CS 466/666 Fall 2004
Naomi Nishimura

Assignment 3
Due November 16 at noon

For all problems you are expected to justify your answers, by showing your work or stating arguments, as is appropriate.

For any algorithm that you provide, briefly justify its correctness and running time, as well as its adherence to any additional requirements stated in the question.

The solutions you hand in must be entirely your own work. Do not look up either full or partial solutions in the literature or on the Internet.

Please read course policies and the course Web page for more information.

1. *[16 marks]* Give a polynomial-time 2-approximation algorithm for the following problem. The running time should be as small as possible.

   Instance: A directed graph $G = (V, E)$ with weight $w(e)$ on edge e.

   Solution: A subset $E' \subseteq E$ such that $G' = (V, E')$ is acyclic.

   Measure: The sum of the weights of edges in E'.

   Maximization

2. *[24 marks]* Consider the greedy colouring algorithm that was used as a subroutine in the colouring of planar graphs; we saw that it provided an absolute approximation for graphs of bounded degree. For each of the following classes of graphs, state and prove whether the algorithm provides an exact solution, an absolute approximation, or neither. Recall that any ordering of the vertices is possible.

   (a) *[4 marks]* Paths.

   (b) *[4 marks]* Odd cycles.

   (c) *[4 marks]* Complete graphs.

   (d) *[12 marks]* Bipartite graphs.

3. *[16 marks]* Give a polynomial-time algorithm that colours a 3-colourable graph with at most $4\lceil\sqrt{n}\rceil + 1$ colours. Hint: you may find it useful to show that the subgraph induced on the neighbourhood of any vertex in a 3-colourable graph is 2-colourable.

4. *[12 marks]* Consider an algorithm for vertex cover that proceeds as follows: starting with a graph of unmarked vertices, repeatedly choose an unmarked vertex v of highest degree to add to the vertex cover, marking v and all its neighbours. After all vertices have been marked in this manner, we remove each edge that has at least one endpoint in the vertex cover, unmark all vertices with at least one remaining edge, and then run the algorithm again on the smaller graph.

   (a) *[4 marks]* Give an example for each n on which the algorithm gives an optimal solution. Justify your claim.

(b) *[8 marks]* Give a nontrivial example that shows that the algorithm is not a 2-approximation algorithm. Justify your claim.

5. *[12 marks]* Consider the following algorithm for TSP: we first form a path, repeatedly selecting edges for the path by taking the cheapest remaining edge, discarding it if it forms a cycle or results in a vertex of degree three in the graph induced on the selected edges, and adding it to the set of selected path edges otherwise. Once we have a path containing all vertices, we add the edge between the two endpoints of the path. Recall that TSP takes as input a complete weighted graph.

   (a) *[4 marks]* Give an example on which the algorithm gives an optimal solution. Justify your claim.

   (b) *[8 marks]* Give an example that shows that the algorithm is not a c-approximation algorithm for any value c. Justify your claim.

6. *[20 marks]* Show that unless P=NP, no polynomial-time absolute approximation algorithm exists for maximum independent set.