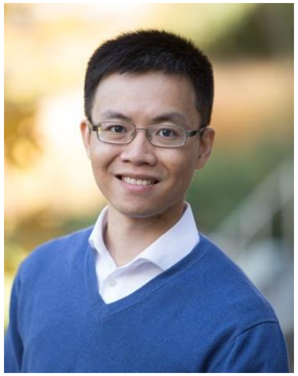
Switch Scheduling via Reinforcement Learning

Dongyan (Lucy) Huo

Cornell ORIE

Joint Work with



Prof Yudong Chen

Wisconsin-Madison CS

Prof Jim Dai

Cornell ORIE

CUHK(Shenzhen) SDS

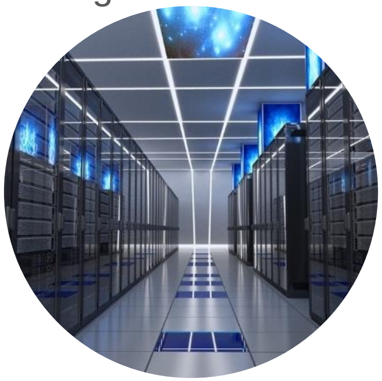
Prof Qiaomin Xie

Wisconsin-Madison ISyE

Reinforcement Learning (RL) for Stochastic Network Control

- Stochastic network control problem is to find a policy for a given stochastic network that optimizes certain criteria
- RL is to automatically learn an algorithm to navigate through complex and unpredictable environments

Datacenter Congestion Control



[Tessler-et-al' 21]

Inpatient Flow Management

[Shi-Dai' upcoming]

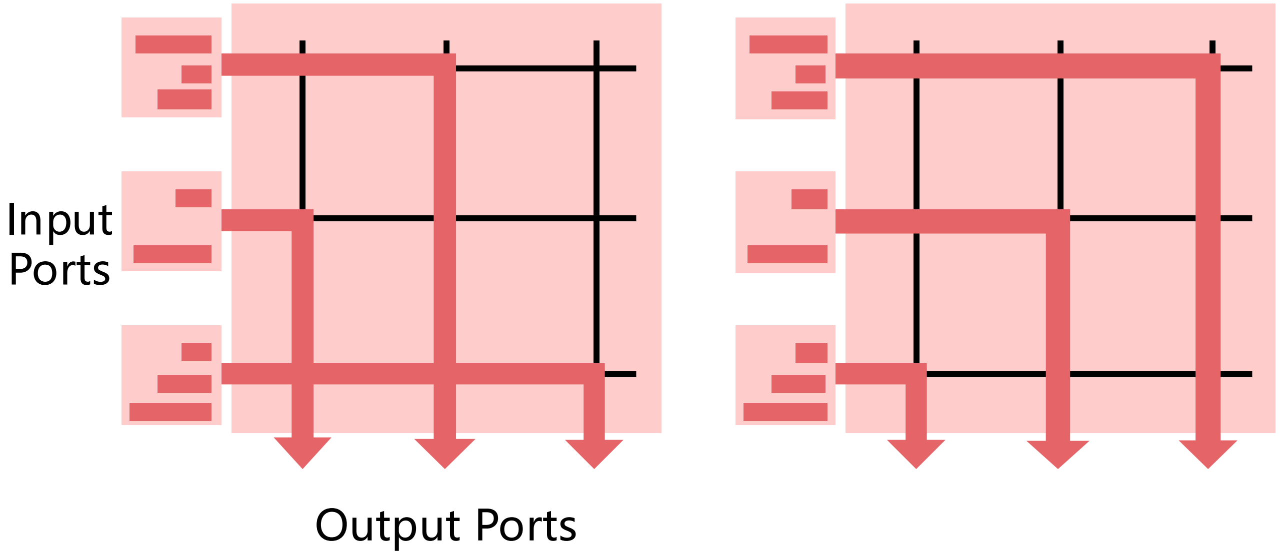
Ride Hailing Matching

[Feng-Gluzman-Dai' 21]

CONTENT

MDP Formulation of Switch Scheduling Problem

- N input ports × N output ports
- Virtual Output Queues (VOQ), $S = \{Q_{ij}\}_{1 \le i,j \le N} \subseteq \mathbb{N}^{N \times N}$
- Combinatorial matching problem, $|\mathcal{A}| = \mathrm{N}!$



MDP Formulation

- Find optimal matching to minimize long-run average cost (LRAC), hence smaller packet delay
- $\mathrm{c}(S,\sigma) = \sum_{i,j} Q_{i,j}$

$$\min_{\pi} \lim_{k\to\infty} \frac{1}{k} \sum_{t=1}^{k} c(Q(t), \pi(Q(t)))$$

- P : induced by the arrival traffic
  - $A_{i,j} \sim Bernoulli\ (\lambda_{i,j})$

Challenges and Goals

- Unbounded state space: $\mathcal{S} \subseteq \mathbb{N}^{N \times N}$
- Large action space: $|\mathcal{A}| = N!$ $(10! \approx 3.6 \times 10^6)$

- Goals: use RL to find a policy with low LRAC
  - Across different arrival traffic patterns
  - Especially when existing algorithms are known to be sub-optimal
- Compare with algorithms
  - MaxWeight(MW): best known and most well-studied algorithm
    - MW-alpha: conjectured to be asymptotically optimal under uniform traffic
    - Asymptotic in alpha decreasing to 0
  - Random d-Flip: low complexity [Jhunjhunwala-Maguluri' 21]

CONTENT

Proximal Policy Optimization (PPO) Algorithm

- Proximal Policy Optimization Algorithms [Schulman-et-al’ 17]
- State-of-the-Art
  - Continuous control, Atari games, etc.

- Designed for discounted reward
  - vs. LRAC in switch scheduling problem
- Actor-critic model
  - Value Function Approximation
  - Policy Function Approximation

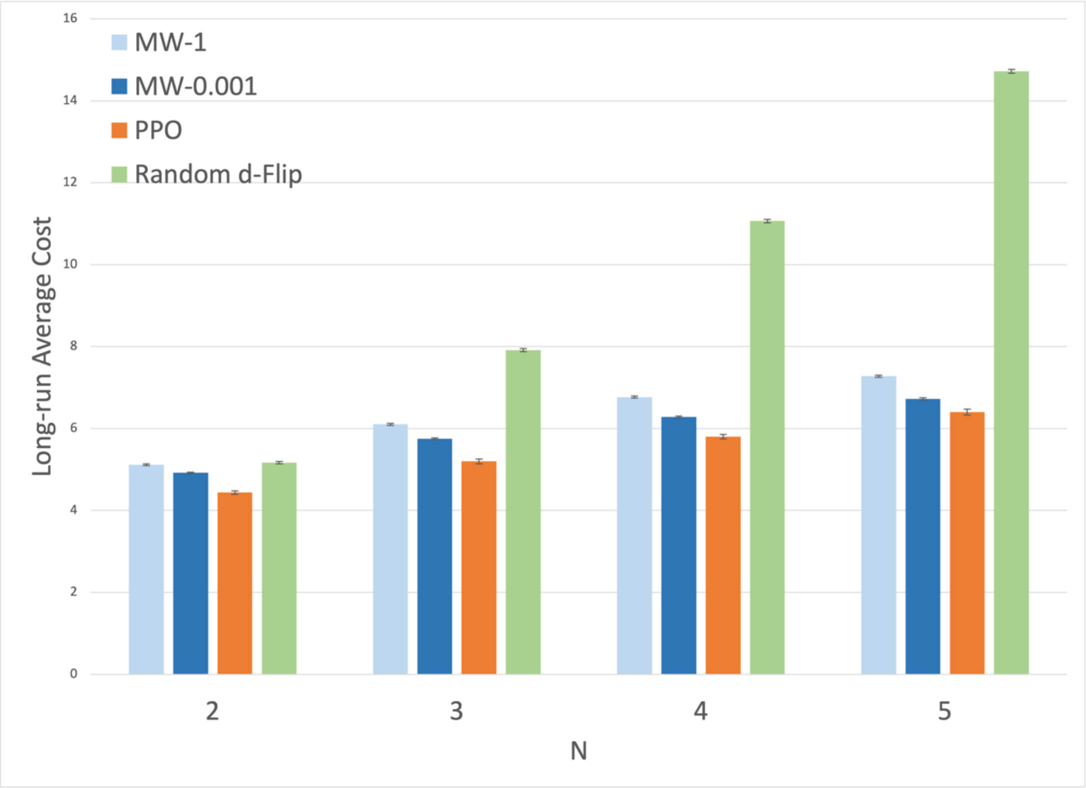
Neural Network

PPO Learning Near Optimal Policy

- Bottom-skewed arrival
- Load $\rho = 0.9$

$\frac{1}{6}\rho$	$\frac{1}{6}\rho$	$\frac{1}{6}\rho$
$\frac{1}{6}\rho$	$\frac{1}{6}\rho$	$\frac{1}{6}\rho$
$\frac{1}{6}\rho$	$\frac{1}{6}\rho$	$\frac{2}{3}\rho$

- PPO Policy beating MW-α

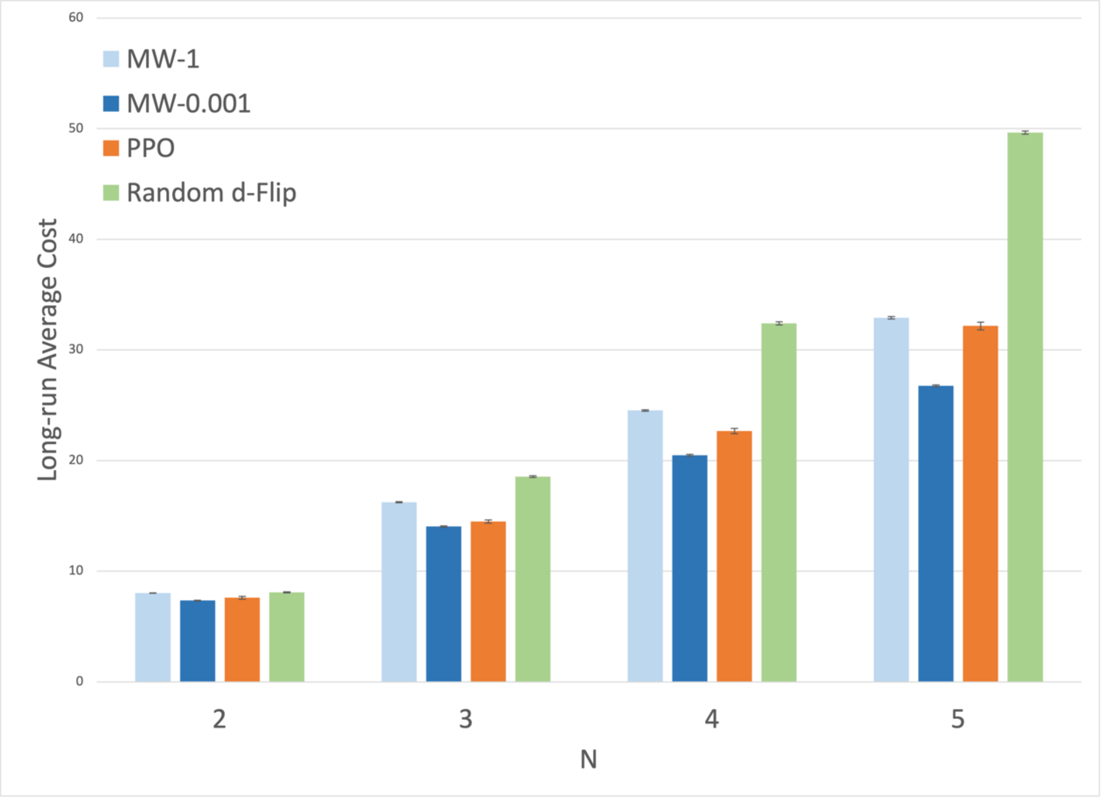


PPO Learning Near Optimal Policy

- Uniform Traffic
- Load $\rho = 0.9$

$\frac{1}{3}\rho$	$\frac{1}{3}\rho$	$\frac{1}{3}\rho$
$\frac{1}{3}\rho$	$\frac{1}{3}\rho$	$\frac{1}{3}\rho$
$\frac{1}{3}\rho$	$\frac{1}{3}\rho$	$\frac{1}{3}\rho$

- MW-α as a near-optimal performance benchmark



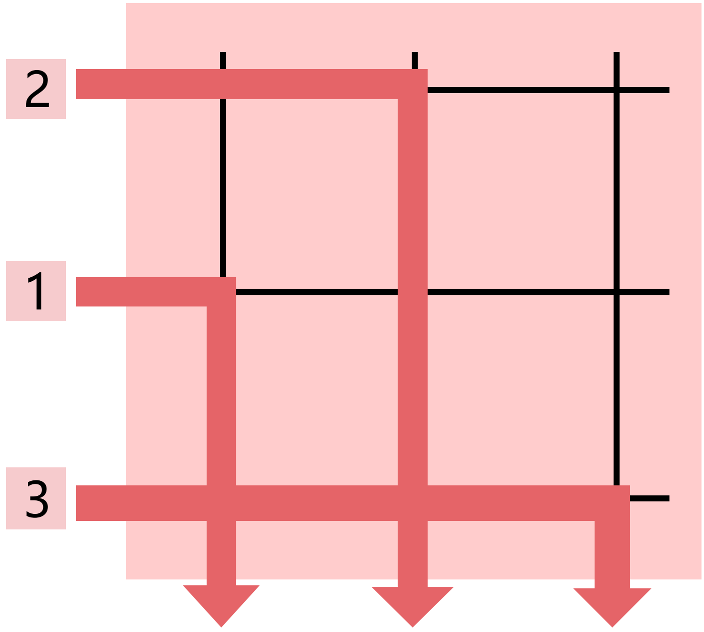
CONTENT

Atomic Action (AA) Decomposition

- Atomic action: sequential decision-making process
  - Successful application in ride-hailing [Feng-Gluzman-Dai' 21], and inpatient flow control [Shi-Dai' upcoming]

- $a_t \in \mathcal{A}, |\mathcal{A}| = N!$
- $a_t = \left(a_{t,1}, a_{t,1}, \dots, a_{t,N}\right) \in \mathcal{A},$
- $a_{t,k} = (i, j) \in \mathcal{A}', |\mathcal{A}'| = N^2$

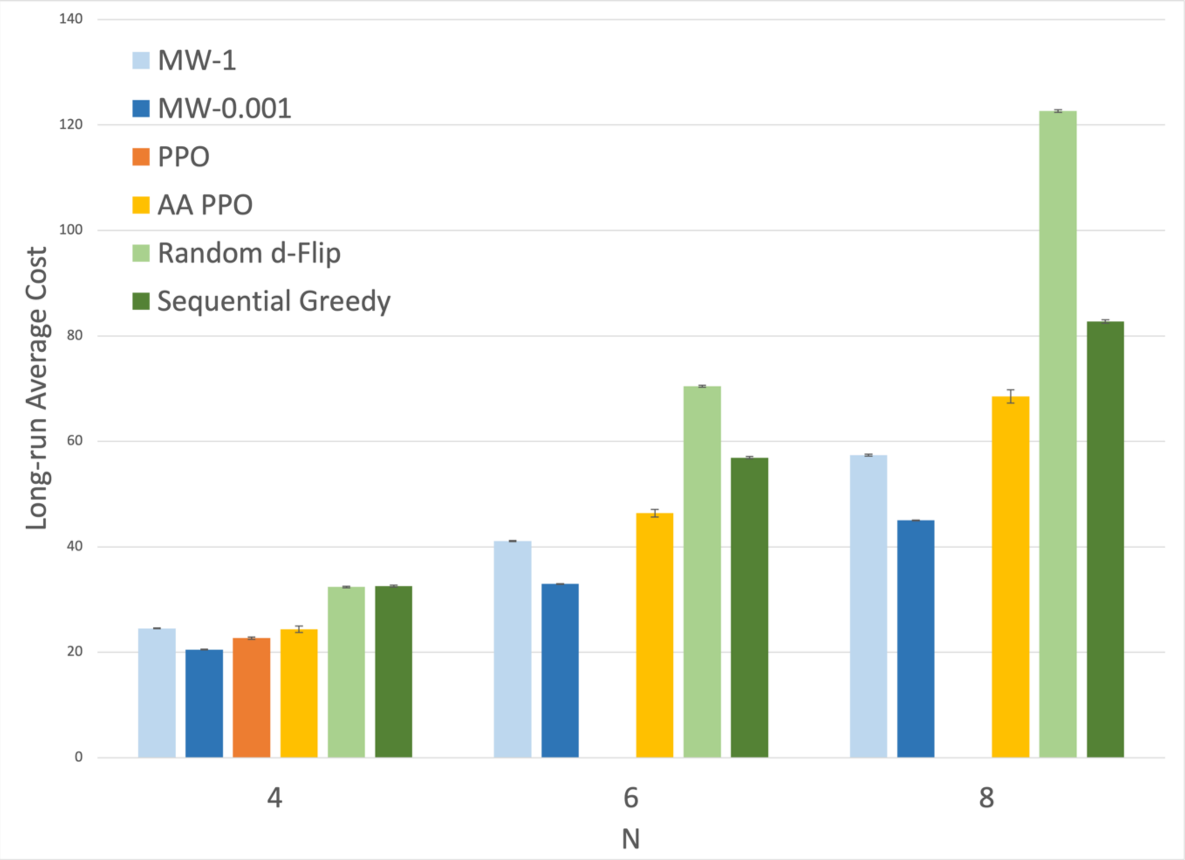
- Problem-specific design
  - To satisfy matching constraint



AA PPO Scalable while Learning Good Policy

- Uniform traffic
- N^2 vs. N!

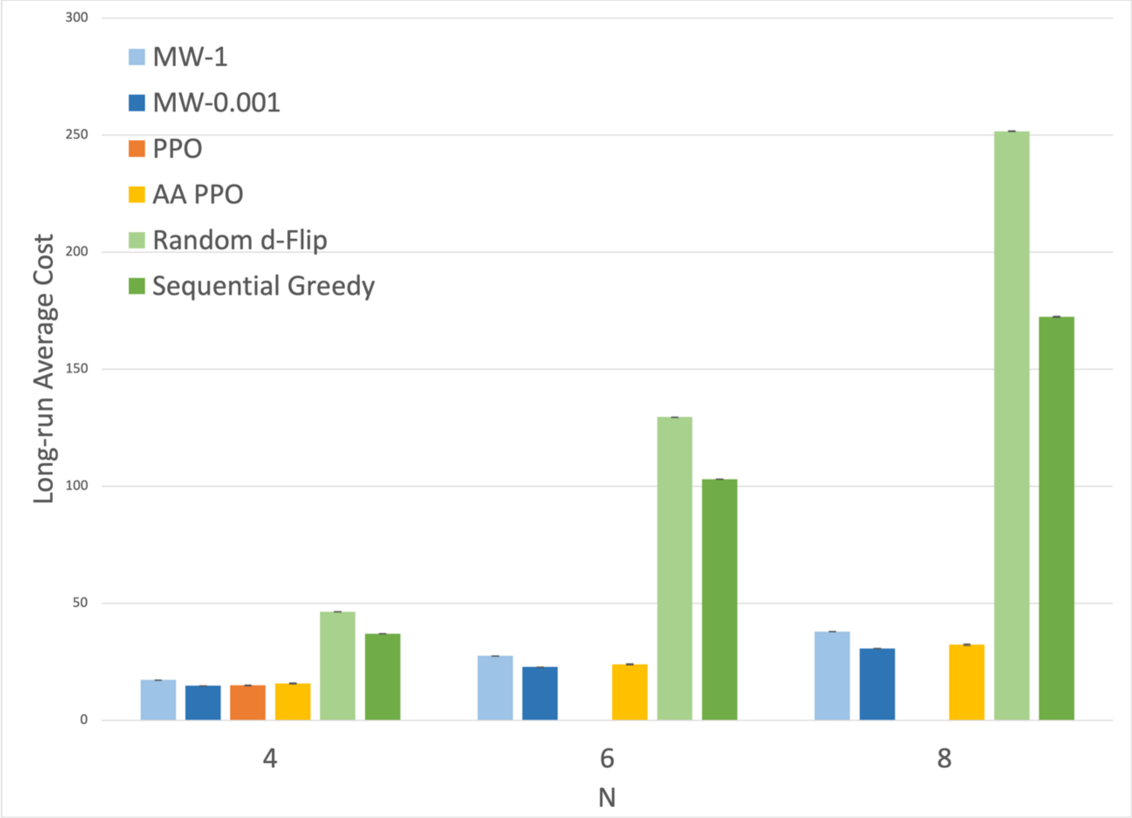
- AA PPO
  - outperforms the greedy policy
  - stays close to MW-1



AA PPO Picking up Traffic Patterns

- Diagonal traffic [Giaccone-Prabhakar-Shah' 02]
  - Favor 2 out of N! possible matchings
  - difficult to schedule with stochastic policies

$\frac{2}{3}\rho$	$\frac{1}{3}\rho$	
	$\frac{2}{3}\rho$	$\frac{1}{3}\rho$
$\frac{1}{3}\rho$		$\frac{2}{3}\rho$



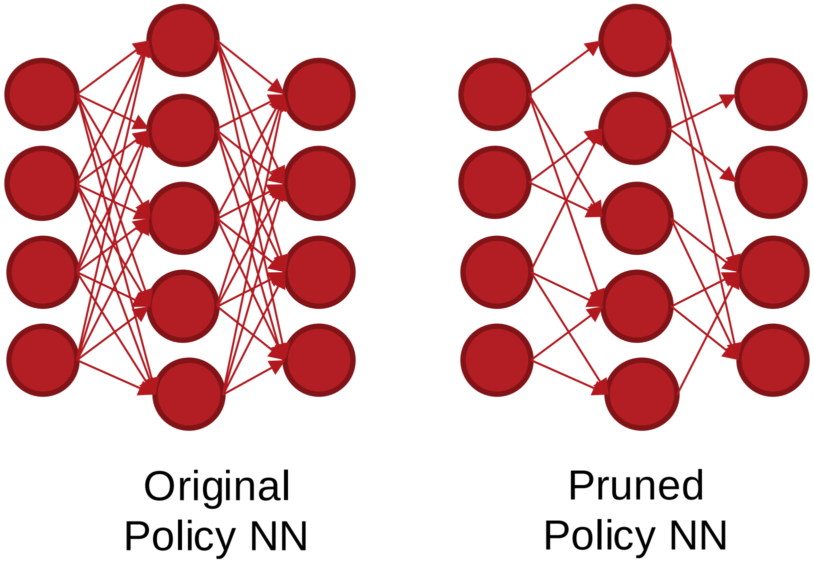
CONTENT

1. Switch Scheduling Problem Formulation
2. Proximal Policy Optimization (PPO)
3. Atomic Action (AA) + PPO
4. **AA + PPO + Neural Network Pruning**

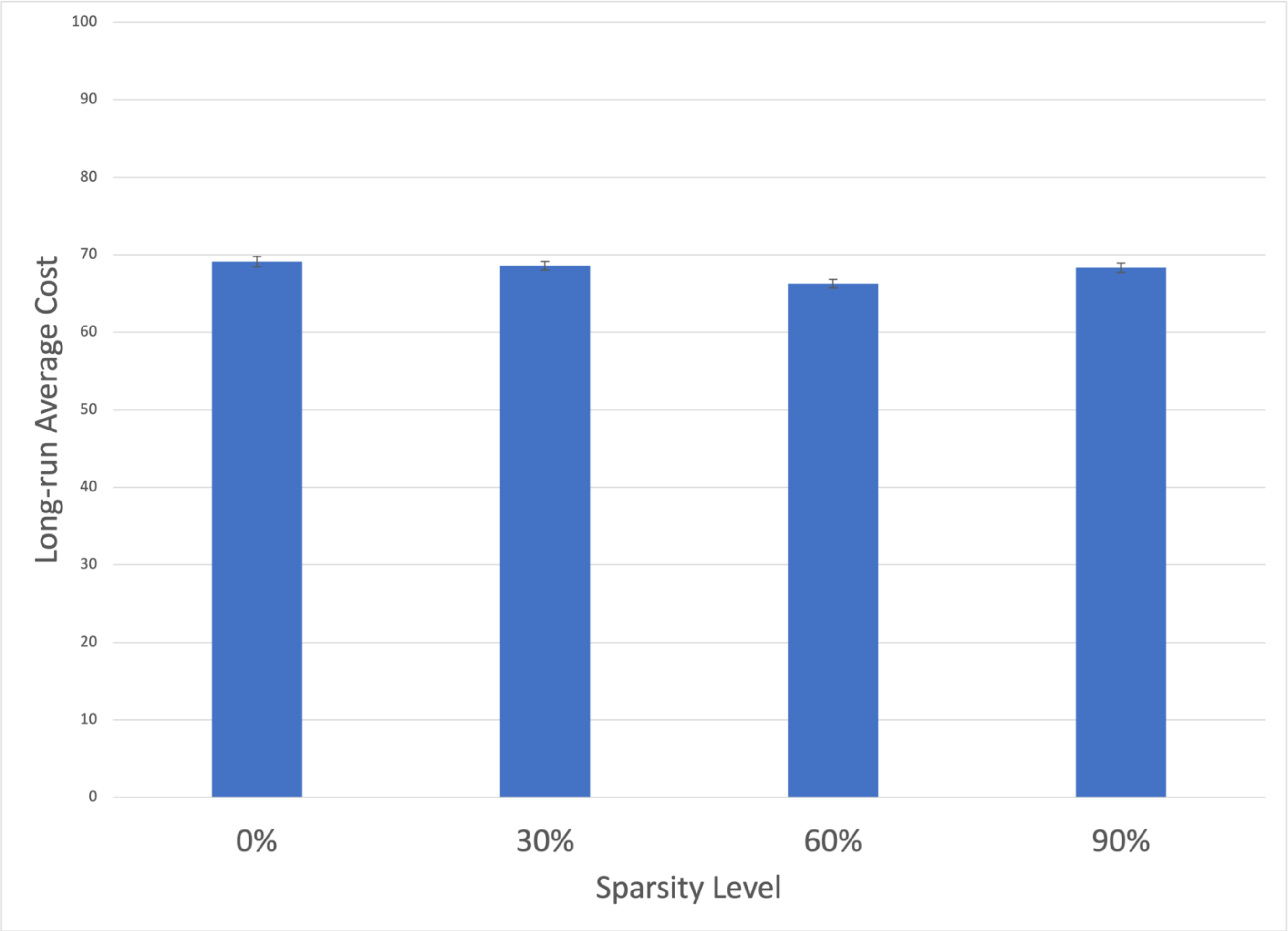
Neural Network (NN) Pruning

- Policy deployment: policy inferencing time matters

- NN pruning
  - deleting parameters from an existing NN [Han-et-al' 15]
  - Aim to keep the policy's performance while reducing its inferencing time



AA PPO policy is robust to NN pruning



Summary

PPO

- Demonstrates the potential of **RL in tackling challenging stochastic network control problems**

AA + PPO

- **Improves the scalability** of the algorithm while maintaining policy performance

AA + PPO + NN Pruning

- **Reduces policy inference time**

Lucy Huo
Cornell University

Thank You