CSCI-GA 3520: Honors Analysis of Algorithms

Final Exam: Thu, Dec 21 2017, Room WWH-101, 10:00-2:00pm.

- This is a four hour exam. There are six questions, worth 10 points each. Answer all questions and all their subparts.
- This is a closed book exam. No books, notes, reference material, either hard-copy, soft-copy or online, is allowed.
- In the past years, to pass this exam, one needed to answer 3 or 4 problems well, instead of answering all the problems poorly.
- Please print your name and SID on the front of the envelope only (not on the exam booklets). Please answer each question in a separate booklet, and number each booklet according to the question.
- Read the questions carefully. Keep your answers legible, and brief but precise. Assume standard results and algorithms (i.e. those taught in class or referred to in the homeworks).
- **You must prove correctness of your algorithm and prove its time bound unless stated otherwise. The algorithm can be written in plain English (preferred) or as a pseudo-code.**

Best of luck!

Problem 1

A *queue* is a data structure that maintains a sequence of (integer) values

$$(a_1, a_2, \ldots, a_n),$$

so as to perform two operations: (1) Add: add a new value at the *end* of the queue (2) Delete: delete the value at the *front* of the queue. For example, the operations Add b, Delete, Delete, applied to the queue above would result in the updated queue

$$(a_3, \ldots, a_n, b).$$

Clearly, a queue can be implemented simply as a linked list. However (for reasons below) we wish to implement a queue using stacks. Recall that a stack supports Push and Pop operations that push or pop *one* element from the top of the stack.

(a) Show how to implement a queue using two stacks.

(b) Show, using amortized analysis, that a sequence of n Add and Delete operations can be performed in $O(n)$ time. You could define a suitable potential function and show that each operation takes $O(1)$ amortized time.

(c) Show that your implementation can also support the Find-Min operation that reports the minimum value in the queue in $O(1)$ time. A minor modification may be needed.

Hint: Keep all the data in two stacks. When one stack is empty and you need to delete a value from this stack, what would you do?

Problem 2

Let $G = (V, E)$ be a directed graph. Each edge has a non-negative integer weight and a color, which is either red, white, or blue. Let $s, t \in V$ be the start and target nodes respectively. A path from s to t is called patriotic if it starts with a sequence of zero or more red edges, followed by a sequence of zero or more white edges, and ends with a sequence of zero or more blue edges.

Design an algorithm, as efficient as you can, that finds a patriotic path from s to t of minimum weight. What is the running time of your algorithm? You may use any standard algorithm as subroutine. Assume adjacency list representation of the graph.

Hint: One could reduce the given problem to the standard shortest path problem. A less efficient (but still polynomial time) algorithm would get partial credit.

Problem 3

In a directed graph $G(V, E)$, a subset of nodes $S \subseteq V$ is called a *core set* if every node in the graph is reachable from some node in S. That is, for every $w \in V$, there is a path from v to w for some $v \in S$.

(a) Show that if $G(V, E)$ is an acyclic graph, then there is a unique core set of minimum size.

(b) Now suppose that $G(V, E)$ is an arbitrary directed graph (i.e. not necessarily acyclic). Suppose moreover that each node $v \in V$ has an associated positive integer cost $c(v)$. For a set $S \subseteq V$, we define the cost of S to be $\sum_{v \in S} c(v)$. Give an $O(|V| + |E|)$-time algorithm to find a core set S with the least cost.

You may use any standard algorithm as subroutine. Assume adjacency list representation of the graph.

Problem 4

A particle takes an n-step *random walk* on the set of integers below, starting at the origin:

$$\{-n, -(n-1), \ldots, -2, -1, 0, 1, 2, \ldots, n\}.$$

That is, the particle starts at 0 and at each step, moves one unit to the left or to the right with probability $\frac{1}{2}$ each, independently for each of the n steps. Let X denote the position (an integer value) of the particle when the walk ends. Answer the questions below along with short justifications.

(a) Find $\Pr[X = n]$.

(b) Find $\mathbb{E}[X]$.

(c) Find $\mathbb{E}[X^2]$.

(d) Give an upper bound (as low as you can) on the probability

$$\Pr\left[|X| \geqslant \frac{n}{2}\right].$$

(e) Find a function $t(n)$ (as low as you can) so that

$$\Pr\left[|X| \geqslant t(n)\right] \leqslant \frac{1}{100}.$$

Hint: You could express X as a sum of independent random variables.

Problem 5

An instance ϕ of *Monotone 2SAT* consists of n Boolean variables $x_1, \dots, x_n$ and m clauses $C_1, \dots, C_m$, where each clause is of the form:

$$x_i \vee x_j,$$

i.e. the variables do not appear in negated form. Recall that a Boolean assignment to the variables is a map $\sigma : \{x_1, \dots, x_n\} \to \{\mathsf{True}, \mathsf{False}\}$. The assignment is said to satisfy ϕ if it satisfies every clause of ϕ. The *Hamming weight* of an assignment σ is the number of variables that σ assigns True. Consider the following language (k denotes a positive integer):

$$\begin{aligned} L \;=\; \{(\phi, k) \mid & \; \phi \text{ is a Monotone 2SAT instance that has a satisfying assignment } \sigma \\ & \text{with Hamming weight at most } k.\} \end{aligned}$$

Show that L is NP-complete.

Hint: You may use a reduction from any well-known graph theoretic NP-complete problem (which includes any problem mentioned in class).

Problem 6

The road-builder's problem: When building a road, the goal is twofold: first, to keep it as flat as possible, and second, to not move too much soil. We formalize this problem as follows.

Suppose a road will run for n meters, where the underlying terrain has non-negative integer heights $h_1, h_2, \dots, h_n$ on each successive meter. You may assume that $0 \leqslant h_i \leqslant n$ for all i. Let the constructed road have non-negative integer heights $k_1, k_2, \dots, k_n$ on these same meters of length. Suppose that all the changes of height can be achieved by moving the soil corresponding to differences in height from one location to another, hence it holds that

$$\sum_{i=1}^{n} h_i = \sum_{i=1}^{n} k_i.$$

Suppose further that the goal is to minimize the following function:

$$\sum_{i=1}^{n-1} (k_{i+1} - k_i)^2 + \sum_{i=1}^{n-1} \left| \sum_{j=1}^{i} (k_i - h_i) \right|. \tag{1}$$

The first term is a cost determined by the height differences of the road from one unit of length to the next. The second term relates to the total distance the different amounts of soil are moved. Note that $V_i = \sum_{j=1}^{i} (k_i - h_i)$ is the volume of soil moved across the boundary between the i^{th} meter and $(i+1)^{st}$ meter (it could be negative).

Give a polynomial time algorithm to compute heights $k_1, k_2, \dots, k_n$ minimizing the above function.

Hint: Note that it must be the case that $0 \leqslant k_i \leqslant n^2$ and moreover $-n^2 \leqslant V_i \leqslant n^2$ (Why?). Use the possible values of V_i and k_i as parameters in your algorithm. Also, you should treat (1) *as a black box. Don't try to do anything clever with this formula.*