601.436/636 Algorithmic Game Theory **Spring 2022**
Homework #1 **Due:** February 10, 2022, 4:30pm

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 1.3 from NRTV). 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.

Solution: Consider a general-sum two player game, with strategy sets S_1 and S_2 and utility functions $u_1 : S_1 \times S_2 \to \mathbb{R}$ and $u_2 : S_1 \times S_2 \to \mathbb{R}$. We will define a three-player zero-sum game in which every Nash equilibrium can be (in polynomial time) transformed into a Nash equilibrium of our original two-player game. Thus if we can find a Nash in our three-player zero-sum game then we can find a Nash in our two-player game, and hence finding a Nash in a three-player zero-sum game is at least as hard as finding a Nash in a two-player general-sum game.

Given the above two-player general-sum game, we define our three-player zero-sum game as follows. The strategy sets will be $S'_1 = S_1, S'_2 = S_2$, and $S'_3 = [1]$ (so player 3 only has one strategy). The utility functions will be

$$\begin{aligned} u'_1(i,j,1) &= u_1(i,j) \\ u'_2(i,j,1) &= u_2(i,j) \\ u'_3(i,j,1) &= -(u_1(i,j) + u_2(i,j)) \end{aligned}$$

Obviously this is zero-sum by construction.

Let $(\boldsymbol{x}, \boldsymbol{y}, 1)$ be a Nash equilibrium of the three-player game. Then we claim that $(\boldsymbol{x}, \boldsymbol{y})$ is a Nash equilibrium of the two-player game. To see this, for player 1, consider some other mixed strategy $\boldsymbol{x'}$. Then we get that

$$\begin{aligned} \mathbf{E}_{a_1 \sim \boldsymbol{x'}, a_2 \sim \boldsymbol{y}}[u_1(a_1, a_2)] &= \mathbf{E}_{a_1 \sim \boldsymbol{x'}, a_2 \sim \boldsymbol{y}}[u'_1(a_1, a_2, 1)] \\ &\le \mathbf{E}_{a_1 \sim \boldsymbol{x}, a_2 \sim \boldsymbol{y}}[u'_1(a_1, a_2, 1)] \\ &= \mathbf{E}_{a_1 \sim \boldsymbol{x}, a_2 \sim \boldsymbol{y}}[u_1(a_1, a_2)] \end{aligned}$$

where the equalities are by the definition of u'_1 and the inequalities are by the fact that $(\boldsymbol{x}, \boldsymbol{y}, 1)$ is a Nash of the three-player game. Thus in the two-player game, player 1 has no incentive to deviate. A similar argument works for player 2, and thus $(\boldsymbol{x}, \boldsymbol{y})$ is a Nash equilibrium of the two-player game.

2 Pure Nash of Tree Games (50 points)

(Exercise 1.4 from NRTV). 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.

Solution: To fix notation, we will let the utility function of node v be $u_v(a, b, c, d)$ where v plays a, the left child of v plays b, the right child of v plays c, and the parent of v plays d (if any of these nodes do not exist, i.e., v is the root or a leaf, then we can just skip the argument in the function).

We'll design a dynamic programming algorithm. Consider the following subproblems. For the root r, let

$$T(r, 0) = \begin{cases} 1 & \text{if there is a pure Nash where } r \text{ plays } 0 \\ 0 & \text{otherwise} \end{cases}$$

$$T(r, 1) = \begin{cases} 1 & \text{if there is a pure Nash where } r \text{ plays } 1 \\ 0 & \text{otherwise} \end{cases}$$

For any non-root node v, let $p(v)$ denote the parent of v. Let T_v denote the subtree rooted at v, let M_v^0 be the game restricted to T_v but where for the purpose of the utility of v we hardwire $p(v)$ to play 0, and let M_v^1 be the game restricted to T_v but where for the purpose of the utility of v we hardwire $p(v)$ to play 1. For each $i, j \in \{0, 1\}$, let

$$T(v, i, j) = \begin{cases} 1 & \text{if there is a pure Nash of } M_v^j \text{ where } v \text{ plays } i \\ 0 & \text{otherwise} \end{cases}$$

For any node v, let $\ell(v)$ denote its left child and let $r(v)$ denote its right child (if v is a leaf then both of these are empty). We first claim that for $i \in \{0,1\}$,

$$T(v,i,j) = \max_{a,b \in \{0,1\}} \{\min\{T(\ell(v),a,i), T(r(v),b,i)\} : u_v(i,a,b,j) \geq u_v(i,a',b',j) \; \forall a',b' \in \{0,1\}\}.$$

Put into words, this is the claim that there is a Nash of M_v^j where v plays i if and only if there is an action a for the left child of v and an action b for the right child of v such that there is a pure Nash for $M_{\ell(v)}^i$ where $\ell(v)$ plays a and there is a pure Nash for $M_{r(v)}^i$ where $r(v)$ plays b and v itself has no incentive to deviate when its children play these actions (and its parent plays j).

This is essentially obvious, but can be proved easily enough. Clearly if a Nash of M_v^j exists where v plays i then such an a and b exist (these are the actions played by the children at this Nash). Similarly, if such a and b exist then no descendent of v has incentive to deviate (by induction), and v has no incentive to deviate by the restriction on a and b.

Given this relation for $T(v,i,j)$, the total number of subproblems is at most $4n$ and the time it takes to compute the entry for a single subproblem is $O(1)$. Hence the total running time to fill in the table for T is $O(n)$.

Once the table is filled in, we can just return YES is either $T(r,0) = 1$ or if $T(r,1) = 1$. By the definition of the T function, this gives the correct solution.