Homework 11 – Due Friday, April 20, 2012

Reminders  Collaboration is permitted, but you must write the solutions by yourself without assistance, and be ready to explain them orally to a member of the course staff if asked. You must also identify your collaborators. Getting solutions from outside sources such as the Web or students not enrolled in the class is strictly forbidden.

Exercises  These should not be handed in, but the material they cover may appear on exams.

• Show how “Kevin Bacon numbers” (the length of the shortest chain of movies connecting any actor to the actor Kevin Bacon) can be represented as distances in a graph. Which algorithms from class would you use to compute Kevin Bacon numbers, and how long would it take as a function of the number \( n \) of actors, number \( t \) of movies, and the number \( A \) of actors who star in each movie?


• Rewrite DFS to use an explicit stack instead of recursion.

• Rewrite DFS to use only 1 bit of extra memory per vertex (assuming the memory used to store the graph’s adjacency lists representation is read-only). [In other words, how would you explore a maze using only pennies to mark where you’ve been?]

• A sink in a graph is a vertex with no outgoing edges. Give an algorithm takes as input the adjacency list representation of a graph and finds a sink (or returns an error if no sink exists) in time \( O(|V|) \), independent of the number of edges.

• Give an algorithm takes as input the adjacency list representation of a graph and determines if the graph contains a cycle in time \( O(|V|) \), independent of the number of edges.

• (Courtesy of Jeff Erickson) After a long algorithms exam, you decide to go home by bus. Since you planned ahead, you have a schedule that lists the times and locations of every stop of every bus in State College.

You want to determine the sequence of bus rides that will get you home as early as possible (you have to transfer several times, unfortunately, to get home from Wartik). There are \( b \) different bus lines, at most \( s \) stops on each bus line, and a bus runs on each bus line at most \( k \) times per day. Your goal is to minimize your arrival time, not the time you actually spend traveling.

Assume that the buses run exactly on schedule, that you have an accurate watch, and that you are too tired to walk between bus stops. [Also pretend, for the purposes of the question, that State College has enough bus routes for asymptotic analysis to matter.]
Describe and analyze an efficient algorithm to compute the best route. Remember to state your time bounds in terms of $k, b$ and $s$. Faster correct algorithms are worth slightly more points than slower ones. *Hint:* How would you phrase this as a graph problem?

**Problems to be handed in.** *Please submit each problem on a separate sheet of paper.* Corrections/clarifications in red.

1. **(Shortest Cycles)** Give algorithms that are as fast as you can give for the following two problems:

   (a) Given an undirected graph $G = (V, E)$ and a vertex $u \in V$, find a shortest cycle involving $u$.
   
   (b) Given an undirected graph $G = (V, E)$, find a shortest cycle in the whole graph $G$.

   For each problem, (i) give an English description of the algorithm, as well as pseudocode (ii) prove that your algorithm is correct, (iii) analyze the running time. As usual, you can use any algorithms we’ve covered in class as subroutines and there is no need to reprove statements that we proved in class or from the book.

2. **(Motion Planning)** Some friends of yours are working on techniques for coordinating groups of mobile robots. Each robot has a radio transmitter that is used to communicate with a base station, and your friends find that if the robots get too close to one another, then there are problems with interference among the transmitters. Your friends need to plan the motion of the robots in such a way that each robot gets to its intended destination, but in the process the robots don’t come close enough together to cause interference problems.

   We can model this problem abstractly as follows. The floor plan of a building is represented by an undirected graph $G = (V, E)$, and there are two robots initially located at nodes $a$ and $b$ in the graph. The robot at node $a$ wants to travel to node $c$, and the robot at node $b$ wants to travel to node $d$. This is accomplished by means of a schedule: at each time step, the schedule specifies that one of the robots moves across a single edge, from one node to a neighboring node; at the end of the schedule, the robot from node $a$ should be sitting on $c$ and the robot from node $b$ should be sitting on $d$. A schedule is *interference-free* if there is no point at which the two robots occupy nodes that are at distance $\leq r$ from one another in the graph, for a given parameter $r$. Assume that the starting nodes $a$ and $b$ are at a distance of greater than $r$, and so are the ending nodes $c$ and $d$.

   In this problem you’ll design an algorithm that finds an interference-free schedule if one exists.

   (a) A *configuration* is a pair of nodes in $G$, where the two robots are located. Your goal is to get from a given starting configuration $(a, b)$ to a given ending configuration $(c, d)$, by making “legal” moves between configurations (one robot can change its location to a neighboring node), and choosing only “legal” configurations (the robots must always be at distance at least $r + 1$ from one another).

   Define the following graph $H$: the nodes are all possible configurations of the robots; that is, the set of nodes in $H$ consists of all possible pairs of nodes in $G$. We join two nodes in $H$ by an edge if they represent configurations that could be consecutive in a schedule: that is, $(u, v)$ is connected to $(u', v')$ if either $u = u'$ and there is an edge in $G$ from $v$ to $v'$, or if $v = v'$ and there is an edge in $G$ from $u$ to $u'$. 

2
i. How many vertices does $H$ have?
ii. How many edges does $H$ have?

(b) Note that paths in $H$ from $(a, b)$ to $(c, d)$ correspond to schedules for the robots. However, the resulting schedules might not be interference-free. We'll form a graph $H'$ by deleting certain nodes from $H$, so that any path in $H'$ is an interference-free schedule.

i. Which nodes should we delete from $H$ to obtain $H'$?
ii. Once you have $H'$, how can you obtain an interference-free schedule? State the running time of the algorithm in terms of $n_{H'}$ and $m_{H'}$, the number of vertices and edges in $H'$.

(c) Give an algorithm for computing the adjacency list representation of $H'$ from the adjacency list representation of $G$. What is the running time of your algorithm?

(d) State the running time (in terms of $n$ and $m$) of the algorithm for the original problem resulting from combining parts (c) and (b[ii]).

3. (Implementing simple traversals) Implement BFS and DFS in Python for the data structure used during the last programming assignment. Modify the algorithms to compute the average depth of a node in the BFS (respectively, DFS) tree for the graph. Note that this number may depend on the order in which you visit the neighbors of a vertex (since that order may affect the tree) and on the starting vertex, which we called $s$ in class. To keep things well-defined: start your search from a uniformly random vertex in the graph. (Repeat several times to get an estimate of the average, and include error bars.)

Run your algorithms on the two large graphs from the last programming assignment. What are the average depths of BFS and DFS trees in those graphs?

As with the last assignment, please submit:

(a) Documented Python code
(b) Tests on small graphs of your own devising
(c) Results from your executions on the two graphs from last week.

As with the previous programming assignment, this one may be done in teams of up to three.

4. (Extra Credit) Show that an undirected graph with no cycles of length 4 has $O(n^{3/2})$ edges.