An Introduction to Game Trees

Ross Casebolt — Graph Theory

Willamette University

April 26, 2012
What is a game?

Definition (Game)

A *game* $G$ must satisfy the following properties:

- There is a finite set of *players*, $\{P_i\}$

Also, a game may or may not include chance moves such as rolling dice. However, not ones that we will consider.
What is a game?

Definition (Game)

A game $G$ must satisfy the following properties:

- There is a finite set of players, $\{P_i\}$
- Each player has full knowledge of the rules
What is a game?

Definition (Game)

A game \( G \) must satisfy the following properties:

- There is a finite set of players, \( \{ P_i \} \)
- Each player has full knowledge of the rules
- At each point in the game, a player has only finitely many moves (or none)

Also, a game may or may not include chance moves such as rolling dice. However, not ones that we will consider.
What is a game?

**Definition (Game)**

A *game* $G$ must satisfy the following properties:

- There is a finite set of *players*, $\{P_i\}$
- Each player has full knowledge of the rules
- At each point in the game, a player has only finitely many moves (or none)
- The game ends after finitely many moves

Also, a game may or may not include chance moves such as rolling dice. However, not ones that we will consider.
What is a game?

Definition (Game)

A game $G$ must satisfy the following properties:

- There is a finite set of players, $\{P_i\}$
- Each player has full knowledge of the rules
- At each point in the game, a player has only finitely many moves (or none)
- The game ends after finitely many moves
- At the end of the game, each player receives a numerical payoff

Also, a game may or may not include chance moves such as rolling dice. However, not ones that we will consider.
What is a game?

Definition (Game)

A game $G$ must satisfy the following properties:

- There is a finite set of players, $\{P_i\}$
- Each player has full knowledge of the rules
- At each point in the game, a player has only finitely many moves (or none)
- The game ends after finitely many moves
- At the end of the game, each player receives a numerical payoff

Also, a game may or may not include chance moves such as rolling dice. However, not ones that we will consider.
Definition (Directed Tree)

A directed graph $T$ is a tree if it has a distinct vertex $r$, called the root, such that $r$ has no edges going into it and such that for other vertex $v$ of $T$ there is a unique directed path from $r$ to $v$.
Definition (Directed Tree)
A directed graph $T$ is a tree if it has a distinct vertex $r$, called the root, such that $r$ has no edges going into it and such that for other vertex $v$ of $T$ there is a unique directed path from $r$ to $v$.

Definition (Families)
Let $T$ be a tree. A vertex $v$ is a child of a parent vertex $u$ if $(u, v)$ is an edge. A vertex $v$ is a descendant of an ancestor vertex $u$ if there is a directed path from $u$ to $v$. 
**Definition (Directed Tree)**

A directed graph $T$ is a tree if it has a distinct vertex $r$, called the root, such that $r$ has no edges going into it and such that for other vertex $v$ of $T$ there is a unique directed path from $r$ to $v$.

**Definition (Families)**

Let $T$ be a tree. A vertex $v$ is a child of a parent vertex $u$ if $(u, v)$ is an edge. A vertex $v$ is a descendant of an ancestor vertex $u$ if there is a directed path from $u$ to $v$.

Also, denote the subtree of $T$ restricted to only descendants of $u$ by $T_u$. 
Time for an example:

Example (Caterpillars vs Lobsters)

Each player has two cards, a 1 and 2. Player 1, or $P_1$, starts by playing a card, and the turns alternate until each player has no cards in their hand. A player gets 1 point if they play a card with higher value on top of a card with lower value, and the player with the most points wins.
Note: if $T$ is a directed tree and $G$ is a game with no chance moves, we can assign vertices uniquely to each player, similar to a coloring. Then $T$ is called a game tree, and we say that a player $P_i$ owns a vertex if it was assigned to them. Also, call $\pi_i(c_1, c_2, \ldots, c_n)$ the payoff for player $P_i$ by following a path uniquely determined by the players’ choice functions $c_i$. 

**Definition (Information Set)**

An information set $S$ for a player $P$ is a set of vertices, all belonging to $P$, such that at a specific point in the game, $P$ knows they are at one of the vertices in $S$ but not which one exactly.

**Definition (Choice Function)**

Let $T$ be a game tree, and $P$ be a player. Define a choice function for $P$ to be a function $c_i$ from the set of vertices of $T$ belonging to $P$, such that $c_i(u)$ is a child of $u$ for every vertex $u$ belonging to $P$. 
Note: if $T$ is a directed tree and $G$ is a game with no chance moves, we can assign vertices uniquely to each player, similar to a coloring. Then $T$ is called a *game tree*, and we say that a player $P_i$ *owns* a vertex if it was assigned to them. Also, call $\pi_i(c_1, c_2, ..., c_n)$ the payoff for player $P_i$ by following a path uniquely determined by the players’ choice functions $c_i$.

**Definition (Information Set)**

An *information set* $S$ for a player $P$ is a set of vertices, all belonging to $P$, such that at a specific point in the game, $P$ knows they are at one of the vertices in $S$ but not which one exactly.
Definitions, Definitions Everywhere!

Note: if $T$ is a directed tree and $G$ is a game with no chance moves, we can assign vertices uniquely to each player, similar to a coloring. Then $T$ is called a game tree, and we say that a player $P_i$ owns a vertex if it was assigned to them. Also, call $\pi_i(c_1, c_2, \ldots, c_n)$ the payoff for player $P_i$ by following a path uniquely determined by the players’ choice functions $c_i$.

Definition (Information Set)

An information set $S$ for a player $P$ is a set of vertices, all belonging to $P$, such that at a specific point in the game, $P$ knows they are at one of the vertices in $S$ but not which one exactly.

Definition (Choice Function)

Let $T$ be a game tree, and $P$ be a player. Define a choice function for $P$ to be a function $c$, from the set of vertices of $T$ belonging to $P$, such that $c(u)$ is a child of $u$ for every vertex $u$ belonging to $P$. 
Definition (Choice Subtree)

Let $T$ be a game tree, $P$ a player, and $c$ a choice function of $P$. Then the choice subtree determined by $P$ and $c$ is defined to be the union of all the $(P, c)$ paths, where a $(P, c)$ path is a path from the root of $T$ to a leaf in such a way that if $u$ is a vertex on the path belonging to $P$, then $(u, c(u))$ is an edge of $(P, c)$. 

Definition (Extensive Form)

Let $T$ be a game tree with players $P_1, P_2, \ldots, P_n$. A game in extensive form based on $T$ consists of $T$ together with a non-empty set $\Sigma_i$ of choice subtrees for each player $P_i$. The set $\Sigma_i$ is called the strategy set for $P_i$, and an element of $\Sigma_i$ is called a strategy for $P_i$. 

Ross Casebolt — Graph Theory

Lobsters vs Caterpillars!
Definition (Choice Subtree)
Let $T$ be a game tree, $P$ a player, and $c$ a choice function of $P$. Then the choice subtree determined by $P$ and $c$ is defined to be the union of all the $(P, c)$ paths, where a $(P, c)$ path is a path from the root of $T$ to a leaf in such a way that if $u$ is a vertex on the path belonging to $P$, then $(u, c(u))$ is an edge of $(P, c)$.

Definition (Extensive Form)
Let $T$ be a game tree with players $P_1, P_2, \ldots, P_n$. A game in extensive form based on $T$ consists of $T$ together with a non-empty set $\Sigma_i$ of choice subtrees for each player $P_i$. The set $\Sigma_i$ is called the strategy set for $P_i$, and an element of $\Sigma_i$ is called a strategy for $P_i$. 
Definition (Equilibrium)

Let $\Gamma$ be an $n$-player game in extensive form and denote the player’s strategy sets by $\Sigma_1, \Sigma_2, \ldots, \Sigma_n$. An $n$-tuple $(S_i)$, where $S_i \in \Sigma_i$, is an equilibrium $n$-tuple if for every $i$ and $S_i^\alpha \in \Sigma_i$, 

$$\pi_i(S_1, \ldots, S_i^\alpha, \ldots, S_n) \leq \pi_i(S_1, \ldots, S_i, \ldots, S_n).$$

Theorem (The Big One)

Let $\Gamma$ be a game in extensive form. If $\Gamma$ is of perfect information, then there exists an equilibrium $N$-tuple of strategies.
Definition (Equilibrium)

Let $\Gamma$ be an $n$-player game in extensive form and denote the player’s strategy sets by $\Sigma_1, \Sigma_2, \ldots, \Sigma_n$. An $n$-tuple $(S_i)$, where $S_i \in \Sigma_i$, is an equilibrium $n$-tuple if for every $i$ and $S_i^{\alpha} \in \Sigma_i$, $\pi_i(S_1, \ldots, S_i^{\alpha}, \ldots, S_n) \leq \pi_i(S_1, \ldots, S_i, \ldots, S_n)$.

Theorem (The Big One)

Let $\Gamma$ be a game in extensive form. If $\Gamma$ is of perfect information, then there exists an equilibrium $N$-tuple of strategies.
Suppose $T$ is game tree with depth 1. Then only one player has a move. Assuming that the root belongs to player $P$, let the strategy for $P$ be to choose a (terminal) child of the root for which $P$’s payoff is a maximum. The corresponding subtree for $P$ then consists of a single RLTG. Let the choice subtrees for the other players be all of $T$ (in fact, no other possibilities exist). Then the resulting $n$-tuple is in equilibrium.
Now suppose that the depth of $T$ is $m > 1$ and that the theorem holds for all games of perfect information for which the depth of the game tree is less than $m$.

Let $r$ be the root of $T$. For each child $u$ of $r$, $T_u$ is a game tree and $\text{Depth}(T_u) < \text{Depth}(T)$. Now, we can consider each $T_u$ as the game tree for a game of perfect information by defining each player’s strategy set to be the set of all choice subtrees determined by that player. By the inductive assumption, there is an equilibrium $n$-tuple $(S_1^u, S_2^u, ..., S_n^u)$ of strategies in $T_u$. 
We want to use these equilibrium solutions for the $T_u$s to form one for $T$. If the root of $T$ belongs to a player $P_j$, choose $u$ to be a child of the root such that the payoff $\pi_j(S_1^u, S_2^u, \ldots, S_n^u)$ is a maximum over the children of $r$.

Now define $S_j^\alpha$ to be the union of $S_j^u$ and the edge $(r, u)$. For $1 \leq i \leq n$ and $i \neq j$, define $S_i^\alpha = \bigcup_{v \in \text{Ch}(r)} ((r, v) \cup S_i^v)$. 

We have now defined an $n$-tuple of choice subtrees. Since the game is of perfect information, all these choice subtrees are strategies. Now, we must show that this $n$-tuple is in equilibrium. Then there are two cases. But first, we’re going to need this theorem,

**Theorem (The Payoff Theorem)**

Let $T$ be a game tree with players $P_1, P_2, \ldots, P_n$. Let $S_1, \ldots, S_n$ be choice subtrees for $P_1, \ldots, P_n$, respectively. Then, with $r$ as the root of $T$, we have:

If $r$ belongs to player $P_i$ and $(r, u)$ is in $S_i$ then, for $1 \leq j \leq n$,

$$\pi_j(S_1, S_2, \ldots, S_n) = \pi_j(S_1 \cap T_u, S_2 \cap T_u, \ldots, S_n \cap T_u).$$
Case #1.
Case #1. Suppose the root belongs to player $P_i$. Now if $(r, w)$ is in $S_i$ we can apply the Payoff Theorem to get
\[
\pi_i(S_1^\alpha, ..., S, ..., S_n^\alpha) = \pi_i(S_1^\alpha \cap T_w, ..., S_i^\alpha \cap T_w, ..., S_n^\alpha \cap T_w).
\]
Case #1. Suppose the root belongs to player $P_i$. Now if $(r, w)$ is in $S_i$ we can apply the Payoff Theorem to get

$$\pi_i(S_1^\alpha, \ldots, S_i, \ldots, S_n^\alpha) = \pi_i(S_1^\alpha \cap T_w, \ldots, S_i^\alpha \cap T_w, \ldots, S_n^\alpha \cap T_w).$$

Then because $(S_1^w, \ldots, S_n^w)$ is an equilibrium $n$-tuple in $T_w$, we have

$$\pi_i(S_1^\alpha \cap T_w, \ldots, S_i^\alpha \cap T_w, \ldots, S_n^\alpha \cap T_w) \leq \pi_i(S_1^w, \ldots, S_n^w).$$
Case #1. Suppose the root belongs to player $P_i$. Now if $(r, w)$ is in $S_i$ we can apply the Payoff Theorem to get

$$\pi_i(S_1^\alpha, ..., S_i, ..., S_n^\alpha) = \pi_i(S_1^\alpha \cap T_w, ..., S_i^\alpha \cap T_w, ..., S_n^\alpha \cap T_w).$$

Then because $(S_1^w, ..., S_n^w)$ is an equilibrium $n$-tuple in $T_w$, we have

$$\pi_i(S_1^\alpha \cap T_w, ..., S_i^\alpha \cap T_w, ..., S_n^\alpha \cap T_w) \leq \pi_i(S_1^w, ..., S_n^w).$$

Now, if the edge $(r, u)$ belongs to $S_i^\alpha$, we know by the way $S_i^\alpha$ was chosen that

$$\pi_i(S_1^w, ..., S_n^w) \leq \pi_i(S_1^u, ..., S_n^u).$$
Case #1. Suppose the root belongs to player $P_i$. Now if $(r, w)$ is in $S_i$ we can apply the Payoff Theorem to get

$$
\pi_i(S_1^\alpha, ..., S_i, ..., S_n^\alpha) = \pi_i(\bigcap S_1^\alpha \cap T_w, ..., \bigcap S_i^\alpha \cap T_w, ..., \bigcap S_n^\alpha \cap T_w).
$$

Then because $(S_1^w, ..., S_n^w)$ is an equilibrium $n$-tuple in $T_w$, we have

$$
\pi_i(\bigcap S_1^\alpha \cap T_w, ..., \bigcap S_i^\alpha \cap T_w, ..., \bigcap S_n^\alpha \cap T_w) \leq \pi_i(S_1^w, ..., S_n^w).
$$

Now, if the edge $(r, u)$ belongs to $S_i^\alpha$, we know by the way $S_i^\alpha$ was chosen that $\pi_i(S_1^w, ..., S_n^w) \leq \pi_i(S_1^u, ..., S_n^u)$. Then by definition of $S_i^\alpha$'s, we have

$$
\pi_i(S_1^\alpha \cap T_u, ..., S_i^\alpha \cap T_u, ..., S_n^\alpha \cap T_u) = \pi_i(S_1^u, ..., S_n^u).
$$
Case #1. Suppose the root belongs to player $P_i$. Now if $(r, w)$ is in $S_i$ we can apply the Payoff Theorem to get

$$\pi_i(S_1^\alpha, ..., S_i, ..., S_n^\alpha) = \pi_i(S_1^\alpha \cap T_w, ..., S_i^\alpha \cap T_w, ..., S_n^\alpha \cap T_w).$$

Then because $(S_1^w, ..., S_n^w)$ is an equilibrium $n$-tuple in $T_w$, we have

$$\pi_i(S_1^\alpha \cap T_w, ..., S_i^\alpha \cap T_w, ..., S_n^\alpha \cap T_w) \leq \pi_i(S_1^w, ..., S_n^w).$$

Now, if the edge $(r, u)$ belongs to $S_i^\alpha$, we know by the way $S_i^\alpha$ was chosen that $\pi_i(S_1^w, ..., S_n^w) \leq \pi_i(S_1^u, ..., S_n^u)$. Then by definition of $S_i^\alpha$'s, we have

$$\pi_i(S_1^\alpha \cap T_u, ..., S_i^\alpha \cap T_u, ..., S_n^\alpha \cap T_u) = \pi_i(S_1^u, ..., S_n^u).$$

By the Payoff Theorem again, we get

$$\pi_i(S_1^\alpha, ..., S_i, ..., S_n^\alpha) = \pi_i(S_1^\alpha \cap T_u, ..., S_i^\alpha \cap T_u, ..., S_n^\alpha \cap T_u).$$
Case #1. Suppose the root belongs to player $P_i$. Now if $(r, w)$ is in $S_i$ we can apply the Payoff Theorem to get

$$\pi_i(S_1^\alpha, ..., S_i, ..., S_n^\alpha) = \pi_i(S_1^\alpha \cap T_w, ..., S_i^\alpha \cap T_w, ..., S_n^\alpha \cap T_w).$$

Then because $(S_1^w, ..., S_n^w)$ is an equilibrium $n$-tuple in $T_w$, we have

$$\pi_i(S_1^\alpha \cap T_w, ..., S_i^\alpha \cap T_w, ..., S_n^\alpha \cap T_w) \leq \pi_i(S_1^w, ..., S_n^w).$$

Now, if the edge $(r, u)$ belongs to $S_i^\alpha$, we know by the way $S_i^\alpha$ was chosen that $\pi_i(S_1^w, ..., S_n^w) \leq \pi_i(S_1^u, ..., S_n^u)$. Then by definition of $S_i^\alpha$’s, we have

$$\pi_i(S_1^\alpha \cap T_u, ..., S_i^\alpha \cap T_u, ..., S_n^\alpha \cap T_u) = \pi_i(S_1^u, ..., S_n^u).$$

By the Payoff Theorem again, we get

$$\pi_i(S_1^\alpha, ..., S_i, ..., S_n^\alpha) = \pi_i(S_1^\alpha \cap T_u, ..., S_i^\alpha \cap T_u, ..., S_n^\alpha \cap T_u)$$

and so combining everything we get

$$\pi_i(S_1^\alpha, ..., S_i, ..., S_n^\alpha) \leq \pi_i(S_1^\alpha, ..., S_n^\alpha).$$
The Final Induction!

Case #2.

Suppose that the root belongs to a player $P_j$ different from $P_i$. Let $u$ be that child of the root so that $(r, u)$ is in $S_{\alpha_j}$. Then we have 

$$\pi_i(S_{\alpha_1}, \ldots, S_{\alpha_i}, \ldots, S_{\alpha_n}) = \pi_i(S_{\alpha_1} \cap T_u, \ldots, S_{\alpha_i} \cap T_u, \ldots, S_{\alpha_n} \cap T_u).$$

But then 

$$\pi_i(S_{\alpha_1} \cap T_u, \ldots, S_{\alpha_i} \cap T_u, \ldots, S_{\alpha_n} \cap T_u) \leq \pi_i(S_{u_1}, \ldots, S_{u_n}).$$

Finally, 

$$\pi_i(S_{\alpha_1}, \ldots, S_{\alpha_i}, \ldots, S_{\alpha_n}) = \pi_i(S_{u_1}, \ldots, S_{u_n}).$$

And once again, combining these yields the desired inequality.

— Neumann, Morgenstern, 1944.
Case #2. Suppose that the root belongs to a player $P_j$ different from $P_i$. Let $u$ be that child of the root so that $(r, u)$ is in $S_j^\alpha$. Then we have

$$
\pi_i(S_\alpha^1, \ldots, S_\alpha^i, \ldots, S_\alpha^n) = \pi_i(S_u^1, \ldots, S_u^n).
$$

But then

$$
\pi_i(S_\alpha^1 \cap T_u, \ldots, S_\alpha^i \cap T_u, \ldots, S_\alpha^n \cap T_u) \leq \pi_i(S_u^1, \ldots, S_u^n).
$$

Finally,

$$
\pi_i(S_\alpha^1, \ldots, S_\alpha^i, \ldots, S_\alpha^n) = \pi_i(S_u^1, \ldots, S_u^n).
$$

And once again, combining these yields the desired inequality.

— Neumann, Morgenstern, 1944.
Case #2. Suppose that the root belongs to a player $P_j$ different from $P_i$. Let $u$ be that child of the root so that $(r, u)$ is in $S_j^\alpha$. Then we have

$$\pi_i(S_1^\alpha, ..., S_i, ..., S_n^\alpha) = \pi_i(S_1^\alpha \cap T_u, ..., S_i^\alpha \cap T_u, ..., S_n^\alpha \cap T_u).$$
Case #2. Suppose that the root belongs to a player $P_j$ different from $P_i$. Let $u$ be that child of the root so that $(r, u)$ is in $S_j^\alpha$. Then we have

$$\pi_i(S_1^\alpha, \ldots, S_i, \ldots, S_n^\alpha) = \pi_i(S_1^\alpha \cap T_u, \ldots, S_i^\alpha \cap T_u, \ldots, S_n^\alpha \cap T_u).$$

But then

$$\pi_i(S_1^\alpha \cap T_u, \ldots, S_i^\alpha \cap T_u, \ldots, S_n^\alpha \cap T_u) \leq \pi_i(S_1^u, \ldots, S_n^u).$$
Case #2. Suppose that the root belongs to a player $P_j$ different from $P_i$. Let $u$ be that child of the root so that $(r, u)$ is in $S_j^\alpha$. Then we have

$$\pi_i(S_1^\alpha, ..., S_i, ..., S_n^\alpha) = \pi_i(S_1^\alpha \cap T_u, ..., S_i^\alpha \cap T_u, ..., S_n^\alpha \cap T_u).$$

But then

$$\pi_i(S_1^\alpha \cap T_u, ..., S_i^\alpha \cap T_u, ..., S_n^\alpha \cap T_u) \leq \pi_i(S_1^u, ..., S_n^u).$$

Finally,

$$\pi_i(S_1^\alpha, ..., S_i, ..., S_n^\alpha) = \pi_i(S_1^u, ..., S_n^u).$$
Case #2. Suppose that the root belongs to a player $P_j$ different from $P_i$. Let $u$ be that child of the root so that $(r, u)$ is in $S_j^\alpha$. Then we have

$$\pi_i(S_1^\alpha, \ldots, S_i, \ldots, S_n^\alpha) = \pi_i(S_1^\alpha \cap T_u, \ldots, S_i^\alpha \cap T_u, \ldots, S_n^\alpha \cap T_u).$$

But then

$$\pi_i(S_1^\alpha \cap T_u, \ldots, S_i^\alpha \cap T_u, \ldots, S_n^\alpha \cap T_u) \leq \pi_i(S_1^u, \ldots, S_n^u).$$

Finally,

$$\pi_i(S_1^\alpha, \ldots, S_i, \ldots, S_n^\alpha) = \pi_i(S_1^u, \ldots, S_n^u).$$

And once again, combining these yields the desired inequality.

— Neumann, Morgenstern, 1944.
Example (Alpha-Beta Pruning)
Searching a game tree for optimal strategies, and discarding moves when it’s been shown that a move’s subtree is worse than another’s.

Example (Monte-Carlo)
Using random playouts and sampling a number of them, then choosing the move with the best results in the sample.

Example (Quiescence Searches)
Searching down paths with the most complications (determined by some sort of “noise” function), and ignoring paths which are relatively simple.