Monte Carlo CFR (MCCFR): Sampling for Regret Minimization in Games

Marc Lanctot[1]

Department of Computing Science
University of Alberta
lanctot@ualberta.ca

June 18, 2009

[1] Joint work with Michael Bowling, Kevin Waugh, and Martin Zinkevich

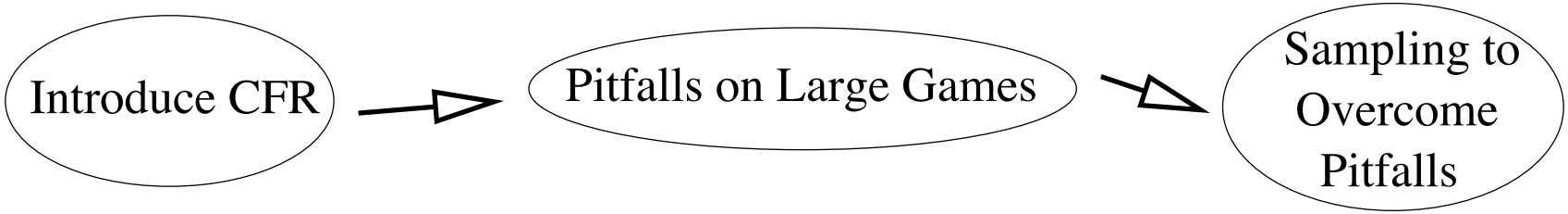
Intro + Motivation

- Our goal is to:

  find Nash equilibria in large, zero-sum games.

- *This is a difficult problem.*
- Many multi-agent problems can be formulated as games.
- We want to know how to do this efficiently for very large games.

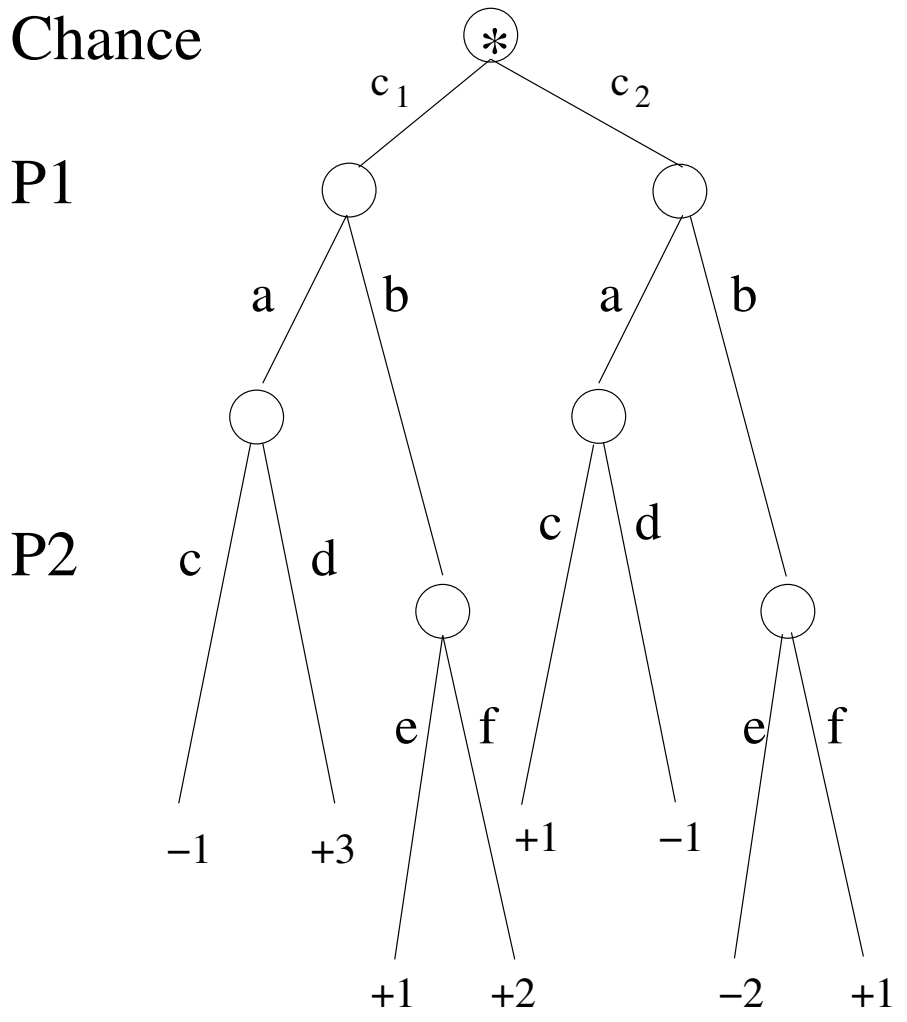
Talk overview:



Terminology I : Extensive-Form Games

An **extensive game** is represented in tree form.

Example:

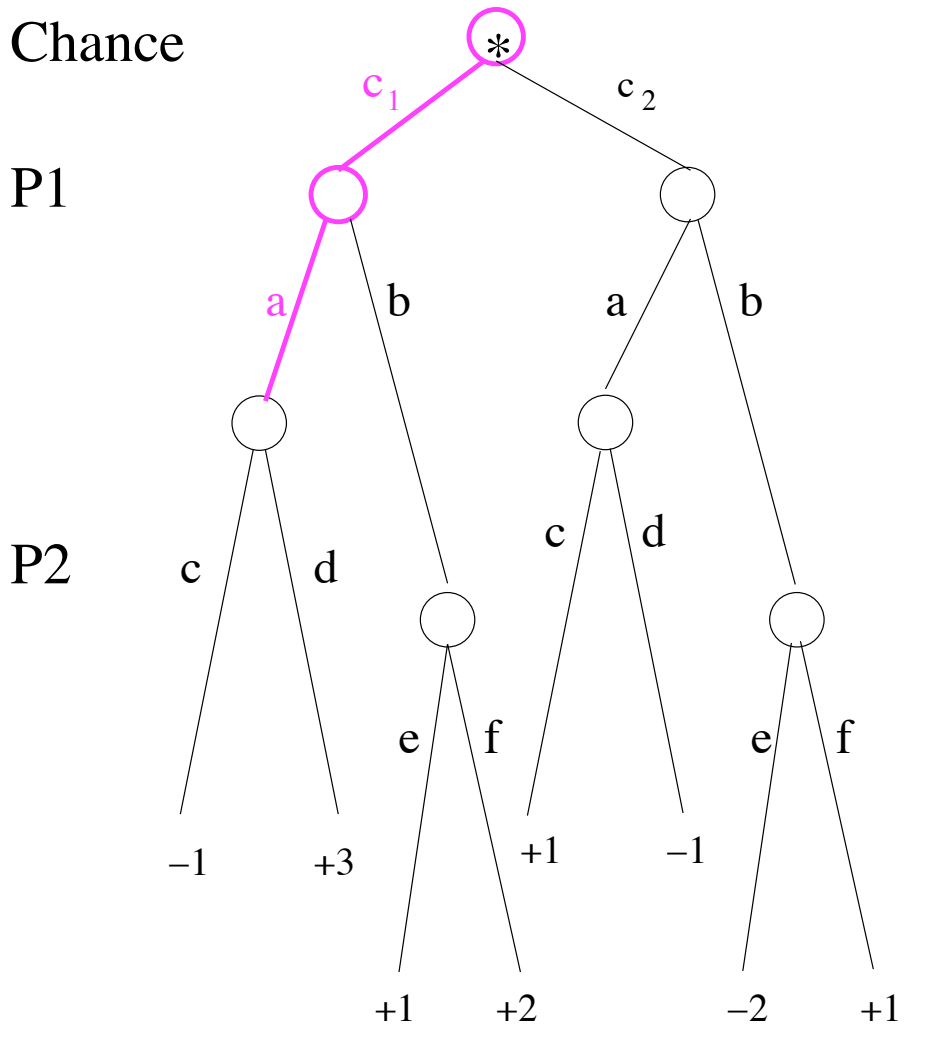


Terminology I : Extensive-Form Games

An **extensive game** is represented in tree form.

- $h \in H$ is possible history;

Example:

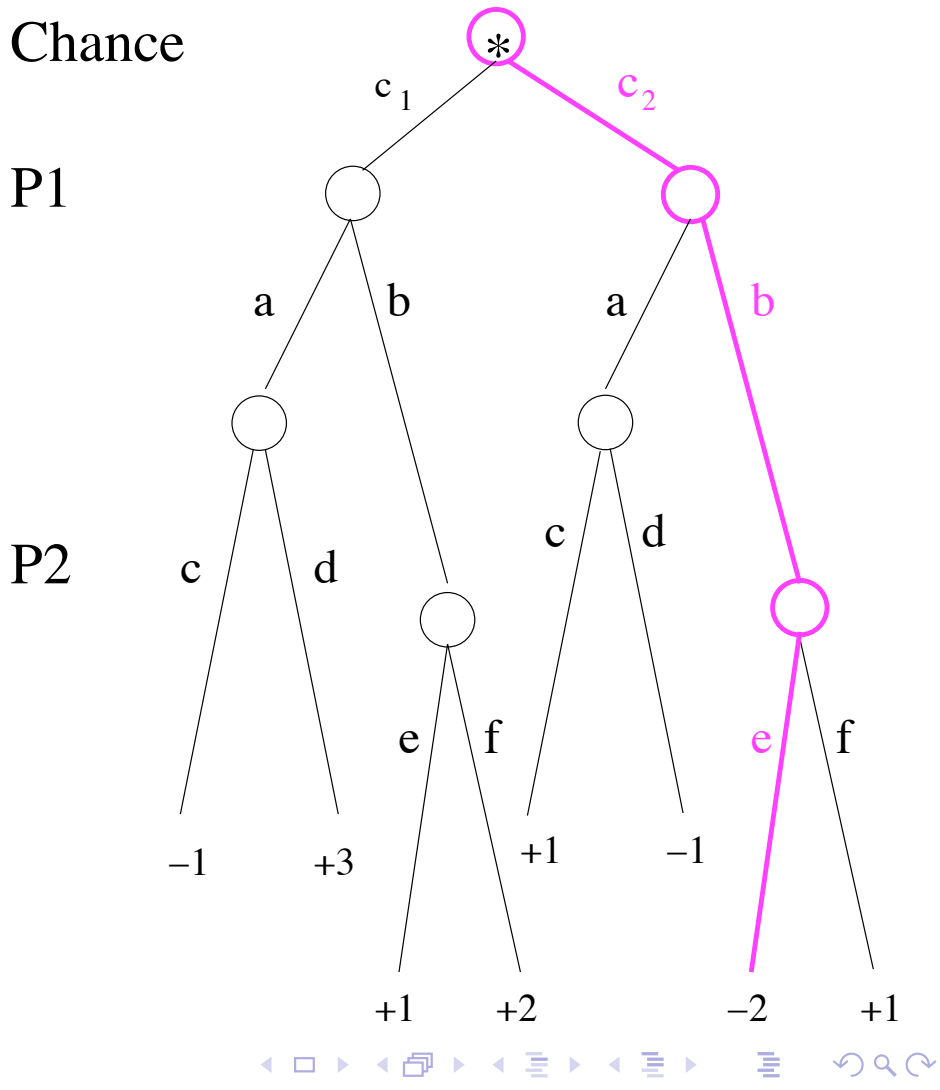


Terminology I : Extensive-Form Games

An **extensive game** is represented in tree form.

- $h \in H$ is possible history;
  $z \in Z, Z \subseteq H$ is a *terminal* history.

Example:

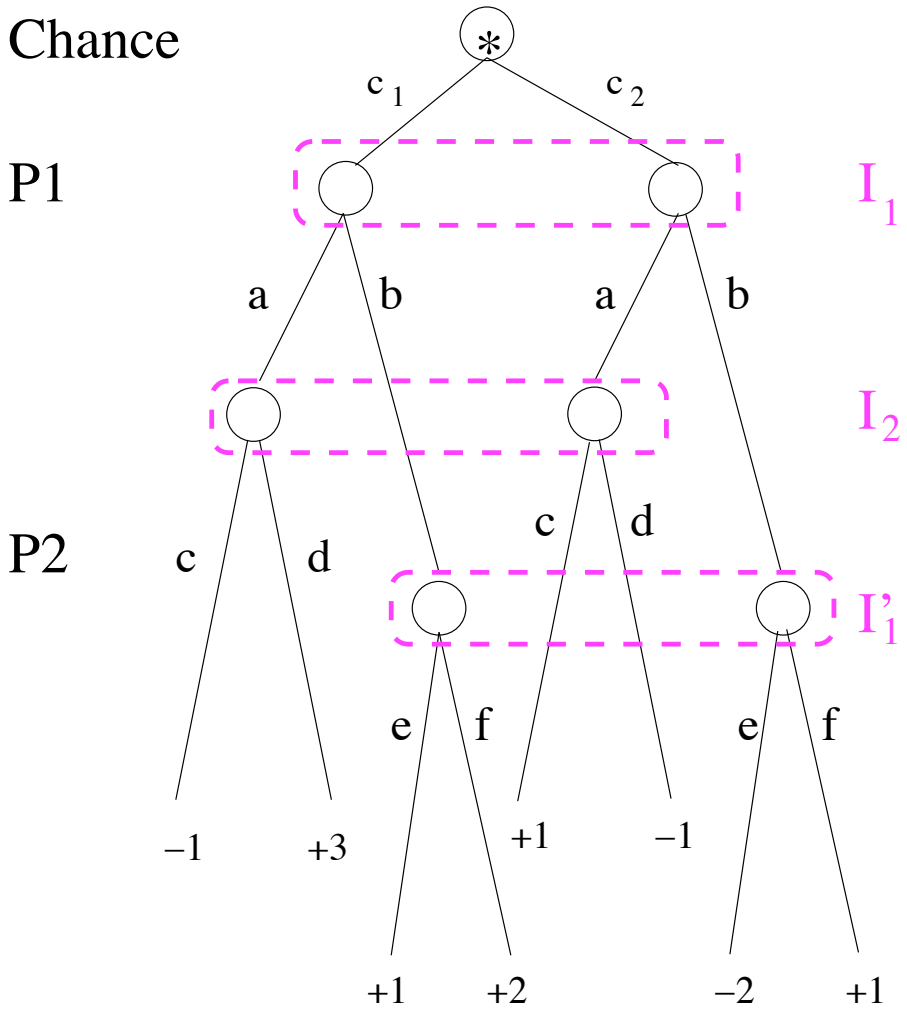


Terminology I : Extensive-Form Games

An **extensive game** is represented in tree form.

- $h \in H$ is possible history; $z \in Z, Z \subseteq H$ is a *terminal* history.
- An information set $I_i \in \mathcal{I}$ is an information set for player i.

Example:

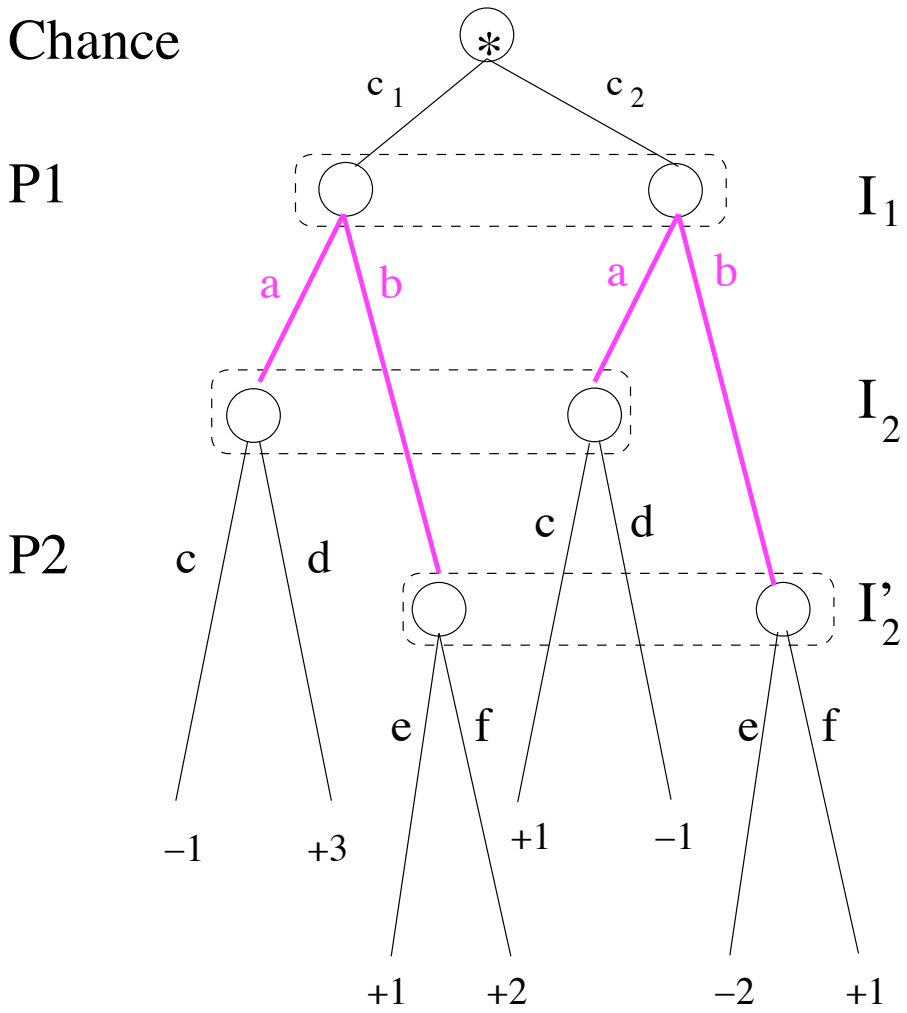


Terminology I : Extensive-Form Games

An **extensive game** is represented in tree form.

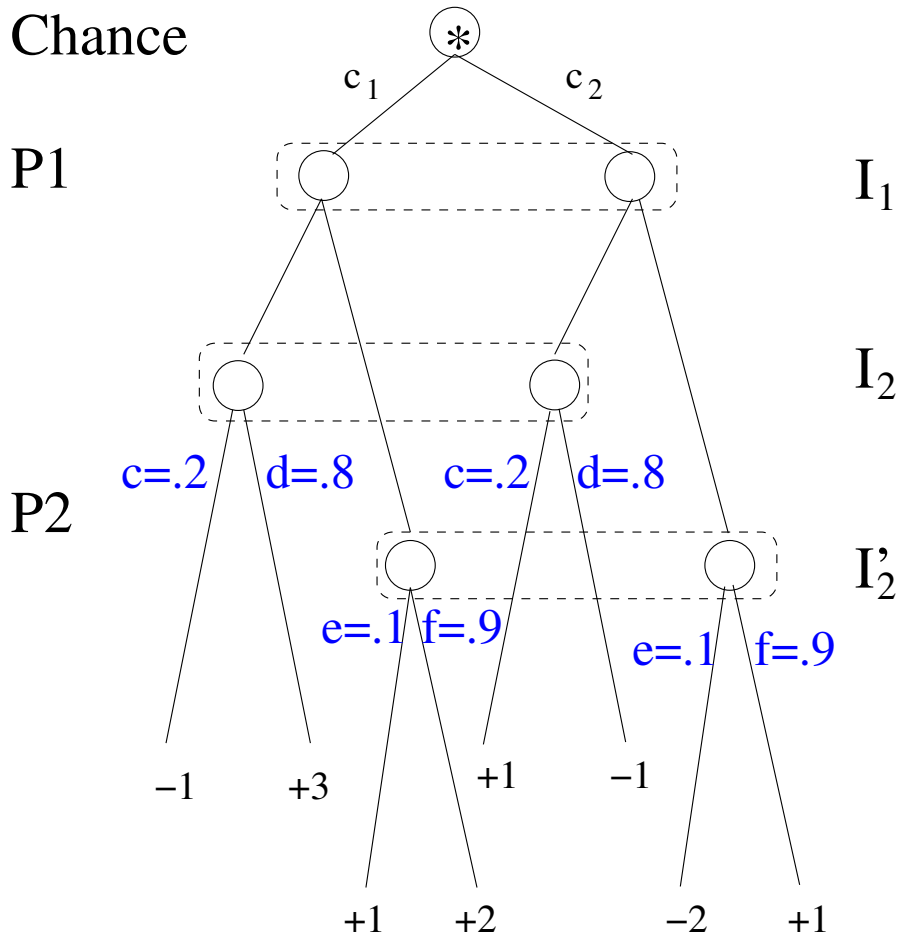
- $h \in H$ is possible history; $z \in Z, Z \subseteq H$ is a *terminal* history.
- An information set $I_i \in \mathcal{I}$ is an information set for player i.
- $A(I_i)$ is the action set for i at information set I_i.

Example:



Terminology II : Strategies

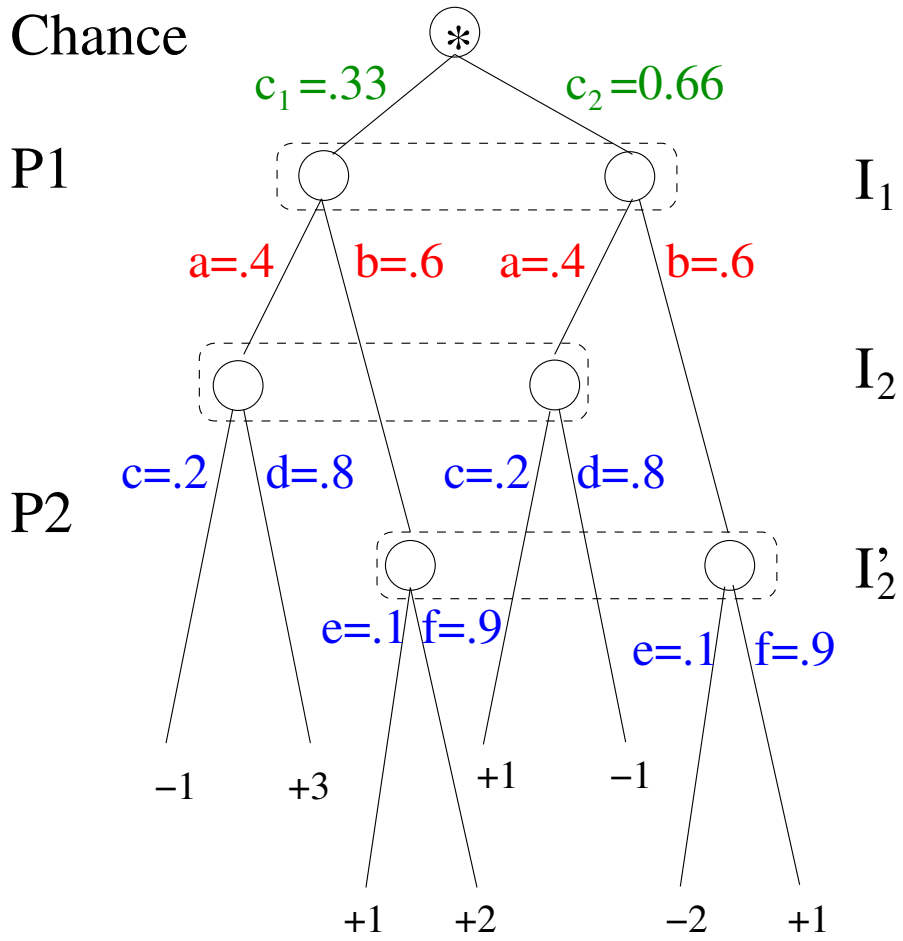
A **strategy** $\sigma_i \in \Sigma_i$ is a distribution from $I_i \longrightarrow A(I_i)$.



Terminology II : Strategies

A **strategy** $\sigma_i \in \Sigma_i$ is a distribution from $I_i \longrightarrow A(I_i)$.

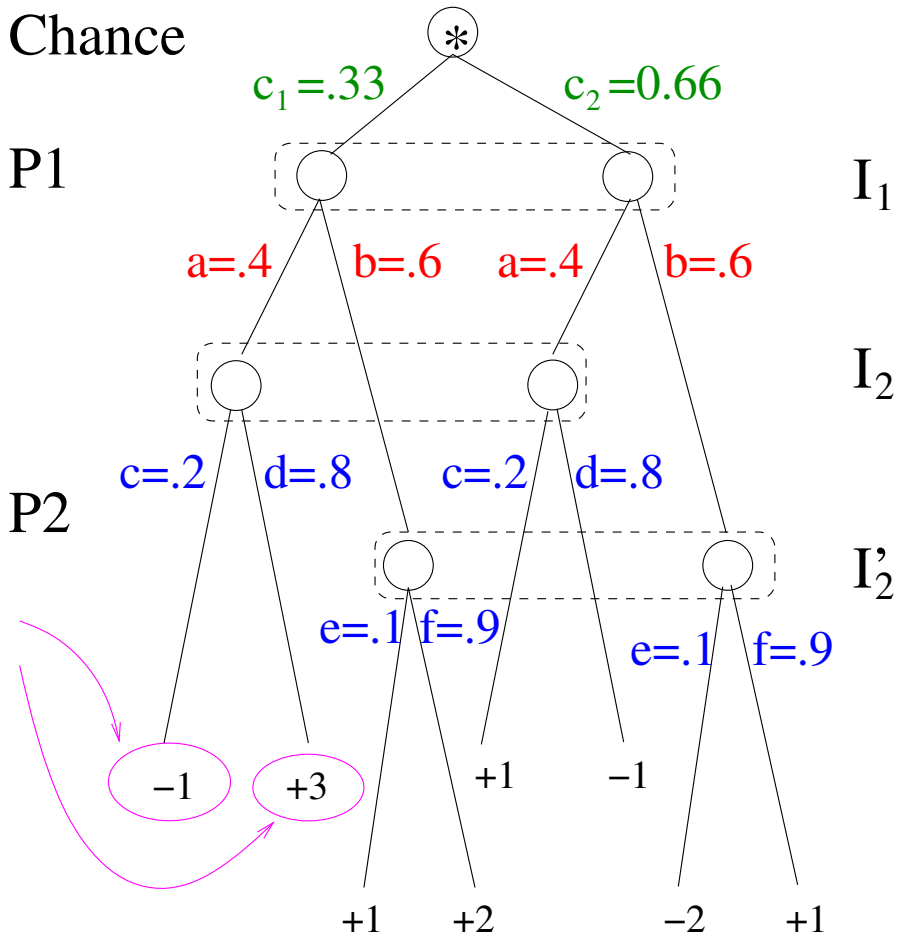
- A strategy σ_{-i} is a strategy for the opponents of i and chance.
- A strategy profile $\sigma = (\sigma_1, \sigma_2)$.



Terminology II : Strategies

A **strategy** $\sigma_i \in \Sigma_i$ is a distribution from $I_i \rightarrow A(I_i)$.

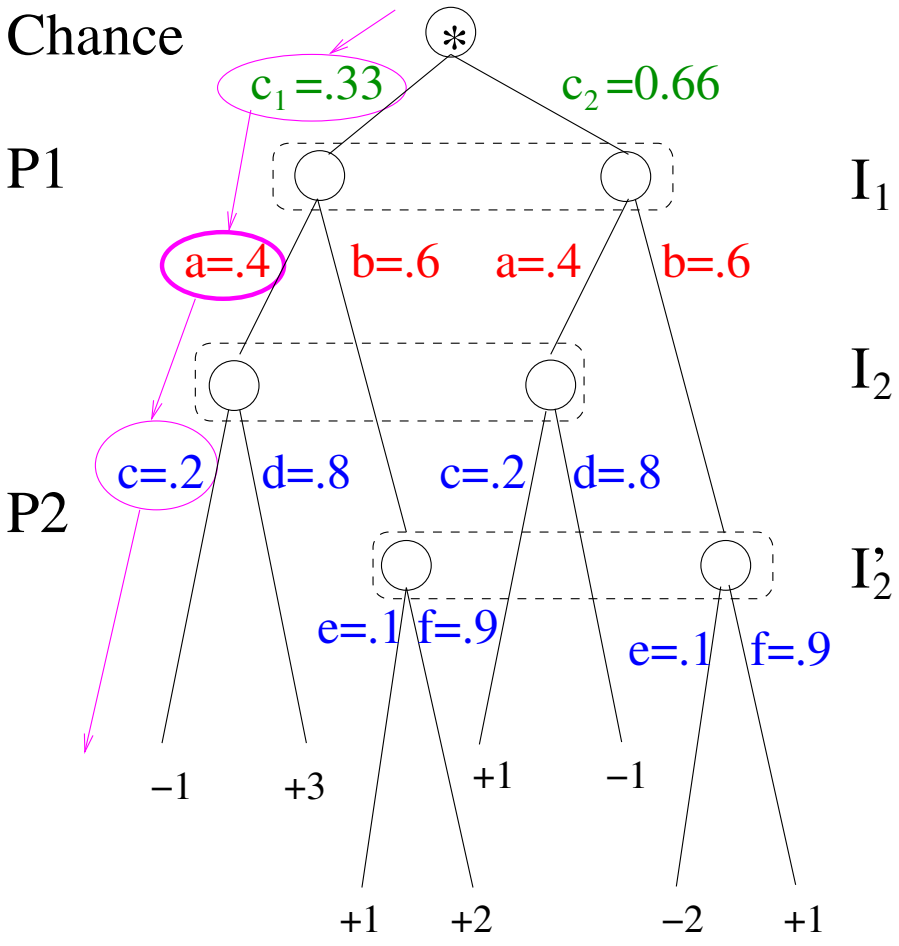
- A strategy σ_{-i} is a strategy for the opponents of i and chance.
- A strategy profile $\sigma = (\sigma_1, \sigma_2)$.
- $u_i(z)$ is the payoff to player i when players play z.



Terminology II : Strategies

A **strategy** $\sigma_i \in \Sigma_i$ is a distribution from $I_i \rightarrow A(I_i)$.

- A strategy σ_{-i} is a strategy for the opponents of i and chance.
- A strategy profile $\sigma = (\sigma_1, \sigma_2)$.
- $u_i(z)$ is the payoff to player i when players play z.
- $\pi^\sigma(h)$ is a product of probabilities along history h. $\pi_i^\sigma(h)$ is player i's contribution.



Counterfactual Regret Minimization (CFR) : Overview

CFR is an iterative strategy altering algorithm:

$$t = 1$$

Player 1 strategies: σ_1^1

Player 2 strategies: σ_2^1

Counterfactual Regret Minimization (CFR) : Overview

CFR is an iterative strategy altering algorithm:

$$t = 1 \qquad\qquad t = 2$$

Player 1 strategies: $\sigma_1^1 \quad \rightarrow \quad \sigma_1^2$

Player 2 strategies: $\sigma_2^1 \quad \rightarrow \quad \sigma_2^2$

Counterfactual Regret Minimization (CFR) : Overview

CFR is an iterative strategy altering algorithm:

	$t=1$		$t=2$		$t=3$	$\cdots$
Player 1 strategies:	σ_1^1	$\rightarrow$	σ_1^2	$\rightarrow$	σ_1^3	$\cdots$
Player 2 strategies:	σ_2^1	$\rightarrow$	σ_2^2	$\rightarrow$	σ_2^3	$\cdots$

Counterfactual Regret Minimization (CFR) : Overview

CFR is an iterative strategy altering algorithm:

$$\begin{array}{lccccccc} & t=1 & & t=2 & & t=3 & \cdots \\ \text{Player 1 strategies:} & \sigma_1^1 & \rightarrow & \sigma_1^2 & \rightarrow & \sigma_1^3 & \cdots \\ \text{Player 2 strategies:} & \sigma_2^1 & \rightarrow & \sigma_2^2 & \rightarrow & \sigma_2^3 & \cdots \end{array}$$

Let R_i^T be the **average overall regret** of using σ^t after T steps.

$$R_i^T \leq \epsilon \quad \Rightarrow \quad \text{the average profile } (\bar{\sigma}_1^T, \bar{\sigma}_2^T) \text{ is a } 2\epsilon\text{-Nash.}$$

- σ is ϵ-Nash if a player can do ϵ better by switching to σ_i'.
- σ is Nash if no player can do better by switching strategies.

CFR Algorithm I

1. Minimize **immediate counterfactual regret** $R^T_{i,\text{imm}}(I)$
2. Overall regret bounded[2] by

$$R_i^T \leq \sum_{I \in \mathcal{I}_i} R^{T,+}_{i,\text{imm}}(I)$$

3. Using regret-matching[3] and Blackwell's approachability to update strategies σ^t at each information set, then[2]

$$R_i^T \leq \frac{\Delta_{u,i}|\mathcal{I}_i|\sqrt{|A_i|}}{\sqrt{T}}$$

where $\Delta_{u,i}$ is a payoff range for i.

---

[2] See [Zinkevich et. al., 2008] for details.

[3] Normalizing positive portions of accumulated regret.

CFR Algorithm II (Example)

Define **counterfactual value** as

$$v_i(\sigma, I) = \sum_{h \in I, z \in Z, h \sqsubset z} \pi^{\sigma}_{-i}(h) \pi^{\sigma}(h, z) u_i(z)$$

Define $v_i(\sigma_{(I \to a)}, I)$ similarly, except take a at I

CFR Algorithm II (Example)

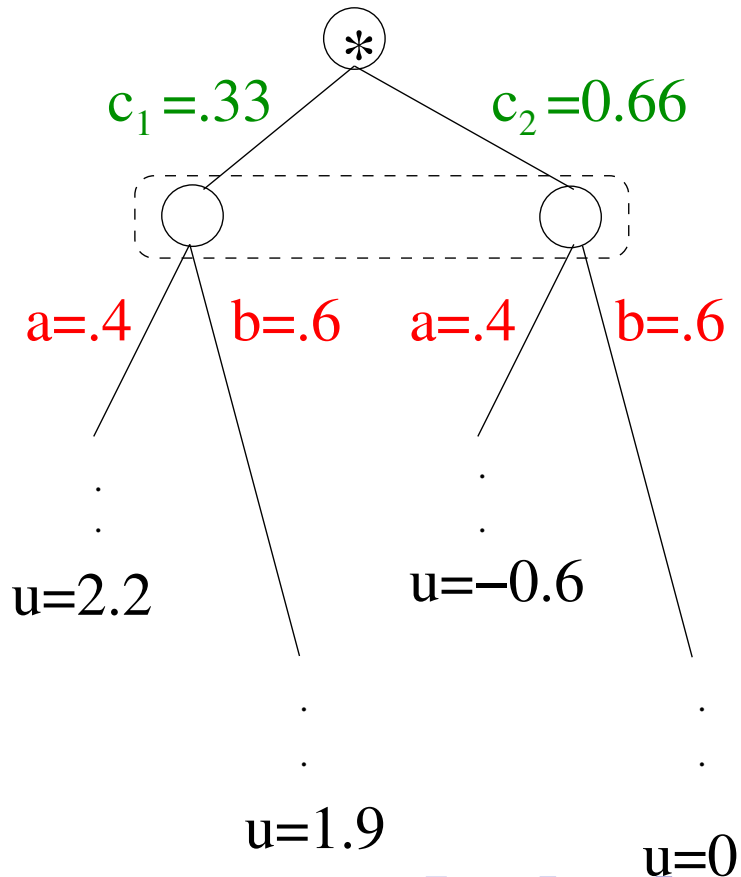
Define **counterfactual value** as

$$v_i(\sigma, I) = \sum_{h \in I, z \in Z, h \sqsubset z} \pi^{\sigma}_{-i}(h) \pi^{\sigma}(h, z) u_i(z)$$

Define $v_i(\sigma_{(I \to a)}, I)$ similarly, except take a at I

Repeat until sufficiently small ϵ:

1. Walk the game tree computing $r(I, a) = v_i(\sigma_{(I \to a)}, I) - v_i(\sigma, I)$



CFR Algorithm II (Example)

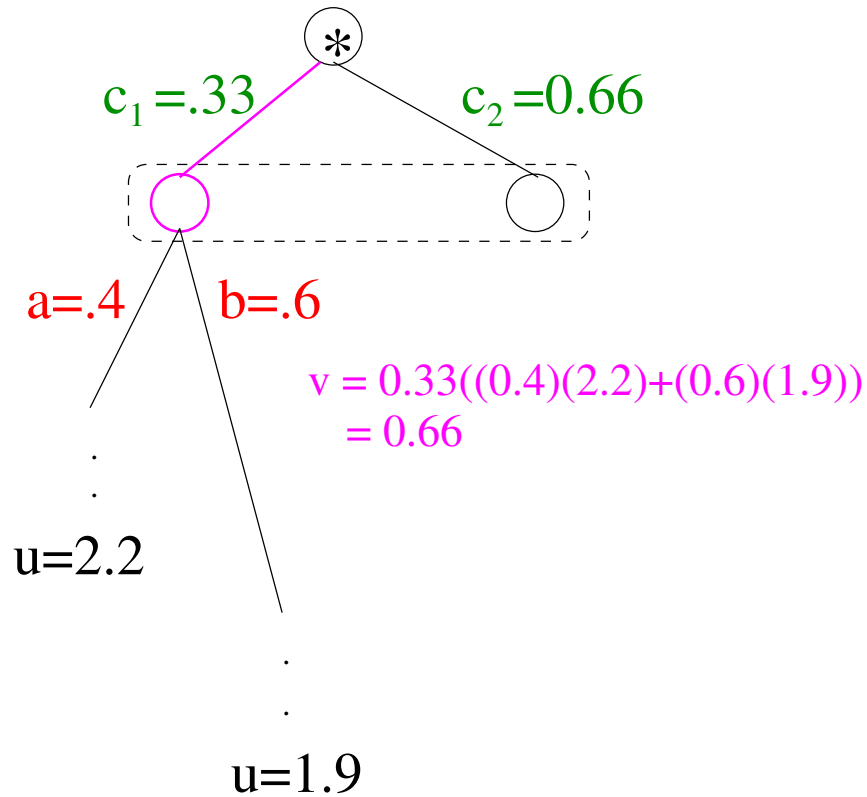
Define **counterfactual value** as

$$v_i(\sigma, I) = \sum_{h \in I, z \in Z, h \sqsubset z} \pi^{\sigma}_{-i}(h) \pi^{\sigma}(h, z) u_i(z)$$

Define $v_i(\sigma_{(I \to a)}, I)$ similarly, except take a at I

Repeat until sufficiently small ϵ:

1. Walk the game tree computing $r(I, a) = v_i(\sigma_{(I \to a)}, I) - v_i(\sigma, I)$
   - 1.1 Recursively compute $r(I, a)$ at a particular node
   - 1.2 Add to accumulated values $r[I, a]$ += $r(I, a)$



CFR Algorithm II (Example)

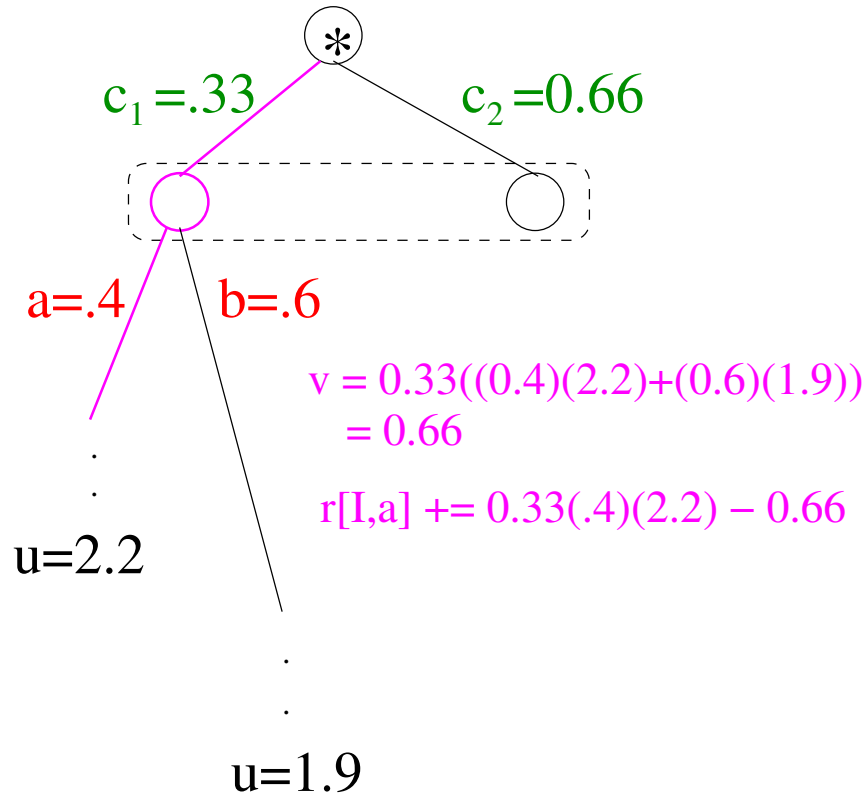
Define **counterfactual value** as

$$v_i(\sigma, I) = \sum_{h \in I, z \in Z, h \sqsubset z} \pi^{\sigma}_{-i}(h)\pi^{\sigma}(h, z)u_i(z)$$

Define $v_i(\sigma_{(I \to a)}, I)$ similarly, except take a at I

Repeat until sufficiently small ϵ:

1. Walk the game tree computing $r(I, a) = v_i(\sigma_{(I \to a)}, I) - v_i(\sigma, I)$
   - 1.1 Recursively compute $r(I, a)$ at a particular node
   - 1.2 Add to accumulated values $r[I, a]$ += $r(I, a)$



CFR Algorithm II (Example)

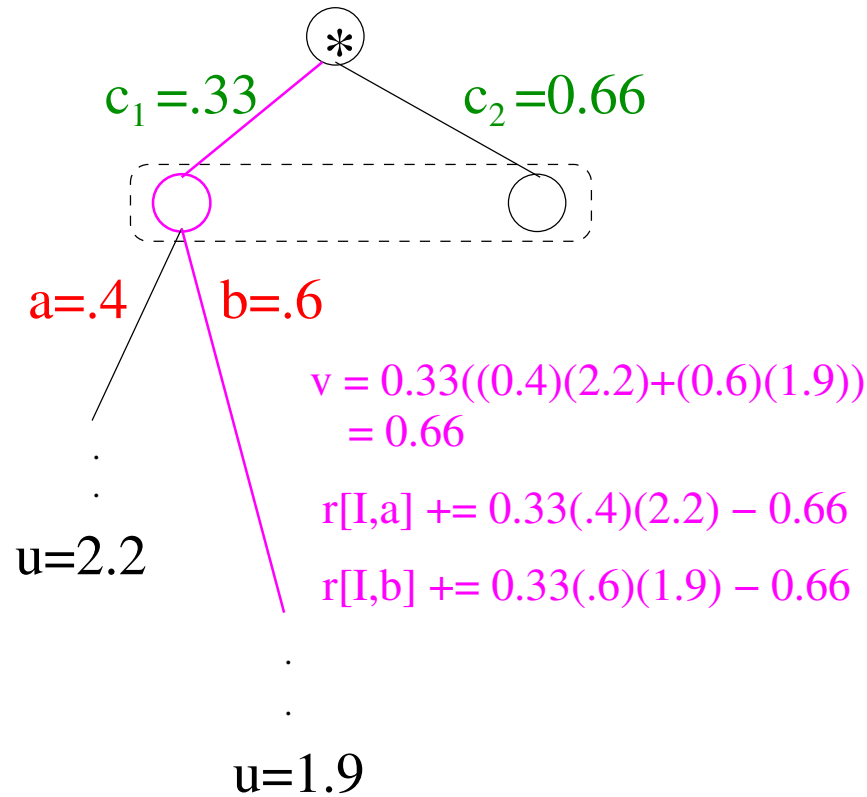
Define **counterfactual value** as

$$v_i(\sigma, I) = \sum_{h \in I, z \in Z, h \sqsubset z} \pi^{\sigma}_{-i}(h)\pi^{\sigma}(h, z)u_i(z)$$

Define $v_i(\sigma_{(I \to a)}, I)$ similarly, except take a at I

Repeat until sufficiently small ϵ:

1. Walk the game tree computing $r(I, a) = v_i(\sigma_{(I \to a)}, I) - v_i(\sigma, I)$
   - 1.1 Recursively compute $r(I, a)$ at a particular node
   - 1.2 Add to accumulated values $r[I, a]$ += $r(I, a)$



CFR Algorithm II (Example)

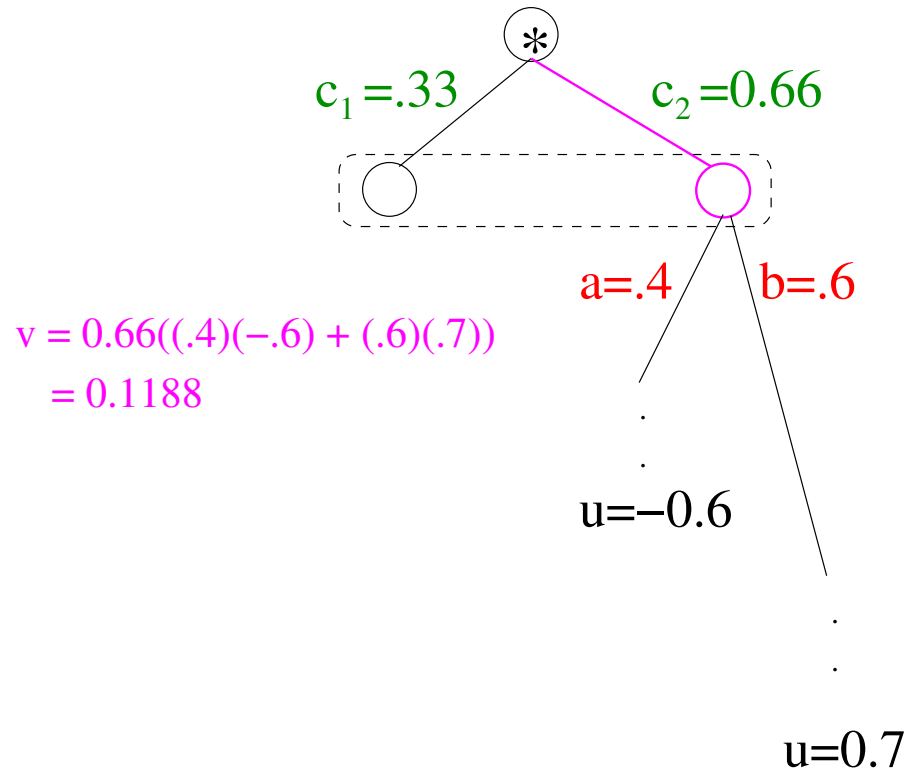
Define **counterfactual value** as

$$v_i(\sigma, I) = \sum_{h \in I, z \in Z, h \sqsubset z} \pi^{\sigma}_{-i}(h) \pi^{\sigma}(h, z) u_i(z)$$

Define $v_i(\sigma_{(I \to a)}, I)$ similarly, except take a at I

Repeat until sufficiently small ϵ:

1. Walk the game tree computing $r(I, a) = v_i(\sigma_{(I \to a)}, I) - v_i(\sigma, I)$
   - 1.1 Recursively compute $r(I, a)$ at a particular node
   - 1.2 Add to accumulated values $r[I, a]$ += $r(I, a)$



CFR Algorithm II (Example)

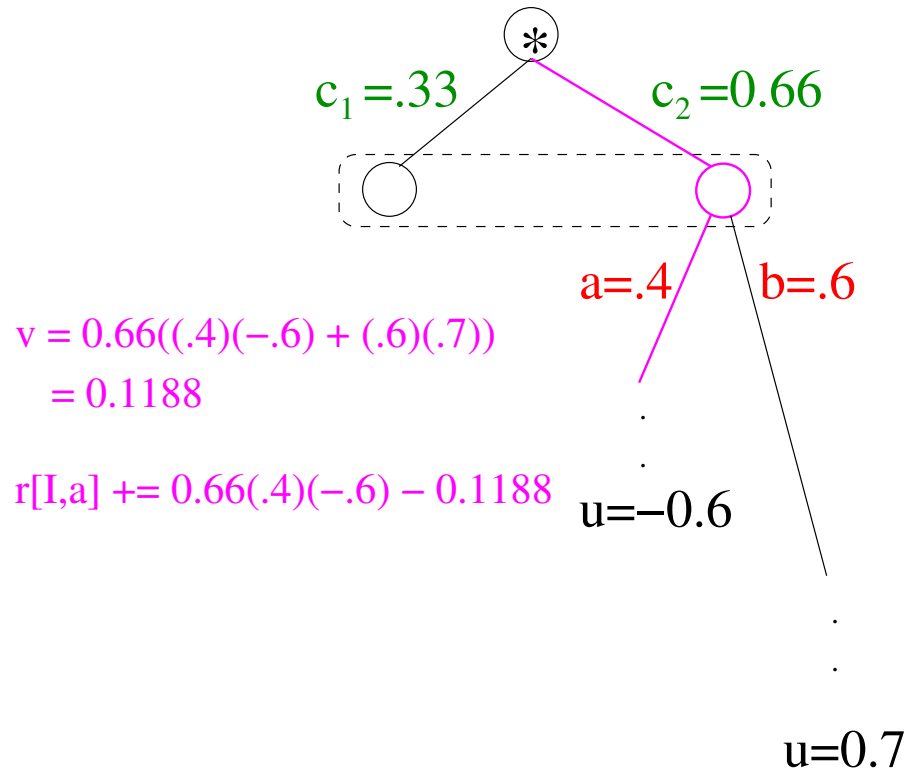
Define **counterfactual value** as

$$v_i(\sigma, I) = \sum_{h \in I, z \in Z, h \sqsubset z} \pi^{\sigma}_{-i}(h)\pi^{\sigma}(h, z)u_i(z)$$

Define $v_i(\sigma_{(I \to a)}, I)$ similarly, except take a at I

Repeat until sufficiently small ϵ:

1. Walk the game tree computing $r(I, a) = v_i(\sigma_{(I \to a)}, I) - v_i(\sigma, I)$
   - 1.1 Recursively compute $r(I, a)$ at a particular node
   - 1.2 Add to accumulated values $r[I, a]$ += $r(I, a)$



CFR Algorithm II (Example)

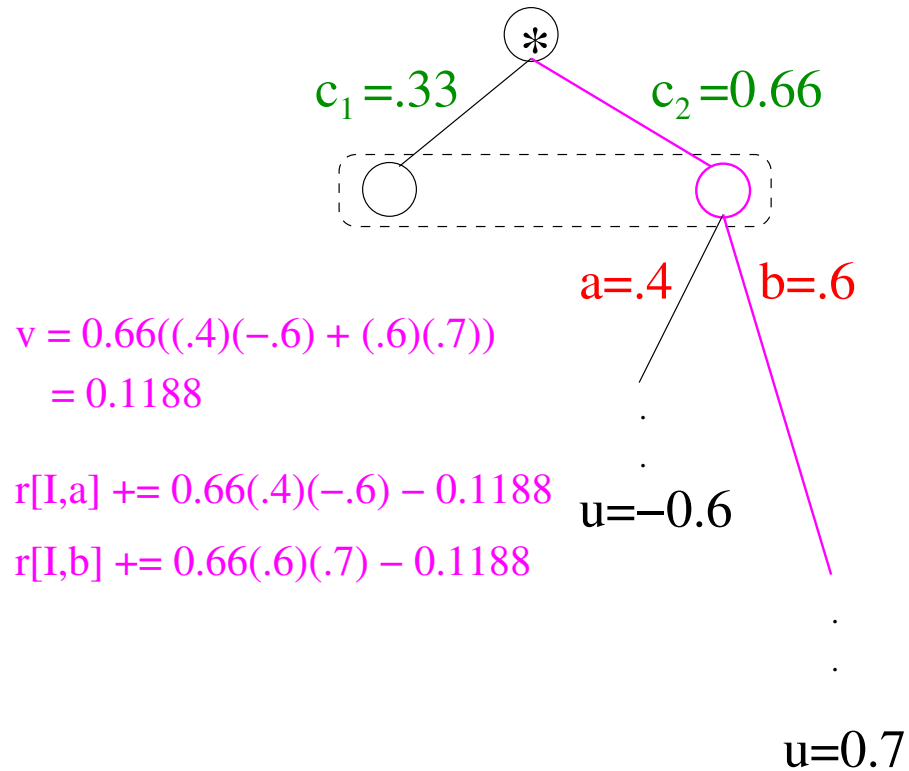
Define **counterfactual value** as

$$v_i(\sigma, I) = \sum_{h \in I, z \in Z, h \sqsubset z} \pi^{\sigma}_{-i}(h)\pi^{\sigma}(h, z)u_i(z)$$

Define $v_i(\sigma_{(I \to a)}, I)$ similarly, except take a at I

Repeat until sufficiently small ϵ:

1. Walk the game tree computing $r(I, a) = v_i(\sigma_{(I \to a)}, I) - v_i(\sigma, I)$
   - 1.1 Recursively compute $r(I, a)$ at a particular node
   - 1.2 Add to accumulated values $r[I, a]$ += $r(I, a)$



CFR Algorithm II (Example)

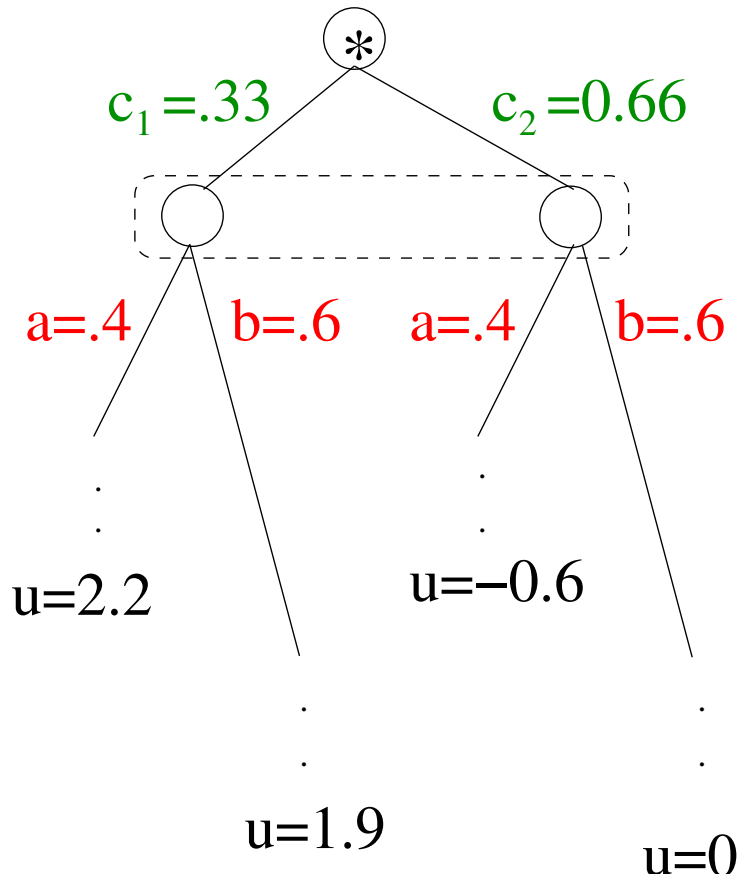
Define **counterfactual value** as

$$v_i(\sigma, I) = \sum_{h \in I, z \in Z, h \sqsubset z} \pi^{\sigma}_{-i}(h) \pi^{\sigma}(h, z) u_i(z)$$

Define $v_i(\sigma_{(I \to a)}, I)$ similarly, except take a at I

Repeat until sufficiently small ϵ:

1. Walk the game tree computing $r(I, a) = v_i(\sigma_{(I \to a)}, I) - v_i(\sigma, I)$
   - 1.1 Recursively compute $r(I, a)$ at a particular node
   - 1.2 Add to accumulated values $r[I, a]$ += $r(I, a)$
2. $\sigma_i^{t+1}(I) \leftarrow$ Blackwell$(r[I])$



CFR Algorithm II (Example)

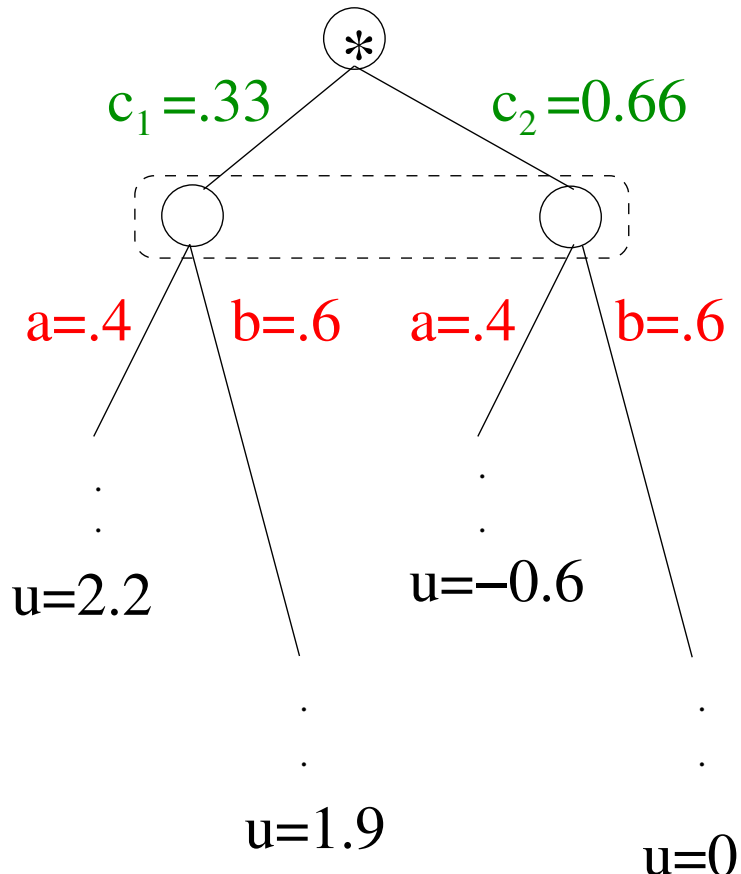
Define **counterfactual value** as

$$v_i(\sigma, I) = \sum_{h \in I, z \in Z, h \sqsubset z} \pi^{\sigma}_{-i}(h) \pi^{\sigma}(h, z) u_i(z)$$

Define $v_i(\sigma_{(I \to a)}, I)$ similarly, except take a at I

Repeat until sufficiently small ϵ:

1. Walk the game tree computing $r(I, a) = v_i(\sigma_{(I \to a)}, I) - v_i(\sigma, I)$
   - 1.1 Recursively compute $r(I, a)$ at a particular node
   - 1.2 Add to accumulated values $r[I, a]$ += $r(I, a)$
2. $\sigma_i^{t+1}(I) \leftarrow$ Blackwell$(r[I])$
3. Update average profile $\bar{\sigma}$



Vanilla CFR Properties

The Good:

- CFR requires space $O(|\mathcal{I}|)$.
- Solved games with up to 10^{12} states.
- Poker bots defeated human experts.

Vanilla CFR Properties

The Good:

- CFR requires space $O(|\mathcal{I}|)$.
- Solved games with up to 10^{12} states.
- Poker bots defeated human experts.

The Not-So-Good:

- CFR not suited for online use.

Vanilla CFR Properties

The Good:

- CFR requires space $O(|\mathcal{I}|)$.
- Solved games with up to 10^{12} states.
- Poker bots defeated human experts.

The Not-So-Good:

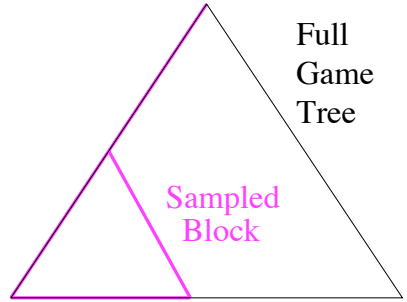
- CFR not suited for online use.

The Even-Less-Than-Not-So-Good:

- Iterations require a full tree traversal.
- (Too costly in some large games.)

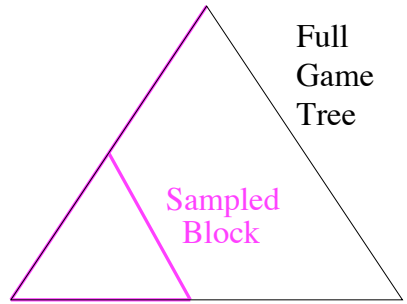
Solution: Sampling vs. Full Traversals

Define $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_r\}$. Here, $Q_j \subseteq Z$ is a **block** of terminal histories, and the $\mathcal{Q}$ spans Z.



Solution: Sampling vs. Full Traversals

Define $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_r\}$. Here, $Q_j \subseteq Z$ is a **block** of terminal histories, and the $\mathcal{Q}$ spans Z.



Define **sampled counterfactual value** to be:

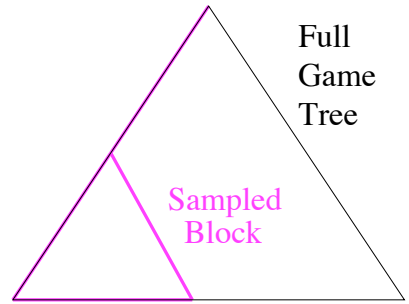
$$\tilde{v}_i(\sigma, I|j) = \sum_{h \in I, z \in Q_j \cap Z_I, h \sqsubset z} \frac{1}{q(z)} \pi^{\sigma}_{-i}(h) \pi^{\sigma}(h, z) u_i(z)$$

where:

- Z_I is the set of all terminal histories that pass through I.
- $q(z)$ is the probability of sampling terminal history z.

Solution: Sampling vs. Full Traversals

Define $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_r\}$. Here, $Q_j \subseteq Z$ is a **block** of terminal histories, and the $\mathcal{Q}$ spans Z.



Define **sampled counterfactual value** to be:

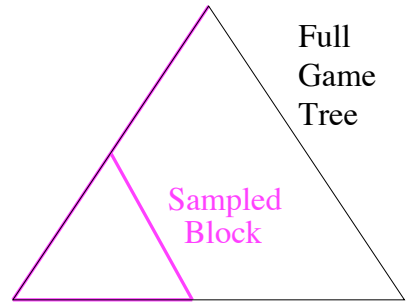
$$\tilde{v}_i(\sigma, I|j) = \sum_{h \in I, \mathbf{z} \in \mathbf{Q_j} \cap \mathbf{Z_I}, h \sqsubset z} \frac{\mathbf{1}}{\mathbf{q(z)}} \pi^{\sigma}_{-i}(h) \pi^{\sigma}(h, z) u_i(z)$$

where:

- Z_I is the set of all terminal histories that pass through I.
- $q(z)$ is the probability of sampling terminal history z.

Solution: Sampling vs. Full Traversals

Define $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_r\}$. Here, $Q_j \subseteq Z$ is a **block** of terminal histories, and the $\mathcal{Q}$ spans Z.



Define **sampled counterfactual value** to be:

$$\tilde{v}_i(\sigma, I|j) = \sum_{h \in I, \mathbf{z} \in \mathbf{Q_j} \cap \mathbf{Z_I}, h \sqsubset z} \frac{\mathbf{1}}{\mathbf{q(z)}} \pi^{\sigma}_{-i}(h) \pi^{\sigma}(h, z) u_i(z)$$

where:

- Z_I is the set of all terminal histories that pass through I.
- $q(z)$ is the probability of sampling terminal history z.

$$E_{j \sim q_j}\left[\tilde{v}_i(\sigma, I|j)\right] = v_i(\sigma, I)$$

$\Rightarrow$We can perform the "same" updates *in expectation*!

Sampled vs. Unsampled Counterfactual Value

$$
\begin{aligned}
& E_{j\sim q_j}\left[\tilde{v}_i(\sigma, I|j)\right] && (1)\\
&= \sum_j q_j \tilde{v}_i(\sigma, I|j) && (2)\\
&= \sum_j \sum_{z\in Q_j\cap Z_I} \frac{q_j}{q(z)} u_i(z)\pi^{\sigma}_{-i}(z[I])\pi^{\sigma}(z[I], z) && (3)\\
&= \sum_{z\in Z_I} \frac{\sum_{j:z\in Q_j} q_j}{q(z)} u_i(z)\pi^{\sigma}_{-i}(z[I])\pi^{\sigma}(z[I], z) && (4)\\
&= \sum_{z\in Z_I} u_i(z)\pi^{\sigma}_{-i}(z[I])\pi^{\sigma}(z[I], z) && (5)\\
&= \sum_{z\in Z} \sum_{h\in I} u_i(z)\pi^{\sigma}_{-i}(h)\pi^{\sigma}(h, z) && (6)\\
&= v_i(\sigma, I) && (7)
\end{aligned}
$$

MCCFR Algorithm

Monte Carlo CFR applies corrected updates to a *sampled block* Q_j rather than performing a full traversal.

Repeat:

1. Sample a block $Q_j \subseteq Z$ using some sampling scheme.
2. Perform analogous regret updates on subset of prefix histories in Q_j using $\tilde{v}$ to calculate **sampled immediate counterfactual regret** $\tilde{R}^T_{i,imm}$.
3. Use regret matching and Blackwell's to give new strategies σ^{t+1}.
4. Update the average profile given new σ^{t+1}.

MCCFR: Five Questions

Q1. What do we choose as a “sampling scheme” ?

Q2. Does MCCFR converge to ϵ-Nash ?

Q3. If so, is it true for any sampling scheme ?

Q4. What is the rate of convergence ?

Q5. How well does MCCFR perform empirically ?

MCCFR: Five Answers

Q1. What do we choose as a “sampling scheme” ?
A1. We will define two in particular.

Q2. Does MCCFR converge to ϵ-Nash ?
A2. Yes!

Q3. If so, is it true for any sampling scheme ?
A3. Yes, for “reasonable” sampling schemes.

Q4. What is the rate of convergence ?
A4. It depends on the sampling scheme.

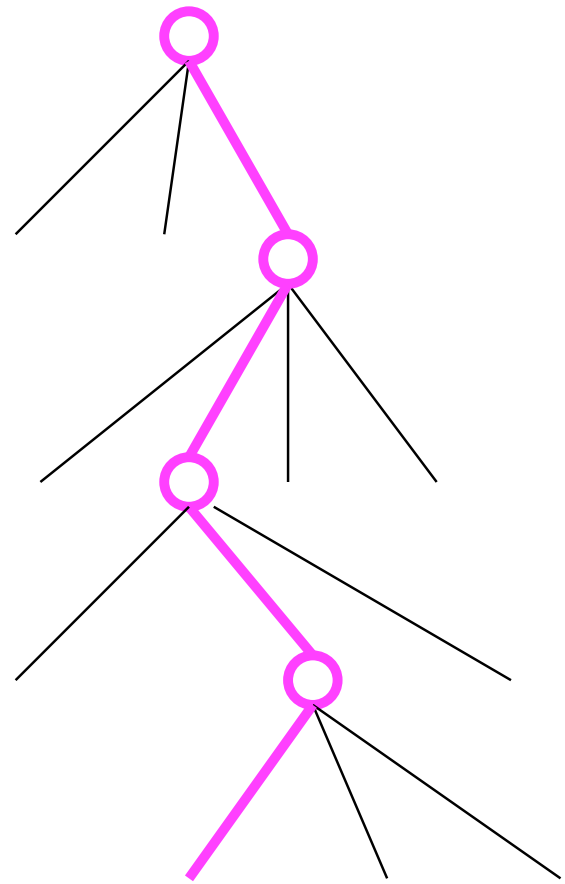
Q5. How well does MCCFR perform empirically ?
A5. On four selected games, great!

Outcome Sampling I

Choose blocks Q_j such that they contain a *single* terminal history, $|Q_j| = 1$.

Outcome Sampling I

Choose blocks Q_j such that they contain a *single* terminal history, $|Q_j| = 1$.



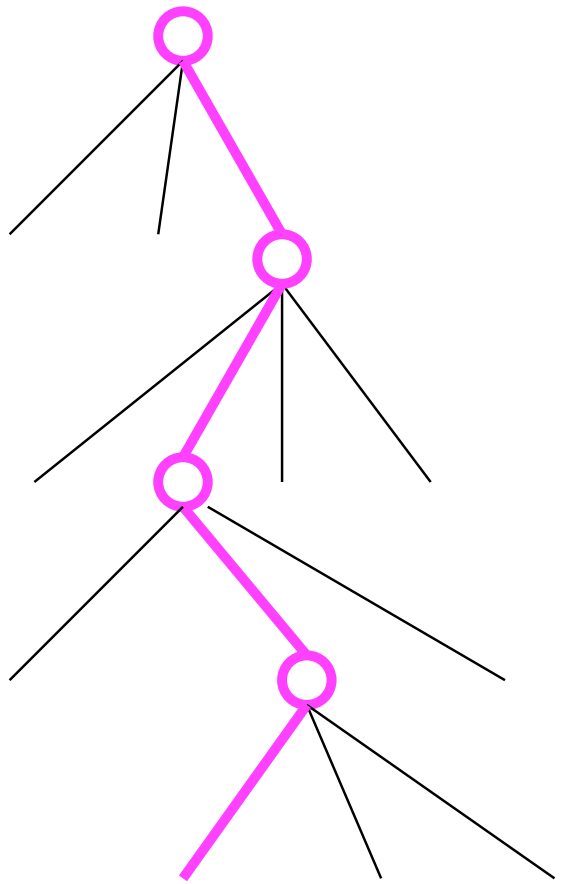
Outcome Sampling I

Choose blocks Q_j such that they contain a *single* terminal history, $|Q_j| = 1$.

Sample according to a **sampling profile** σ' so that $q(z) = \pi^{\sigma'}(z)$. Using an ϵ-greedy profile, with $\epsilon > 0$ then $\exists \delta > 0$ such that $q(z) > \delta$ which ensures that the $\tilde{v}$ is still well-formed.

$P(\text{sample } a \in A(I)) \text{ is}$

$$\begin{cases} \text{randomly} & = \epsilon \\ \text{using } \sigma^t & = 1 - \epsilon \end{cases}$$



Outcome Sampling I

Choose blocks Q_j such that they contain a *single* terminal history, $|Q_j| = 1$.

Sample according to a **sampling profile** σ' so that $q(z) = \pi^{\sigma'}(z)$. Using an ϵ-greedy profile, with $\epsilon > 0$ then $\exists \delta > 0$ such that $q(z) > \delta$ which ensures that the $\tilde{v}$ is still well-formed.

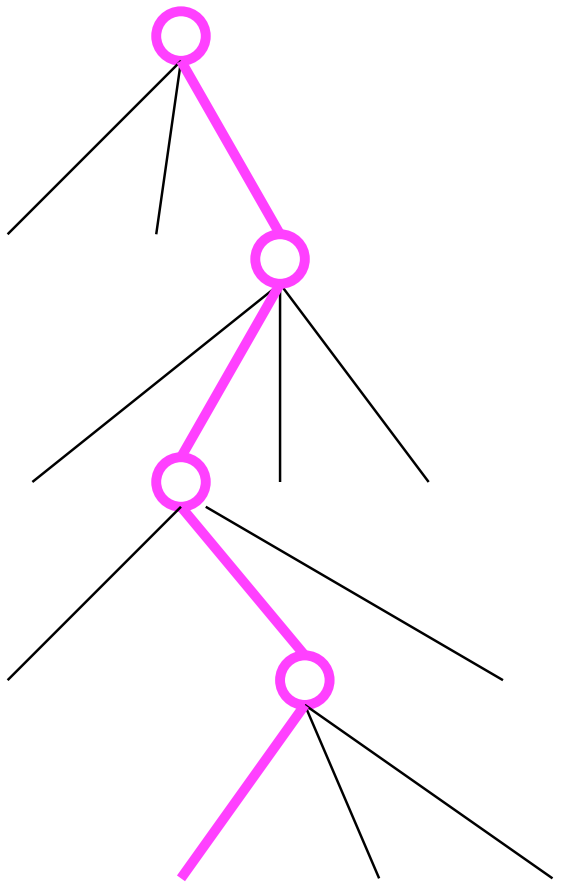
Now define $\tilde{r}(I, a) = \tilde{v}_i(\sigma^t_{(I \to a)}, I) - \tilde{v}_i(\sigma^t, I)$.

$$\tilde{r}(I,a) = \begin{cases} w_I \cdot \big(1 - \sigma(a|z[I])\big) & \text{if } (z[I]a) \sqsubseteq z \\ -w_I \cdot \sigma(a|z[I]) & \text{otherwise} \end{cases}$$

$$w_I = \frac{u_i(z)\pi^{\sigma}_{-i}(z)\pi^{\sigma}_i(z[I], z)}{\pi^{\sigma'}(z)}$$

$P(\text{sample } a \in A(I)) \text{ is}$

$$\begin{cases} \text{randomly} & = \epsilon \\ \text{using } \sigma^t & = 1 - \epsilon \end{cases}$$



Outcome Sampling II

If we sample according to our opponent's strategy: $\sigma'_{-i} = \sigma_{-i}$ then σ_{-i} cancels in the numerator and we have

$$w_I = \frac{u_i(z)\pi_i^{\sigma}(z[I], z)}{\pi_i^{\sigma'}(z)}$$

Outcome Sampling II

If we sample according to our opponent's strategy: $\sigma'_{-i} = \sigma_{-i}$ then σ_{-i} cancels in the numerator and we have

$$w_I = \frac{u_i(z)\pi_i^{\sigma}(z[I], z)}{\pi_i^{\sigma'}(z)}$$

$\Rightarrow$ The regret update no longer depends on the opponent!

Outcome Sampling II

If we sample according to our opponent's strategy: $\sigma'_{-i} = \sigma_{-i}$ then σ_{-i} cancels in the numerator and we have

$$w_I = \frac{u_i(z)\pi_i^{\sigma}(z[I], z)}{\pi_i^{\sigma'}(z)}$$

$\Rightarrow$ The regret update no longer depends on the opponent!

Therefore we can use outcome-sampling MCCFR for *general regret minimization* in the same way that was done for normal form games in EXP3 [Auer et. al., 1995].

If we know the number of playings T is known in advance, we can bound the average overall regret.

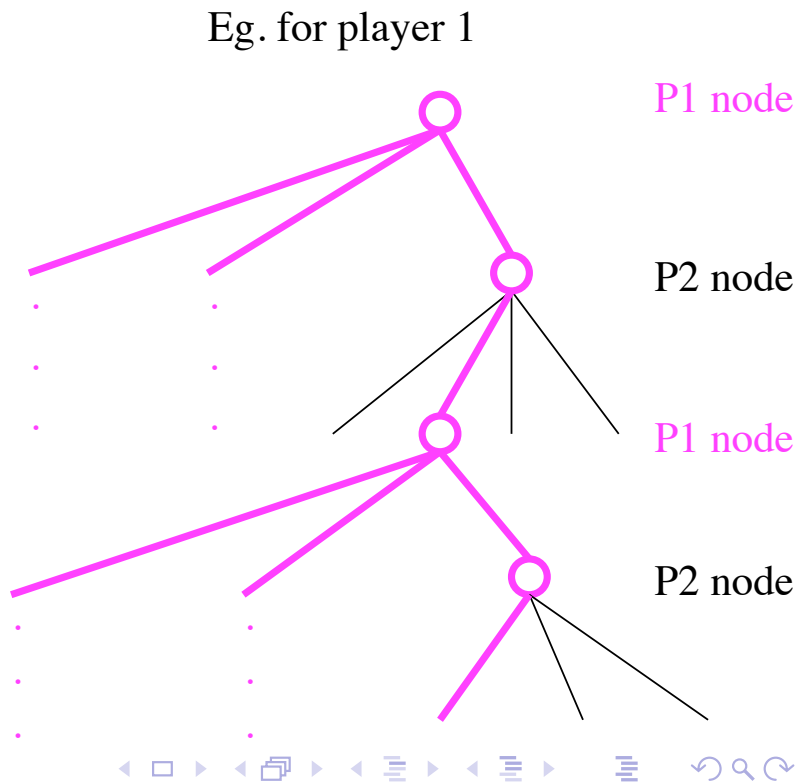
External Sampling

Only sample the actions of the opponent and chance.

External Sampling

Only sample the actions of the opponent and chance.

So $Q_\tau \in \mathcal{Q}$ corresponds to the subsets of Z reachable by a particular deterministic strategy τ that is sampled according to σ^t_{-i}.



External Sampling

Only sample the actions of the opponent and chance.

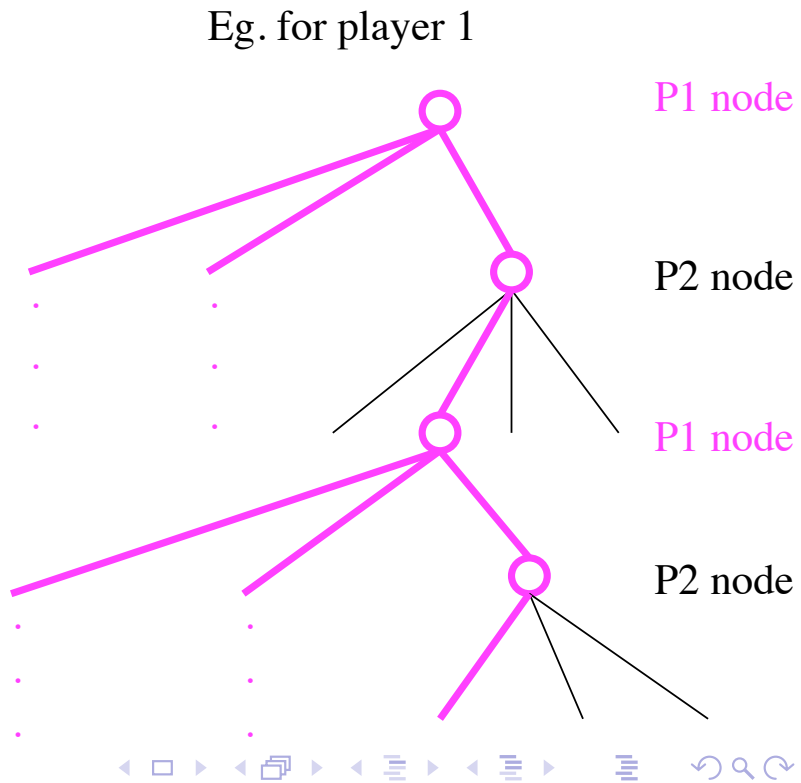
So $Q_\tau \in \mathcal{Q}$ corresponds to the subsets of Z reachable by a particular deterministic strategy τ that is sampled according to σ^t_{-i}.

Block probabilities result in

$$q(z) = \pi^\sigma_{-i}(z)$$

and regret updates end up being:

$$\tilde{r}(I,a) = (1 - \sigma(a|I)) \sum_{z \in Q \cap Z_I} u_i(z) \pi^\sigma_i(z[I], z)$$



Theoretical Results

- When using *vanilla CFR* for player i,

$$R_i^T \leq \Delta_{u,i} M_i \sqrt{|A_i|} / \sqrt{T}$$

where M_i is a game-dependent value, $\sqrt{|\mathcal{I}_i|} \leq M_i \leq |\mathcal{I}_i|$, and there exist games that satisfy both extremities.

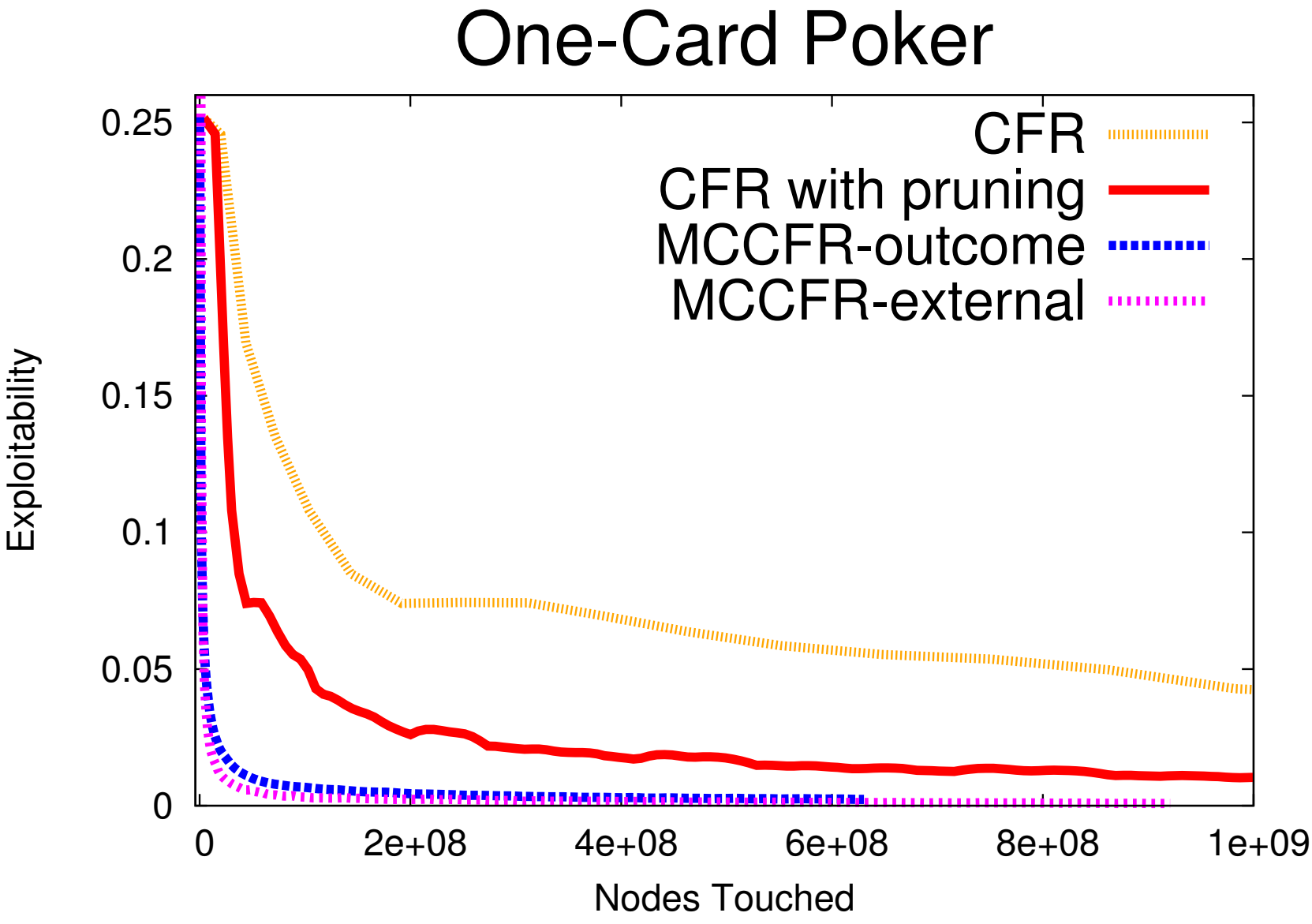
- For any $p \in (0, 1]$, when using *external-sampling MCCFR*, with probability at least $1 - p$,

$$R_i^T \leq \left(1 + \frac{2}{\sqrt{p}}\right) \Delta_{u,i} M_i \sqrt{|A_i|} / \sqrt{T}$$

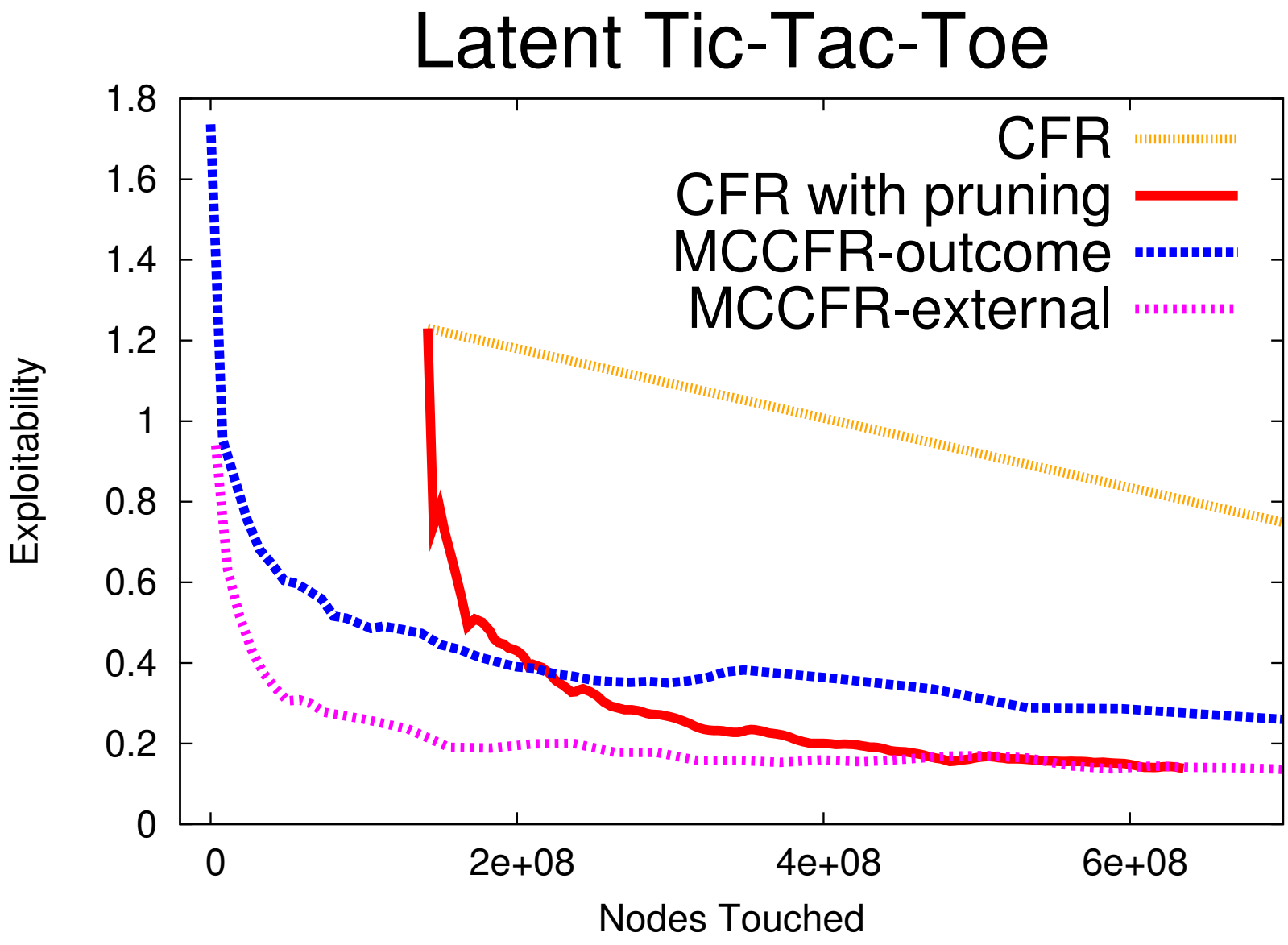
- For any $p \in (0, 1]$, when using *outcome-sampling MCCFR* where $\forall z \in Z$ either $\pi_{-i}^{\sigma}(z) = 0$ or $q(z) \geq \delta > 0$ at every timestep, with probability $1 - p$,

$$R_i^T \leq \left(1 + \frac{2}{\sqrt{p}}\right) \left(\frac{1}{\delta}\right) \Delta_{u,i} M_i \sqrt{|A_i|} / \sqrt{T}$$

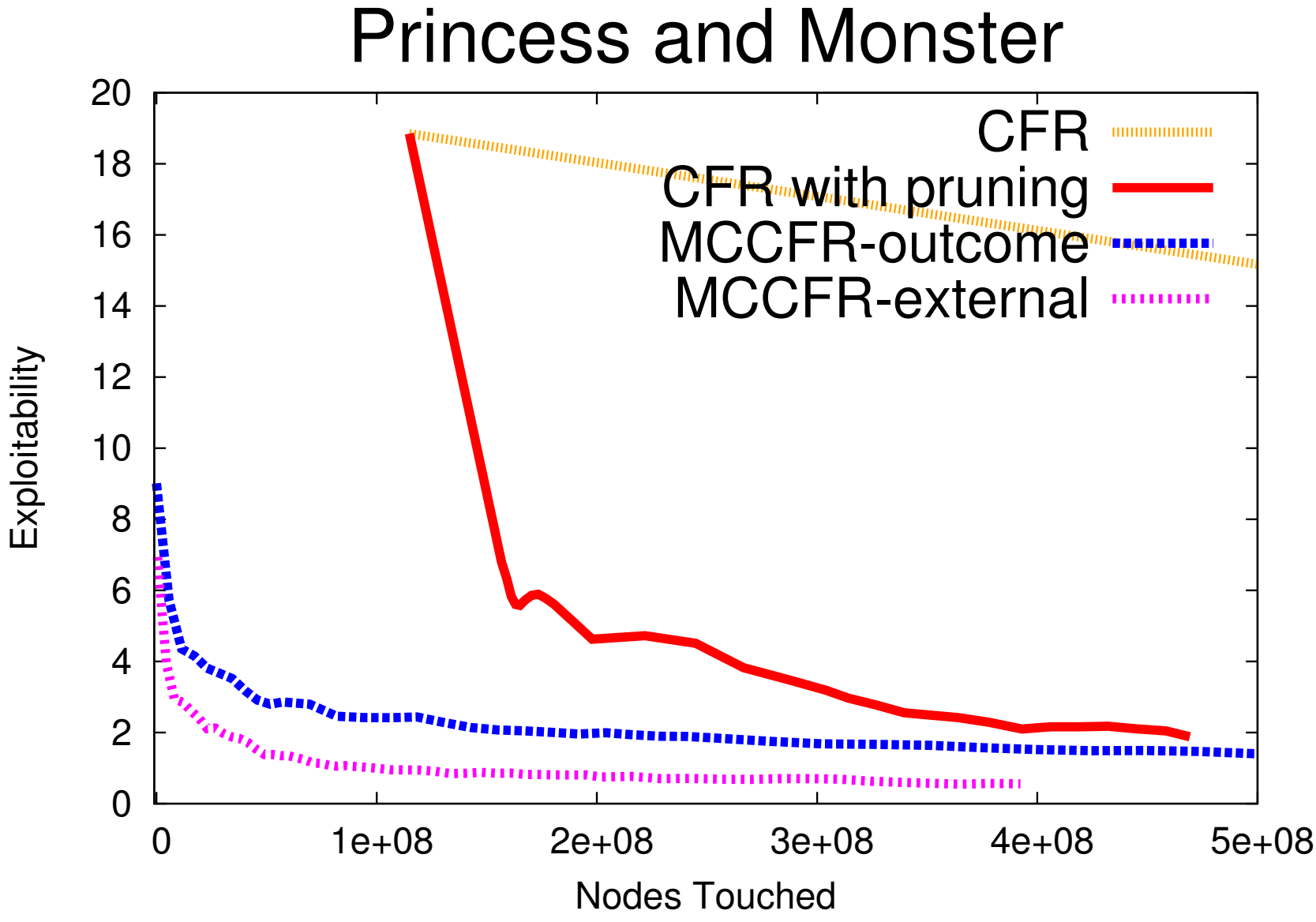
Empirical Results I



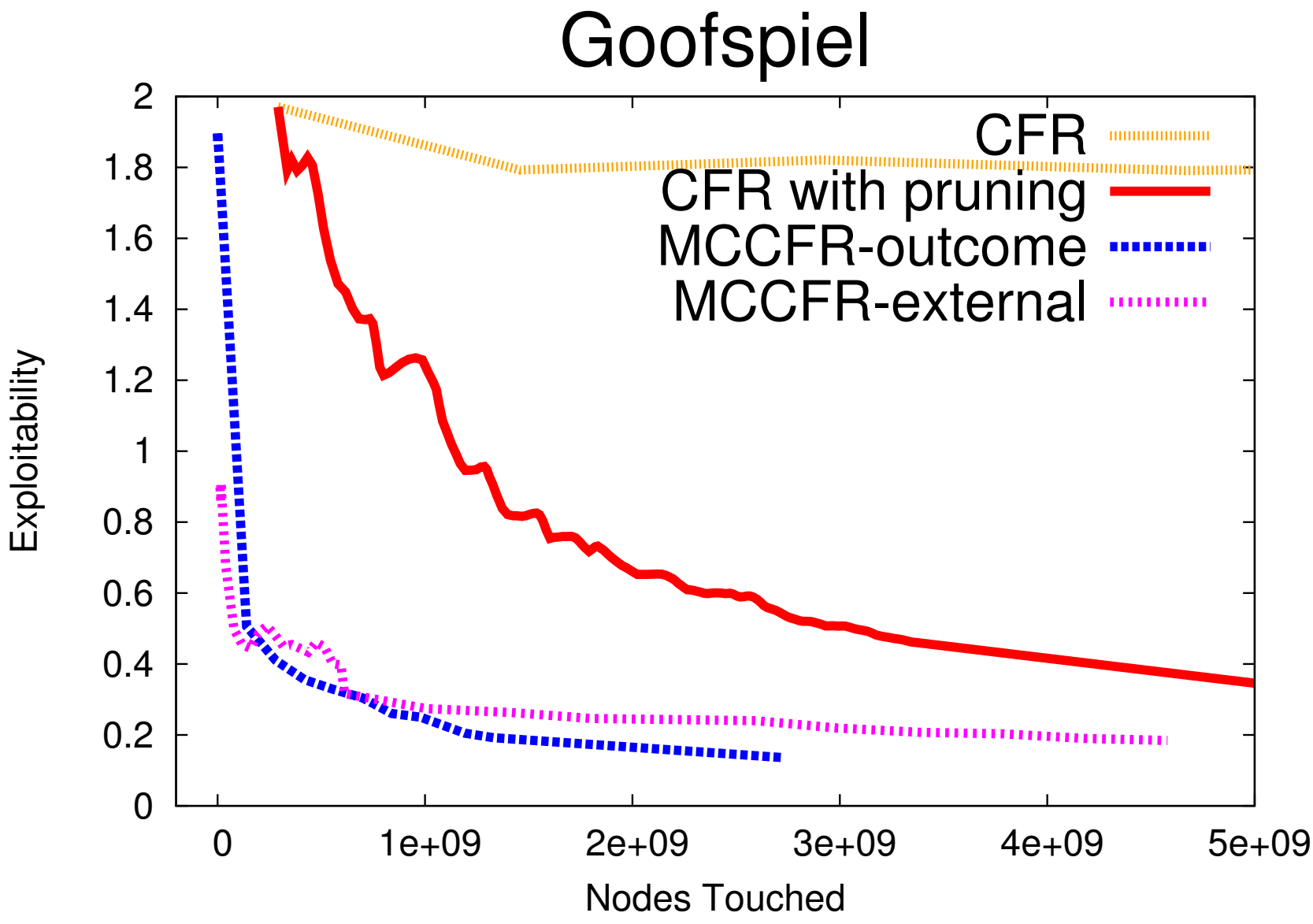
Empirical Results II



Empirical Results III



Empirical Results IV



Conclusion + Future Work

Bottom line: The less computation time per iteration is worth-while even though more iterations are needed for convergence.

Future work:

1. Try outcome-sampling in general regret minimization.
2. Analyze the selection of ϵ in the outcome-sampling profile σ'.
3. Perform updates on abstract states using a subset of the sequence to describe the information sets in outcome-sampling (eg. lose the perfect recall assumption)