601.436/636 Algorithmic Game Theory **Spring 2020**
Homework #1 **Due:** February 13, 2020, 3pm

Remember: you may work in groups of up to three people, but must write up your solution entirely on your own. Collaboration is limited to discussing the problems – you may not look at, compare, reuse, etc. any text from anyone else in the class. Please include your list of collaborators on the first page of your submission. You may use the internet to look up formulas, definitions, etc., but may not simply look up the answers online.

Please include proofs with all of your answers, unless stated otherwise. Your solution must by typeset (*not* handwritten), and must be submitted by gradescope.

---

1 Three-Player Zero-Sum Games (50 points)

(Exercise 13 from the book). We saw in class that there is a polynomial time algorithm to find a Nash equilibrium in any two-player zero-sum game, and discussed that it is much more difficult to compute a Nash of a general (non zero-sum) two-player game. What about three player zero-sum games, i.e., games with three players such that the sum of the three utilities is always zero? Prove that finding a Nash equilibrium (mixed or pure) in such a game is at least as hard as finding a Nash equilibrium in a general two-player game.

2 Pure Nash of Tree Games (50 points)

(Exercise 1.4 from the book). Consider an n player game in which each player has only two (pure) strategies. This game has 2^n possible outcomes (for the 2^n ways the n players can play), therefore the game in matrix form is exponentially large. To circumvent this, we can consider a special class of games called *graphical games* (these appear in detail in the Chapter 7 of the book, but in this class we will likely not spend too much more time on them). The idea is that the utility of a player can depend only on a *subset* of other players. We will define a dependence graph G, whose nodes are the players, and an edge between two players i and j represents the fact that the utility of player i depends on the strategy of player j or vice versa (this is an undirected graph – an edge implies a dependence in either one or both directions). Thus, if node i has k neighbors, then its payoff depends only on its own strategy and the strategies of its k neighbors.

Consider a game where the players have 2 pure strategies each and assume that the graph G is a full binary rooted tree (i.e., there is a root r with 2 children, every node has either 0 or 2 children, and G is a tree). Give a polynomial time algorithm to decide if such a game has a pure Nash equilibrium (recall that there are 2^n possible pure strategy vectors, yet your algorithm must run in time polynomial in n). Prove correctness and polynomial running time.