CS 125 Algorithms & Complexity — Fall 2016
Problem Set 6

Due: 11:59pm, Friday, October 21st

See homework submission instructions at `http://seas.harvard.edu/~cs125/fall16/schedule.htm`

Problem 5 is worth one-third of this problem set, and problems 1-4 constitute the remaining two-thirds.

Problem 1

For each of the following languages, determine whether or not they are regular and prove your answer.

(a) (2.5 points) $\{w \in \{a, b\}^* : w \text{ has more } a\text{'s than } b\text{'s}\}$.

(b) (2.5 points) $\{w \in \{a, b\}^* : \text{the number of occurrences of } ab \text{ in } w \text{ equals the number of occurrences of } ba\}$.

(c) (2.5 points) $\{w \in \{a, b, \ldots, z\}^* : |w| \text{ is a perfect square}\}$.

(d) (2.5 points) $\{w \in \{0, 1\}^* : w \text{ is the binary representation of a number divisible by } 3\}$.

Problem 2

Let $G = (V, E)$ be an unweighted, undirected graph with n vertices and m edges. Suppose that we do not want to find just one minimum cut, but want to count the *number* of minimum cuts (recall in class that we said the number of minimum cuts is never more than $\binom{n}{2}$, which is achieved by the n-cycle, but in general the number of minimum cuts could be any integer between 1 and $\binom{n}{2}$). In this problem we will give a randomized algorithm to accomplish this task.

(a) (3 points) Suppose we have n colored balls in a bucket, each with a different color. At each time step, we pick a uniformly random ball, observe its color, then put it back in the bucket. Show that the expected number of time steps before we observe each color at least once is $O(n \log n)$.

(b) (7 points) Give a randomized Monte Carlo algorithm to exactly count the number of minimum cuts. You may assume that one run of the contraction algorithm, to output a single cut (which we said in class is a mincut with probability at least $1/\binom{n}{2}$), can be implemented to take time $O(n^2)$. A modified version of Karger's basic contraction algorithm to solve this problem part is sufficient to receive full credit — you need not attempt to modify Karger-Stein. Your algorithm should fail to output the correct answer with probability at most P, for some given $0 < P < 1$.

Problem 3

Let $L_k = \{w \in \{a,b\}^* : \text{the } k\text{th symbol from the end of } w \text{ is } a\}$.

(a) (5 points) Show that L_k is recognized by a $(k+1)$-state NFA N_3. Draw the state diagram of N_3 and apply the subset construction to N_3 to obtain a DFA for L_3.

(b) (5 points) Show that every DFA to recognize L_k requires at least 2^k states. (Hint: use the Myhill-Nerode Theorem.)

(c) (**Challenge problem**, 0 points) The above shows that the subset construction is within a factor of 2 of optimal (since a language given by an NFA with $|Q| = k+1$ states requires at least $2^k = 2^{|Q|}/2$ states as a DFA). Close the gap between the upper bound and lower bound as much as you can.

Problem 4

In class, we saw how to decide whether a pattern $w \in \Sigma^*$ of length m is a substring of a string $x \in \Sigma^*$ of length n in time $O(m^3 \cdot |\Sigma| + n)$ by constructing a DFA $M_w = (Q = \{0, \ldots, m\}, \Sigma, \delta_w, q_0 = 0, F = \{m\})$ from w and then running M_w on x. Here you will see how to improve the algorithm to run in time $O(m+n)$. Given a pattern w, define an array $\pi_w = (\pi_w(1), \ldots, \pi_w(m)) \in \{0, \ldots, m\}^m$ where $\pi_w(i)$ is defined to be the largest $j < i$ such that $w_1 w_2 \cdots w_j = w_{i-j+1} w_{i-j+2} \cdots w_i$.

(a) (3 points) Show that given w, π_w, $q \in \{0, \ldots, m\}$, and $\sigma \in \Sigma$, the transition function $\delta_w(q, \sigma)$ can be evaluated in time at most $O(q + 2 - \delta_w(q, \sigma))$.

(b) (3 points) Show that given w, π_w, and a string $x \in \Sigma^*$ of length n, we can decide whether w is a substring of x in time $O(n)$. **Hint:** use (a) and look for a telescoping sum to obtain an amortized analysis.

(c) (4 points) Show that given w, the array π_w can be constructed in time $O(m)$. **Hint:** use $\pi_w(1), \ldots, \pi_w(i-1)$ to help construct $\pi_w(i)$ and again use an amortized analysis.

Problem 5 (Programming Problem)

Solve "FIELD" on the programming server `https://cs125.seas.harvard.edu`.
(under "Problem Set 6").