Problem Set 5

This problem set is due in lecture on Wednesday, November 28th, 2007.

Instructions:

- Mark the top of each sheet with your name, the course number, the problem number, your recitation section, the date and the names of any students with whom you collaborated.

- You will often be called upon to “give an algorithm” to solve a certain problem. Your write-up should take the form of a short essay. A topic paragraph should summarize the problem you are solving and what your results are. The body of the essay should provide the following:
  1. A description of the algorithm in English and, if helpful, pseudo-code.
  2. At least one worked example or diagram to show more precisely how your algorithm works.
  3. A proof (or indication) of the correctness of the algorithm.
  4. An analysis of the running time of the algorithm.

Remember, your goal is to communicate. Full credit will be given only to the correct solution which are described clearly. Convoluted and obtuse descriptions might receive low marks, even when they are correct. Also, aim for concise solutions, as it will save you time spent on write-ups, prevent you from writing convoluted solutions, and also help you conceptualize the key idea of the problem.

- Be sure to know the section on “Guide to writing up homework” in the “Course Information” handout before writing your solutions.

Problem 5-1. Finding the mode of a set without much space.

(a) This problem is a preparation for the next part. Suppose that you are given an array $A$ of $n$ elements. Give an $O(n \log n)$ algorithm to find the most frequent element of $A$. Can you do better if you know that all the elements of $A$ are small positive integers?

(b) Suppose now that you are given a read-only media containing a HUGE string $A$. The string $A$ has a total length of $L$ characters and consists of $n$ phrases separated by a special character. Your task is to find the most frequent phrase in $A$. (For instance, think of $A$ as the concatenation of the text of all English webpages found in Internet. Your mission is to find the most common English phrase in Internet).
Assume that \( L \) is so big that you can’t fit \( A \) in memory (and since \( A \) is read-only, you can’t use any in-place algorithm). Also assume that \( n \ll L \). Devise an algorithm based on hashing that returns the most frequent phrase in \( A \) using only \( O(n \log n) \) bits of extra space. Can you make the running time of your algorithm linear in \( L \)?

(c) (Optional) Assume now that \( A \) is given to you as a stream of data (i.e. you can only read it once from left to right). Modify your previous algorithm to output, with high probability, an index \( i \) such that the \( i \)-th phrase of \( A \) is the most frequent phrase of \( A \).

**Problem 5-2.** Maximizing skiing fun

A group of 6.046 students are preparing to go skiing during winter break. They plan to use what they learned in 6.046 to maximize their fun. They have a map, which they naturally regard as an undirected graph \( G = (V, E) \); where vertices represent locations and edges represent available trails that can be either climbed up or skied down. Some of the locations are bus stops, denoted by \( S \subset V \).

To simplify their problem, they decide to look for a route such that all of the uphill segments come at the beginning and all of the downhill segments come at the end. (The idea is to climb uphill first, and then go downhill skiing). Call such a route a “valid” route.

Let \( l(e) \in \mathbb{R}^+ \) denote the length of a trail \( e \in E \) and \( h(v) \in \mathbb{R} \) denote the elevation (height) of a location \( v \in V \). Assume that no two locations have the same elevation.

(a) Define the “funniness value” of a route to be the sum of the length of the downhill segments minus the sum of the length of the uphill segments. The students want to find a good valid route with the property that they can reach the start of the route by bus and they can also leave from the end of the route by bus.

Give an efficient algorithm to find the funniest valid route that starts and ends at a bus stop. Analyze the running time of your algorithm as a function of \( n = |V|, m = |E| \) and \( s = |S| \).

(b) For a fixed \( k \geq 1 \), we say that a route \( P \) is a \( k \)-valid route if \( P \) is a (possible self-intersecting) walk that can be decomposed as \( P = U_1D_1U_2D_2 \ldots U_kD_k \) where for every \( i, U_i \) is a (possibly empty) path with only uphill segments and \( D_i \) is a (possibly empty) path with only downhill segments. Under this definition, a \( 1 \)-valid route is simply what we denoted before as a valid route. Give an efficient algorithm to find the funniest \( 2 \)-valid route that starts and ends at a bus stop.

(c) (Optional). Give an efficient algorithm to find the funniest \( k \)-valid route that starts and ends at a bus stop.

**Problem 5-3.** Odd couple roadtrips.

Alice and Bob are a quite odd couple. They both like driving a lot, so everytime they go out together they argue about who should be the driver.
For every trip they do, they use the same route map which they can represent as an undirected graph $G = (V, E)$, where vertices represent locations and edges represent routes. Also, for every $e \in E$, $l(e) \in \mathbb{R}^+$ denotes the length of that route. (Assume that $G$ is connected and has no loops).

To avoid conflicts they have one golden rule: Everytime they stop at (or pass through) a location, they swap driver; and these are the only times when they swap. Due to some really awkward situation in the past they never break this rule.

(a) Next weekend, Alice and Bob need to make a trip from Somerville ($s$) to Toronto ($t$). Since Alice knows both places better, they decide that Alice should drive in the first and last segment of the route.

Devise an efficient algorithm to find a route of minimum distance between both places, such that their trip start with Alice driving and, according to their golden rule, also ends with Alice driving. Your algorithm should also return a message saying “No path” if there is no such path.

(b) Although Bob does not like to admit it, he has a problem getting the car in and out the garage. For that reason, everytime they need to go out for a roundtrip ride, they prefer that Alice drives both the first and last segment of the trip.

They discovered that the place where they live now is such that no matter what closed route they take that starts and ends with Alice driving, the trip takes too long (remember that $G$ has no loops). For that reason they are planing to move to a place where this does not happen.

Devise an algorithm for finding one location $v$ that minimizes the shortest closed route that starts and ends at $v$ with Alice driving in the first and last segment.