Algorithmic game theory

Martin Balko

9th lecture

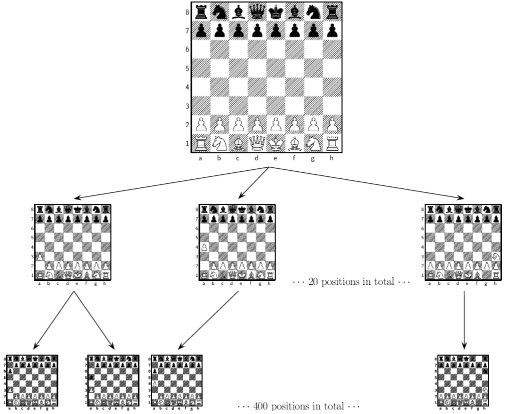
December 2nd 2025

Games in extensive form

Games in extensive form

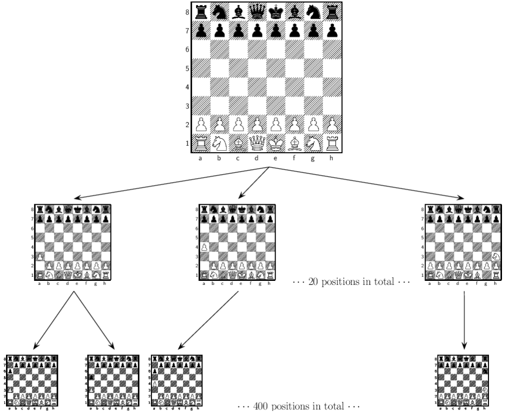
Games in extensive form

- Last lecture, we introduced a new notion of games, so-called games in extensive form, which are described using trees.



Games in extensive form

- Last lecture, we introduced a new notion of games, so-called games in extensive form, which are described using trees.



- Today, we describe strategies for such games and how to compute Nash equilibria.

Basic definitions

Basic definitions

- Extensive game consists of a directed tree where nodes represent game states.

Basic definitions

- Extensive game consists of a directed tree where nodes represent game states.
- The game starts at the root of the tree and ends at a leaf, where each player receives a payoff.

Basic definitions

- Extensive game consists of a directed tree where nodes represent game states.
- The game starts at the root of the tree and ends at a leaf, where each player receives a payoff. Each node that is not a leaf is a decision node.

Basic definitions

- Extensive game consists of a directed tree where nodes represent game states.
- The game starts at the root of the tree and ends at a leaf, where each player receives a payoff. Each node that is not a leaf is a decision node.
- Moves a player can make in a given state are assigned to the outgoing edges of the corresponding decision node.

Basic definitions

- Extensive game consists of a directed tree where nodes represent game states.
- The game starts at the root of the tree and ends at a leaf, where each player receives a payoff. Each node that is not a leaf is a decision node.
- Moves a player can make in a given state are assigned to the outgoing edges of the corresponding decision node.

- We partition decision nodes into information sets where all nodes in an information set belong to the same player and have the same moves.

Basic definitions

- Extensive game consists of a directed tree where nodes represent game states.
- The game starts at the root of the tree and ends at a leaf, where each player receives a payoff. Each node that is not a leaf is a decision node.
- Moves a player can make in a given state are assigned to the outgoing edges of the corresponding decision node.

- We partition decision nodes into information sets where all nodes in an information set belong to the same player and have the same moves.
- For player i, we let H_i be the set of information sets of i and, for an information set $h \in H_i$, we let C_h be the set of moves at h.

Basic definitions

- Extensive game consists of a directed tree where nodes represent game states.
- The game starts at the root of the tree and ends at a leaf, where each player receives a payoff. Each node that is not a leaf is a decision node.
- Moves a player can make in a given state are assigned to the outgoing edges of the corresponding decision node.

- We partition decision nodes into information sets where all nodes in an information set belong to the same player and have the same moves.
- For player i, we let H_i be the set of information sets of i and, for an information set $h \in H_i$, we let C_h be the set of moves at h.
- In perfect-information games all information sets are singletons.

Basic definitions

- Extensive game consists of a directed tree where nodes represent game states.
- The game starts at the root of the tree and ends at a leaf, where each player receives a payoff. Each node that is not a leaf is a decision node.
- Moves a player can make in a given state are assigned to the outgoing edges of the corresponding decision node.

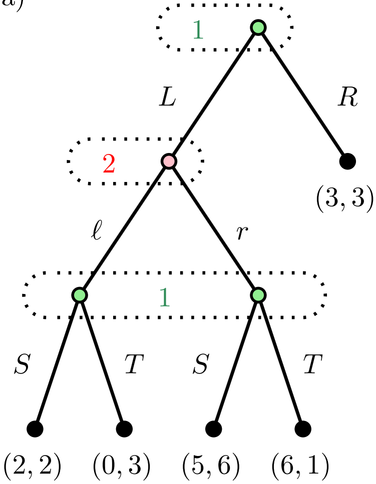
- We partition decision nodes into information sets where all nodes in an information set belong to the same player and have the same moves.
- For player i, we let H_i be the set of information sets of i and, for an information set $h \in H_i$, we let C_h be the set of moves at h.
- In perfect-information games all information sets are singletons. Otherwise, we have an imperfect-information game where players have only partial knowledge of the states that they are in.

Example: imperfect-information game

Example: imperfect-information game

- An example of an imperfect-information game in extensive form (part (a)) and its normal-form (part (b)).

(a)



(b)

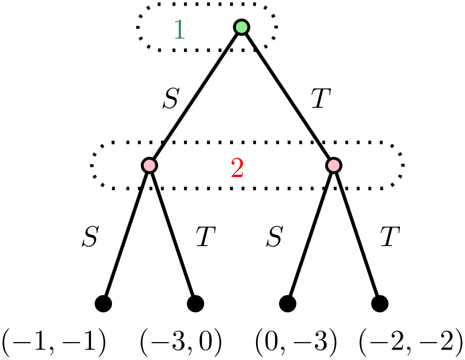
	(ℓ)	(r)
(L,S)	(2,2)	(5,6)
(L,T)	(0,3)	(6,1)
(R,S)	(3,3)	(3,3)
(R,T)	(3,3)	(3,3)

Example: Prisoner's dilemma

Example: Prisoner's dilemma

- Prisoner's dilemma in extensive form (part (a)) and its normal-form (part (b)).

(a)



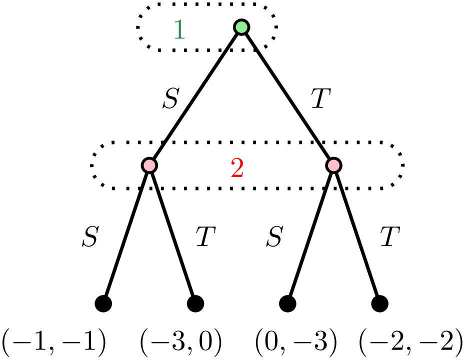
(b)

	T	S
T	(-2,-2)	(0,-3)
S	(-3,0)	(-1,-1)

Example: Prisoner's dilemma

- Prisoner's dilemma in extensive form (part (a)) and its normal-form (part (b)).

(a)



(b)

	T	S
T	(-2,-2)	(0,-3)
S	(-3,0)	(-1,-1)

- Every normal-form game can be expressed as an imperfect-information extensive game.

Strategies in extensive games

Strategies in extensive games

- A pure strategy for player i is a complete specification of which deterministic action to take at every information set belonging to i.

Strategies in extensive games

- A pure strategy for player i is a complete specification of which deterministic action to take at every information set belonging to i.
  - Formally, a pure strategy of player i is a vector $(c_h)_{h \in H_i}$ from the Cartesian product $\prod_{h \in H_i} C_h$.

Strategies in extensive games

- A pure strategy for player i is a complete specification of which deterministic action to take at every information set belonging to i.
  - Formally, a pure strategy of player i is a vector $(c_h)_{h \in H_i}$ from the Cartesian product $\prod_{h \in H_i} C_h$.
  - Using pure strategies, we can transform an extensive game G into a normal-form game G' simply by tabulating all pure strategies of the players and recording the resulting expected payoffs.

Strategies in extensive games

- A pure strategy for player i is a complete specification of which deterministic action to take at every information set belonging to i.
  - Formally, a pure strategy of player i is a vector $(c_h)_{h \in H_i}$ from the Cartesian product $\prod_{h \in H_i} C_h$.
  - Using pure strategies, we can transform an extensive game G into a normal-form game G' simply by tabulating all pure strategies of the players and recording the resulting expected payoffs.
- Mixed strategies of G are the mixed strategies of G'.

Strategies in extensive games

- A pure strategy for player i is a complete specification of which deterministic action to take at every information set belonging to i.
  - Formally, a pure strategy of player i is a vector $(c_h)_{h \in H_i}$ from the Cartesian product $\prod_{h \in H_i} C_h$.
  - Using pure strategies, we can transform an extensive game G into a normal-form game G' simply by tabulating all pure strategies of the players and recording the resulting expected payoffs.
- Mixed strategies of G are the mixed strategies of G'.
- In the same way, we also define the set of Nash equilibria of G.

Strategies in extensive games

- A pure strategy for player i is a complete specification of which deterministic action to take at every information set belonging to i.
  - Formally, a pure strategy of player i is a vector $(c_h)_{h \in H_i}$ from the Cartesian product $\prod_{h \in H_i} C_h$.
  - Using pure strategies, we can transform an extensive game G into a normal-form game G' simply by tabulating all pure strategies of the players and recording the resulting expected payoffs.
- Mixed strategies of G are the mixed strategies of G'.
- In the same way, we also define the set of Nash equilibria of G.
- A behavioral strategy of player i is a probability distribution on C_h for each $h \in H_i$.

Strategies in extensive games

- A pure strategy for player i is a complete specification of which deterministic action to take at every information set belonging to i.
  - Formally, a pure strategy of player i is a vector $(c_h)_{h \in H_i}$ from the Cartesian product $\prod_{h \in H_i} C_h$.
  - Using pure strategies, we can transform an extensive game G into a normal-form game G' simply by tabulating all pure strategies of the players and recording the resulting expected payoffs.
- Mixed strategies of G are the mixed strategies of G'.
- In the same way, we also define the set of Nash equilibria of G.
- A behavioral strategy of player i is a probability distribution on C_h for each $h \in H_i$.
  - This is a strategy in which each player's choice at each information set is made independently of his choices at other information sets.

Strategies in extensive games

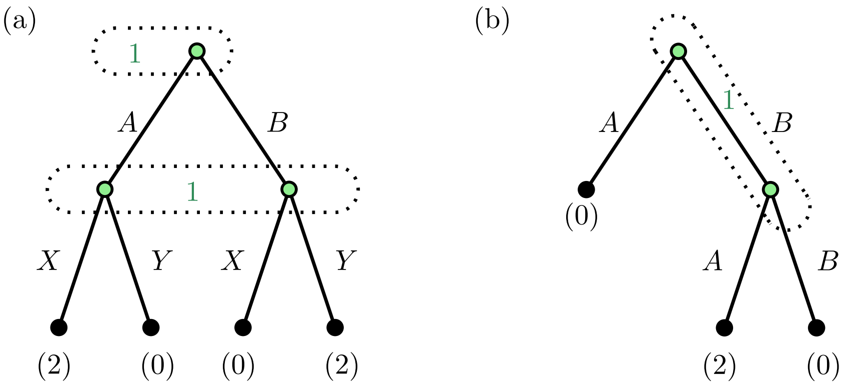
- A pure strategy for player i is a complete specification of which deterministic action to take at every information set belonging to i.
  - Formally, a pure strategy of player i is a vector $(c_h)_{h \in H_i}$ from the Cartesian product $\prod_{h \in H_i} C_h$.
  - Using pure strategies, we can transform an extensive game G into a normal-form game G' simply by tabulating all pure strategies of the players and recording the resulting expected payoffs.
- Mixed strategies of G are the mixed strategies of G'.
- In the same way, we also define the set of Nash equilibria of G.
- A behavioral strategy of player i is a probability distribution on C_h for each $h \in H_i$.
  - This is a strategy in which each player's choice at each information set is made independently of his choices at other information sets.
  - So a behavioral strategy is a vector of probability distributions while a mixed strategy is a probability distribution over vectors.

Strategies in extensive games

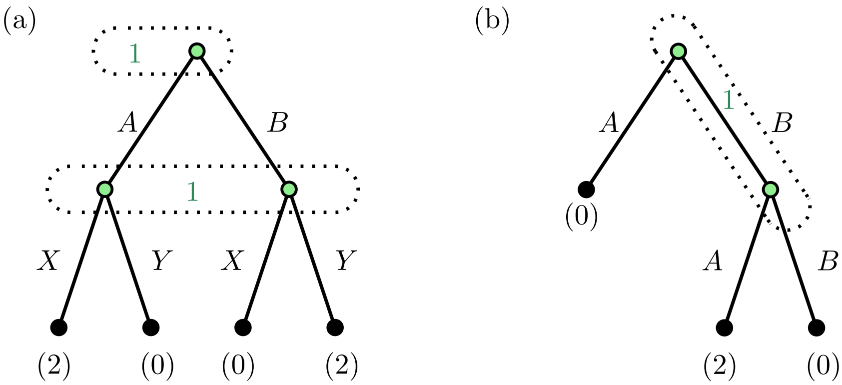
- A pure strategy for player i is a complete specification of which deterministic action to take at every information set belonging to i.
  - Formally, a pure strategy of player i is a vector $(c_h)_{h \in H_i}$ from the Cartesian product $\prod_{h \in H_i} C_h$.
  - Using pure strategies, we can transform an extensive game G into a normal-form game G' simply by tabulating all pure strategies of the players and recording the resulting expected payoffs.
- Mixed strategies of G are the mixed strategies of G'.
- In the same way, we also define the set of Nash equilibria of G.
- A behavioral strategy of player i is a probability distribution on C_h for each $h \in H_i$.
  - This is a strategy in which each player's choice at each information set is made independently of his choices at other information sets.
  - So a behavioral strategy is a vector of probability distributions while a mixed strategy is a probability distribution over vectors.
  - Unlike in mixed strategy, here a player might play different moves in different encounters of h.

Example: behavioral strategy vs. mixed strategy

Example: behavioral strategy vs. mixed strategy

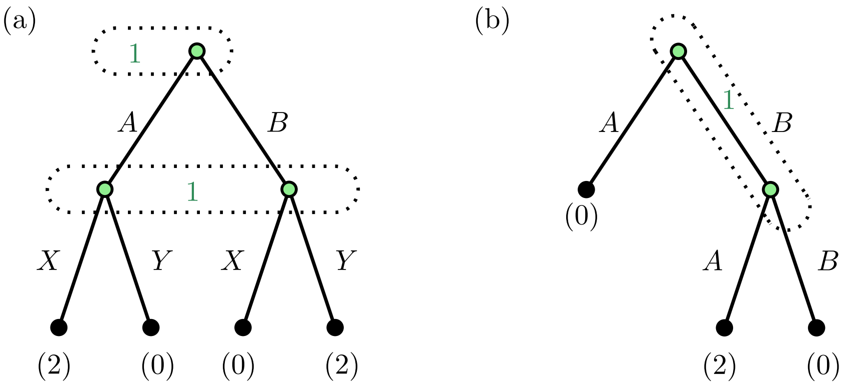


Example: behavioral strategy vs. mixed strategy



- (a) No behavioral strategy can remember whether A or B was played, while the mixed strategy $(\frac{1}{2}(A, X), \frac{1}{2}(B, Y))$ can employ this.

Example: behavioral strategy vs. mixed strategy



- (a) No behavioral strategy can remember whether A or B was played, while the mixed strategy $(\frac{1}{2}(A, X), \frac{1}{2}(B, Y))$ can employ this.
- (b) A mixed strategy has to commit to A or B both times, while the behavioral strategy $\frac{1}{2}A + \frac{1}{2}B$ can give a different move each time and yields a better payoff.

Games of perfect recall

Games of perfect recall

- In general, the expressive power of behavioral strategies and mixed strategies is incomparable.

Games of perfect recall

- In general, the expressive power of behavioral strategies and mixed strategies is incomparable.
- However, there is a large class of extensive games for which the two definitions coincide.

Games of perfect recall

- In general, the expressive power of behavioral strategies and mixed strategies is incomparable.
- However, there is a large class of extensive games for which the two definitions coincide. To define it, we need some auxiliary terms.

Games of perfect recall

- In general, the expressive power of behavioral strategies and mixed strategies is incomparable.
- However, there is a large class of extensive games for which the two definitions coincide. To define it, we need some auxiliary terms.
- A sequence $\sigma_i(t)$ of moves of player i to a node t is the sequence of his moves (disregarding the moves of other players) on the unique path from the root of the tree to t.

Games of perfect recall

- In general, the expressive power of behavioral strategies and mixed strategies is incomparable.
- However, there is a large class of extensive games for which the two definitions coincide. To define it, we need some auxiliary terms.
- A sequence $\sigma_i(t)$ of moves of player i to a node t is the sequence of his moves (disregarding the moves of other players) on the unique path from the root of the tree to t. The empty sequence is denoted $\emptyset$.

Games of perfect recall

- In general, the expressive power of behavioral strategies and mixed strategies is incomparable.
- However, there is a large class of extensive games for which the two definitions coincide. To define it, we need some auxiliary terms.
- A sequence $\sigma_i(t)$ of moves of player i to a node t is the sequence of his moves (disregarding the moves of other players) on the unique path from the root of the tree to t. The empty sequence is denoted $\emptyset$.
- Player i has perfect recall if and only if, for every $h \in H_i$ and any nodes $t, t' \in h$, we have $\sigma_i(t) = \sigma_i(t')$.

Games of perfect recall

- In general, the expressive power of behavioral strategies and mixed strategies is incomparable.
- However, there is a large class of extensive games for which the two definitions coincide. To define it, we need some auxiliary terms.
- A sequence $\sigma_i(t)$ of moves of player i to a node t is the sequence of his moves (disregarding the moves of other players) on the unique path from the root of the tree to t. The empty sequence is denoted $\emptyset$.
- Player i has perfect recall if and only if, for every $h \in H_i$ and any nodes $t, t' \in h$, we have $\sigma_i(t) = \sigma_i(t')$.
- In such case, we use σ_h to denote the unique sequence leading to any node t in h.

Games of perfect recall

- In general, the expressive power of behavioral strategies and mixed strategies is incomparable.
- However, there is a large class of extensive games for which the two definitions coincide. To define it, we need some auxiliary terms.
- A sequence $\sigma_i(t)$ of moves of player i to a node t is the sequence of his moves (disregarding the moves of other players) on the unique path from the root of the tree to t. The empty sequence is denoted $\emptyset$.
- Player i has perfect recall if and only if, for every $h \in H_i$ and any nodes $t, t' \in h$, we have $\sigma_i(t) = \sigma_i(t')$.
- In such case, we use σ_h to denote the unique sequence leading to any node t in h.
- A game G is a game of perfect recall if each player has perfect recall.

Games of perfect recall

- In general, the expressive power of behavioral strategies and mixed strategies is incomparable.
- However, there is a large class of extensive games for which the two definitions coincide. To define it, we need some auxiliary terms.
- A sequence $\sigma_i(t)$ of moves of player i to a node t is the sequence of his moves (disregarding the moves of other players) on the unique path from the root of the tree to t. The empty sequence is denoted $\emptyset$.
- Player i has perfect recall if and only if, for every $h \in H_i$ and any nodes $t, t' \in h$, we have $\sigma_i(t) = \sigma_i(t')$.
- In such case, we use σ_h to denote the unique sequence leading to any node t in h.
- A game G is a game of perfect recall if each player has perfect recall.
  - No player forgets any information he knew about moves made so far.

Games of perfect recall

- In general, the expressive power of behavioral strategies and mixed strategies is incomparable.
- However, there is a large class of extensive games for which the two definitions coincide. To define it, we need some auxiliary terms.
- A sequence $\sigma_i(t)$ of moves of player i to a node t is the sequence of his moves (disregarding the moves of other players) on the unique path from the root of the tree to t. The empty sequence is denoted $\emptyset$.
- Player i has perfect recall if and only if, for every $h \in H_i$ and any nodes $t, t' \in h$, we have $\sigma_i(t) = \sigma_i(t')$.
- In such case, we use σ_h to denote the unique sequence leading to any node t in h.
- A game G is a game of perfect recall if each player has perfect recall.
    - No player forgets any information he knew about moves made so far. That is, each player remembers what he did in prior moves, and each player remembers everything that he knew before.

Games of perfect recall

- In general, the expressive power of behavioral strategies and mixed strategies is incomparable.
- However, there is a large class of extensive games for which the two definitions coincide. To define it, we need some auxiliary terms.
- A sequence $\sigma_i(t)$ of moves of player i to a node t is the sequence of his moves (disregarding the moves of other players) on the unique path from the root of the tree to t. The empty sequence is denoted $\emptyset$.
- Player i has perfect recall if and only if, for every $h \in H_i$ and any nodes $t, t' \in h$, we have $\sigma_i(t) = \sigma_i(t')$.
- In such case, we use σ_h to denote the unique sequence leading to any node t in h.
- A game G is a game of perfect recall if each player has perfect recall.
  - No player forgets any information he knew about moves made so far. That is, each player remembers what he did in prior moves, and each player remembers everything that he knew before.
  - Every perfect-information game is a game of perfect recall.

Kuhn's Theorem

Kuhn's Theorem

- In games of perfect recall, mixed strategies and behavioral strategies are equivalent.

Kuhn's Theorem

- In games of perfect recall, mixed strategies and behavioral strategies are equivalent.

Kuhn's theorem (Theorem 2.62)

In a game of perfect recall, any mixed strategy of a given player can be replaced by an equivalent behavioral strategy, and any behavioral strategy can be replaced by an equivalent mixed strategy.

Kuhn's Theorem

- In games of perfect recall, mixed strategies and behavioral strategies are equivalent.

Kuhn's theorem (Theorem 2.62)

In a game of perfect recall, any mixed strategy of a given player can be replaced by an equivalent behavioral strategy, and any behavioral strategy can be replaced by an equivalent mixed strategy.

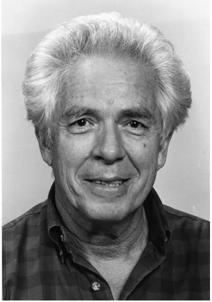


Figure: Harold William Kuhn (1925–2014).

Sources: https://alchetron.com/Harold-W-Kuhn and https://www.cantorsparadise.com/

Finding NE in extensive games

Finding NE in extensive games

- Every extensive game G can be converted into an equivalent normal-form game G'.

Finding NE in extensive games

- Every extensive game G can be converted into an equivalent normal-form game G'.
- So we can find NE of G by converting it into G' and applying the Lemke–Howson algorithm to G'.

Finding NE in extensive games

- Every extensive game G can be converted into an equivalent normal-form game G'.
- So we can find NE of G by converting it into G' and applying the Lemke–Howson algorithm to G'.
- However, this is inefficient, as the number of actions in G' is exponential in the size of G.

Finding NE in extensive games

- Every extensive game G can be converted into an equivalent normal-form game G'.
- So we can find NE of G by converting it into G' and applying the Lemke–Howson algorithm to G'.
- However, this is inefficient, as the number of actions in G' is exponential in the size of G. So the number of steps can be double exponential!

Finding NE in extensive games

- Every extensive game G can be converted into an equivalent normal-form game G'.
- So we can find NE of G by converting it into G' and applying the Lemke–Howson algorithm to G'.
- However, this is inefficient, as the number of actions in G' is exponential in the size of G. So the number of steps can be double exponential!
- To avoid this problem, we will work directly with G using so-called sequence form.

Finding NE in extensive games

- Every extensive game G can be converted into an equivalent normal-form game G'.
- So we can find NE of G by converting it into G' and applying the Lemke–Howson algorithm to G'.
- However, this is inefficient, as the number of actions in G' is exponential in the size of G. So the number of steps can be double exponential!
- To avoid this problem, we will work directly with G using so-called sequence form.

- From now on, we consider only games of perfect recall.

Finding NE in extensive games

- Every extensive game G can be converted into an equivalent normal-form game G'.
- So we can find NE of G by converting it into G' and applying the Lemke–Howson algorithm to G'.
- However, this is inefficient, as the number of actions in G' is exponential in the size of G. So the number of steps can be double exponential!
- To avoid this problem, we will work directly with G using so-called sequence form.

- From now on, we consider only games of perfect recall.
- By Kuhn's Theorem, NE do not change if we restrict ourselves to behavioral strategies.

Finding NE in extensive games

- Every extensive game G can be converted into an equivalent normal-form game G'.
- So we can find NE of G by converting it into G' and applying the Lemke–Howson algorithm to G'.
- However, this is inefficient, as the number of actions in G' is exponential in the size of G. So the number of steps can be double exponential!
- To avoid this problem, we will work directly with G using so-called sequence form.

- From now on, we consider only games of perfect recall.
- By Kuhn's Theorem, NE do not change if we restrict ourselves to behavioral strategies. So we will work with them.

Sequence form

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,
  - $S = (S_1, \ldots, S_n)$, where S_i is a set of sequences of player i,

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,
  - $S = (S_1, \ldots, S_n)$, where S_i is a set of sequences of player i,
  - $u = (u_1, \ldots, u_n)$, where $u_i \colon S \to \mathbb{R}$ is the payoff function of player i, and

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,
  - $S = (S_1, \ldots, S_n)$, where S_i is a set of sequences of player i,
  - $u = (u_1, \ldots, u_n)$, where $u_i \colon S \to \mathbb{R}$ is the payoff function of player i, and
  - $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_n)$ is a set of linear constraints on the realization probabilities of player i.

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,
  - $S = (S_1, \ldots, S_n)$, where S_i is a set of sequences of player i,
  - $u = (u_1, \ldots, u_n)$, where $u_i \colon S \to \mathbb{R}$ is the payoff function of player i, and
  - $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_n)$ is a set of linear constraints on the realization probabilities of player i.
- Now, we will define all these terms properly.

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,
  - $S = (S_1, \ldots, S_n)$, where S_i is a set of sequences of player i,
  - $u = (u_1, \ldots, u_n)$, where $u_i \colon S \to \mathbb{R}$ is the payoff function of player i, and
  - $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_n)$ is a set of linear constraints on the realization probabilities of player i.
- Now, we will define all these terms properly. It will take some time...

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,
  - $S = (S_1, \ldots, S_n)$, where S_i is a set of sequences of player i,
  - $u = (u_1, \ldots, u_n)$, where $u_i : S \to \mathbb{R}$ is the payoff function of player i, and
  - $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_n)$ is a set of linear constraints on the realization probabilities of player i.
- Now, we will define all these terms properly. It will take some time...
- First, we explain the set of sequences S in more detail.

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,
  - $S = (S_1, \ldots, S_n)$, where S_i is a set of sequences of player i,
  - $u = (u_1, \ldots, u_n)$, where $u_i \colon S \to \mathbb{R}$ is the payoff function of player i, and
  - $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_n)$ is a set of linear constraints on the realization probabilities of player i.
- Now, we will define all these terms properly. It will take some time...

- First, we explain the set of sequences S in more detail.
  - Any sequence σ from S_i is either the empty sequence $\emptyset$ or it is uniquely determined by the last move c at the information set h,

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,
  - $S = (S_1, \dots, S_n)$, where S_i is a set of sequences of player i,
  - $u = (u_1, \dots, u_n)$, where $u_i \colon S \to \mathbb{R}$ is the payoff function of player i, and
  - $\mathcal{C} = (\mathcal{C}_1, \dots, \mathcal{C}_n)$ is a set of linear constraints on the realization probabilities of player i.
- Now, we will define all these terms properly. It will take some time...

- First, we explain the set of sequences S in more detail.
  - Any sequence σ from S_i is either the empty sequence $\emptyset$ or it is uniquely determined by the last move c at the information set h, that is, $\sigma = \sigma_h c$.

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,
  - $S = (S_1, \ldots, S_n)$, where S_i is a set of sequences of player i,
  - $u = (u_1, \ldots, u_n)$, where $u_i \colon S \to \mathbb{R}$ is the payoff function of player i, and
  - $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_n)$ is a set of linear constraints on the realization probabilities of player i.
- Now, we will define all these terms properly. It will take some time...

- First, we explain the set of sequences S in more detail.
  - Any sequence σ from S_i is either the empty sequence $\emptyset$ or it is uniquely determined by the last move c at the information set h, that is, $\sigma = \sigma_h c$.
  - Thus, $S_i = \{\emptyset\} \cup \{\sigma_h c \colon h \in H_i, c \in C_h\}$.

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,
  - $S = (S_1, \ldots, S_n)$, where S_i is a set of sequences of player i,
  - $u = (u_1, \ldots, u_n)$, where $u_i \colon S \to \mathbb{R}$ is the payoff function of player i, and
  - $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_n)$ is a set of linear constraints on the realization probabilities of player i.
- Now, we will define all these terms properly. It will take some time...

- First, we explain the set of sequences S in more detail.
  - Any sequence σ from S_i is either the empty sequence $\emptyset$ or it is uniquely determined by the last move c at the information set h, that is, $\sigma = \sigma_h c$.
  - Thus, $S_i = \{\emptyset\} \cup \{\sigma_h c \colon h \in H_i, c \in C_h\}$.
  - It follows that $|S_i| = 1 + \sum_{h \in H_i} |C_h|$,

Sequence form

- The sequence form of an imperfect-information game G is a 4-tuple $(P, S, u, \mathcal{C})$ where
  - P is a set of n players,
  - $S = (S_1, \ldots, S_n)$, where S_i is a set of sequences of player i,
  - $u = (u_1, \ldots, u_n)$, where $u_i \colon S \to \mathbb{R}$ is the payoff function of player i, and
  - $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_n)$ is a set of linear constraints on the realization probabilities of player i.
- Now, we will define all these terms properly. It will take some time...

- First, we explain the set of sequences S in more detail.
  - Any sequence σ from S_i is either the empty sequence $\emptyset$ or it is uniquely determined by the last move c at the information set h, that is, $\sigma = \sigma_h c$.
  - Thus, $S_i = \{\emptyset\} \cup \{\sigma_h c \colon h \in H_i, c \in C_h\}$.
  - It follows that $|S_i| = 1 + \sum_{h \in H_i} |C_h|$, which is linear in the size of the tree of G.

Still defining the sequence form

Still defining the sequence form

- We now explain the payoff function u in more detail.

Still defining the sequence form

- We now explain the payoff function u in more detail.
  - For player i and sequences $\sigma = (\sigma_1, \ldots, \sigma_n) \in S$, the payoff $u_i(\sigma)$ equals $u_i(\ell)$ where ℓ is the leaf that would be reached if each player j played his sequence σ_j.

Still defining the sequence form

- We now explain the payoff function u in more detail.
  - For player i and sequences $\sigma = (\sigma_1, \ldots, \sigma_n) \in S$, the payoff $u_i(\sigma)$ equals $u_i(\ell)$ where ℓ is the leaf that would be reached if each player j played his sequence σ_j. Otherwise, $u_i(\sigma) = 0$.

Still defining the sequence form

- We now explain the payoff function u in more detail.
  - For player i and sequences $\sigma = (\sigma_1, \ldots, \sigma_n) \in S$, the payoff $u_i(\sigma)$ equals $u_i(\ell)$ where ℓ is the leaf that would be reached if each player j played his sequence σ_j. Otherwise, $u_i(\sigma) = 0$.
  - Similarly as in normal-form games, we can represent the payoffs u using matrices with entries indexed by elements from S.

Still defining the sequence form

- We now explain the payoff function u in more detail.
  - For player i and sequences $\sigma = (\sigma_1, \ldots, \sigma_n) \in S$, the payoff $u_i(\sigma)$ equals $u_i(\ell)$ where ℓ is the leaf that would be reached if each player j played his sequence σ_j. Otherwise, $u_i(\sigma) = 0$.
  - Similarly as in normal-form games, we can represent the payoffs u using matrices with entries indexed by elements from S.
  - Note that these matrices are sparse as most entries are 0.

Still defining the sequence form

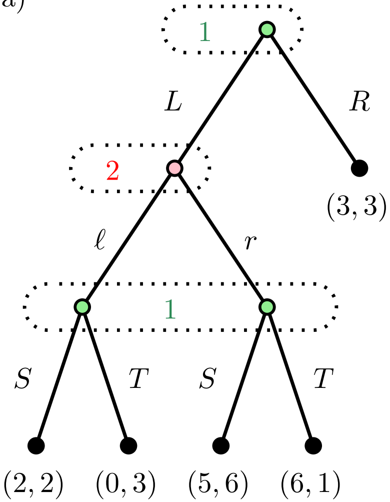
- We now explain the payoff function u in more detail.
  - For player i and sequences $\sigma = (\sigma_1, \ldots, \sigma_n) \in S$, the payoff $u_i(\sigma)$ equals $u_i(\ell)$ where ℓ is the leaf that would be reached if each player j played his sequence σ_j. Otherwise, $u_i(\sigma) = 0$.
  - Similarly as in normal-form games, we can represent the payoffs u using matrices with entries indexed by elements from S.
  - Note that these matrices are sparse as most entries are 0.
  - If there are only two players, then we capture their payoffs with matrices A and B.

Example: sequence form payoff matrices

Example: sequence form payoff matrices

- An example of an imperfect-information game in extensive form (part (a)) and its sequence form payoff matrices (part (b)).

(a)



(b)

$$A = \begin{array}{c} \\ \\ \end{array}\begin{array}{cc} & \begin{array}{ccc} \emptyset & \ell & r \end{array} \\ & \begin{pmatrix} & & \\ & & \\ 3 & & \\ & 2 & 5 \\ & 0 & 6 \end{pmatrix} \begin{array}{l} \emptyset \\ L \\ R \\ LS \\ LT \end{array} \end{array}$$

$$B = \begin{array}{cc} & \begin{array}{ccc} \emptyset & \ell & r \end{array} \\ & \begin{pmatrix} & & \\ & & \\ 3 & & \\ & 2 & 6 \\ & 3 & 1 \end{pmatrix} \begin{array}{l} \emptyset \\ L \\ R \\ LS \\ LT \end{array} \end{array}$$

Finally finishing the definition of the sequence form

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf.

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf. This is why we will work with behavioral strategies.

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf. This is why we will work with behavioral strategies.
  - However, working with them directly is computationally difficult.

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf. This is why we will work with behavioral strategies.
  - However, working with them directly is computationally difficult. So we develop an alternate concept of a realization plan.

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf. This is why we will work with behavioral strategies.
  - However, working with them directly is computationally difficult. So we develop an alternate concept of a realization plan.
  - The realization plan of a behavioral strategy β_i for player i is a mapping $x\colon S_i \to [0,1]$ defined as $x(\sigma_i) = \prod_{c \in \sigma_i} \beta_i(c)$.

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf. This is why we will work with behavioral strategies.
  - However, working with them directly is computationally difficult. So we develop an alternate concept of a realization plan.
  - The realization plan of a behavioral strategy β_i for player i is a mapping $x\colon S_i \to [0,1]$ defined as $x(\sigma_i) = \prod_{c\in\sigma_i} \beta_i(c)$. The value $x(\sigma_i)$ is called the realization probability.

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf. This is why we will work with behavioral strategies.
  - However, working with them directly is computationally difficult. So we develop an alternate concept of a realization plan.
  - The realization plan of a behavioral strategy β_i for player i is a mapping $x\colon S_i \to [0,1]$ defined as $x(\sigma_i) = \prod_{c \in \sigma_i} \beta_i(c)$. The value $x(\sigma_i)$ is called the realization probability.
    - The realization plan is the probability that a sequence arises under a given behavioral strategy.

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf. This is why we will work with behavioral strategies.
  - However, working with them directly is computationally difficult. So we develop an alternate concept of a realization plan.
  - The realization plan of a behavioral strategy β_i for player i is a mapping $x\colon S_i \to [0,1]$ defined as $x(\sigma_i) = \prod_{c \in \sigma_i} \beta_i(c)$. The value $x(\sigma_i)$ is called the realization probability.
    - The realization plan is the probability that a sequence arises under a given behavioral strategy.
    - Equivalently, we can work with it using a set of linear equations:

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf. This is why we will work with behavioral strategies.
  - However, working with them directly is computationally difficult. So we develop an alternate concept of a realization plan.
  - The realization plan of a behavioral strategy β_i for player i is a mapping $x\colon S_i \to [0,1]$ defined as $x(\sigma_i) = \prod_{c \in \sigma_i} \beta_i(c)$. The value $x(\sigma_i)$ is called the realization probability.
    - The realization plan is the probability that a sequence arises under a given behavioral strategy.
    - Equivalently, we can work with it using a set of linear equations: A realization plan for player i is a mapping $x\colon S_i \to [0,1]$ satisfying

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf. This is why we will work with behavioral strategies.
  - However, working with them directly is computationally difficult. So we develop an alternate concept of a realization plan.
  - The realization plan of a behavioral strategy β_i for player i is a mapping $x\colon S_i \to [0,1]$ defined as $x(\sigma_i) = \prod_{c \in \sigma_i} \beta_i(c)$. The value $x(\sigma_i)$ is called the realization probability.
    - The realization plan is the probability that a sequence arises under a given behavioral strategy.
    - Equivalently, we can work with it using a set of linear equations: A realization plan for player i is a mapping $x\colon S_i \to [0,1]$ satisfying $x(\emptyset) = 1$, and

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf. This is why we will work with behavioral strategies.
  - However, working with them directly is computationally difficult. So we develop an alternate concept of a realization plan.
  - The realization plan of a behavioral strategy β_i for player i is a mapping $x\colon S_i \to [0,1]$ defined as $x(\sigma_i) = \prod_{c \in \sigma_i} \beta_i(c)$. The value $x(\sigma_i)$ is called the realization probability.
    - The realization plan is the probability that a sequence arises under a given behavioral strategy.
    - Equivalently, we can work with it using a set of linear equations: A realization plan for player i is a mapping $x\colon S_i \to [0,1]$ satisfying $x(\emptyset) = 1$, and $\sum_{c \in C_h} x(\sigma_h c) = x(\sigma_h)$ for every $h \in H_i$.

Finally finishing the definition of the sequence form

- We now explain the linear constraints $\mathcal{C}$ in more detail.
  - We still do not have everything to describe G since a player cannot choose sequences as actions because other players might not play in a way that would allow him to follow to a leaf. This is why we will work with behavioral strategies.
  - However, working with them directly is computationally difficult. So we develop an alternate concept of a realization plan.
  - The realization plan of a behavioral strategy β_i for player i is a mapping $x\colon S_i \to [0,1]$ defined as $x(\sigma_i) = \prod_{c \in \sigma_i} \beta_i(c)$. The value $x(\sigma_i)$ is called the realization probability.
    - The realization plan is the probability that a sequence arises under a given behavioral strategy.
    - Equivalently, we can work with it using a set of linear equations: A realization plan for player i is a mapping $x\colon S_i \to [0,1]$ satisfying $x(\emptyset) = 1$, and $\sum_{c \in C_h} x(\sigma_h c) = x(\sigma_h)$ for every $h \in H_i$.
    - We let $\mathcal{C}_i$ be the set of constraints of the second type.

Using the sequence form

Using the sequence form

- Consider an extensive game G of two players.

Using the sequence form

- Consider an extensive game G of two players.
- We show how to actually use the sequence form to compute NE.

Using the sequence form

- Consider an extensive game G of two players.
- We show how to actually use the sequence form to compute NE.
- Consider realization plans as vectors $x = (x_\sigma)_{\sigma \in S_1} \in \mathbb{R}^{|S_1|}$ and $y = (y_\tau)_{\tau \in S_2} \in \mathbb{R}^{|S_2|}$.

Using the sequence form

- Consider an extensive game G of two players.
- We show how to actually use the sequence form to compute NE.
- Consider realization plans as vectors $x = (x_\sigma)_{\sigma \in S_1} \in \mathbb{R}^{|S_1|}$ and $y = (y_\tau)_{\tau \in S_2} \in \mathbb{R}^{|S_2|}$.
- Then, the linear constraints from $\mathcal{C}$ can be written as

Using the sequence form

- Consider an extensive game G of two players.
- We show how to actually use the sequence form to compute NE.
- Consider realization plans as vectors $x = (x_\sigma)_{\sigma \in S_1} \in \mathbb{R}^{|S_1|}$ and $y = (y_\tau)_{\tau \in S_2} \in \mathbb{R}^{|S_2|}$.
- Then, the linear constraints from $\mathcal{C}$ can be written as

$$Ex = e, \quad x \geq \mathbf{0} \quad \text{and} \quad Fy = f, \quad y \geq \mathbf{0}$$

Using the sequence form

- Consider an extensive game G of two players.
- We show how to actually use the sequence form to compute NE.
- Consider realization plans as vectors $x = (x_\sigma)_{\sigma \in S_1} \in \mathbb{R}^{|S_1|}$ and $y = (y_\tau)_{\tau \in S_2} \in \mathbb{R}^{|S_2|}$.
- Then, the linear constraints from $\mathcal{C}$ can be written as

$$Ex = e, \quad x \geq \mathbf{0} \quad \text{and} \quad Fy = f, \quad y \geq \mathbf{0}$$

  where the constraint matrices E and F have $1 + |H_1|$ and $1 + |H_2|$ rows with first row of $Ex = e$ and $Fy = f$ corresponding to $x(\emptyset) = 1$ for E and $y(\emptyset) = 1$ for F.

Using the sequence form

- Consider an extensive game G of two players.
- We show how to actually use the sequence form to compute NE.
- Consider realization plans as vectors $x = (x_\sigma)_{\sigma \in S_1} \in \mathbb{R}^{|S_1|}$ and $y = (y_\tau)_{\tau \in S_2} \in \mathbb{R}^{|S_2|}$.
- Then, the linear constraints from $\mathcal{C}$ can be written as

$$Ex = e, \quad x \geq \mathbf{0} \quad \text{and} \quad Fy = f, \quad y \geq \mathbf{0}$$

  where the constraint matrices E and F have $1 + |H_1|$ and $1 + |H_2|$ rows with first row of $Ex = e$ and $Fy = f$ corresponding to $x(\emptyset) = 1$ for E and $y(\emptyset) = 1$ for F.
- The other rows of $Ex = e$ are $-x(\sigma_h) + \sum_{c \in C_h} x(\sigma_h c) = 0$ for every $h \in H_1$.

Using the sequence form

- Consider an extensive game G of two players.
- We show how to actually use the sequence form to compute NE.
- Consider realization plans as vectors $x = (x_\sigma)_{\sigma \in S_1} \in \mathbb{R}^{|S_1|}$ and $y = (y_\tau)_{\tau \in S_2} \in \mathbb{R}^{|S_2|}$.
- Then, the linear constraints from $\mathcal{C}$ can be written as

$$Ex = e, \quad x \geq \mathbf{0} \quad \text{and} \quad Fy = f, \quad y \geq \mathbf{0}$$

  where the constraint matrices E and F have $1 + |H_1|$ and $1 + |H_2|$ rows with first row of $Ex = e$ and $Fy = f$ corresponding to $x(\emptyset) = 1$ for E and $y(\emptyset) = 1$ for F.
- The other rows of $Ex = e$ are $-x(\sigma_h) + \sum_{c \in C_h} x(\sigma_h c) = 0$ for every $h \in H_1$.
- For $Fy = f$, we have the rows $-y(\sigma_h) + \sum_{c \in C_h} y(\sigma_h c) = 0$ for every $h \in H_2$.

Using the sequence form

- Consider an extensive game G of two players.
- We show how to actually use the sequence form to compute NE.
- Consider realization plans as vectors $x = (x_\sigma)_{\sigma \in S_1} \in \mathbb{R}^{|S_1|}$ and $y = (y_\tau)_{\tau \in S_2} \in \mathbb{R}^{|S_2|}$.
- Then, the linear constraints from $\mathcal{C}$ can be written as

$$Ex = e, \quad x \geq \mathbf{0} \quad \text{and} \quad Fy = f, \quad y \geq \mathbf{0}$$

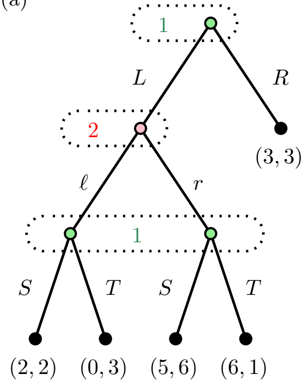
  where the constraint matrices E and F have $1 + |H_1|$ and $1 + |H_2|$ rows with first row of $Ex = e$ and $Fy = f$ corresponding to $x(\emptyset) = 1$ for E and $y(\emptyset) = 1$ for F.
- The other rows of $Ex = e$ are $-x(\sigma_h) + \sum_{c \in C_h} x(\sigma_h c) = 0$ for every $h \in H_1$.
- For $Fy = f$, we have the rows $-y(\sigma_h) + \sum_{c \in C_h} y(\sigma_h c) = 0$ for every $h \in H_2$.
- The vectors e and f also have $1 + |H_1|$ and $1 + |H_2|$ rows.

Example: sequence form constraints

Example: sequence form constraints

- An example of an imperfect-information game in extensive form (part (a)) and linear constraints in its sequence form (part (b)).

(a)



(b)

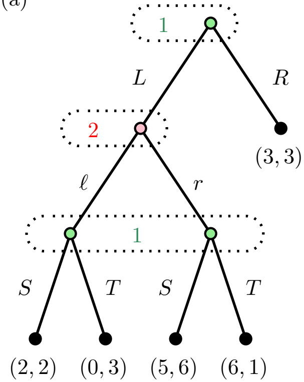
$$E = \begin{pmatrix} 1 & & & & \\ -1 & 1 & 1 & & \\ & -1 & & 1 & 1 \end{pmatrix}, \quad e = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix},$$

$$F = \begin{pmatrix} 1 & & \\ -1 & 1 & 1 \end{pmatrix}, \quad f = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Example: sequence form constraints

- An example of an imperfect-information game in extensive form (part (a)) and linear constraints in its sequence form (part (b)).

(a)



(b)

$$E = \begin{pmatrix} 1 & & & & \\ -1 & 1 & 1 & & \\ & -1 & & 1 & 1 \end{pmatrix}, \quad e = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix},$$

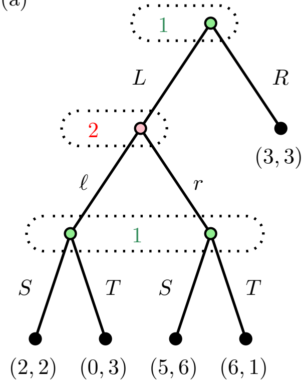
$$F = \begin{pmatrix} 1 & & \\ -1 & 1 & 1 \end{pmatrix}, \quad f = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

- We can uniquely recover behavioral strategy β_i from a realization plan x for player i in all relevant information sets $h \in H_i$ with $x(\sigma_h) > 0$.

Example: sequence form constraints

- An example of an imperfect-information game in extensive form (part (a)) and linear constraints in its sequence form (part (b)).

(a)



(b)

$$E = \begin{pmatrix} 1 & & & & \\ -1 & 1 & 1 & & \\ & -1 & & 1 & 1 \end{pmatrix}, \quad e = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix},$$

$$F = \begin{pmatrix} 1 & & \\ -1 & 1 & 1 \end{pmatrix}, \quad f = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

- We can uniquely recover behavioral strategy β_i from a realization plan x for player i in all relevant information sets $h \in H_i$ with $x(\sigma_h) > 0$.
- For each such $h \in H_i$ and $c \in C_h$, we set $\beta_i(h, c) = \frac{x(\sigma_h c)}{x(\sigma_h)}$.

Using the sequence form to find best responses

Using the sequence form to find best responses

- We first show how to compute best responses.

Using the sequence form to find best responses

- We first show how to compute best responses.
- For a fixed realization plan y of player 2, a best response of player 1 is a realization plan x that maximizes the expected payoff.

Using the sequence form to find best responses

- We first show how to compute best responses.
- For a fixed realization plan y of player 2, a best response of player 1 is a realization plan x that maximizes the expected payoff.
- Thus, x is a solution to the following linear program P

Using the sequence form to find best responses

- We first show how to compute best responses.
- For a fixed realization plan y of player 2, a best response of player 1 is a realization plan x that maximizes the expected payoff.
- Thus, x is a solution to the following linear program P

$$
\begin{aligned}
&\max x^\top Ay \text{ subject to} \\
&Ex = e, \\
&x \geq \mathbf{0}.
\end{aligned}
$$

Using the sequence form to find best responses

- We first show how to compute best responses.
- For a fixed realization plan y of player 2, a best response of player 1 is a realization plan x that maximizes the expected payoff.
- Thus, x is a solution to the following linear program P

$$\max x^\top Ay \text{ subject to}$$
$$Ex = e,$$
$$x \geq \mathbf{0}.$$

- The dual D of P uses unconstrained variables u and is of the form

Using the sequence form to find best responses

- We first show how to compute best responses.
- For a fixed realization plan y of player 2, a best response of player 1 is a realization plan x that maximizes the expected payoff.
- Thus, x is a solution to the following linear program P

$$\begin{aligned} &\max x^\top Ay \text{ subject to} \\ &Ex = e, \\ &x \geq \mathbf{0}. \end{aligned}$$

- The dual D of P uses unconstrained variables u and is of the form

$$\begin{aligned} &\min e^\top u \text{ subject to} \\ &E^\top u \geq Ay. \end{aligned}$$

Using the sequence form to find best responses

- We first show how to compute best responses.
- For a fixed realization plan y of player 2, a best response of player 1 is a realization plan x that maximizes the expected payoff.
- Thus, x is a solution to the following linear program P

$$\begin{aligned}&\max x^\top Ay \text{ subject to}\\&Ex = e,\\&x \geq \mathbf{0}.\end{aligned}$$

- The dual D of P uses unconstrained variables u and is of the form

$$\begin{aligned}&\min e^\top u \text{ subject to}\\&E^\top u \geq Ay.\end{aligned}$$

- Analogous LPs can be used to compute best responses of player 2.

Using the sequence form to find NE in zero-sum games

Using the sequence form to find NE in zero-sum games

- Assume G is zero-sum, that is, $A = -B$.

Using the sequence form to find NE in zero-sum games

- Assume G is zero-sum, that is, $A = -B$.
- Then, using the Duality Theorem to P and D, similarly as we did in the proof of the Minimax theorem, gives an LP for finding NE in G.

Using the sequence form to find NE in zero-sum games

- Assume G is zero-sum, that is, $A = -B$.
- Then, using the Duality Theorem to P and D, similarly as we did in the proof of the Minimax theorem, gives an LP for finding NE in G.
- The reason is that player 2 wants to minimize $x^\top Ay$, which by duality equals $e^\top u$ if player 1 maximizes his payoff $x^\top Ay$.

Using the sequence form to find NE in zero-sum games

- Assume G is zero-sum, that is, $A = -B$.
- Then, using the Duality Theorem to P and D, similarly as we did in the proof of the Minimax theorem, gives an LP for finding NE in G.
- The reason is that player 2 wants to minimize $x^\top Ay$, which by duality equals $e^\top u$ if player 1 maximizes his payoff $x^\top Ay$.

Theorem 2.65

NE of a 2-player zero-sum extensive game of perfect recall are solutions of the following LP:

$$\min_{u,y} e^\top u \text{ subject to } Fy = f, E^\top u - Ay \geq \mathbf{0}, y \geq \mathbf{0}.$$

Using the sequence form to find NE in zero-sum games

- Assume G is zero-sum, that is, $A = -B$.
- Then, using the Duality Theorem to P and D, similarly as we did in the proof of the Minimax theorem, gives an LP for finding NE in G.
- The reason is that player 2 wants to minimize $x^\top Ay$, which by duality equals $e^\top u$ if player 1 maximizes his payoff $x^\top Ay$.

Theorem 2.65

NE of a 2-player zero-sum extensive game of perfect recall are solutions of the following LP:

$$\min_{u,y} e^\top u \text{ subject to } Fy = f, E^\top u - Ay \geq \mathbf{0}, y \geq \mathbf{0}.$$

- The dual of this program has variables x and v and is of the form

$$\max_{v,x} f^\top v \text{ subject to } Ex = e, F^\top v - A^\top x \leq \mathbf{0}, x \geq \mathbf{0}.$$

Using the sequence form to find NE in zero-sum games

- Assume G is zero-sum, that is, $A = -B$.
- Then, using the Duality Theorem to P and D, similarly as we did in the proof of the Minimax theorem, gives an LP for finding NE in G.
- The reason is that player 2 wants to minimize $x^\top Ay$, which by duality equals $e^\top u$ if player 1 maximizes his payoff $x^\top Ay$.

Theorem 2.65

NE of a 2-player zero-sum extensive game of perfect recall are solutions of the following LP:

$$\min_{u,y} e^\top u \text{ subject to } Fy = f, E^\top u - Ay \geq \mathbf{0}, y \geq \mathbf{0}.$$

- The dual of this program has variables x and v and is of the form

$$\max_{v,x} f^\top v \text{ subject to } Ex = e, F^\top v - A^\top x \leq \mathbf{0}, x \geq \mathbf{0}.$$

- This LP finds realization plan x with payoff $f^\top v$ for player 1.

Using the sequence form to find NE in zero-sum games

- Assume G is zero-sum, that is, $A = -B$.
- Then, using the Duality Theorem to P and D, similarly as we did in the proof of the Minimax theorem, gives an LP for finding NE in G.
- The reason is that player 2 wants to minimize $x^\top Ay$, which by duality equals $e^\top u$ if player 1 maximizes his payoff $x^\top Ay$.

Theorem 2.65

NE of a 2-player zero-sum extensive game of perfect recall are solutions of the following LP:

$$\min_{u,y} e^\top u \text{ subject to } Fy = f, E^\top u - Ay \geq \mathbf{0}, y \geq \mathbf{0}.$$

- The dual of this program has variables x and v and is of the form

$$\max_{v,x} f^\top v \text{ subject to } Ex = e, F^\top v - A^\top x \leq \mathbf{0}, x \geq \mathbf{0}.$$

- This LP finds realization plan x with payoff $f^\top v$ for player 1.
- The number of nonzero entries in matrices $E, F.A, B$ is linear in the size of the game tree.

Using the sequence form to find NE in zero-sum games

- Assume G is zero-sum, that is, $A = -B$.
- Then, using the Duality Theorem to P and D, similarly as we did in the proof of the Minimax theorem, gives an LP for finding NE in G.
- The reason is that player 2 wants to minimize $x^\top Ay$, which by duality equals $e^\top u$ if player 1 maximizes his payoff $x^\top Ay$.

Theorem 2.65

NE of a 2-player zero-sum extensive game of perfect recall are solutions of the following LP:

$$\min_{u,y} e^\top u \text{ subject to } Fy = f, E^\top u - Ay \geq \mathbf{0}, y \geq \mathbf{0}.$$

- The dual of this program has variables x and v and is of the form

$$\max_{v,x} f^\top v \text{ subject to } Ex = e, F^\top v - A^\top x \leq \mathbf{0}, x \geq \mathbf{0}.$$

- This LP finds realization plan x with payoff $f^\top v$ for player 1.
- The number of nonzero entries in matrices $E, F.A, B$ is linear in the size of the game tree. Thus, the LPs can be solved in polynomial time with respect to the size of G.

Using the sequence form to find NE in zero-sum games

- Assume G is zero-sum, that is, $A = -B$.
- Then, using the Duality Theorem to P and D, similarly as we did in the proof of the Minimax theorem, gives an LP for finding NE in G.
- The reason is that player 2 wants to minimize $x^\top Ay$, which by duality equals $e^\top u$ if player 1 maximizes his payoff $x^\top Ay$.

Theorem 2.65

NE of a 2-player zero-sum extensive game of perfect recall are solutions of the following LP:

$$\min_{u,y} e^\top u \text{ subject to } Fy = f, E^\top u - Ay \geq \mathbf{0}, y \geq \mathbf{0}.$$

- The dual of this program has variables x and v and is of the form

$$\max_{v,x} f^\top v \text{ subject to } Ex = e, F^\top v - A^\top x \leq \mathbf{0}, x \geq \mathbf{0}.$$

- This LP finds realization plan x with payoff $f^\top v$ for player 1.
- The number of nonzero entries in matrices $E, F.A, B$ is linear in the size of the game tree. Thus, the LPs can be solved in polynomial time with respect to the size of G. This is an exponential improvement!

Using the sequence form to find NE in 2-player games

Using the sequence form to find NE in 2-player games

- A similar idea based on the Duality theorem and complementary slackness, we can also compute NE in 2-player extensive games.

Using the sequence form to find NE in 2-player games

- A similar idea based on the Duality theorem and complementary slackness, we can also compute NE in 2-player extensive games. See the lecture notes for more details.

Using the sequence form to find NE in 2-player games

- A similar idea based on the Duality theorem and complementary slackness, we can also compute NE in 2-player extensive games. See the lecture notes for more details.

Theorem 2.66

A pair (x, y) of realization plans in a 2-player game in the extensive form of perfect recall is NE iff there are vectors u and v such that:

$$\begin{aligned} x^\top(E^\top u - Ay) = 0, &\quad y^\top(F^\top v - B^\top x) = 0, \\ Ex = e, x \geq \mathbf{0}, &\quad Fy = f, y \geq \mathbf{0}, \\ E^\top u - Ay \geq \mathbf{0}, &\quad F^\top v - B^\top x \geq \mathbf{0}. \end{aligned}$$

Using the sequence form to find NE in 2-player games

- A similar idea based on the Duality theorem and complementary slackness, we can also compute NE in 2-player extensive games. See the lecture notes for more details.

Theorem 2.66

A pair (x, y) of realization plans in a 2-player game in the extensive form of perfect recall is NE iff there are vectors u and v such that:

$$x^\top(E^\top u - Ay) = 0, \quad y^\top(F^\top v - B^\top x) = 0,$$
$$Ex = e, x \geq \mathbf{0}, \quad Fy = f, y \geq \mathbf{0},$$
$$E^\top u - Ay \geq \mathbf{0}, \quad F^\top v - B^\top x \geq \mathbf{0}.$$

- This is not an LP.

Using the sequence form to find NE in 2-player games

- A similar idea based on the Duality theorem and complementary slackness, we can also compute NE in 2-player extensive games. See the lecture notes for more details.

Theorem 2.66

A pair (x, y) of realization plans in a 2-player game in the extensive form of perfect recall is NE iff there are vectors u and v such that:

$$\begin{aligned} x^\top(E^\top u - Ay) = 0, &\quad y^\top(F^\top v - B^\top x) = 0, \\ Ex = e, x \geq \mathbf{0}, &\quad Fy = f, y \geq \mathbf{0}, \\ E^\top u - Ay \geq \mathbf{0}, &\quad F^\top v - B^\top x \geq \mathbf{0}. \end{aligned}$$

- This is not an LP. It is the so-called linear complementarity problem.

Using the sequence form to find NE in 2-player games

- A similar idea based on the Duality theorem and complementary slackness, we can also compute NE in 2-player extensive games. See the lecture notes for more details.

Theorem 2.66

A pair (x, y) of realization plans in a 2-player game in the extensive form of perfect recall is NE iff there are vectors u and v such that:

$$x^\top(E^\top u - Ay) = 0, \quad y^\top(F^\top v - B^\top x) = 0,$$
$$Ex = e, x \geq \mathbf{0}, \quad Fy = f, y \geq \mathbf{0},$$
$$E^\top u - Ay \geq \mathbf{0}, \quad F^\top v - B^\top x \geq \mathbf{0}.$$

- This is not an LP. It is the so-called linear complementarity problem.
- These can be solved with Lemke's algorithm,

Using the sequence form to find NE in 2-player games

- A similar idea based on the Duality theorem and complementary slackness, we can also compute NE in 2-player extensive games. See the lecture notes for more details.

Theorem 2.66

A pair (x, y) of realization plans in a 2-player game in the extensive form of perfect recall is NE iff there are vectors u and v such that:

$$\begin{aligned} x^\top(E^\top u - Ay) = 0, &\quad y^\top(F^\top v - B^\top x) = 0, \\ Ex = e, x \geq \mathbf{0}, &\quad Fy = f, y \geq \mathbf{0}, \\ E^\top u - Ay \geq \mathbf{0}, &\quad F^\top v - B^\top x \geq \mathbf{0}. \end{aligned}$$

- This is not an LP. It is the so-called linear complementarity problem.
- These can be solved with Lemke's algorithm, which can take exponentially many steps, similarly to the Lemke–Howson algorithm.

Using the sequence form to find NE in 2-player games

- A similar idea based on the Duality theorem and complementary slackness, we can also compute NE in 2-player extensive games. See the lecture notes for more details.

Theorem 2.66

A pair (x, y) of realization plans in a 2-player game in the extensive form of perfect recall is NE iff there are vectors u and v such that:

$$\begin{aligned} x^\top(E^\top u - Ay) = 0, &\quad y^\top(F^\top v - B^\top x) = 0, \\ Ex = e, x \geq \mathbf{0}, &\quad Fy = f, y \geq \mathbf{0}, \\ E^\top u - Ay \geq \mathbf{0}, &\quad F^\top v - B^\top x \geq \mathbf{0}. \end{aligned}$$

- This is not an LP. It is the so-called linear complementarity problem.
- These can be solved with Lemke's algorithm, which can take exponentially many steps, similarly to the Lemke–Howson algorithm.
- Still, it is exponentially faster than to run the Lemke–Howson algorithm on the induced normal-form game.

- More about games in extensive form + implementation of the algorithms is taught in a lecture by Martin Schmid.

- More about games in extensive form + implementation of the algorithms is taught in a lecture by Martin Schmid.

Novinky.cz

Novinky.cz » Internet a PC » Češi vytvořili umělou inteligenci, která drtí v … Podrubriky: Hardware • Software • Testy • Hry a herní systémy • Mobil

Češi vytvořili umělou inteligenci, která drtí v pokeru jednoho hráče za druhým

3. 3. 2017, 15:44 - mif, Novinky

Facebook Twitter

Vědci z Matematicko-fyzikální fakulty Univerzity Karlovy a Fakulty elektrotechnické ČVUT v Praze pracovali několik posledních měsíců na vývoji umělé inteligence, jejímž hlavním úkolem bude stát se špičkou v karetní hře Poker Texas Hold'em. A to se skutečně podařilo, program porazil hned několik profesionálních hráčů.


Pokerový software DeepStack vytvořili Češi Novinky

- More about games in extensive form + implementation of the algorithms is taught in a lecture by Martin Schmid.

Novinky.cz

Novinky.cz » Internet a PC » Češi vytvořili umělou inteligenci, která drtí v ... Podrubriky: Hardware • Software • Testy • Hry a herní systémy • Mobil

Češi vytvořili umělou inteligenci, která drtí v pokeru jednoho hráče za druhým

3. 3. 2017, 15:44 - mif, Novinky

Facebook Twitter

Vědci z Matematicko-fyzikální fakulty Univerzity Karlovy a Fakulty elektrotechnické ČVUT v Praze pracovali několik posledních měsíců na vývoji umělé inteligence, jejímž hlavním úkolem bude stát se špičkou v karetní hře Poker Texas Hold'em. A to se skutečně podařilo, program porazil hned několik profesionálních hráčů.

Pokerový software DeepStack vytvořili Češi Novinky

Thank you for your attention.