COMP 5711: Advanced Algorithms
Fall 2014 Midterm Exam

Instructions

1. Print your name, student ID at the top of every page (in case the staple falls out!).
2. This is a open-book, open-notes, open-brain exam.
3. Time limit: 120 minutes.
4. You can write on the back of the paper if you run out of space. Please let us know if you need more scratch paper.
5. For a maximization problem where the algorithm's solution is at least a fraction of $1/c$ of the optimal solution for some $c \geq 1$, we may either say it is a c-approximation or (equivalently) a $(1/c)$-approximation.

1. **Short Questions** (15 pts)

   (a) For a problem parametrized by k (in addition to the input size n), suppose we have an algorithm with running time $O\left(k^k \cdot \text{poly}(n)\right)$. For what values of k is this algorithm a polynomial-time algorithm?

   (b) We know that whether a problem belongs to FPT crucially depends on the choice of the parameter k. If we choose $k = n$, what would the class FPT become?

   (c) The vertex cover problem and the independent set problem are equivalent, since on any graph $G = (V, E)$, $C \subseteq V$ is a vertex cover iff $V - C$ is an independent set. So the 2-approximation algorithm for the vertex cover algorithm is also a 2-approximation algorithm for the independent set problem. Is this true or false? Give a one-sentence justification for your answer.

2. **Dominating Set** (20 pts)

A *dominating set* for a graph $G = (V, E)$ is a subset D of V such that every vertex not in D is adjacent to at least one member of D. The dominating set problem asks to find the smallest dominating set for a given graph G. Define $x_u = 1$ if vertex u is picked in the dominating set and 0 otherwise. Write an integer linear program for this problem using the x_u's. Then use LP relaxation and rounding to give a $(d+1)$-approximation for this problem, where d is the maximum degree of G.

3. **FPTAS for Knapsack** (20 pts)

   Recall that in the FPTAS we gave for the *knapsack problem*, we rounded *up* all the values to the nearest multiple of θ, and then run the dynamic programming algorithm. Now consider the other two possible ways of rounding:

   (a) If we round all values *down* to the nearest multiple of θ (for whatever θ that is necessary), and run the dynamic programming algorithm, does this still give an FPTAS? If yes, give a proof. If no, say why.

   (b) If we round all values up to the nearest power of θ (for whatever θ that is necessary), and run the dynamic programming algorithm, does this still give an FPTAS? If yes, give a proof. If no, say why.

4. **Local Search for Vertex Cover** (15 pts)

   We have given two different algorithms for finding a 2-approximation for the vertex cover problem, one using the pricing method and the other using LP rounding. In fact, there is also a local search algorithm that yields a 2-approximation. The algorithm turns out to be exactly the same as the one in Assignment 2 for finding a locally optimal matching:

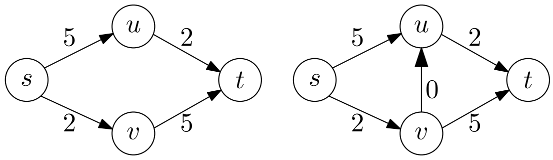
   As long as there is an edge whose endpoints are unmatched, add it to the current matching. When there is no longer such an edge, terminate with a locally optimal matching.

   To use this algorithm for the vertex cover problem, after the local search terminates, we simply add both endpoints of every edge in the matching to the vertex cover. Show that this gives a 2-approximation for the vertex cover problem. Note that we do not assume that the graph is bipartite here.

5. **Selfish Multicast Routing with Congestion** (10 pts)

Recall that in the *multicast routing problem* discussed in class, we are given k player, each wanting to route one unit of flow from some source to some destination. Every edge e is associated with a cost c_e, and each player pays a price of c_e/x_e where x_e is the number of players using edge e. Now suppose that the price each player pays is now $c_e \cdot x_e$. This for example can model the congestion caused by multiple players using the same edge, where the delay on an edge increases linearly as the number of players sharing that edge.

In each of the following two networks, find a Nash equilibrium as well as the social optimum. Suppose there are two players, and both want to route from the same source s to the same destination t. Note that (b) is simply (a) but with one additional, zero-cost edge (which might correspond to a high-speed optical cable).



(a) Original network (b) Augmented network

6. **Maximum Coverage** (20 pts)

   Given a set U of n elements, a list $S_1, \ldots, S_m$ of subsets of U, and an integer k, the *maximum coverage problem* asks us to pick k sets so as to cover as many elements as possible. Show that the obvious algorithm, which greedily picks the set that the covers the most currently uncovered elements and does so for k iterations, is a constant factor approximation, and find the approximation factor. [Hint: Let O denote the set of elements covered by the optimal solution. Consider how the size of O reduces as these elements are covered by the greedy algorithm over the k iterations.]