Deep Learning in Tree-Based Game Solving 1

Stephen McAleer

Outline of the next few lectures

- Deep learning in tree-based game solving 1
  - Deep learning recap
  - NFSP
  - Deep CFR
  - Policy gradient methods
- Deep learning in tree-based game solving 2
  - MCCFR
  - DREAM
  - ESCHER
  - NeuRD
- Deep learning in tree-based game solving 3
  - DeepNash for expert-level Stratego
- Deep learning in tree-based game solving 4
  - AlphaStar and OpenAI 5 for SOTA in video games
  - Double Oracle brief intro
- SOTA in double oracle algorithms
  - PSRO
  - XDO
  - SP-PSRO

A Taxonomy of Game-Theoretic RL

- Counterfactual Regret Minimization (Zinkevich et al. 2007)
    - CFR: Zinkevich et al. 2007
    - MC-CFR: Lanctot et al. 2009
    - **Deep CFR: Brown et al. 2019**
    - DREAM: Steinberger et al. 2020
    - ESCHER: McAleer et al. 2022
- Policy Gradients
    - **Regret Policy Gradient (Srinivasan et al. 2018)**
    - OpenAI Five (OpenAI 2019)
    - Neural Replicator Dynamics (Hennes, Morrill, and Omidshafiei et al. 2020)
    - Actor Critic Hedge (Fu et al. 2022)
    - DeepNash for expert-level Stratego (Perolat, de Vylder, and Tuyls et al. 2022)
    - Magnetic Mirror Descent (Sokota et al. 2022)
- PSRO (McMahan et. al. 2003, Lanctot et al. 2017)
    - AlphaStar for expert-level Starcraft (Vinyals et al. 2019)
    - Pipeline PSRO (McAleer and Lanier et al. 2020)
    - α-PSRO (Muller et al. 2020)
    - XDO (McAleer et al. 2021)
    - Joint-PSRO (Marris et al. 2021)
    - Anytime PSRO (McAleer et al. 2022)
    - Self-Play PSRO (McAleer et al. 2022)
- **Neural Fictitious Self Play (Heinrich and Silver 2016)**

Lecture 1 (This Lecture)

A Taxonomy of Game-Theoretic RL

- Counterfactual Regret Minimization (Zinkevich et al. 2007)
    - CFR: Zinkevich et al. 2007
    - **MC-CFR: Lanctot et al. 2009**
    - Deep CFR: Brown et al. 2019
    - **DREAM: Steinberger et al. 2020**
    - **ESCHER: McAleer et al. 2022**
- Policy Gradients
    - Regret Policy Gradient (Srinivasan et al. 2018)
    - OpenAI Five (OpenAI 2019)
    - **Neural Replicator Dynamics (Hennes, Morrill, and Omidshafiei et al. 2020)**
    - Actor Critic Hedge (Fu et al. 2022)
    - DeepNash for expert-level Stratego (Perolat, de Vylder, and Tuyls et al. 2022)
    - **Magnetic Mirror Descent (Sokota et al. 2022)**
- PSRO (McMahan et. al. 2003, Lanctot et al. 2017)
    - AlphaStar for expert-level Starcraft (Vinyals et al. 2019)
    - Pipeline PSRO (McAleer and Lanier et al. 2020)
    - α-PSRO (Muller et al. 2020)
    - XDO (McAleer et al. 2021)
    - Joint-PSRO (Marris et al. 2021)
    - Anytime PSRO (McAleer et al. 2022)
    - Self-Play PSRO (McAleer et al. 2022)
- Neural Fictitious Self Play (Heinrich and Silver 2016)

Lecture 2

A Taxonomy of Game-Theoretic RL

Lecture 3

- Counterfactual Regret Minimization (Zinkevich et al. 2007)
    - CFR: Zinkevich et al. 2007
    - MC-CFR: Lanctot et al. 2009
    - Deep CFR: Brown et al. 2019
    - DREAM: Steinberger et al. 2020
    - ESCHER: McAleer et al. 2022
- Policy Gradients
    - Regret Policy Gradient (Srinivasan et al. 2018)
    - OpenAI Five (OpenAI 2019)
    - Neural Replicator Dynamics (Hennes, Morrill, and Omidshafiei et al. 2020)
    - Actor Critic Hedge (Fu et al. 2022)
    - **DeepNash for expert-level Stratego (Perolat, de Vylder, and Tuyls et al. 2022)**
    - Magnetic Mirror Descent (Sokota et al. 2022)
- PSRO (McMahan et. al. 2003, Lanctot et al. 2017)
    - AlphaStar for expert-level Starcraft (Vinyals et al. 2019)
    - Pipeline PSRO (McAleer and Lanier et al. 2020)
    - α-PSRO (Muller et al. 2020)
    - XDO (McAleer et al. 2021)
    - Joint-PSRO (Marris et al. 2021)
    - Anytime PSRO (McAleer et al. 2022)
    - Self-Play PSRO (McAleer et al. 2022)
- Neural Fictitious Self Play (Heinrich and Silver 2016)

A Taxonomy of Game-Theoretic RL

Lecture 4

- Counterfactual Regret Minimization (Zinkevich et al. 2007)
  - CFR: Zinkevich et al. 2007
  - MC-CFR: Lanctot et al. 2009
  - Deep CFR: Brown et al. 2019
  - DREAM: Steinberger et al. 2020
  - ESCHER: McAleer et al. 2022
- Policy Gradients
  - Regret Policy Gradient (Srinivasan et al. 2018)
  - **OpenAI Five (OpenAI 2019)**
  - Neural Replicator Dynamics (Hennes, Morrill, and Omidshafiei et al. 2020)
  - Actor Critic Hedge (Fu et al. 2022)
  - DeepNash for expert-level Stratego (Perolat, de Vylder, and Tuyls et al. 2022)
  - Magnetic Mirror Descent (Sokota et al. 2022)
- PSRO (McMahan et. al. 2003, Lanctot et al. 2017)
  - **AlphaStar for expert-level Starcraft (Vinyals et al. 2019)**
  - Pipeline PSRO (McAleer and Lanier et al. 2020)
  - α-PSRO (Muller et al. 2020)
  - XDO (McAleer et al. 2021)
  - Joint-PSRO (Marris et al. 2021)
  - Anytime PSRO (McAleer et al. 2022)
  - Self-Play PSRO (McAleer et al. 2022)
- Neural Fictitious Self Play (Heinrich and Silver 2016)

A Taxonomy of Game-Theoretic RL

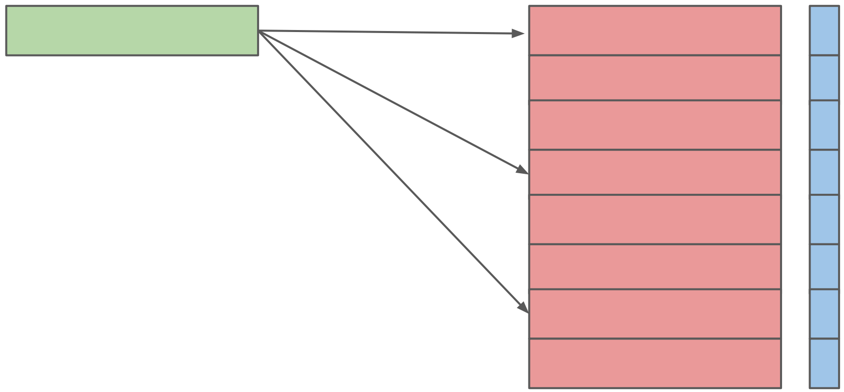
Lecture 5

- Counterfactual Regret Minimization (Zinkevich et al. 2007)
    - CFR: Zinkevich et al. 2007
    - MC-CFR: Lanctot et al. 2009
    - Deep CFR: Brown et al. 2019
    - DREAM: Steinberger et al. 2020
    - ESCHER: McAleer et al. 2022
- Policy Gradients
    - Regret Policy Gradient (Srinivasan et al. 2018)
    - OpenAI Five (OpenAI 2019)
    - Neural Replicator Dynamics (Hennes, Morrill, and Omidshafiei et al. 2020)
    - Actor Critic Hedge (Fu et al. 2022)
    - DeepNash for expert-level Stratego (Perolat, de Vylder, and Tuyls et al. 2022)
    - Magnetic Mirror Descent (Sokota et al. 2022)
- **PSRO (McMahan et. al. 2003, Lanctot et al. 2017)**
    - AlphaStar for expert-level Starcraft (Vinyals et al. 2019)
    - **Pipeline PSRO (McAleer and Lanier et al. 2020)**
    - **α-PSRO (Muller et al. 2020)**
    - **XDO (McAleer et al. 2021)**
    - **Joint-PSRO (Marris et al. 2021)**
    - **Anytime PSRO (McAleer et al. 2022)**
    - **Self-Play PSRO (McAleer et al. 2022)**
- Neural Fictitious Self Play (Heinrich and Silver 2016)

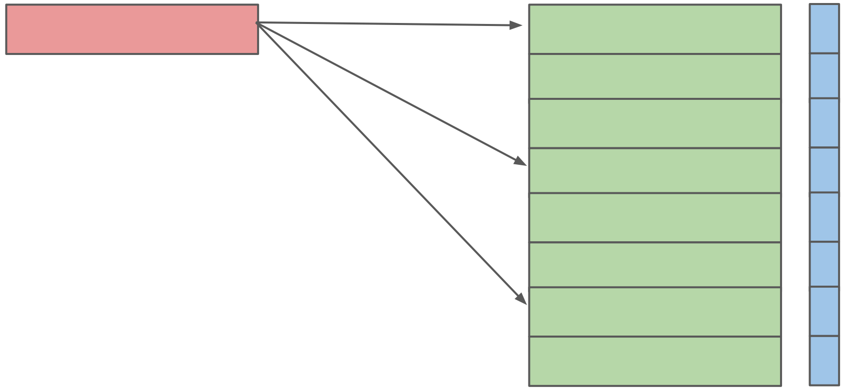
Fictitious Play

- Both players learn best response to opponent's average strategy
- Average strategy converges to a Nash equilibrium

Player 1 Best Responds to Player 2's Average Policy



Player 2 Best Responds to Player 1's Average Policy



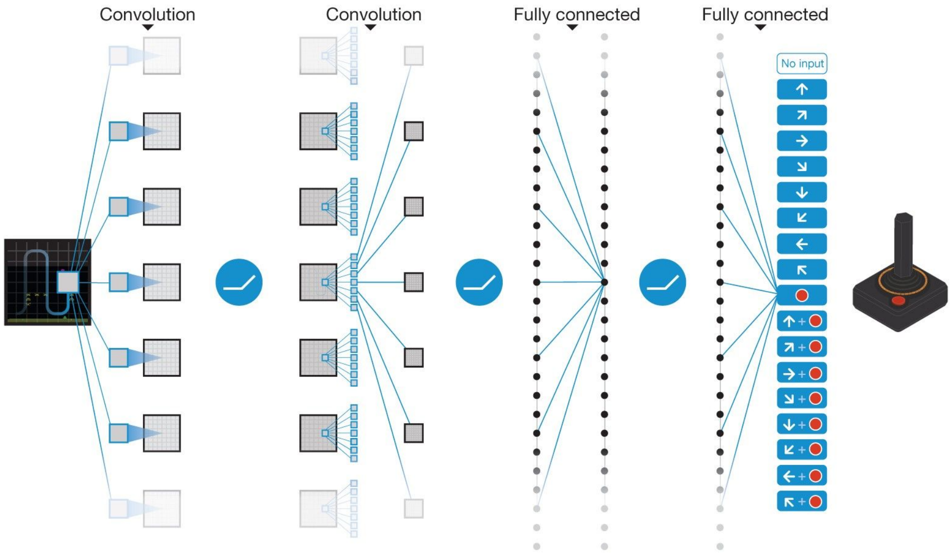
Q-Learning Recap

- Maintain a table of Q-values for each state-action pair
- Iteratively update this table via bootstrapped target until convergence
- Improvement comes from the max operator

$$\underbrace{Q(S_t, A_t)}_{\text{New Q-value estimation}} \leftarrow \underbrace{Q(S_t, A_t)}_{\text{Former Q-value estimation}} + \underbrace{\alpha}_{\text{Learning Rate}} [\underbrace{\underbrace{\underbrace{R_{t+1}}_{\text{Immediate Reward}} + \underbrace{\gamma max_a Q(S_{t+1}, a)}_{\text{Discounted Estimate optimal Q-value of next state}}}_{\text{TD Target}} - \underbrace{Q(S_t, A_t)}_{\text{Former Q-value estimation}}}_{\text{TD Error}}]$$

Deep Q Network (DQN)

- Q network is a neural network trained via gradient descent
- Use TD target to train neural network
- Store experience in replay buffer
- *Off-policy:* Can use arbitrary data



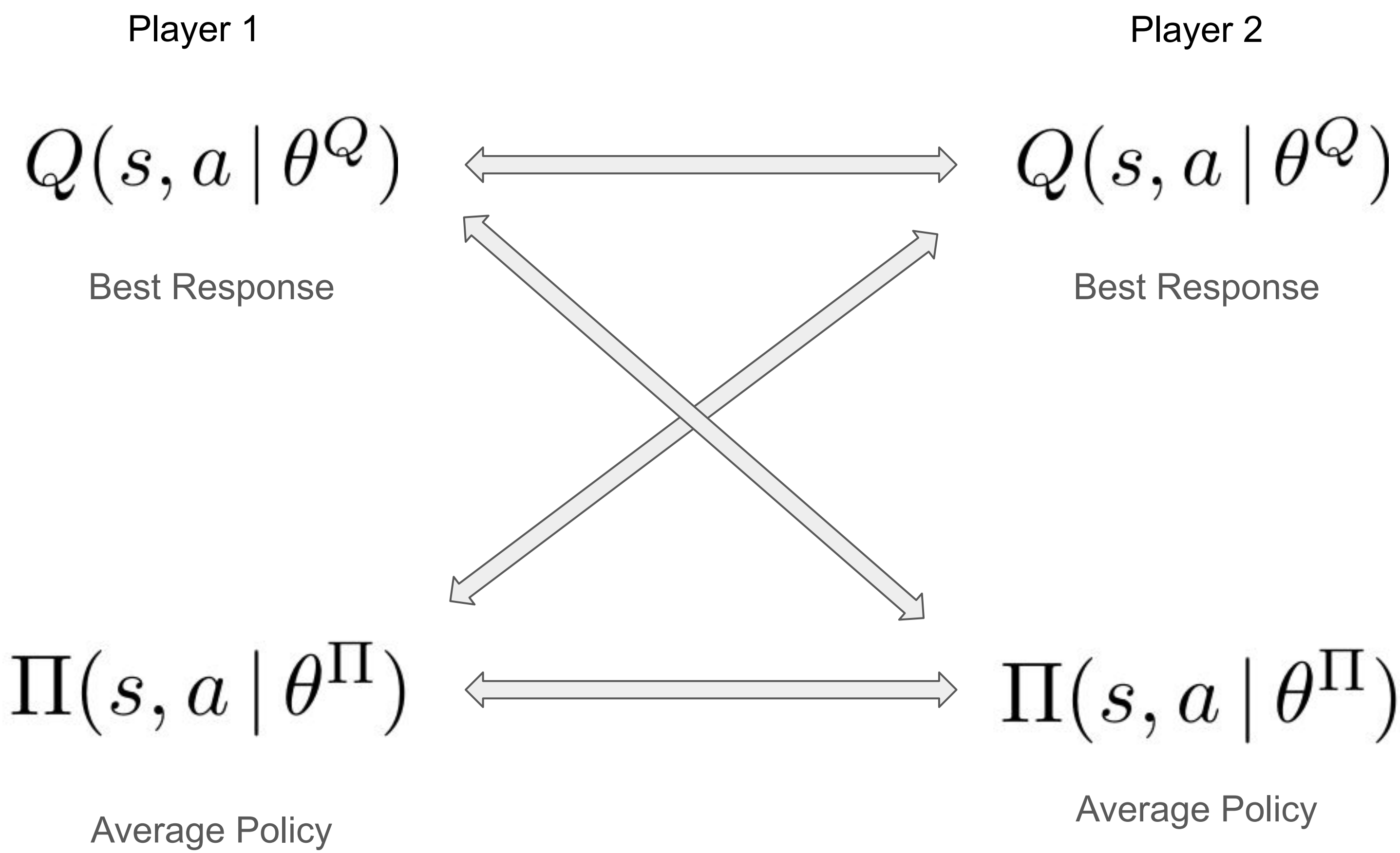
$$\mathcal{L}\left(\theta^{Q}\right)=\mathbb{E}_{(s,a,r,s')\sim\mathcal{M}_{RL}}\left[\left(r+\max_{a'} Q(s',a'\,|\,\theta^{Q'})-Q(s,a\,|\,\theta^{Q})\right)^{2}\right]$$

Neural Fictitious Self Play (NFSP)

with probability η

$$\eta = 0.1$$

with probability $1 - \eta$

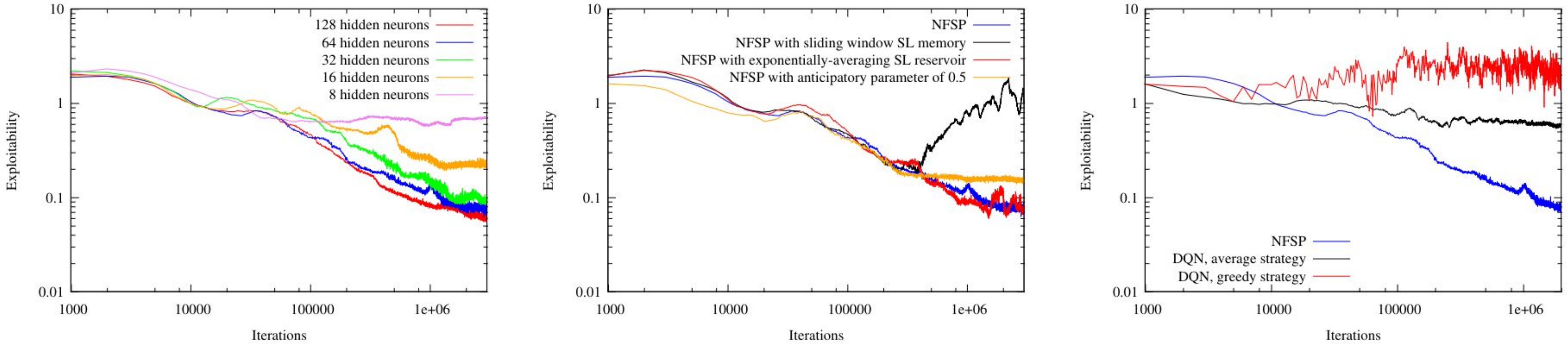


Heinrich and Silver, *Deep Reinforcement Learning from Self-Play in Imperfect-Information Games,* 2016

Algorithm 1 Neural Fictitious Self-Play (NFSP) with fitted Q-learning

Initialize game Γ and execute an agent via RUNAGENT for each player in the game
function RUNAGENT(Γ)
  Initialize replay memories $\mathcal{M}_{RL}$ (circular buffer) and $\mathcal{M}_{SL}$ (reservoir)
  Initialize average-policy network $\Pi(s, a \mid \theta^{\Pi})$ with random parameters θ^{Π}
  Initialize action-value network $Q(s, a \mid \theta^{Q})$ with random parameters θ^{Q}
  Initialize target network parameters $\theta^{Q'} \leftarrow \theta^{Q}$
  Initialize anticipatory parameter η
  for each episode **do**
    Set policy $\sigma \leftarrow \begin{cases} \epsilon\text{-greedy}\,(Q)\,, & \text{with probability } \eta \\ \Pi, & \text{with probability } 1-\eta \end{cases}$
    Observe initial information state s_1 and reward r_1
    for $t = 1, T$ **do**
      Sample action a_t from policy σ
      Execute action a_t in game and observe reward r_{t+1} and next information state s_{t+1}
      Store transition $(s_t, a_t, r_{t+1}, s_{t+1})$ in reinforcement learning memory $\mathcal{M}_{RL}$
      if agent follows best response policy $\sigma = \epsilon\text{-greedy}\,(Q)$ **then**
        Store behaviour tuple (s_t, a_t) in supervised learning memory $\mathcal{M}_{SL}$
      end if
      Update θ^{Π} with stochastic gradient descent on loss
        $\mathcal{L}(\theta^{\Pi}) = \mathbb{E}_{(s,a)\sim\mathcal{M}_{SL}}\left[-\log \Pi(s, a \mid \theta^{\Pi})\right]$
      Update θ^{Q} with stochastic gradient descent on loss
        $\mathcal{L}\left(\theta^{Q}\right) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{M}_{RL}}\left[\left(r + \max_{a'} Q(s', a' \mid \theta^{Q'}) - Q(s, a \mid \theta^{Q})\right)^2\right]$
      Periodically update target network parameters $\theta^{Q'} \leftarrow \theta^{Q}$
    end for
  end for
end function

Results

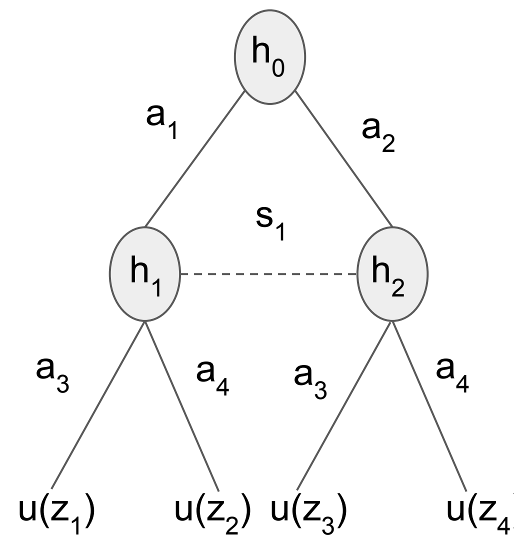


(a) Varying network sizes (b) Removed essential components (c) Comparison to DQN

Figure 1: Learning performance of NFSP in Leduc Hold'em.

Extensive-Form Games

- ***History h*** is ground truth state of the game
  - All cards for all players
- ***Information set s*** is observation for one player
  - Set of histories consistent with observation
  - The hand for one player
- ***Policy $\pi_i(a \mid s)$*** gives distribution over actions at information set s
- ***Reach probability $\eta^{\pi}(h)$*** is joint probability of reaching history h under π
- ***Terminal history z*** is history at end of game
- ***Utility $u_i(z)$*** is utility for player i



CFR Recap

- Independently minimize counterfactual regret at every information set

$$v_i(\pi, h) = \sum_{z \sqsupset h} \eta^\pi(h, z) u_i(z)$$

CFR Recap

- Independently minimize counterfactual regret at every information set

$$v_i(\pi, h) = \sum_{z \sqsupset h} \eta^\pi(h, z) u_i(z)$$

$$v_i^c(\pi, s) = \sum_{h \in s} \eta_{-i}^\pi(h) v_i(\pi, h)$$

CFR Recap

- Independently minimize counterfactual regret at every information set
- Tabular CFR traverses entire tree and updates policy via no-regret at every information set

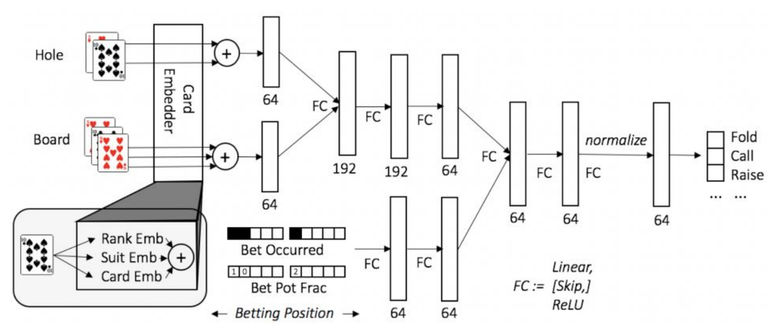
$$v_i(\pi, h) = \sum_{z \sqsupset h} \eta^\pi(h, z) u_i(z)$$

$$v_i^c(\pi, s) = \sum_{h \in s} \eta_{-i}^\pi(h) v_i(\pi, h)$$

$$R_s^T := \max_{\hat{a} \in A_s} \sum_{t=1}^{T} r_i^c(\pi^t, s, a) = \max_{\hat{a} \in A_s} \sum_{t=1}^{T} q_i^c(\pi^t, s, a) - v_i^c(\pi^t, s)$$

Deep CFR

- Estimate counterfactual regret
- Regrets are updated only for the traverser on an iteration.
- At infosets where the traverser acts, all actions are explored. At other infosets and chance nodes, only a single action is explored.
- Add counterfactual regret estimates to replay buffer
- Train neural network to estimate cumulative regret conditioned on information set



Brown, Lerer, Gross, and Sandholm, *Deep Counterfactual Regret Minimization,* ICML 2019

External Sampling Traversal

Algorithm 2 CFR Traversal with External Sampling

function TRAVERSE($h, p, \theta_1, \theta_2, \mathcal{M}_V, \mathcal{M}_\Pi$, t)

Input: History h, traverser player p, regret network parameters θ for each player, advantage memory $\mathcal{M}_V$ for player p, strategy memory $\mathcal{M}_\Pi$, CFR iteration t.

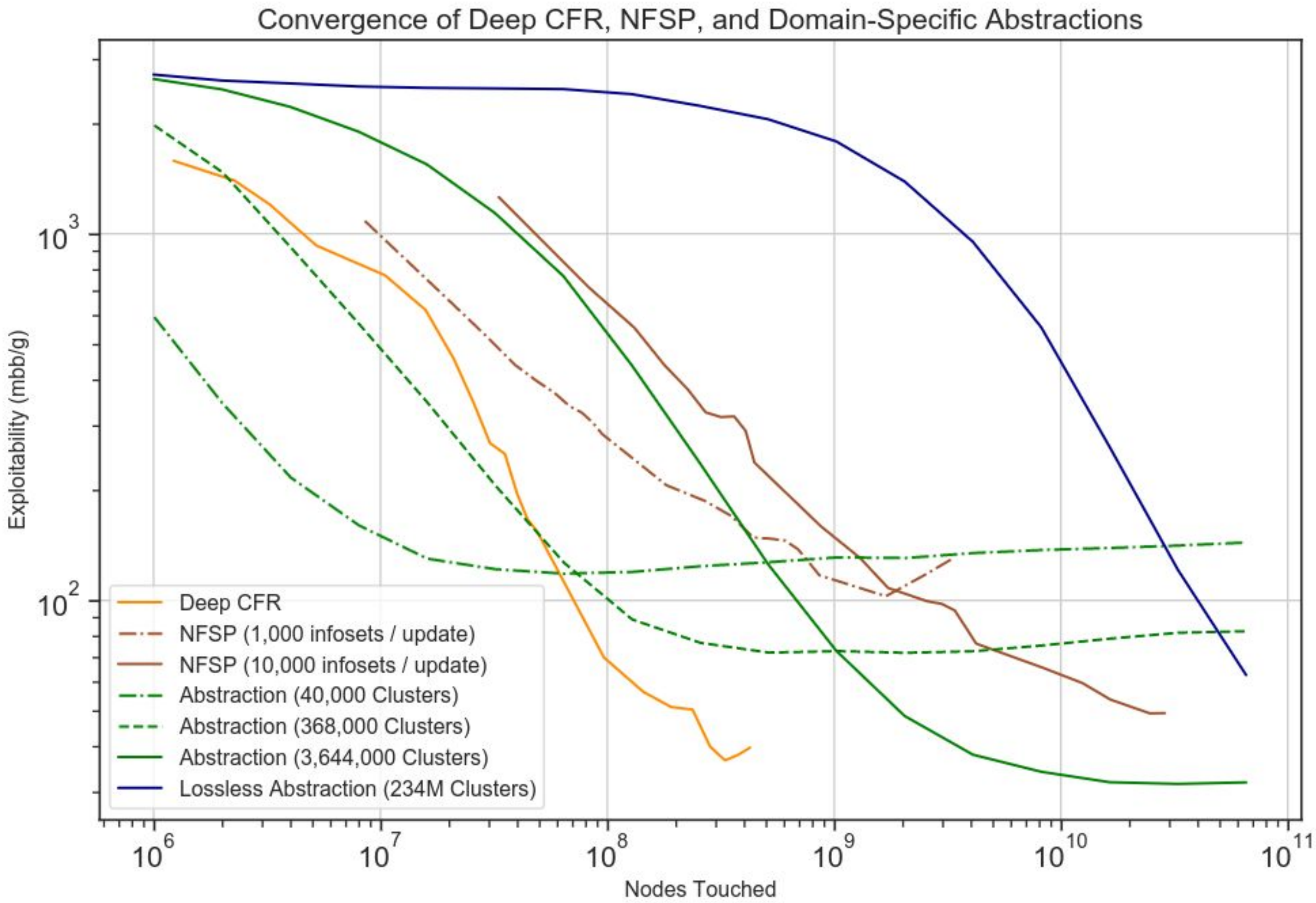
if h is terminal **then**
  return the payoff to player p
else if h is a chance node **then**
  $a \sim \sigma(h)$
  return TRAVERSE($h \cdot a, p, \theta_1, \theta_2, \mathcal{M}_V, \mathcal{M}_\Pi$, t)
else if $P(h) = p$ **then** ▷ If it's the traverser's turn to act
  Compute strategy $\sigma^t(I)$ from predicted advantages $V(I(h), a|\theta_p)$ using regret matching.
  for $a \in A(h)$ **do**
    $v(a) \leftarrow$ TRAVERSE($h \cdot a, p, \theta_1, \theta_2, \mathcal{M}_V, \mathcal{M}_\Pi$, t) ▷ Traverse each action
  for $a \in A(h)$ **do**
    $\tilde{r}(I, a) \leftarrow v(a) - \sum_{a' \in A(h)} \sigma(I, a') \cdot v(a')$ ▷ Compute advantages
  Insert the infoset and its action advantages $(I, t, \tilde{r}^t(I))$ into the advantage memory $\mathcal{M}_V$
else ▷ If it's the opponent's turn to act
  Compute strategy $\sigma^t(I)$ from predicted advantages $V(I(h), a|\theta_{3-p})$ using regret matching.
  Insert the infoset and its action probabilities $(I, t, \sigma^t(I))$ into the strategy memory $\mathcal{M}_\Pi$
  Sample an action a from the probability distribution $\sigma^t(I)$.
  return TRAVERSE($h \cdot a, p, \theta_1, \theta_2, \mathcal{M}_V, \mathcal{M}_\Pi$, t)

Deep CFR Pseudocode

Algorithm 1 Deep Counterfactual Regret Minimization

function DEEPCFR
  Initialize each player's advantage network $V(I, a|\theta_p)$ with parameters θ_p so that it returns 0 for all inputs.
  Initialize reservoir-sampled advantage memories $\mathcal{M}_{V,1}, \mathcal{M}_{V,2}$ and strategy memory $\mathcal{M}_{\Pi}$.
  for CFR iteration $t = 1$ to T **do**
    for each player p **do**
      for traversal $k = 1$ to K **do**
        TRAVERSE($\emptyset, p, \theta_1, \theta_2, \mathcal{M}_{V,p}, \mathcal{M}_{\Pi}$) ▷ Collect data from a game traversal with external sampling
      Train θ_p from scratch on loss $\mathcal{L}(\theta_p) = \mathbb{E}_{(I,t',\tilde{r}^{t'})\sim\mathcal{M}_{V,p}}\left[t' \sum_a \left(\tilde{r}^{t'}(a) - V(I, a|\theta_p)\right)^2\right]$
  Train θ_{Π} on loss $\mathcal{L}(\theta_{\Pi}) = \mathbb{E}_{(I,t',\sigma^{t'})\sim\mathcal{M}_{\Pi}}\left[t' \sum_a \left(\sigma^{t'}(a) - \Pi(I, a|\theta_{\Pi})\right)^2\right]$
  return θ_{Π}

Deep CFR Results–Heads Up Limit Hold 'Em



Connection Between CF-Regret and Q Values

$$q_{\pi,i}(s_t, a_t) = \mathbb{E}_{\rho\sim\pi}[G_{t,i} \mid S_t = s_t, A_t = a_t]$$

Connection Between CF-Regret and Q Values

$$q_{\pi,i}(s_t, a_t) = \mathbb{E}_{\rho\sim\pi}[G_{t,i} \mid S_t = s_t, A_t = a_t]$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \Pr(h \mid s_t)\eta^{\pi}(ha, z)u_i(z) \qquad \text{where } \eta^{\pi}(ha, z) = \frac{\eta^{\pi}(z)}{\eta^{\pi}(h)\pi(s, a)}$$

Connection Between CF-Regret and Q Values

$$q_{\pi,i}(s_t, a_t) = \mathbb{E}_{\rho\sim\pi}[G_{t,i} \mid S_t = s_t, A_t = a_t]$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \Pr(h \mid s_t)\eta^{\pi}(ha, z)u_i(z)$$

where $\eta^{\pi}(ha, z) = \dfrac{\eta^{\pi}(z)}{\eta^{\pi}(h)\pi(s, a)}$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\Pr(s_t \mid h)\Pr(h)}{\Pr(s_t)}\eta^{\pi}(ha, z)u_i(z)$$

by Bayes' rule

Connection Between CF-Regret and Q Values

$$q_{\pi,i}(s_t, a_t) = \mathbb{E}_{\rho\sim\pi}[G_{t,i} \mid S_t = s_t, A_t = a_t]$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \Pr(h \mid s_t)\eta^\pi(ha, z)u_i(z) \qquad \text{where } \eta^\pi(ha, z) = \frac{\eta^\pi(z)}{\eta^\pi(h)\pi(s,a)}$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\Pr(s_t \mid h)\Pr(h)}{\Pr(s_t)}\eta^\pi(ha, z)u_i(z) \qquad \text{by Bayes' rule}$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\Pr(h)}{\Pr(s_t)}\eta^\pi(ha, z)u_i(z) \qquad \text{since } h \in s_t,\ h \text{ is unique to } s_t$$

Connection Between CF-Regret and Q Values

$$
\begin{aligned}
q_{\pi,i}(s_t, a_t) &= \mathbb{E}_{\rho\sim\pi}[G_{t,i} \mid S_t = s_t, A_t = a_t] \\
&= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \Pr(h \mid s_t)\eta^\pi(ha,z)u_i(z) && \text{where } \eta^\pi(ha,z) = \frac{\eta^\pi(z)}{\eta^\pi(h)\pi(s,a)} \\
&= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\Pr(s_t \mid h)\Pr(h)}{\Pr(s_t)}\eta^\pi(ha,z)u_i(z) && \text{by Bayes' rule} \\
&= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\Pr(h)}{\Pr(s_t)}\eta^\pi(ha,z)u_i(z) && \text{since } h \in s_t,\ h \text{ is unique to } s_t \\
&= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\eta^\pi(h)}{\sum_{h'\in s_t}\eta^\pi(h')}\eta^\pi(ha,z)u_i(z)
\end{aligned}
$$

Connection Between CF-Regret and Q Values

$$q_{\pi,i}(s_t, a_t) = \mathbb{E}_{\rho\sim\pi}[G_{t,i} \mid S_t = s_t, A_t = a_t]$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \Pr(h \mid s_t)\eta^\pi(ha, z)u_i(z) \qquad \text{where } \eta^\pi(ha, z) = \frac{\eta^\pi(z)}{\eta^\pi(h)\pi(s, a)}$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\Pr(s_t \mid h)\Pr(h)}{\Pr(s_t)}\eta^\pi(ha, z)u_i(z) \qquad \text{by Bayes' rule}$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\Pr(h)}{\Pr(s_t)}\eta^\pi(ha, z)u_i(z) \qquad \text{since } h \in s_t,\ h \text{ is unique to } s_t$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\eta^\pi(h)}{\sum_{h'\in s_t}\eta^\pi(h')}\eta^\pi(ha, z)u_i(z)$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\eta_i^\pi(h)\eta_{-i}^\pi(h)}{\sum_{h'\in s_t}\eta_i^\pi(h')\eta_{-i}^\pi(h')}\eta^\pi(ha, z)u_i(z)$$

Connection Between CF-Regret and Q Values

$$q_{\pi,i}(s_t, a_t) = \mathbb{E}_{\rho\sim\pi}[G_{t,i} \mid S_t = s_t, A_t = a_t]$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \Pr(h \mid s_t)\eta^\pi(ha,z)u_i(z) \qquad \text{where } \eta^\pi(ha,z) = \frac{\eta^\pi(z)}{\eta^\pi(h)\pi(s,a)}$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\Pr(s_t \mid h)\Pr(h)}{\Pr(s_t)}\eta^\pi(ha,z)u_i(z) \qquad \text{by Bayes' rule}$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\Pr(h)}{\Pr(s_t)}\eta^\pi(ha,z)u_i(z) \qquad \text{since } h \in s_t,\ h \text{ is unique to } s_t$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\eta^\pi(h)}{\sum_{h'\in s_t}\eta^\pi(h')}\eta^\pi(ha,z)u_i(z)$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\eta_i^\pi(h)\eta_{-i}^\pi(h)}{\sum_{h'\in s_t}\eta_i^\pi(h')\eta_{-i}^\pi(h')}\eta^\pi(ha,z)u_i(z)$$

$$= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\eta_i^\pi(s)\eta_{-i}^\pi(h)}{\eta_i^\pi(s)\sum_{h'\in s_t}\eta_{-i}^\pi(h')}\eta^\pi(ha,z)u_i(z) \qquad \text{due to def. of } s_t \text{ and perfect recall}$$

Connection Between CF-Regret and Q Values

$$
\begin{aligned}
q_{\pi,i}(s_t, a_t) &= \mathbb{E}_{\rho\sim\pi}[G_{t,i} \mid S_t = s_t, A_t = a_t] \\
&= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \Pr(h \mid s_t)\eta^{\pi}(ha,z)u_i(z) && \text{where } \eta^{\pi}(ha,z) = \frac{\eta^{\pi}(z)}{\eta^{\pi}(h)\pi(s,a)} \\
&= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\Pr(s_t \mid h)\Pr(h)}{\Pr(s_t)}\eta^{\pi}(ha,z)u_i(z) && \text{by Bayes' rule} \\
&= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\Pr(h)}{\Pr(s_t)}\eta^{\pi}(ha,z)u_i(z) && \text{since } h\in s_t\text{, } h \text{ is unique to } s_t \\
&= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\eta^{\pi}(h)}{\sum_{h'\in s_t}\eta^{\pi}(h')}\eta^{\pi}(ha,z)u_i(z) \\
&= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\eta_i^{\pi}(h)\eta_{-i}^{\pi}(h)}{\sum_{h'\in s_t}\eta_i^{\pi}(h')\eta_{-i}^{\pi}(h')}\eta^{\pi}(ha,z)u_i(z) \\
&= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\eta_i^{\pi}(s)\eta_{-i}^{\pi}(h)}{\eta_i^{\pi}(s)\sum_{h'\in s_t}\eta_{-i}^{\pi}(h')}\eta^{\pi}(ha,z)u_i(z) && \text{due to def. of } s_t \text{ and perfect recall} \\
&= \sum_{h,z\in\mathcal{Z}(s_t,a_t)} \frac{\eta_{-i}^{\pi}(h)}{\sum_{h'\in s_t}\eta_{-i}^{\pi}(h')}\eta^{\pi}(ha,z)u_i(z) \;=\; \frac{1}{\sum_{h\in s_t}\eta_{-i}^{\pi}(h)}v_i^c(\pi, s_t, a_t).
\end{aligned}
$$

Review: Policy Gradient

$$\underbrace{p_\theta(\mathbf{s}_1, \mathbf{a}_1, \ldots, \mathbf{s}_T, \mathbf{a}_T)}_{\pi_\theta(\tau)} = p(\mathbf{s}_1) \prod_{t=1}^{T} \pi_\theta(\mathbf{a}_t | \mathbf{s}_t) p(\mathbf{s}_{t+1} | \mathbf{s}_t, \mathbf{a}_t)$$

$$\theta^\star = \arg\max_\theta E_{\tau \sim p_\theta(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right]$$

Review: Policy Gradient

$$\theta^{\star} = \arg\max_{\theta} \underbrace{E_{\tau \sim p_{\theta}(\tau)} \left[\sum_{t} r(\mathbf{s}_t, \mathbf{a}_t) \right]}_{J(\theta)}$$

$$J(\theta) = E_{\tau \sim \pi_{\theta}(\tau)} [\underbrace{r(\tau)}_{\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t)}] = \int \pi_{\theta}(\tau) r(\tau) d\tau$$

Review: Policy Gradient

$$\nabla_\theta J(\theta) \;=\int \underline{\nabla_\theta \pi_\theta(\tau)} r(\tau) d\tau \;=\int \underline{\pi_\theta(\tau)\nabla_\theta \log \pi_\theta(\tau)} r(\tau) d\tau \;= E_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$$

Review: Policy Gradient

$$\nabla_\theta J(\theta) = \int \nabla_\theta \pi_\theta(\tau) r(\tau) d\tau = \int \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau) r(\tau) d\tau = E_{\tau \sim \pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$$

$$\nabla_\theta J(\theta) = E_{\tau \sim \pi_\theta(\tau)}[\nabla_\theta \underbrace{\log \pi_\theta(\tau)}_{\left[\cancel{\log p(\mathbf{s}_1)} + \sum_{t=1}^{T} \log \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) + \cancel{\log p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)}\right]} r(\tau)]$$

$$\nabla_\theta \left[\cancel{\log p(\mathbf{s}_1)} + \sum_{t=1}^{T} \log \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) + \cancel{\log p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)} \right]$$

$$\nabla_\theta J(\theta) = E_{\tau \sim \pi_\theta(\tau)} \left[\left(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) \right) \left(\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \right) \right]$$

Regret Policy Gradient

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{QPG}}(s) = \sum_{a} [\nabla_{\theta} \pi(s, a; \boldsymbol{\theta})] \left(q(s, a; \mathbf{w}) - \sum_{b} \pi(s, b; \boldsymbol{\theta}) q(s, b, \mathbf{w}) \right)$$

Srinivasan and Lanctot, et al., *Actor-Critic Policy Optimization in Partially Observable Multiagent Environments,* NeurIPS 2018

Regret Policy Gradient

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{QPG}}(s) = \sum_a [\nabla_\theta \pi(s,a;\boldsymbol{\theta})] \left(q(s,a;\mathbf{w}) - \sum_b \pi(s,b;\boldsymbol{\theta}) q(s,b,\mathbf{w}) \right)$$

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{RMPG}}(s) = \sum_a [\nabla_\theta \pi(s,a;\boldsymbol{\theta})] \left(q(s,a;\mathbf{w}) - \sum_b \pi(s,b;\boldsymbol{\theta}) q(s,b,\mathbf{w}) \right)^+$$

Srinivasan and Lanctot, et al., *Actor-Critic Policy Optimization in Partially Observable Multiagent Environments,* NeurIPS 2018

Regret Policy Gradient

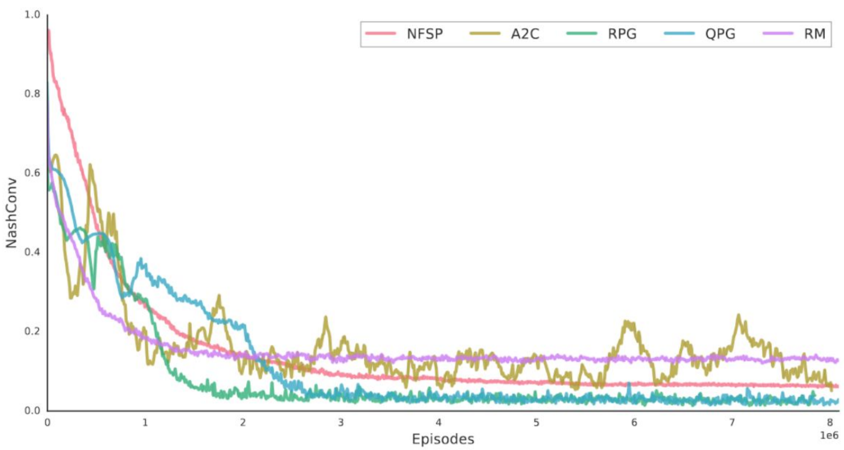
$$\nabla_{\boldsymbol{\theta}}^{\mathrm{QPG}}(s) = \sum_a [\nabla_\theta \pi(s,a;\boldsymbol{\theta})] \left(q(s,a;\mathbf{w}) - \sum_b \pi(s,b;\boldsymbol{\theta}) q(s,b,\mathbf{w}) \right)$$

$$\nabla_{\boldsymbol{\theta}}^{\mathrm{RMPG}}(s) = \sum_a [\nabla_\theta \pi(s,a;\boldsymbol{\theta})] \left(q(s,a;\mathbf{w}) - \sum_b \pi(s,b;\boldsymbol{\theta}) q(s,b,\mathbf{w}) \right)^+$$

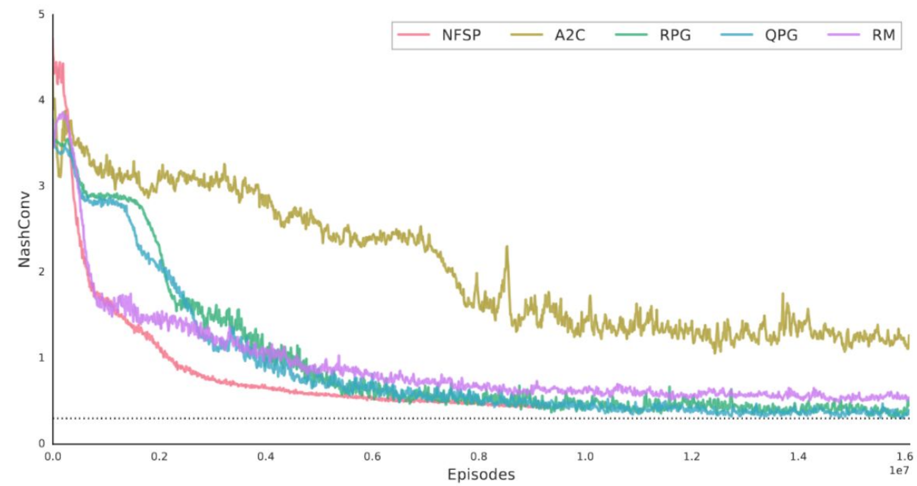
$$\nabla_{\boldsymbol{\theta}}^{\mathrm{RPG}}(s) = -\sum_a \nabla_\theta \left(q(s,a;\mathbf{w}) - \sum_b \pi(s,b;\boldsymbol{\theta}) q(s,b;\mathbf{w}) \right)^+$$

Srinivasan and Lanctot, et al., *Actor-Critic Policy Optimization in Partially Observable Multiagent Environments,* NeurIPS 2018

Results



NASHCONV in 2-player Kuhn



NASHCONV in 2-player Leduc

Regret Policy Gradient

- **Not guaranteed to converge to a Nash equilibrium!**
  - Because it doesn’t take into account the reach weight
- In practice can converge
- Makes connection between policy gradients and game-theoretic algorithms
- Can bring in tools from reinforcement learning to solve games 🙂