CS 170 Algorithms
Spring 2015 Christos Papadimitriou and Prasad Raghavendra **HW 3**

Due Feb. 13, 5:00pm

Instructions. This homework is due Friday, February 13, at 5:00pm electronically. Same rules as for prior homeworks.

1. (30 pts.) Pattern matching, with tolerance for noise

We are given binary strings s,t; s is m bits long, and t is n bits long, and $m < n$. We are also given an integer k. We want to find whether s occurs as a substring of t, but with $\leq k$ errors, and if so, find all such matches. In other words, we want to determine whether there exists an index i such that $s_0, s_1, \ldots, s_{m-1}$ agrees with $t_i, t_{i+1}, t_{i+2}, \ldots, t_{i+m-1}$ in all but k bits; and if yes, find all such indices i.

(a) Describe an $O(mn)$ time algorithm for this string matching problem. Just show the pseudocode; you don't need to give a proof of correctness or show the running time.

(b) Let's work towards a faster algorithm. Suggest a way to choose polynomials $p(x), q(x)$ of degree $m-1, n-1$, respectively, with the following property: the coefficient of x^{m-1+i} in $p(x)q(x)$ is $m - 2d(i)$, where $d(i)$ is the number of bits that differ between $s_0, s_1, \ldots, s_{m-1}$ and $t_i, t_{i+1}, t_{i+2}, \ldots, t_{i+m-1}$.

Hint: use coefficients $+1$ and -1.

(c) Describe an $O(n \lg n)$ time algorithm for this string matching problem, taking advantage of the polynomials $p(x), q(x)$ from part (b).

(d) Now imagine that s,t are not binary strings, but DNA sequences: each position is either A, C, G, or T (rather than 0 or 1). As before, we want to check whether s matches any substring of t with $\leq k$ errors (i.e., $s_0, s_1, \ldots, s_{m-1}$ agrees with $t_i, t_{i+1}, t_{i+2}, \ldots, t_{i+m-1}$ in all but k letters), and if so, output the location of all such matches. Describe an $O(n \lg n)$ time algorithm for this problem.

Hint: encode each letter into 4 bits.

2. (20 pts.) Polynomial from roots

Given a polynomial with exactly n distinct roots at $r_1, \ldots, r_n$, compute the coefficient representation of this polynomial in time strictly faster than $O(n^2)$. There may be multiple possible answers, but your algorithm should return the polynomial where the coefficient of the highest degree term is 1.

Note: A root of a polynomial p is a number r such that $p(r) = 0$. The polynomial with roots $r_1, ..., r_k$ can be expressed as $\prod_i (x - r_i)$.

3. (20 pts.) Triple sum

Design an efficient algorithm for the following problem. We are given an array $A[0..n-1]$ with n elements, where each element of A is an integer in the range $-10n \leq A[i] \leq 10n$. The algorithm must answer the following yes-or-no question: does there exist indices i, j, k such that $A[i] + A[j] + A[k] = 0$?

Design an $O(n \lg n)$ time algorithm for this problem. Note that you do not need to actually return the indices; just yes or no is enough.

Hint: define a polynomial of degree $O(n)$ based upon A, then use FFT for fast polynomial multiplication. Remember that the multiplication of two polynomial terms will add their exponents.

Reminder: don't forget to include explanation, pseudocode, running time analysis, and proof of correctness.

4. (10 pts.) Cycle Detection

Design a linear-time algorithm which, given an undirected graph G and a particular edge e in it, determines whether G has a cycle containing e.

5. (20 pts.) Bipartite Graphs

A *bipartite graph* is a graph $G = (V, E)$ whose vertices can be partitioned into two sets ($V = V_1 \cup V_2$ and $V_1 \cap V_2 = \emptyset$) such that there are no edges between vertices in the same set (for instance, if $u, v \in V_1$, then there is no edge between u and v).

(a) Give a linear-time algorithm to determine whether an undirected graph is bipartite.

(b) There are many other ways to formulate this property. For instance, an undirected graph is bipartite if and only if it can be colored with just two colors. We say that a graph can be colored with k colors if we can assign one of k different colors to each vertex such that no adjacent vertices share the same color.

Prove the following formulation: an undirected graph is bipartite if and only if it contains no cycles of odd length.

(c) At most how many colors are needed to color in an undirected graph with exactly *one* odd-length cycle?