AuTO: Scaling Deep Reinforcement Learning for Datacenter-Scale Automatic Traffic Optimization

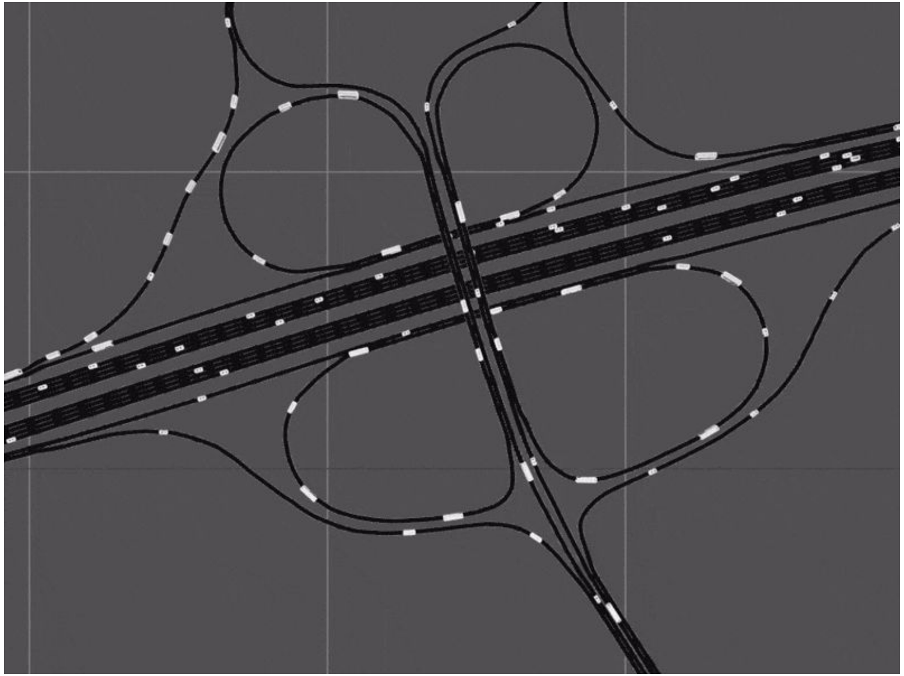
Li Chen, Justinas Lingys, Kai Chen, Feng Liu

Datacenter Network

- Network Flow
  - A sequence of packets from a destination to source
- Network Congestion
- *Big Switch* Assumption and Pitfalls

Traffic Optimization

- Routing Optimizations
- Load balancing
- Scheduling Optimizations



Important Ideas

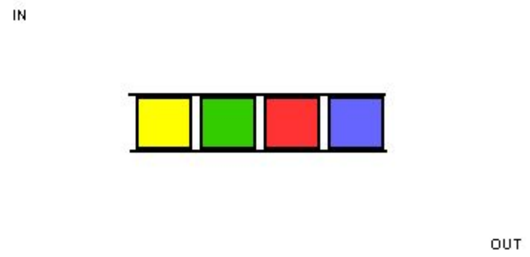
- Traffic optimizations (TO) require specialized knowledge
- TO based on heuristics
- Turn around time is denominated in weeks

Key Problems when Implementing RL

- Using RL for flow calculation at runtime has high latency
- Calculating flow based on past results in poor performance
- High turn around time of traffic optimization

Traditional RL Approach

- Reinforcement learning for flow scheduling
- Leverages Priority queues
    - Flows with higher priorities get processed first
- Deep reinforcement learning is unable to handle datacenter level traffic
    - Computation time > Flow life cycle



Expansion of Past Research

- REINFORCE
  - Demonstrates policy iteration can converge to locally optimal policy
- Other TO systems
  - Only consider stochastic policies
  - State selected according to probability of distribution
- MLFQ (Multi-feedback queueing)
  - Divides process into multiple queues with independent priority

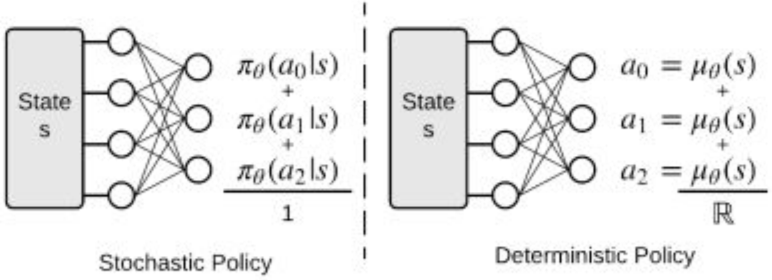
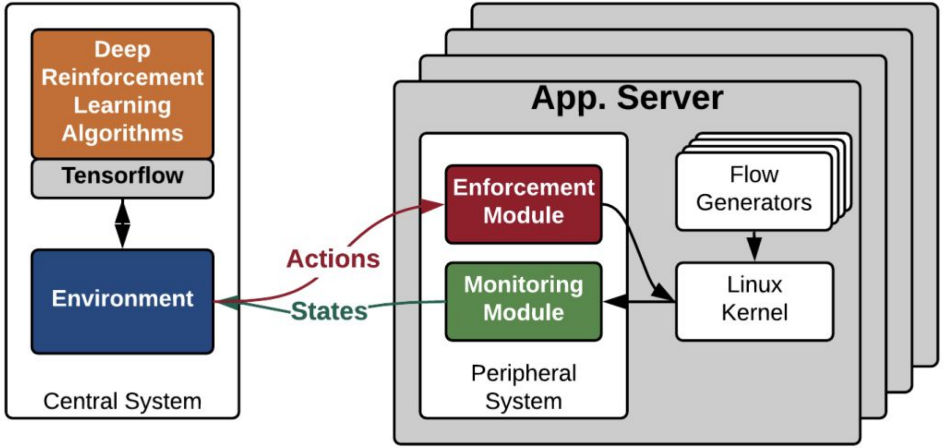


Figure 6: Comparison of deep stochastic and deep deterministic policies.

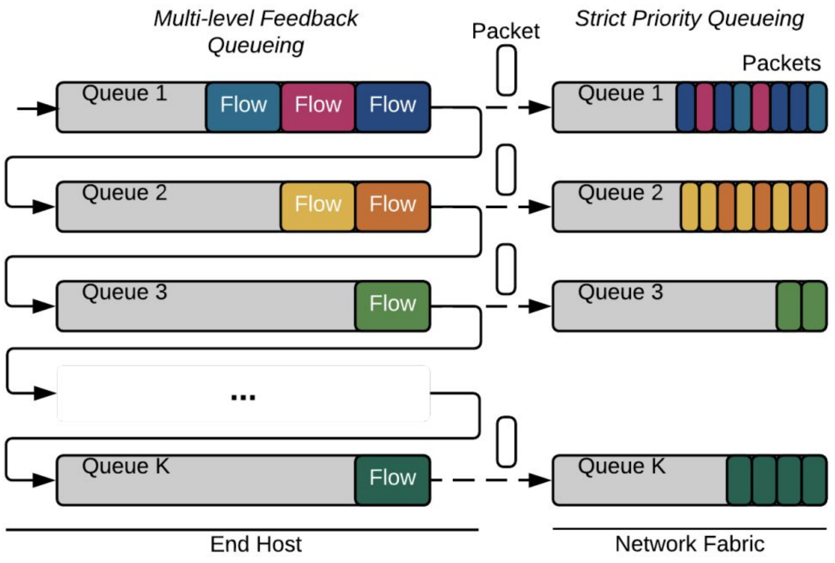
AuTO

- Two level system
  - Peripheral (PS) and Central (CS)
- Peripheral system on end-hosts
  - Collects flow information
  - Executes local traffic optimizations
- Central System
  - Aggregates peripheral system actions
  - Network Described as $\{n_1,m_1,m_{\square}\}$



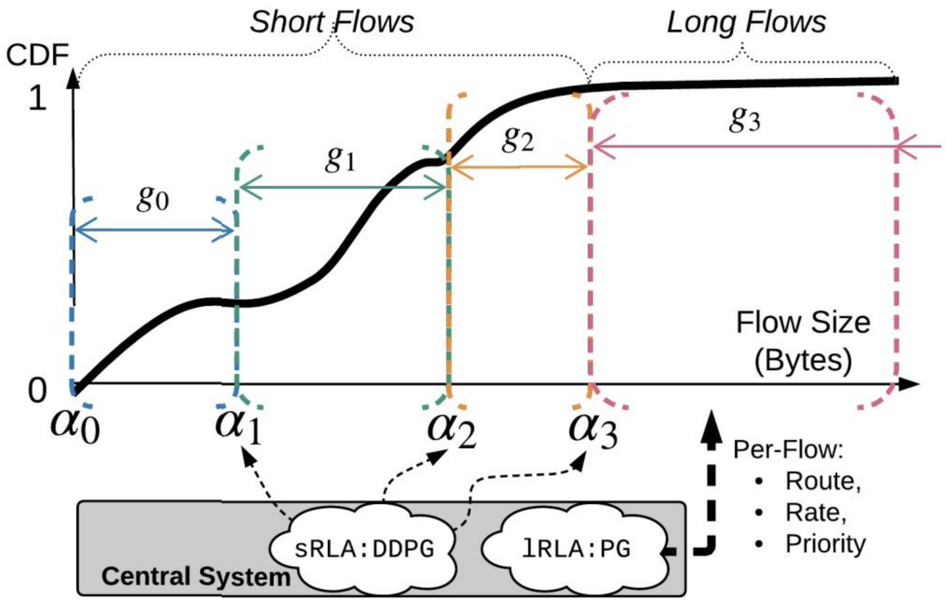
Peripheral System

- Collects and tags flows
  - Tagged actions are influence from Central System
- Monitoring Module
- Enforcement Model
  - Receives actions from central system
  - Traffic Optimizations Decision



Central System

- Uses two RL agents
  - sRLA & IRLA
- sRLA
  - Deep Deterministic Policy Gradient
  - 700 features per-server
  - Outputs MLFQ threshold
- IRLA
  - Generates actions for long flows
  - Fully Connected
  - 10 hidden layers
  - 136 features per-server
  - Outputs probabilities of actions for active flows



sRLA in Depth

- Inspired by staged (SEDA) event driven architecture design
- DDPG
  - Actors have two fully-connected hidden layers
  - Outputs optimizes thresholds for MLFQ
  - Critics are three hidden layers
- Leverages CDF of flow size distributions
- Optimal set of thresholds to minimize FCT (flow completion time)

Algorithm 1: DDPG Actor-Critic Update Step

---

1 Sample a random mini-batch of N transitions (s_i, a_i, r_i, s_{i+1}) from buffer

2 Set $y_i = r_i + \gamma Q'_{\theta^{Q'}}(s_{i+1}, \mu'_{\theta^{\mu'}}(s_{i+1}))$

3 Update critic by minimizing the loss: $L = \frac{1}{N}\sum_{i=1}^{N}(y_i - Q_{\theta^Q}(s_i, a_i))^2$

4 Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_{\theta^\mu}(s_i)\mu_{\theta^Q}(s_i)\nabla_{a_i} Q_{\theta^Q}(s_i, a_i)\Big|_{a_i = \mu_{\theta^Q}(s_i)}$$

5 Update the target networks:

$$\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'}$$

$$\theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'}$$

where γ and τ are small values for stable learning

Environment

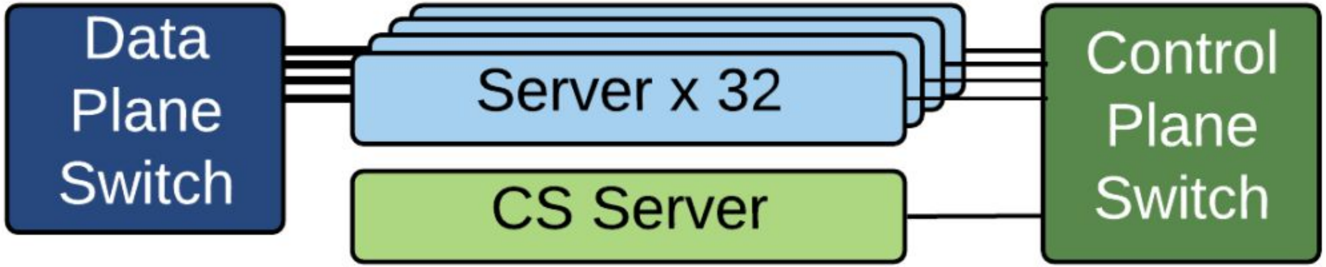


Figure 7: Testbed topology.

Evaluation

- How does AuTO compare to standard Heuristics?
- How does AuTO adapt?
- How fast can AuTO respond?
- What is the system overhead?

System is trained for 8 hours and then compared against generated heuristics

Traffic Distributions

- Characteristics
  - flow size, distribution and load
- Homogeneous
- Spatially Heterogeneous
  - Cluster of for servers with fixed characteristics
- Spatially and Temporally Heterogeneous
  - Characteristics change periodically

Figure 7: Testbed topology.

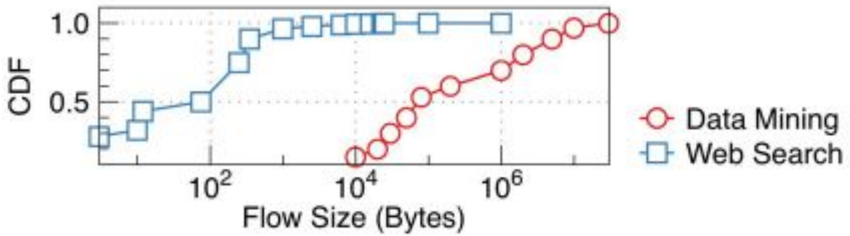
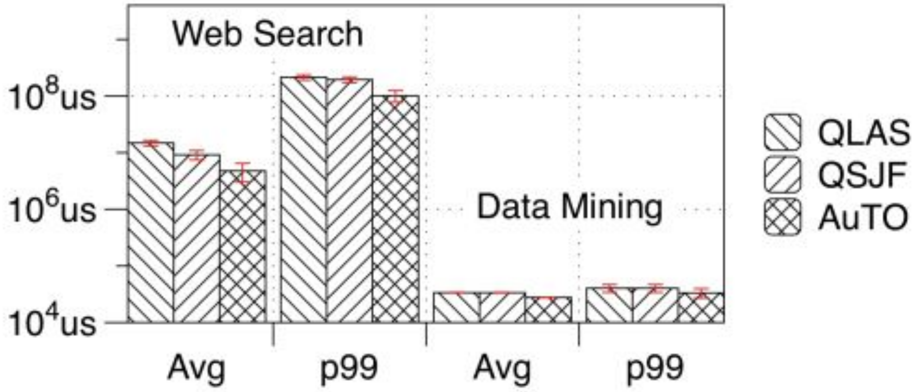


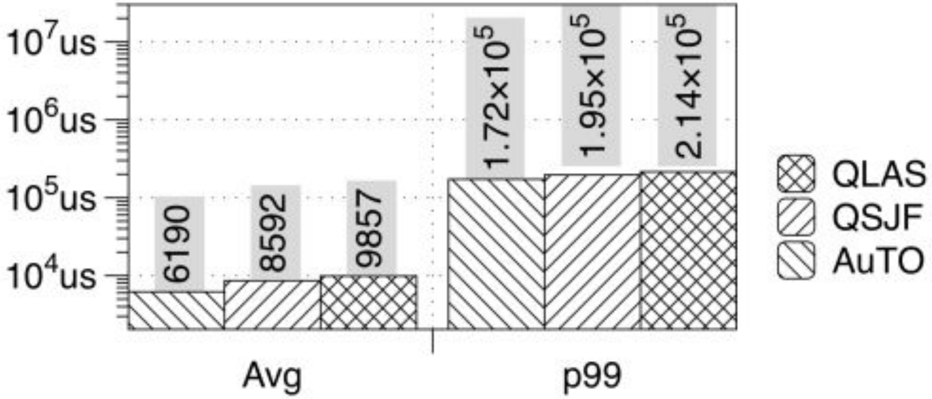
Figure 8: Traffic distributions in evaluation.

Homogeneous Traffic



Average Flow Time Completion vs. Percentile

Spatially Heterogeneous Traffic



Average Flow Time Completion vs. Percentile

Temporally and Heterogenous Traffic

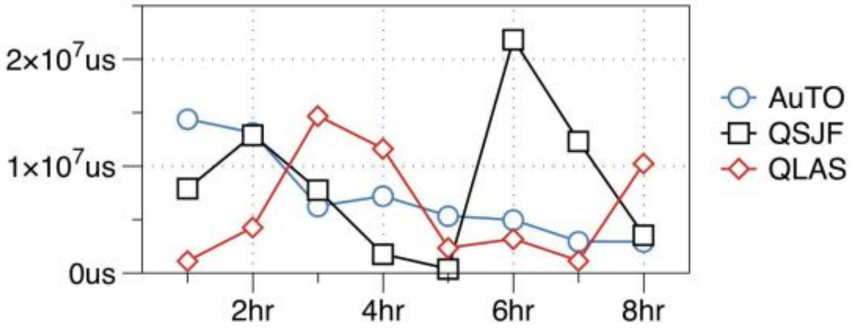


Figure 11: Dynamic scenarios: average FCT.

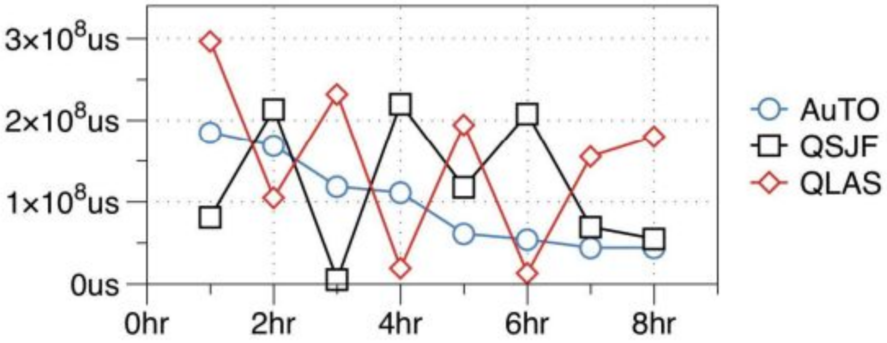


Figure 12: Dynamic scenarios: p99 FCT.

Impact of MLFQ Thresholds on FCT

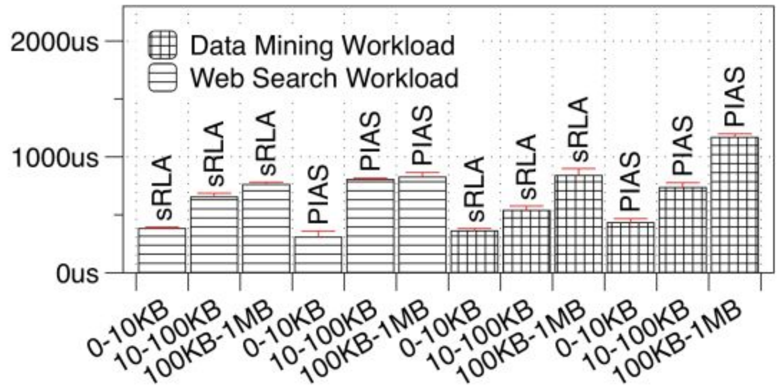


Figure 14: Average FCT using MLFQ thresholds from sRLA vs. optimal thresholds.

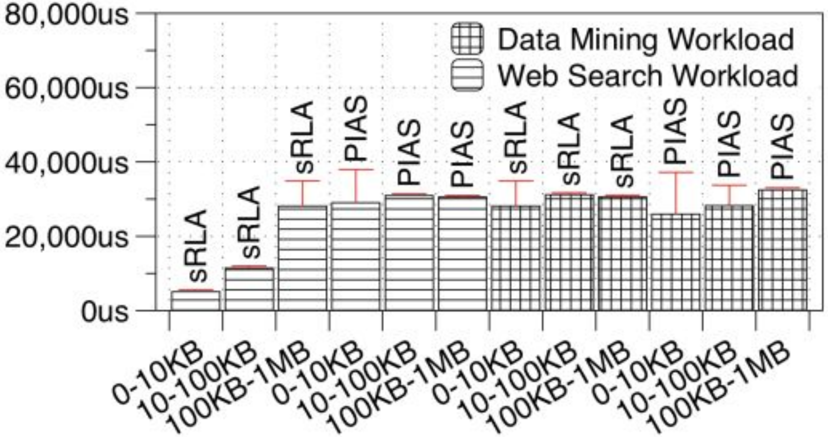


Figure 15: p99 FCT using MLFQ Thresholds from sRLA vs. optimal thresholds.

Load Balancing

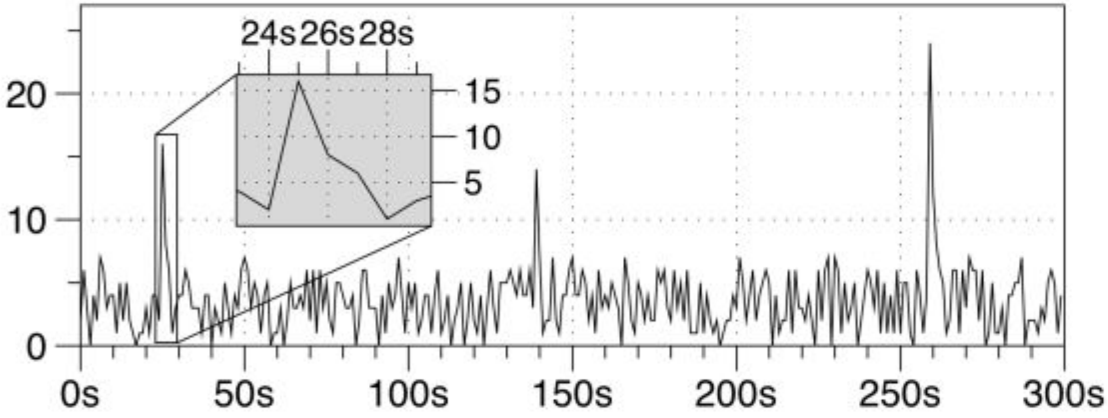


Figure 16: Load balancing using lRLA (PG algorithm): difference in number of long flows on links.

Central Sysetm Latency

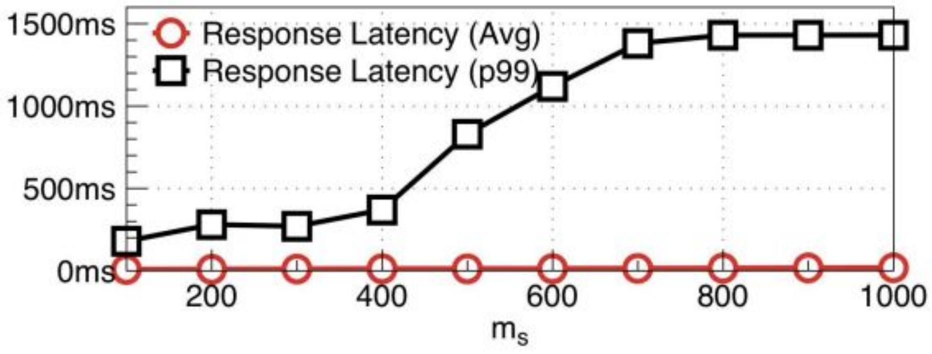


Figure 18: CS response latency: Scaling short flows

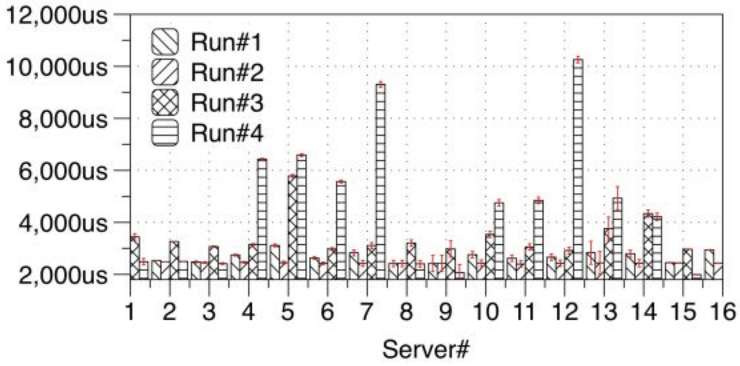


Figure 17: CS response latency: Traces from 4 runs.

Commentary

- Design can be described as over-complicated
  - Does not take into account current network advancements
  - System can leverage abstractions of software defined networking
- Scaling implications of an approach that relies on agents on every server

Reinforcement Learning for Data center Congestion Control

Chen Tessler, Yuval Shpigelman, Gal Dalal, Amit Mandelbaum, Doron Haritan Kazakov, Benjamin Fuhrer, Gal Chechik, and Shie Mannor

Reinforcement Learning in Context

- Congestion Control
  - Requires observibility
  - Multi-objective management
- Problem Structure
  - Multi-agent
  - Multi-objective
  - Partially observed

Contributions

- PCC-RL
    - Capable of maintaining high switch utilization
- OMNeT++ Evaluation Suite
- Testing Agents
    - RL POMDP

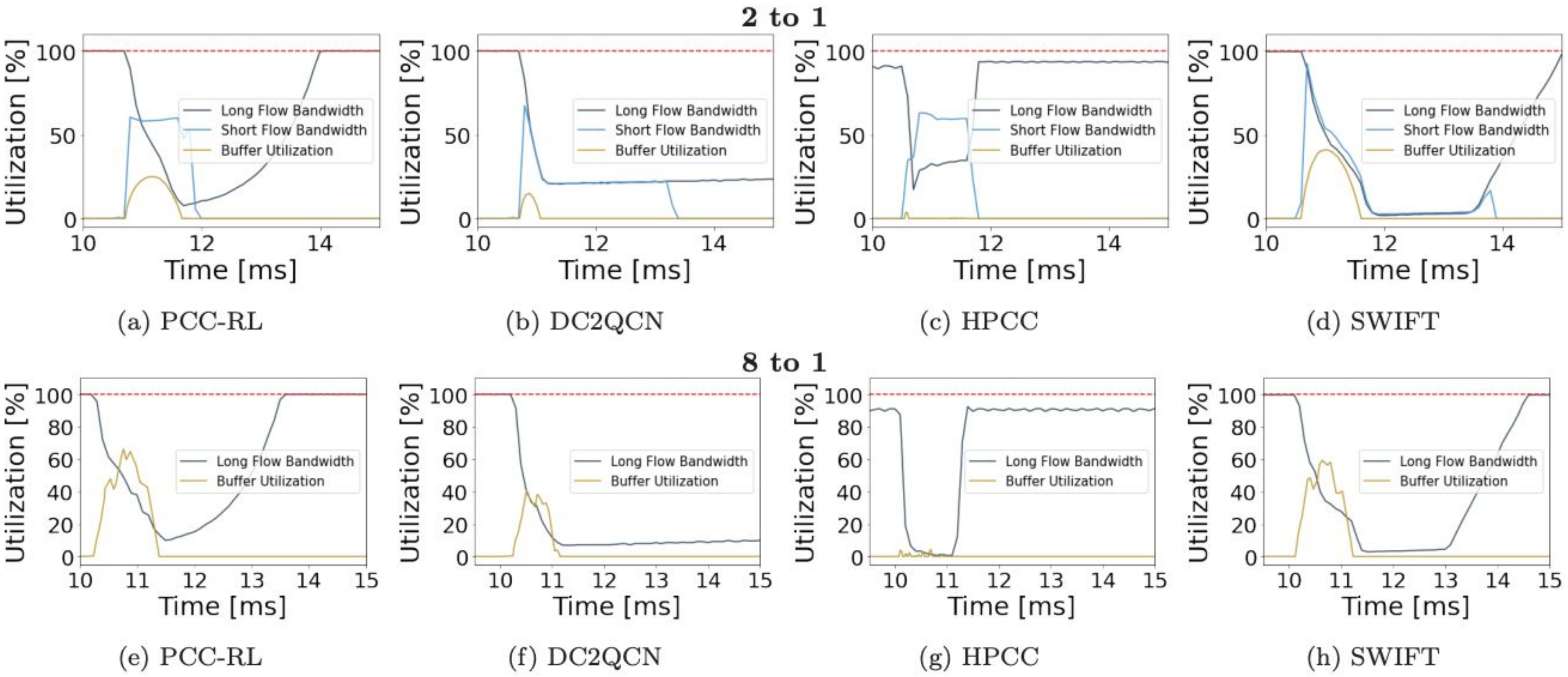
Baseline

Alg.	4 hosts			8 hosts		
	SU	FR	QL	SU	FR	QL
PCC-RL	**94**	**77**	**6**	**94**	**97**	**8**
DC2QCN	**90**	**91**	**5**	91	89	6
HPCC	71	18	3	69	60	3
SWIFT	76	100	11	76	98	13

Alg.	128 to 1				1024 to 1				4096 to 1				8192 to 1			
	SU	FR	QL	DR	SU	FR	QL	DR	SU	FR	QL	DR	SU	FR	QL	DR
PCC-RL	**92**	**95**	**8**	**0**	90	70	15	0	**91**	**44**	**26**	**0**	**92**	**29**	**42**	**0**
DC2QCN	96	84	8	0	**88**	**82**	**17**	**0**	85	67	110	0.2	100	72	157	1.3
HPCC	83	96	5	0	59	48	27	0	73	13	79	0.2	86	8	125	0.9
SWIFT	98	99	40	0	**91**	**98**	**66**	**0**	90	56	120	0.1	92	50	123	0.2

2 to 1

Findings


2 to 1
Utilization [%]
Time [ms]
Long Flow Bandwidth
Short Flow Bandwidth
Buffer Utilization
(a) PCC-RL
(b) DC2QCN
(c) HPCC
(d) SWIFT
8 to 1
Long Flow Bandwidth
Buffer Utilization
(e) PCC-RL
(f) DC2QCN
(g) HPCC
(h) SWIFT