

Lecture 7

Foundations of extensive-form games

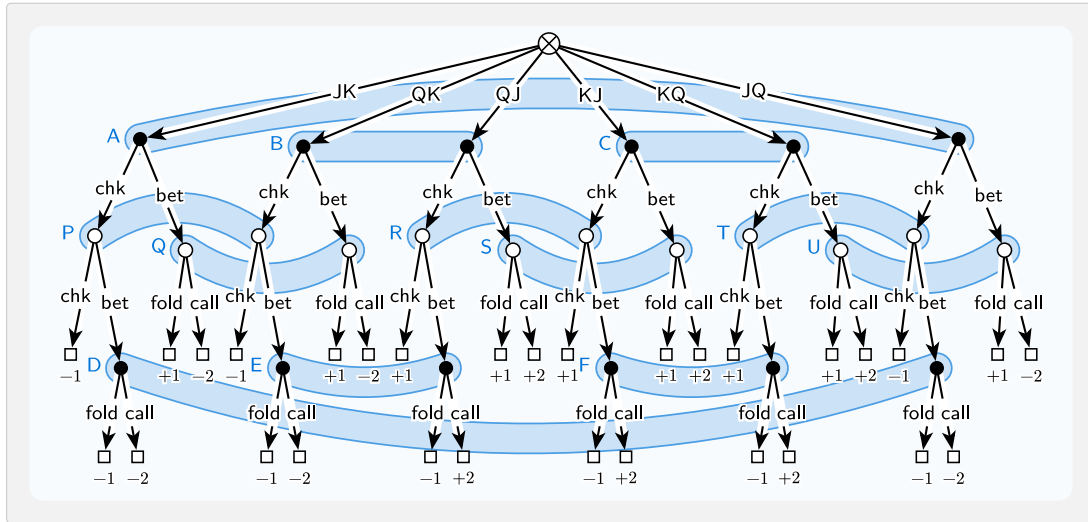
Instructor: Prof. Gabriele Farina (gfarina@mit.edu)

Imperfect-information extensive-form games model tree-form strategic interactions in which not all actions might be observed by all players. They represent an ample majority of strategic interactions encountered in the real world, ranging from recreational games such as poker, to negotiation, and auctions.

L7.1 Game trees and information sets

The standard representation of an imperfect-information extensive-form game is through its *game tree*, which formalizes the interaction of the players as a directed tree. In the game tree, each non-terminal node belongs to exactly one player, who acts at the node by picking one of the outgoing edges (each labeled with an action name). Imperfect information is captured in this representation by partitioning the nodes of each player into sets (called information sets) of nodes that are indistinguishable to that player given his or her observations.

Example L7.1. As a running example, we will illustrate the representation by analyzing the game tree of a simplified two-player variant of poker (perhaps the archetype of imperfect-information extensive-form games), known as *Kuhn poker* [Kuh50]. As noted by Kuhn himself, even the previous small game already captures central aspects of deceptive behavior such as *bluffing* and *underbidding*.



▲ **The rules of Kuhn poker** In the game tree of Kuhn poker, the root history of the tree (the first move in the game) belongs to the *nature player* c . It models a dealer that privately deals one card to each player from a shuffled deck containing cards Jack, Queen, King. The actions of the nature player correspond to the six possible assignments of two cards from the deck, which are annotated on the edges; for example, the leftmost edge JK corresponds to the case in which Player 1 is dealt a Jack and Player 2 is dealt a King. Since the deck is shuffled, each of the six actions are selected with probability $1/6$ by the nature player. No matter the action selected by the dealer, the game transitions to a history of Player 1, which marks the beginning of what in poker is called a “betting round”. First, Player 1 decides to either check (continue without betting any money) or bet \$1. Then,

- If Player 1 checks, Player 2 can either check, or bet \$1.
 - If Player 2 checks, the game terminates with a showdown: the player with the higher card receives from the other player whatever amount the other player bet, plus an ante amount of \$1.
 - If, instead, Player 2 bets the additional \$1, then Player 1 can either fold his hand or call, that is, raise his bet by \$1.
 - If Player 1 folds, he has to give Player 2 only the \$1 ante;
 - if Player 1 calls, a showdown with the same dynamics as before.
- If Player 1 bets the \$1, Player 2 can either fold her hand or call.
 - If Player 2 folds her hand, Player 2 gives Player 1 the \$1 ante.
 - If, instead, Player 2 calls the bet, she increases her bet by \$1 and a showdown occurs, with the same dynamics as before.

L7.1.1 Histories, actions, and payoffs

The game tree represents the strategic interaction of players as a finite directed tree. The nodes of the game tree are called *histories*. Each history that is not a leaf of the game tree is associated with a unique acting player. In an n -player game, the set of valid players is

the set $[n] \cup \{c\} = \{1, \dots, n, c\}$, where c denotes the chance (or nature) player—a fictitious player that selects actions according to a known fixed probability distribution and models exogenous stochasticity of the environment (say, a roll of the dice, or drawing a card from a deck). The player is free to pick any one of the actions available at the history, which correspond to the outgoing edges at the histories. The players keep acting until a leaf of the game tree—called a *terminal history*—is reached. Terminal histories are not associated with any acting player; the set of terminal histories is denoted Z . When a terminal node $z \in Z$ is reached, each player $i \in [n]$ receives a payoff according to the payoff function $u_i : Z \rightarrow \mathbb{R}$.

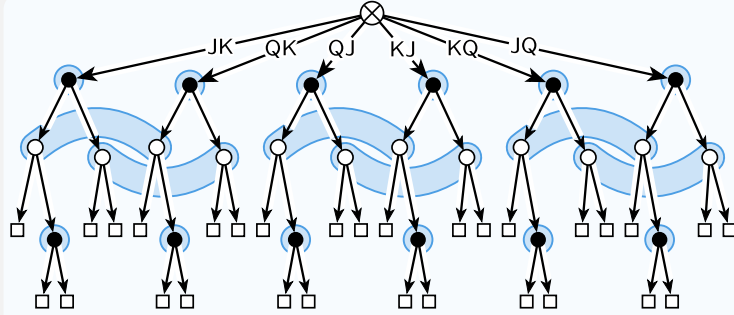
■ L7.1.2 Imperfect information and information sets

To model imperfect information, the histories of each player $i \in [n]$ are partitioned into a collection $\mathcal{I}_i$ of so-called information sets. Each information set $I \in \mathcal{I}_i$ groups together histories that Player i cannot distinguish between when he or she acts there. In the limit case in which all information sets are singleton, the player never has any uncertainty about which history they are acting at, and the game is said to have perfect information. Since a player always knows what actions are available at a decision node, any two histories h, h' belonging to the same information set I must have the same set of available actions. Correspondingly, we can write A_I to denote the set of actions available at any node that belongs to information set I .

Example L7.2. To illustrate how information sets capture private information, in this example we speculate on how different rules for Kuhn poker would translate into different information set structures.

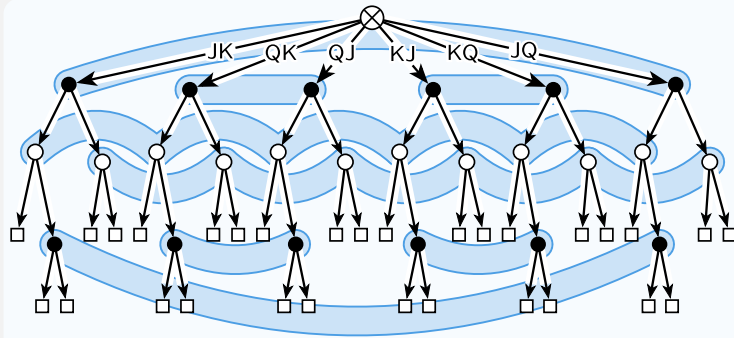
■ First variation

Player 1 is informed of the private card of Player 2 by the dealer.



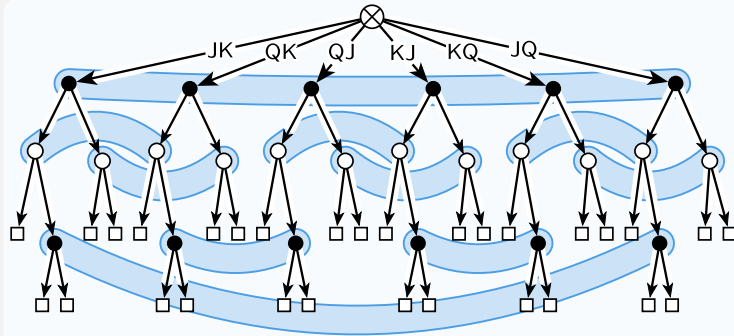
■ Second variation

Player 2 does not get to observe her private card.



■ Third variation

Player 1 is allowed to look at his private card only if he decides to check.



Example L7.3. In Kuhn poker, each player observes their own private card and the actions of the opponent, but not the opponent's private card. The twelve information sets, six for Player 1 denoted A through F , and six for Player 2 denoted P through U , reflect this partial information. For example, Player 1's histories following actions QK and QJ of the nature player (the dealer) are part of the same information set B , in that Player 1 cannot distinguish between the two histories, having observed only their private *Queen* card. As another example, Player 2's information set P captures the uncertainty the player has on the underlying history after having observed a private *King* card, and a check from Player 1.

L7.1.3 Perfect recall

A standard assumption in extensive-form games is that no player forgets about their actions, and about information once acquired. Without this assumption, called *perfect recall*, solving extensive-form games can be intractable. The perfect recall condition can be formalized as follows.

Definition L7.4 (Perfect recall). A player $i \in [n]$ is said to have *perfect recall* if, for any information set $I \in \mathcal{I}_i$, for any two histories $h, h' \in I$ the sequence of Player i 's actions encountered along the path from the root to h and from the root to h' must coincide (or otherwise Player i would be able to distinguish among the histories, since the player remembers all of the actions they played in the past). The game is perfect recall if all players have perfect recall.

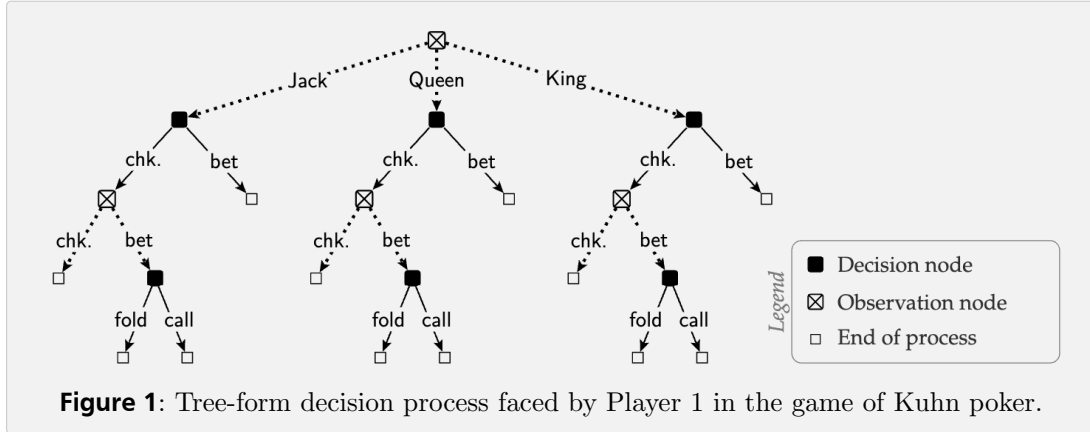
L7.2 Player's perspective: Tree-form decision processes

The game tree representation introduced above provides a description of the global dynamics of the game, without taking the side of any player in particular. But what is the strategy space from the point of view of *one* decision maker (player) in the game? In a normal-form game, the strategy space of a player is a set of probability distributions over the set of actions available to that player. In an extensive-form game, the strategy space of a player is a *tree-form decision process* (TFDP).

Example L7.5 (Player 1's decision process in Kuhn poker). As an example, consider Player 1 in Kuhn poker Example L7.1. From the player's point of view, playing the game could be summarized as follows:

- As soon as the game starts, the player observes a private card that has been dealt to them; the set of possible signals is $\{Jack, Queen, King\}$.
- No matter the card observed, the player now needs to select one action from the set $\{check, bet\}$.
 - If the player *bets*, the player does not have a chance to act further
 - Otherwise, if the player *checks*, the player will then observe whether the opponent *checks* (at which point the interaction terminates) or *bets*. In the latter case, a new decision needs to be made, between *folding* the hand, or *calling* the bet. In either case, after the action has been selected, the interaction terminates.

By arranging the structure of decisions and observations along a tree as follows, we obtain the tree-form decision process for Player 1.



The tree-form decision process lays out the player's opportunities to act. Unlike the game tree, in which each node belongs to one of many players, the tree-form decision process is a directed tree made of only two types of nodes: decision nodes, at which the player must act by picking an action from a set of legal actions, and observation nodes, at which the player does not act but rather observes a signal drawn from a set of possible signals. Furthermore, the information structure of the player, previously defined indirectly through information sets, is captured directly in the TFDP representation.

L7.2.1 Extracting a tree-form decision process from the game tree

In some cases, like in Figure 1, it is straightforward to compile the tree-form decision process faced by a player starting from our intuitive understanding of the game. In this subsection we discuss how the TFDP for the player can be constructed programmatically starting from the game tree when such an understanding is missing. We assume that an n -player imperfect-information extensive-form game with perfect recall and a player $i \in [n]$ of interest, have been fixed.

The set of decision nodes $\mathcal{J}_i$ of the player's TFDP coincides with the set of his or her information sets, that is, $\mathcal{J}_i = \mathcal{I}_i$. This is consistent with the fact that the player cannot condition their behavior on anything other than their information set, given that they cannot distinguish between histories in the same information set. Furthermore, the set of actions available at any decision node $j = I \in \mathcal{J}_i$ coincides with the set of actions A_I available at any history in information set I .

Example L7.6. Consider Kuhn poker from the point of view of Player 1 (Figure 1).

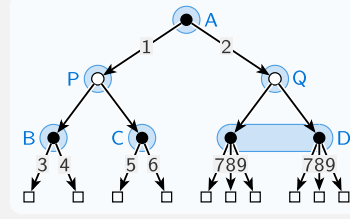
- The trace of any history in A is the sequence (A) .
- The trace of any history in E is the sequence (B, check, E) .

From the point of view of Player 2, the trace of any history in R is the sequence (R) .

Example L7.7.

Consider the following small game tree.

Taking the side of Player 1, the trace of the only history in B is the sequence $(A, 1, B)$, the trace of any history in D is $(A, 2, D)$, and the trace of the only history in A is (A) . Taking the side of Player 2, the trace of the only history in P is (P) , and the trace of the only history in Q is (Q) .



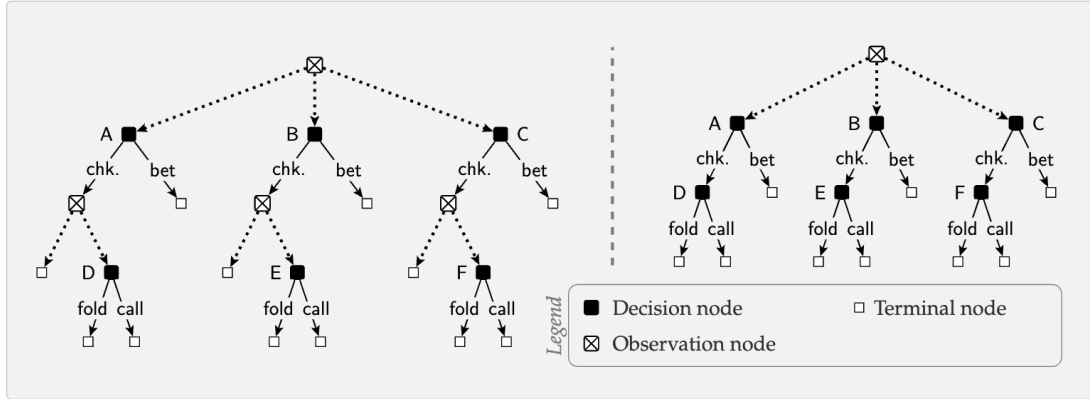
Traces implicitly encode a notion of partial chronological ordering between information sets, of which the player has recall—see Definition L7.4. Hence, for the TFDP of Player i to be an accurate representation of the decision process the player faces while playing the game, it is necessary that *traces of the information sets are the same in the game tree and in the TFDP*. In other words, we require that decision points in the TFDP be structured so as to satisfy that the trace of any information set I matches the sequence of information sets and actions encountered from the root of the TFDP to decision node I .

Definition L7.8 (Tree-form decision process). Fix the game tree of an n -player imperfect information game, and a player $i \in [n]$. A *tree-form decision process (TFDP)* for Player i is a directed rooted tree made of *decision*, *observation*, and *terminal nodes*, satisfying the following properties.

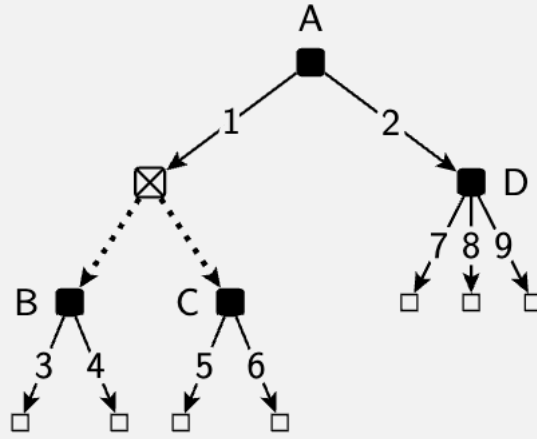
- The set of decision nodes $\mathcal{J}_i$ of the TFDP is equal to the set $\mathcal{I}_i$ of information sets.
- The set of actions available at each decision node $j = I \in \mathcal{I}_i$ (i.e., the set of outgoing edges from the decision node) is equal to the set of actions A_I available at any history $h \in I$ in the game tree.
- Given any decision node $j = I \in \mathcal{I}_i$, the sequence of decision nodes and actions encountered from the root of the TFDP to j is equal to the trace of any history $h \in I$.

As a remark, Definition L7.8 leaves the labeling and structure of observation nodes unspecified. In fact, a player might have multiple TFDPs that satisfy the definition, and differ in how the observation nodes are placed. We illustrate this in the next example.

Example L7.9. The following are both valid TFDPs capturing Player 1's decision process when playing Kuhn poker (Figure 1).



Example L7.10. A valid TFDP representing the decision process of Player 1 in the small game of Example L7.7 is shown below.



L7.2.2 Some notation

Trees always require a bit of notation to be handled properly. We introduce some notation that will be useful when discussing decision problems faced by players in tree-form games.

Decision and observation nodes, transition function:

- We denote the set of decision nodes in the TFDP as $\mathcal{J}$, and the set of observation nodes as $\mathcal{K}$. At each decision node $j \in \mathcal{J}$, the player selects an action from the set A_j of available actions. At each observation node $k \in \mathcal{K}$, the player observes a signal s from the environment out of a set of possible signals S_k .
- We denote ρ the transition function of the process. Picking action $a \in A_j$ at decision node $j \in \mathcal{J}$ results in the process transitioning to $\rho(j, a) \in \mathcal{J} \cup \mathcal{K} \cup \{\perp\}$, where $\perp$ denotes the end of the decision process. Similarly, the process transitions to $\rho(k, s) \in \mathcal{J} \cup \mathcal{K} \cup \{\perp\}$ after the player observes signal $s \in S_k$ at observation node $k \in \mathcal{K}$.

■ Sequences:

- A pair (j, a) where $j \in \mathcal{J}$ and $a \in A_j$ is called a non-empty sequence. The set of all non-empty sequences is denoted as $\Sigma_* := \{(j, a) : j \in \mathcal{J}, a \in A_j\}$. For notational convenience, we will often denote an element (j, a) in Σ as ja without using parentheses, especially when used as a subscript.
- The symbol $\emptyset$ denotes a special sequence called the empty sequence. The set of all sequences, including the empty one, is denoted Σ .
- Given a decision node $j \in \mathcal{J}$, we denote by p_j its *parent sequence*, defined as the last sequence (that is, decision point-action pair) encountered on the path from the root of the decision process to j . If the player does not act before j (that is, j is the root of the process or only observation nodes are encountered on the path from the root to j), we let $p_j = \emptyset$.
- Given a sequence $\sigma \in \Sigma$, we denote with C_σ the set of decision nodes j whose parent sequence is σ : $C_\sigma := \{j \in \mathcal{J} : p_j = \sigma\}$.

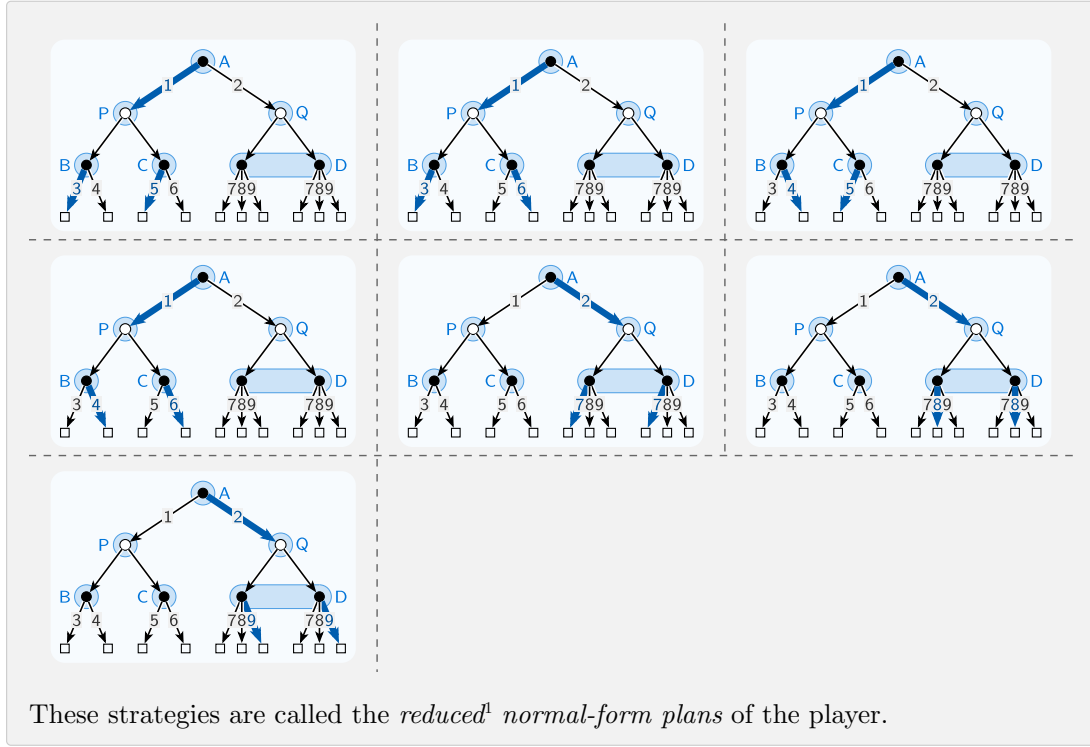
■ L7.3 Strategy representations in extensive-form games

What does it mean to have a mixed strategy for an extensive-form game? We discuss ways in which one could decide to represent a strategy, and contrast their pros and cons.

■ L7.3.1 Strategic form: Extensive-form games as normal-form games

One classical answer is the following. Consider a player, and imagine enumerating all their deterministic strategies for the tree. A mixed strategy is then a probability distribution over these deterministic strategies.

Example L7.11. In the small game of Example L7.7, a mixed strategy for Player 1 is a probability distribution over the following 7 strategies $\pi_1, \dots, \pi_7$.



By considering the normal-form game in which each player's strategy space is the set of all deterministic strategies in the tree, we have converted the extensive-form game into its *normal-form equivalent*.

Cons Of course, the most glaring issue with this representation is that the number of strategies in the normal-form game is exponential in the size of the game tree. For this reason, it was long believed that operating with this normal-form representation was computationally infeasible. Historically, this led to the development of specialized algorithms for extensive-form games, such as CFR and its variants. This belief was actually unfounded. In fact, it is possible to simulate the OMWU dynamics in the normal-form equivalent of an extensive-form game exactly in polynomial time. We will talk more about this in the future.

Pros By virtue of this conversion, the machinery and concepts of normal-form games can be applied to extensive-form games, such as Nash equilibria, correlated equilibria, and so on.

L7.3.2 Behavioral form

A different conceptualization of a strategy for a player is as a choice of (independent) distributions over the set of actions A_j at each decision node $j \in \mathcal{J}$. This is called a *behavioral strategy*. We can represent it accordingly as a vector $x \in \mathbb{R}_{\geq 0}^{\Sigma}$ indexed over sequences. Each entry x_{ja} assigns to action a at decision node j the probability of picking that action at

¹The term “reduced” refers to the fact that no actions are specified for parts of the games not reached due to decisions of the player in higher parts of the game tree.

that decision node. The set of all possible behavioral strategies is clearly convex, as it is the Cartesian product of probability simplexes—one per each decision node.

Cons However, this representation has a major drawback: the probability of reaching a particular terminal state in the decision process is the product of all actions on the path from the root to the terminal state. This makes many expressions of interest that depend on the probability of reaching terminal states (including crucially the expected utility in the game) non-convex.

Example L7.12. Consider the game of Kuhn poker, and let x, y be behavioral strategies for both players. The expected utility function for Player 1 is given by

$$u_1(x, y) := (-1) \cdot x_{A,chk} \cdot y_{P,chk} + (-1) \cdot x_{A,chk} \cdot y_{P,bet} \cdot x_{D,fold} \\ + (-2) \cdot x_{A,chk} \cdot y_{P,bet} \cdot x_{D,call} + \dots$$

This is not a convex function of x , as it contains products of entries of x .

Pros One might wonder whether behavioral strategies have the same representational power as normal-form strategies. After all, a normal-form strategy is an object in a much larger space, so it is conceivable that “more might be possible” in the strategic form. A positive answer equating the powers of normal-form and behavioral strategies is given by Kuhn’s theorem, which requires that the game is perfect recall.

Theorem L7.13 (Kuhn’s theorem). In a perfect recall extensive-form game, any distribution over terminal nodes that can be induced via normal-form strategies can be induced via behavioral strategies, and *vice versa*.

L7.3.3 Sequence form

The *sequence-form representation* [KMv96; Rom62; von96] soundly resolves the issue of non-convexity. Like behavioral strategies, in the sequence-form representation a strategy is a vector $x \in \mathbb{R}_{\geq 0}^{\Sigma}$ whose entries are indexed by Σ . However, the generic entry x_{ja} contains the *product* of the probabilities of all actions at all decision nodes on the path from the root of the process to action a at decision node j . In order to be a valid sequence-form strategy, the entries in x must therefore satisfy the following probability-flow-conservation constraints:

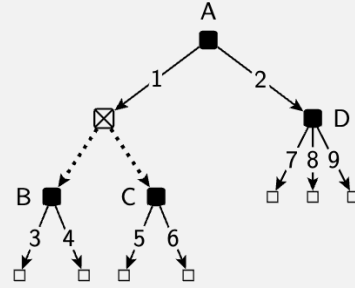
Definition L7.14. The polytope of sequence-form strategies of a TFDP is the convex polytope

$$\mathcal{Q} := \left\{ x \in \mathbb{R}_{\geq 0}^{\Sigma} : \quad x_{\emptyset} = 1, \quad \sum_{a \in A_j} x_{ja} = x_{p_j} \quad \forall j \in \mathcal{J} \right\}.$$

Conversely, it is easy to see that any x that satisfies the above constraints is the sequence-form representation of at least one behavioral strategy.

Example L7.15. Consider the tree-form decision process faced by Player 1 in the small game of Example L7.7. The decision process has four decision nodes $J = \{A, B, C, D\}$ and nine sequences including the empty sequence $\emptyset$. For decision node D, the parent sequence is $p_D = A2$; for B and C it is $p_B = p_C = A1$; for A it is the empty sequence $p_A = \emptyset$. The constraints that define the sequence-form polytope Definition L7.14, besides nonnegativity, are

$$\begin{cases} x_{\emptyset} = 1 \\ x_{A1} + x_{A2} = x_{\emptyset} \\ x_{B3} + x_{B4} = x_{A1} \\ x_{C5} + x_{C6} = x_{A1} \\ x_{D7} + x_{D8} + x_{D9} = x_{A2}. \end{cases}$$



Pros This representation of extensive-form games is convex. Hence, we can use all the convex optimization tools we have seen so far.

L7.4 Bibliography for this lecture

- [Kuh50] H. W. Kuhn, “A Simplified Two-Person Poker,” in *Contributions to the Theory of Games*, vol. 1, H. W. Kuhn and A. W. Tucker, Eds., in Annals of Mathematics Studies, 24, vol. 1., Princeton, New Jersey: Princeton University Press, 1950, pp. 97–103.
- [Rom62] I. Romanovskii, “Reduction of a Game with Complete Memory to a Matrix Game,” *Soviet Mathematics*, vol. 3, 1962.
- [KMv96] D. Koller, N. Megiddo, and B. von Stengel, “Efficient Computation of Equilibria for Extensive Two-Person Games,” *Games and Economic Behavior*, vol. 14, no. 2, 1996.
- [von96] B. von Stengel, “Efficient Computation of Behavior Strategies,” *Games and Economic Behavior*, vol. 14, no. 2, pp. 220–246, 1996.