Problem Set 6

This problem set is due **at the beginning of** class on *Thursday, April 24, 2003*. PLEASE study for the quiz before working on this problem set.

Each problem is to be done on a separate sheet (or sheets) of paper. Mark the top of each sheet with your name, 6.046J/18.410J, the problem number, your recitation section, the date, and the names of any students with whom you collaborated.

Problem 6-1. On-line String Matching

In an on-line algorithm, the input is generated as the algorithm is running. The idea is to solve the problem efficiently before seeing all the input. You can't scan forward to look at 'future' input, but you can store all input seen so far, or some computation on it.

(a) In this setting, the text $T[1 \ldots n]$ is being broadcast on the network, one letter at a time, in the order $T[1], T[2], \ldots$. You are interested in checking if the text seen so far contains a pattern P. Every time you see the next letter of the text T, you want to check if the text seen so far contains P.

Design an algorithm that solves this problem efficiently. Your algorithm should use no more than $\Theta(m)$ time on preprocessing P (with m being the length of P). In addition it should do only constant amount of work per letter received. Your algorithm can be randomized, with constant probability of correctness.

(b) Now say that you have the same pattern P, but the text $T[1 \ldots n]$ is being broadcast in reverse. That is, in the order $T[n], T[n-1], \ldots$ Modify your algorithm so that it still detects the occurence of P in the text $T[i \ldots n]$ immediately (i.e., in constant time) after the letter $T[i]$ is seen.

Problem 6-2. Average-case analysis

Assume that for a pattern matcher, both the pattern and the text are randomly generated. That is, each symbol in both is selected independently and uniformly at random from $\{0, 1\}$. Show that the simple string matching algorithm given in the lecture, slide 3, runs in expected time $O(n)$.

Can you reproduce the result if the pattern in fixed, and only the text is random ?

Problem 6-3. Summations revisited

We will once again revisit the summations problem.

(a) Assume you are given two sets $A, B \subset \{0 \ldots m\}$. Your goal is to compute the set $C = \{x + y : x \in A, y \in B\}$. Note that the set of values in C could be in the range $0 \ldots 2m$.

Your solution should run in time $O(m \log m)$ (the sizes of $|A|$ and $|B|$ do not matter).

Example:

$$\begin{aligned} A &= \{1, 4\} \\ B &= \{1, 3\} \\ \\ C &= \{2, 4, 5, 7\} \end{aligned}$$

(b) Let us now return to an old problem from PS3. Assume you are given an array of n integers A which can take on values from $\{1 \ldots m\}$, $n \leq m$. Additionally, you are given a target sum x. The goal is to check if there are three different indices i, j, k such that $A[i] + A[j] + A[k] = x$. Show how to do it in time $O(m \log m)$.

Hint for both parts: Use fast polynomial multiplication.

Problem 6-4. String matching reduction.

Prof. Tidor designed a very efficient algorithm for the string matching problem with "don't care" symbols. Given a pattern P of size m and text T of size n, Tidor's algorithm runs in time $T(n, m)$. However, it only accepts strings (pattern and text) over the alphabet $\{A, G, T, C\}$ (plus the "don't care" symbols).

Show how to use Tidor's algorithm as a black-box to obtain an algorithm that handles *any* alphabet Σ. Your algorithm should have running time $O((n + T(n, m)) \log |\Sigma|)$. You can assume that the elements of Σ are represented by integers from $\{0 \ldots |\Sigma| - 1\}$.