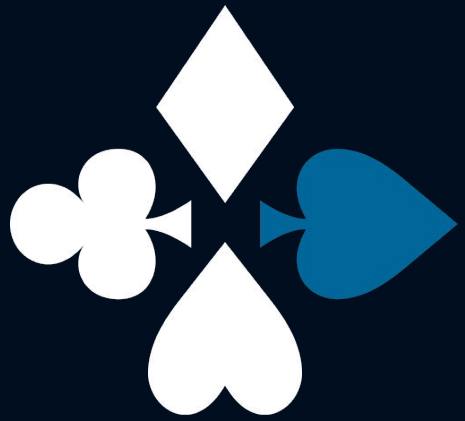


# Pokerbots 2025

Lecture 11: Noam Brown

# Sponsors




CITADEL | CITADEL Securities










# Announcements

# Subject Evaluations Open!

**pkr.bot/eval**

Please leave an honest review of your experience in this class!

# Team Strategy Reports due Last Night

- Contact us if you had issues submitting or need an extension
- Assignment details and submission on Canvas: **pkr.bot/canvas**

# Today 1/29

Guest Lecture: Dr. Noam Brown

Poker Social After!

- During office hours block
- 32-044, 2-4PM
- Come play with Noam!

# Today 1/29 (cont.)

## Final Bot Submission due 11:59PM EST

- Upload and select bot as active on scrimmage server
- Both report and bot needed to pass this class!
- Bot will compete in last and final Pokerbots tournament
- Non-secret prize amounts listed on syllabus
- **No extensions**

| Final Tournament Prizes | |
|---|---|
| First place | $10,000 |
| Second place | $6,500 |
| Third place | $3,500 |
| Fourth place | $2,000 |
| Fifth place | $1,000 |
| First place in language (Python, Java, or C++) | $500 x 3 |
| Second place in language (Python, Java, or C++) | $250 x 3 |
| Third place in language (Python, Java, or C++) | $125 x 3 |
| Best freshman-majority (>51%) team | $2,000 |

# Friday 1/31

Pokerbots Final Event 4:30-7PM in Kresge Auditorium

- Presentation of Awards
- Closing Ceremony
- Sponsor Event and Puzzle Hunt
- Lots of free merch and raffle prizes!
- Dinner provided 😋

RSVP at **pkr.bot/rsvp**

Sat 2/1

**Jump Trading** Poker Tournament!

5-8PM in BC Porter Room (tentative)


**$3500**

cash prize pool

RSVP at **pkr.bot/tournament**

# Roadmap

| | |
|---|---|
| 1/28 Yesterday | Final Report Due |
| 1/29 Today | Noam Brown Talk<br>Social Event<br>Final Bot Due |
| 1/31 Friday | Final Event |
| 2/1 Saturday | Jump Tournament |

# Giveaways!

# RSVP for final event: **pkr.bot/rsvp**

- RSVP for our final event!
- Also complete our course evaluation at **pkr.bot/eval**
- Must have proof of completion to claim prize
- Two winners selected at random
- Prize: Sony XM4 and GTO Wizard Subscription

# Guest: Noam Brown

# Building a Superhuman AI for No-Limit Poker

**Noam Brown**

OpenAI Reasoning

"And that's why there's never going to be a computer that will play World Class Poker. It's a people game."
-Doyle Brunson, *Super/System* 1979


"The analysis of a more realistic poker game than our very simple model should be quite an interesting affair."
-John Forbes Nash, 1951

# Who is the better poker player?

Option 1: Someone who, over a large enough sample size, wins head-to-head vs. any other player

Option 2: Someone who makes more money playing poker than anyone else

# Who is the better poker player?

**Minimax Equilibrium**

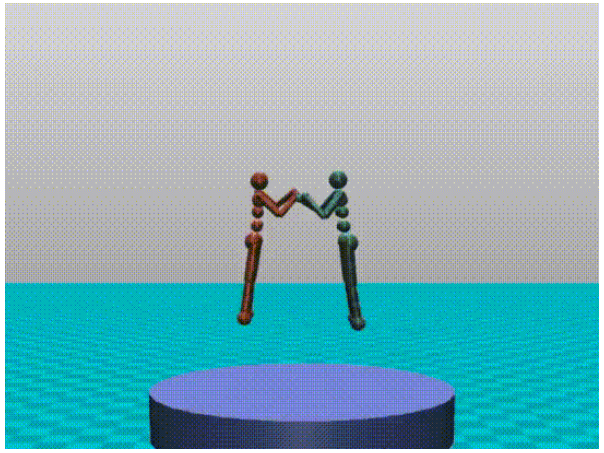
Option 1: Someone who, over a large enough sample size, wins head-to-head vs. any other player

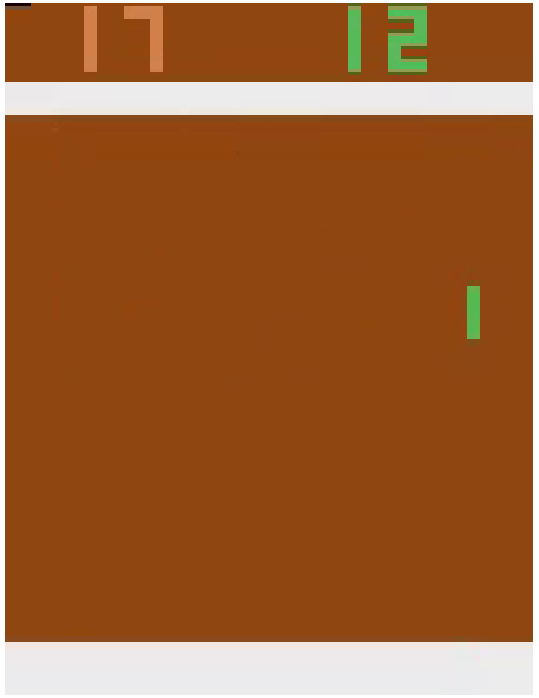
**Population Best Response**

Option 2: Someone who makes more money playing poker than anyone else

# Self-play in two-player zero-sum games

- In **self-play**, an agent gradually improves by playing against copies of itself
- Initial strategy can be completely random
- In balanced **two-player zero-sum** games, **sound self-play** provably converges to a **minimax equilibrium**
- Thus, given sufficient memory and compute, **any finite two-player zero-sum game can be “solved” via self-play**





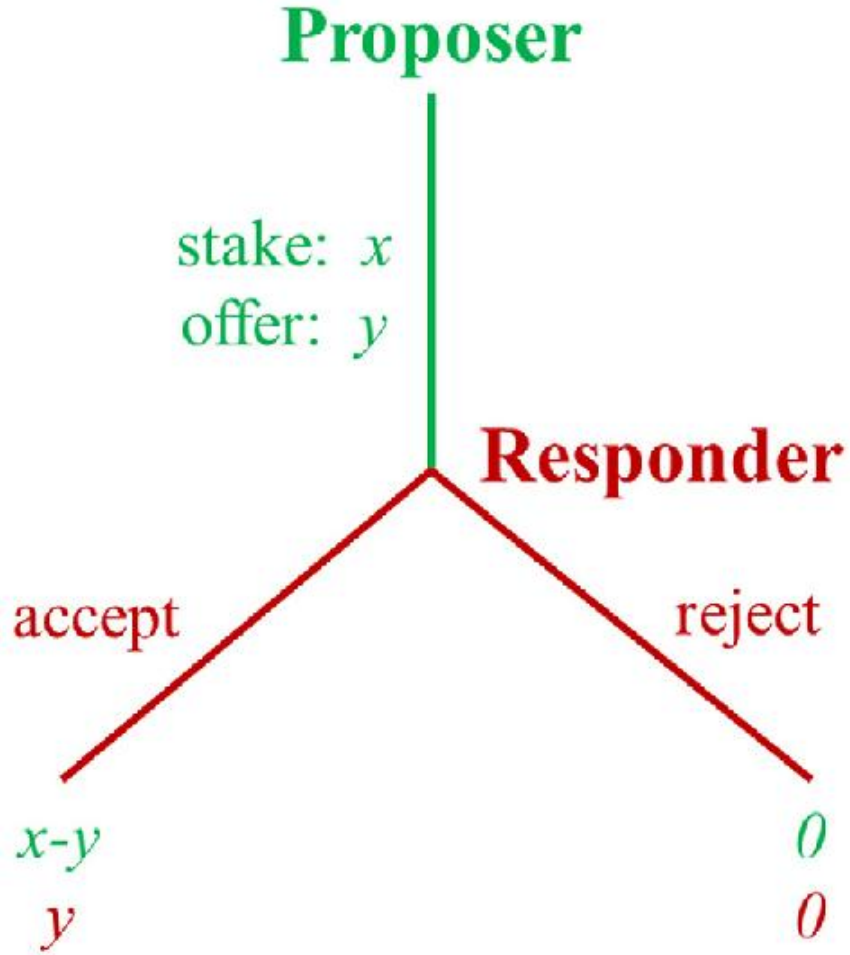
# Self-play in two-player zero-sum games

- In **self-play**, an agent gradually improves by playing against copies of itself
- Initial strategy can be completely random
- In balanced **two-player zero-sum** games, **sound self-play** provably converges to a **minimax equilibrium**
- Thus, given sufficient memory and compute, **any finite two-player zero-sum game can be “solved” via self-play**

**Question:** What about non-two-player zero-sum games?

# Ultimatum Game

- Alice is given $100
- First, Alice offers $0 - $100 to Bob
- Then, Bob must decide whether to **accept** or **reject**
  - If Bob **accepts**, then Alice and Bob keep their money
  - If Bob **rejects**, then Alice and Bob get nothing

# Who is the better poker player?

**~~Minimax Equilibrium~~**

~~Option 1: Someone who, over a large enough sample size, wins head-to-head vs. any other player~~

Not meaningful in general games!

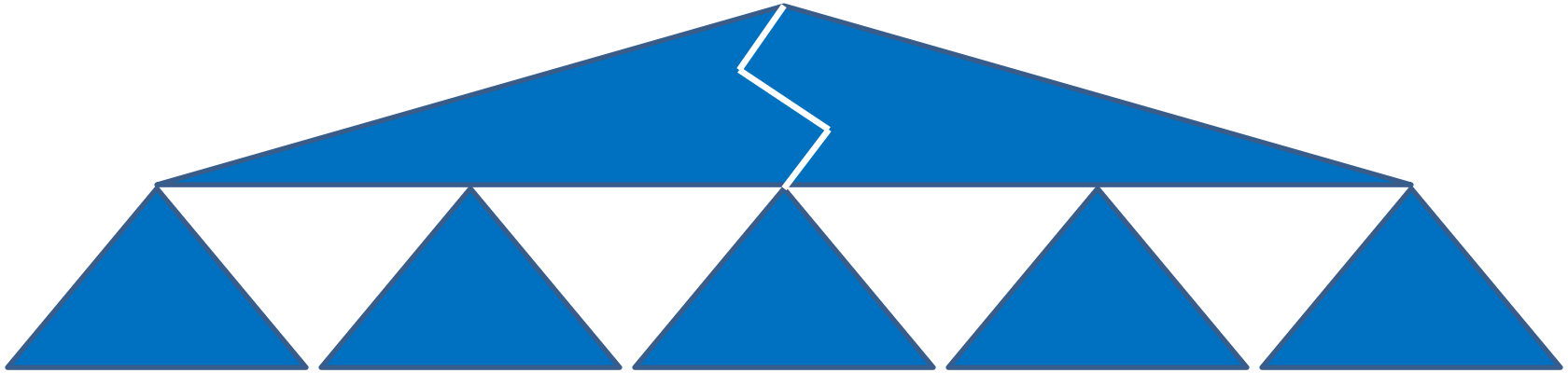
**Population Best Response**

Option 2: Someone who wins more often against the population of players than anyone else

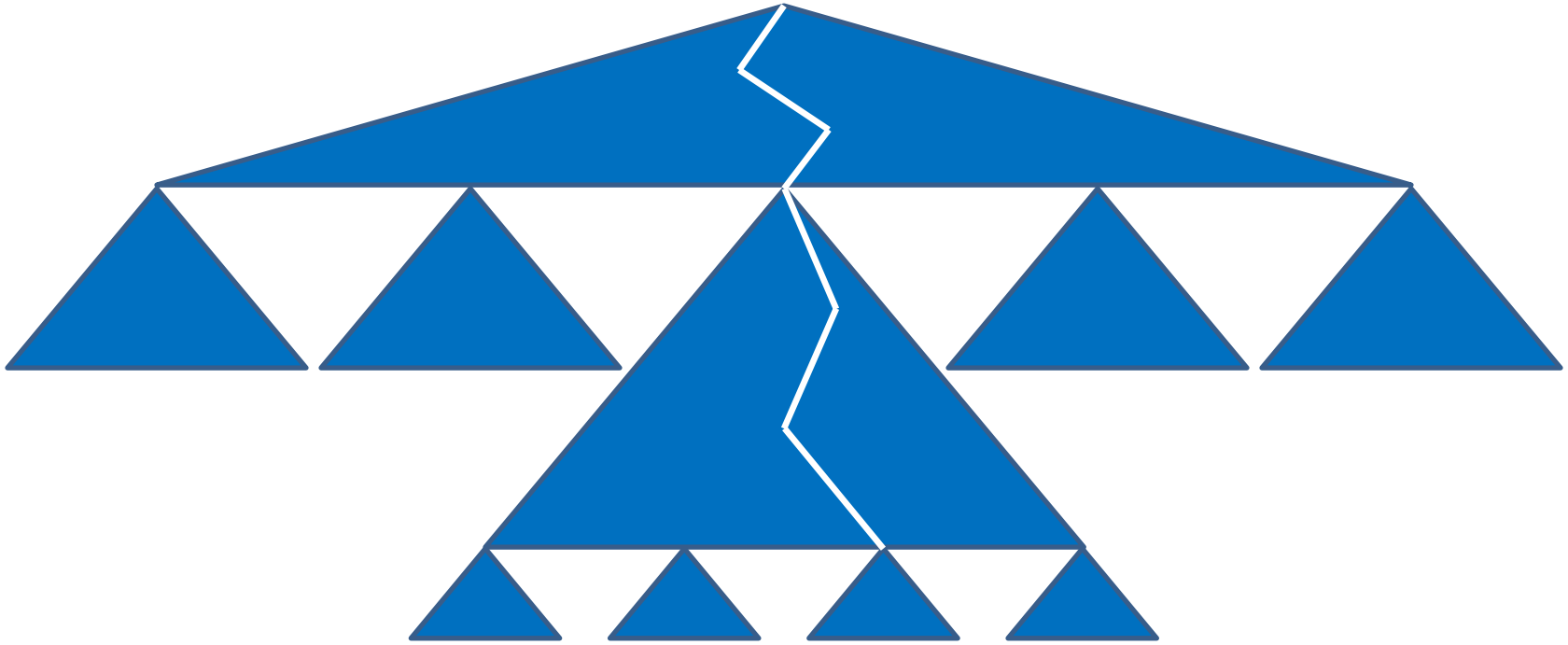
Requires data on the population of players, i.e., human data

# Libratus/Pluribus

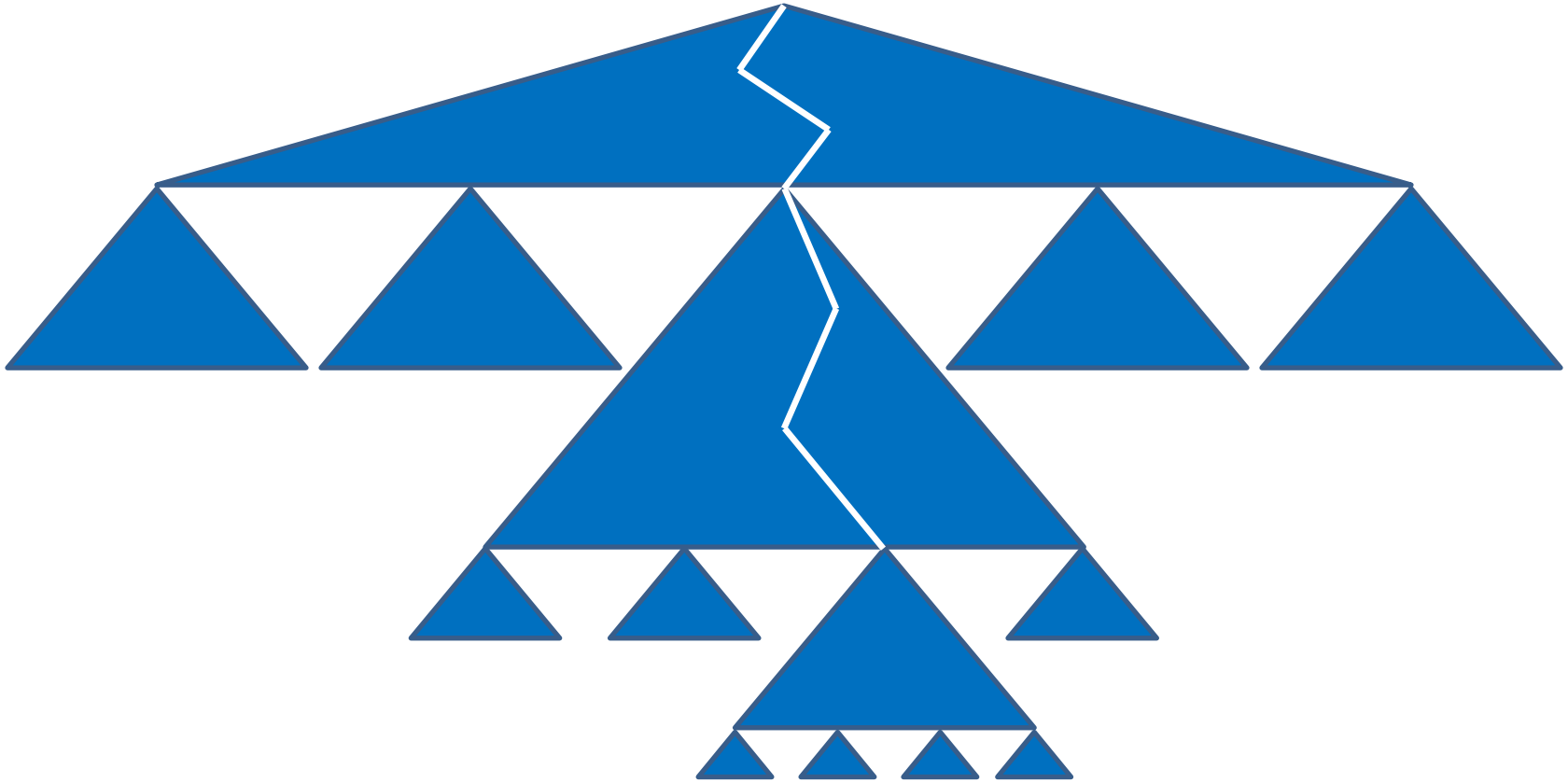
# Libratus/Pluribus



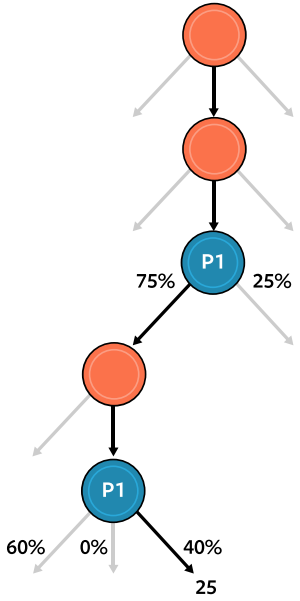
# Libratus/Pluribus

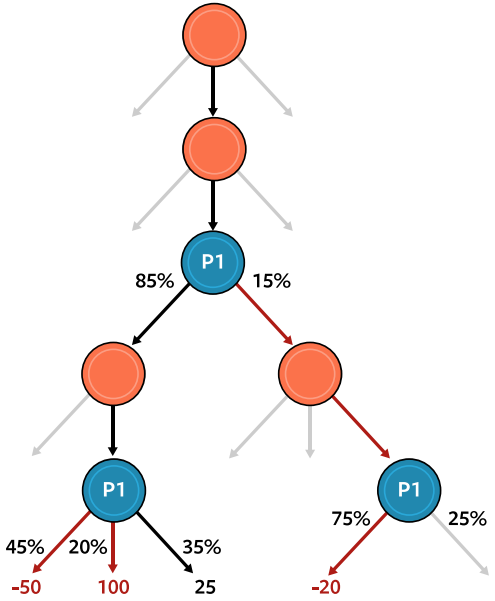


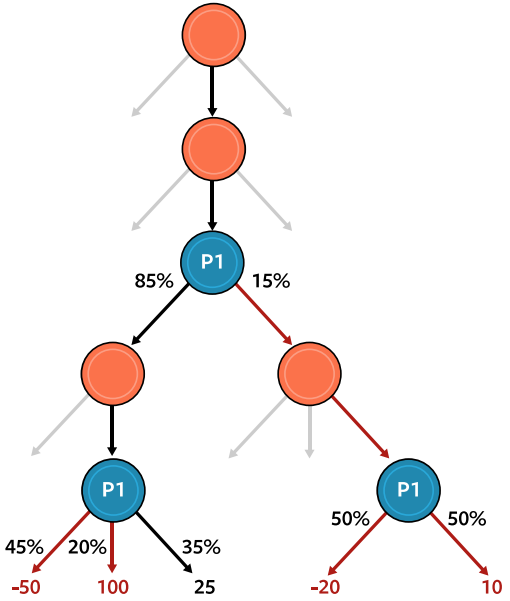
# Libratus/Pluribus



# Computing Equilibria with Counterfactual Regret Minimization



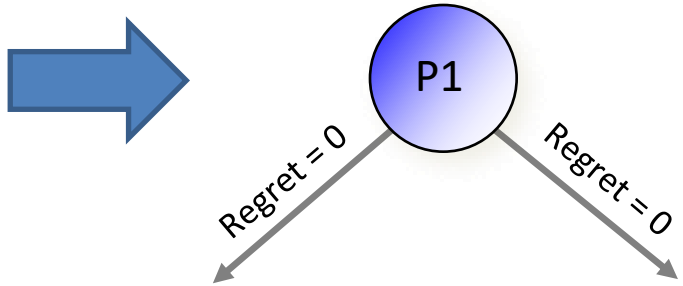



# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]

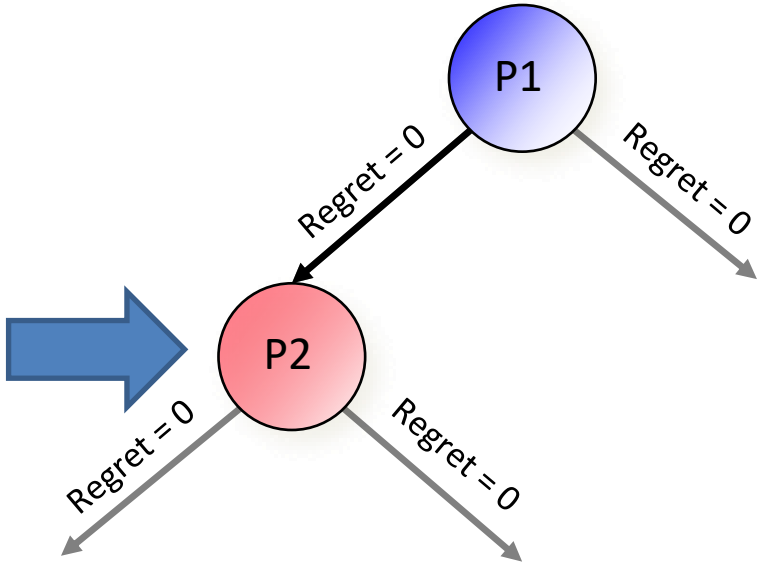
Pick action proportional to **positive regret**



# Monte Carlo Counterfactual Regret Minimization (MCCFR)

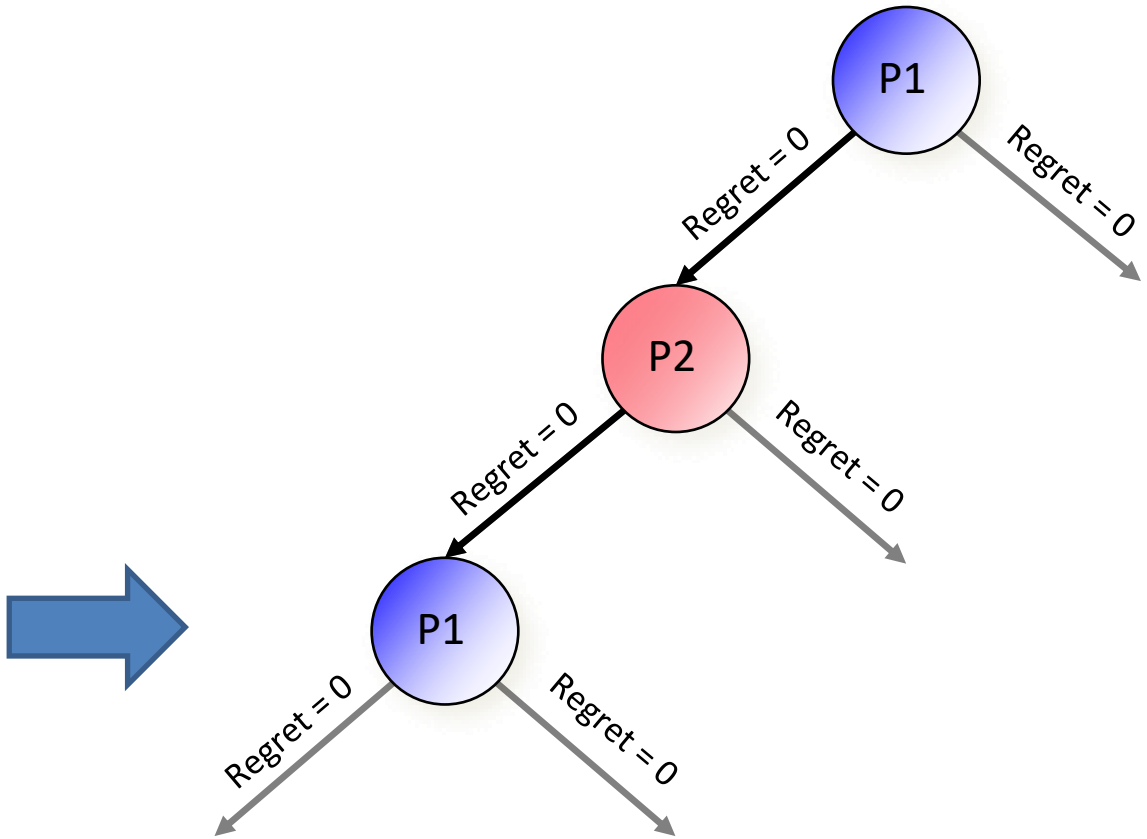
[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]

Pick action proportional to **positive regret**



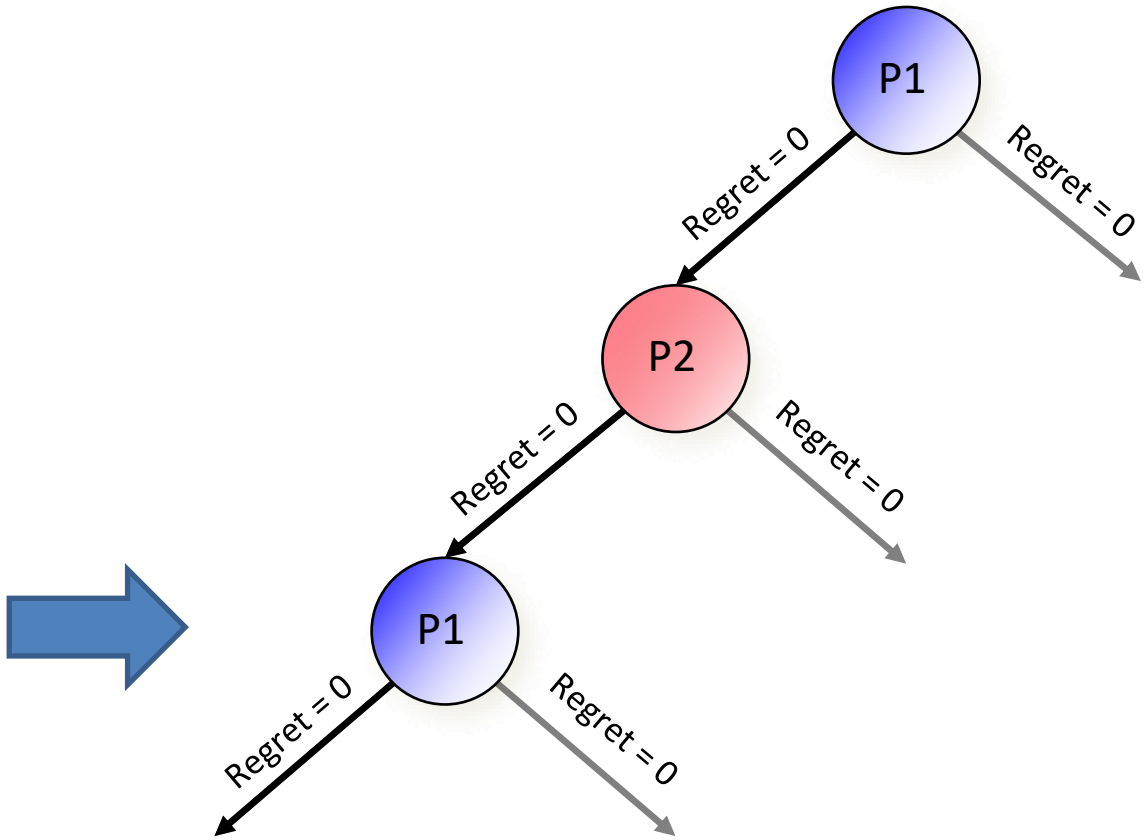
# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



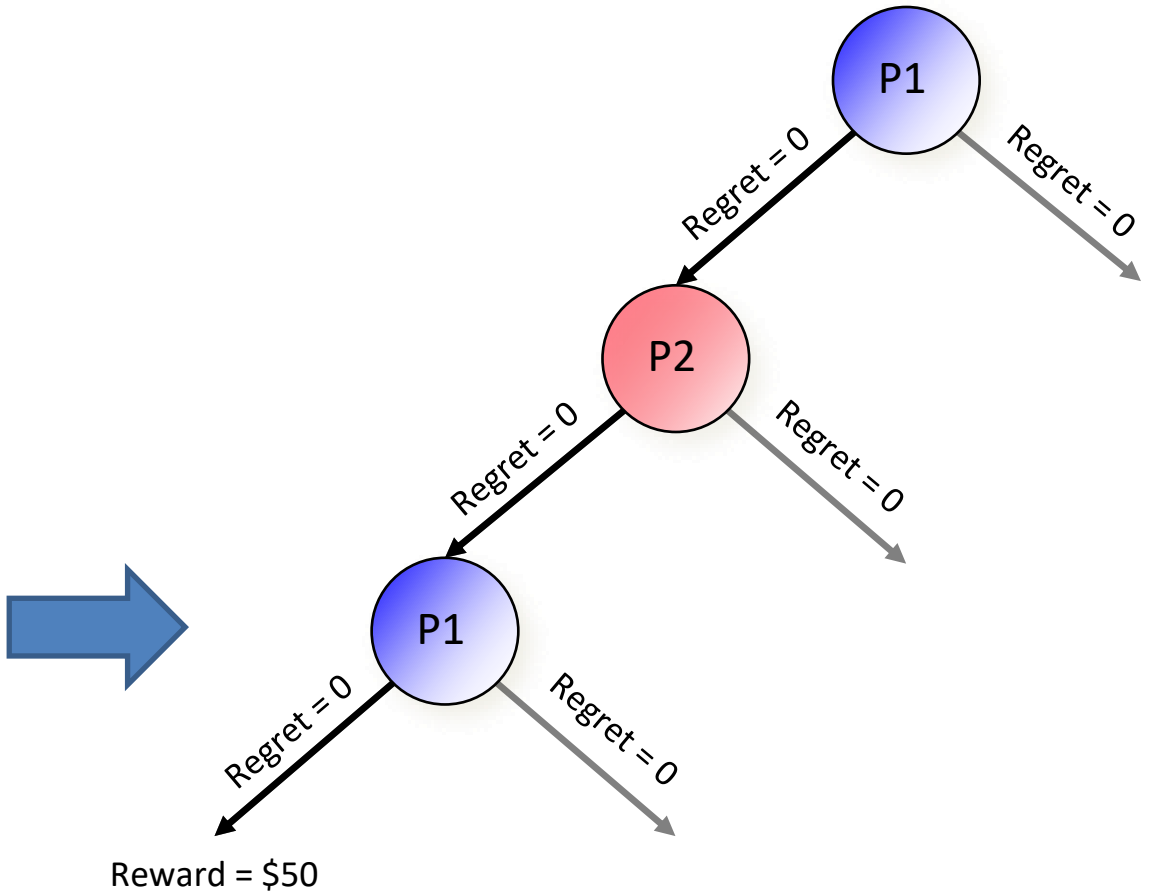
# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



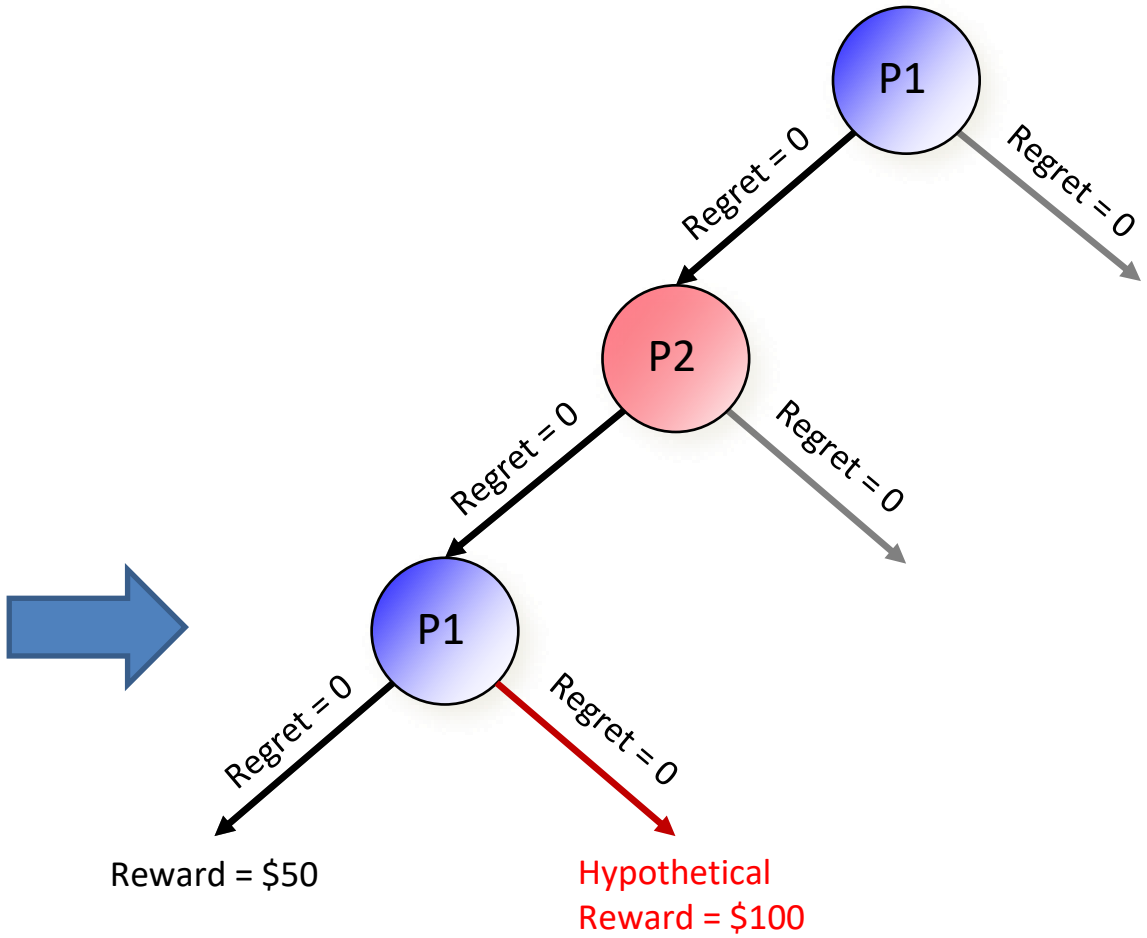
# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



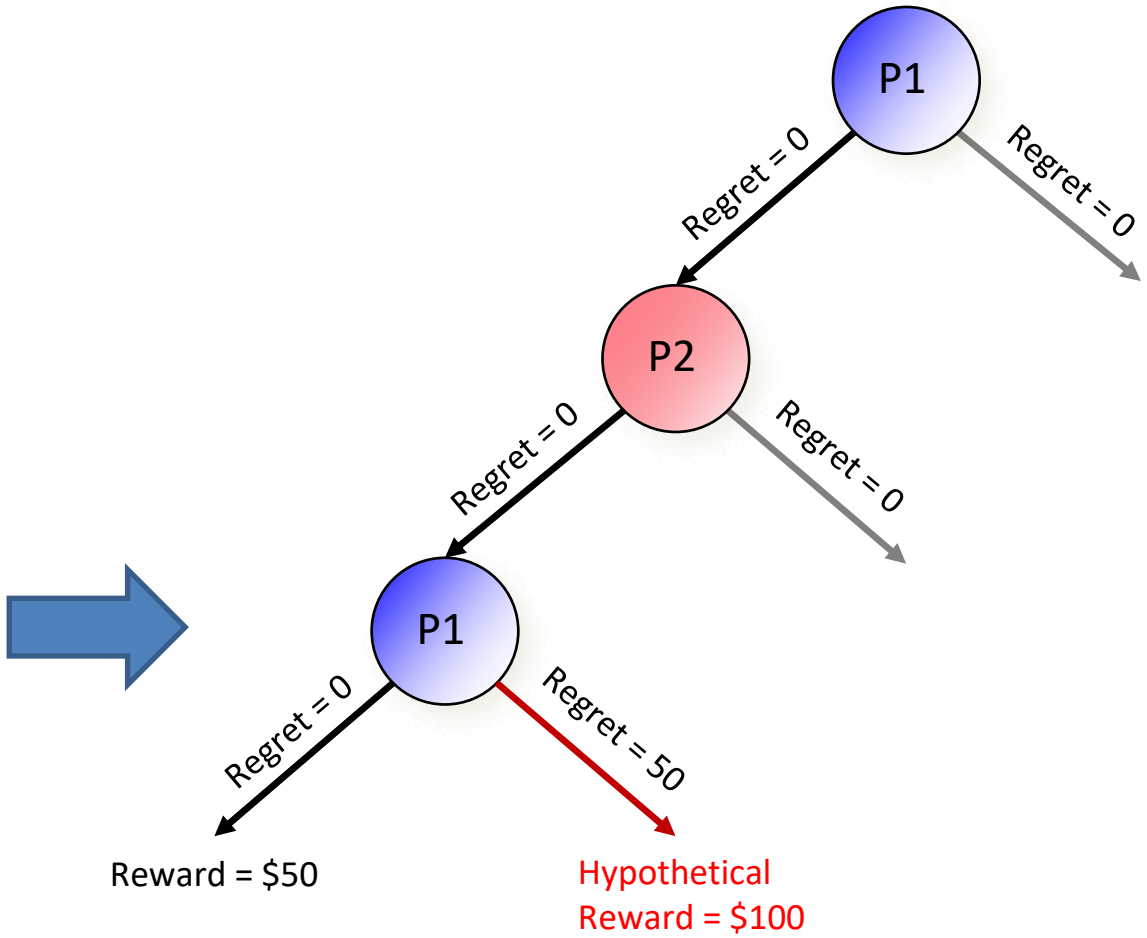
# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



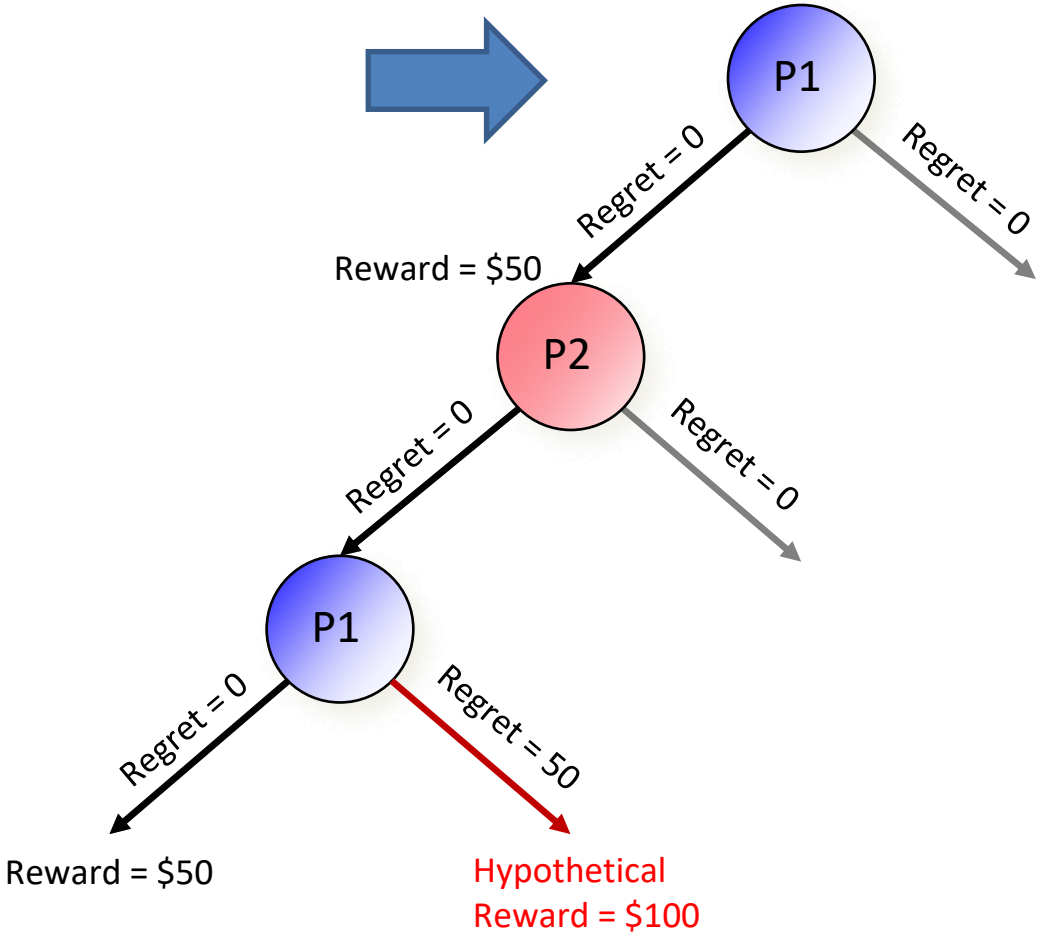
# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



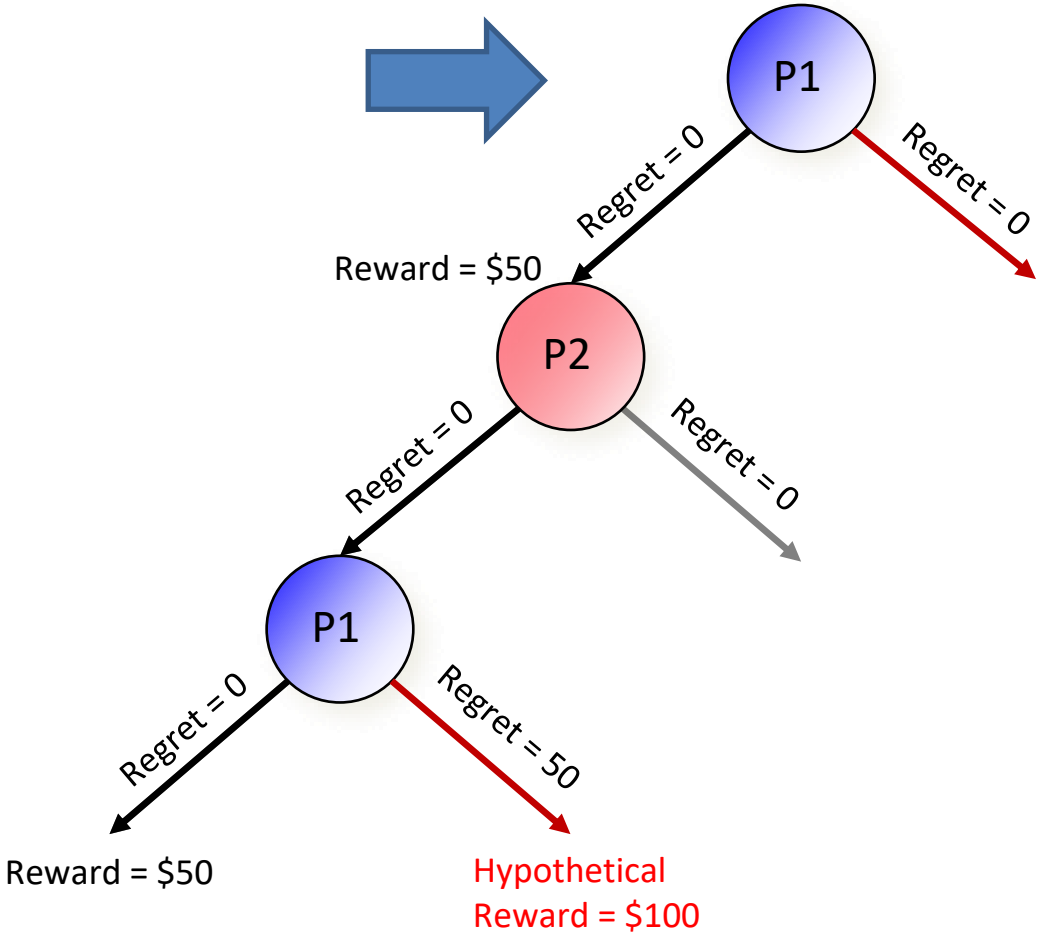
# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



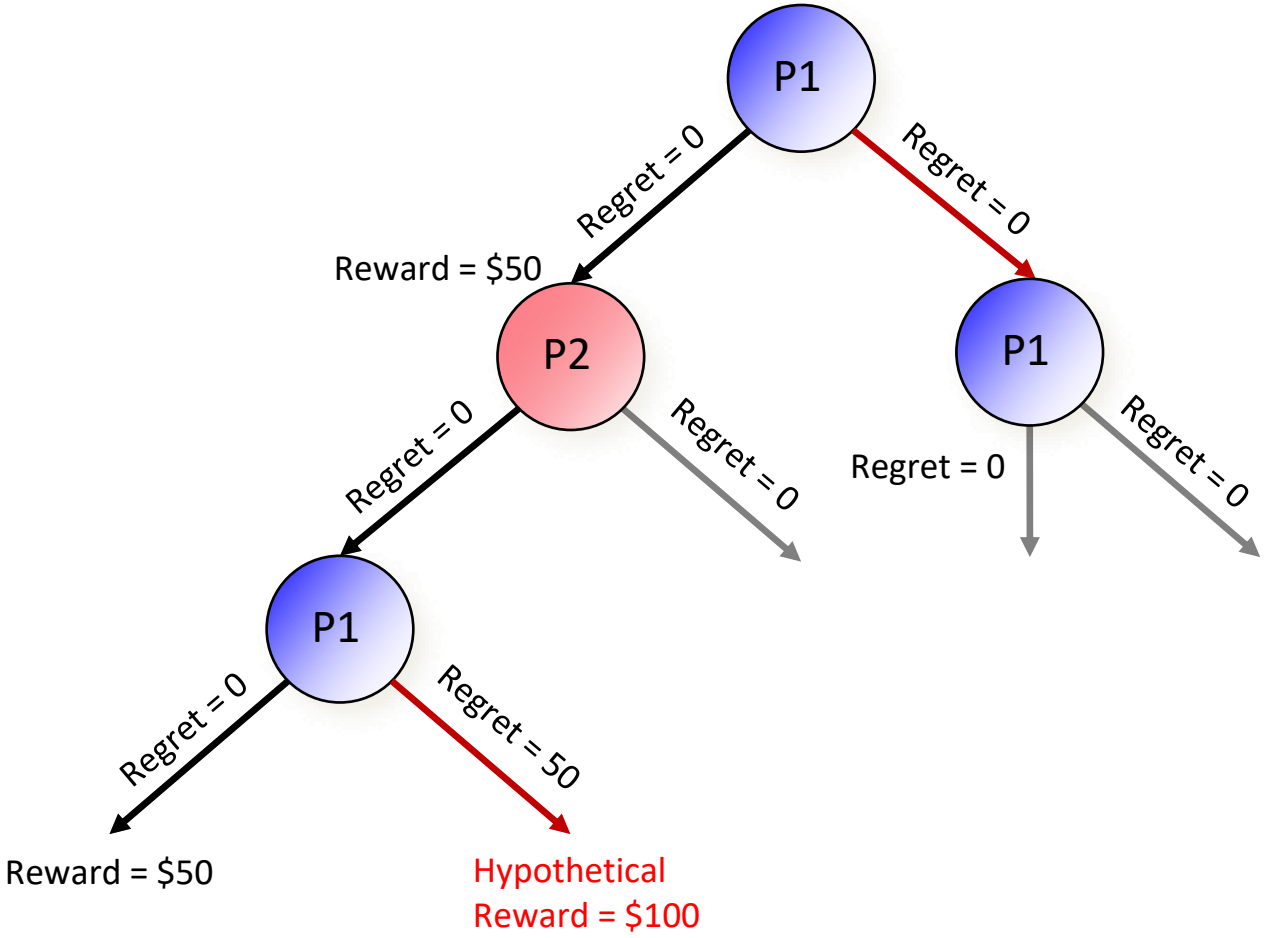
# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



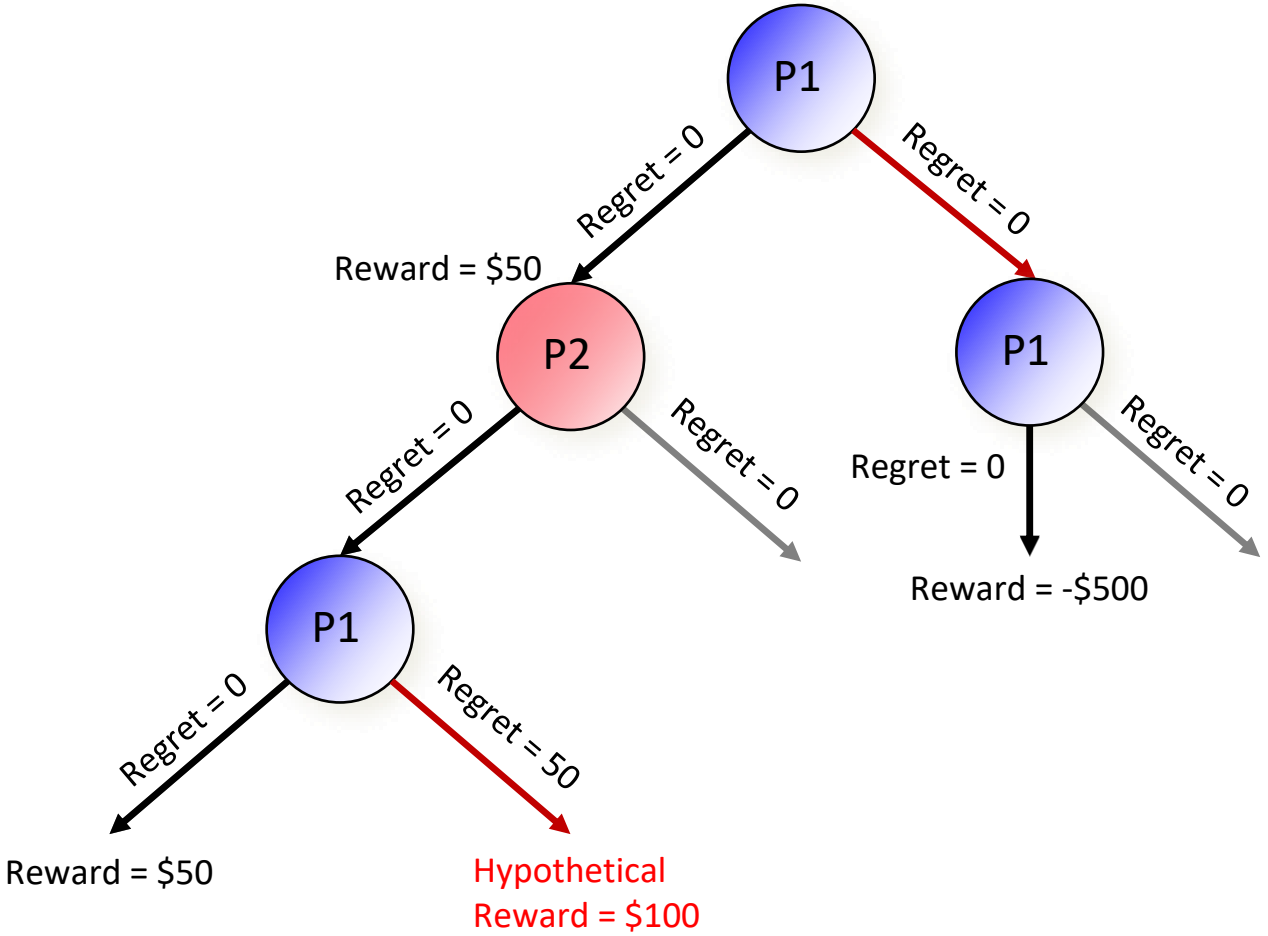
# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



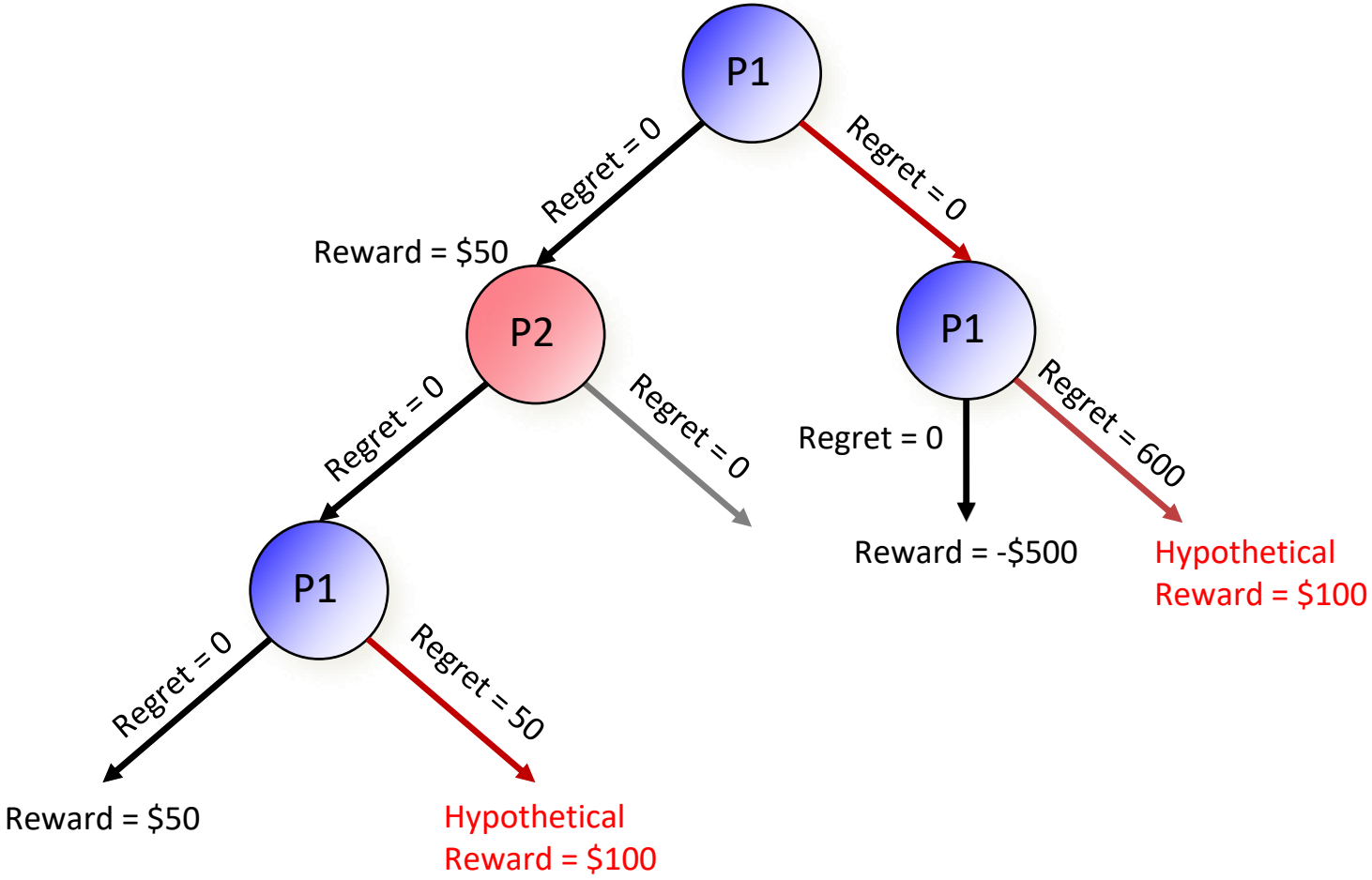
# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



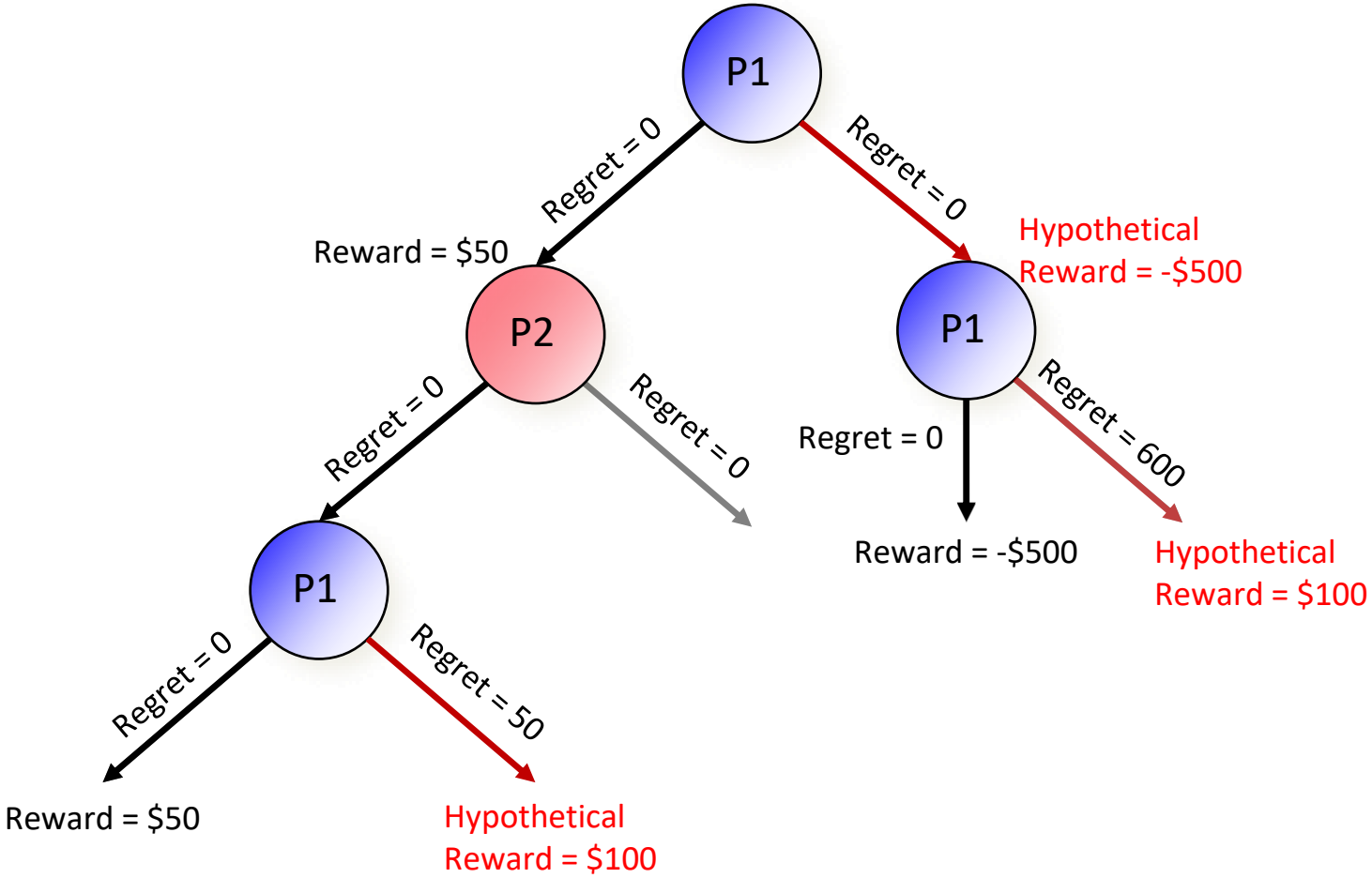
# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



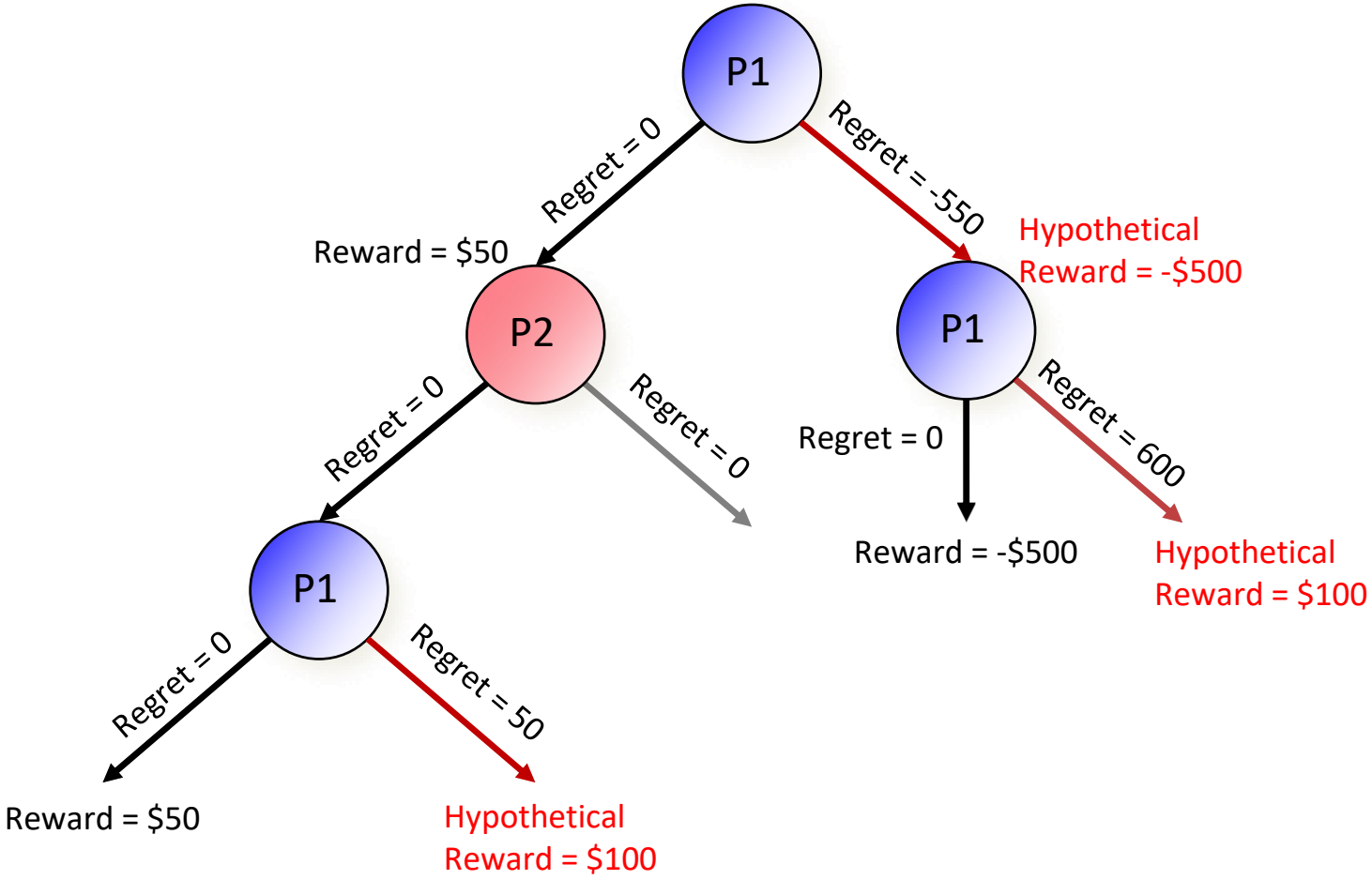
# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



# Monte Carlo Counterfactual Regret Minimization (MCCFR)

[Zinkevich *et al.* NeurIPS-07, Lanctot *et al.* NeurIPS-09]



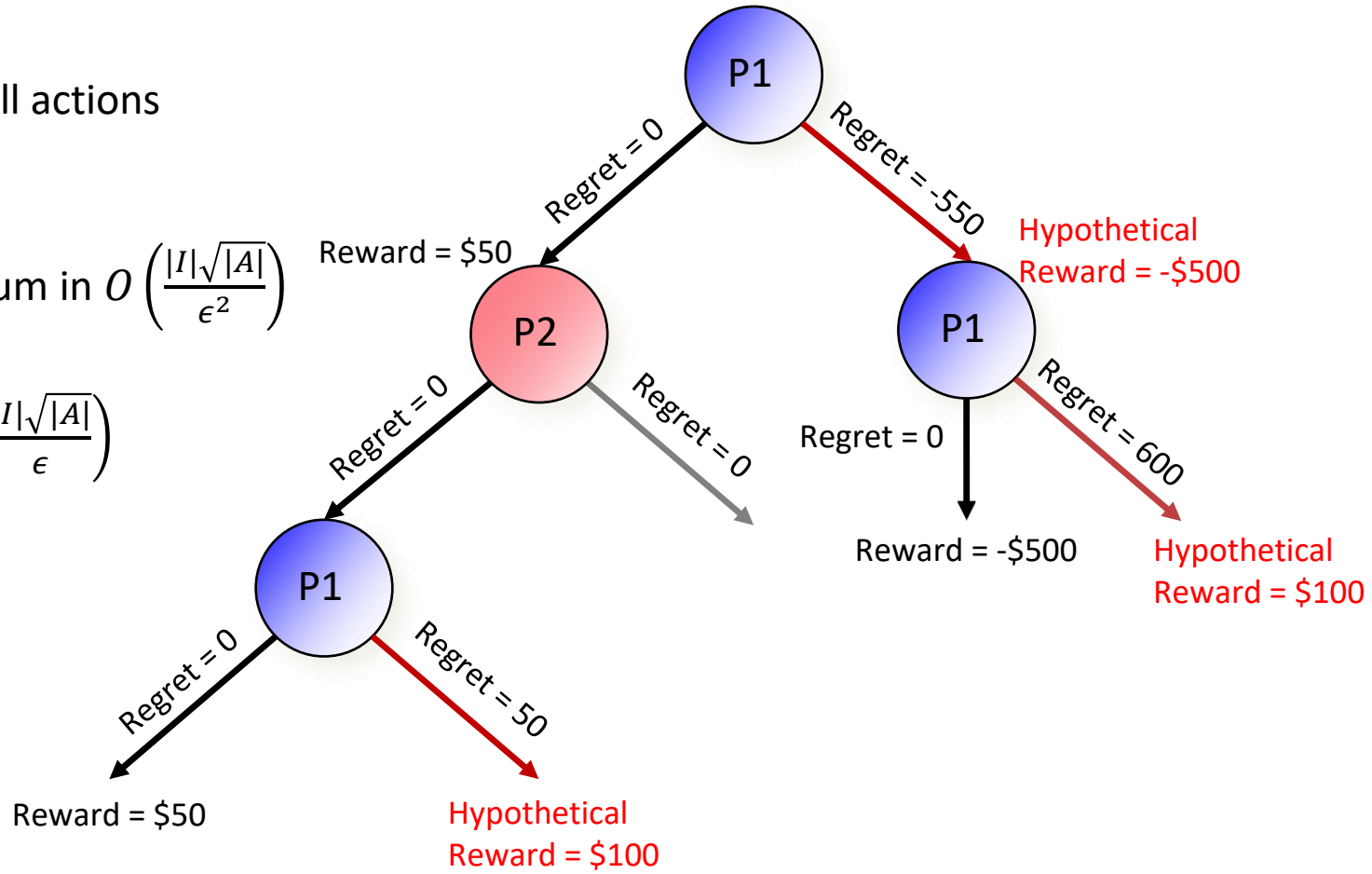
# Counterfactual Regret Minimization (CFR)

[Zinkevich *et al.* NeurIPS-07]

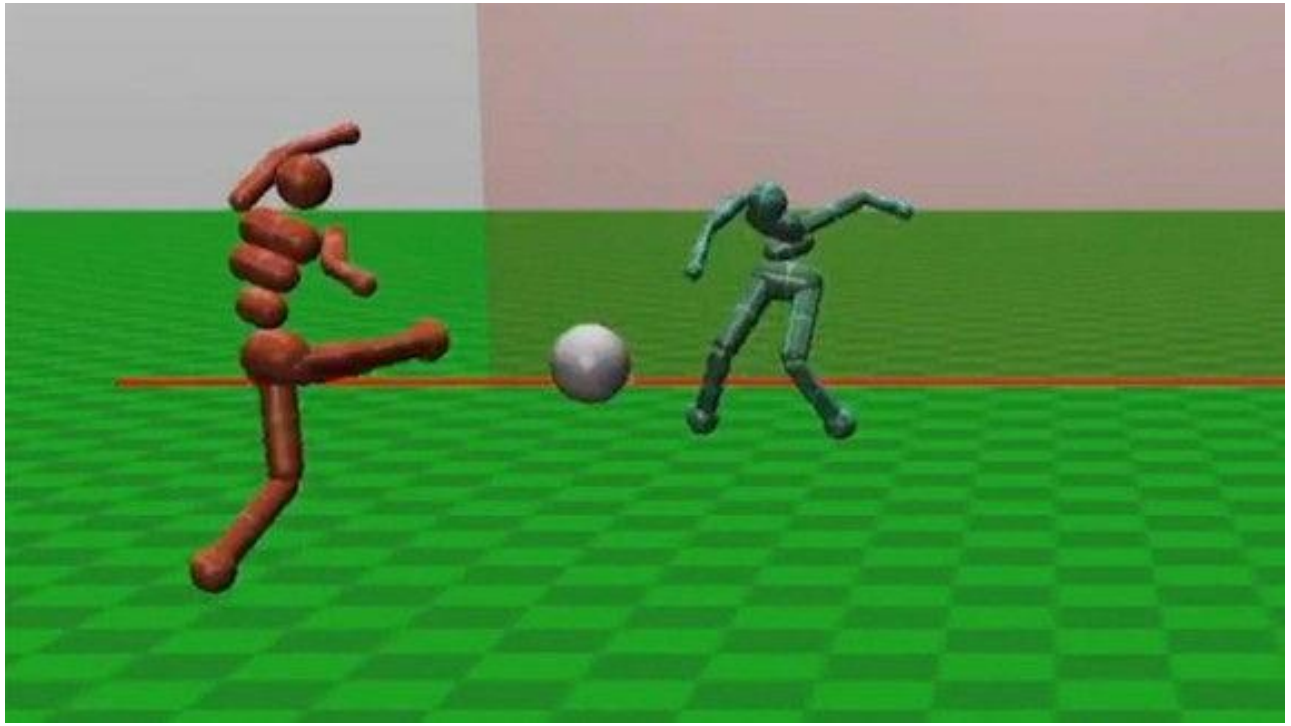
Similar, but takes the EV over all actions rather than sampling

**Average** converges to equilibrium in $O\left(\frac{|I|\sqrt{|A|}}{\epsilon^2}\right)$

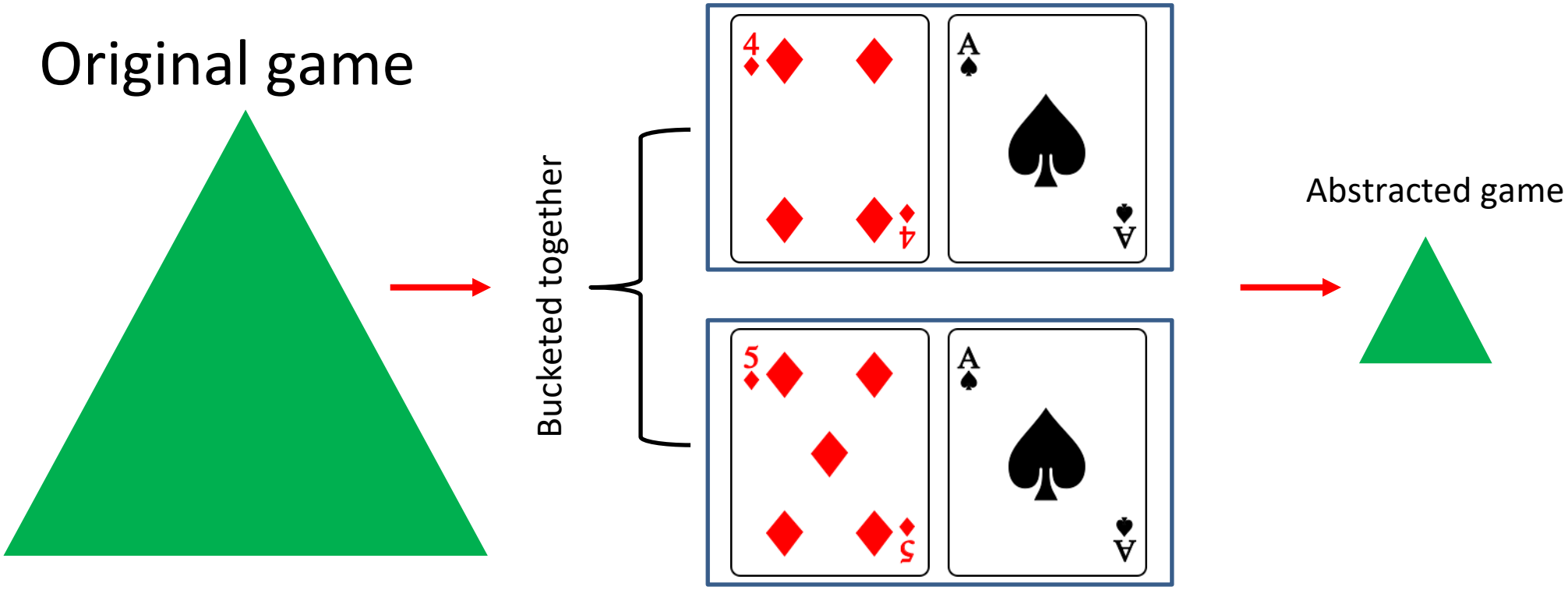
But in practice converge in $O\left(\frac{|I|\sqrt{|A|}}{\epsilon}\right)$



# Extending CFR to Large Games



# Prior Approach: Abstraction in Games

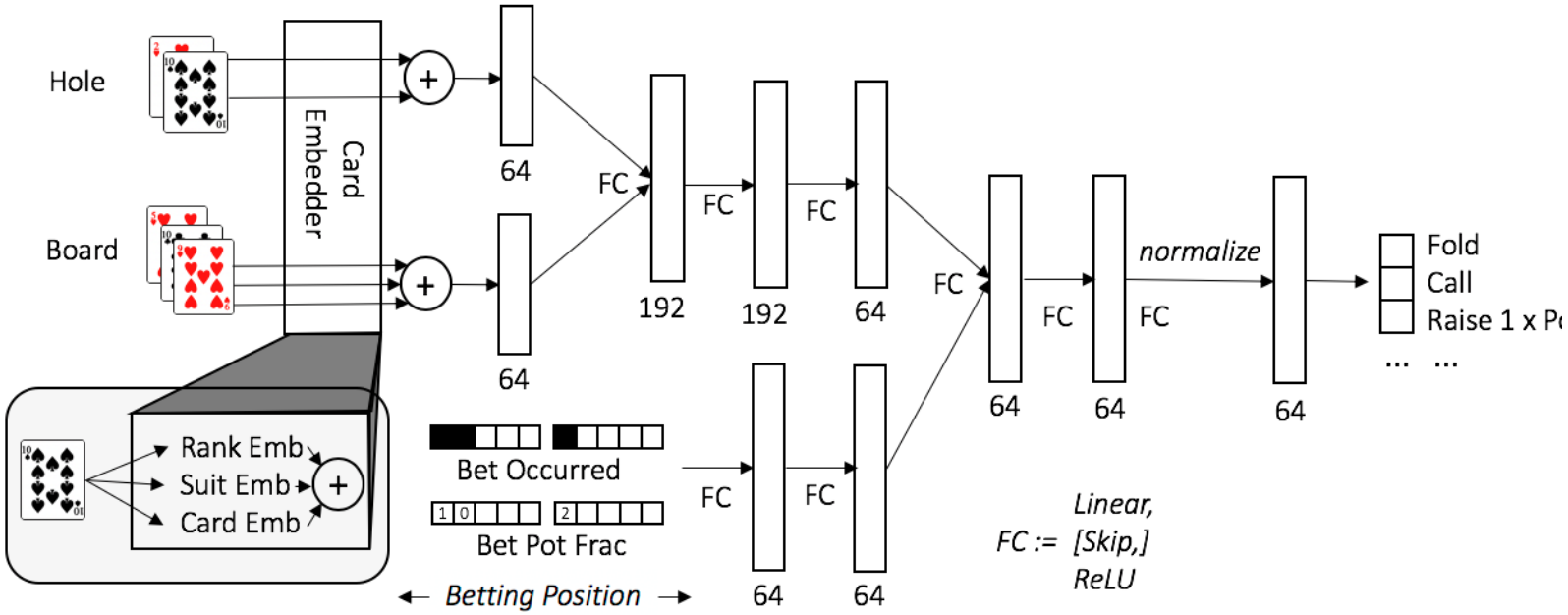



- Requires extensive domain knowledge
  - Several papers written on how to do abstraction just in poker [Johanson et al. AAMAS-13, Ganzfried & Sandholm AAAI-14]
  - Difficult to extend to other games

# Deep CFR / DREAM

[Brown et al. ICML-19; Steinberger, Lerer, Brown arXiv-20]

- Replaces abstraction with neural network approximation of regrets
- Deep CFR / DREAM require far less domain knowledge



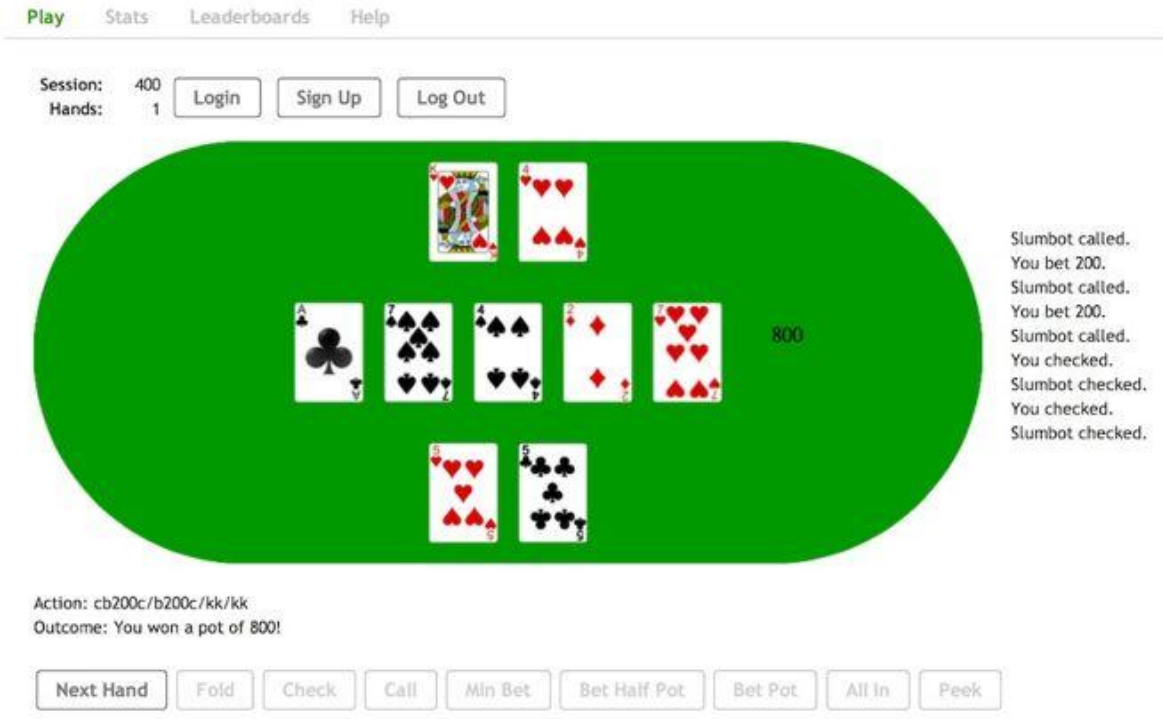
# Searching for a better strategy in real time

Image Credit: UC Berkeley CS-188 Lecture 6

# Annual Computer Poker Competition

- Each year, research labs would make poker bots and play them against each other.

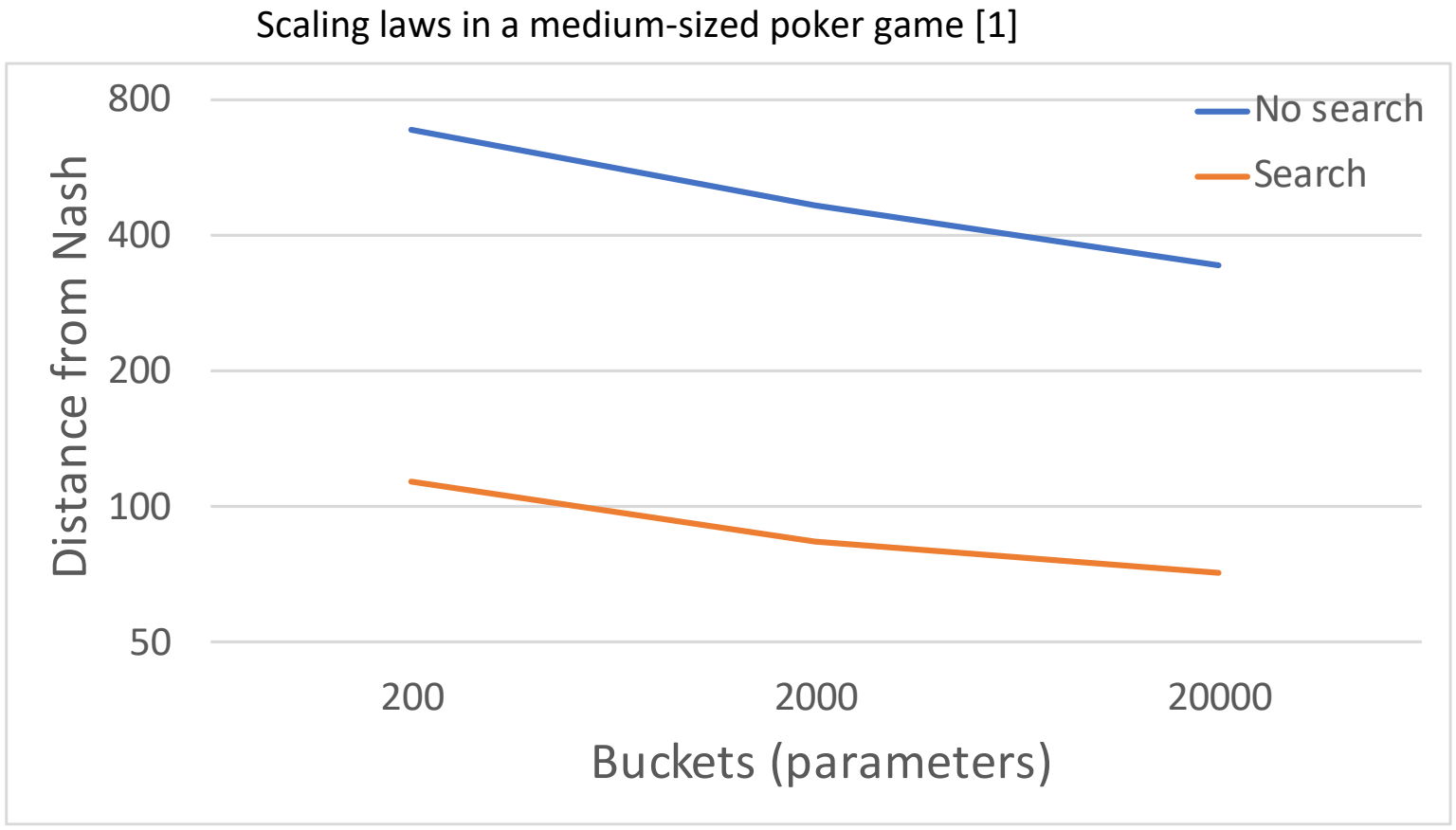
- It turned into a competition of scaling models:
  - **2012:** 5,000 buckets
  - **2013:** 30,000 buckets
  - **2014:** 90,000 buckets
  - **2015:** 600,000 buckets



# 2015 Brains vs. AI Poker Competition

- In 2015 we (CMU) challenged 4 top poker pros to an 80,000-hand poker competition
- $120,000 in prize money to incentivize them
- Our bot (Claudico) lost by 9.1 bb/100

# The importance of search in poker




[1] “Safe and Nested Subgame Solving in Imperfect-Information Games.” Brown & Sandholm. NeurIPS 2017 Best Paper.

# 2017 Brains vs AI Two-Player Poker AI

[Brown & Sandholm Science-17]

- Libratus against 4 top poker pros


- 120,000 hands of poker
- $200,000 in prize money
- **Won by 15 bb/100** (Claudico had lost by 9 bb/100)
  - P-value ≈ 0.0002
- Each human lost individually to Libratus

# 2019 Pluribus Six-Player Poker AI

[Brown & Sandholm Science-19]

- Pluribus against 15 top pros in ***six-player*** no-limit Texas Hold'em

- 10,000 hands over 12 days in June 2019
  - Used variance-reduction techniques to decrease luck
  - One bot playing with five humans

- Won with >95% statistical significance

- **Cost under $150 to train**, runs on 28 CPU cores (no GPUs)

# Why wasn’t search considered important in poker before?

- Cultural factors: researchers wanted the solution for the entire game upfront
- Scaling test-time compute makes experiments more expensive
- Incentives:
  - People were always thinking about winning the next Annual Computer Poker Competition (ACPC)
  - The ACPC limited test-time compute to 2 CPU cores
- **Most important:** people underestimated the difference it would make

# Why is search in imperfect-information games hard?

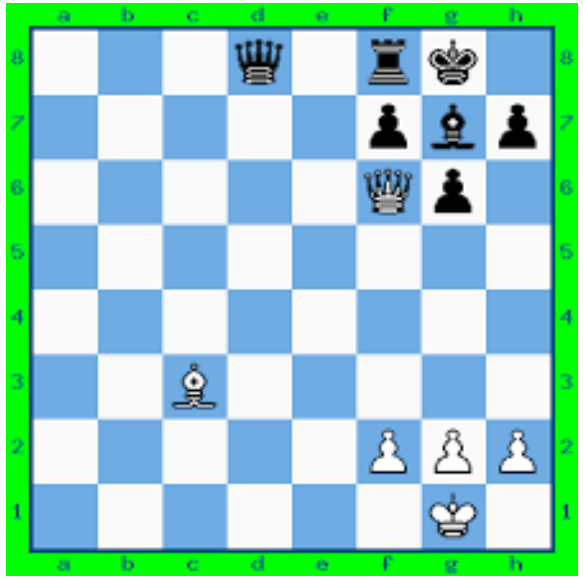
**Because “states” as traditionally defined don’t have well-defined values**

# Search in Perfect-Information Games

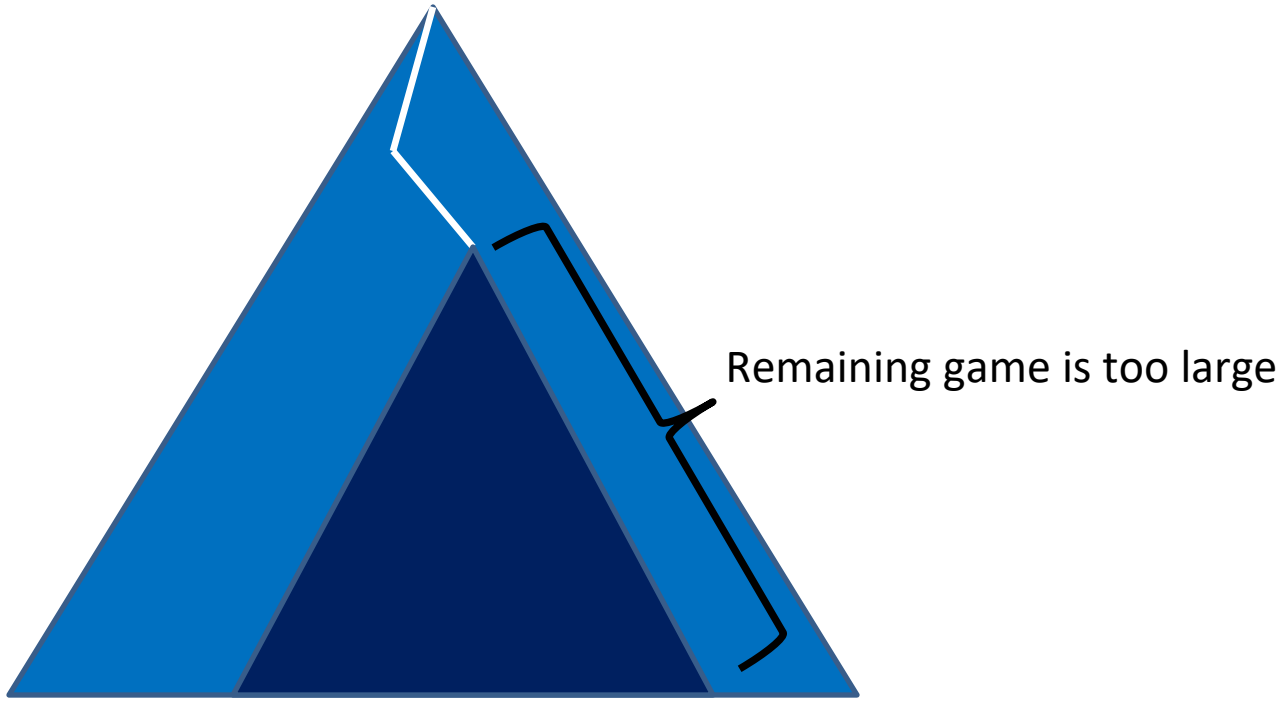
- In perfect-information games, the **value of a state** is the **unique** value of both players playing optimally from that point forward

- A **value network** takes a state as input and outputs an estimate of the state value

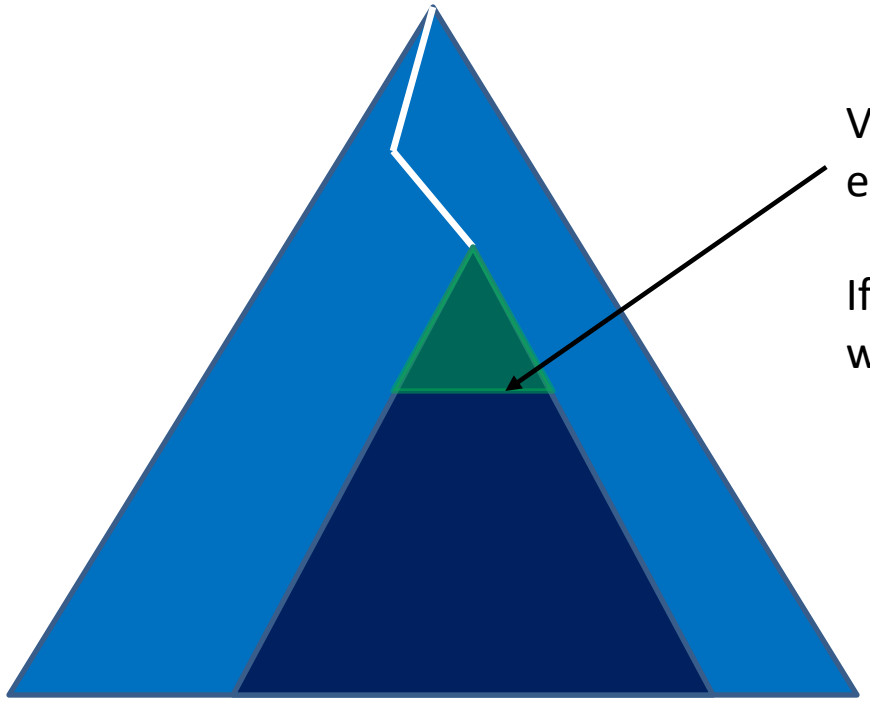
$$f_{white}(\quad) = 1$$



# Search in Perfect-Information Games



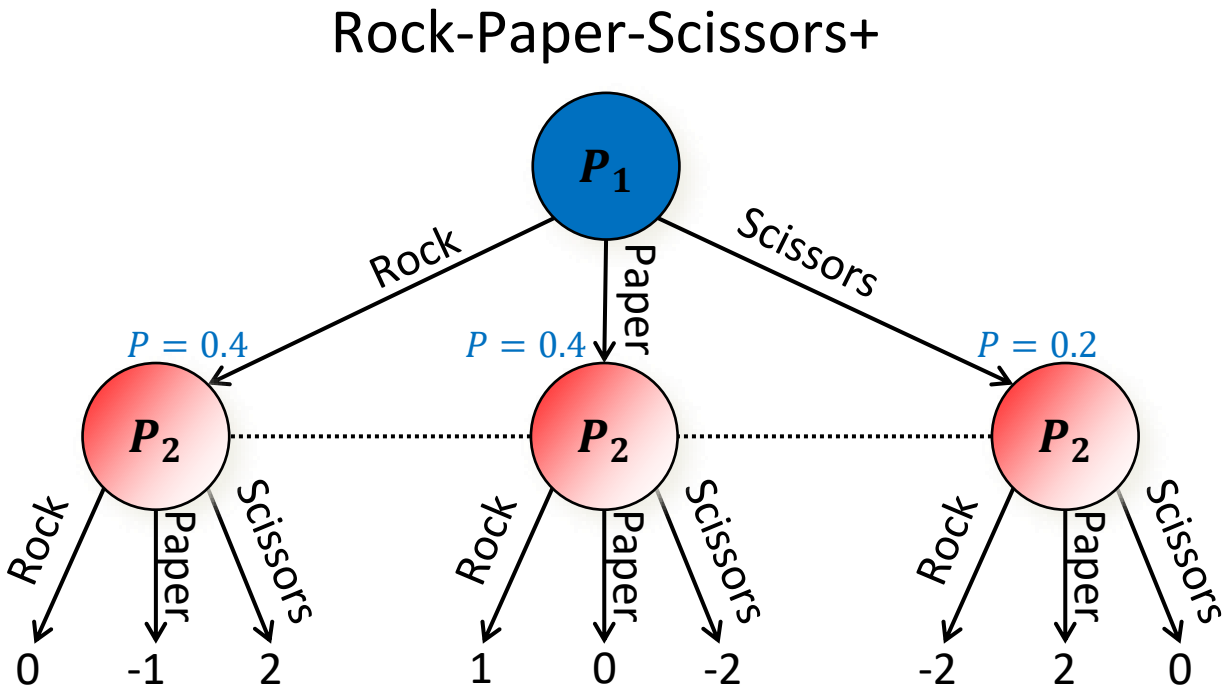
# Search in Perfect-Information Games



Value substituted at leaf node $s$ is estimate of equilibrium value $\hat{v}(s)$

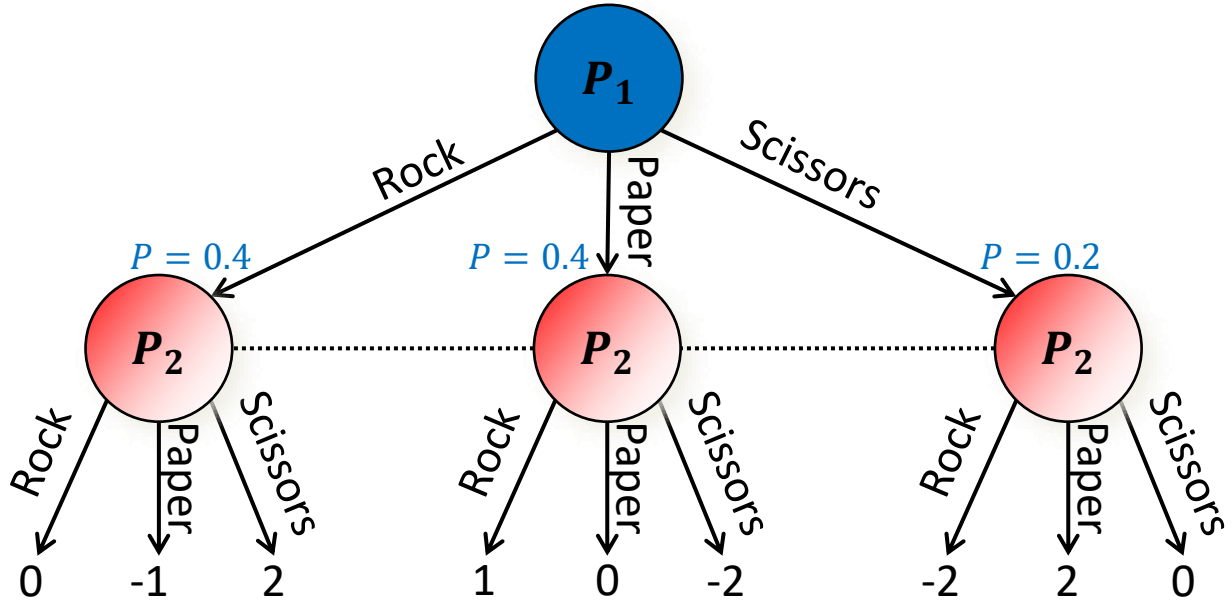
If estimate is perfect, depth-limited search will find optimal strategy

# Depth-Limited Search

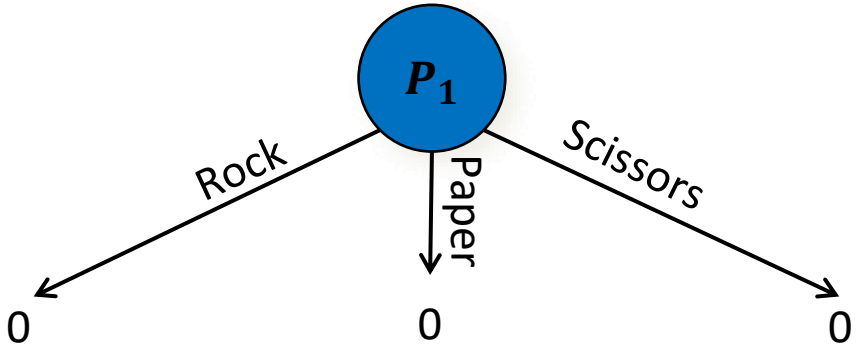


# Depth-Limited Search

Rock-Paper-Scissors+

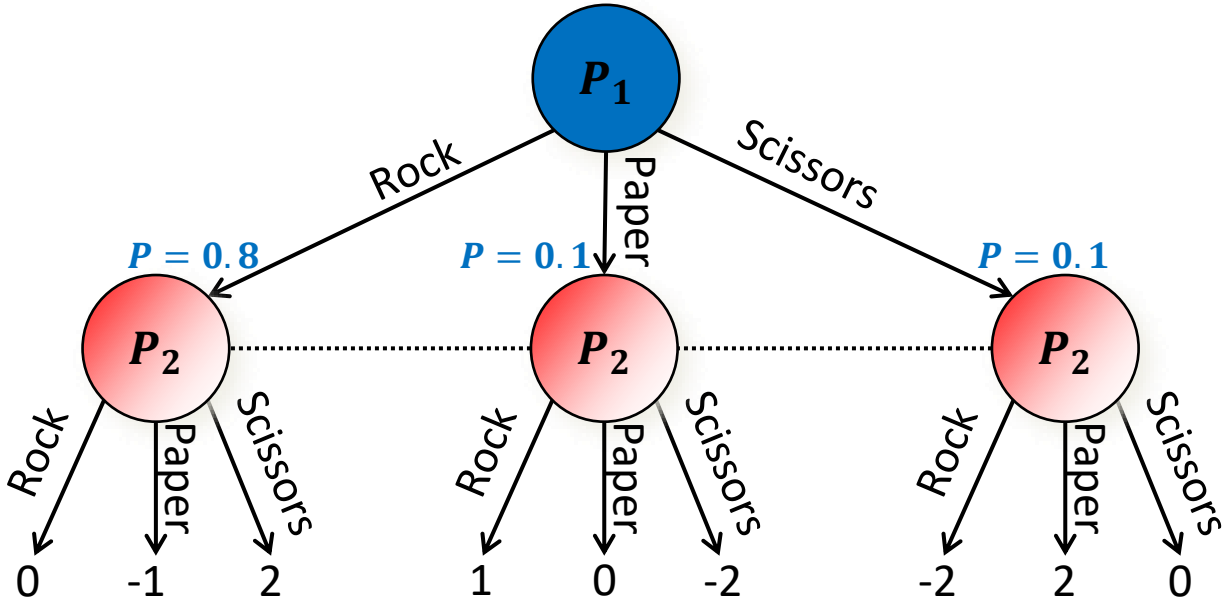



Depth-Limited Rock-Paper-Scissors+

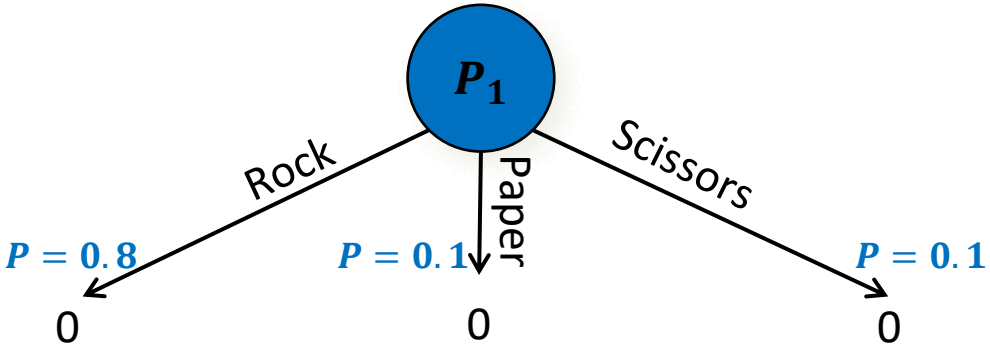


# Depth-Limited Search

Rock-Paper-Scissors+

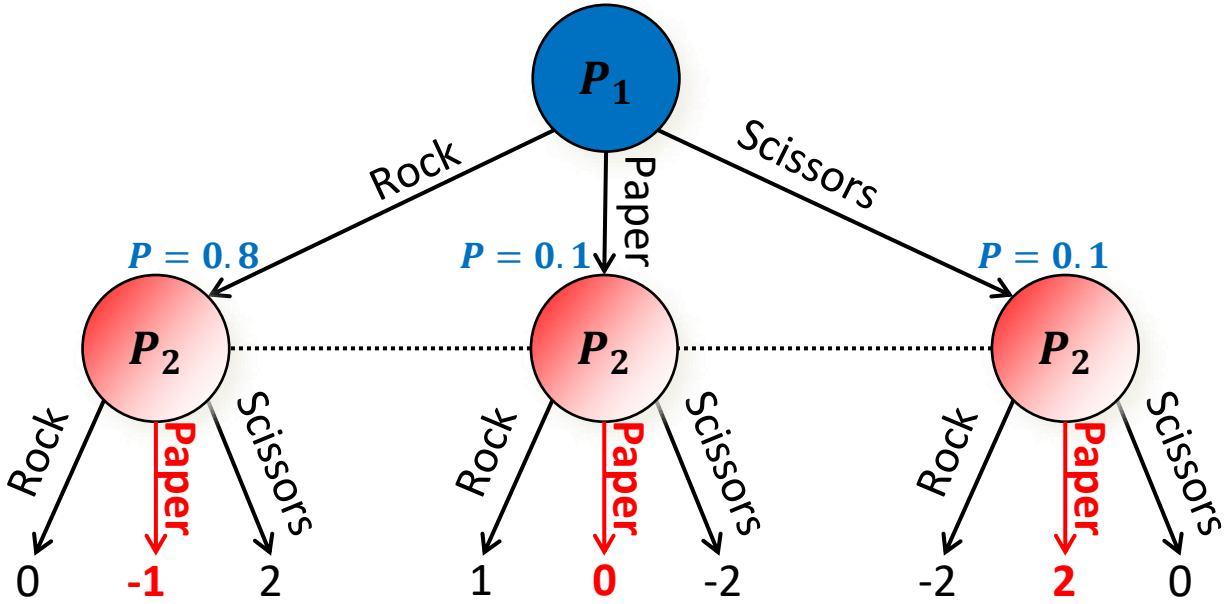



Depth-Limited Rock-Paper-Scissors+

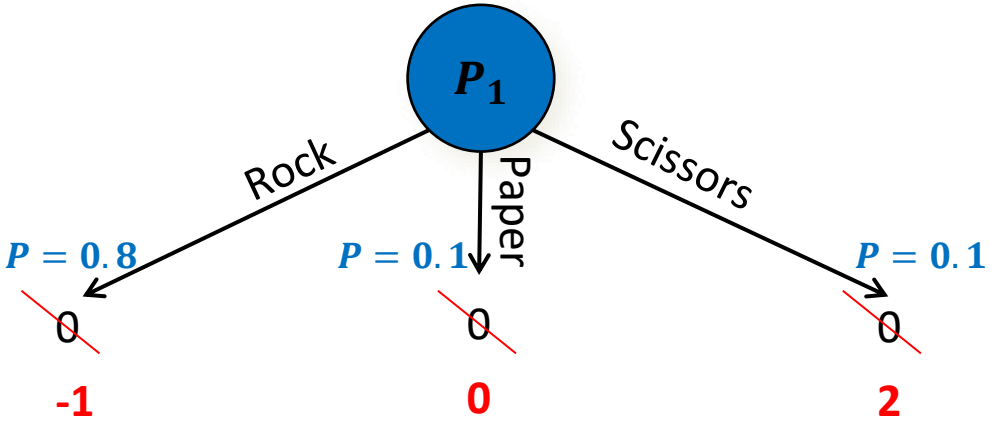


# Depth-Limited Search

Rock-Paper-Scissors+

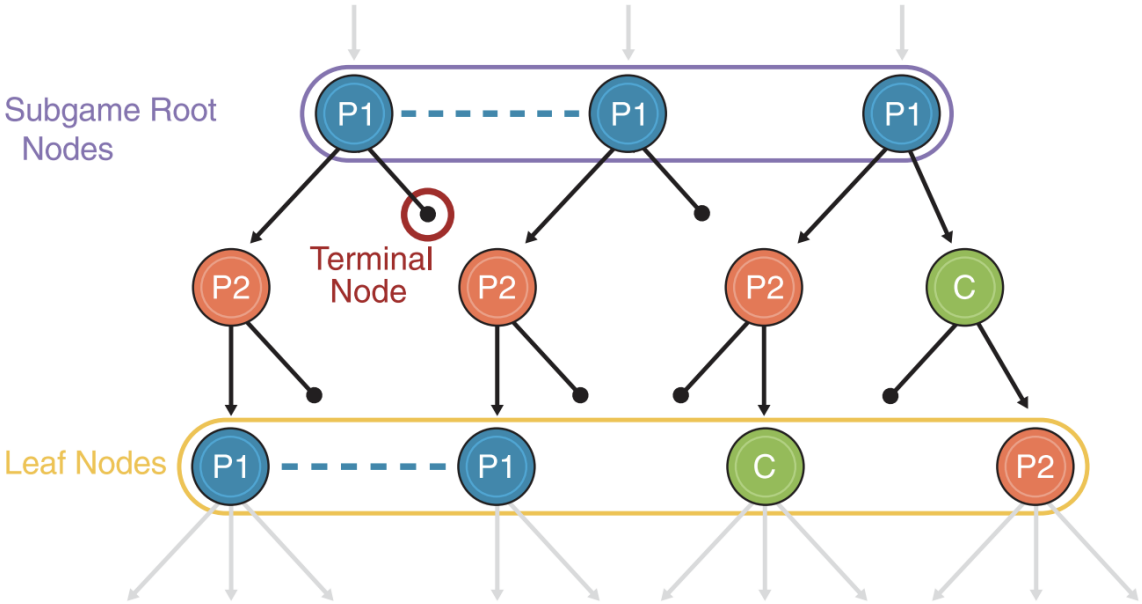


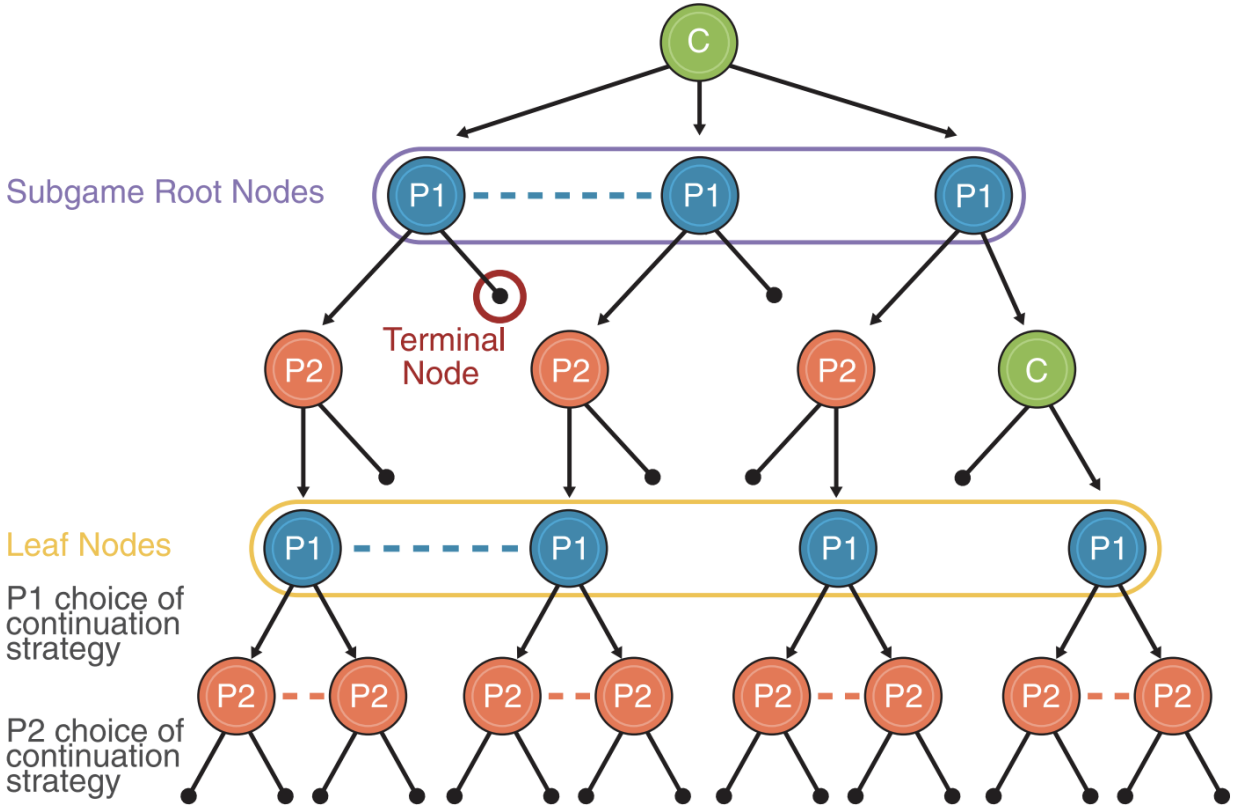

Depth-Limited Rock-Paper-Scissors+



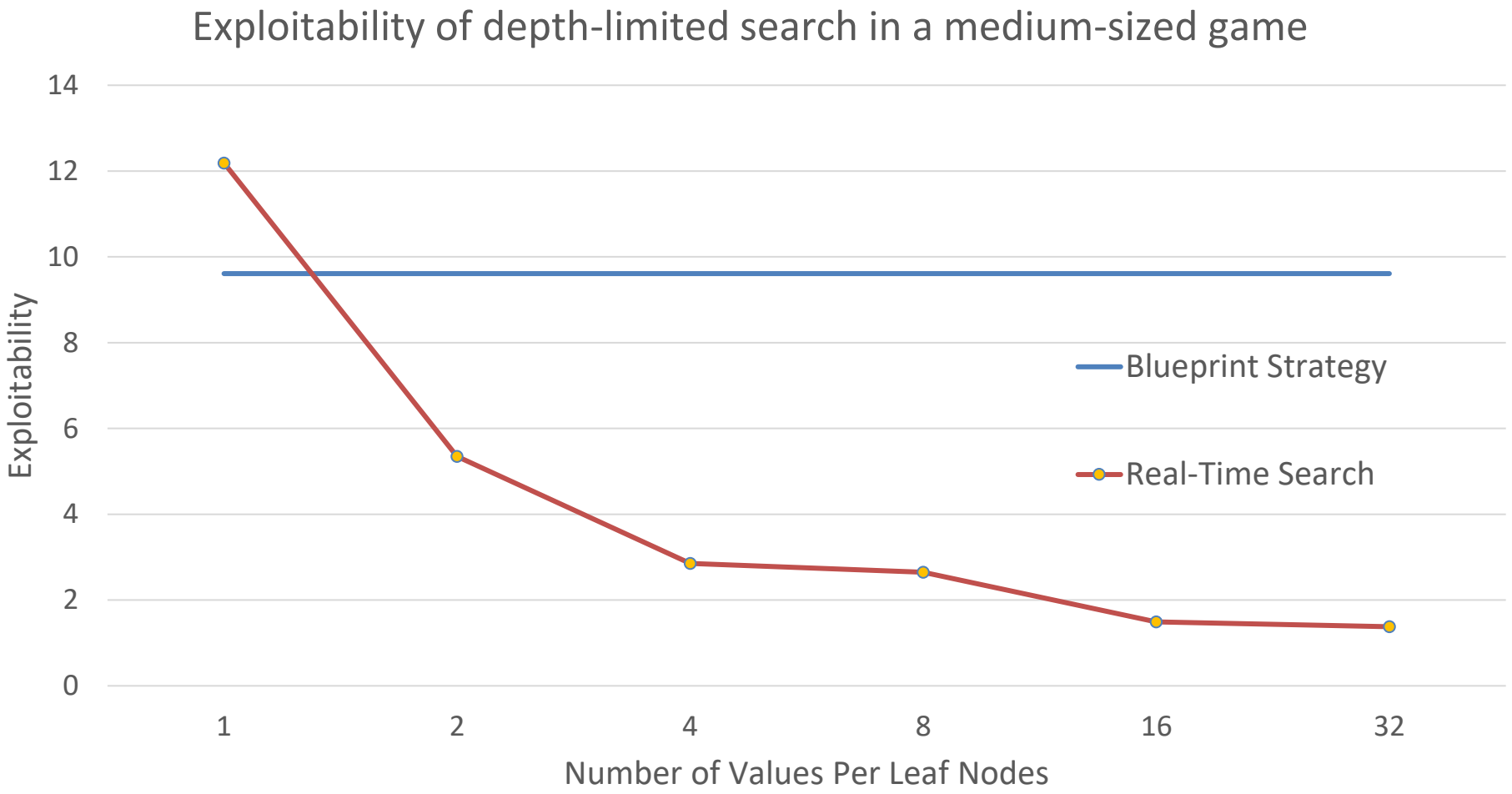
# Depth-Limited Search in Pluribus

[Brown, Sandholm, Amos NeurIPS-18; Brown & Sandholm Science-19]



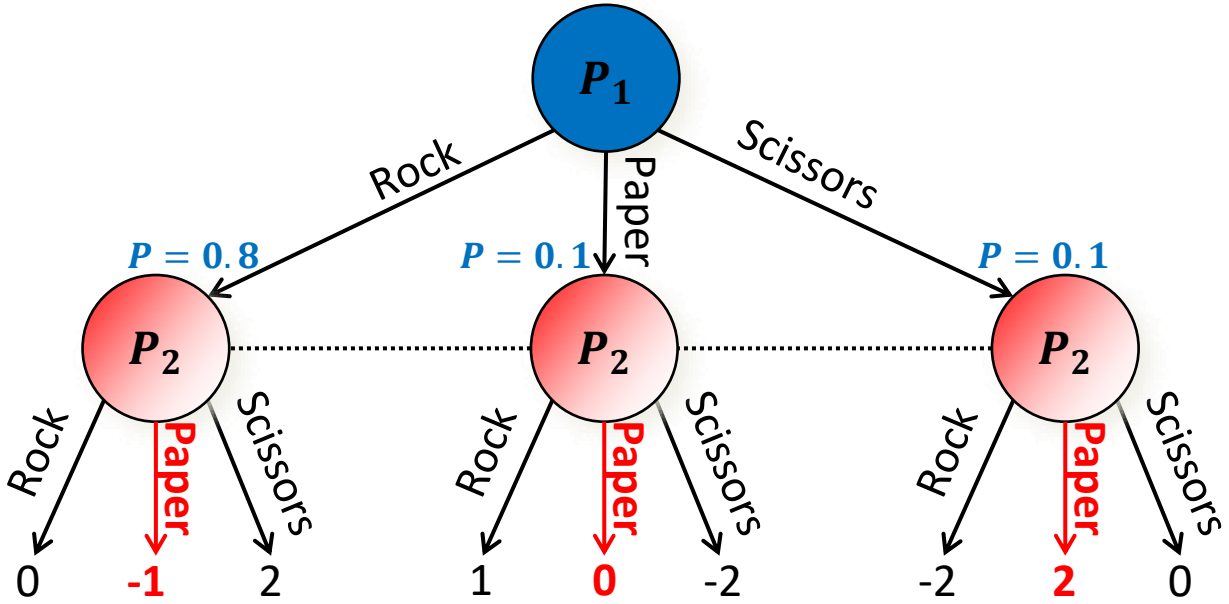



# Exploitability Measurements

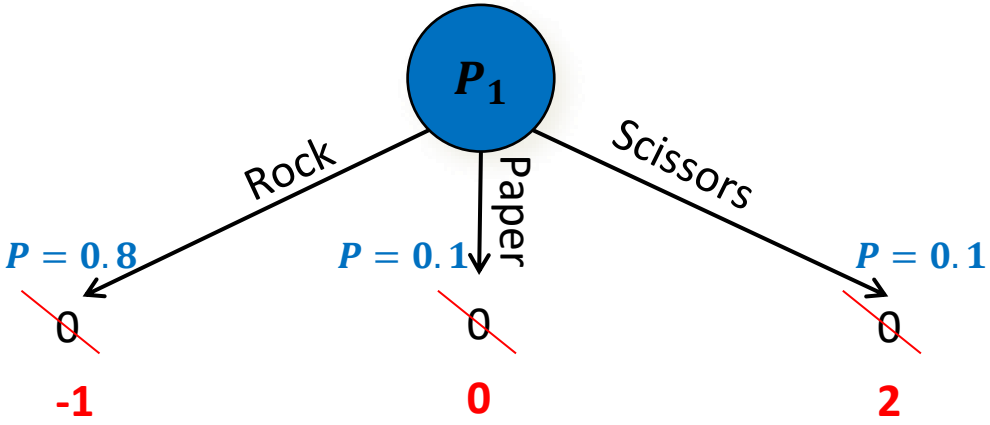


# Search in Imperfect-Information Games

Rock-Paper-Scissors+

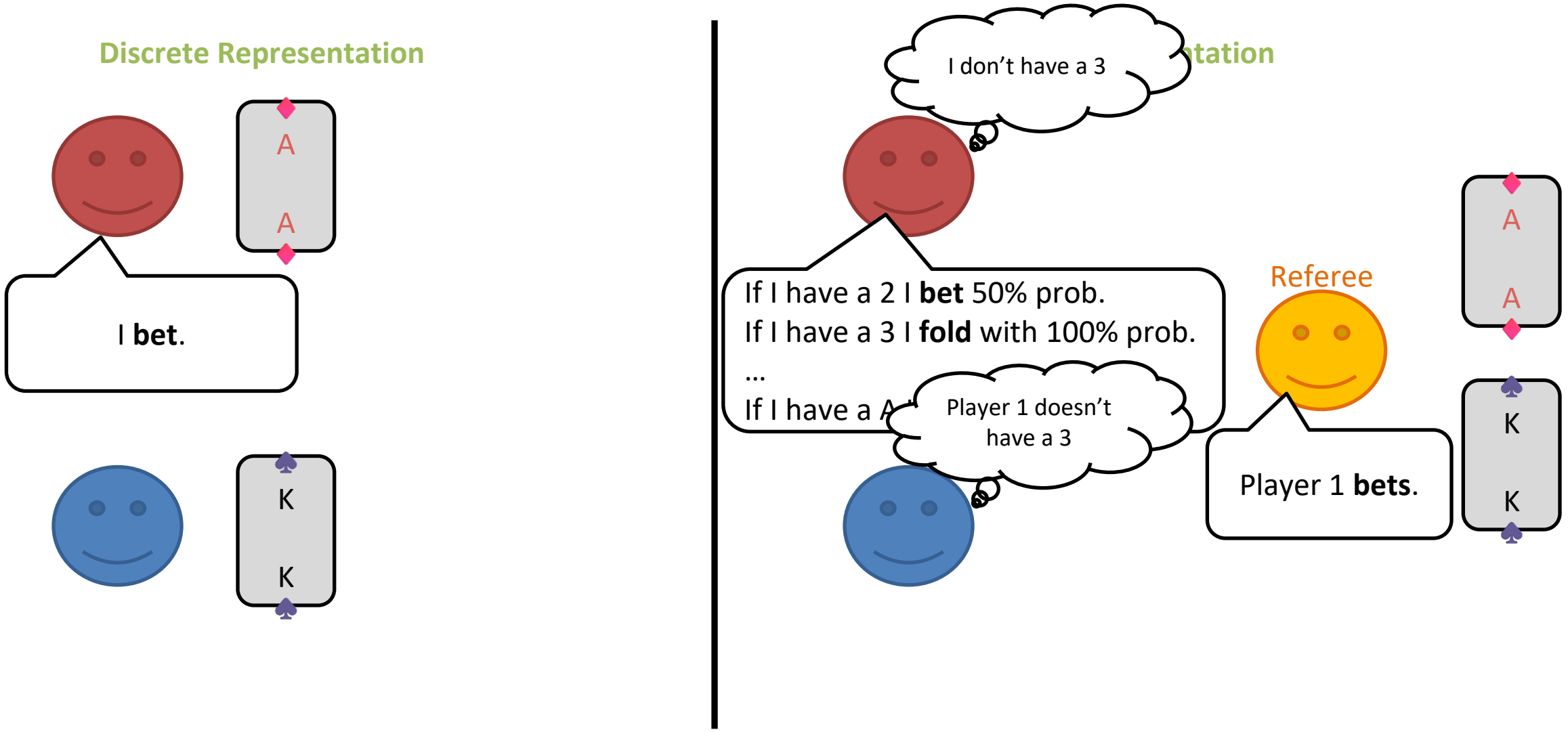



Depth-Limited Rock-Paper-Scissors+

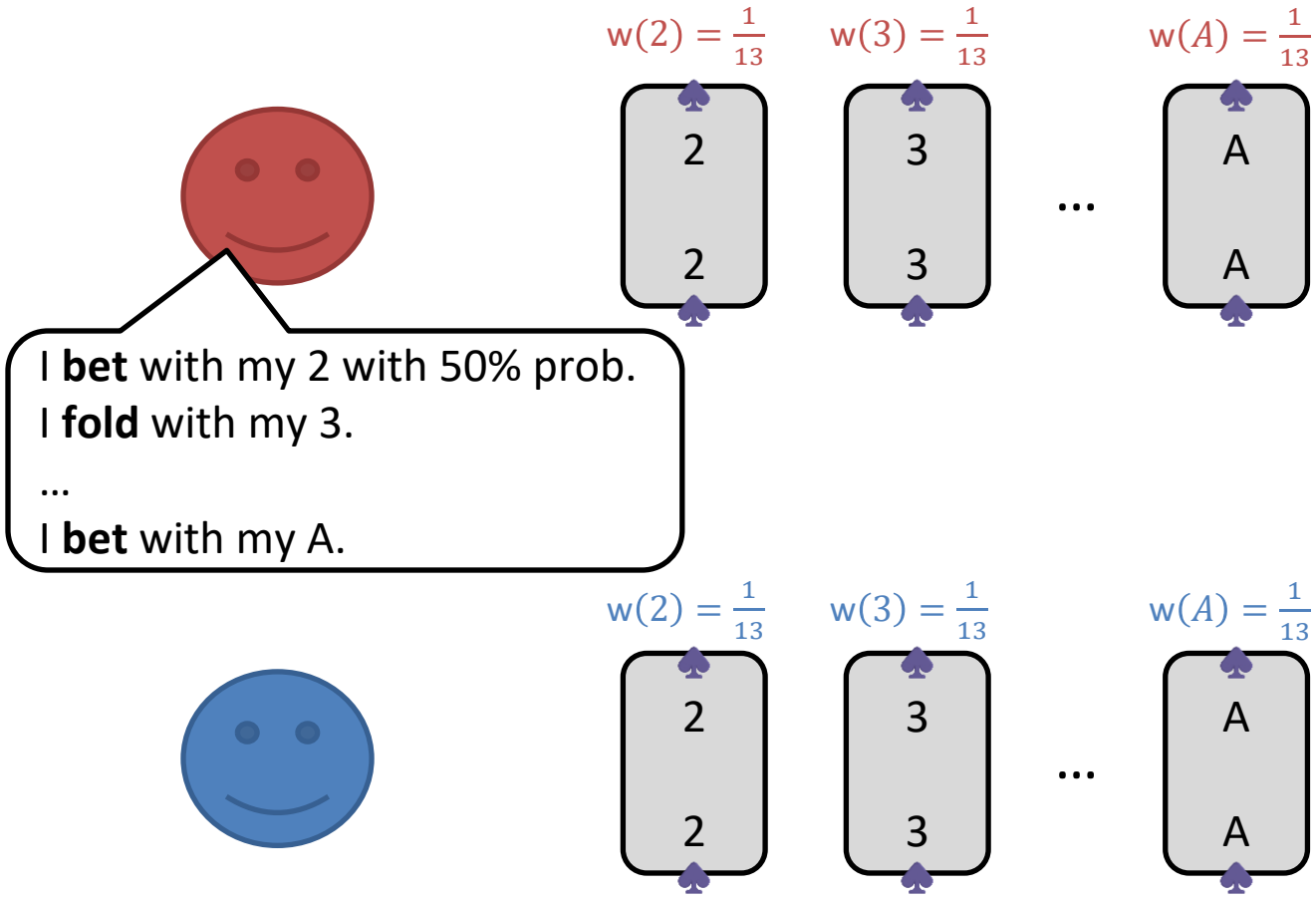



- Another solution: define an imperfect-information game "state" as a **probability distribution over infostates** [Nayyar et al. IEEE-13]
  - $v(Rock)$ is not well-defined
  - $v([0.8\ Rock, 0.1\ Paper, 0.1\ Scissors]) = -0.6$
  - In more complex games, need to include probability distribution for **both** players

# Converting imperfect-info games to continuous-state perfect-info games



# Converting imperfect-info games to continuous-state perfect-info games

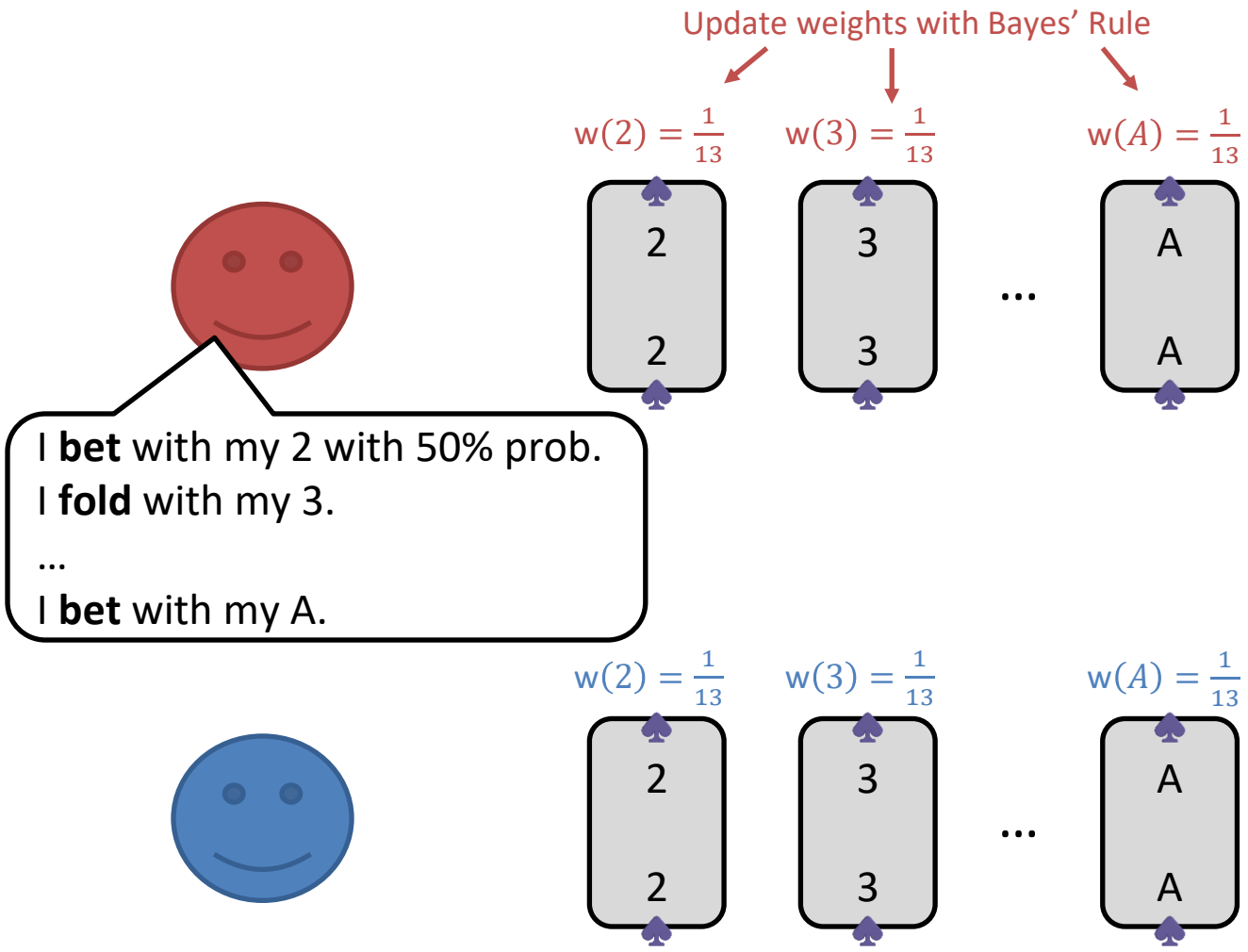




$$P(fold) = 0.08 = \frac{\sum_s P(fold|s)w(s)}{\sum_s w(s)}$$

$$P(bet) = 0.92 = \frac{\sum_s P(bet|s)w(s)}{\sum_s w(s)}$$

# Converting imperfect-info games to continuous-state perfect-info games



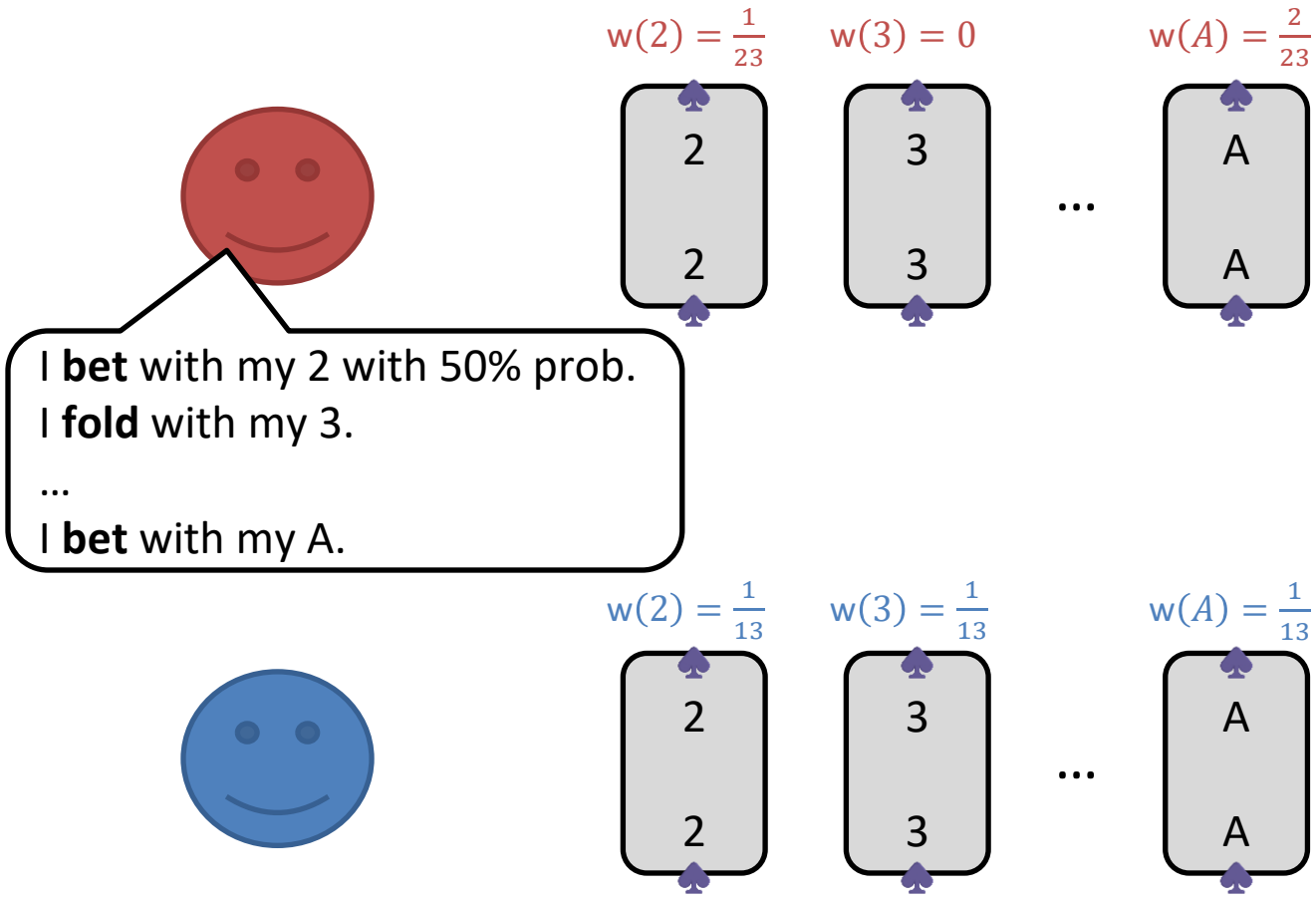

Referee

$$P(fold) = 0.08 = \frac{\sum_s P(fold|s)w(s)}{\sum_s w(s)}$$

$$P(bet) = 0.92 = \frac{\sum_s P(bet|s)w(s)}{\sum_s w(s)}$$

Player 1 **bets**.

# Converting imperfect-info games to continuous-state perfect-info games



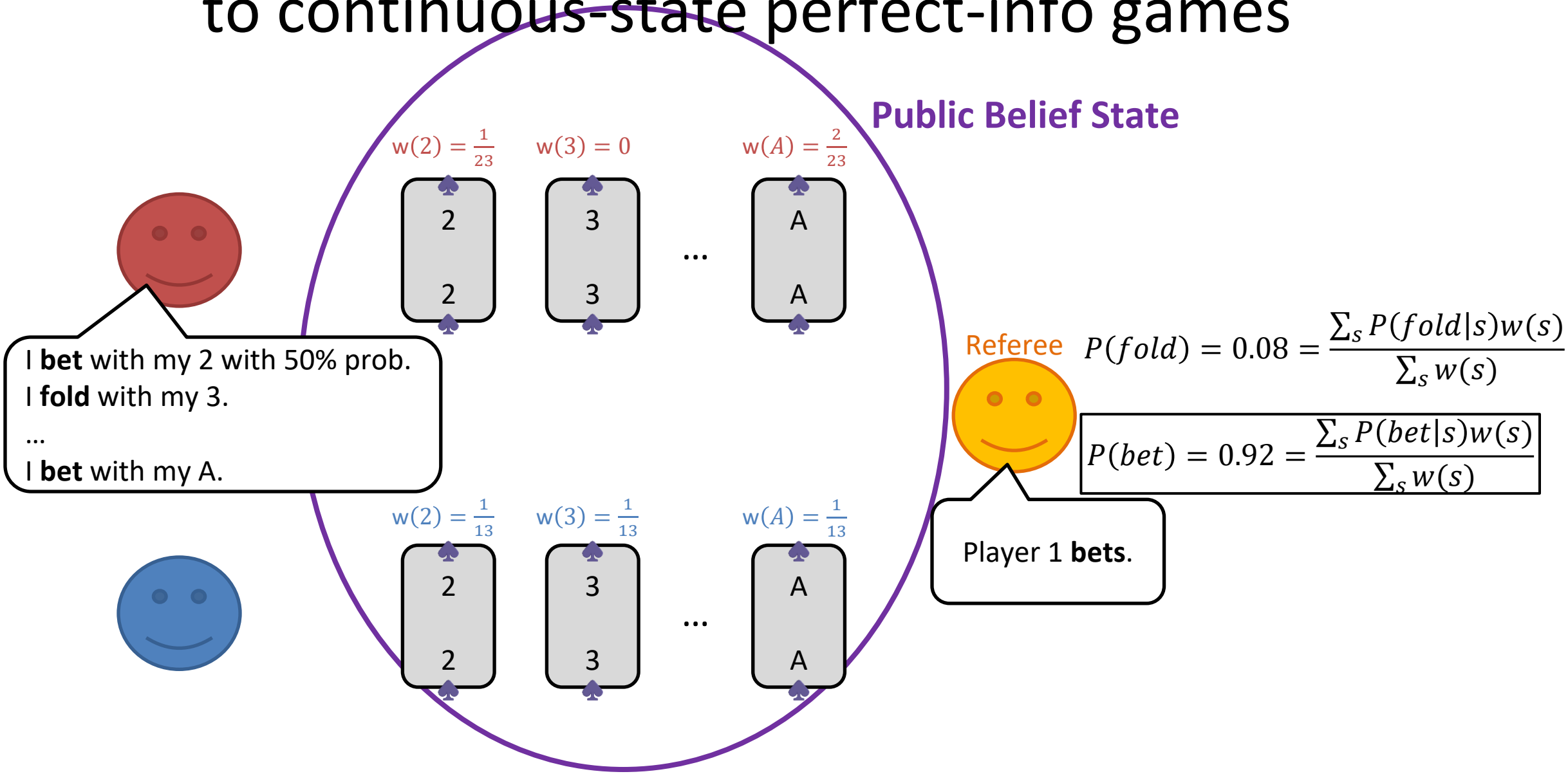

Referee

$$P(fold) = 0.08 = \frac{\sum_s P(fold|s)w(s)}{\sum_s w(s)}$$

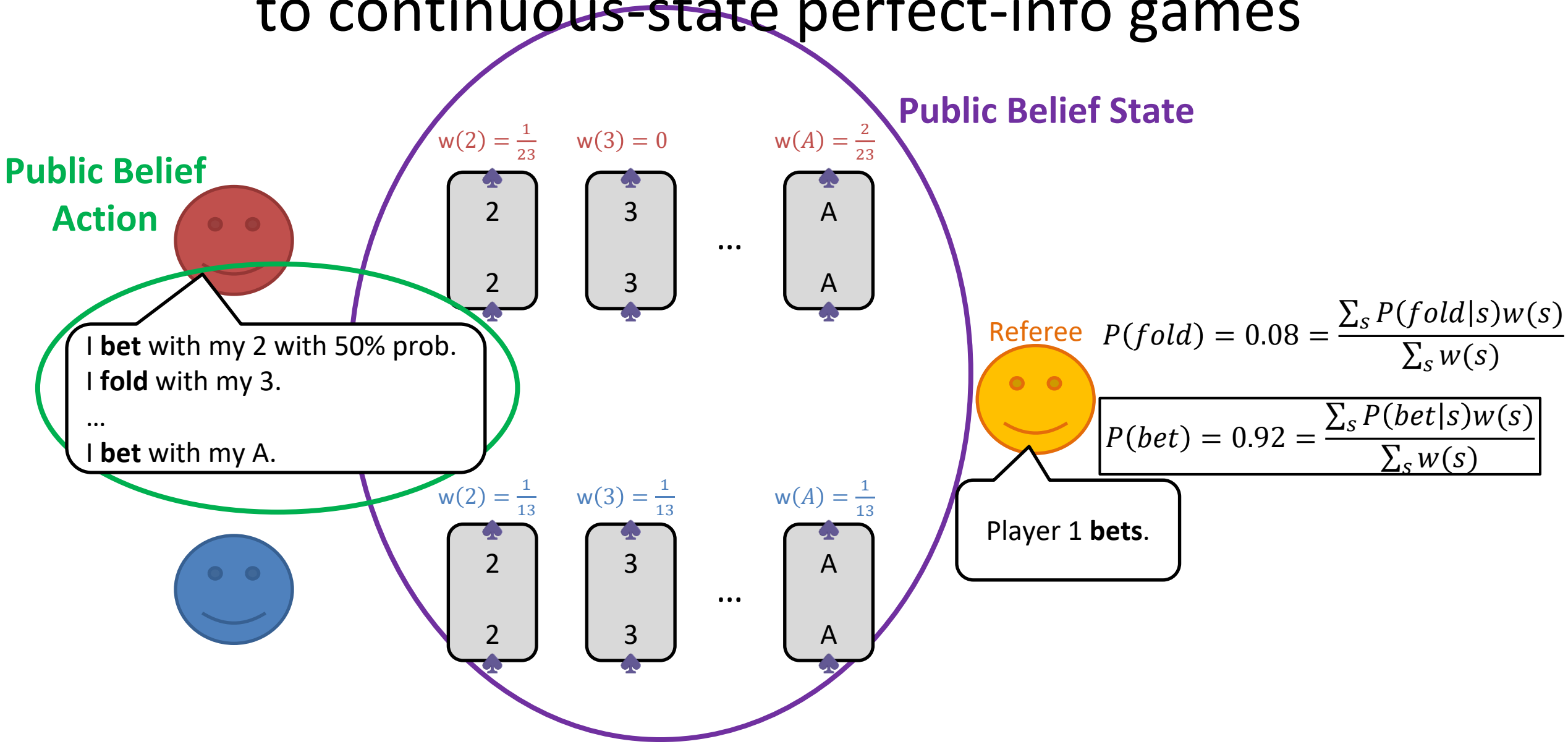
$$P(bet) = 0.92 = \frac{\sum_s P(bet|s)w(s)}{\sum_s w(s)}$$

Player 1 **bets**.

# Converting imperfect-info games to continuous-state perfect-info games



# Converting imperfect-info games to continuous-state perfect-info games



# Search in ReBeL

- We've shown all imperfect-information games can be converted into perfect-information games! Can we now run AlphaZero?

- In theory, yes*. In practice, no.
  - Action space is continuous with potentially *thousands* of dimensions
  - AlphaZero's Monte Carlo tree search would be completely intractable

*Requires techniques introduced in [Sokota, D'Orazio, Ling, Wu, Kolter, Brown (Under review)] that results in unique solutions to subgames

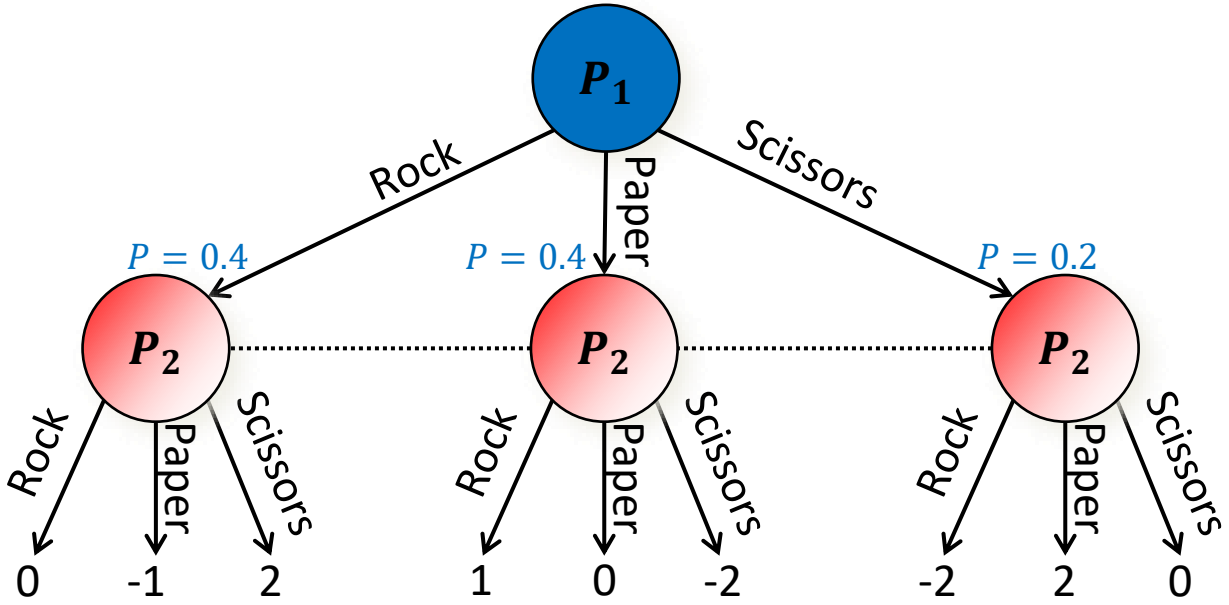
# Search in ReBeL

- But! The continuous action space has special structure
  - Known as a “bilinear saddle point problem”
  - Basically, we leverage convexity

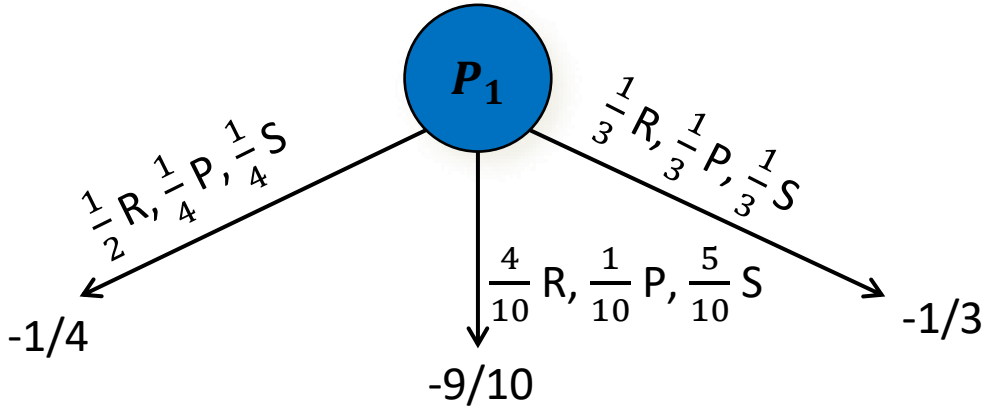
- We can efficiently solve the imperfect-information subgames using regret minimization
  - Other equilibrium-finding algorithms, like gradient descent, also work

# Search in Belief Representation with MCTS

Rock-Paper-Scissors+

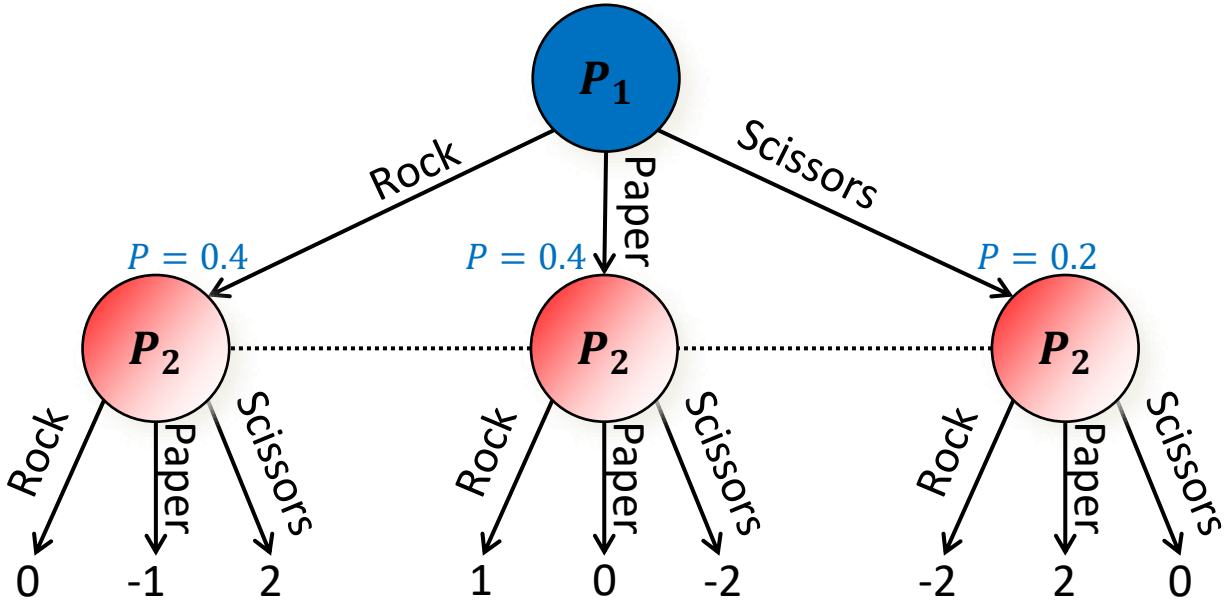



Search in Rock-Paper-Scissors+

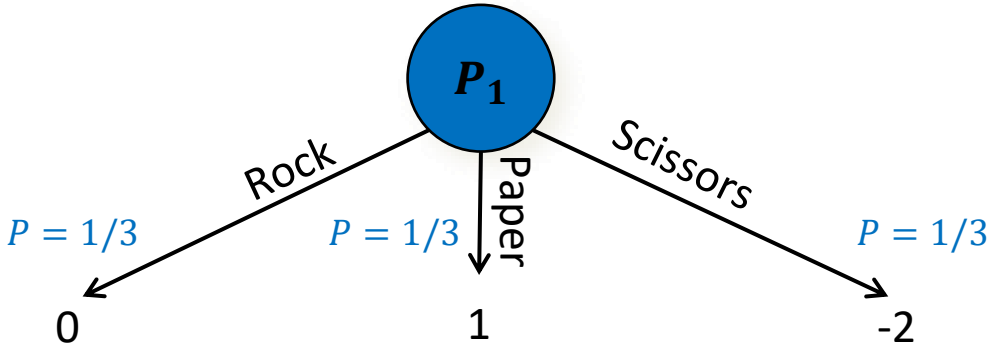


# Search in Discrete Representation with Regret Minimization

Rock-Paper-Scissors+

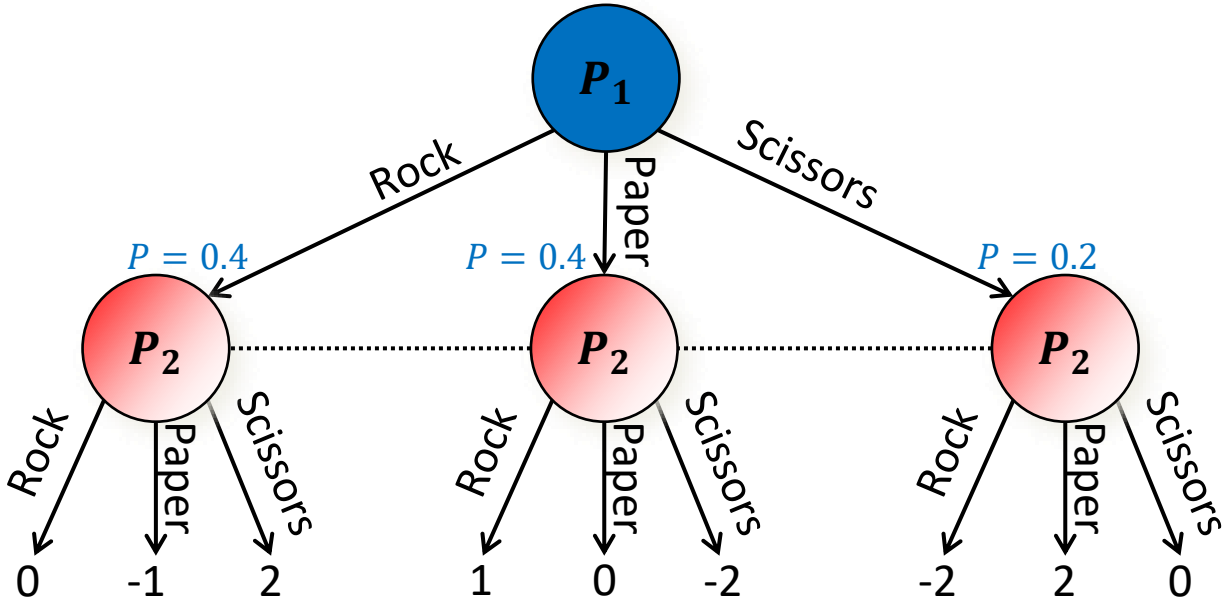



Search in Rock-Paper-Scissors+

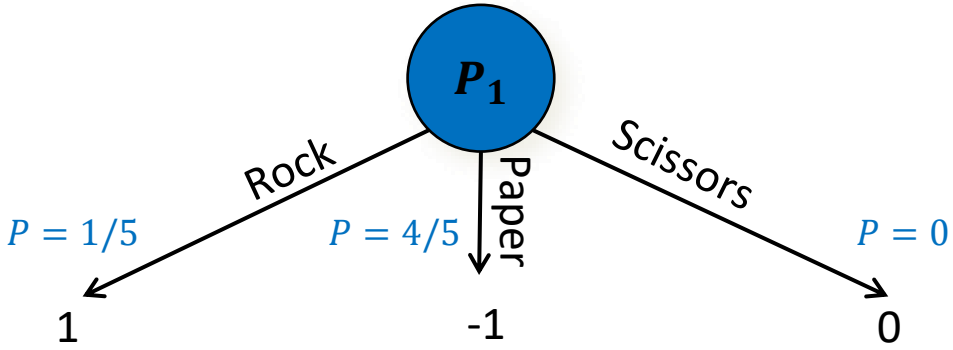


# Search in Discrete Representation with Regret Minimization

Rock-Paper-Scissors+



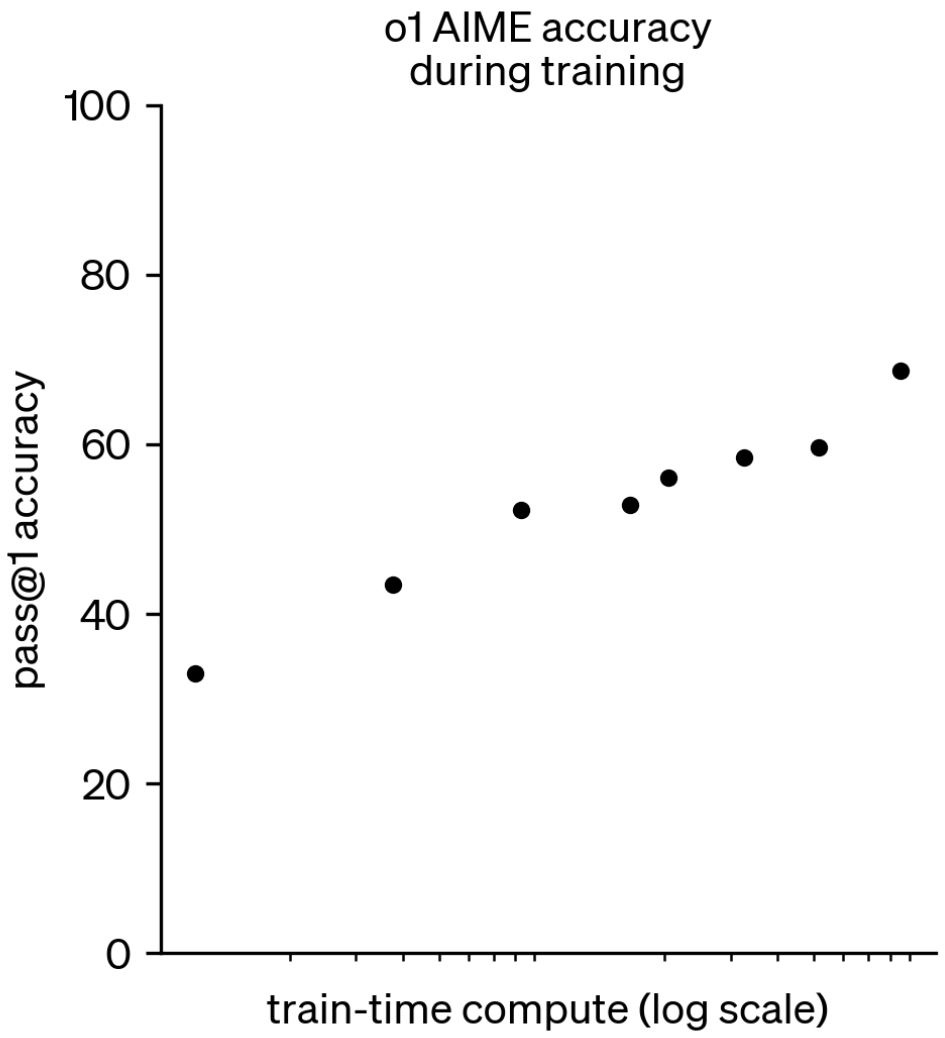

Search in Rock-Paper-Scissors+

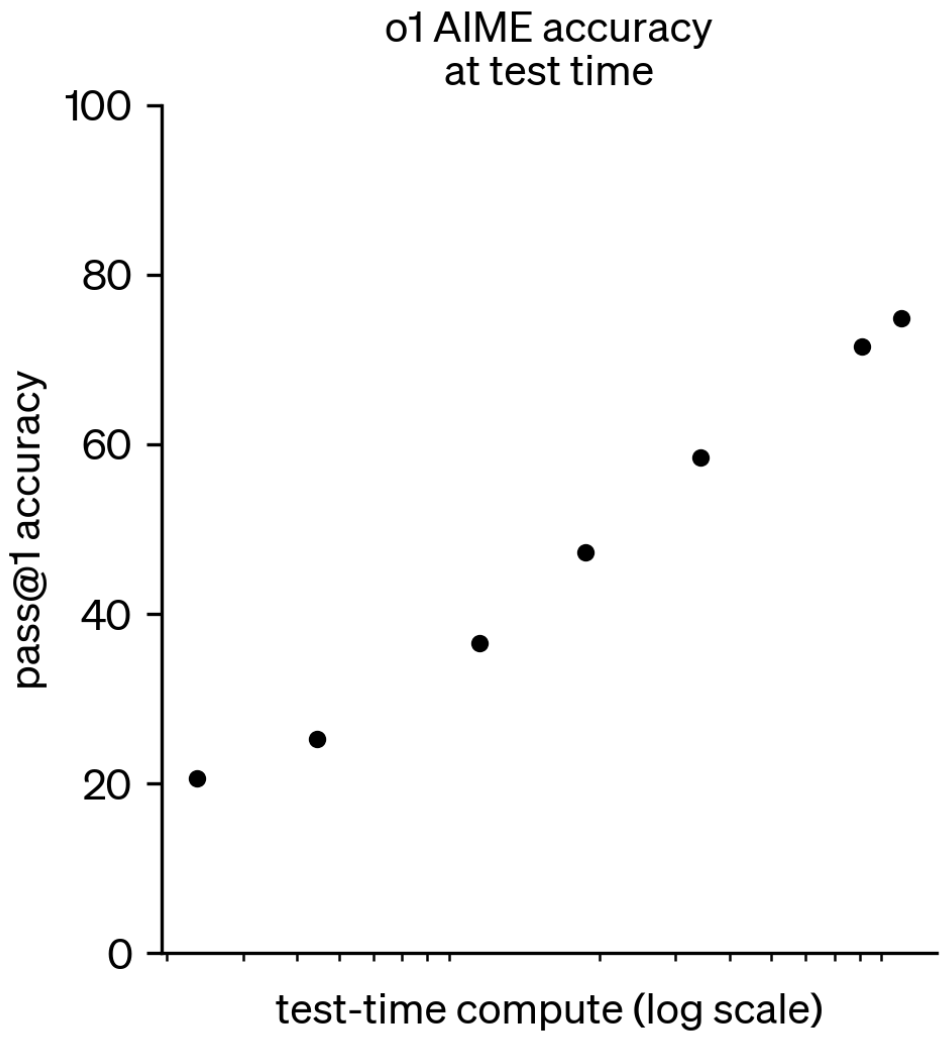



- Theorem: the **average policy over all iterations** converges to a **Nash equilibrium**
  [Hart & Mas-Colell '00]
  - Equivalently, playing a **random iteration's** policy results in a **Nash equilibrium in expectation**

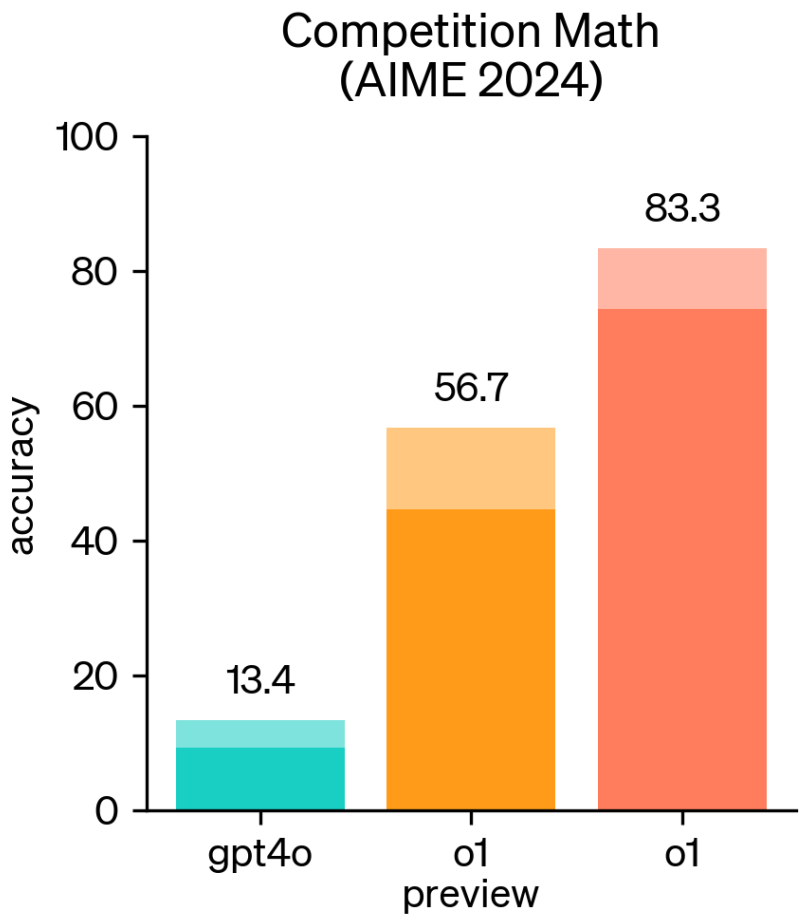
Is there a general way to scale inference compute in LLMs?

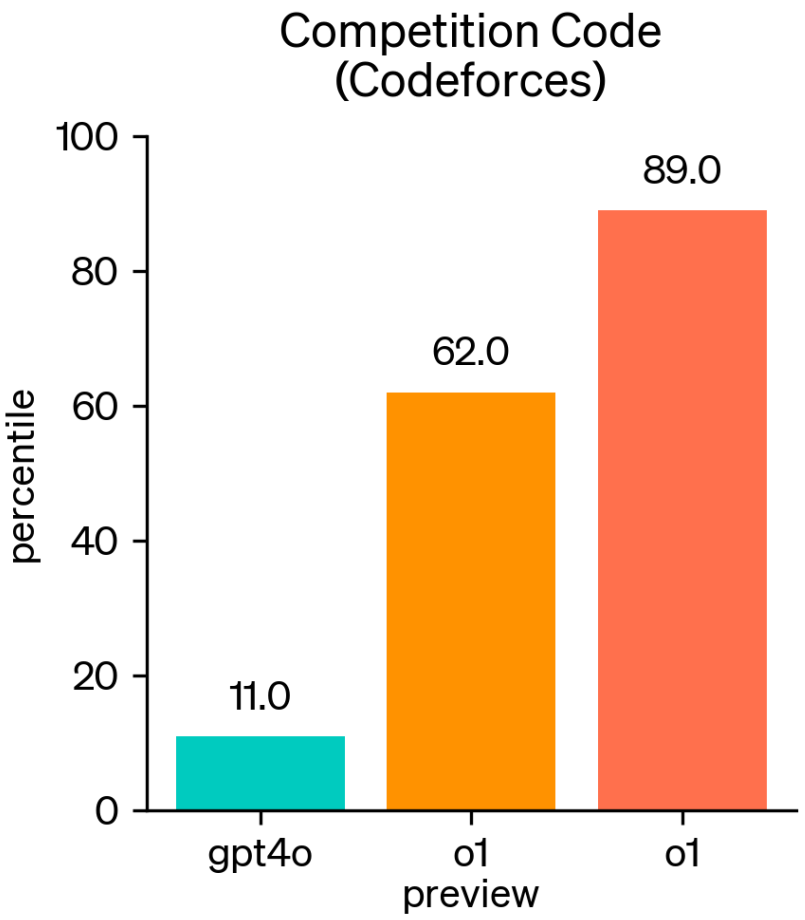
# OpenAI o1

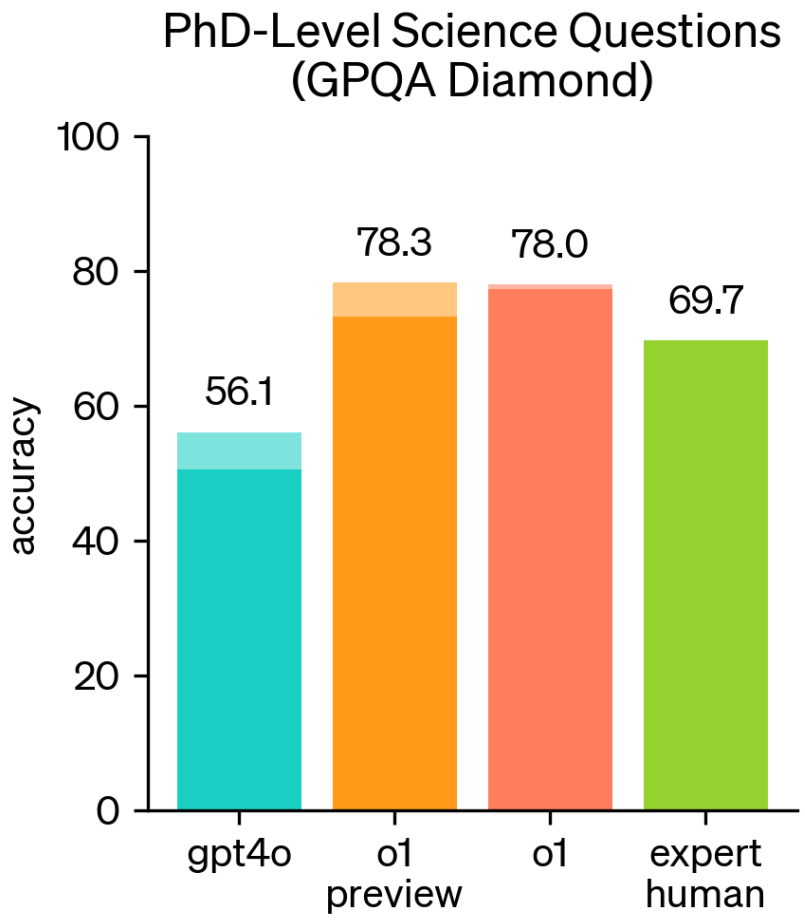

o1 AIME accuracy
during training
100
80
60
40
20
0
pass@1 accuracy
train-time compute (log scale)


o1 AIME accuracy
at test time
100
80
60
40
20
0
pass@1 accuracy
test-time compute (log scale)

# Evals of OpenAI o1-preview and o1



# Scaling Scaling Laws with Board Games (Hex)

[Andy L. Jones, arXiv-2021]

Knowing now that compute can be spent in two places, at train time and test time, the immediate question is: how do these two budgets trade off? This is illustrated in Fig. 9, which shows that the trade-off is linear in log-compute: for each additional 10× of train-time compute, about 15× of test-time compute can be eliminated, down to a floor of a single-node tree search.

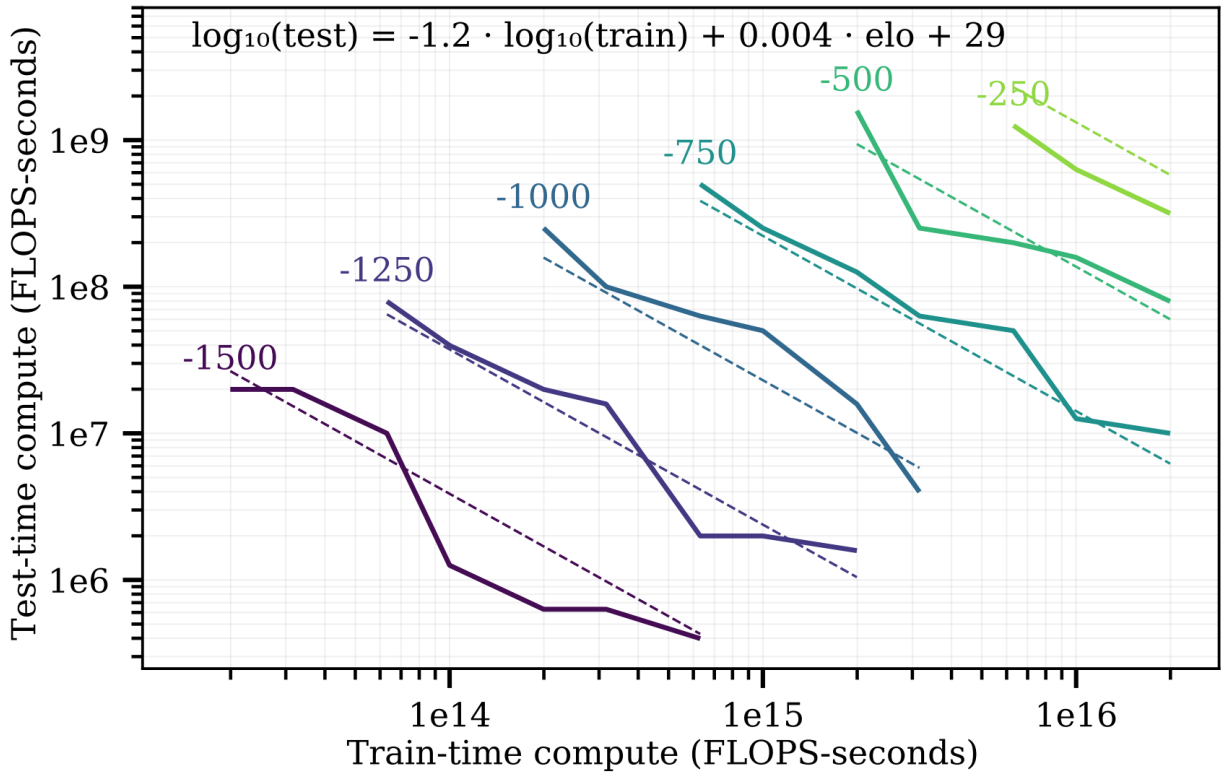



Fig. 9. The trade-off between train-time compute and test-time compute. Each dotted line gives the minimum train-test compute required for a certain Elo on a 9 × 9 board

# Where does this go?

- Much **higher inference compute**, but much **more capable models**
  - What inference cost are you willing to pay for a proof of the Riemann Hypothesis?
  - What inference cost are you willing to pay for new life-saving drugs?
- There are still algorithmic improvements
- There is still room to push inference compute much further
- AI can be more than chatbots

# What's next for me?

- We are launching a new **multi-agent** reasoning team
  - Self-play
  - Multi-agent cooperation
  - AI debate

- Looking for strong engineers that are interested in research
  - Prior multi-agent research experience not required
  - RL, tool use, and LLM experience is helpful

- If you are interested, apply at https://jobs.ashbyhq.com/openai/form/oai-multi-agent

# The Bitter Lesson by Richard Sutton

"The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective… The two methods that seem to scale arbitrarily in this way are ***search*** and ***learning***."

# Thank You!

**Noam Brown**
**noam.brown@gmail.com**
**@polynoamial**
**www.noambrown.com**

# Lunch 😋

Leave any type of feedback at **pkr.bot/feedback**!

Be back in your seats in 5 min!

# Giveaway Winners!

# Final Event RSVP Raffle:
# kerbs “justinwz” (1st) and “annieguo”

# Thanks for coming!

Poker Social with Noam: 32-044, 2-4PM

RSVP for Final Event: **pkr.bot/rsvp**

RSVP for Poker Tournament: **pkr.bot/tournament**

Make sure to check **pkr.bot/piazza** for updates