Algorithmic Game Theory	**Problem Set 1 (missing assumptions added)**
CS 684 Spring 2004	**Due Monday, February 23, 2004**

Solve at least 3 of the following 4 problems. You may solve all 4 for extra credit. The problems are of varying difficulty, but are worth equal credit. We will maintain a FAQ for the problem set on the course Web page. You may use any fact we proved in class without proving the proof or reference.

(1) Consider the following load balancing game. There are n jobs each controlled by a separate and selfish user. There are m servers S that can serve jobs, and each job j has an associated set $S_j \subseteq S$ of servers where it can possibly be served. For this problem we assume that the load of each jobs is 1, and each server i has a load dependent response time: $r_i(x)$ is the response time of server i if its load is x. We assume that $r_i(x)$ is a monotone increasing function for all i. We showed in class that this is an (atomic) potential game, and hence has a deterministic Nash equilibrium.

(a) Give a polynomial time algorithm to find an equilibrium.

(b) We considered two possible definitions of social optimum for this game. First consider the assignment of jobs to servers that minimizes the maximum response time, and give a polynomial time algorithm to find the best assignment for this objective function.

(c) Next considered the assignment of jobs to servers that minimizes the sum of all response times, and give a polynomial time algorithm to find the best assignment for this objective function.

Hint: The minimum cost matching problem (defined below) can be solved in polynomial time. This may be useful as a subroutine. The minimum cost matching problem is given by a bipartite graph G, costs on the edges and an integer k, and the problem is to find a matching in G of size k of minimum possible cost.

(2) Consider the game from the previous problem in the special case that the response time is directly proportional to the load, that is $r_i(x) = x$ for all i, so the goal of the users is to be on servers with small load. In this problem we consider the ratio of the worst possible Nash equilibrium and the optimum under the objective function minimizing the maximum load. (Often referred to as the *min-max* objective.) (Recall that in class we proved that if $S_j = S$ for each j, than the cost of any Nash equilibrium under this objective function is at most twice the minimum possible load, and this was true even if the jobs can have different loads. Here we assume all jobs have load 1, but are only allowed to be served from the subset S_j of the servers.)

(a) Give an example of a set of n jobs and n servers when there is an assignment of jobs to servers with maximum load 1, and there is a Nash equilibrium where a machine has load $\approx \log n$.

(b) Show that if there are m machines, than the maximum load in a Nash equilibrium is at most an $O(\log m)$ factor above the minimum possible value of the maximum load.

(3) Consider a continuous version of the above game analogous to the routing game we considered in class. Assume there are m servers S, and n types of jobs, and for each job type $j \in J$ have a subset S_j of the servers that can serve jobs of type j, and for each job type there is 1 unit of jobs of this type. An assignment of jobs to machines is now a vector $x_{ij} \geq 0$ so that $x_{ij} = 0$ when $i \notin S_j$, and $\sum_{i \in S} x_{ij} = 1$ for each job type j.

The load of a server i is now defined as $L_i = \sum_{j \in J} x_{ij}$. Finally, assume each server has a load dependent response time. The response time of server i is $r_i(x)$ if the load is x, and assume that $r_i(x)$ is a continuous, and monotone increasing function of x for all servers i. We define a Nash equilibrium to be a solution where, if a job type j is assigned to a server, than no other server jobs of type j can be assigned to, have smaller response time. (Formally, $x_{ij} > 0$, and $k \in S_j$ implies that $r_i(L_i) \leq r_k(L_k)$.)

(a) Show that a Nash equilibrium always exists in this game.

(b) Consider the objective function of minimizing the maximum response time. Show that any Nash equilibrium minimizes this objective function over all assignments. (We will see in class that this is not true in the general routing game, e.g., see Braess's paradox.)

(c) Now consider the average response time objective function, which is $\sum_i L_i r_i(L_i)$, as the load of L_i jobs all experience the servers $r_i(L_i)$ response time. Assume for this part that $r_i(x) = x$ for all x and i. Show that any Nash equilibrium minimizes this objective function over all assignments. (Note that the routing example with two parallel links, discussed in class shows that this is not true with general response functions.)

(4) Consider the one commodity special case of the continuous routing game discussed in class where all traffic goes from a common source s to a common destination t. Again more formally, we are given a graph G, and the problem will be to route 1 unit of flow from s to t in G. Each edge has a delay function $d_e(x)$ which is the delay incurred by the flow along edge e if there is x units of flow on e. Assume **for all parts of this problem** that the load on each edge is a nonegative, linear and monotone increasing function of the load.

We think of flow as defined by a set of paths from s to t, with $f_P \geq 0$ the amount of flow carried from s to t along the path P. Now $f(e) = \sum_{\{P: e \in P\}} f_P$, and the delay along the path P is $d_P(f) = \sum_{e \in P} d_e(f(e))$. For this problem we define a flow f^* to be optimal if the longest paths that carries flow is as short as possible, and we define a flow to be *fair* if all flow is carried on equal length paths. (This definition assumes that users realize the existence of a better path only by seeing other users who use that path, and the length of path not carrying flow is not relevant for the definition.) We know from class (essentially by definition) that the Nash flow is fair. From the Braess paradox example, we also see that there can be a fair flow that is better than the flow at Nash equilibrium.

(a) Prove that the Nash flow is at most a factor of 4/3 worse than the optimal for the objective of minimizing the longest path carrying flow.

(b) Prove that there is an optimal flow for the above objective that is also fair.

(c) For this part consider a flow f^* that minimizes average delay, that is, minimizes $\sum_P f_P d_P(f)$. We know that this optimal flow may not be fair. We measure the unfairness of this flow by the ratio of the lengths of the longest and shortest (s,t) paths that carries flow. Prove that the unfairness of the flow f^* is at most 2.