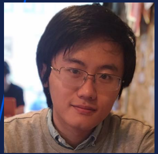
DeepMind

ICML 2020
Paper 3655

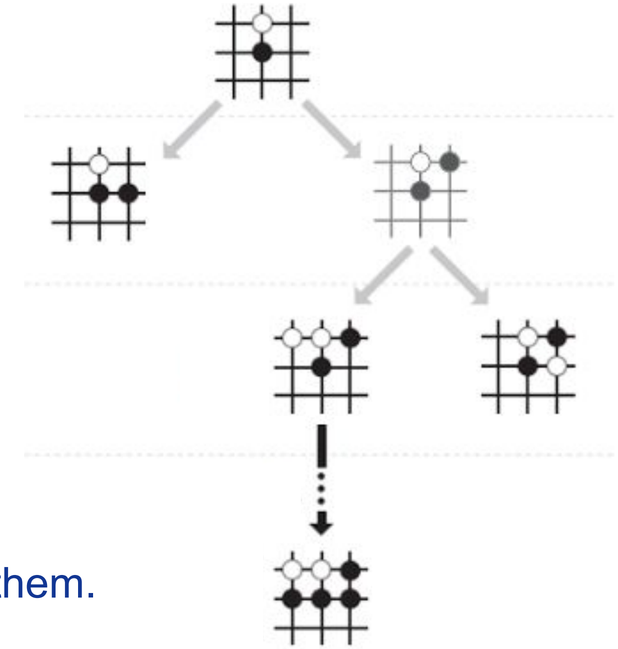
MCTS as Regularized Policy Optimization

Jean-Bastien Grill, Florent Altché, Yunhao Tang, Thomas Hubert, Michal Valko, Ioannis Antonoglou, Rémi Munos

Motivation

- Planning is thinking before acting.
- Imaginating all future possible trajectories is too costly.
- Instead Monte Carlo Tree Search (MCTS) selects few of them.



Motivation

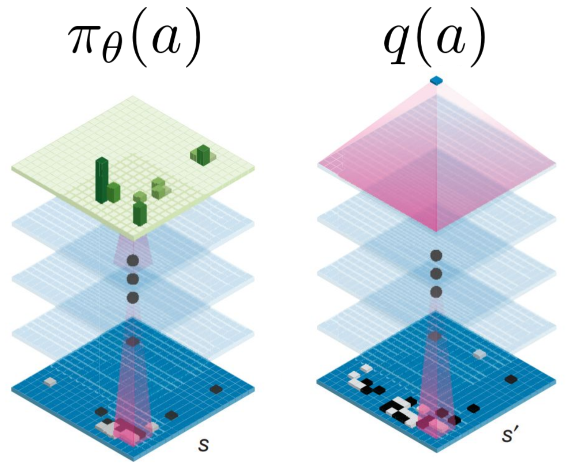
Huge success of MCTS in board games (UCT, AlphaZero), with extensions to richer environments (MuZero)

AlphaZero uses learned **policy** and **value networks** to **select** which path to search during planning.

MCTS in AlphaZero

AlphaZero action selection formula:

$$\arg\max_a \underbrace{q(a)}_{\text{exploitation}} + \underbrace{c\sqrt{N} \cdot \frac{\pi_\theta(a)}{1+n(a)}}_{\text{exploration}}$$



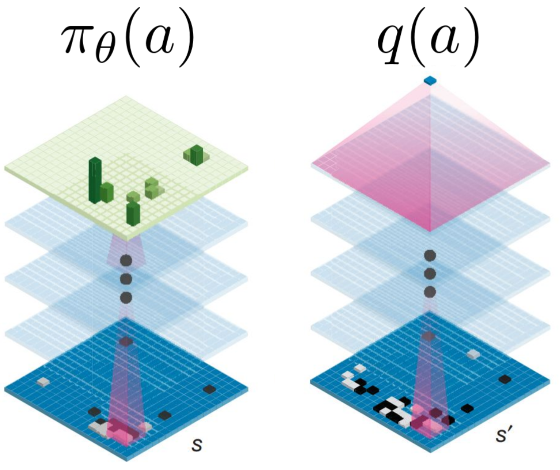
Handcrafted selection formula

Yet, generalised well to other games than Go.

MCTS in AlphaZero

AlphaZero action selection formula:

$$\arg\max_a \underbrace{q(a)}_{\text{exploitation}} + \underbrace{c\sqrt{N} \cdot \frac{\pi_\theta(a)}{1+n(a)}}_{\text{exploration}}$$



Handcrafted selection formula

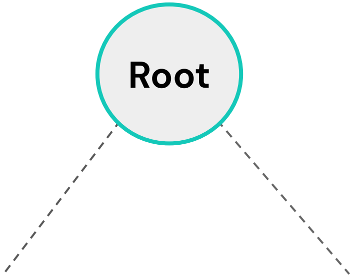
Yet, generalised well to other games than Go.

AlphaZero ⟷ Regularized policy optimization

approximates

AlphaZero search procedure

Simulation budget = 4

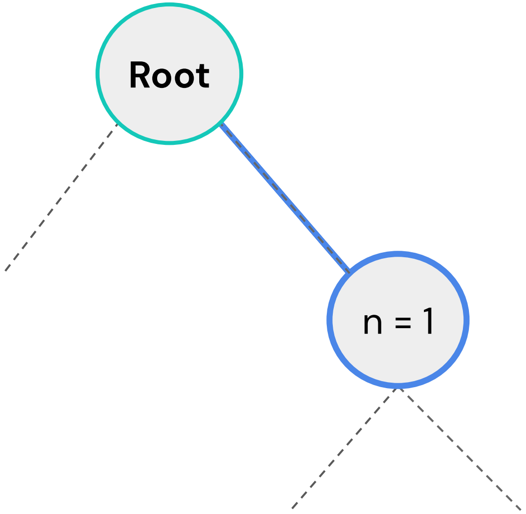


Action selection formula:

$$\arg\max_a q(a) + c\sqrt{N} \cdot \frac{\pi_\theta(a)}{n(a)}$$

AlphaZero search procedure

Simulation budget = 4

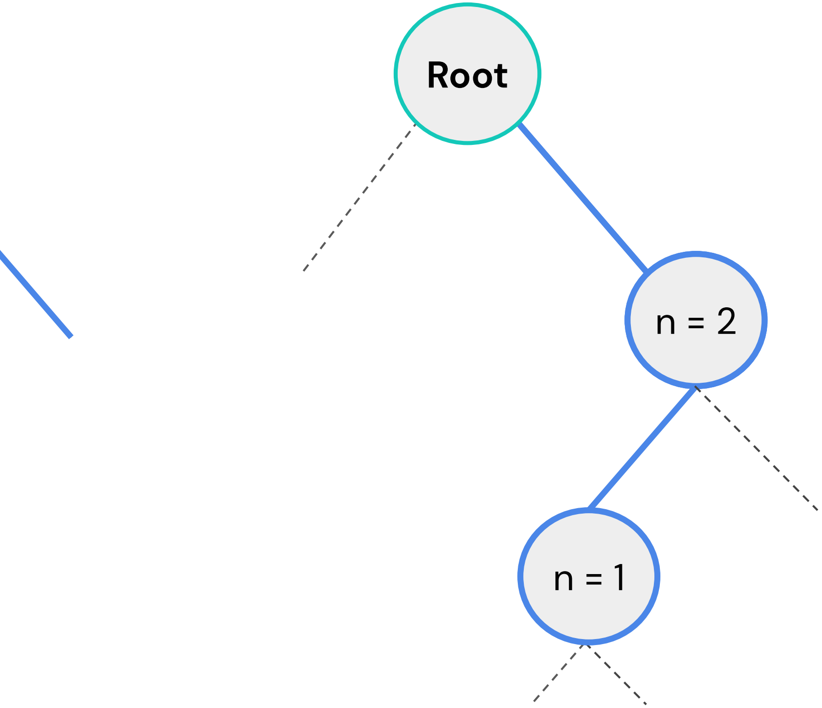


Action selection formula:

$$\arg\max_a q(a) + c\sqrt{N} \cdot \frac{\pi_\theta(a)}{n(a)}$$

AlphaZero search procedure

Simulation budget = 4

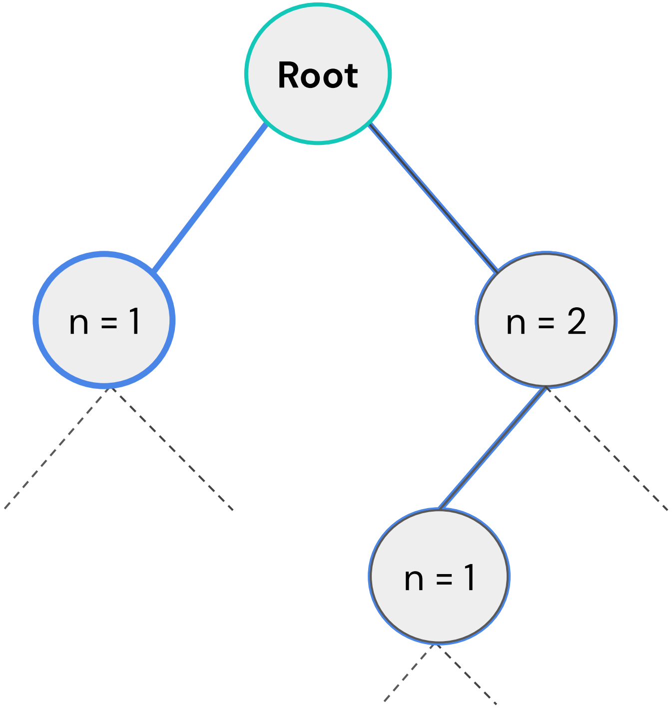


Action selection formula:

$$\arg\max_a q(a) + c\sqrt{N} \cdot \frac{\pi_\theta(a)}{n(a)}$$

AlphaZero search procedure

Simulation budget = 4

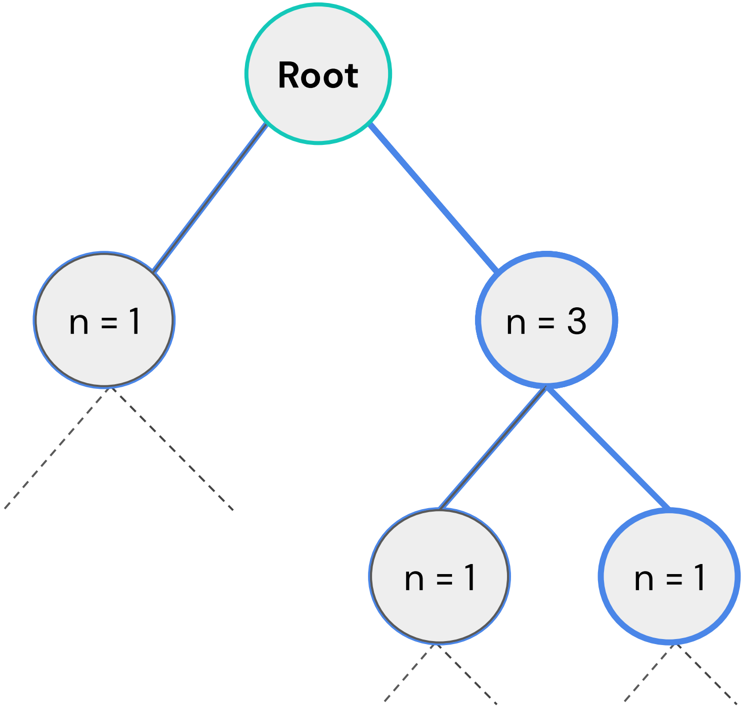


Action selection formula:

$$\arg\max_a q(a) + c\sqrt{N} \cdot \frac{\pi_\theta(a)}{n(a)}$$

AlphaZero search procedure

Simulation budget = 4

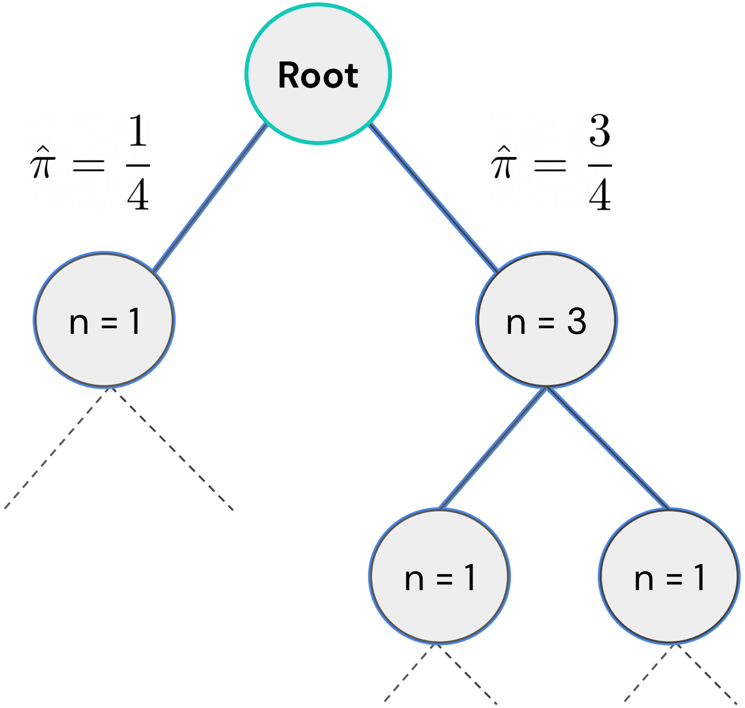


Action selection formula:

$$\arg\max_a q(a) + c\sqrt{N} \cdot \frac{\pi_\theta(a)}{n(a)}$$

AlphaZero search procedure

Simulation budget = 4



Action selection formula:

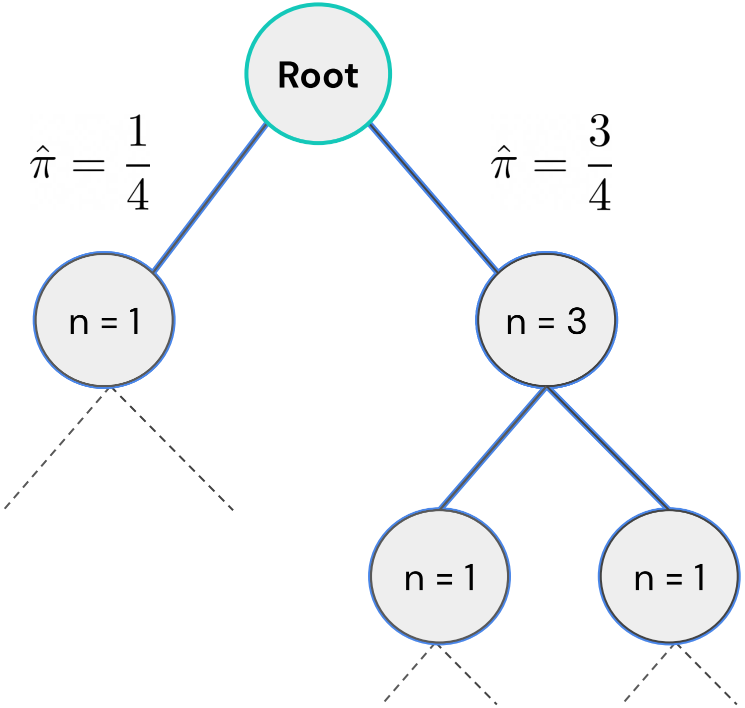
$$\arg\max_a q(a) + c\sqrt{N} \cdot \frac{\pi_\theta(a)}{n(a)}$$

Visit distribution:

$$\hat{\pi}(a) = \frac{n(a)}{\sum_{a'} n(a')}$$

AlphaZero search procedure

Simulation budget = 4



Action selection formula:

$$\arg\max_a q(a) + c\sqrt{N} \cdot \frac{\pi_\theta(a)}{n(a)}$$

Visit distribution:

$$\hat{\pi}(a) = \frac{n(a)}{\sum_{a'} n(a')}$$

Policy network is trained with visit distribution

$$\pi_\theta \longrightarrow \hat{\pi}$$

The action is sampled according to the visit distribution

$$\text{action} \sim \hat{\pi}$$

Main result: AlphaZero as policy optimization

AlphaZero: $$\hat{\pi}(a) = \frac{n(a)}{\sum_{a'} n(a')}$$

Regularized policy optimization:

$$\bar{\pi} = \arg\max_{\pi} \left(\mathbf{q}^T \pi - \lambda_N \mathrm{KL}\left[\pi_\theta, \pi\right]\right)$$

Main result: AlphaZero as policy optimization

AlphaZero: $$\hat{\pi}(a) = \frac{n(a)}{\sum_{a'} n(a')}$$

Regularized policy optimization:

$$\bar{\pi} = \arg\max_{\pi} \left(\mathbf{q}^T \pi - \lambda_N \mathrm{KL}\left[\pi_\theta, \pi\right]\right)$$

AlphaZero $\longleftrightarrow$ Regularized policy optimization

approximates

$\hat{\pi}$ $\bar{\pi}$

Main result: AlphaZero as policy optimization

$$\hat{\pi}(a) = \frac{n(a)}{\sum_{a'} n(a')}$$

Regularized policy optimization:

$$\bar{\pi} = \arg\max_{\pi} \left(\mathbf{q}^T \pi - \lambda_N \mathrm{KL}\left[\pi_\theta, \pi\right]\right)$$

Main result: AlphaZero as policy optimization

$$\hat{\pi}(a) = \frac{n(a)}{\sum_{a'} n(a')}$$

Regularized policy optimization:

$$\bar{\pi} = \arg\max_{\pi} \left(\mathbf{q}^T \pi - \lambda_N \mathrm{KL}\left[\pi_\theta, \pi\right]\right)$$

Gradient ascent step from $\hat{\pi}$:

$$\hat{\pi} \leftarrow \hat{\pi} + \eta \cdot \frac{\partial}{\partial n}\left(\mathbf{q}^T \hat{\pi} - \lambda_N \mathrm{KL}\left[\pi_\theta, \hat{\pi}\right]\right)$$

Main result: AlphaZero as policy optimization

$$\hat{\pi}(a) = \frac{n(a)}{\sum_{a'} n(a')}$$

Regularized policy optimization:

$$\bar{\pi} = \arg\max_{\pi} \left(\mathbf{q}^T \pi - \lambda_N \mathrm{KL}\left[\pi_\theta, \pi\right] \right)$$

Gradient ascent step from $\hat{\pi}$:

$$\hat{\pi} \leftarrow \hat{\pi} + \eta \cdot \frac{\partial}{\partial n} \left(\mathbf{q}^T \hat{\pi} - \lambda_N \mathrm{KL}\left[\pi_\theta, \hat{\pi}\right] \right)$$

Corresponding action selection:

$$a^* = \arg\max_a \left[\frac{\partial}{\partial n} \left(\mathbf{q}^T \hat{\pi} - \lambda_N \mathrm{KL}\left[\pi_\theta, \hat{\pi}\right] \right) \right]$$

How to use $\bar{\pi}$ in AlphaZero

AlphaZero can be broken down into three main components:

- **Act:** sample $\hat{\pi}$
- **Learn:** train towards $\hat{\pi}$
- **Search:** action selection

How to use $\bar{\pi}$ in AlphaZero

AlphaZero can be broken down into three main components:

- **Act:** sample $\hat{\pi}$ $\rightarrow$ sample $\bar{\pi}$
- **Learn:** train towards $\hat{\pi}$ $\rightarrow$ train towards $\bar{\pi}$
- **Search:** action selection $\rightarrow$ sample $\bar{\pi}$

Use $\bar{\pi}$ instead

How to use $\bar{\pi}$ in AlphaZero

The learning becomes regularized policy optimization using search Q-values for its Q-values estimates

AlphaZero can be broken down into three main components:

- **Act:** sample $\hat{\pi}$ $\longrightarrow$ sample $\bar{\pi}$
- **Learn:** train towards $\hat{\pi}$ $\longrightarrow$ train towards $\bar{\pi}$
- **Search:** action selection $\longrightarrow$ sample $\bar{\pi}$

Use $\bar{\pi}$ instead

How to use $\bar{\pi}$ in AlphaZero

The learning becomes regularized policy optimization using search Q-values for its Q-values estimates

AlphaZero can be broken down into three main components:

Act:	sample $\hat{\pi}$	$\rightarrow$ sample $\bar{\pi}$
Learn:	train towards $\hat{\pi}$	$\rightarrow$ train towards $\bar{\pi}$

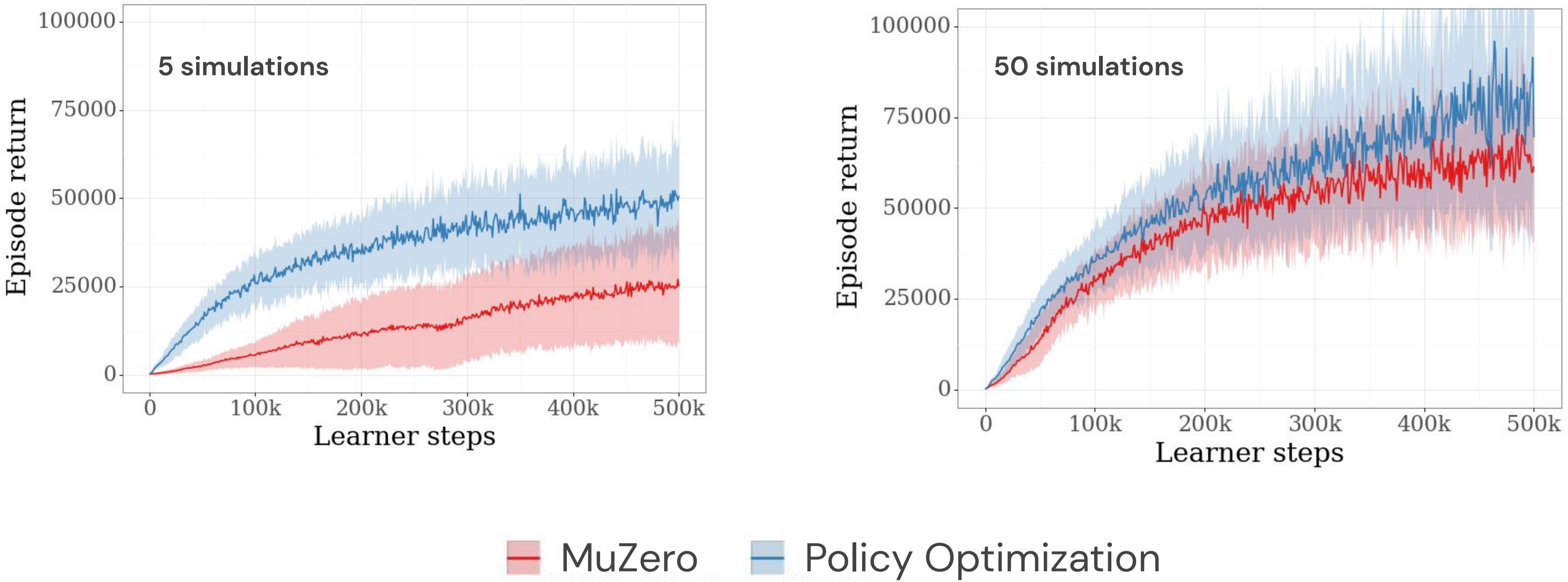
Search:	action selection	$\rightarrow$ sample $\bar{\pi}$

Search becomes regularized policy optimization algorithm on imaginary trajectories

Use $\bar{\pi}$ instead

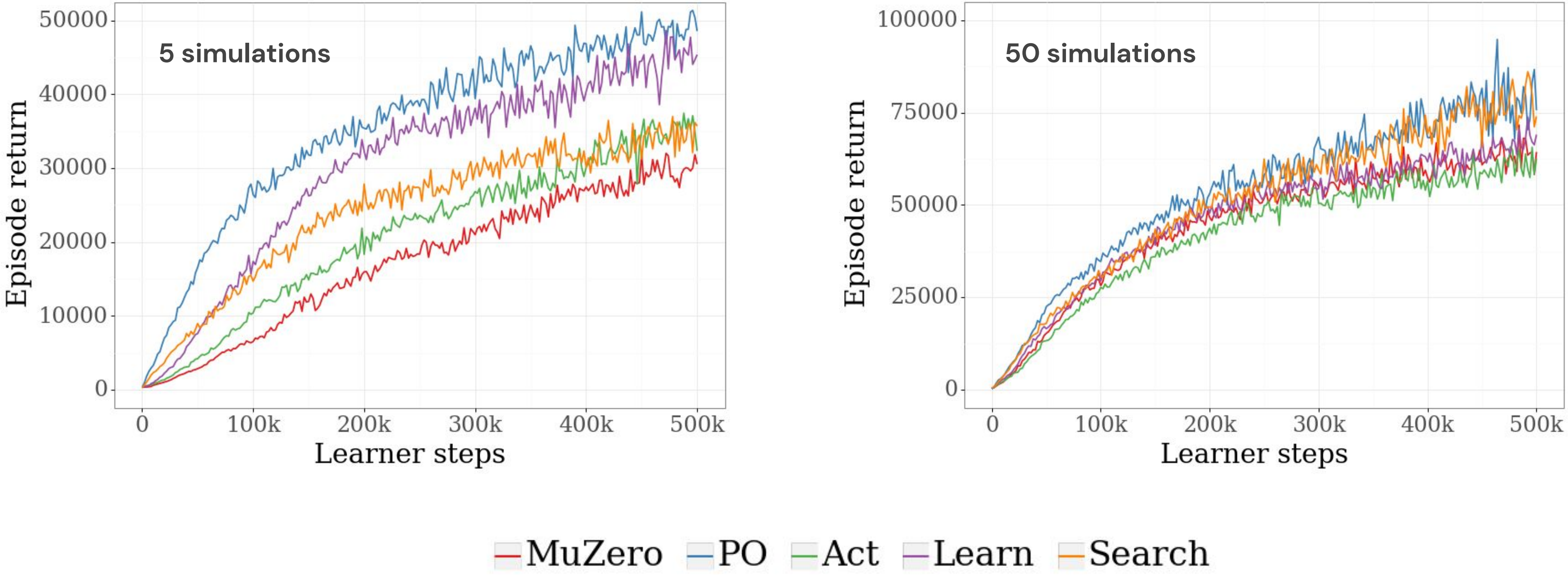
Results on Ms Pacman (Atari)

Results on Ms Pacman (Atari)



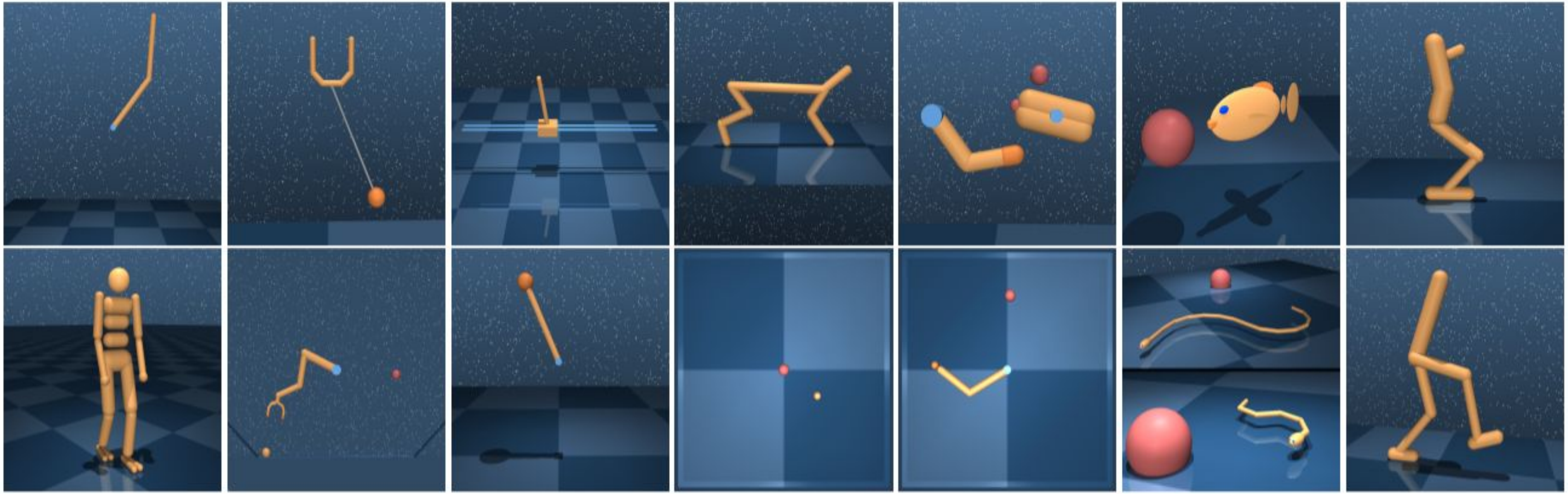
Ms Pacman, 8 seeds -- 8 GPU, 4096 CPU actors

Ablation study on Ms Pacman (Atari)



Ms Pacman, 8 seeds -- 8 GPU, 4096 CPU actors

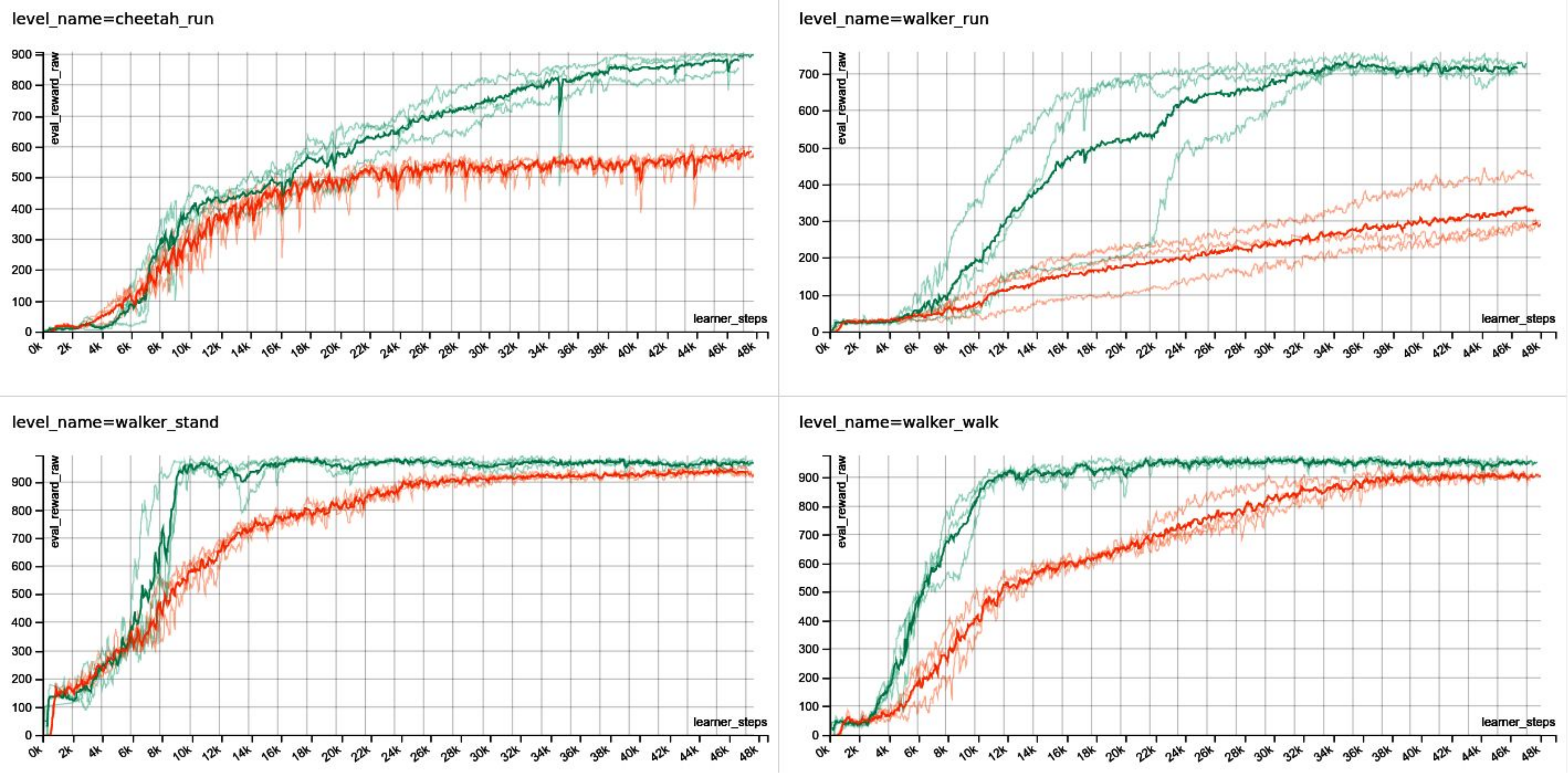
Application: DM Control Suite



Application: DM Control Suite

MuZero

Policy Optimization



Summary

What we showed:

- **AlphaZero** approximates the solution to a regularized policy optimization problem.

Summary

What we showed:

- **AlphaZero** approximates the solution to a regularized policy optimization problem.
- **Experimentally,** using the exact solution provides improved performance without requiring additional parameter tuning.

Summary

What we showed:

- **AlphaZero** approximates the solution to a regularized policy optimization problem.
- **Experimentally,** using the exact solution provides improved performance without requiring additional parameter tuning.

AlphaZero ⟷ Regularized policy optimization

Improvements
and ideas