Algorithmic Game Theory	**Problem Set 1**
CS 6840 Spring 2014	**Due Friday, February 14th**

The questions on this problem set are of varying difficulty. For full credit, you need to solve at least 4 of the 5 problems below. A full solution for each problem includes proving that your answer is correct. If you cannot solve a problem, write down how far you got, and why are you stuck.

You may discuss the questions with other students, but need to write down the solution yourself. Please acknowledge the students you discussed the questions with on your write-up. You may use any fact we proved in class without proving the proof or reference, and may read the relevant chapters of the book. However, you may **not use other published papers, or the Web to find your answer**.

Solutions can be submitted on CMS in pdf format (only). Please type your solution or write extremely neatly to make it easy to read. If your solution is complex, say more than about half a page, please include a 3-line summary to help us understand the argument.

We will post answers to questions on Piazza.

(1) Recall that a game is a potential game if there is a function $\Phi(s)$ of the vector of strategies taken by the players, such that for any strategy vector s, any player i, and any alternate strategy s'_i for this player $\Phi(s'_i, s_{-i}) - \Phi(s_i, s_{-i}) = u_i(s'_i, s_{-i}) - u_i(s_i, s_{-i})$. Here s_{-i} denotes the vector s with coordinate i missing, (s'_i, s_{-i}) is the vector s with coordinate i replaced by s'_i, and $u_i(s_i, s_{-i})$ is the utility of player i when the players strategies form the vector s. Show that a finite game is a potential game if and only if for any two players i and j, and for any pair of strategies s_i, s'_i and s_j, s'_j the following equation holds.

$$u_i(s'_i, s_{-i}) - u_i(s) + u_j(s'_i, s'_j, s_{-i-j}) - u_j(s'_i, s_{-i}) = u_j(s'_j, s_{-j}) - u_j(s) + u_i(s'_i, s'_j, s_{-i-j}) - u_i(s'_j, s_{-j})$$

where (s'_i, s'_j, s_{-i-j}) denotes the vector s with coordinates i and j replaced by s'_i and s'_j respectively.

(2) Consider a continuous version of the Tragedy of the Commons Game from lecture (a simple model of a resource allocation game) where n players are sharing bandwidth of c and each player is choosing his or her rate. More formally, each player i chooses an amount $x_i \geq 0$. The utility of each player depends on both x_i and the sum of bandwidth $X = \sum_j x_j$ used by all. Players prefer higher x_i, but lower total X, as getting close to the available total capacity hurts performance. The utility of player i when the vector of chosen strategies is x is $u_i(x) = x_i(c - \sum_j x_j)$ and the player's goal is to maximize utility. We have seen in class that this game has a symmetric Nash equilibrium, where each player uses strategy $x_i = \frac{1}{n+1}c$.

(a) Show that this Nash equilibrium is unique, that is, the game has no other equilibrium.
(b) Is this game a potential game? Either argue that it is not, or give a potential function.

(3) We showed in class that all congestion games are potential games. The goal of this problem is to show the opposite: that each finite potential game is equivalent to a congestion game, that is, for any finite game that is an exact potential game, there is a congestion game with the same number of players, each player having the same number of strategies, and a one-to-one correspondence between the strategies in the two game, so that the payoffs are equal for each player.

We will call a game a *team game* if for any strategy vector s all players get the same payoff, that is $u_1(s) = u_2(s) = \ldots = u_n(s)$ for all s. We will call a game a *dummy game* if the payoff of the player does not depend on his own strategy (only on the strategy of the others), that is for all i, all s and s'_i we have $u_i(s'_i, s_{-i}) = u_i(s)$.

(a) Show that all potential games are the sum of a dummy game and a team game, that is, for each potential game there is a dummy game with utility u_i^d and a team game with utility u^t and the same set of strategies, so that $u_i(s) = u_i^d(s) + u_i^t(s)$ for all players and strategy vectors.

(b) Show that all team games are equivalent to a congestion game. Hint: it is OK to have a really large set of congestible elements

(c) Show that dummy games where each player has exactly two strategies are equivalent to a congestion game.

(d) Show that all dummy games are equivalent to a congestion game.

(e) Show that all potential games are equivalent to a congestion game.

(5) A strategy s_i of a player i is ϵ-dominated by a different strategy s'_i if for all strategy profiles s_{-i} of the other players $u_i(s_i, s_{-i}) \leq u_i(s'_i, s_{-i}) - \epsilon$, that is s_i is at least an ϵ worse for i than s'_i no matter what the other payers do. Let s_i be an ϵ-dominated action of a player i.

(a) Show that if a player i uses the weighted majority algorithm discussed in class on Monday, February 3 to choose his/her strategies, that the probability $\pi(s_i)$ that he/she is playing strategy s_i goes to zero over time.

(b) Give an example of a game with a coarse correlated equilibrium, and an ϵ-dominated action of a player i, where player i is playing action s_i with positive probability.

(6) Recall the set-up for online regret-minimization from Lecture (Monday, February 3): there is a fixed set A of actions; each day $t = 1, \ldots, T$ you pick an action at $a^t \in A$ (possibly from a probability distribution) based only on information from previous days; and then a cost vector $c^t : A \to [0, 1]$ is unveiled. The goal is to design a (randomized) algorithm that, for every sequence of cost vectors, has small expected average regret. [Recall that the (average, per time-step) regret is the difference between your average cost $\frac{1}{T}\sum_{t=1}^{T} c^t(a^t)$ and the average cost of the best fixed action $\frac{1}{T}\min_{a\in A}\sum_{t=1}^{T} c^t(a)$.]

(a) The most natural algorithm is to pick the strategy each day that seemed best so far, that is at time t pick the strategy a^t that minimizes $\sum_{s=1}^{t-1} c^s(a)$. Show that the average regret of this algorithm can be $\Omega(1)$ as T goes to infinity. How large can you make the ratio between the average cost of the best fixed action, and the average cost of this algorithm?

(b) Let's consider the following randomized pre-processing step. For each action a, initialize the starting cumulative cost to a random variable, X_a. Let X_a be iid random variables, each distributed as the number of coin flips needed until you get heads, assuming that the probability of heads is ϵ, using independent experiments for each action a.

Then, every day t, you pick the action that minimizes the perturbed cumulative cost prior to that day: $-X_a + \sum_{s=0}^{t-1} c^s(a)$. We assume that the random variables X_a are independent from the costs c^t. Note that this selection can be implemented if the algorithm has access to the whole vector c^t after each day t.

Prove that, for each day t, with probability at least $1 - \epsilon$, the smallest perturbed cumulative cost, that is $\min_a -X_a + \sum_{s=0}^{t-1} c^s(a)$, is at least 1 less than the second-smallest item in this minimum.

(c) As a thought experiment, consider the (unimplementable) algorithm that, every day, picks the action that minimizes the perturbed cumulative cost $-X_a + \sum_{s=0}^{t} c^s(a)$, taking into account the current days cost vector. Prove that the average regret of this algorithm is at most $\max_a X_a/T$.

(d) Prove that $E[\max_a X_a] = O(\epsilon^{-1} \log n)$, where n is the number of actions.

(e) Use the previous parts to prove that, for a suitable choice of ϵ, the algorithm in (b) has expected average regret $O(\sqrt{\frac{\log n}{T}})$, just like the multiplicative weights algorithm covered in class. (Make any assumptions you want about how ties between actions are broken.)