

Extensive-Form Games and Counterfactual Regret Minimization



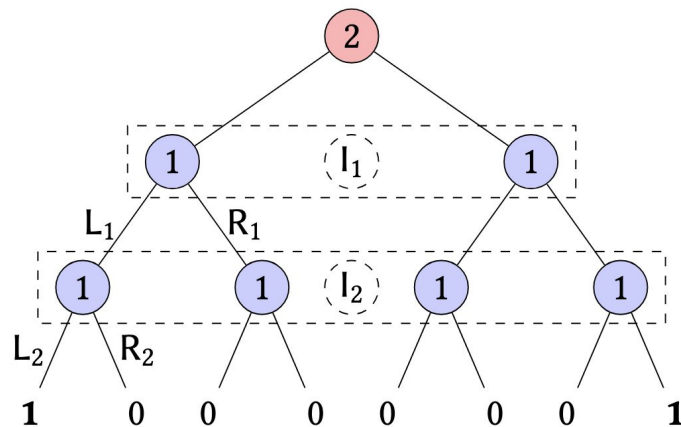
15 888 **Computational Game Solving** (Fall 2026)
Ioannis Anagnostides

Today's lecture

- Extensive-form games
 - Imperfect information and perfect recall
 - Representing strategies
 - Mixed strategies
 - Behavioral strategies
 - Sequence-form strategies
- Tree-form decision problems
 - Inductive decomposition of the strategy set
- Counterfactual regret minimization
 - Regret circuits

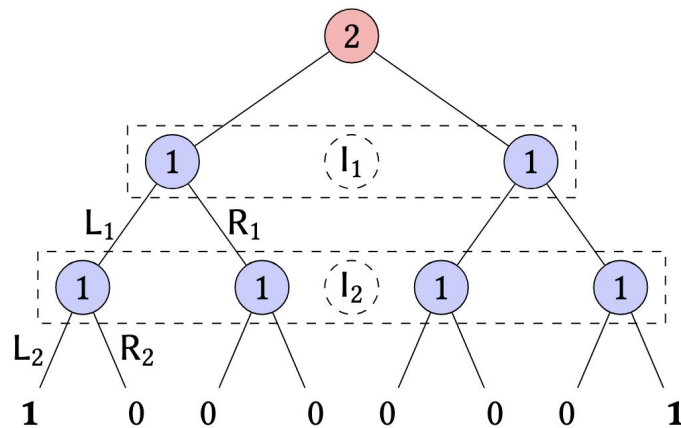
Extensive-form games

- Represented through a rooted **game tree**
- Each node is either a *decision node* or a *leaf (terminal) node*
- Each decision node belongs to a player, who selects an action linking to new node
- Payoffs are given at the terminal nodes
- The nodes of each player are partitioned into **information sets**
- An information set contains all nodes that *cannot be distinguished* by that player



Perfect recall

- A player has **perfect recall** if it never forgets previously acquired information
- For all nodes in the same information set, the *sequence* of previous information sets and actions should coincide
- If the sequence differed, a perfect-recall player could distinguish the nodes
- The game on the right has imperfect recall



Converting to normal form

- A **pure strategy** is a mapping from information sets to actions at those information sets
- A **mixed strategy** is a distribution over pure strategies
- One can create an equivalent normal-form game with actions being the set of pure strategies
- How large is the induced normal-form game?

Converting to normal form

- A **pure strategy** is a mapping from information sets to actions at those information sets
- A **mixed strategy** is a distribution over pure strategies
- One can create an equivalent normal-form game with actions being the set of pure strategies
- How large is the induced normal-form game?
- The issue is that the number of pure strategies *scales with the product* of the information sets  Combinatorial blow up!

Desiderata from an optimization standpoint

When the rest of the players are fixed, we need to be able to *optimize one's utility efficiently*.

- The set of strategies needs to be a compact **convex polytope**
- The utility function needs to be **linear**—or at least concave—in that player's strategy

Desiderata from an optimization standpoint

When the rest of the players are fixed, we need to be able to *optimize one's utility efficiently*.

- The set of strategies needs to be a compact **convex polytope**
- The utility function needs to be **linear**—or at least concave—in that player's strategy

For mixed strategies

Behavioral strategies

- Operating over mixed strategies is prohibitive: the dimension of that set is exponential in the size of the game tree
- A **behavioral strategy** maps information sets to *distributions* over the actions
- We treat each information set separately: we employ *uncorrelated randomization* between information sets
- Does that limit our expressivity?

Behavioral strategies

- Operating over mixed strategies is prohibitive: the dimension of that set is exponential in the size of the game tree
- A **behavioral strategy** maps information sets to *distributions* over the actions
- We treat each information set separately: we employ *uncorrelated randomization* between information sets
- Does that limit our expressivity?



No, for perfect-recall games!

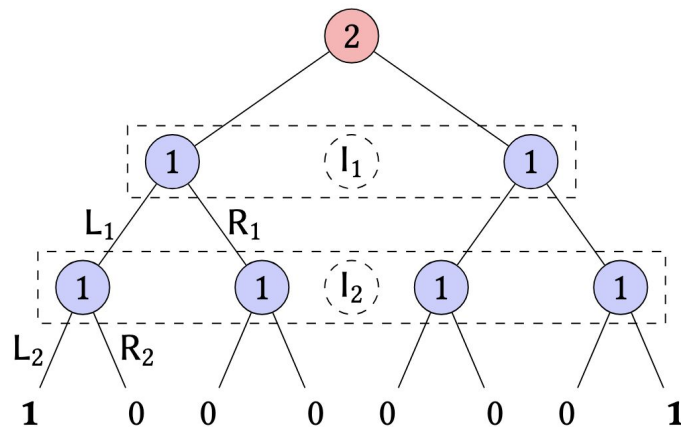
Theorem (Kuhn). For any mixed strategy, there is a behavioral strategy that is utility-equivalent to that mixed strategy no matter the strategies of the rest of the players.



Mixed strategies can be useful even under perfect-recall

Mixed strategies can be superior under imperfect recall

- The conclusion of Kuhn's theorem does **not** hold without perfect recall
- In the game on the right, there is a *mixed strategy* that gets a utility of 0.5
- But *any behavioral strategy* gets at most 0.25 when Player 2 is minimizing the utility of Player 1



The problem with behavioral strategies

- Unlike mixed strategies, behavioral strategies have a compact representation
- But there is still a basic problem
- Let's look at the utility function
- It contains **products** of the same player's strategy
- This is very much **nonlinear**/nonconcave

$$\sum_{z \in \mathcal{Z}} u_i(z) p_c(z) \prod_{i' \in [n]} \prod_{\substack{(h,a) \preceq z \\ h \in \mathcal{H}_{i'}}} b_{i'}[(j, a)].$$

Desiderata from an optimization standpoint

When the rest of the players are fixed, we need to be able to *optimize one's utility efficiently*.

- The set of strategies needs to be a compact **convex polytope**
- The utility function needs to be **linear**—or at least concave—in that player's strategy

For behavioral strategies

Try #3: sequence-form representation

- We apply the basic transformation $\mathbf{x}_i[\sigma] := \prod_{(j,a) \in \sigma} \mathbf{b}_i[(j,a)]$
- A vector is a *sequence-form strategy* if and only if it obeys *probability flow conservation*

$$\mathcal{X}_i := \left\{ \mathbf{x}_i \in \mathbb{R}_{\geq 0}^{\Sigma_i} : \sum_{a \in \mathcal{A}_j} \mathbf{x}_i[(j,a)] = \begin{cases} 1 & \text{if } p_j = \emptyset \\ \mathbf{x}_i[p_j] & \text{otherwise.} \end{cases} \quad \forall j \in \mathcal{J}_i \right\}$$



We have a compact representation

Desiderata from an optimization standpoint

When the rest of the players are fixed, we need to be able to *optimize one's utility efficiently*.

- The set of strategies needs to be a compact **convex polytope**
- The utility function needs to be **linear**—or at least concave—in that player's strategy

For seq.-form strategies

LP for zero-sum games in extensive form

In sequence form, we are dealing again with a bilinear optimization problem

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} x^\top A y$$

Fixing the strategy of P1,

$$\begin{array}{ll} \text{maximize} & x^\top A y \\ \text{subject to} & F_2 y = f_2, \\ & y \geq 0. \end{array}$$

Dual

$$\begin{array}{ll} \text{minimize} & f_2^\top v \\ \text{subject to} & A^\top x \geq F_2^\top v. \end{array}$$

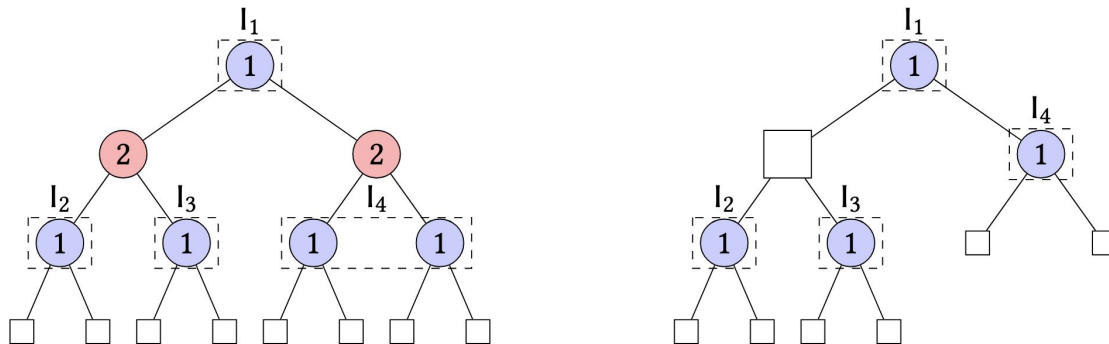
Sequence form

It suffices to solve the following LP:

$$\begin{array}{ll} \text{minimize} & f_2^\top v \\ \text{subject to} & F_1 x = f_1, \\ & x \geq 0, \\ & A^\top x \geq F_2^\top v. \end{array}$$

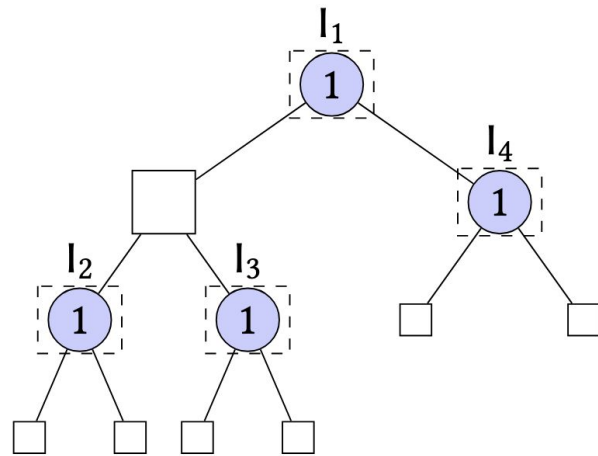
Tree-form decision problems

- Taking the perspective of a single player, we can abstract away all other players
- The player faces a tree-form decision problem
- We have either **decision** nodes or **observation** nodes
- The player *acts* at a decision node and observes a *signal* at observation nodes



Inductive decomposition of strategies

- We proceed in a *bottom-up* fashion
- We start from the terminal decision points, where each strategy set is a probability simplex



Observation nodes can be decomposed using a Cartesian product

$$\mathcal{X}_k = \mathcal{X}_{p_1} \times \mathcal{X}_{p_2} \times \cdots \times \mathcal{X}_{p_v}$$

Decision nodes can be decomposed using a convex hull

$$\mathcal{X}_j := \text{co} \left\{ \begin{pmatrix} \mathbf{e}_1 \\ \mathcal{X}_{p_1} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} \mathbf{e}_2 \\ \mathbf{0} \\ \mathcal{X}_{p_2} \\ \vdots \\ \mathbf{0} \end{pmatrix}, \dots, \begin{pmatrix} \mathbf{e}_v \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathcal{X}_{p_v} \end{pmatrix} \right\}$$

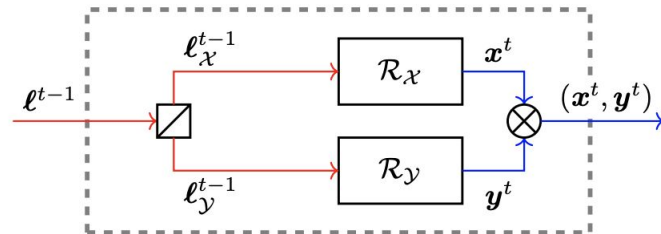
Regret circuits

- We know how to minimize regret over the *simplex* (RM, MWU,...)
- How to compose multiple such regret minimizers to tackle more complex sets, such as the sequence-form polytope?
- We will use the framework of **regret circuits** (Farina et al., 2019)
- Because of the previous decomposition, it's enough to handle
 - *Cartesian products*
 - *Convex hulls*

Cartesian product

Regret minimization over a Cartesian product easily decomposes into independent subproblems

- The next strategy is just the *concatenation* of the individual strategies
- Each utility is split into components and then forwarded to the individual algorithms

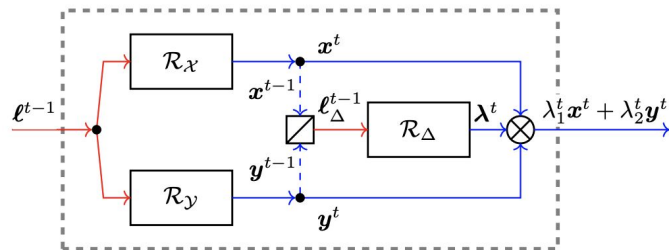


Taken from Farina et al.

Convex hull

Convex hull is relatively more involved

- We require a regret minimizer for *mixing* over the sets
 - Each action keeps track of a different regret minimizer
- The next strategy is the *mixture* of the individual strategies
- The feedback of each action reflects how well the corresponding regret minimizer is doing



Taken from Farina et al.

Counterfactual regret minimization

NEXTSTRATEGY():

for each decision point $j \in \mathcal{J}$ **do**

$\Delta(\mathcal{A}_j) \ni b_j^{(t)} := \mathfrak{R}_j.\text{NEXTSTRATEGY}();$

for each decision point $j \in \mathcal{J}$ **in top-down order do**

for each action $a \in \mathcal{A}_j$ **do**

if $p_j = \emptyset$ **then**

$\mathbf{x}^{(t)}[(j, a)] := b_j^{(t)}[a];$

else

$\mathbf{x}^{(t)}[(j, a)] := \mathbf{x}^{(t)}[p_j] \cdot b_j^{(t)}[a];$

return $\mathbf{x}^{(t)} \in \mathbb{R}^\Sigma;$

OBSERVEUTILITY($\mathbf{u}^{(t)} \in \mathbb{R}^\Sigma$):

$V^{(t)}[\perp] := 0;$

for each node in the tree $p \in \mathcal{J} \cup \mathcal{K}$ **in bottom-up order do**

if $p \in \mathcal{J}$ **then**

 Let $j = v;$

$V^{(t)}[j] := \sum_{a \in \mathcal{A}_j} b_j^{(t)}[a] \cdot (\mathbf{u}^{(t)}[(j, a)] + V^{(t)}[\rho(j, a)]);$

else

 Let $k = p;$

$V^{(t)}[k] := \sum_{s \in \mathcal{S}_k} V^{(t)}[\rho(k, s)];$

for each decision point $j \in \mathcal{J}$ **do**

for each action $a \in \mathcal{A}_j$ **do**

$\mathbf{u}_j^{(t)}[a] := \mathbf{u}^{(t)}[(j, a)] + V^{(t)}[\rho(j, a)];$

$\mathfrak{R}_j.\text{OBSERVEUTILITY}(\mathbf{u}_j^{(t)});$

Remarks on CFR

- It was introduced by Zinkevich et al. (2007)
- It is based on the notion of *counterfactual utilities*
- It is a *family* of algorithms: there are different ways of instantiating the local regret minimizers
- By far the most common choice is RM and its modern variants
- More on that in the next lecture