6.S890: Topics in Multiagent Learning

Lecture 15 – Prof. Farina

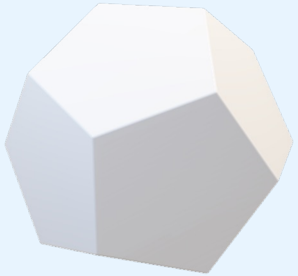
Learning in Extensive-Form Games (Part II)

And Equilibrium Perfection

Fall 2023

Important Facts for Extensive-Form Games

FACT: There exists a representation of strategies in the tree, called **sequence-form strategies**, such that:



- The strategy set is a convex polytope
- The utility of each player is linear in the player's strategy

$\Longrightarrow$: Computing a Nash equilibrium in a two-player zero-sum extensive-form game can be written as

$$\max_{x \in Q_1} \min_{y \in Q_2} x^T A y$$

Sequence-form strategy polytope of players

Sequence-form payoff matrix of the game
(See Lecture 13 for how to compute)

Important Facts for Extensive-Form Games

⟹: Computing a Nash equilibrium in a two-player zero-sum extensive-form game can be written as

$$\max_{x \in Q_1} \min_{y \in Q_2} x^T A y$$

Sequence-form payoff matrix of the game

Sequence-form strategy polytope of players

⟹: As seen in Lecture 13, we can use Linear Programming to solve for Nash equilibrium in two-player zero sum games

As discussed in the previous lecture, **we can also use learning** (more scalable both in theory and in practice)

Quiz: what is learning and how do we use it in games?

Q: What is a no-external-regret algorithm?

X = Simplex for normal-form games

X = sequence-form polytope for extensive-form games

Objective: sublinear (external) regret

$$R^{(T)} := \max_{\hat{x} \in X} \sum_{t=1}^{T} \langle u^{(t)}, \hat{x} - x^{(t)} \rangle$$

Building a regret minimizer means making sure this bound holds, **NO MATTER THE SEQUENCE OF UTILITIES GIVEN TO THE LEARNER**

Q: How do we use no-external-regret algorithms in two-player zero-sum normal-form or extensive-form games?

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

X, Y = Simplex for normal-form games

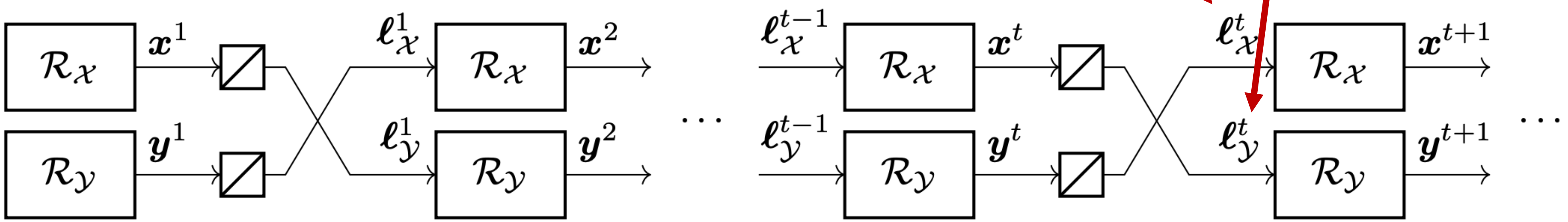
X, Y = sequence-form polytope for extensive-form games

Answer: we let the learners play against each other

Q: What utilities do we supply to the learners?

$$\boldsymbol{\ell}^t_{\mathcal{X}} := \boldsymbol{A}\boldsymbol{y}^t, \qquad \boldsymbol{\ell}^t_{\mathcal{Y}} := -\boldsymbol{A}^\top \boldsymbol{x}^t$$

(Gradients of the players' utility functions)



If we can build a no-external-regret algorithm for outputting sequence-form strategies, then we can use it to compute a Nash equilibrium in two-player zero-sum games (and more)

Q: Other uses of no-external-regret algorithms?

- Q: what happens if we use a no-external-regret algorithm against static opponents (opponents that play from a fixed strategy)?
  - A: The average strategy of the no-external-regret algorithm converges to a best response to the opponents
- Q: what equilibrium do we recover if all players play according to a no-external-regret algorithm against each other in a general-sum multi-player game?
  - A: The average play converges to the set of coarse-correlated equilibria

How can we construct a no-external-regret algorithm for extensive-form games?

No-Regret Algorithms for EFGs

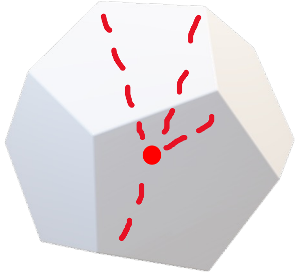
Different conceptual approaches exist:

Exploits structure of problem and specific learning algorithm

Less specialized; general tool

- Conversion to a single simplex of convex combinations of vertices
- Decomposition into local decision problem over actions at each decision point
- Use general convex optimization tools (e.g., FTRL)

Main idea:



Change of variables: instead of picking a point in the strategy polytope, decide how to mix the vertices

Key question:

How to sidestep exponential size?

Kernelized Multiplicative Weights Update

No-Regret Algorithms for EFGs

Different conceptual approaches exist:

Exploits structure of problem and specific learning algorithm

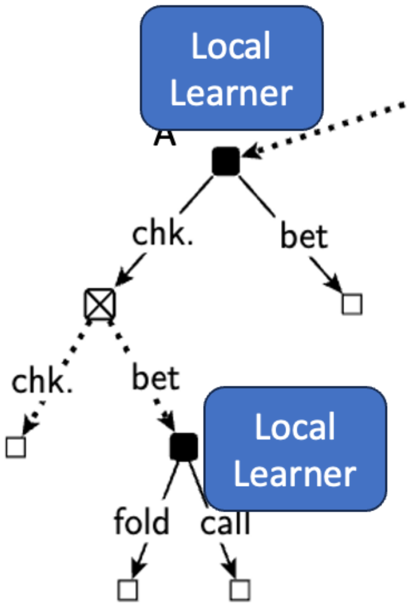
Conversion to a single simplex of convex combinations of vertices

Decomposition into local decision problem over actions at each decision point

Use general convex optimization tools (e.g., FTRL)

Less specialized; general tool

Main idea:



Key question:

What is the local feedback?

Run a local no-regret algorithm at each decision point to update your strategy.

”Process” the utility vector $u^{(t)}$ (which is for the whole sequence-form strategy) and chop it up into local feedback for each decision point.

Counterfactual Regret Minimization

No-Regret Algorithms for EFGs

Different conceptual approaches exist:

Exploits structure of problem and specific learning algorithm

Conversion to a single simplex of convex combinations of vertices

Decomposition into local decision problem over actions at each decision point

Use general convex optimization tools (e.g., FTRL)

Less specialized; general tool

Key question:

What regularizers are easy to deal with?

Main idea:

The sequence-form polytope is a convex set. So, we can apply the FTRL algorithm in its general form, and that guarantees no-regret

$$x^{(t)} = \arg\max_{x \in Q} \langle U^{(t)}, x \rangle - \frac{1}{\eta}\varphi(x)$$

?

Counterfactual Regret Minimization

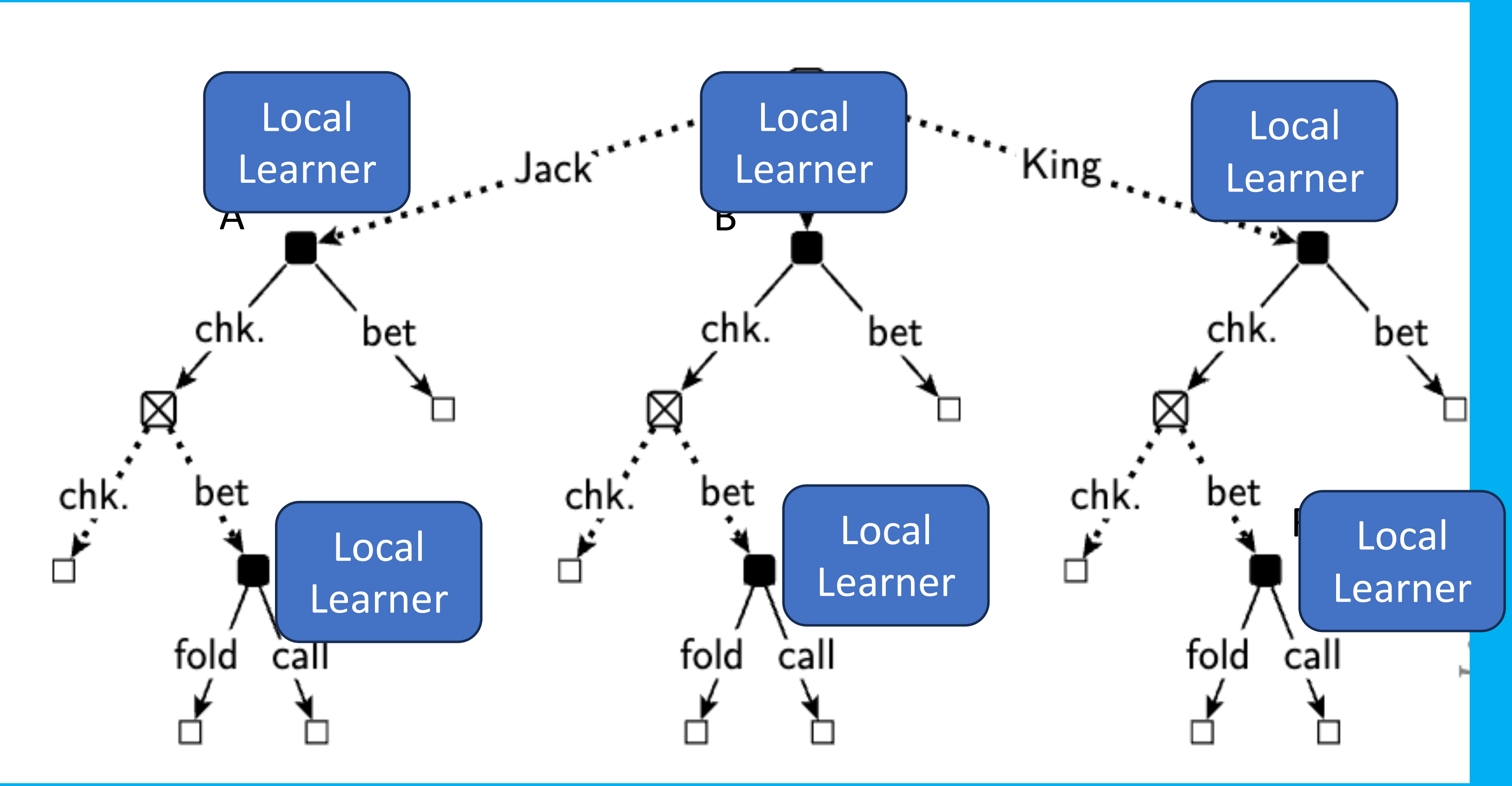
Idea: Minimize regret **globally** on the tree
by **thinking locally** at each decision point

🚨
Papercut Alert™

CFR updates strategies in *behavioral* form…

…but is a no-external-regret algorithm for *sequence-form strategies*

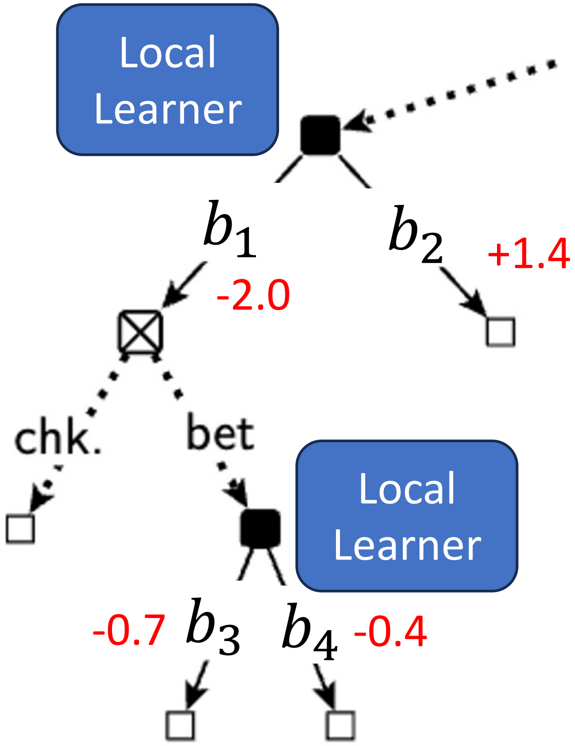
Big Picture Idea:



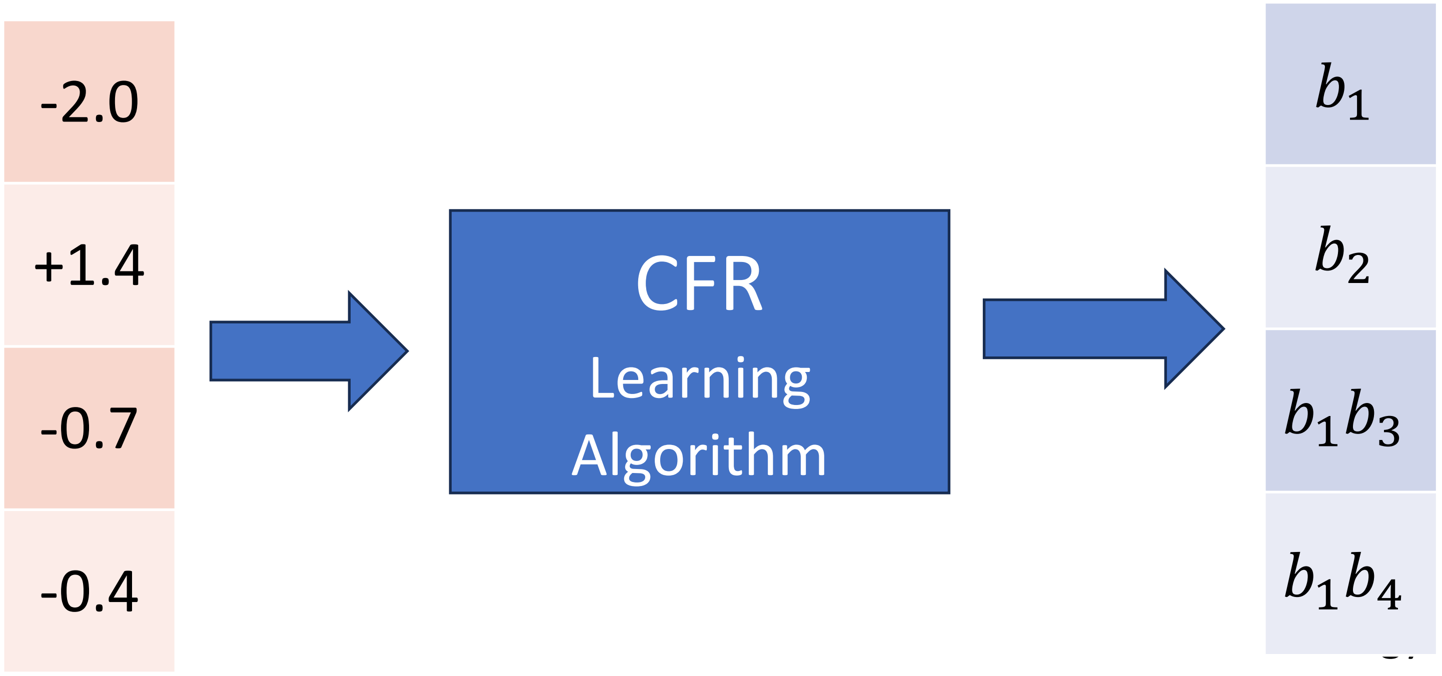
Each local learner is responsible for refining the **behavior** at their decision point

Can locally use regret matching, multiplicative weights update, …

Remember: we are trying to construct a no-external-regret minimizer. Our algorithm must guarantee sublinear regret no matter the sequence of utilities!



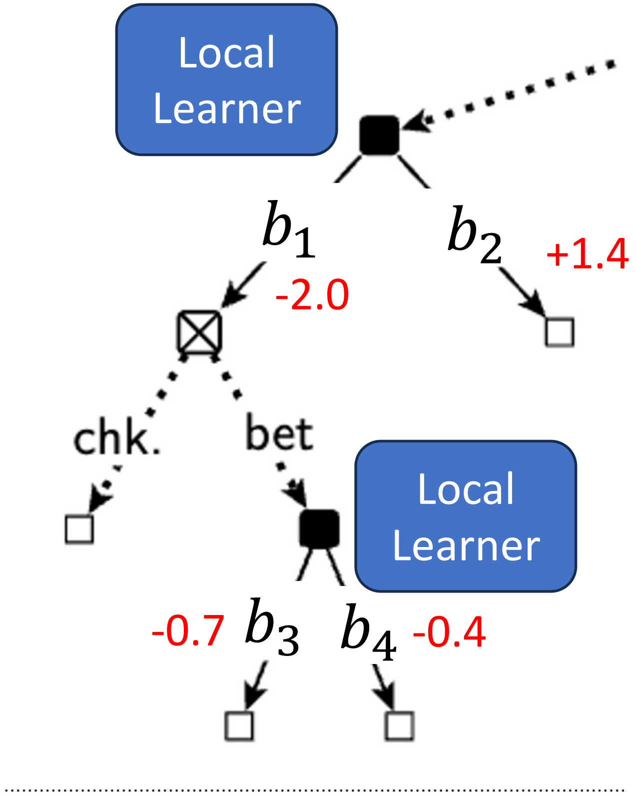
Main question: what utility to pass to the local learners?



Utility vector (for sequence-form strategy)

Strategy (in sequence form)

Counterfactual Utilities



Give to each local learner the **expected utility in the subtree** rooted at each action:

$$\widehat{u_3} = -0.7$$
$$\widehat{u_4} = -0.4$$
$$\widehat{u_2} = +1.4$$
$$\widehat{u_1} = -2.0 + b_3 \cdot (-0.7) + b_4 \cdot (-0.4)$$

Why does it work?

- Proof time!

Regret bound

- Theorem: the regret cumulated by CFR can be bounded as

$$R_{CFR}^{(T)} \le \sum_j \max\left\{0, R_j^{(T)}\right\}$$

Decision points

Local regret cumulated by learner at j

- **Therefore**: if the local regret minimizers all have regret $O(\sqrt{T})$, then CFR has regret $O(\sqrt{T})$ (where the O hides game-dependent constants)

Therefore: if both players in a zero-sum extensive-form game play according to CFR, the average strategy converges to Nash equilibrium at rate $O(1/\sqrt{T})$

FTRL in Extensive-Form Games

Follow-the-Regularized-Leader

$$x^{(t)} = \arg\max_{x \in \mathrm{Q}} \langle U^{(t)}, x \rangle - \frac{1}{\eta}\varphi(x)$$

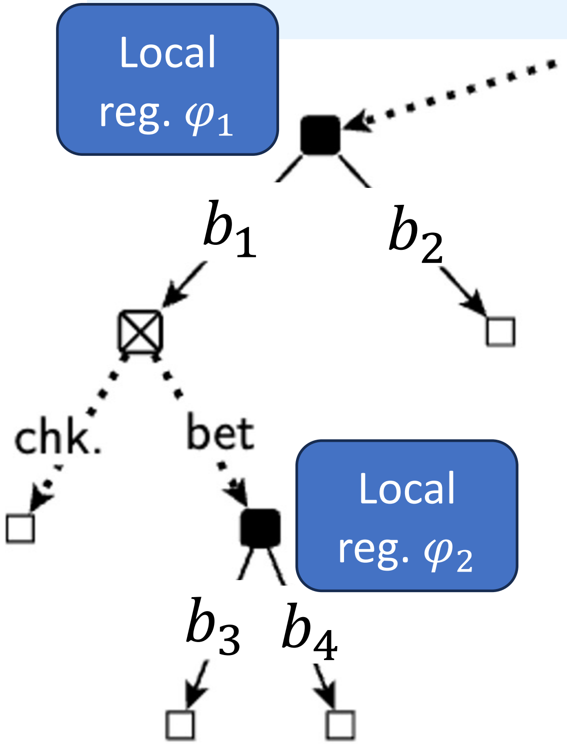
Depending on the choice of strongly convex regularizer φ, solving the step above might be impractical

Example: if φ is the squared Euclidean distance, then the solution can be found in polynomial time **but it is complicated and expensive in practice**! (hence, not a popular approach…)

Efficient Regularizers

Only a high level intuition. Good to know they exist, but they don't perform nearly as well as CFR!

Idea: construct regularizers that mimic the structure of the tree-form decision problem



$$x = \begin{bmatrix} b_1 \\ b_2 \\ b_1 b_3 \\ b_1 b_4 \end{bmatrix}$$

Strategy (in sequence form)

Dilated regularizers

$$\varphi(x) \coloneqq \varphi_1(b_1, b_2) + b_1 \cdot \varphi_2(b_3, b_4)$$

Where f_1 and f_2 are local strongly convex regularizers (e.g., negative entropy)

It can be shown that φ is strongly convex, and the solution to the FTRL problem can be computed in a bottom-up fashion

Overall: kernelization gives better **theoretical bounds** on the regret

CFR gives better empirical performance (beats top poker pros)

FTRL is technically possible, but nobody has figured out how to make it work well in practice

For large games, regret-based methods are today the scalable state of the art

We can use the techniques we discussed to compute some Nash equilibrium in any two-player zero-sum game

We can use the techniques we discussed to compute some Nash equilibrium in any two-player zero-sum game

Are all Nash equilibria equally good? Or should we aim for some but not other?

Not all Nash equilibria are equally sensible, especially in sequential games!

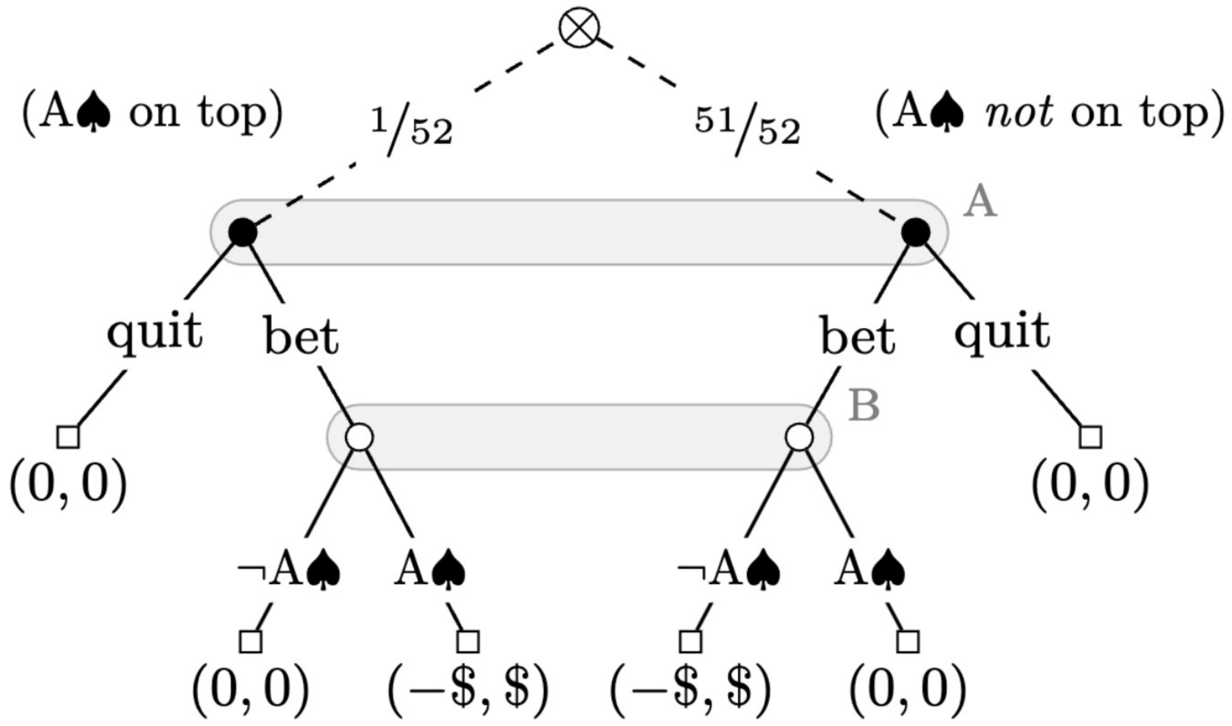
Intuition: Nash equilibria stem from the idea that the opponent is as strong as possible, and might therefore be completely unprepared to handle the case of an imperfect opponent

Very relevant when playing with humans!

Guess-the-Ace game

To make the discussion more concrete, consider the following game (due to Miltersen and Sorensen)

- At the start a standard 52-card deck is perfectly shuffled, face down, by a dealer
- Then, Player 1 decides whether to immediately end the game (no money transfer), or offer $1000 to Player 2 if they can correctly guess whether the top card of the shuffled deck is the ace of spaces or not.
- If Player 2 guesses correctly, the $1000 get transferred from Player 1 to Player 2; if not, no money is transferred

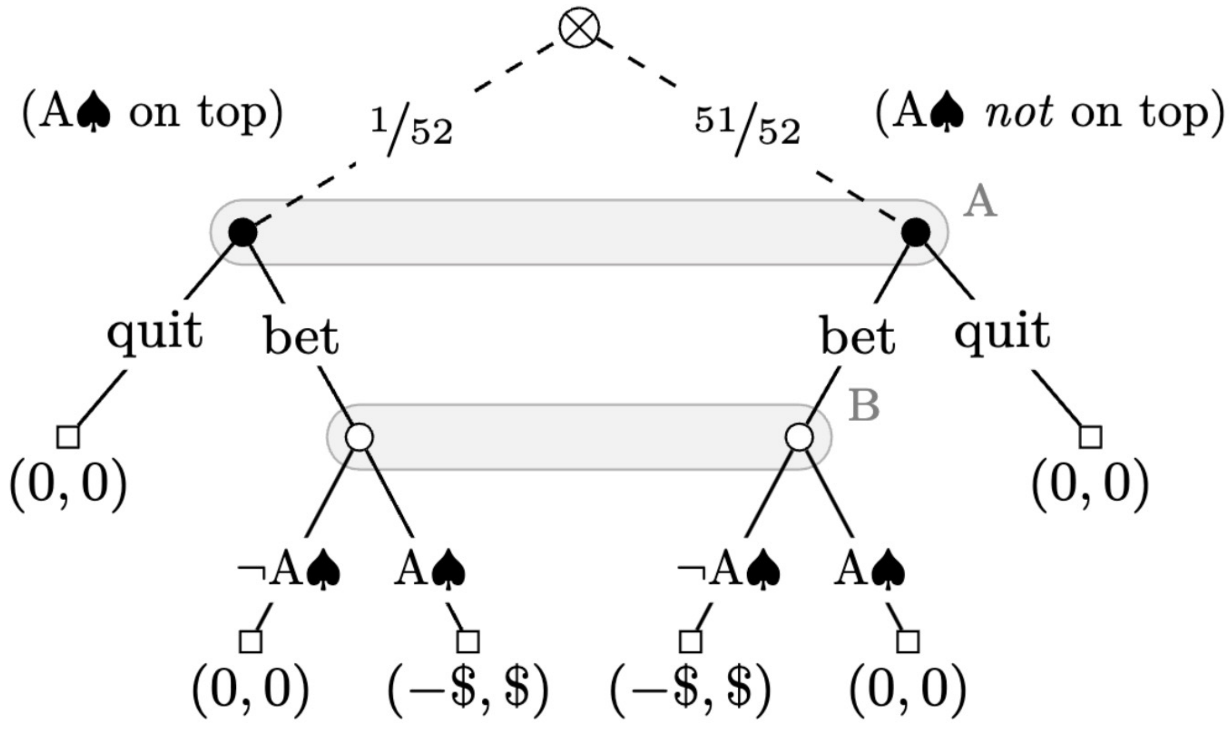


Guess-the-Ace game

To make the discussion more concrete, consider the following game (due to Miltersen and Sorensen)

- At the start a standard 52-card deck is perfectly shuffled, face down, by a dealer
- Then, Player 1 decides whether to immediately end the game (no money transfer), or offer $1000 to Player 2 if they can correctly guess whether the top card of the shuffled deck is the ace of spaces or not.
- If Player 2 guesses correctly
  trans
  mo

Q: As Player 1, what is the only sensible way to play the game?



Answer: the only sensible thing for Player 1 to do is to quit immediately
(anything else loses money to Player 1 in expectation)

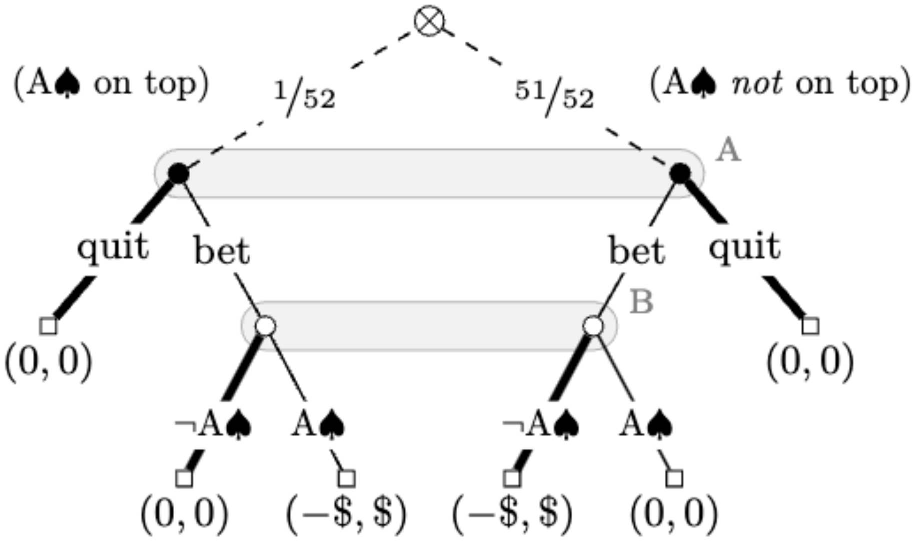
Indeed, that is the only Nash equilibrium strategy for Player 1.

Guess-the-Ace game

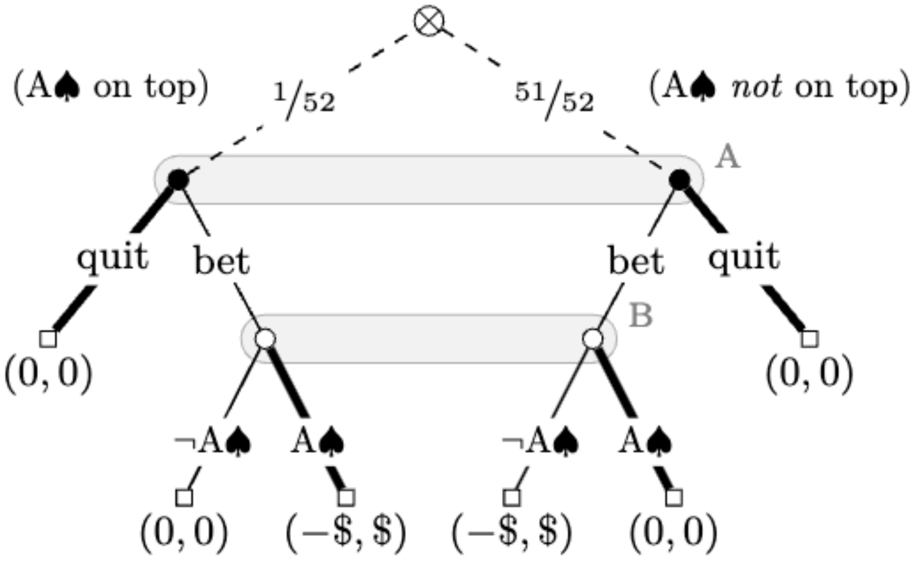
But then, Player 2 does not get to play. From the point of view of the definition of Nash equilibrium, anything that Player 2 does is a Nash equilibrium strategy

Yet, huge difference between the strategies. Only one of the two approaches can be called "rational"

Both of these are Nash equilibria. Nash eq. does not distinguish between the two



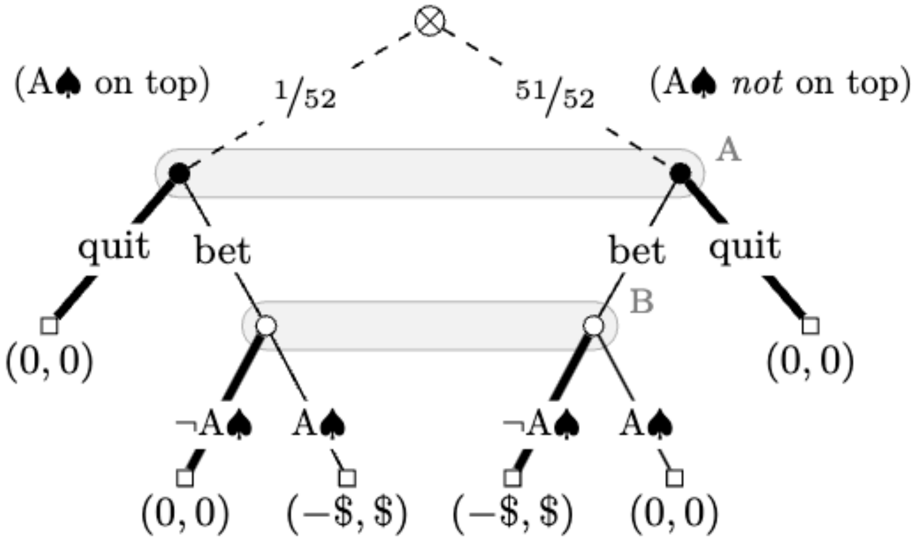
Sensible Nash equilibrium



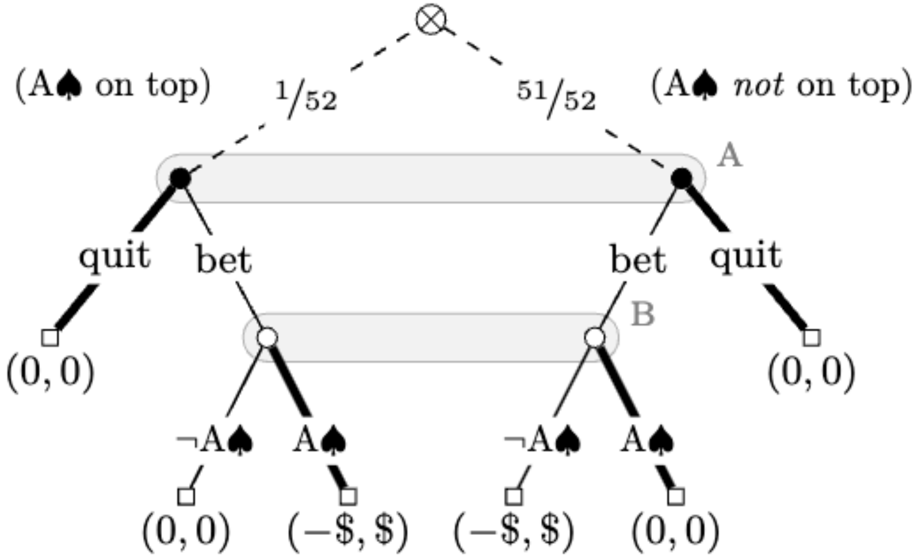
Questionable Nash equilibrium

Imagine that Player 2 is a **bot** playing against opponents in the real world, blindly following the Nash equilibrium strategy it has precomputed

If Player 1 makes a mistake and decides to offer the $1000 instead of immediately quitting, the Nash equilibrium that bets that the top card is not the ace of space has an expected utility of > $980 whereas the Nash equilibrium that bets that the top card is the ace of spades only has an expected utility of < $20.

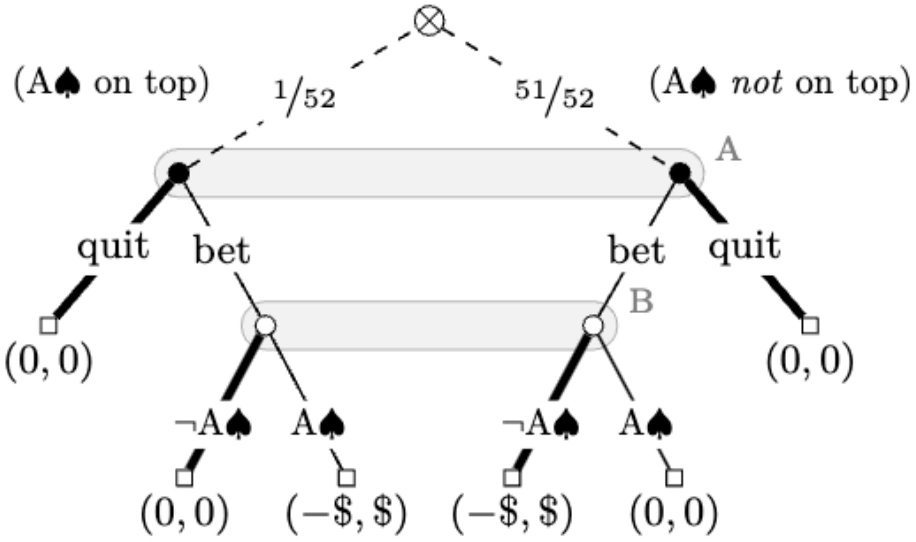


Sensible Nash equilibrium

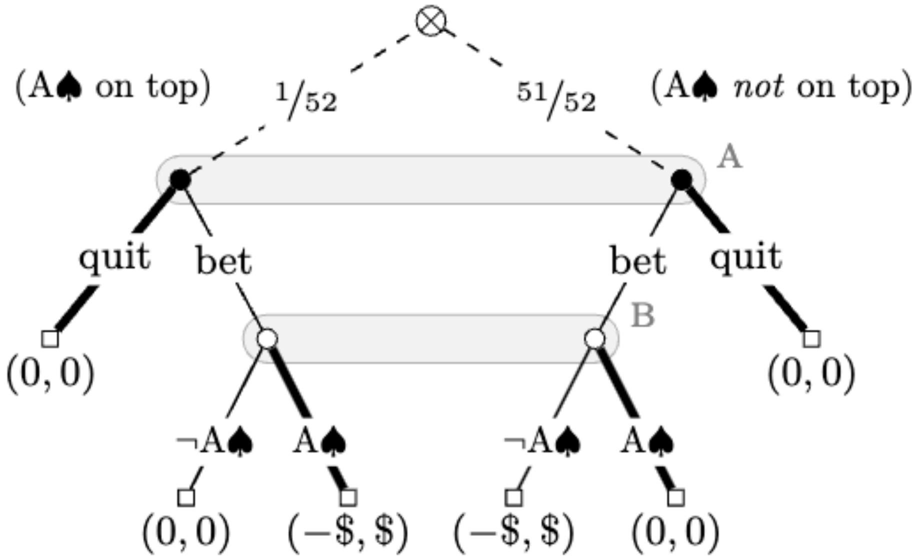


Questionable Nash equilibrium

Formalizing this subtle notion of rationality within the set of Nash equilibria has been a major endeavor for the game-theoretic literature in the 70s and 80s. Today, we say that the equilibrium in Figure 1 (Left) is **sequentially irrational**, while the one on the right is sequentially rational.



Sensible Nash equilibrium



Questionable Nash equilibrium

Formalizing this subtle notion of rationality within the set of Nash equilibria has been a major endeavor for the game-theoretic literature in the 70s and 80s. Today, we say that the equilibrium in Figure 1 (Left) is **sequentially irrational**, while the one on the right is sequentially rational.

Not all Nash equilibria are equally "good" when the agents can make mistakes.

Sequentially-irrational Nash equilibria might leave value on the table, by being incapable of capitalizing on opponents' mistakes

Trivia: this kind of surprising behavior kicked in during the poker tournament with the pros, and people were worried there was possibly a bug in the bot. Instead, it was likely the pro that had made a mistake and entered an off-equilibrium part of the tree