Homework 6. Due March 20

CS180: Algorithms and Complexity
Winter 2017

GUIDELINES:

- Upload your assignments to Gradescope by 6:59 PM.
- Follow the instructions mentioned on the course webpage for uploading to Gradescope very carefully (including starting each problem on a new page and matching the pages with the assignments); this makes it easy and smooth for everyone. As the guidelines are simple enough, bad uploads will not be graded.
- You may use results or algorithms proved in class without proofs as long as you state them clearly.
- Most importantly, make sure you adhere to the policies for academic honesty set out on the course webpage. The policies will be enforced strictly. Homework is a stepping stone for exams; keep in mind that reasonable partial credit will be awarded and trying the problems will help you a lot for the exams.
- Additional Guidelines for this assignment
  - For any universe U and your choice of a range size r, you have access to random hash functions $h : U \to \{1, \ldots, r\}$ as we discussed in class. For any $u \in U$, computing $h(u)$ takes $O(1)$ time and you don't have to take into account the space for computing h.
  - You can use the dictionary data structures we discussed in class. Make sure you explicitly state what your U and r are when you are using them.

1. Suppose that you are given as input a list of n birthdays (in the format $MMDDYYYY$ for example) give an algorithm to check if two people in the list have the same birthday. To get full-credit, your algorithm should always be correct and run in expected time $O(n)$. Your space usage should be $O(n)$ as well (and not $O(10^8)$). [.5 points]

2. Suppose you are writing a plagiarism detector. Students submit documents as part of a homework and each document is an (ordered) sequence of words. For some parameter m decided by the provost, we say two documents are copies of each other if one of them uses a sequences of m words (in that given order) from the other. Give an algorithm which given an integer m and N documents $D_1, \ldots, D_N$ as input, flags all submissions which are copies of some other submission. Your algorithm should run in expected $O(m + N + \text{total-length of documents})$ time (i.e., expected $O(m + N + n_1 + \ldots + n_N)$ time where n_j is the length of j'th document) and be always correct.

If needed, you may assume that your dictionary supports (in the same complexity as in class) INSERT operation for "*(key,value)*" pairs and LOOKUP(*key*) operation which along with checking if *key* exists, returns the associated *value* if it exits. [.5 points]

3. Consider the problem FIND-CLIQUE defined as follows: "Given a graph G and a number k as input, find a clique of size k in G if one exists." Recall the CLIQUE decision problem from class: "Given a graph G and a number k, does G contain a clique of size k?". Give a polynomial-time reduction from FIND-CLIQUE to CLIQUE. [.5 points]

4. Consider the problem LPS defined as follows: "Given a matrix $A \in \mathbb{R}^{n \times n}$, a vector $b \in \mathbb{R}^n$ and an integer $k > 0$, does there exist a vector $x \in \mathbb{R}^n$ with at most k non-zero entries such that $A \cdot x \geq b$". Here $A \cdot x$ denotes the usual matrix-vector product and for two vectors u, v, we say $u \geq v$ if for every i, $u_i \geq v_i$. Give a polynomial-time reduction from 3SAT to LPS. [.75 points]
(Hint: Use the reductions done in class to pick the "right" starting point.)

5. For this problem we need the notion of multi-variate polynomials over variables $x_1, \ldots, x_n$ and how they are specified. To review some terminology, we say a *monomial* is a product of a real-number co-efficient c and each variable x_i raised to some non-negative integer power a_i: we can write this as $cx_1^{a_1}x_2^{a_2}\cdots x_n^{a_n}$. A polynomial is then a sum of a finite set of monomials. (For example, $2x_1^2x_2x_3^4 + x_1x_3 - 6x_2^2x_3^2$ is a polynomial.)

   We say a polynomial P is of degree at most d, if for any monomial $cx_1^{a_1}x_2^{a_2}\cdots x_n^{a_n}$ appearing in P, $a_1 + a_2 + \cdots a_n \leq d$. (For example, the degree of the previous polynomial is 7). One can represent a n-variable polynomial of degree d by at most $(n+1)^d$ numbers (by d-dimensional arrays for instance - but let us ignore the actual details of the representation as this is not important and any representation will work for us).

   Consider the problem POLY-ROOT defined as follows: "Given a polynomial with integer coefficients of degree at most 6 as input, decide if there exists a $x \in \mathbb{R}^n$ such that $P(x) = 0$." Show that 3SAT reduces to POLY-ROOT. You don't have to write down the coefficients of the polynomials explicitly in your reduction - you can leave them as summations if it is more convenient for you (one reason why we didn't specify the exact representation). [.75 points]

   Hint: Try to define a polynomial equation for each clause in your 3SAT instance along with some other equations for variables and combine them into one big polynomial by using the fact that for any set of numbers $(a_1, \ldots, a_m)$, $\sum_i a_i^2 = 0$ if and only if $a_i = 0$ for all $i \in [m]$.

Additional problems. Do not turn in the solutions to these problems; they are meant for your practice and curiosity.

- Problems 1,3, 6, 26, 27, 30, 31, 41 from [KT]
- Problems 8.3, 8.4, 8.6, 8.8, 8.14, 8.23 from Chapter 8 of [DPV].