sigma, \gamma) \to (q', \gamma')$ means "if the machine is in state q, after reading σ from the input string and popping γ from the queue, push γ' onto the queue and transition to state q'". Note that ε-transitions are still allowed. For example, $(q, \varepsilon, \gamma) \to (q', \gamma')$ doesn't read anything from the input string. You can also use an ε for γ to indicate that nothing should be popped from the queue, and an ε for γ' to indicate that nothing should be pushed onto the queue. Initially the queue is empty.

We saw in class that $L = \{a^{2^n} : n \in \mathbb{N}\}$ is not Context-Free, so there isn't a PDA that recognizes it. Construct a Queue Automaton that *does* recognize L (give the 6-tuple for it) and provide a short explanation for why it works.

PROBLEM 4 (4+6+(2) points, suggested length of 1 page)

Recall that allowing nondeterminism did not add any power to finite automata—any language that could be accepted by an NFA could also be accepted by a DFA. In this problem, you will show that this is *not the case* for PDAs by defining Deterministic PDAs (DPDAs).

DPDAs are similar to PDAs with a few subtle differences:

- Most notably, they are deterministic in their transitions and stack operations. Note that this means that they don't allow transitions on ε. In one step, a DPDA reads exactly one input symbol, reads and pops exactly one symbol from the top of the stack, and pushes a *string* onto the stack.
- Instead of starting with an empty stack, they start with a special symbol, \$, as the lone symbol on the stack. This way a DPDA can always know when it has reached the bottom of its stack. The \$ symbol can never be erased; that is, any transition reading \$ *must* push it back on the stack.

(A) Formalize the above by definining a DPDA as a 7-tuple $(Q, \Sigma, \Gamma, \$, \delta, q_0, F)$, filling in the requirements of each component. You can refer to the definitions of a regular PDA for the components that don't change.
(B) Now you will show that DPDAs are more powerful than finite automata. Consider the language $\textsc{Majority} = \{w : w \text{ contains more } a\text{'s than } b\text{'s}\}$. Construct a DPDA that recognizes $\textsc{Majority}$. Prove that no finite automata recognizes $\textsc{Majority}$.
(C) (CHALLENGE!!! Optional, worth 2 extra points.) Show that DPDAs are less powerful than PDAs.

PROBLEM 5 (8 points, suggested length of 3/4 page)

If L is a language, then $\textsc{Permutation}(L) = \{x : \text{there exists a string } w \in L \text{ such that } |x| = |w|$ and the number of occurrences of any letter in w and x are the same$\}$. Show that if L is regular, then $\textsc{Permutation}$(L) is context free when $\Sigma = \{a, b\}$. (Hint: Your proof must use the fact that there are only two symbols in the alphabet—the proof does not generalize to the case $|\Sigma| > 2$.)