Homework 4

Out: 12 Feb. Due: 19 Feb.

Instructions: *Submit your solutions in pdf format on Gradescope by* **5pm on Friday, February 19**. *Solutions may be written either in LaTeX (with either machine-drawn or hand-drawn diagrams) or* **legibly** *by hand. (The LaTeX source for this homework is provided in case you want to use it as a template.) Please be sure to begin the solution for each problem on a new page, and to tag each of your solutions to the correct problem! Per course policy, no late solutions will be accepted. Take time to write* **clear** *and* **concise** *answers; confused and long-winded solutions may be penalized. You are encouraged to form small groups (two to four people) to work through the homework, but you* **must** *write up all your solutions on your own. Depending on grading resources, we reserve the right to grade a random subset of the problems and check off the rest; so you are advised to attempt all the problems.*

1. Construct the DFA used in the KMP pattern matching algorithm for the pattern $ABABBABBA$ over alphabet $\{A, B\}$. You need not explain your construction, but you should label all the transitions clearly.

2. Consider the following generalization of the Majority problem in Note 3, known as the Heavy Hitters problem. We are given a stream of length n over alphabet Σ, and are asked to report any elements that occur more than $\frac{n}{k}$ times in the stream, where $k \geq 2$ is a fixed integer. (Thus the Majority problem corresponds to the case $k = 2$.) Show how to generalize the Majority streaming algorithm in Note 3 so that it solves the Heavy Hitters problem for any fixed value of k, using only $O(\log n + \log |\Sigma|)$ memory. Your algorithm should output at most $k - 1$ elements, including all elements that occur more than $\frac{n}{k}$ times (plus possibly some arbitrary additional elements if fewer than $k - 1$ elements occur more than $\frac{n}{k}$ times). Be sure to carefully justify the correctness of your algorithm.

3. In this problem, you are asked to give a lower bound on the memory size of streaming algorithms for testing whether a graph is connected. Recall the framework from the end of Note 3, where the graph is presented as a sequence of edges over its vertex set $V_n = \{1, \ldots, n\}$. By identifying a suitable set of distinguishable streams for the language $L_{n,\Sigma}$ of connected graphs on n vertices, prove that any streaming algorithm for this problem must use at least $\Omega(n)$ space.

[HINT: Consider graphs consisting of two large connected components.]

4. For each of the following non-regular languages, sketch the design of a (possibly non-deterministic) PDA that recognizes the language, and also give a context-free grammar that generates the language. For the PDA, you need not write out the states, transition function etc. explicitly; instead, you should give a high-level description of its operation. For the CFG, you should specify the production rules and briefly explain the role of each variable. You are not required to provide any proofs of correctness, but you are encouraged to try your constructions for yourself on some sample inputs to check that they work as expected.

(a) The set of 0-1 strings of the form $0^k 1 0^k$ for $k \geq 0$.

(b) The set $\{a^i b^j c^k : i, j, k \geq 0; i = j \vee j = k\}$ of strings over the alphabet $\{a, b, c\}$.

(c) The set of 0-1 strings that have exactly twice as many zeros as ones.

[continued on next page]

5. A *regular grammar* is a restricted type of context-free grammar in which all production rules are of the form $A \to wB$ or $A \to w$, where A, B are variables and w is a (possibly empty) string of terminals. Prove that the regular grammars generate *exactly* the regular languages.

[NOTE: You need to do two things. First, given a regular grammar G, show how to construct an NFA that accepts the language $L(G)$. Second, given a DFA M, show how to construct a regular grammar that generates the language $L(M)$. You do not need to prove formally that your constructions are correct, but you are encouraged to sketch inductive proofs for yourself to check their correctness.]