15-888 Computational Game Solving (Fall 2021) Thu, September 9th 2021

Lecture 2

Representation of strategies in tree-form decision spaces

Instructor: Gabriele Farina*

Consider a one-shot game like playing rock-paper-scissor. A strategy for each player in that game simply corresponds to a probability distribution over the available actions. So, the set of strategies for a player in a one-shot decision making problem corresponds to the probability simplex Δ^n, where n is the number of actions available to that player.

In this lecture, we are going to see how strategies are represented in the more complicated case of *tree-form* decision making. Abstractly, we are looking for a representation that would guarantee the following properties: (i) the set of strategies is a convex and compact set; and (ii) the utility function of each player as a function of the representation of strategies is multilinear. The two properties we just listed enable the application of efficient optimization methods in the context of extensive-form games; for one, they are enough to guarantee that a Nash equilibrium in a two-player zero-sum game be written as a bilinear saddle point problem.

1 Tree-form decision making

In a *tree-form sequential decision process (TFSDP)* problem the agent interacts with the environment in two ways: at *decision points*, the agent must act by picking an action from a set of legal actions; at *observation points*, the agent observes a signal drawn from a set of possible signals. Different decision points can have different sets of legal actions, and different observation points can have different sets of possible signals. Decision and observation points are structured as a *tree*: under the standard assumption that the agent is not forgetful, so, it is not possible for the agent to cycle back to a previously encountered decision or observation point by following the structure of the decision problem. TFSDPs provide a general formalism which captures the decision problem faced by a player of an extensive-form game with perfect recall (for example, poker or bridge), as well as Markov decision processes and partially-observable Markov decision processes for which the agent conditions its policy on the entire history of observations and actions.

As an example, consider the simplified game of *Kuhn poker* [Kuhn, 1950], depicted in Figure 1. Kuhn poker is a standard benchmark in the EFG-solving community. In Kuhn poker, each player puts an ante worth 1 into the pot. Each player is then privately dealt one card from a deck that contains 3 unique cards (Jack, Queen, King). Then, a single round of betting then occurs, with the following dynamics. First, Player 1 decides to either check or bet 1. Then,

- If Player 1 checks Player 2 can check or raise 1.
  - If Player 2 checks a showdown occurs; if Player 2 raises Player 1 can fold or call.
    - If Player 1 folds Player 2 takes the pot; if Player 1 calls a showdown occurs.
- If Player 1 raises Player 2 can fold or call.
  - If Player 2 folds Player 1 takes the pot; if Player 2 calls a showdown occurs.

When a showdown occurs, the player with the higher card wins the pot and the game immediately ends.

As soon as the game starts, the agent observes a private card that has been dealt to them; this is observation point k_1, whose set of possible signals is $S_{k_1} \coloneqq \{\text{jack}, \text{queen}, \text{king}\}$. Should the agent observe

*Computer Science Department, Carnegie Mellon University. ✉ gfarina@cs.cmu.edu.

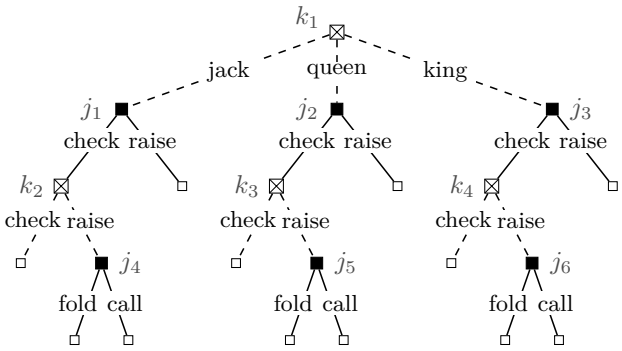


Figure 1: Tree-form sequential decision making process of the first acting player in the game of Kuhn poker.

the 'jack' signal, the decision problem transitions to the decision point j_1, where the agent must pick one action from the set $A_{j_1} \coloneqq \{\text{check}, \text{raise}\}$. If the agent picks 'raise', the decision process terminates; otherwise, if 'check' is chosen, the process transitions to observation point k_2, where the agent will observe whether the opponent checks (at which point the interaction terminates) or raises. In the latter case, the process transitions to decision point j_4, where the agent picks one action from the set $A_{j_4} \coloneqq \{\text{fold}, \text{call}\}$. In either case, after the action has been selected, the interaction terminates.

When the interaction terminates, the agent will receive a form of feedback, typically a payoff. However, in this first part of the course, we will only focus on the set of strategies in the decision process, and not its payoffs. We will reconnect the geometry of strategies with the details of the feedback in the second part of the document.

The example above already reveals some of the notation we will use throughout the document. We now introduce additional symbols, which are also summarized in Table 1.

Symbol	Description
$\mathcal{J}$	Set of decision points
A_j	Set of legal actions at decision point $j \in \mathcal{J}$
$\mathcal{K}$	Set of observation points
S_k	Set of possible signals at observation point $k \in \mathcal{K}$
Σ	Set of sequences, defined as $\Sigma \coloneqq \{(j, a) : j \in \mathcal{J}, a \in A_j\}$
p_j	Parent sequence of decision point $j \in \mathcal{J}$, defined as the last sequence (decision point-action pair) on the path from the root of the TFSDP to decision point j; if the agent does not act before j, $p_j = \varnothing$

Table 1: Summary of notation in TFSDPs.

- We denote the set of decision points in the TFSDP as $\mathcal{J}$, and the set of observation points as $\mathcal{K}$. At each decision point $j \in \mathcal{J}$, the agent selects an action from the set A_j of available actions. At each observation point $k \in \mathcal{K}$, the agent observes a signal s_k from the environment out of a set of possible signals S_k.
- A pair (j, a) where $j \in \mathcal{J}$ and $a \in A_j$ is called a *sequence*. The set of all sequences is denoted as $\Sigma \coloneqq \{(j, a) : j \in \mathcal{J}, a \in A_j\}$. For notational convenience, we will often denote an element (j, a) in Σ as ja without using parentheses, especially when used as a subscript.
- Given a decision point $j \in \mathcal{J}$, 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 agent does not act before j (that is, j is the root of the process or only observation points are encountered on the path from the root to j), we let $p_j = \varnothing$.

From extensive-form games to TFSDPs The primary difference between a TFSDP and an extensive-form game is that the former encodes the decision problem faced by a *single agent*, while the latter directly encodes the dynamics of the interaction for all agents involved. In extensive-form games, each decision node of the underlying game tree belongs to a player in the game. Observation nodes correspond to *stochastic* actions, and are traditionally though of as being decision points for a fictitious player called the *nature* (or *chance*) player.

There is a one-to-one correspondence between decision points in the TFSDP formalism and information sets in the extensive-form game formalism. Similarly, there exists a one-to-one correspondence between sequences in the TFSDP formalism and (information set, action)-pairs in the extensive-form game formalism. In other words, given an EFG extracting the TFSDP faced by any of the players is a mechanical task. Figure 2 provides a small example.

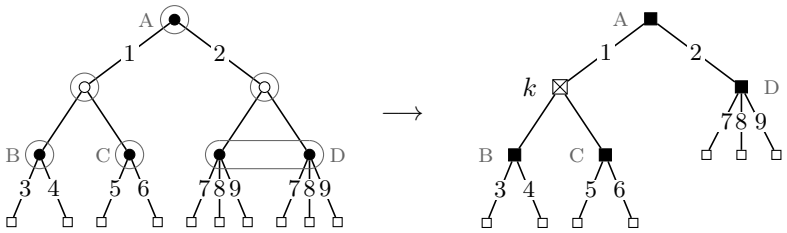


Figure 2: (Left) Example of a small extensive-form game. Black nodes belong to Player 1, White nodes belong to Player 2. The gray partitions represent the information sets of the game. (Right) Sequential decision making problem faced by Player 1 in the small extensive-form game on the left.

2 Strategies in tree-form decision making

Conceptually, a strategy for an agent in an TFSDP specifies a distribution over the set of actions A_j at each decision point $j \in \mathcal{J}$.

2.1 Behavioral strategies

Perhaps, the most intuitive representation of a strategy, called a *behavioral strategy*, is as a vector $\boldsymbol{x} \in \mathbb{R}^{|\Sigma|}_{\geq 0}$ indexed over sequences that assigns at each action a at decision point j the probability of picking that action at that decision point. The set of all possible behavioral strategies is clearly convex, as it is the Cartesian product of probability simplexes—one per each decision point. That 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 expected utilities in extensive-form games) non-convex.

2.2 Sequence-form strategies

To avoid the issue of non-convexity, throughout this document we will almost exclusively use a different representation, called the *sequence-form representation* [Romanovskii, 1962, Koller et al., 1996, von Stengel, 1996]. In the sequence-form representation, a strategy is a vector $\boldsymbol{x} \in \mathbb{R}^{|\Sigma|}_{\geq 0}$ whose entries are indexed by Σ. However, the entry $\boldsymbol{x}[ja]$ contains the *product* of the probabilities of all actions at all decision points on the path from the root of the process to action a at decision point j. In order to be a valid sequence-form strategy, the entries in $\boldsymbol{x}$ must satisfy the probability-mass-conservation constraints:

$$\sum_{a \in A_j} \boldsymbol{x}[ja] = \boldsymbol{x}[p_j]$$

at every "non-root" decision point j (i.e., $p_j \neq \varnothing$), and the constraint $\sum_{a \in A_j} \boldsymbol{x}[ja] = 1$ at every "root" decision point j.

So, we have the following definition.

Definition 2.1. The *polytope of sequence-form strategies* of a TFSDP is the convex polytope

$$Q \coloneqq \left\{ \boldsymbol{x} \in \mathbb{R}_{\geq 0}^{|\Sigma|} : \quad \sum_{a \in A_j} \boldsymbol{x}[ja] = \begin{cases} 1 & \text{if } p_j = \varnothing \\ \boldsymbol{x}[p_j] & \text{otherwise} \end{cases} \quad \forall j \in \mathcal{J} \right\}.$$

Because the constraints in Definition 2.1 are linear, the set of all valid sequence-form strategies is a convex polytope.

Remark 2.1. To avoid breaking into cases depending on whether $p_j = \varnothing$ or not, a popular alternative definition of sequence-form strategies includes the special element $\varnothing$ as an index of the strategy vector, with the fixed value 1. In that case, the set of sequence-form strategies could be rewritten as

$$\left\{ \boldsymbol{x} \in \mathbb{R}_{\geq 0}^{|\Sigma \cup \{\varnothing\}|} : \quad \boldsymbol{x}[\varnothing] = 1, \quad \sum_{a \in A_j} \boldsymbol{x}[ja] = \boldsymbol{x}[p_j] \quad \forall j \in \mathcal{J} \right\}.$$

We do not follow that approach in this lecture, as it makes the inductive characterization of Q (Section 2.4) less clean.

2.3 Deterministic and randomized sequence-form strategies

Out of the polytope of all possible strategies in the decision process, *deterministic* strategies are extremely important. Deterministic strategies are strategies that select exactly one action at each decision point, without ever randomizing the choice. In other words, deterministic strategies assign probability either 0 or 1 to each action at each decision point. Hence, when using the sequence-form representation, the set of all deterministic strategies—denoted Π—corresponds to the subset of Q whose components are 0 or 1.

Definition 2.2. The set of *deterministic* sequence-form strategies is the set

$$\Pi \coloneqq Q \cap \{0, 1\}^{|\Sigma|}.$$

The set of deterministic sequence-form strategies corresponds one-to-one to the concept of *reduced normal-form strategies* in game theory. Because of this connection, a direct consequence of the classical Kuhn's theorem [Kuhn, 1953] is the following.

Theorem 2.1. The set of deterministic sequence-form strategies generates the polytope of sequence-form strategies, that is, $Q = \operatorname{co} \Pi$.

2.4 Bottom-up decomposition of the polytope of sequence-form strategies

The polytope of sequence-form strategies (Definition 2.1) has a strong combinatorial structure that enables speeding up several common optimization procedures. In this subsection we will explore that combinatorial structure by giving a *bottom-up* decomposition based on two standard convexity-preserving operations: convex hulls and Cartesian products.

We illustrate this intuitively by means of a small example.

Example 2.1. Consider the small sequential decision making problem of Figure 3. As our construction is bottom-up, we will denote with the symbol Q_v, with $v \in \mathcal{J} \cup \mathcal{K}$, the partial sequence-form strategy space corresponding to the subtree rooted at v.

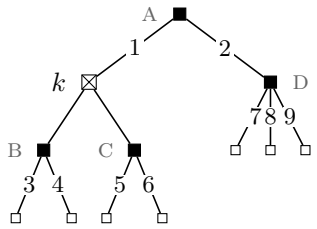


Figure 3: Sequential decision making problem used in the example.

We can characterize the set of sequence-form strategies as follows:

- At the terminal decision points $j = \text{B}, \text{C}, \text{D}$, Q_j is a probability simplex. Specifically, $Q_{\text{B}} = Q_{\text{C}} = \Delta^2$ and $Q_{\text{D}} = \Delta^3$.
- At the (only) observation point k, a strategy for the subtree rooted at k must provide independent strategies for the subtrees rooted at B and C. So,

  $$Q_k = Q_{\text{B}} \times Q_{\text{C}}$$

  is the Cartesian product of the strategy spaces for the subtrees rooted B and C.
- At the decision point A, we need to first pick a probability distribution of play for the two actions (sequences 1 and 2). Let's call the probabilities assigned to those actions as λ_1 and λ_2, respectively. Clearly, $(\lambda_1, \lambda_2) \in \Delta^2$. Once the probabilities of sequences 1 and 2 are chosen, strategies for the subtrees rooted at the observation point k and D. Since sequence-form strategies associate to each sequence σ the product of the probabilities of all actions on the path form the root of the TFSDP to σ, the set of all valid sequence-form strategies for the subtree rooted in A (that is, the whole TFSDP) is

  $$\begin{aligned} Q_{\text{A}} &= \left\{(\lambda_1, \lambda_2, \lambda_1 \boldsymbol{x}_k, \lambda_2 \boldsymbol{x}_{\text{D}}) : (\lambda_1, \lambda_2) \in \Delta^2, \boldsymbol{x}_k \in Q_k, \boldsymbol{x}_{\text{D}} \in Q_{\text{D}}\right\} \\ &= \text{co}\left\{ \begin{pmatrix} 1 \\ 0 \\ \hline Q_k \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ \hline \mathbf{0} \\ Q_{\text{D}} \end{pmatrix} \right\}. \end{aligned}$$

The above approach can be used in any TFSDP. In particular, we always have that the sequence-form strategy space of a subtree rooted at an observation point is the Cartesian product of the sequence-form strategy spaces of the children subtrees. Furthermore, the sequence-form strategy space of a subtree rooted at

a decision point is the convex hull of the sequence-form strategy spaces of the children subtrees, augmented with the indicators of the actions. These inductive rules are summarized in Table 2.

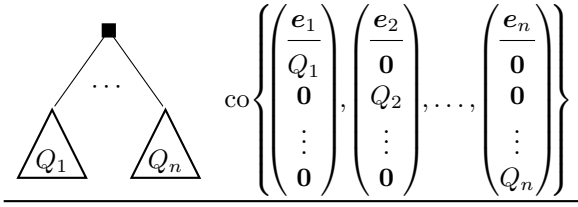
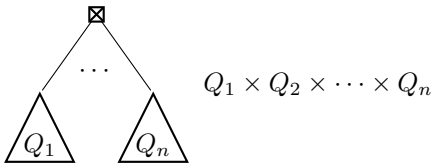
Decision point	**Observation point**
	

Table 2: Bottom-up construction rules for sequence-form strategy spaces. $\boldsymbol{e}_i$ denotes the i-th indicator vector, that is, the vector whose entries are all 0 except for the entry in position i, which is set to 1.

References

H. W. Kuhn. A simplified two-person poker. In H. W. Kuhn and A. W. Tucker, editors, *Contributions to the Theory of Games*, volume 1 of *Annals of Mathematics Studies, 24*, pages 97–103. Princeton University Press, Princeton, New Jersey, 1950.

I. Romanovskii. Reduction of a game with complete memory to a matrix game. *Soviet Mathematics*, 3, 1962.

Daphne Koller, Nimrod Megiddo, and Bernhard von Stengel. Efficient computation of equilibria for extensive two-person games. *Games and Economic Behavior*, 14(2), 1996.

Bernhard von Stengel. Efficient computation of behavior strategies. *Games and Economic Behavior*, 14(2): 220–246, 1996.

H. W. Kuhn. Extensive games and the problem of information. In H. W. Kuhn and A. W. Tucker, editors, *Contributions to the Theory of Games*, volume 2 of *Annals of Mathematics Studies, 28*, pages 193–216. Princeton University Press, Princeton, NJ, 1953.