

MS&E 233

Game Theory, Data Science and AI

Lecture 6

Vasilis Syrgkanis

Assistant Professor

Management Science and Engineering

(by courtesy) Computer Science and Electrical Engineering

Institute for Computational and Mathematical Engineering

Computational Game Theory for Complex Games

- 1
 - Basics of game theory and zero-sum games (T)
 - Basics of online learning theory (T)
 - Solving zero-sum games via online learning (T)
 - *HW1: implement simple algorithms to solve zero-sum games*
 - Applications to ML and AI (T+A)
 - *HW2: implement boosting as solving a zero-sum game*

- 2
 - Basics of extensive-form games
 - **Solving extensive-form games via online learning (T)**
 - *HW3: implement agents to solve very simple variants of poker*

- 3
 - General games and equilibria (T)
 - Online learning in general games, multi-agent RL (T+A)
 - *HW4: implement no-regret algorithms that converge to correlated equilibria in general games*

Data Science for Auctions and Mechanisms

- 4
 - Basics and applications of auction theory (T+A)
 - **Learning to bid in auctions via online learning (T)**
 - *HW5: implement bandit algorithms to bid in ad auctions*

- 5
 - Optimal auctions and mechanisms (T)
 - Simple vs optimal mechanisms (T)
 - *HW6: calculate equilibria in simple auctions, implement simple and optimal auctions, analyze revenue empirically*

- 6
 - Optimizing mechanisms from samples (T)
 - Online optimization of auctions and mechanisms (T)
 - *HW7: implement procedures to learn approximately optimal auctions from historical samples and in an online manner*

Further Topics

- 7
 - Econometrics in games and auctions (T+A)
 - A/B testing in markets (T+A)
 - *HW8: implement procedure to estimate values from bids in an auction, empirically analyze inaccuracy of A/B tests in markets*

Guest Lectures

- TBD
- TBD

Recap: Sequence Form Representation

- The strategies of the player can be represented as $\tilde{x} \in X, \tilde{y} \in Y$
- \tilde{x}_a : product of probabilities of all actions of P1 on the path to a
- \tilde{y}_a : product of probabilities of all actions of P2 on the path to a

$$X := \left\{ \forall j \in \mathcal{J}_1: \sum_{a \in A_j} \tilde{x}_a = \tilde{x}_{p_j} \right\}, \quad Y := \left\{ \forall j \in \mathcal{J}_2: \sum_{a \in A_j} \tilde{y}_a = \tilde{y}_{p_j} \right\}$$

- The payoff to P1 under sequence strategies $\tilde{x} \in X, \tilde{y} \in Y$ is

$$\tilde{x}^\top A \tilde{y}$$

- $A_{a,a'} =$ **if** a was the last action of P1 and a' the last action of P2 before some leaf z , **then** payoff to P1 at z times product of chance probabilities on path to z **else** zero

Recap: From Sequence to Behavioral

- Every sequence form strategy \tilde{x} can be transformed into a behavioral form strategy as (recursively bottom up):

$$\forall a \in A_j: x_a = \frac{\tilde{x}_a}{\tilde{x}_{p_j}}$$

if info-set is un-reachable, i.e. $\tilde{x}_{p_j} = 0$, then use any behavioral

- Every behavioral strategy x can be transformed into a sequence form strategy as (recursively top down):

$$\forall a \in A_j: \tilde{x}_a = \tilde{x}_{p_j} \cdot x_a$$

Recap: No-Regret Learning in Sequence Form

- We have successfully turned imperfect information extensive form zero-sum games into a familiar object

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

- X, Y are convex sets, i.e., sequence-form strategies
- We can invoke minimax theorem to prove existence of equilibria
- We can calculate equilibria via LP duality
- We can calculate equilibria via no-regret learning!

Solving Extensive Form Games via No-Regret Learning

Recap from Lecture 2: Regret of FTRL

$$\text{(FTRL)} \quad x_t = \operatorname{argmin}_{x \in X} \underbrace{\sum_{\tau < t} \langle x, \ell_\tau \rangle}_{\substack{\text{Historical performance} \\ \text{of always choosing} \\ \text{strategy } x}} + \underbrace{\frac{1}{\eta} \mathcal{R}(x)}_{\substack{\text{1-strongly convex} \\ \text{function of } x \text{ that} \\ \text{stabilizes the minimizer}}}$$

Theorem. Assuming the loss function at each period
 $f_t(x) = \langle x, \ell_t \rangle$

is L -Lipschitz with respect to some norm $\|\cdot\|$ and the regularizer is 1-strongly convex with respect to the same norm then

$$\text{Regret} - \text{FTRL}(T) \leq \underbrace{\eta L}_{\substack{\text{Average stability} \\ \text{induced by regularizer}}} + \underbrace{\frac{1}{\eta T} \left(\max_{x \in X} \mathcal{R}(x) - \min_{x \in X} \mathcal{R}(x) \right)}_{\substack{\text{Average loss distortion} \\ \text{caused by regularizer}}}$$

Same for utilities

(FTRL)
$$x_t = \operatorname{argmax}_{x \in X} \underbrace{\sum_{\tau < t} \langle x, u_\tau \rangle}_{\substack{\text{Historical performance} \\ \text{of always choosing} \\ \text{strategy } x}} - \underbrace{\frac{1}{\eta} \mathcal{R}(x)}_{\substack{\text{1-strongly convex} \\ \text{function of } x \text{ that} \\ \text{stabilizes the maximizer}}}$$

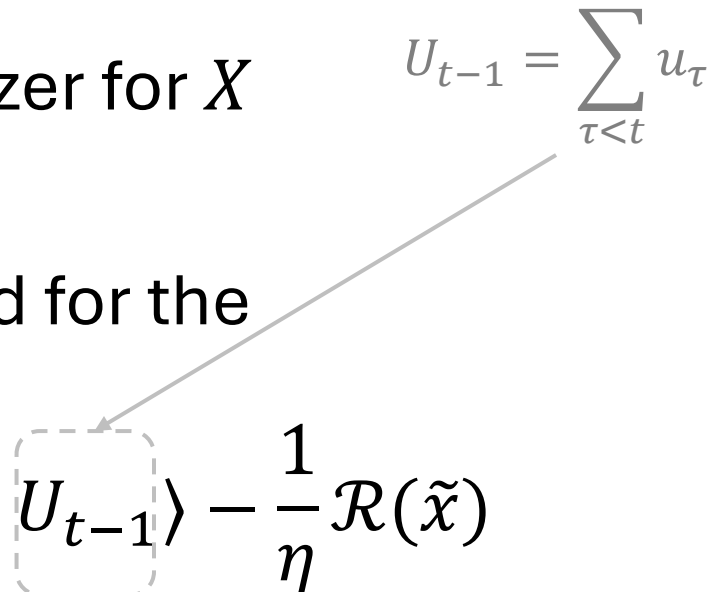
Theorem. Assuming the utility function at each period
 $f_t(x) = \langle x, u_t \rangle$

is L -Lipschitz with respect to some norm $\|\cdot\|$ and the regularizer is 1-strongly convex with respect to the same norm then

$$\text{Regret} - \text{FTRL}(T) \leq \underbrace{\eta L}_{\substack{\text{Average stability} \\ \text{induced by regularizer}}} + \underbrace{\frac{1}{\eta T} \left(\max_{x \in X} \mathcal{R}(x) - \min_{x \in X} \mathcal{R}(x) \right)}_{\substack{\text{Average loss distortion} \\ \text{caused by regularizer}}}$$

Regularizer for the Space X

- The only thing we are missing is a good Regularizer for X
- **Desiderata.** Be strongly convex in x within X and for the optimization problem to be fast to solve

$$\tilde{x}_t = \operatorname{argmax}_{\tilde{x} \in X} \sum_{\tau < t} \langle \tilde{x}, u_\tau \rangle - \frac{1}{\eta} \mathcal{R}(\tilde{x}) = \operatorname{argmax}_{\tilde{x} \in X} \langle \tilde{x}, U_{t-1} \rangle - \frac{1}{\eta} \mathcal{R}(\tilde{x})$$


- X is no longer a “simplex”, so entropy is not a good Regularizer

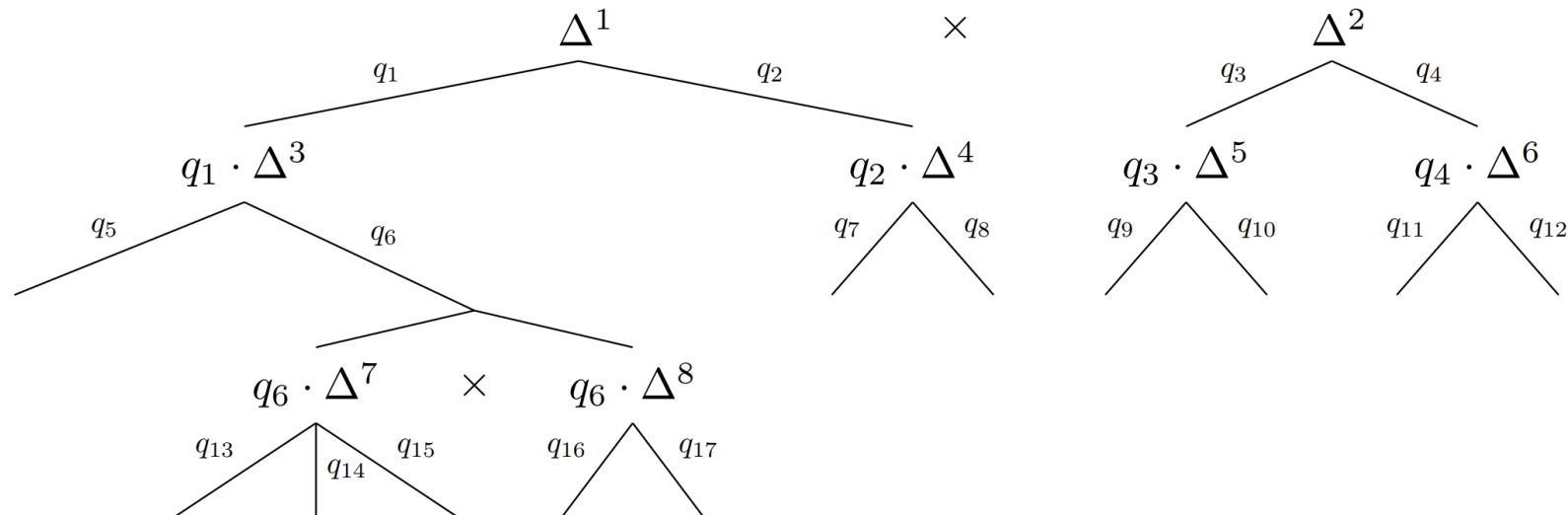
TreePlex Representation of Strategy Space

The strategy space of each player is a set of interconnected “scaled” simplices

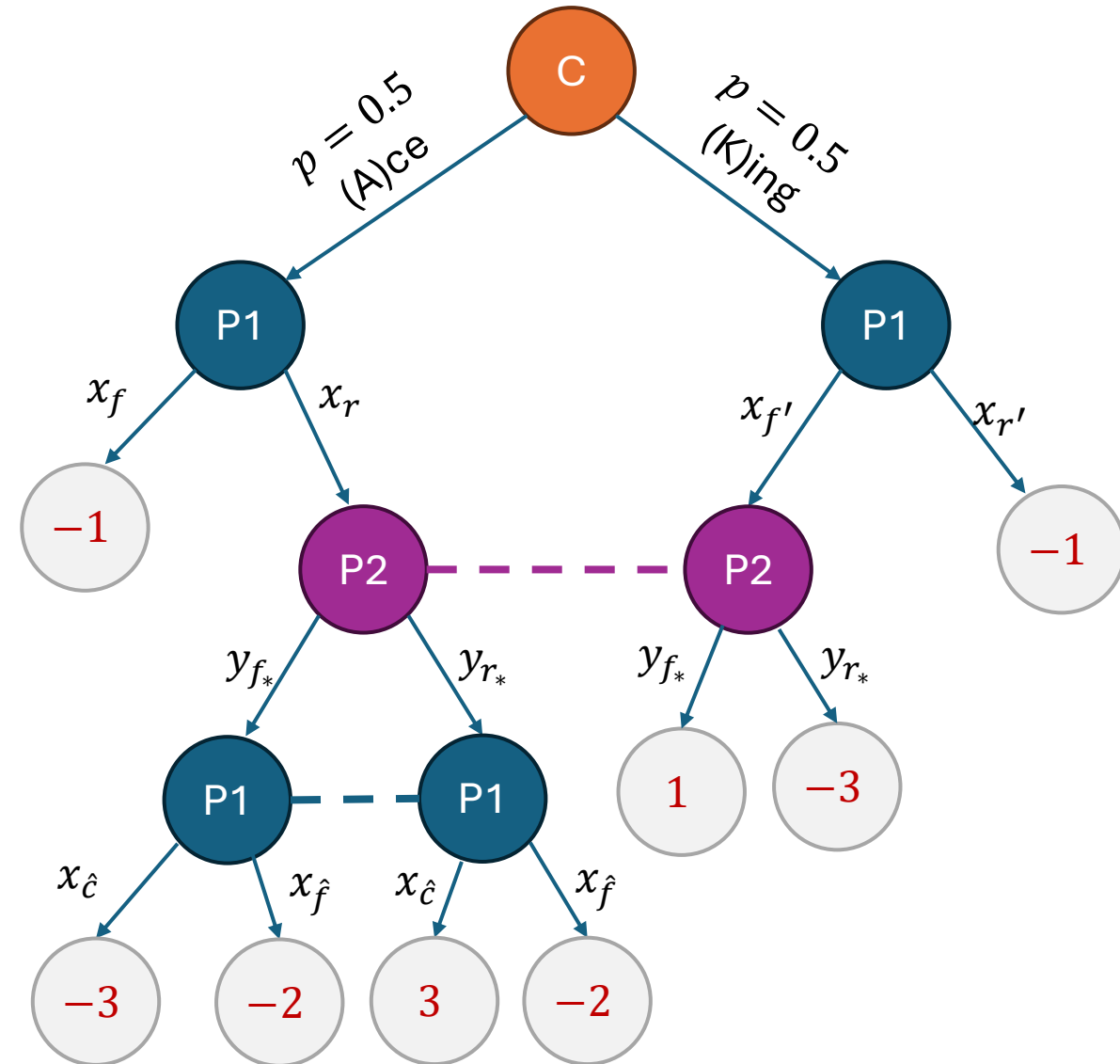
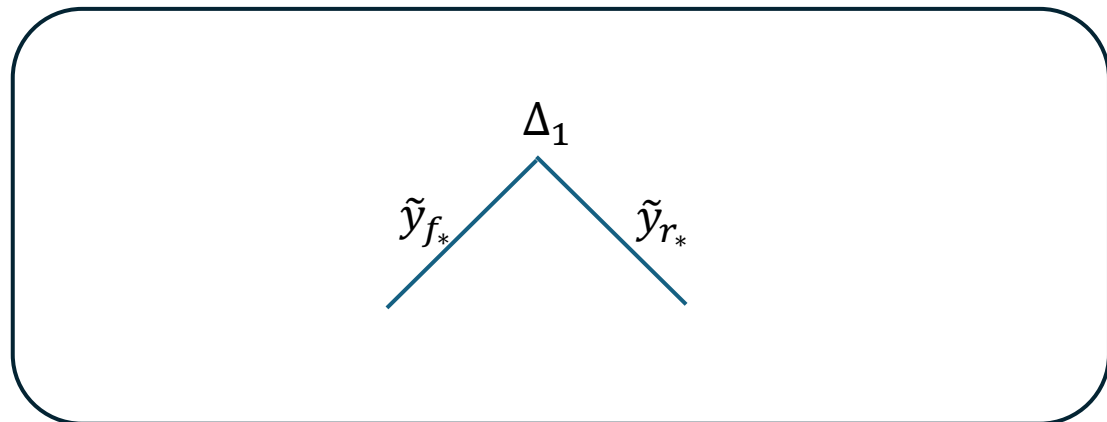
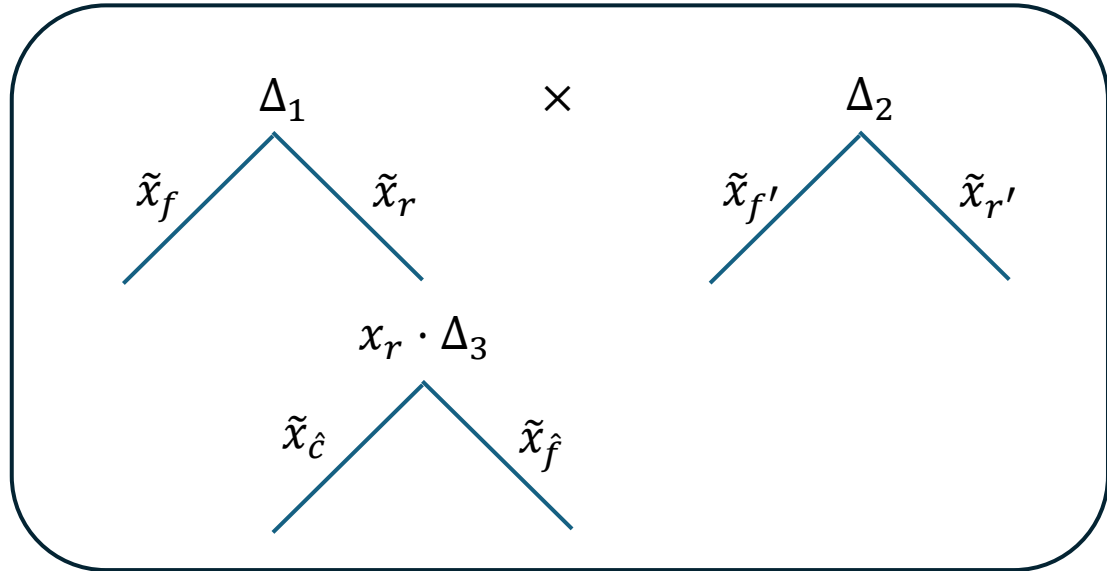
$$\forall j \in J_1: \sum_{a \in A_j} \tilde{x}_a = \tilde{x}_{p_j}$$

To generate \tilde{x}_a

- Generate an element of the simplex (i.e. a behavioral strategy x_a)
- Scale all its coordinates by \tilde{x}_{p_j} , i.e. $\tilde{x}_a = \tilde{x}_{p_j} \cdot x_a$



TreePlex Representation



Dilated Entropy

- X is a combination of *scaled simplices*, i.e., $\tilde{x} = (\tilde{x}^j)_{j \in \mathcal{J}_1}$
- $\tilde{x}^j = (\tilde{x}_a)_{a \in A_j}$: sequence-form strategies for actions in info set $j \in \mathcal{J}_1$

$$\tilde{x}^j \in \tilde{x}_{p_j} \cdot \Delta_j \quad \Leftrightarrow \quad \tilde{x}^j / \tilde{x}_{p_j} \in \Delta_j$$

- Consider a *weighted combination of local negative entropies*

$$\mathcal{R}(\tilde{x}) := \sum_j \beta_j \tilde{x}_{p_j} \text{H} \left(\tilde{x}^j / \tilde{x}_{p_j} \right), \quad \text{H}(u) = \sum_i u_i \log(u_i)$$

Lies in a simplex Δ_j
Negative Entropy

Equivalent to the behavioral strategy x^j

- $\mathcal{R}(\tilde{x})$ is $1/M$ strongly convex w.r.t. ℓ_1 norm, where $M = \max_{\tilde{x} \in X} \|\tilde{x}\|_1$, for appropriate choice of β_j based on game tree structure

Solving the Optimization Problem

- Optimization problem decomposes into local simplex problems

$$\sum_{j \in \mathcal{J}_1} \left\langle \tilde{x}^j, U_{t-1}^j \right\rangle - \underbrace{\frac{1}{\eta} \beta_j}_{:= \frac{1}{\eta_j}} \tilde{x}_{p_j} H \left(\frac{\tilde{x}^j}{\tilde{x}_{p_j}} \right) = \sum_{j \in \mathcal{J}_1} \tilde{x}_{p_j} \left\{ \left\langle \frac{\tilde{x}^j}{\tilde{x}_{p_j}}, U_{t-1}^j \right\rangle - \frac{1}{\eta_j} H \left(\frac{\tilde{x}^j}{\tilde{x}_{p_j}} \right) \right\}$$

- Quantity $\frac{\tilde{x}^j}{\tilde{x}_{p_j}}$ is essentially the behavioral strategy x^j at info set j

$$\sum_{j \in \mathcal{J}_1} \tilde{x}_{p_j} \left\{ \left\langle x^j, U_{t-1}^j \right\rangle - \frac{1}{\eta_j} H(x^j) \right\}$$

- Quantity x^j over simplex Δ_j is independent of solution x_a for all ancestral actions and only appears in subsequent info sets

Solving the Optimization Problem

- Decomposes in local max over behavioral strategies x^j solved bottom up

$$V^j = \max_{x^j \in \Delta_j} \left\langle x^j, U_{t-1}^j \right\rangle - \frac{1}{\eta_j} H(x^j) \Rightarrow \begin{cases} x^j \propto \exp(\eta_j U_{t-1}^j) \\ V^j = \log \sum_{a \in A_j} \exp(\eta_j U_{t-1}^a) = \text{softmax}_{\eta_j}(U_{t-1}^j) \end{cases}$$

- Value V^j multiplies \tilde{x}_{p_j} ; when solving for \tilde{x}_{p_j} we need to take it into account. If $p_j \in A_k$

$$\max_{x^k \in \Delta_k} \left\langle \tilde{x}^k, U_{t-1}^k \right\rangle - \eta_k \tilde{x}_{p_k} H\left(\frac{\tilde{x}^k}{\tilde{x}_{p_k}}\right) + \tilde{x}_{p_j} V^j + \dots$$

- Add V^j to “cumulative utility” Q_{p_j} (initialized at U_{t-1,p_j}) associated with p_j

$$Q_{p_j} \leftarrow Q_{p_j} + V^j$$

Sum: Nash via FTRL with Dilated Entropy

Each player chooses \tilde{x}_t, \tilde{y}_t based on FTRL with dilated entropy

- For x-player $u_t = A\tilde{y}_t$ and $U_t = U_{t-1} + u_t$ and initialize $Q = U_t$
- Traverse the tree bottom-up; for each info set $j \in \mathcal{J}_1$
$$x_{t+1}^j \propto \exp(\eta_j Q^j), \quad V^j = \text{softmax}_{\eta_j}(Q^j), \quad Q_{p_j} \leftarrow Q_{p_j} + V^j$$
- Define sequence-form strategies top-down: $\tilde{x}_{t+1}^j = \tilde{x}_{p_j} \cdot x_{t+1}^j$

Similarly, for y player

Return average of sequence-form strategies as equilibrium

Interpreting utility vector

$$u_{t,a} = A\tilde{y}_t = \sum_{a' \in A_{P2}} A_{a,a'} \tilde{y}_{t,a'}$$

$A_{a,a'}$ is zero if the combination of a, a' does not lead to a leaf node

$$u_{t,a} = \sum_{\text{Leaf } z: \substack{a \text{ was last P1 action} \\ a' \text{ was last P2 action}}} u(z) \Pr \left(\begin{array}{c} \text{Chance chooses} \\ \text{sequence on} \\ \text{path to } z \end{array} \right) \Pr \left(\begin{array}{c} \text{P2 plays} \\ \text{sequence} \\ \text{leading to } a' \end{array} \right)$$

Interpretation. If I play with the intend to arrive at action a (i.e. $\tilde{x}_a = 1$) *and then don't make any other moves*, what is the expected reward that I will collect, in expectation over the choices of my opponent and nature

Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(\begin{array}{c} \end{array} \right)$$

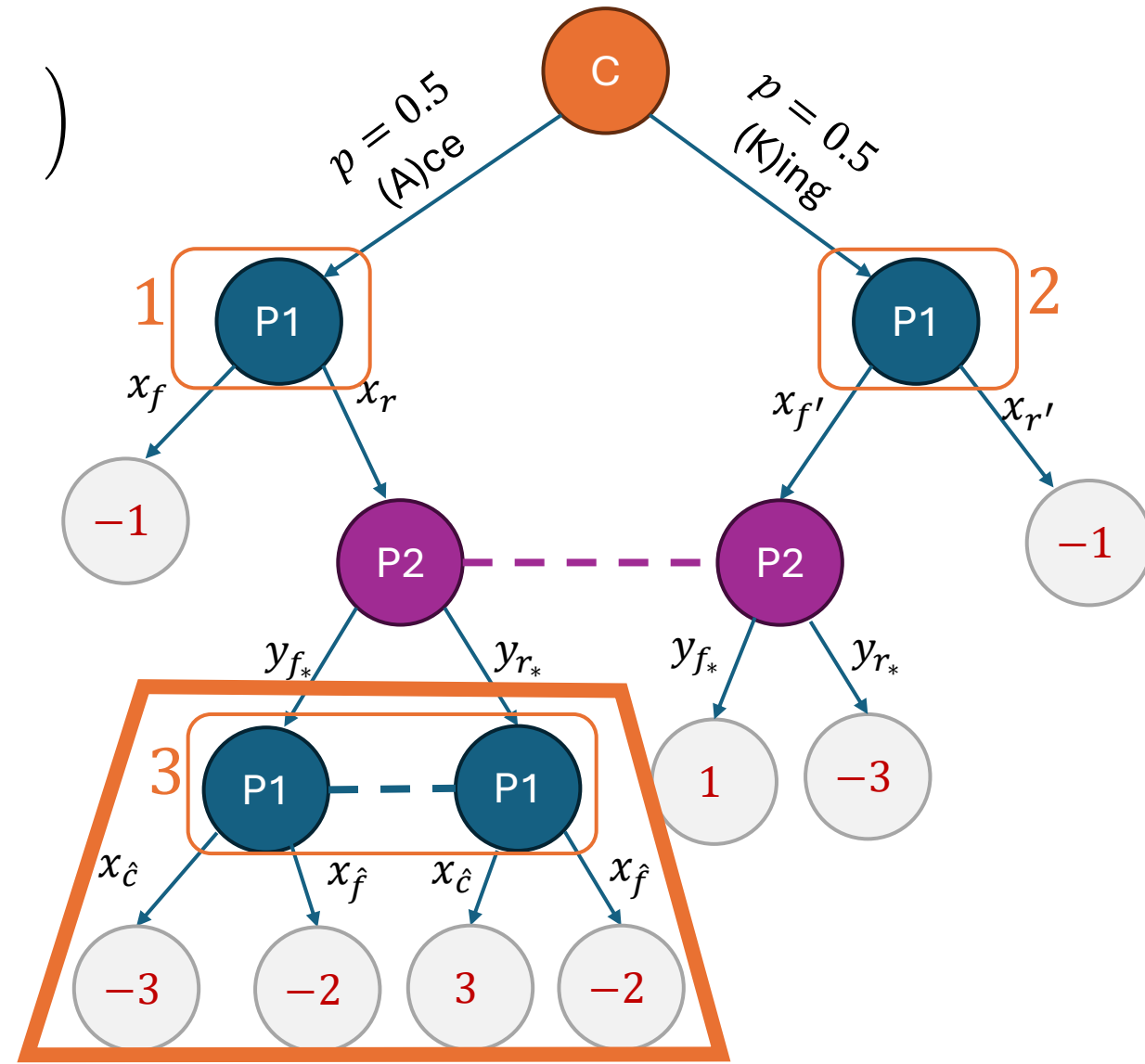


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

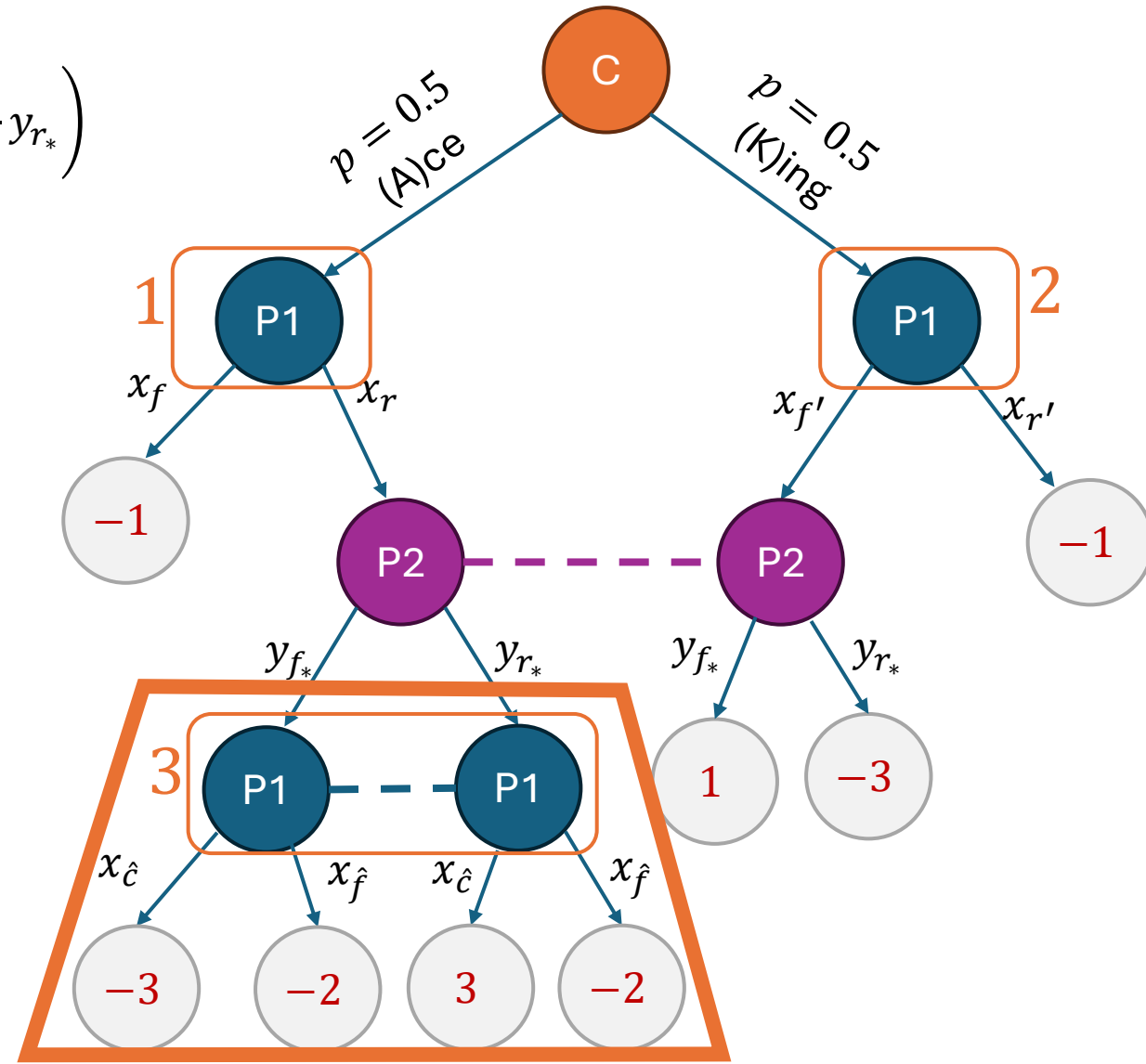


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

$$Q^3 = U^3, \quad x^3 = (x_{\hat{c}}, x_{\hat{f}}) \propto \exp(\eta_3 Q^3)$$

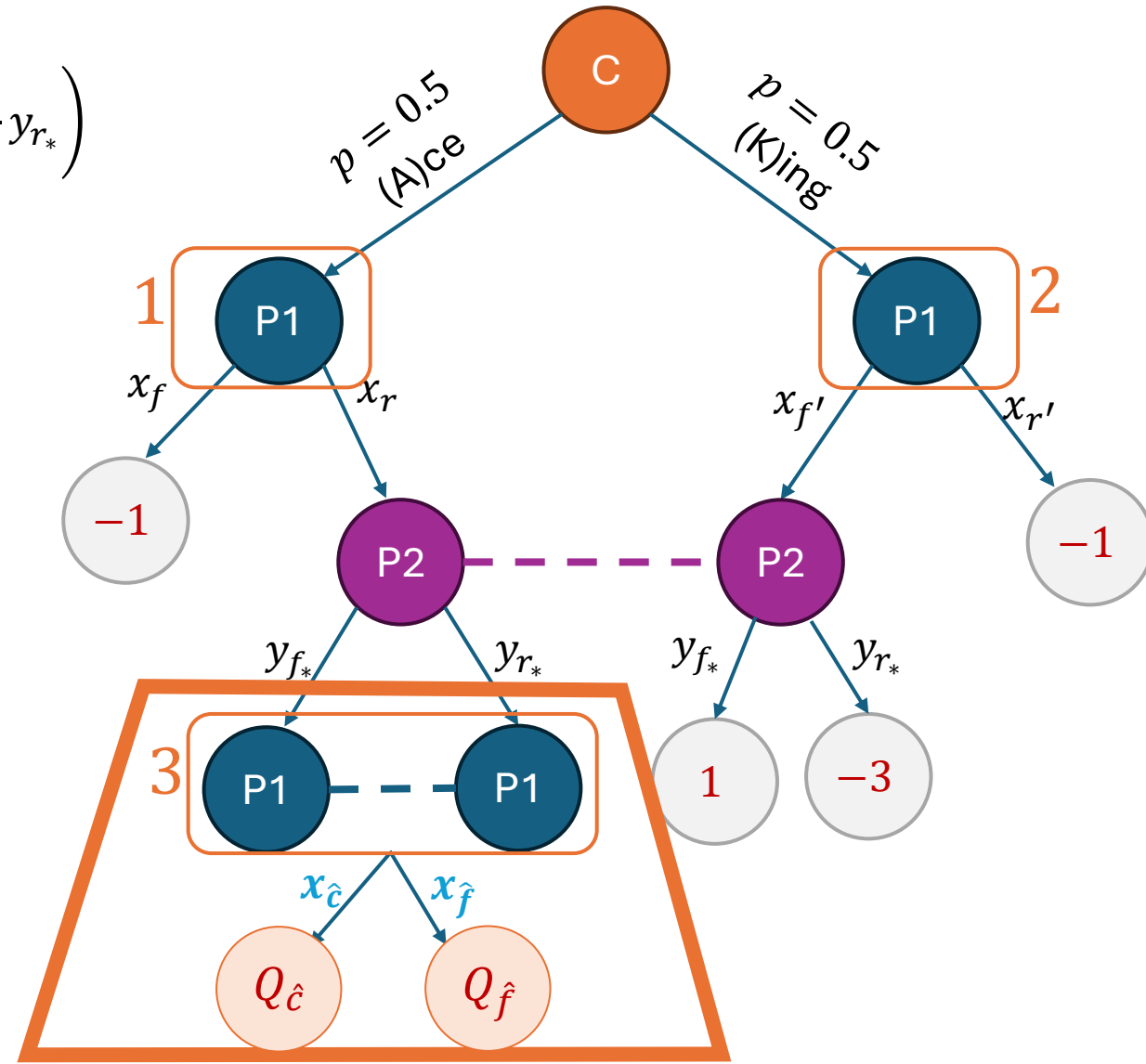


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

$$Q^3 = U^3, \quad x^3 = (x_{\hat{c}}, x_{\hat{f}}) \propto \exp(\eta_3 Q^3)$$

$$V^3 = \text{softmax}(\eta_3 Q^3)$$

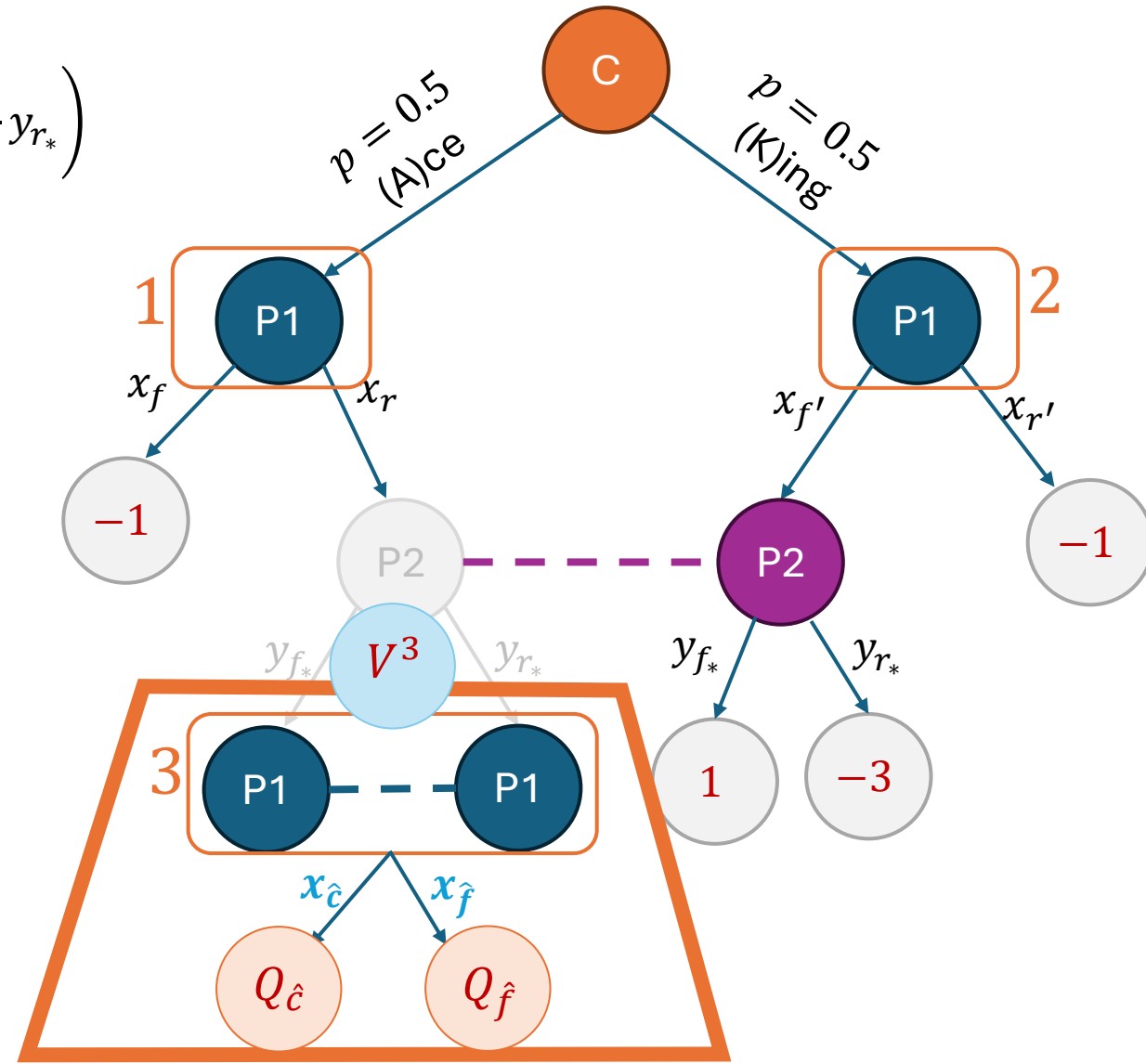


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

$$Q^3 = U^3, \quad x^3 = (x_{\hat{c}}, x_{\hat{f}}) \propto \exp(\eta_3 Q^3)$$

$$V^3 = \text{softmax}(\eta_3 Q^3)$$

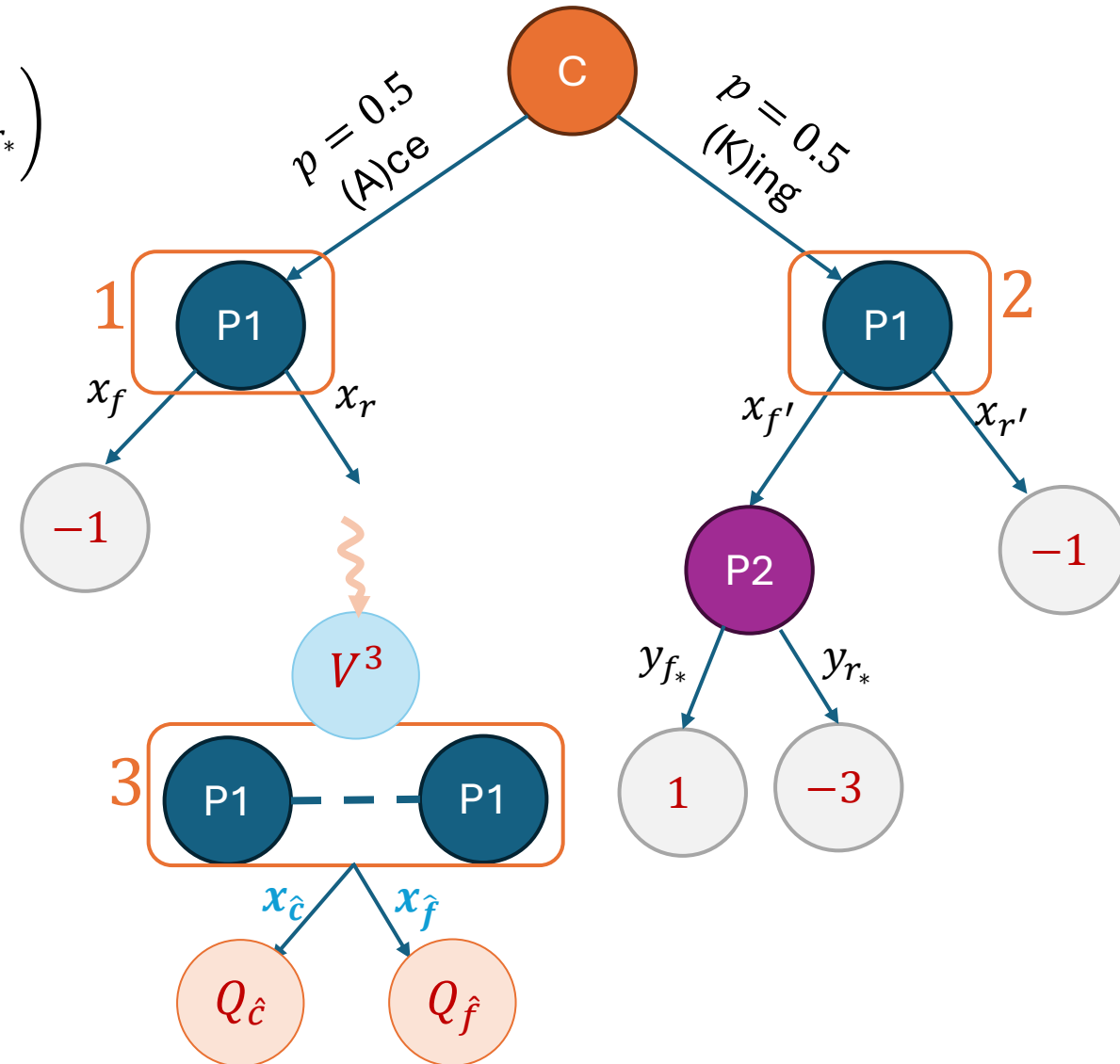


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

$$Q^3 = U^3, \quad x^3 = (x_{\hat{c}}, x_{\hat{f}}) \propto \exp(\eta_3 Q^3)$$

$$V^3 = \text{softmax}(\eta_3 Q^3)$$

- Go to **InfoSet 1**

$$U^1 += (u_f, u_r) = \left(\begin{array}{c} \\ \end{array} \right)$$

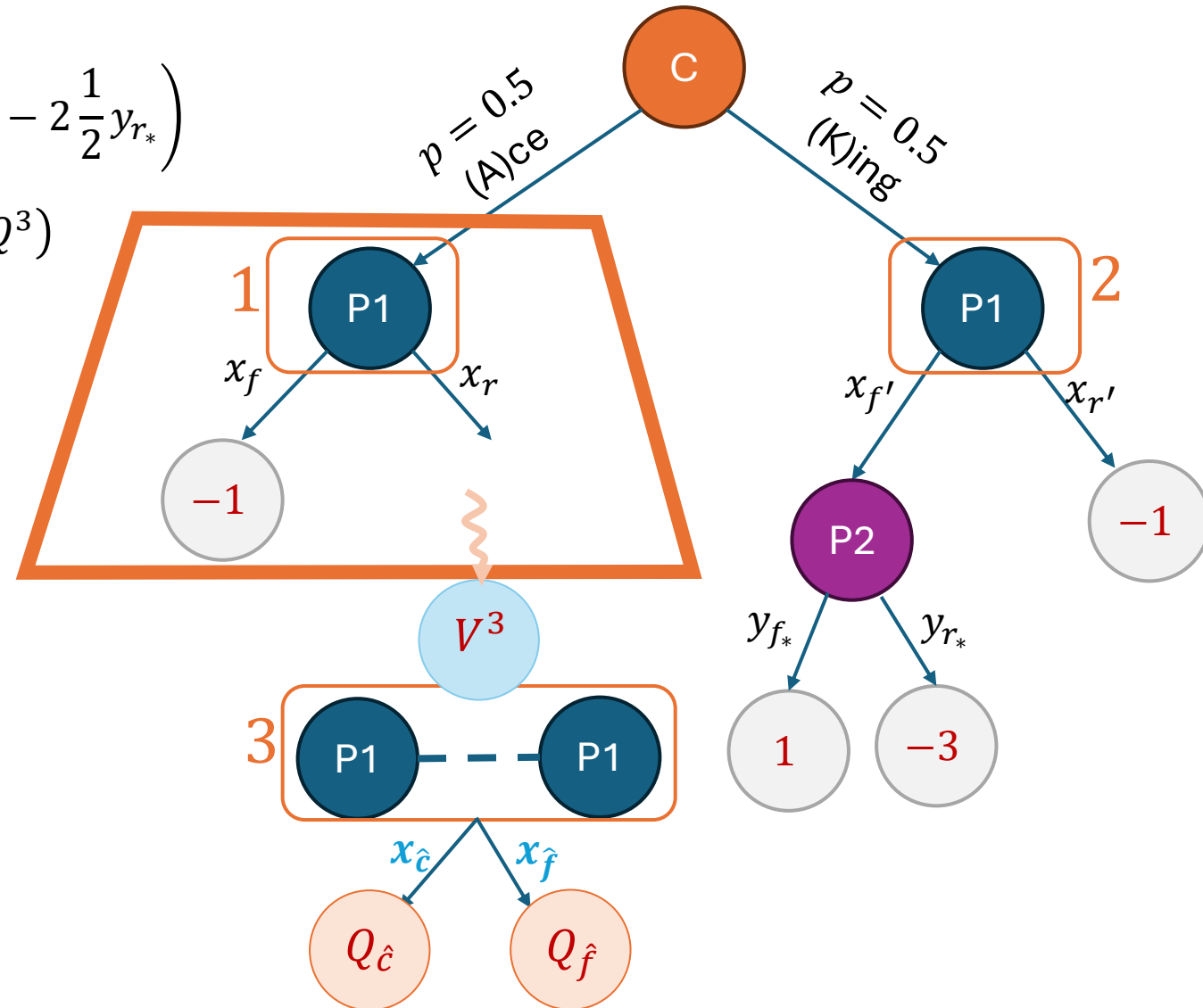


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

$$Q^3 = U^3, \quad x^3 = (x_{\hat{c}}, x_{\hat{f}}) \propto \exp(\eta_3 Q^3)$$

$$V^3 = \text{softmax}(\eta_3 Q^3)$$

- Go to **InfoSet 1**

$$U^1 += (u_f, u_r) = \left(-1\frac{1}{2}, 0 \right)$$

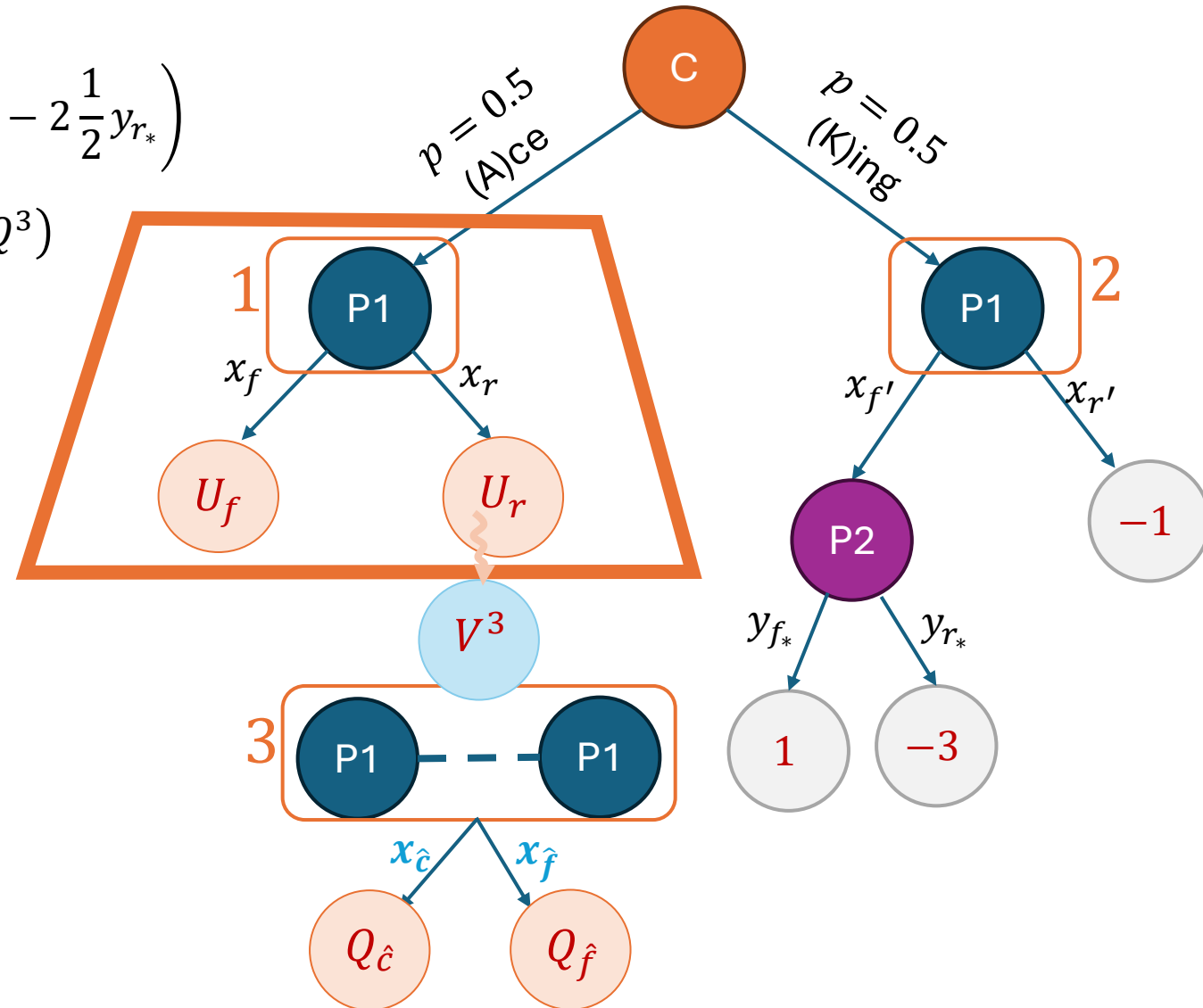


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

$$Q^3 = U^3, \quad x^3 = (x_{\hat{c}}, x_{\hat{f}}) \propto \exp(\eta_3 Q^3)$$

$$V^3 = \text{softmax}(\eta_3 Q^3)$$

- Go to **InfoSet 1**

$$U^1 += (u_f, u_r) = \left(-1\frac{1}{2}, 0 \right)$$

$$Q^1 = U^1 + (0, V^3) = \left(-1\frac{1}{2}, V^3 \right)$$

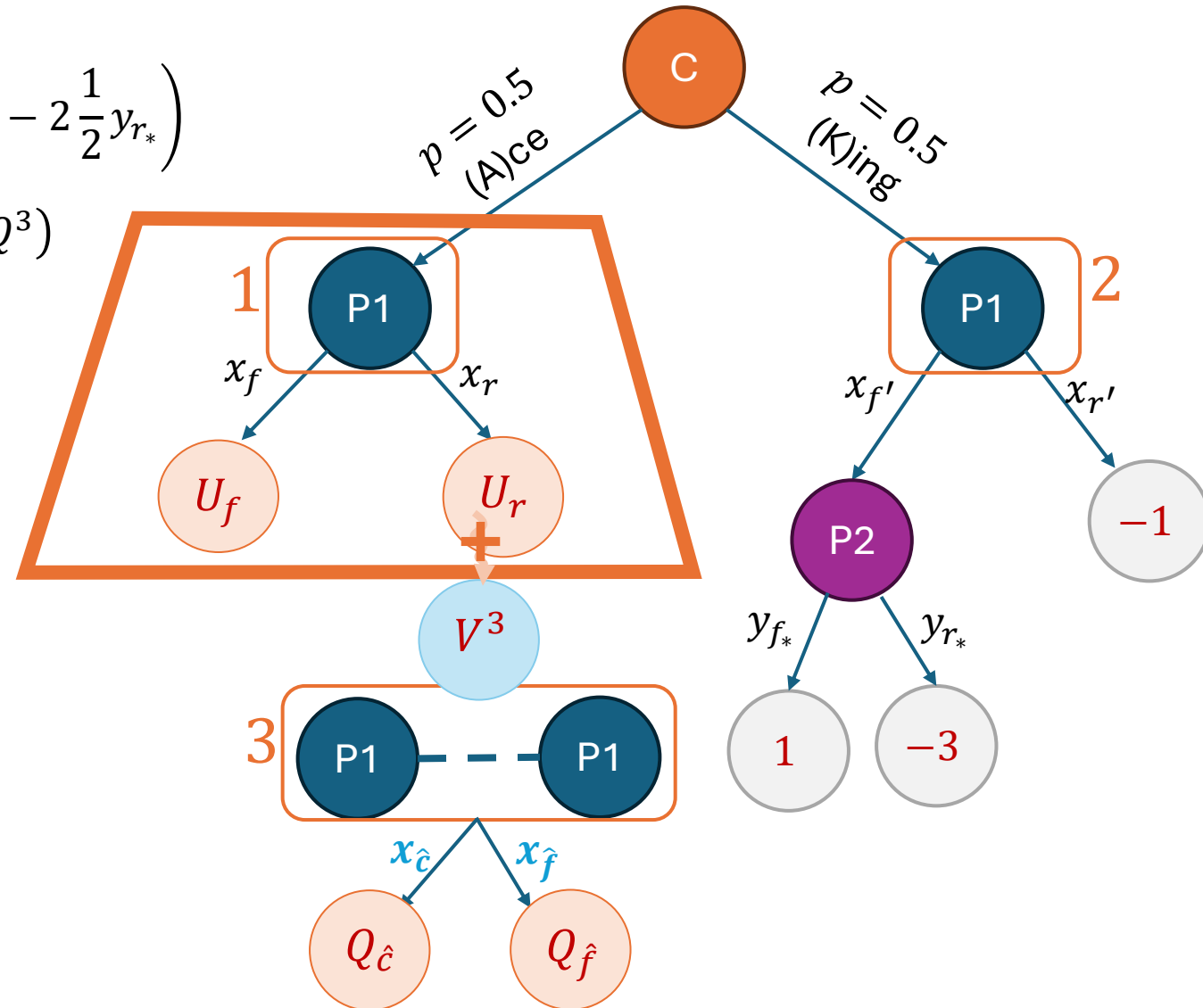


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

$$Q^3 = U^3, \quad x^3 = (x_{\hat{c}}, x_{\hat{f}}) \propto \exp(\eta_3 Q^3)$$

$$V^3 = \text{softmax}(\eta_3 Q^3)$$

- Go to **InfoSet 1**

$$U^1 += (u_f, u_r) = \left(-1\frac{1}{2}, 0 \right)$$

$$Q^1 = U^1 + (0, V^3) = \left(-1\frac{1}{2}, V^3 \right)$$

$$x^1 = (x_f, x_r) \propto \exp(\eta_1 Q^1)$$

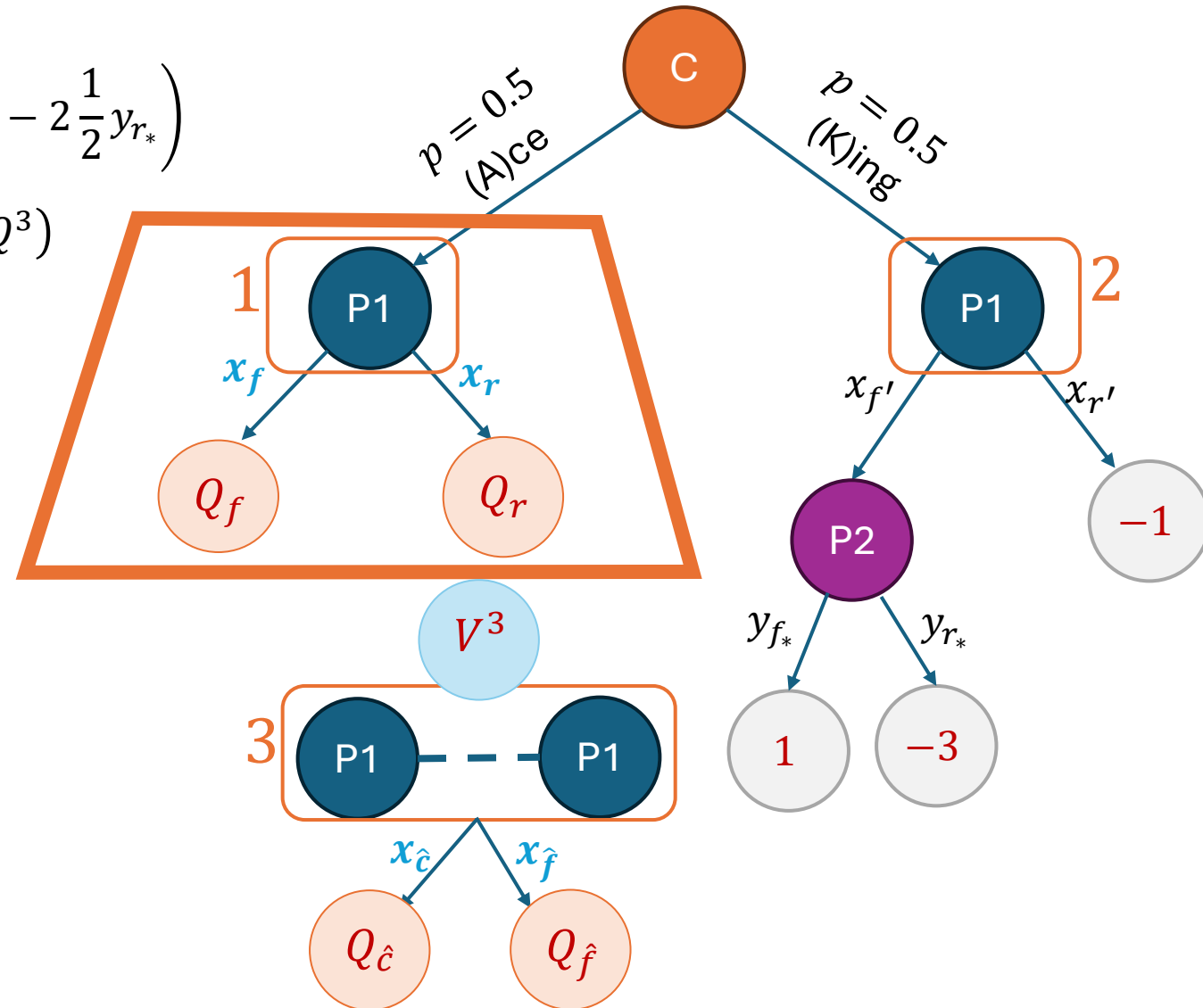


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(-3\frac{1}{2}y_{f*} + 3\frac{1}{2}y_{r*}, -2\frac{1}{2}y_{f*} - 2\frac{1}{2}y_{r*} \right)$$

$$Q^3 = U^3, \quad x^3 = (x_{\hat{c}}, x_{\hat{f}}) \propto \exp(\eta_3 Q^3)$$

$$V^3 = \text{softmax}(\eta_3 Q^3)$$

- Go to **InfoSet 1**

$$U^1 += (u_f, u_r) = \left(-1\frac{1}{2}, 0 \right)$$

$$Q^1 = U^1 + (0, V^3) = \left(-1\frac{1}{2}, V^3 \right)$$

$$x^1 = (x_f, x_r) \propto \exp(\eta_1 Q^1)$$

- Go to **InfoSet 2**

$$U^2 += (u_{f'}, u_{r'}) = \left(\begin{array}{c} \\ \end{array} \right)$$

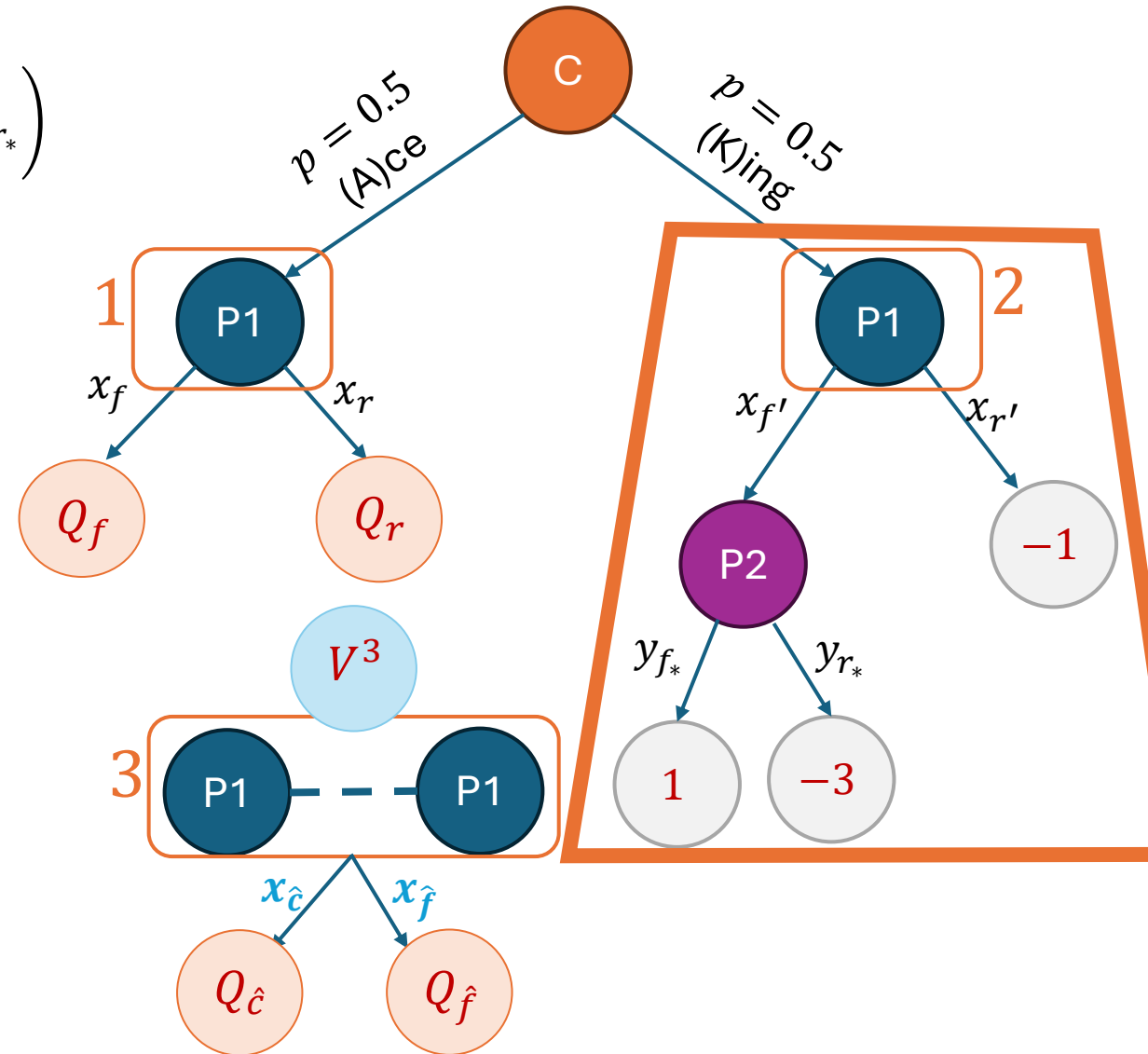


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

$$Q^3 = U^3, \quad x^3 = (x_{\hat{c}}, x_{\hat{f}}) \propto \exp(\eta_3 Q^3)$$

$$V^3 = \text{softmax}(\eta_3 Q^3)$$

- Go to **InfoSet 1**

$$U^1 += (u_f, u_r) = \left(-1\frac{1}{2}, 0 \right)$$

$$Q^1 = U^1 + (0, V^3) = \left(-1\frac{1}{2}, V^3 \right)$$

$$x^1 = (x_f, x_r) \propto \exp(\eta_1 Q^1)$$

- Go to **InfoSet 2**

$$U^2 += (u_{f'}, u_{r'}) = \left(1\frac{1}{2}y_{f_*} - 3\frac{1}{2}y_{r_*}, -1\frac{1}{2} \right)$$

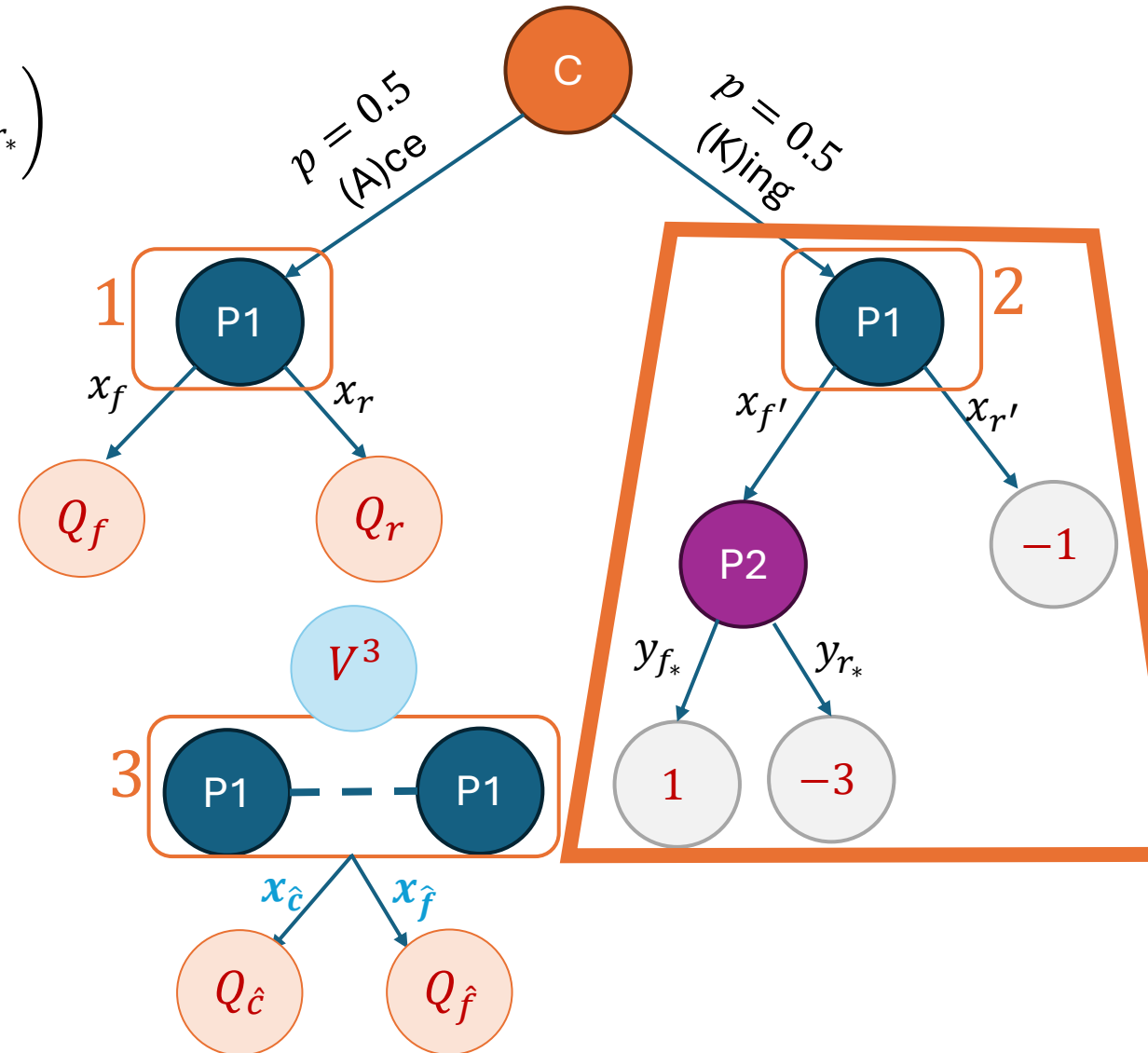


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$U^3 += (u_{\hat{c}}, u_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

$$Q^3 = U^3, \quad x^3 = (x_{\hat{c}}, x_{\hat{f}}) \propto \exp(\eta_3 Q^3)$$

$$V^3 = \text{softmax}(\eta_3 Q^3)$$

- Go to **InfoSet 1**

$$U^1 += (u_f, u_r) = \left(-1\frac{1}{2}, 0 \right)$$

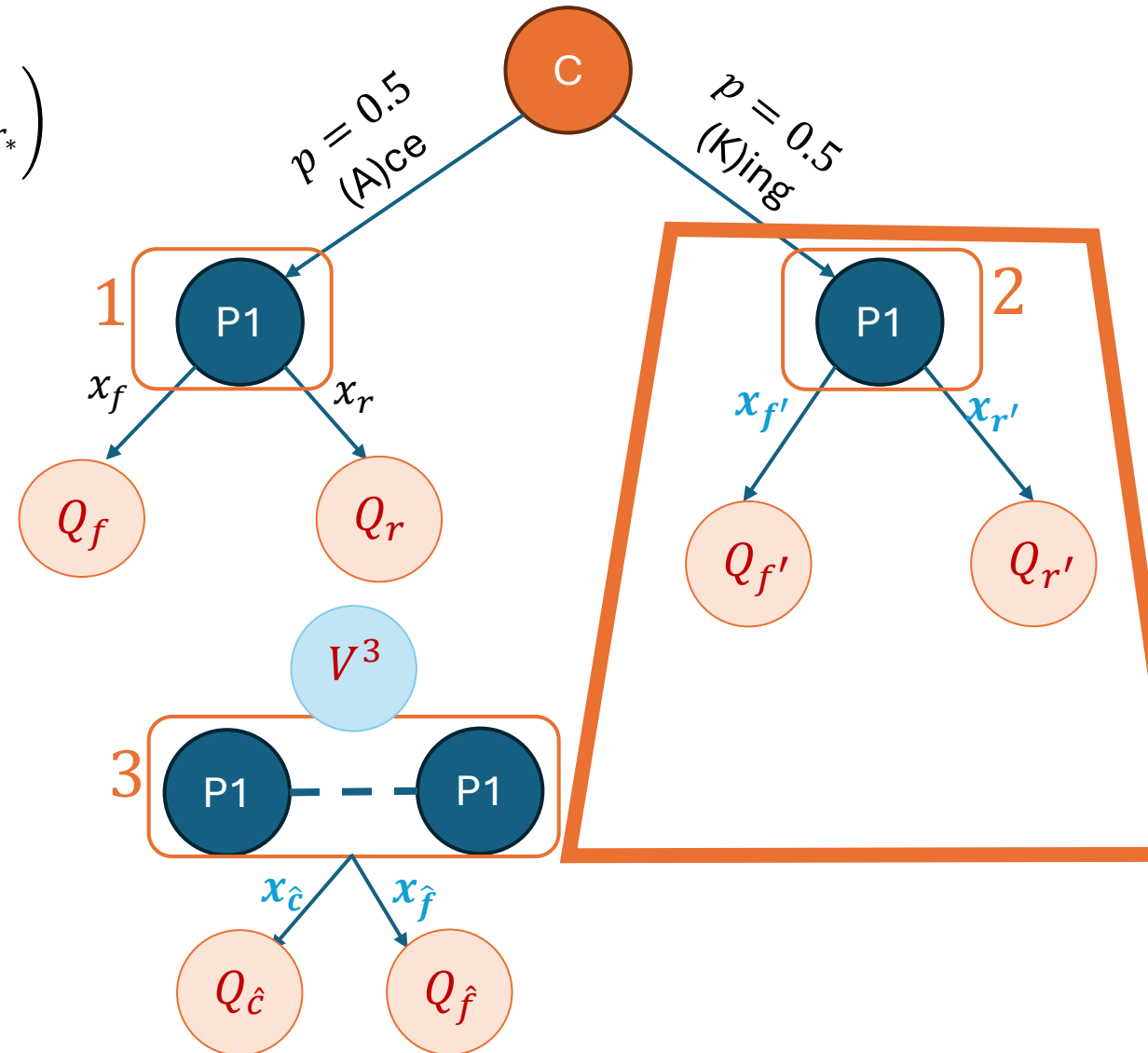
$$Q^1 = U^1 + (0, V^3) = \left(-1\frac{1}{2}, V^3 \right)$$

$$x^1 = (x_f, x_r) \propto \exp(\eta_1 Q^1)$$

- Go to **InfoSet 2**

$$U^2 += (u_{f'}, u_{r'}) = \left(1\frac{1}{2}y_{f_*} - 3\frac{1}{2}y_{r_*}, -1\frac{1}{2} \right)$$

$$Q^2 = U^2, \quad x^2 = (x_{f'}, x_{r'}) \propto \exp(\eta_2 Q^2)$$



Sum: Nash via FTRL with Dilated Entropy

Each player chooses \tilde{x}_t, \tilde{y}_t based on FTRL with dilated entropy

- For x-player $u_t = A\tilde{y}_t$ and $U_t = U_{t-1} + u_t$ and initialize $Q = U_t$
- Traverse the tree bottom-up; for each info set $j \in \mathcal{J}_1$
$$x_{t+1}^j \propto \exp(\eta_j Q^j), \quad V^j = \text{softmax}_{\eta_j}(Q^j), \quad Q_{p_j} \leftarrow Q_{p_j} + V^j$$
- Define sequence-form strategies top-down: $\tilde{x}_{t+1}^j = \tilde{x}_{p_j} \cdot x_{t+1}^j$

Similarly, for y player

Return average of sequence-form strategies as equilibrium

Fast Rates

Theorem. If we use Optimistic FTRL instead of FTRL then we get faster convergence to a Nash equilibrium at rate $1/T$ instead of $1/\sqrt{T}$. Plus, we get last-iterate convergence instead of only average iterate convergence.

Local Dynamics

- These dynamics seem to be doing “local updates” at each node
- They came out of a specific algorithm FTRL with Dilated Entropy
- Is this a general paradigm?
- Can we decompose the no-regret learning problem into local no-regret learners at each node?
- What feedback should each node receive from the learners in nodes below?
- What loss should each learner be optimizing?

Counterfactual Regret Minimization (CRM)

Re-interpreting Utilities

Interpretation of u_a . If I play with the intent to arrive at action a (i.e. $\tilde{x}_a = 1$) *and then don't make any other moves*, what is the expected reward that I will collect, in expectation over the choices of my opponent and nature

What if we now want to express: If I play with the intent to arrive at action a (i.e. $\tilde{x}_a = 1$) *and then continue playing based on some behavioral policy x* , what is the expected reward that I will collect, in expectation over the choices of my opponent and nature

Re-interpreting Utilities

Interpretation of u_a . If I play with the intent to arrive at action a (i.e. $\tilde{x}_a = 1$) *and then don't make any other moves*, what is the expected reward that I will collect, in expectation over the choices of my opponent and nature

What if we now want to express: If I play with the intent to arrive at action a (i.e. $\tilde{x}_a = 1$) *and then continue playing based on some behavioral policy x* , what is the expected reward that I will collect, in expectation over the choices of my opponent and nature

- Let C_a be all infosets of the player that are reachable **as next infosets** after playing a

$$\tilde{u}_a(x) = \boxed{u_a} + \sum_{k \in C_a} \boxed{V^k(x)}$$

“Instantaneous $E[\text{utility}]$ ”, if this is the last action I play

Continuation $E[\text{utility}]$ from paths that pass through infoset k , if I continue playing based on behavioral strategy x

Re-interpreting Utilities

Interpretation of u_a . If I play with the intend to arrive at action a (i.e. $\tilde{x}_a = 1$) *and then don't make any other moves*, what is the expected reward that I will collect, in expectation over the choices of my opponent and nature

What if we now want to express: If I play with the intend to arrive at action a (i.e. $\tilde{x}_a = 1$) *and then continue playing based on some behavioral policy x* , what is the expected reward that I will collect, in expectation over the choices of my opponent and nature

- Let C_a be all infosets of the player that are reachable **as next infosets** after playing a

$$\tilde{u}_a(x) = \underbrace{u_a}_{\text{"Instantaneous E[utility]", if this is the last action I play}} + \sum_{k \in C_a} \underbrace{V^k(x)}_{\text{"Continuation E[utility] from paths that pass through info set } k, \text{ if I continue playing based on behavioral strategy } x}$$

- Continuation utility $V^j(x)$ from paths that pass through info set j recursively defined:

$$V^j(x) = \sum_{a \in A^j} x_a \tilde{u}_a(x) = \underbrace{\sum_{a \in A^j} x_a u_a}_{\text{"Instantaneous utility", if this is the last move I make}} + \underbrace{\sum_{a \in A^j} x_a \left(\sum_{k \in C_a} V^k(x) \right)}_{\text{"Continuation utility", if I continue playing based on } x}$$

Re-interpreting Utilities

- Continuation utility $V^j(x)$ from paths that pass through j , assuming I play to arrive deterministically at the parent action p_j (i.e., $\tilde{x}_{p_j} = 1$)

$$V^j(x) = \sum_{a \in A^j} x_a \tilde{u}_a(x) = \sum_{a \in A^j} x_a \left(u_a + \sum_{k \in C_a} V^k(x) \right)$$

- Obviously $V^{\text{root}}(x)$ is total expected utility from behavior strategy x
- From equivalence of behavioral and sequence-form strategies

$$V^{\text{root}}(x) = \langle \tilde{x}, u \rangle$$

- The same also holds for regrets

$$R^{\text{root}}(x) = \max_{x'} V^{\text{root}}(x') - V^{\text{root}}(x) = \max_{\tilde{x}' \in X} \langle \tilde{x}', u \rangle - \langle \tilde{x}, u \rangle = R(\tilde{x})$$

Local Regrets

- We can also define infoset regrets based on local utilities \tilde{u}_a

$$R^j(x) = \max_{x'} V^j(x') - V^j(x) = \max_{x'} \sum_a x'_a \tilde{u}_a(x') - x_a \tilde{u}_a(x)$$

- Right-hand-side can be decomposed as:

$$\max_{x'} \left(\sum_a x'_a \tilde{u}_a(x) - x_a \tilde{u}_a(x) \right) + \left(\sum_a x'_a (\tilde{u}_a(x') - \tilde{u}_a(x)) \right)$$

Fix continuation strategy to current strategy and only change the behavioral strategy at the current infoset

Weighted average of changes in continuation strategy

Local Regrets

- We can also define info set regrets based on local utilities \tilde{u}_a

$$R^j(x) = \max_{x'} V^j(x') - V^j(x) = \max_{x'} \sum_a x'_a \tilde{u}_a(x') - x_a \tilde{u}_a(x)$$

- Right-hand-side can be decomposed as:

$$\max_{x'} \sum_a x'_a \tilde{u}_a(x) - x_a \tilde{u}_a(x) + \sum_a x'_a (\tilde{u}_a(x') - \tilde{u}_a(x))$$

- Maximum is upper bounded by the decoupled optima

$$\boxed{\max_{x'} \sum_a x'_a \tilde{u}_a(x) - x_a \tilde{u}_a(x)} + \sum_a \max_{x'} (\tilde{u}_a(x') - \tilde{u}_a(x))$$

Local Regret: $LR^j(x)$

Regret if you only change current info set behavioral strategy and keep continuation strategy

Recursive Bound of Local Regrets

- Infoset regrets are bounded by local regret plus continuation terms

$$R^j(x) \leq \text{LR}^j(x) + \sum_a \max_{x'} (\tilde{u}_a(x') - \tilde{u}_a(x))$$

- The continuation terms are recursive infoset regrets!

$$\tilde{u}_a(x') - \tilde{u}_a(x) = \cancel{u_a} + \sum_{k \in C_a} V^k(x') - \cancel{u_a} - \sum_{k \in C_a} V^k(x)$$

- Deriving the recursive upper bound

$$\begin{aligned} R^j(x) &\leq \text{LR}^j(x) + \sum_a \sum_{k \in C_a} \max_{x'} V^k(x') - V^k(x) \\ &\leq \text{LR}^j(x) + \sum_a \sum_{k \in C_a} R^k(x) \end{aligned}$$

Recursive Bound of Local Regrets

- Deriving the recursive upper bound

$$R^j(x) \leq \text{LR}^j(x) + \sum_a \sum_{k \in C_a} R^k(x)$$

Recursive Bound of Local Regrets

- Deriving the recursive upper bound

$$R^j(x) \leq LR^j(x) + \sum_a \sum_{k \in C_a} R^k(x)$$

Theorem. By induction:

$$R^j(x) \leq LR^j(x) + \sum_{k \text{ eventually reachable from } j} LR^k(x)$$

Local Regrets Upper Bound Total Regret

- Deriving the recursive upper bound

$$R^j(x) \leq LR^j(x) + \sum_a \sum_{k \in C_a} R^k(x)$$

Theorem. By induction:

$$R^j(x) \leq LR^j(x) + \sum_{k \text{ eventually reachable from } j} LR^k(x)$$

Main Corollary. Regret is upper bounded by sum of local regrets

$$R(\tilde{x}) = R^{\text{root}}(x) \leq \sum_{k \in \mathcal{J}_1} LR^k(x)$$

Regret over Time

Same inequalities can be followed for the average regret over time

$$R = \max_{\tilde{x}' \in X} \frac{1}{T} \sum_t \langle \tilde{x}', u_t \rangle - \langle \tilde{x}_t, u_t \rangle$$

$$LR^j = \max_{x^j} \frac{1}{T} \sum_t \langle x^j, \tilde{u}_t(x_t) \rangle - \langle x_t^j, \tilde{u}_t(x_t) \rangle$$

Main CFR Theorem. Regret is upper bounded by local regrets

$$R \leq \sum_{j \in \mathcal{L}_1} LR^j$$

Achieving vanishing Local Regrets

$$\text{LR}^j(x) = \max_{x^j} \frac{1}{T} \sum_t \langle x^j, \tilde{u}_t(x_t) \rangle - \langle x_t^j, \tilde{u}_t(x_t) \rangle$$

Counterfactual Regret Minimization

- Device local regret algorithms for local regret

$$\text{LR}^j(x) = \max_{x^j} \frac{1}{T} \sum_t \langle x^j, \tilde{u}_t(x_t) \rangle - \left\langle x_t^j, \tilde{u}_t(x_t) \right\rangle$$

- Standard n -action no-regret problem: reward vector at period t is $\tilde{u}^j(x_t)$ and reward for choice x^j is $\langle x^j, \tilde{u}^j(x_t) \rangle$
- At period t run bottom-up recursion to calculate $\tilde{u}^j(x_t)$ for $j \in \mathcal{J}_1$
- Update probabilities x_{t+1}^j using reward vectors $\tilde{u}^j(x_t)$ for $j \in \mathcal{J}_1$

Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$(\tilde{u}_{\hat{c}}, \tilde{u}_{\hat{f}}) = \left(-3\frac{1}{2}y_{f*} + 3\frac{1}{2}y_{r*}, -2\frac{1}{2}y_{f*} - 2\frac{1}{2}y_{r*} \right)$$

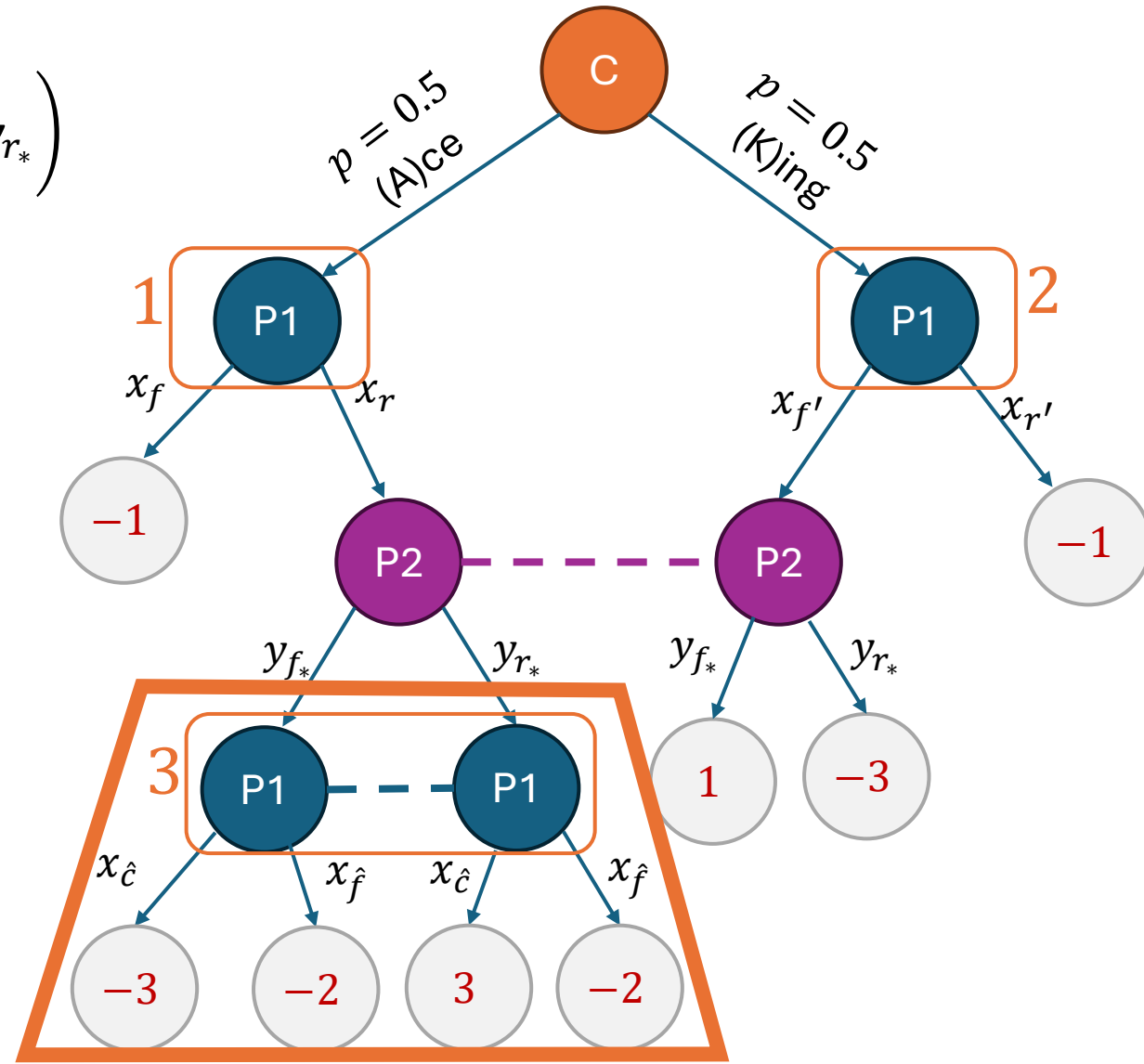


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$(\tilde{u}_{\hat{c}}, \tilde{u}_{\hat{f}}) = \left(-3\frac{1}{2}y_{f*} + 3\frac{1}{2}y_{r*}, -2\frac{1}{2}y_{f*} - 2\frac{1}{2}y_{r*} \right)$$

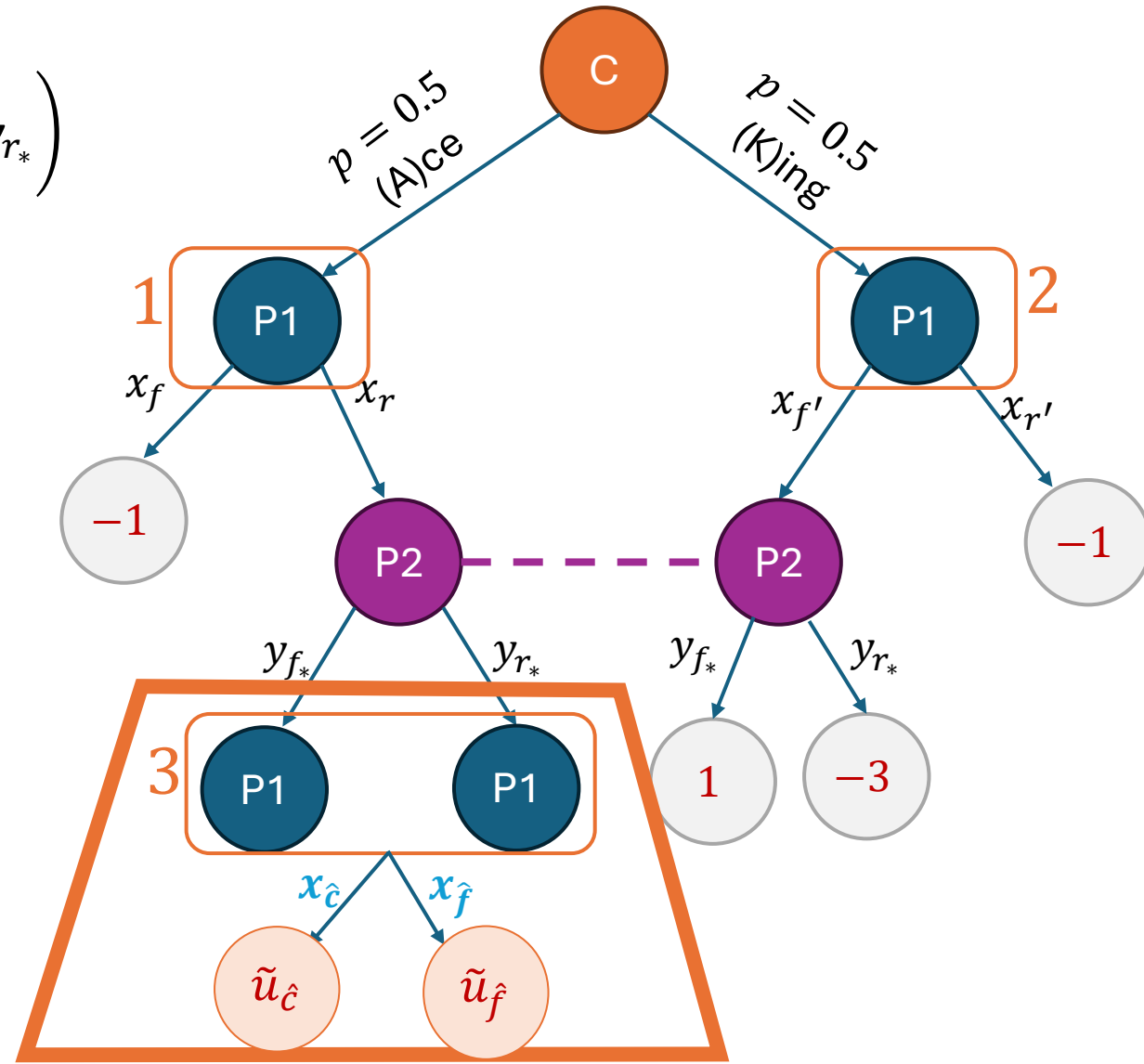


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$(\tilde{u}_{\hat{c}}, \tilde{u}_{\hat{f}}) = \left(-3\frac{1}{2}y_{f*} + 3\frac{1}{2}y_{r*}, -2\frac{1}{2}y_{f*} - 2\frac{1}{2}y_{r*} \right)$$

$$V^3 \leftarrow x_{\hat{c}}\tilde{u}_{\hat{c}} + x_{\hat{f}}\tilde{u}_{\hat{f}}$$

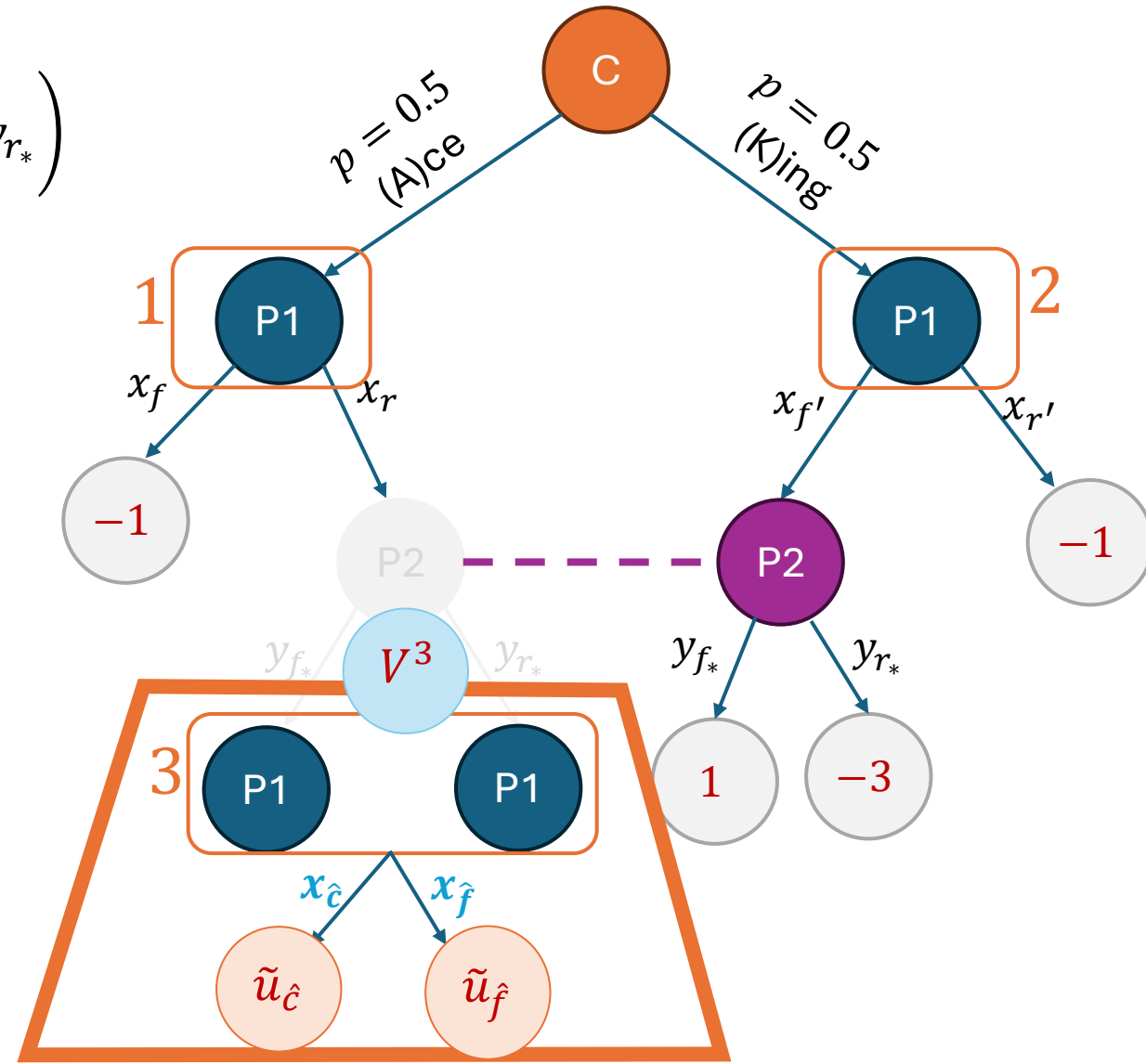


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$(\tilde{u}_{\hat{c}}, \tilde{u}_{\hat{f}}) = \left(-3\frac{1}{2}y_{f*} + 3\frac{1}{2}y_{r*}, -2\frac{1}{2}y_{f*} - 2\frac{1}{2}y_{r*} \right)$$

$$V^3 \leftarrow x_{\hat{c}}\tilde{u}_{\hat{c}} + x_{\hat{f}}\tilde{u}_{\hat{f}}$$

- Go to **InfoSet 1**

$$(\tilde{u}_f, \tilde{u}_r) = \left(-1\frac{1}{2}, V^3 \right)$$

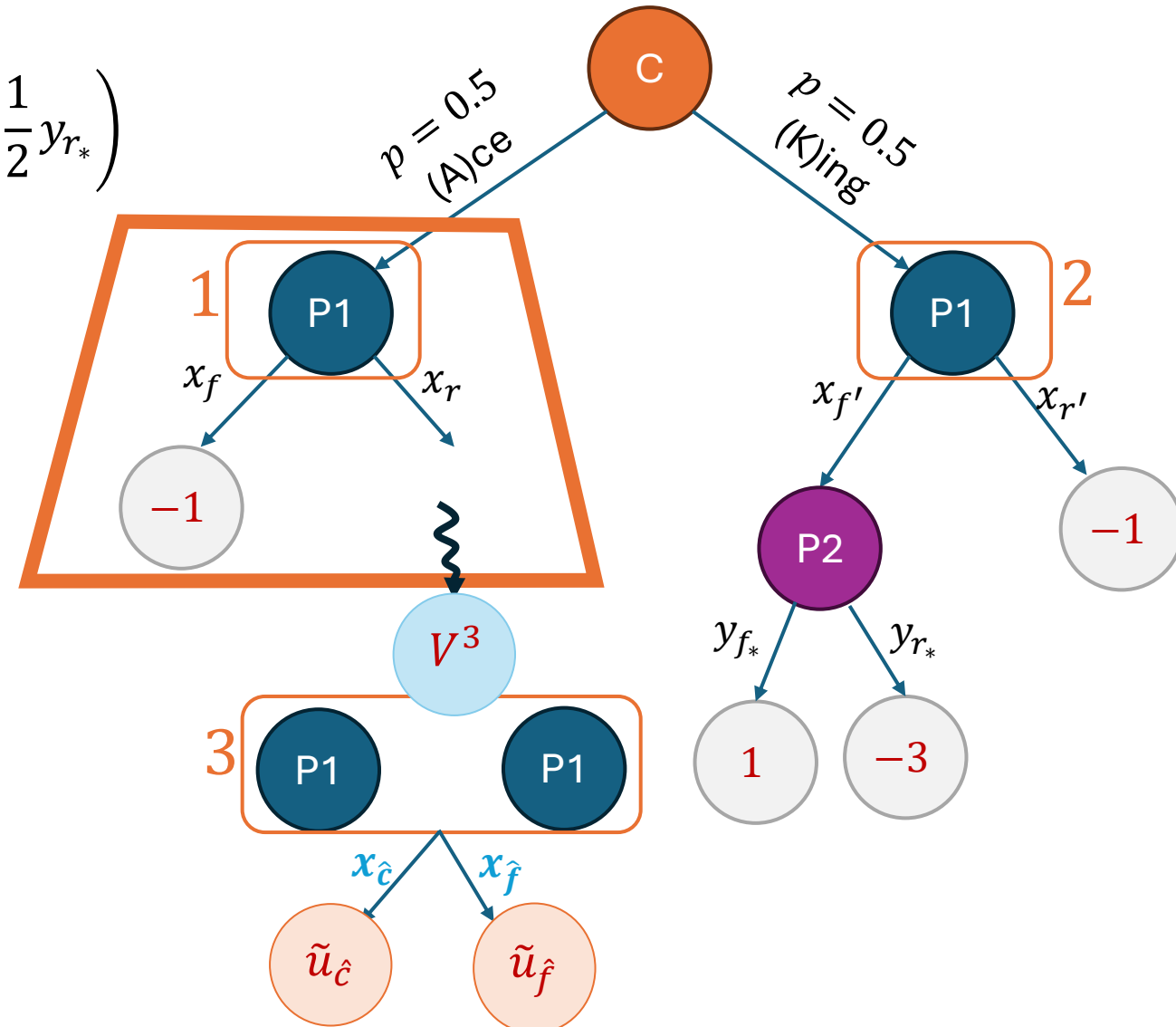


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$(\tilde{u}_{\hat{c}}, \tilde{u}_{\hat{f}}) = \left(-3\frac{1}{2}y_{f*} + 3\frac{1}{2}y_{r*}, -2\frac{1}{2}y_{f*} - 2\frac{1}{2}y_{r*} \right)$$

$$V^3 \leftarrow x_{\hat{c}}\tilde{u}_{\hat{c}} + x_{\hat{f}}\tilde{u}_{\hat{f}}$$

- Go to **InfoSet 1**

$$(\tilde{u}_f, \tilde{u}_r) = \left(-1\frac{1}{2}, V^3 \right)$$

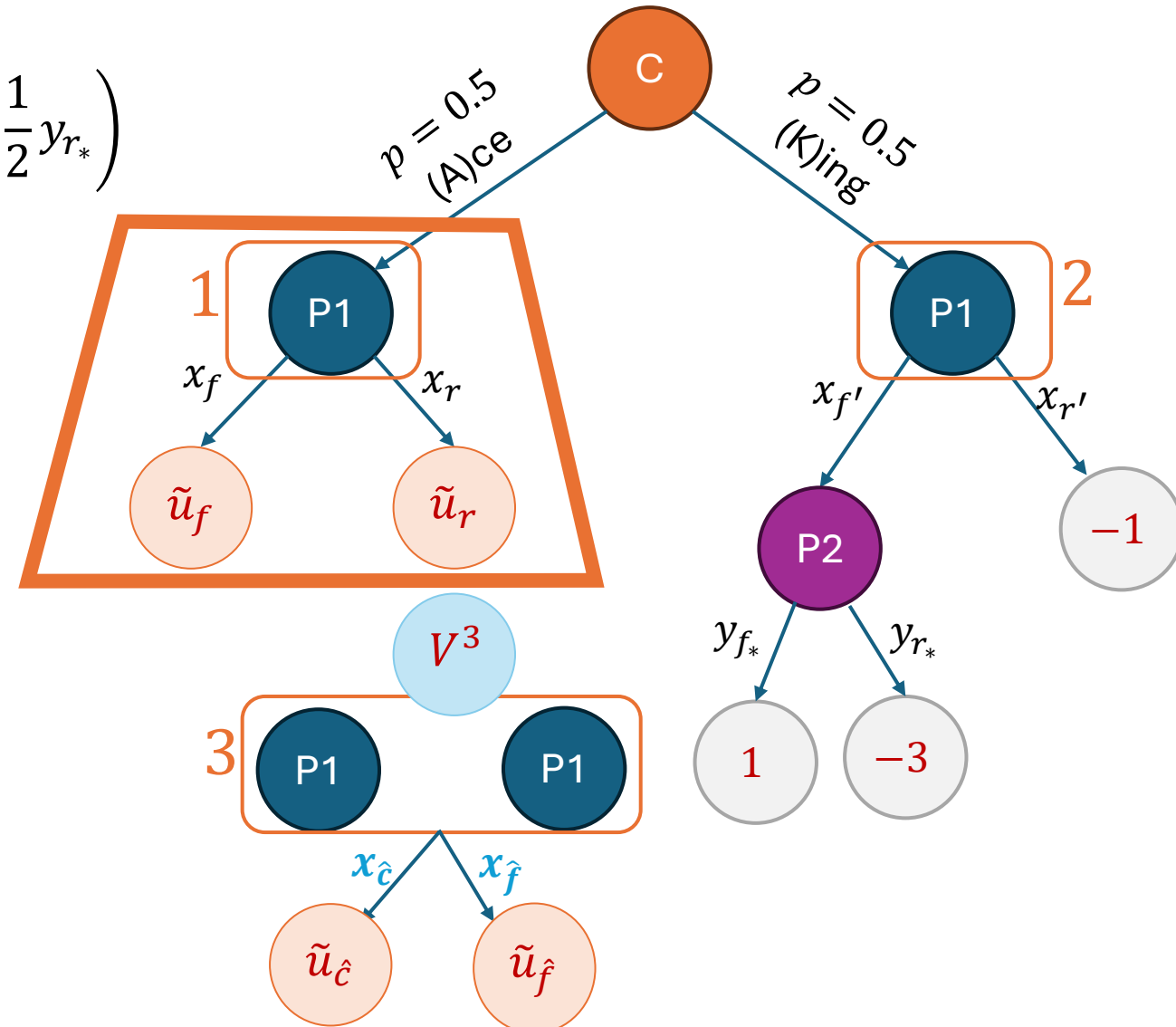


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$(\tilde{u}_{\hat{c}}, \tilde{u}_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

$$V^3 \leftarrow x_{\hat{c}}\tilde{u}_{\hat{c}} + x_{\hat{f}}\tilde{u}_{\hat{f}}$$

- Go to **InfoSet 1**

$$(\tilde{u}_f, \tilde{u}_r) = \left(-1\frac{1}{2}, V^3 \right)$$

- Go to **InfoSet 2**

$$(\tilde{u}_{f'}, \tilde{u}_{r'}) = \left(1\frac{1}{2}y_{f_*} - 3\frac{1}{2}y_{r_*}, -1\frac{1}{2} \right)$$

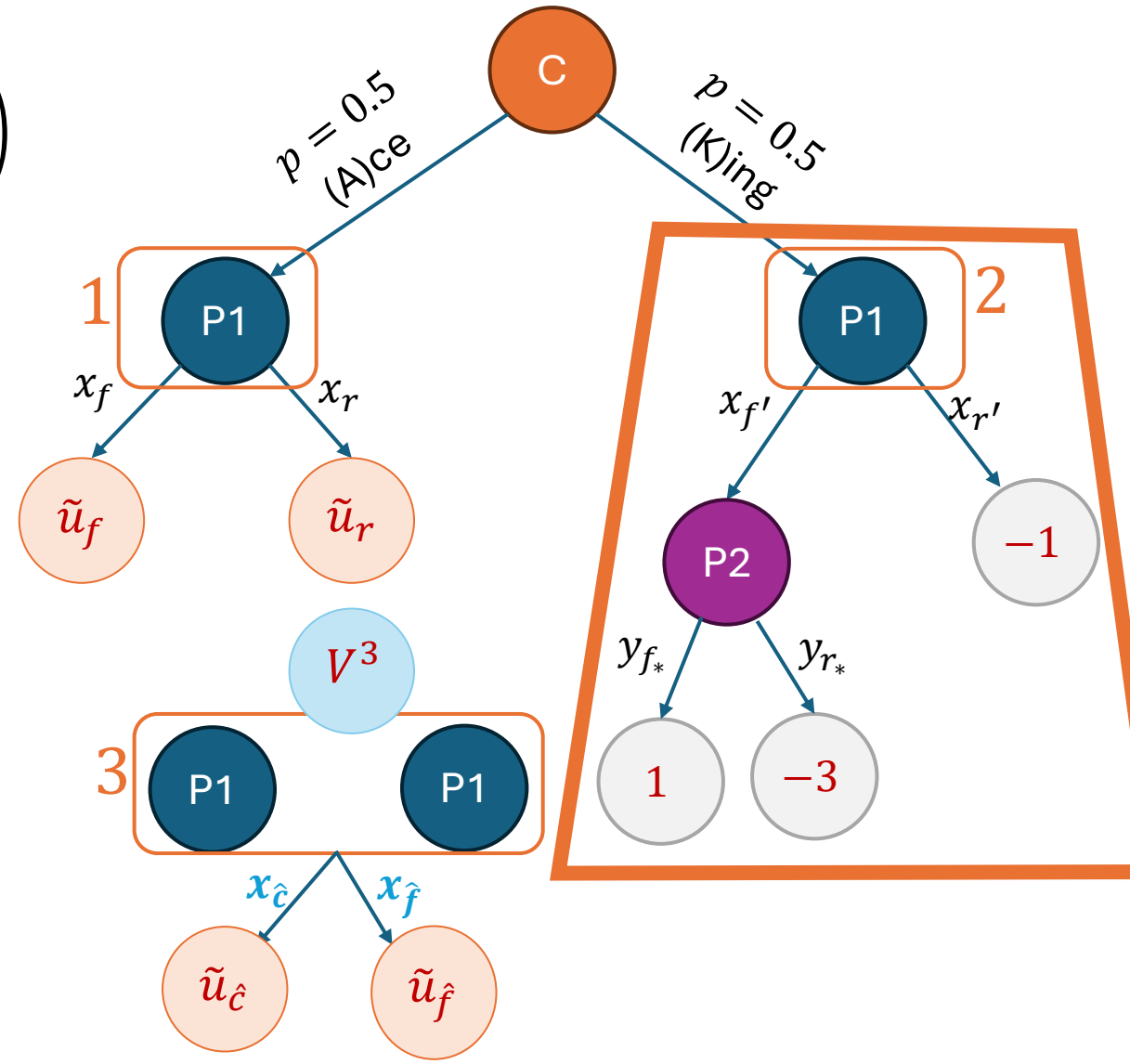


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$(\tilde{u}_{\hat{c}}, \tilde{u}_{\hat{f}}) = \left(-3\frac{1}{2}y_{f_*} + 3\frac{1}{2}y_{r_*}, -2\frac{1}{2}y_{f_*} - 2\frac{1}{2}y_{r_*} \right)$$

$$V^3 \leftarrow x_{\hat{c}}\tilde{u}_{\hat{c}} + x_{\hat{f}}\tilde{u}_{\hat{f}}$$

- Go to **InfoSet 1**

$$(\tilde{u}_f, \tilde{u}_r) = \left(-1\frac{1}{2}, V^3 \right)$$

- Go to **InfoSet 2**

$$(\tilde{u}_{f'}, \tilde{u}_{r'}) = \left(1\frac{1}{2}y_{f_*} - 3\frac{1}{2}y_{r_*}, -1\frac{1}{2} \right)$$

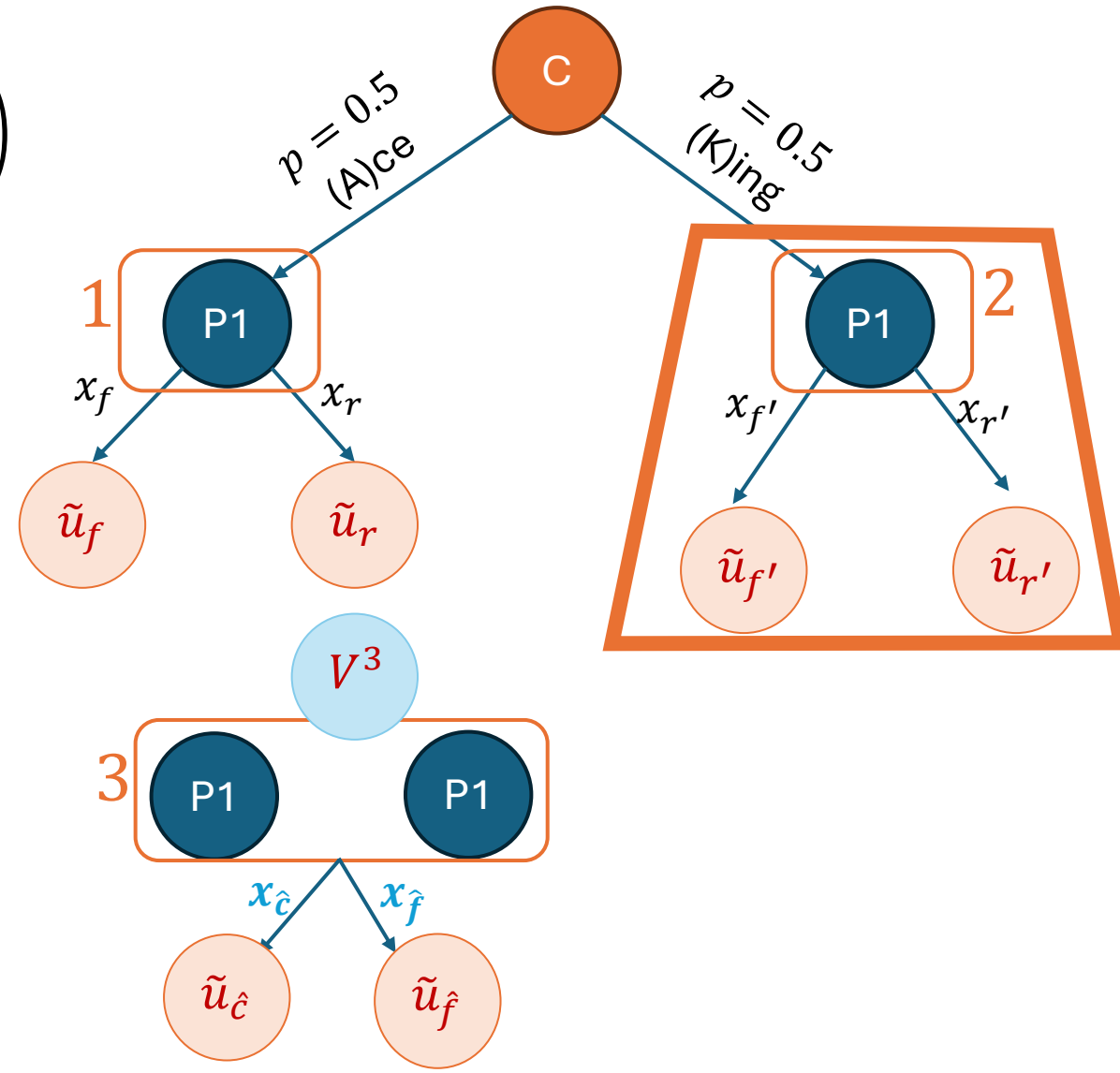


Illustration: First Step of Dynamics

- Go to **InfoSet 3**

$$(\tilde{u}_{\hat{c}}, \tilde{u}_{\hat{f}}) = \left(-3\frac{1}{2}y_{f*} + 3\frac{1}{2}y_{r*}, -2\frac{1}{2}y_{f*} - 2\frac{1}{2}y_{r*} \right)$$

$$V^3 \leftarrow x_{\hat{c}}\tilde{u}_{\hat{c}} + x_{\hat{f}}\tilde{u}_{\hat{f}}$$

- Go to **InfoSet 1**

$$(\tilde{u}_f, \tilde{u}_r) = \left(-1\frac{1}{2}, V^3 \right)$$

- Go to **InfoSet 2**

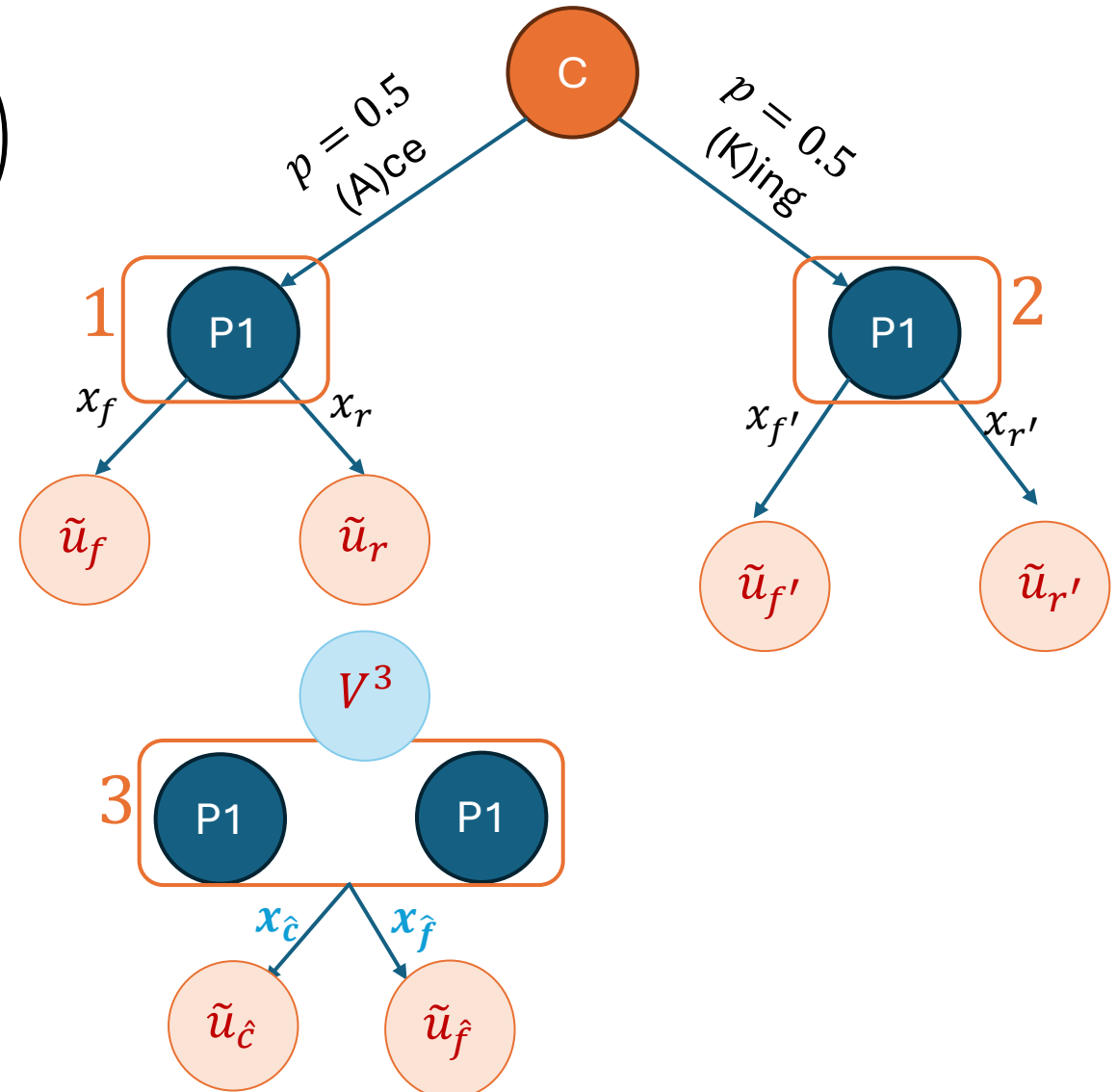
$$(\tilde{u}_{f'}, \tilde{u}_{r'}) = \left(1\frac{1}{2}y_{f*} - 3\frac{1}{2}y_{r*}, -1\frac{1}{2} \right)$$

- Update probabilities**

$$(x_f, x_r) \leftarrow \text{Update}(\tilde{u}_f, \tilde{u}_r)$$

$$(x_{f'}, x_{r'}) \leftarrow \text{Update}(\tilde{u}_{f'}, \tilde{u}_{r'})$$

$$(x_{\hat{c}}, x_{\hat{f}}) \leftarrow \text{Update}(\tilde{u}_{\hat{c}}, \tilde{u}_{\hat{f}})$$



Recursive Algorithm

Value(ActionHistory h , AccOtherProb π_{-1})

Let I be info set corresponding to h

if I is terminal node z return $\pi_{-1} \cdot u(z)$

if Player(I) = chance

Return $\sum_{a \in A_I} \text{Value}(ha, \pi_{-1} \pi_a^C)$

if Player(I) = 2

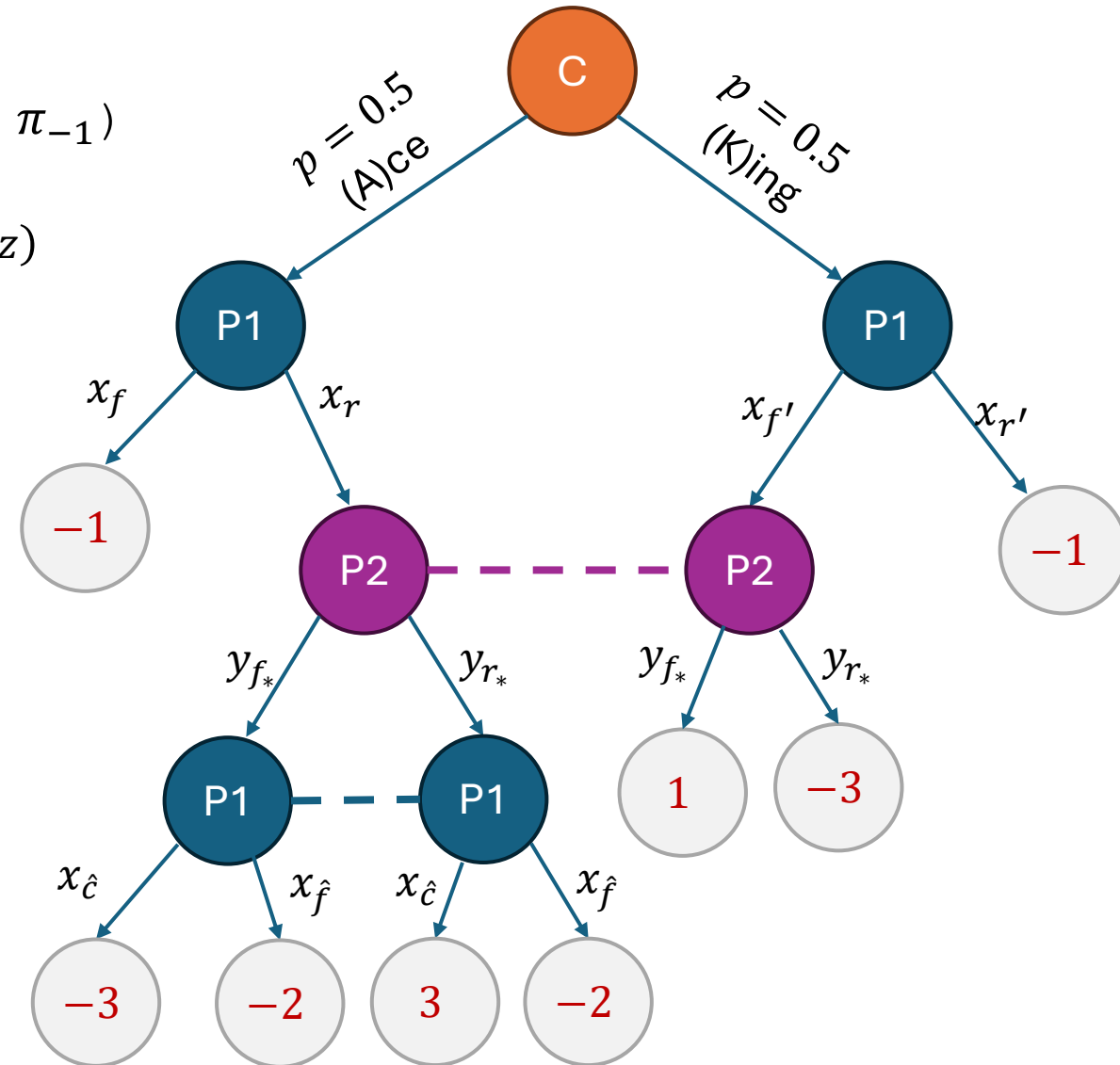
Return $\sum_{a \in A_I} \text{Value}(ha, \pi_{-1} y_a)$

if Player(I) = 1

For $a \in A_I$: $\tilde{u}_a += \text{Value}(ha, \pi_{-1})$

Return $\sum_{a \in A_I} x_a \cdot \text{Value}(ha, \pi_{-1})$

Value(\emptyset , 1)



Recursive Algorithm

Value(ActionHistory h , AccOtherProb π_{-1})

Let I be info set corresponding to h

if I is terminal node z return $\pi_{-1} \cdot u(z)$

if Player(I) = chance

Return $\sum_{a \in A_I} \text{Value}(ha, \pi_{-1} \pi_a^C)$

if Player(I) = 2

Return $\sum_{a \in A_I} \text{Value}(ha, \pi_{-1} y_a)$

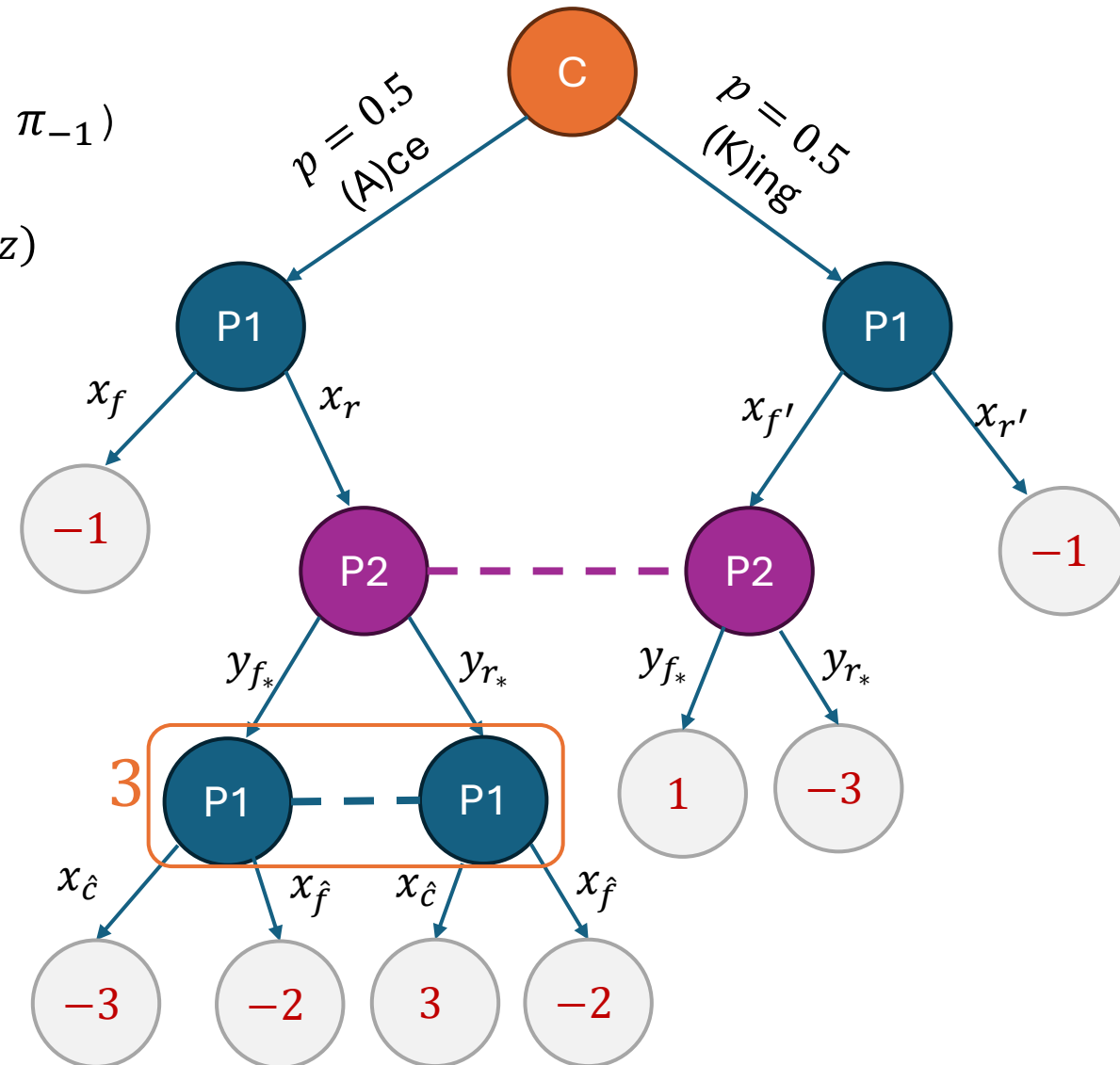
if Player(I) = 1

For $a \in A_I$: $\tilde{u}_a += \text{Value}(ha, \pi_{-1})$

Return $\sum_{a \in A_I} x_a \cdot \text{Value}(ha, \pi_{-1})$

We arrive at the same info set I multiple times, once for each node in the set; \tilde{u}_a accumulates continuation utility from taking action a from all these possible “arrival paths”.

Example. In info set 3 we arrive once on the left node and add $-3 \frac{1}{2} y_{f*}$ and once on the right node and add $3 \frac{1}{2} y_{r*}$ to $u_{\hat{c}}$



Recursive Algorithm

Value(ActionHistory h , AccOtherProb π_{-1})

Let I be info set corresponding to h

if I is terminal node z return $\pi_{-1} \cdot u(z)$

if Player(I) = chance

Return $\sum_{a \in A_I} \text{Value}(ha, \pi_{-1} \pi_a^C)$

if Player(I) = 2

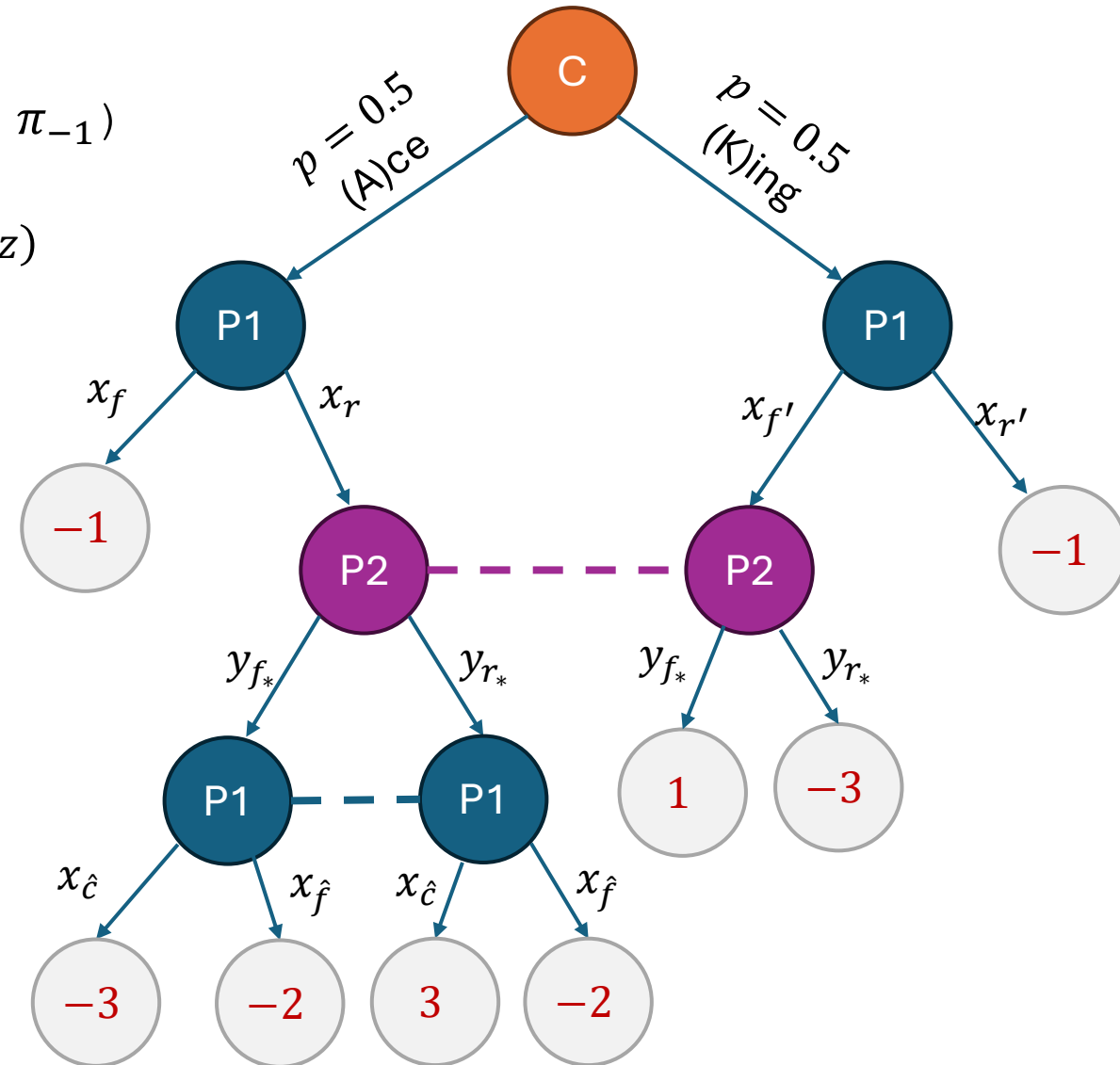
Return $\sum_{a \in A_I} \text{Value}(ha, \pi_{-1} y_a)$

if Player(I) = 1

For $a \in A_I$: $\tilde{u}_a += \text{Value}(ha, \pi_{-1})$

Return $\sum_{a \in A_I} x_a \cdot \text{Value}(ha, \pi_{-1})$

Value(\emptyset , 1)



Equivalent Recursive Algorithm

$\text{CValue}(\text{ActionHistory } h, \text{AccOtherProb } \pi_{-1})$

Let I be info set corresponding to h

if I is terminal node z return ~~π_{-1}~~ $\cdot u(z)$

if $\text{Player}(I) = \text{chance}$

Return $\sum_{a \in A_I} \pi_a^C \cdot \text{CValue}(ha, \pi_{-1} \pi_a^C)$

if $\text{Player}(I) = 2$

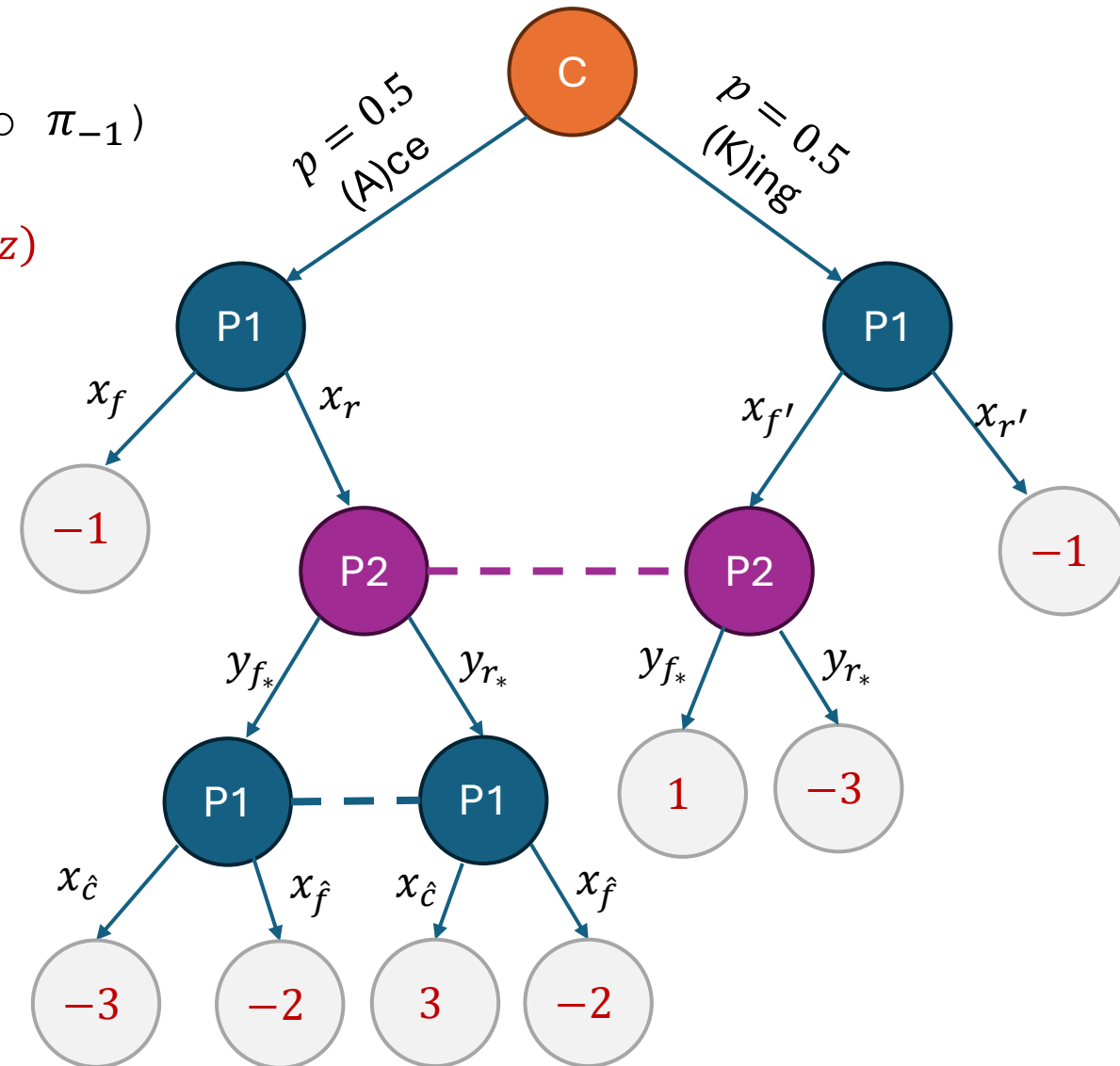
Return $\sum_{a \in A_I} y_a \cdot \text{CValue}(ha, \pi_{-1} y_a)$

if $\text{Player}(I) = 1$

For $a \in A_I$: $\tilde{u}_a += \pi_{-1} \cdot \text{CValue}(ha, \pi_{-1})$

Return $\sum_{a \in A_I} x_a \cdot \text{CValue}(ha, \pi_{-1})$

$\text{CValue}(\emptyset, 1)$



The Typical CRM Algorithm Implementation

$\text{CValue}(\text{ActionHistory } h, \text{AccOtherProb } \pi_{-1})$

Let I be info set corresponding to h

if I is terminal node z return $u(z)$

if $\text{Player}(I) = \text{chance}$

Return $\sum_{a \in A_I} \pi_a^C \cdot \text{CValue}(ha, \pi_{-1} \pi_a^C)$

if $\text{Player}(I) = 2$

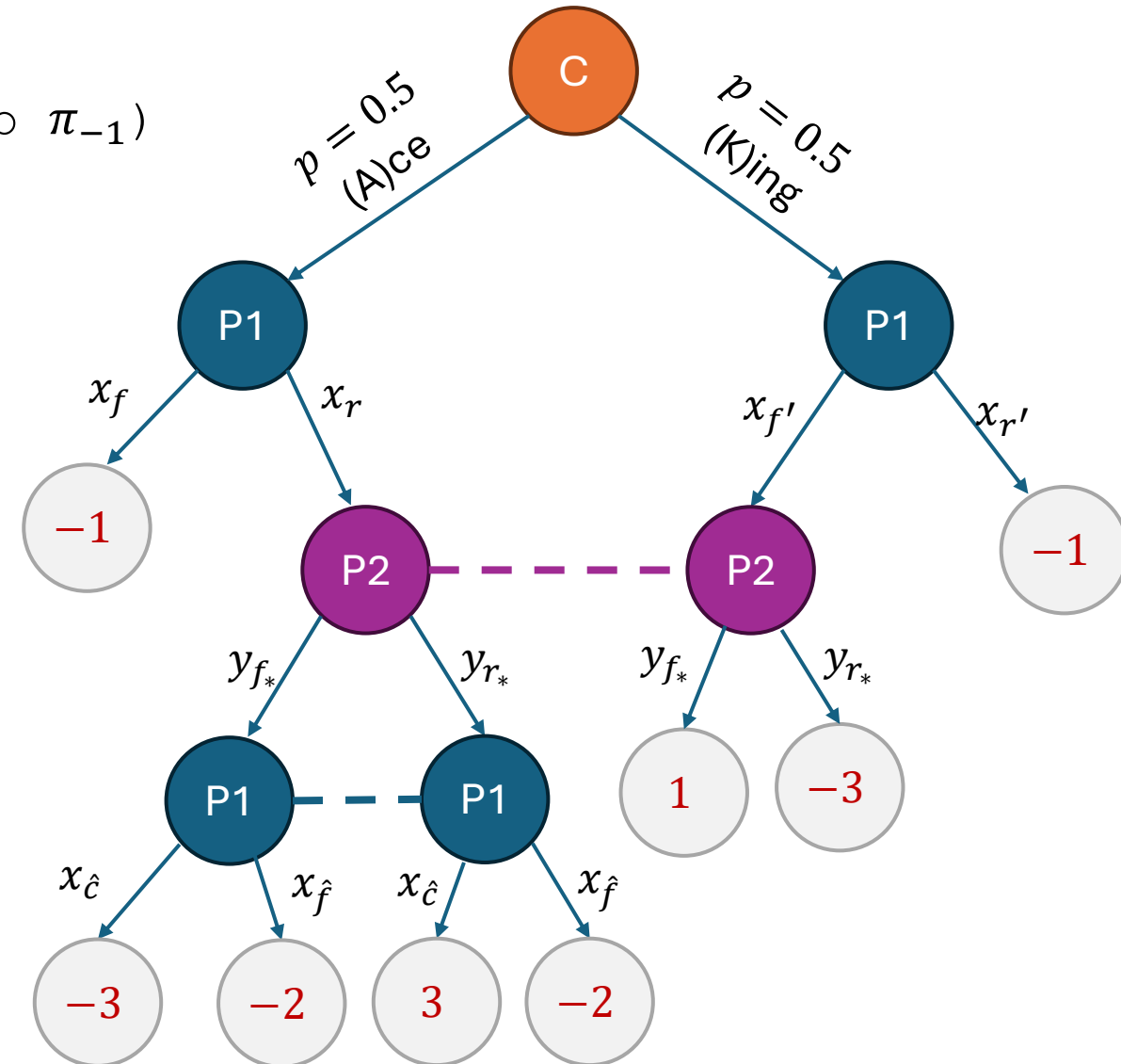
Return $\sum_{a \in A_I} \gamma_a \cdot \text{CValue}(ha, \pi_{-1} \gamma_a)$

if $\text{Player}(I) = 1$

For $a \in A_I$: $\tilde{u}_a += \pi_{-1} \cdot \text{CValue}(ha, \pi_{-1})$

Return $\sum_{a \in A_I} x_a \cdot \text{CValue}(ha, \pi_{-1})$

$\text{CValue}(\emptyset, 1)$



Recovering Equilibrium from CRM Dynamics

We have run CRM dynamics generating behavioral strategies x_t, y_t for T periods.

How do we calculate the behavioral strategies x^*, y^* that are an approximate Nash equilibrium?

Recovering Nash Equilibrium

- We need to translate the behavioral strategies into sequence-form

$$\forall a \in A_j: \tilde{x}_{t,a} = \tilde{x}_{t,p_j} \cdot x_t$$

- Then average the sequence-form strategies

$$\bar{\tilde{x}} = \frac{1}{T} \sum_{t=1}^T \tilde{x}_t$$

Product of probabilities of actions of player P1 on path to info set of action i

- Then translate back to equilibrium behavioral strategies x^*

$$\forall a \in A_j: x_a^* = \frac{\bar{\tilde{x}}_a}{\bar{\tilde{x}}_{p_j}}$$

Recovering Nash Equilibrium

- We need to translate the behavioral strategies into sequence-form

$$\forall a \in A_j: \tilde{x}_{t,a} = \tilde{x}_{t,p_j} \cdot x_t$$

- Then average the sequence-form strategies

$$\bar{\tilde{x}} = \frac{1}{T} \sum_{t=1}^T \tilde{x}_t = \frac{1}{T} \sum_{t=1}^T \tilde{x}_{t,p_j} \cdot x_t$$

Product of probabilities of actions of player P1 on path to info set of action i

- Then translate back to equilibrium behavioral strategies x^*

$$\forall a \in A_j: x_a^* = \frac{\bar{\tilde{x}}_a}{\bar{\tilde{x}}_{p_j}} = \frac{\sum_{t=1}^T \tilde{x}_{t,p_j} \cdot x_{t,a}}{\sum_{t=1}^T \tilde{x}_{t,p_j}}$$

The Typical CRM Algorithm Implementation

```
CValue(ActionHistory  $h$ , AccOtherProb  $\pi_{-1}$ , AccProb  $\pi_1$ )
```

```
Let  $I$  be info set corresponding to  $h$ 
```

```
if  $I$  is terminal node  $z$  return  $u(z)$ 
```

```
if Player( $I$ ) = chance
```

```
    Return  $\sum_{a \in A_I} \pi_a^c \cdot \text{CValue}(ha, \pi_{-1} \pi_a^c, \pi_1)$ 
```

```
if Player( $I$ ) = 2
```

```
    Return  $\sum_{a \in A_I} y_a \cdot \text{CValue}(ha, \pi_{-1} y_a, \pi_1)$ 
```

```
if Player( $I$ ) = 1
```

```
    For  $a \in A_I$ :  $\tilde{u}_a += \pi_{-1} \cdot \text{CValue}(ha, \pi_{-1}, \pi_1 x_a)$ 
```

```
    { Set  $q(I) = \pi_1$ 
```

```
    Return  $\sum_{a \in A_I} x_a \cdot \text{CValue}(ha, \pi_{-1}, \pi_1 x_a)$ 
```

```
CValue( $\emptyset$ , 1)
```

This is the product of the probabilities of prior actions of player $P1$ before arriving at info set I

Note. Due to perfect recall this product is the same every time we visit the info set; irrespective of which node of the info set we arrived at.

The Typical CRM Algorithm Implementation

```
CValue(ActionHistory  $h$ , AccOtherProb  $\pi_{-1}$ , AccProb  $\pi_1$ )
```

```
Let  $I$  be info set corresponding to  $h$ 
```

```
if  $I$  is terminal node  $z$  return  $u(z)$ 
```

```
if Player( $I$ ) = chance
```

```
    Return  $\sum_{a \in A_I} \pi_a^c \cdot \text{CValue}(ha, \pi_{-1} \pi_a^c, \pi_1)$ 
```

```
if Player( $I$ ) = 2
```

```
    Return  $\sum_{a \in A_I} y_a \cdot \text{CValue}(ha, \pi_{-1} y_a, \pi_1)$ 
```

```
if Player( $I$ ) = 1
```

```
    For  $a \in A_I$ :  $\tilde{u}_a += \pi_{-1} \cdot \text{CValue}(ha, \pi_{-1}, \pi_1 x_a)$ 
```

```
    Set  $q(I) = \pi_1$ 
```

```
    Return  $\sum_{a \in A_I} x_a \cdot \text{CValue}(ha, \pi_{-1}, \pi_1 x_a)$ 
```

```
CValue( $\emptyset$ , 1)
```


The Overall Equilibrium Algorithm with CRM

After each period $t \in \{1, \dots, T\}$:

- With last period behavior strategies x_t, y_t call CValue($\emptyset, 1, 1$)
- Store $\tilde{u}_{t,a}$ and $q_t(I)$ for each action a and info set I of P1
- Symmetrically, do so for player P2
- Update strategies at all information sets
$$\forall j \in \mathcal{J}_1: x_{t+1}^j \leftarrow \text{Update}(\tilde{u}_t^j), \quad \forall j \in \mathcal{J}_2: y_{t+1}^j \leftarrow \text{Update}(\tilde{u}_t^j)$$

At the end:

$$\forall I \in \mathcal{J}_1 \forall a \in A_I: x_a^* = \frac{\sum_t q_t(I) x_{t,a}}{\sum_t q_t(I)}$$
$$\forall I \in \mathcal{J}_2 \forall a \in A_I: y_a^* = \frac{\sum_t q_t(I) y_{t,a}}{\sum_t q_t(I)}$$

Approximate Equilibrium in Behavioral Strategies

What algorithm to use for local regret updates?

The Overall Equilibrium A

Any no-regret algorithm for the n -action no-regret problem can be used, e.g. FTRL, OFTRL, EXP, etc.

What performs well in practice is what is known as **Regret Matching!**

After each period $t \in \{1, \dots, T\}$:

- With last period behavior strategies x_t, y_t call CValue($\emptyset, 1, 1$)
- Store $\tilde{u}_{t,a}$ and $q_t(I)$ for each action a and info set I of P1
- Symmetrically, do so for player P2
- Update strategies at all information sets

$$\forall j \in \mathcal{J}_1: x_{t+1}^j \leftarrow \text{Update}(\tilde{u}_t^j), \quad \forall j \in \mathcal{J}_2: y_{t+1}^j \leftarrow \text{Update}(\tilde{u}_t^j)$$

At the end:

$$\forall I \in \mathcal{J}_1 \forall a \in A_I: x_a^* = \frac{\sum_t q_t(I) x_{t,a}}{\sum_t q_t(I)}$$

$$\forall I \in \mathcal{J}_2 \forall a \in A_I: y_a^* = \frac{\sum_t q_t(I) y_{t,a}}{\sum_t q_t(I)}$$

Approximate Equilibrium in Behavioral Strategies

Regret Matching and Regret Matching+

- Consider the n action no-regret learning setting; at each period we choose $x_t \in \Delta(n)$, observe utility vector u_t and get utility $\langle x_t, u_t \rangle$
- At each period t calculate regret of not playing action a

$$r_{t,a} = u_{t,a} - \langle u_t, x_t \rangle$$

- Calculate cumulative regret of not playing action a

$$R_{t,a} = \sum_{\tau \leq t} r_{\tau,a} = R_{t-1,a} + r_{t,a}$$

- Choose next distribution, proportional to positive part of regret

$$x_{t+1,a} \propto [R_{t,a}]^+ := \max\{R_{t,a}, 0\}$$

- People typically refer to CFR with RegretMatching as simply “CFR”

Regret Matching+

- Consider the n action no-regret learning setting; at each period we choose $x_t \in \Delta(n)$, observe utility vector u_t and get utility $\langle x_t, u_t \rangle$
- At each period t calculate regret of not playing action a

$$r_{t,a} = u_{t,a} - \langle u_t, x_t \rangle$$

- Continuously clip above zero, as you accumulate regret of a

$$R_{t,a} = [R_{t-1,a} + r_{t,a}]^+$$

- Choose next distribution, proportional to $R_{t,a}$

$$x_{t+1,a} \propto R_{t,a}$$

- Regret Matching and Regret Matching+ achieve $\text{Regret} \leq \sqrt{n/T}$

Extra Tricks for Empirical Improvement

Monte-Carlo Stochastic Approximation of Utilities

- Calculating utilities on all nodes of the tree can be very expensive
- In linear online learning it suffices that we use an unbiased estimate of the utility vector

$$\tilde{x}_t = \operatorname{argmax}_{x \in X} \sum_{\tau < t} \langle x, \hat{u}_\tau \rangle - \frac{1}{\eta} \mathcal{R}(x), \quad E[\hat{u}_\tau \mid F_\tau] = u_\tau$$

All random variables observed before period τ

- By standard martingale concentration inequality arguments, the error vanishes with the number of iterations (*we will see later*)
- In this setting, it suffices that we “sample a path for opponent” and that we “sample chance moves”

Monte-Carlo Stochastic Approximation of Utilities

- Sample chance move (e.g. sampled A)

- Go to **InfoSet 1**

$$\hat{u}_f = -1, \quad \hat{u}_r = 0$$

- Go down tree the r path

- Sample P2 move (e.g. sampled f_*)

- Go down to **InfoSet 3**

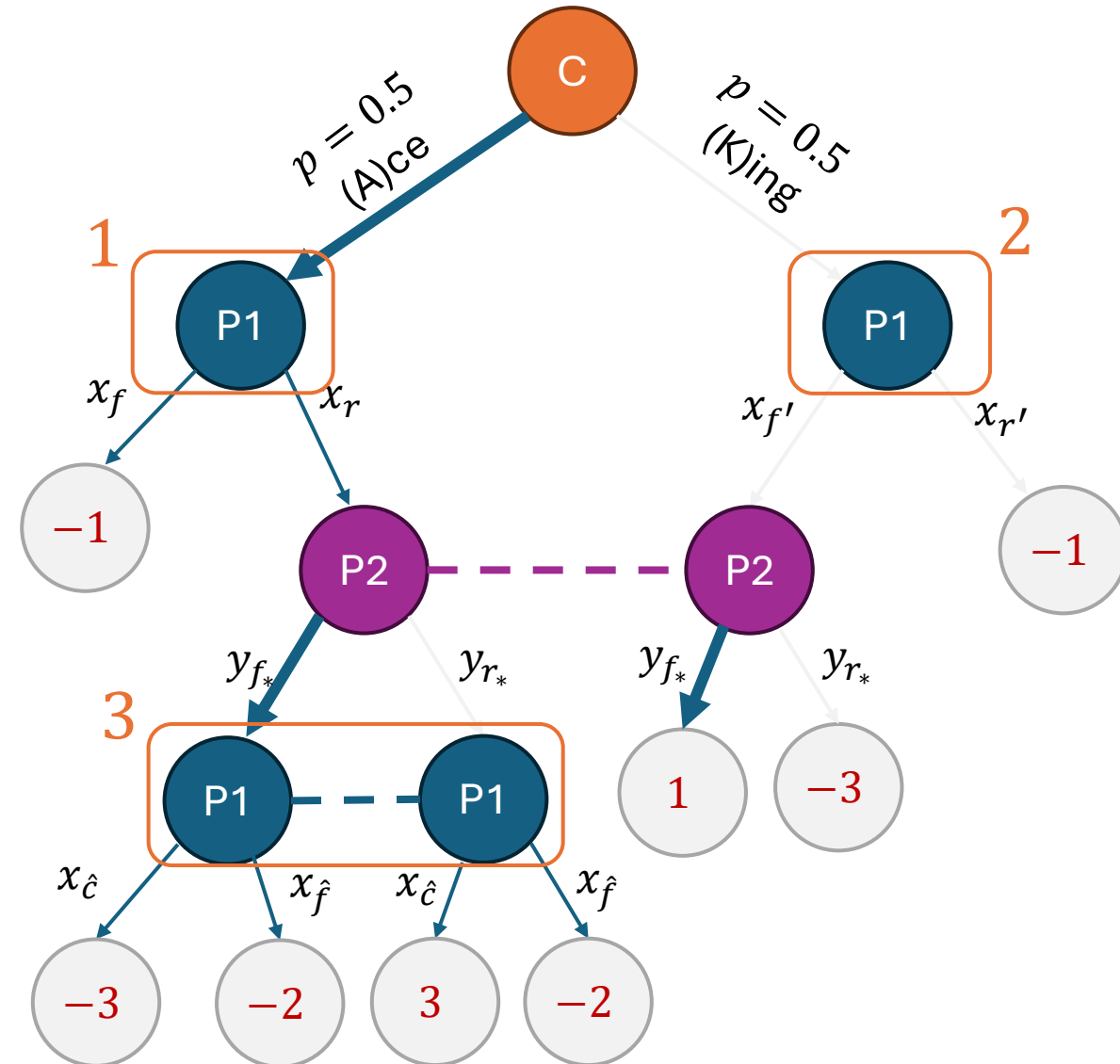
$$\hat{u}_{\hat{c}} = -3, \quad \hat{u}_{\hat{f}} = -1$$

$$\hat{u}_r += x_{\hat{c}} \hat{u}_{\hat{c}} + x_{\hat{f}} \hat{u}_{\hat{f}}$$

- Update probabilities of visited infosets

$$(x_f, x_r) \leftarrow \text{Update}(\hat{u}_f, \hat{u}_r)$$

$$(x_{\hat{c}}, x_{\hat{f}}) \leftarrow \text{Update}(\hat{u}_{\hat{c}}, \hat{u}_{\hat{f}})$$



Typical Monte Carlo Algorithm Implementation

```
MCCValue(ActionHistory  $h$ , AccProb  $\pi_1$ )
```

```
  Let  $I$  be info set corresponding to  $h$ 
```

```
  if  $I$  is terminal node  $z$  return  $u(z)$ 
```

```
  if Player( $I$ ) = chance
```

```
    Sample  $a \sim \pi^C$ 
```

```
    Return MCCValue( $ha, \pi_1$ )
```

```
  if Player( $I$ ) = 2
```

```
    Sample  $a \sim y^I$ 
```

```
    Return MCCValue( $ha, \pi_1$ )
```

```
  if Player( $I$ ) = 1
```

```
    For  $a \in A_I$ :  $\tilde{u}_a += \text{MCCValue}(ha, \pi_1 \cdot x_a)$ 
```

```
    Set  $q(I) = \pi_1$ 
```

```
    Return  $\sum_{a \in A_I} x_a \cdot \text{MCCValue}(ha, \pi_1 \cdot x_a)$ 
```

```
Value( $\emptyset$ , 1)
```

Can Combine with Update Step in One Pass

```
MCCValue(ActionHistory  $h$ , AccProb  $\pi_1$ )
```

```
  Let  $I$  be info set corresponding to  $h$ 
```

```
  if  $I$  is terminal node  $z$  return  $u(z)$ 
```

```
  if Player( $I$ ) = chance
```

```
    Sample  $a \sim \pi^C$ 
```

```
    Return MCCValue( $ha, \pi_1$ )
```

```
  if Player( $I$ ) = 2
```

```
    Sample  $a \sim y^I$ 
```

```
    Return MCCValue( $ha, \pi_1$ )
```

```
  if Player( $I$ ) = 1
```

```
    For  $a \in A_I$ :  $\tilde{u}_a += \text{MCCValue}(ha, \pi_1 \cdot x_a)$ 
```

```
    Set  $q(I) = \pi_1$ 
```

```
    Update  $x_{\text{next}}^I \leftarrow \text{Update}(\tilde{u}^I)$ 
```

```
    Return  $\sum_{a \in A_I} x_a \cdot \text{MCCValue}(ha, \pi_1 \cdot x_a)$ 
```

Alternation

After each period t :

- If t is odd then update the strategy of the x -player
- If t is even then update strategy of the y -player

For most natural algorithms, alternation can only help in terms of reducing the violation of best response constraints!

Can converge faster to equilibrium

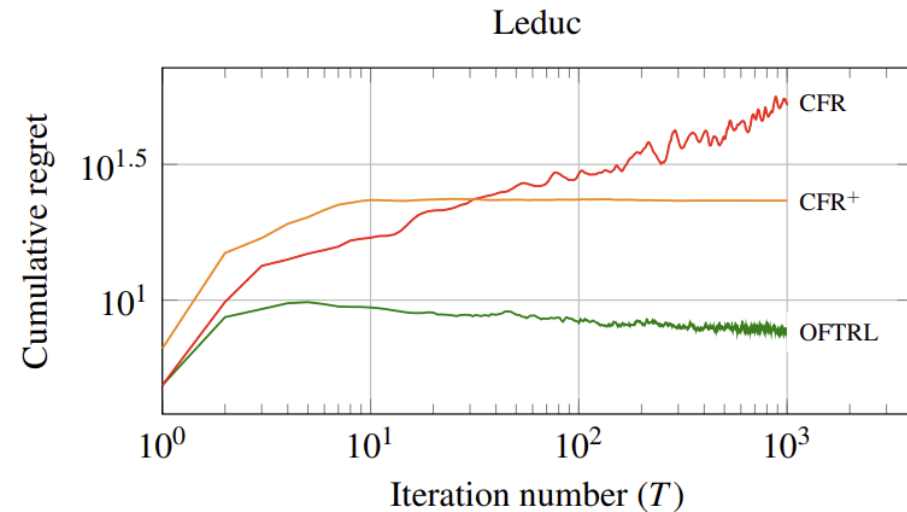
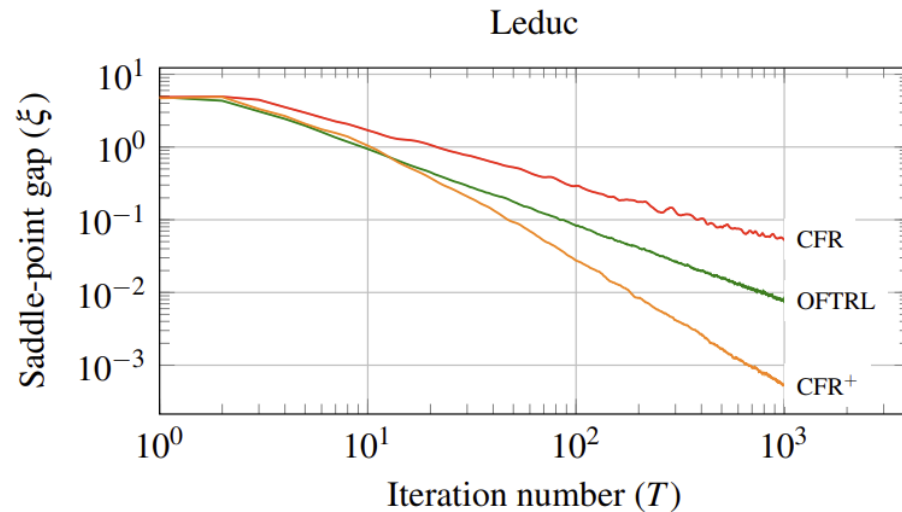
Weighted Averaging

- Instead of uniformly weighting all rounds, put more weight on more recent rounds of play

$$\frac{1}{\sum_t t^\alpha} \sum_t t^\alpha \tilde{x}_t$$

- Typically, one uses linear averaging (i.e., $\alpha = 1$)
- The CFR algorithm that uses RegretMatching+, alternation and linear averaging is typically referred to as “CFR+”

Empirical Comparisons



Violations of best response

$$\text{Regret}_y(x_*, y_*) + \text{Regret}_x(x_*, y_*) := \max_y x_*^\top A y - x_*^\top A y_* + x_*^\top A y_* - \min_x x^\top A y_* = \max_y x_*^\top A y - \min_x x^\top A y_*$$

saddle-point gap

$$[R_y + R_x] = \max_y \bar{x}^\top A y - \frac{1}{T} \sum_t x_t^\top A y_t + \frac{1}{T} \sum_t x_t^\top A y_t - \min_x x^\top A \bar{y} = \max_y \bar{x}^\top A y - \min_x x^\top A \bar{y}$$

Sum of learning
algorithm regrets

saddle-point gap of
average strategies \bar{x}, \bar{y}

Elements of the Libratus AI

- The first agent to achieve superhuman performance in two player No-Limit Texas Hold'em poker (10^{161} decision points)
- Prior best was Limit Texas Hold'em (10^{13} decision points); solution is basically “run CFR+”
- For No-Limit Texas Hold'em game is too big for this approach!

Elements of Libratus AI

