CS 6840 Algorithmic Game Theory November 25, 2024

Lecture 27: Equilibria in 0-sum extensive form games

Instructor: Eva Tardos *Scribe: Surendra Ghentiyala*

1 Recap

Definition 1 (Extensive form games)**.** Extensive form games are specified with a game tree. Each node of the tree is associated with a player or a random choice. Leaf nodes are labeled with payoffs. The game proceeds as follows. A pebble starts at the root node. When the pebble is at node v, the player who owns node v gets to decide to which child of v to move the pebble. At a "random choice" node v, the pebble is moved to a random child of v. The game ends when the pebble reaches a leaf node l and the players receive the payoffs associated with l.

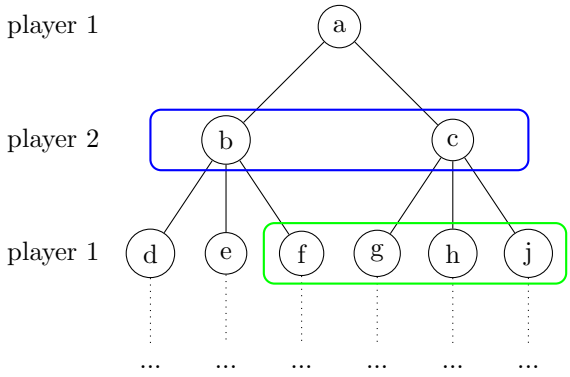

Figure 1: A sample game tree and information sets

We say that a game is full information if a player knows which node they are in at any give point. Most games are not full information. For example, if the second and third move in a game are made simultaneously, we think of the moves being made sequentially but we do not allow the move of either player to depend on the move of the other player. To model this, we introduce the notion of information sets. An information set is simply a set of nodes associated with a player i. For every information set $\mathcal{I}$ associated with player i, player i will never know which node of $\mathcal{I}$ they are in, they will only know that they are in one of the nodes of $\mathcal{I}$. Therefore, for every player i and information set $\mathcal{I}$ for i, we impose the constraint that player i must play the same strategy at every $v \in \mathcal{I}$.

For example, in Figure 1, the blue set and green set are information sets. We can think of the first two moves of the associated game as being made simultaneously since the blue set imposes the constraint that player 2's first move does not depend on player 1's first move.

We assume perfect recall, which means that every player knows which moves they have played to this point. So for example, the green set in Figure 1 is an invalid information set in a perfect recall game since player 1 remembers whether they took the left or right path in their first move, and thus should be allowed to make different decisions on f vs. g, h, j.

2 Strategy formulation

There are several different ways to write a strategy for an extensive form game. One is to write it as a combination of pure strategies. But notice that if we let k be the number of options at each node in the tree and n be the number of decision nodes for player i, then the number of pure strategies for player i is only bounded above by k^n, which is exponential in the size of the game tree. We therefore propose the following, better, description of a randomized strategy.

Definition 2 (The x-formulation)**.** For each node v, we give a probability distribution on choices. In particular, we let $x_v^i(j)$ denote the probability that player i chooses option j at node v. Furthermore, to impose the aforementioned information set restriction, we introduce the constraints that for every $\mathcal{I}$ belonging to player i, for every $v \in \mathcal{I}$, $x_v^i(j)$ is the same.

Notice that the size of a strategy in the x-formulation is at most $n \cdot k$, where n is the number of nodes and k is the number of choices at each node. The x-formulation gives us an efficient way to represent strategies since $n \cdot k$ is polynomial in the size of the game tree.

However, in the x-formulation, the probability of reaching leaf l is the product of some x variables and this leads to a highly non-linear system.

We note that such formulations only lead to strategy formulations which are polynomial in the size of the game tree, not necessarily polynomial in the size of the input. There exist some games where the most natural input is a rule-set which implicitly defines an exponentially sized game tree. In these cases, our strategy formulation is not polynomial in the input size. But we restrict our attention to cases where the input is an explicit game tree.

3 An unintuitive strategy formulation

We now modify the above formulation to give an unintuitive, yet equivalent formulation for extensive form game strategies.

Definition 3 (The z-formulation)**.** We define new variables $z(v)$ for every node v. For a node v and player i, $z_i(v)$ is the probability that player i makes the correct moves to make it to node v (assuming that all other players made moves which allow player i to make the necessary decisions to arrive at v). Formally, $z_i(v)$ is the product of probabilities on the path from the root of the tree to node v.

Let v be a node with children $w_1, \ldots, w_m$ where decision j on node v leads to w_j. Notice that the following constraints must apply to z_i.

1. $z_i(v) \geq 0$.
2. For all i, $z_i(\text{root}) = 1$. The probability that player i makes the correct moves to arrive at the root is 1 since all players start at the root by default.
3. If v is not a node belonging to player i, then for all $j \in [m]$, $z_i(v) = z_i(w_j)$. This follows immediately from the fact that $z_i(v)$ is the product of probabilities on the path from the root of the tree to node v and i does not own v.
4. If v is a node belonging to i, then
$$\sum_{j \in [m]} z_i(w_j) = z_i(v) \ .$$

Proof. By the definition of z, we have that for all $j \in [m]$, $z_i(w_j) = z_i(v) \cdot x_v^i(j)$. Summing up over all j we get

$$\sum_{j\in[m]} z_i(w_j) = \sum_{j\in[m]} z_i(v) \cdot x_v^i(j) = z_i(v) \sum_{j\in[m]} x_v^i(j) = z_i(v)$$

where the last equality follows from the fact that $\sum_{j\in[m]} x_v^i(j) = 1$ because player i must go to one of the children of v from v. ■

5. Say v and v' are nodes for player i which are in the same information set. If the same move allows player i to move from v to w_j and from v' to w'_j, then $z_i(w_j) = z_i(w'_j)$.

We now state the key insight that lets us work with the z-formulation instead of the x-formulation.

Theorem 1. *$x_v^i(j)$ and $z_i(v)$ are in one-to-one correspondence.*

For every x-formulation there exists a unique z-formulation, and vice versa. This is not too hard to see since there exist natural algorithms to transform z-formulations to x-formulations and vice versa.

4 Zero-sum games

We now show that the z-formulation of a 2-person zero-sum game allows us to solve for Nash equilibrium in polynomial time.

The strategy z_1 (in the z-formulation) is a Nash equilibrium with expected payoff λ if 1) it satisfies all above constraints on z and 2) given z_1, the best response Γ for player 2 played against z_1 results in a payoff of at least λ. Since this is a zero-sum game, this formulation of Nash equilibria naturally leads to a max-min problem to compute Nash equilibria, player 1 is maximizing payoff over z while player two then minimizes over Γ. This is very close to the linear programming formulation we used to solve zero sum games in a previous lecture (since all constraints on z are linear), but Γ gives us some trouble.

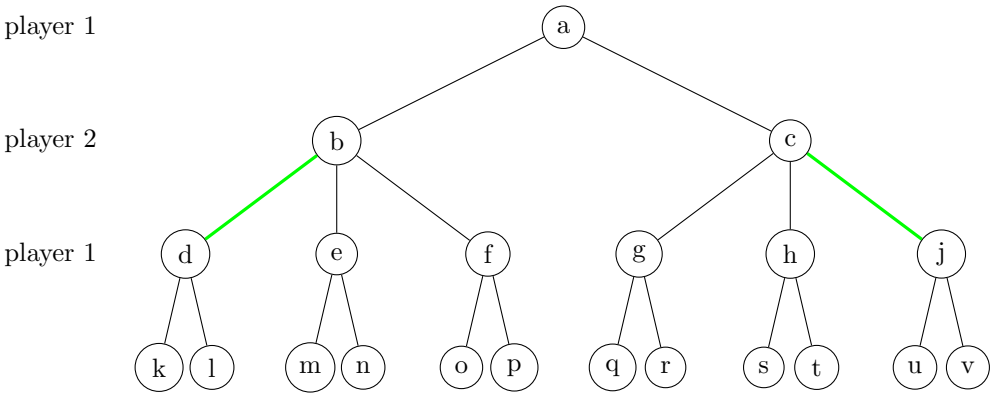

Figure 2: A game tree where the Γ is highlighted green

Let us consider Γ, the best response for player 2 given that player 1 plays z_1. Notice that many leaves will be unreachable under Γ. For example, in Figure 2 node m is unreachable under Γ. Let $a_1(v)$ denote

the payoff that player 1 receives upon reaching leaf node v. We see that the payoff that player 1 receives playing z_1 against player 2's best response Γ is

$$\sum_{v \text{ reachable}} z_1(v)a_1(v),$$

which is polynomial-time computable.

However, we still have too many Γ to quantify over, there are an exponential number of pure strategies Γ that player 2 could play. Fortunately, it was shown in the 1980s that linear programming techniques (ellipsoid method) can be used to solve such problems.

Fact: If given a proposed z, we can find

$$\arg\min_{\Gamma} \sum_{v \text{ reachable}} z_1(v)a_1(v),$$

then we can find a Nash equilibrium for zero-sum extensive form games.

Can we find such a Γ given $z_1(v)$? We need a subroutine to compute a best response for player 2 given player 1's strategy. This turns out to be easy. We can use a similar approach to the one we saw in the last lecture to solve perfect information games. In particular, we can find Γ bottom-up by choosing Γ at each level to be the strategy (compatible with information set constraints for player 2) which maximizes the expected payoff for player 2 given that we know what will be played at all lower levels.