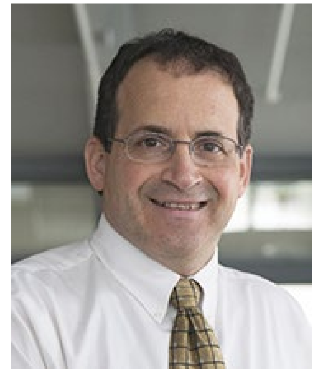
Data-Driven Stochastic Network Control via Reinforcement Learning

Qiaomin Xie, Cornell ORIE

YEQT, June 9th 2021

Devavrat Shah

Zhi Xu



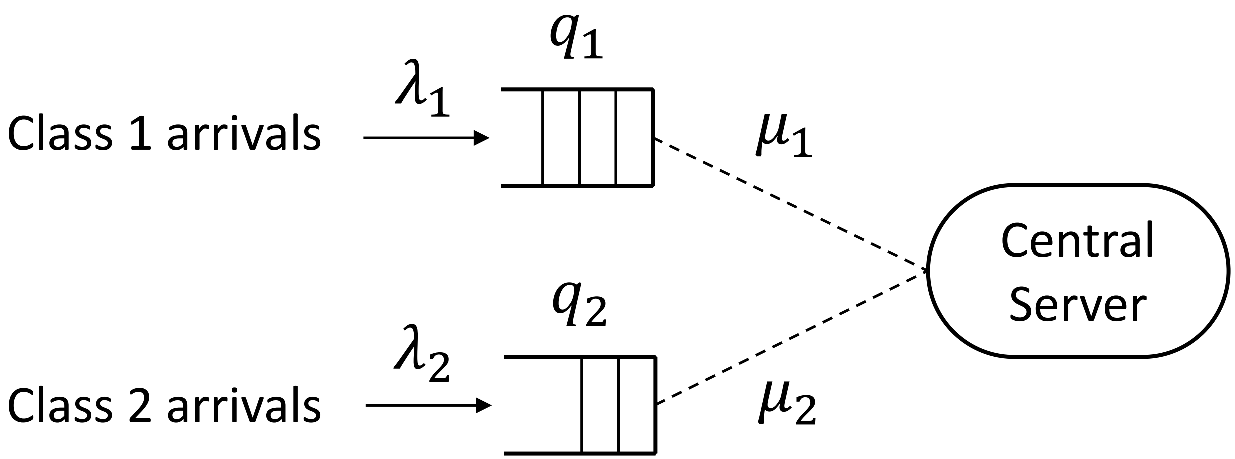
Eytan Modiano

Bai Liu

1. *Stable Reinforcement Learning with Unbounded State Space*, D. Shah, Q. Xie, Z. Xu. Learning for Dynamics & Control (L4DC) Conference, 2020.
2. *RL-QN: A Reinforcement Learning Framework for Optimal Control of Queueing Systems*, B Liu, Q. Xie, E Modiano. In Allerton Conference, 2019.
3. *Non-Asymptotic Analysis: Monte Carlo Tree Search*, D. Shah, Q. Xie, Z. Xu. In SIGMETRICS, 2020.

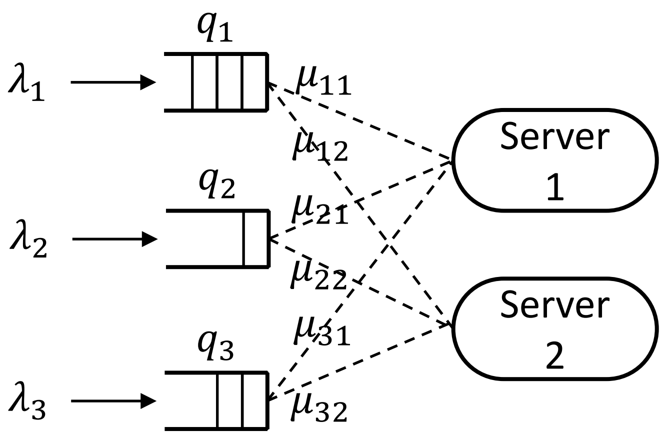
❑ Longer talks for 1 & 2: In SIGMETRICS workshop "Reinforcement Learning in Networks and Queues", June 14, 2021

Example I: Multi-Class Single Server

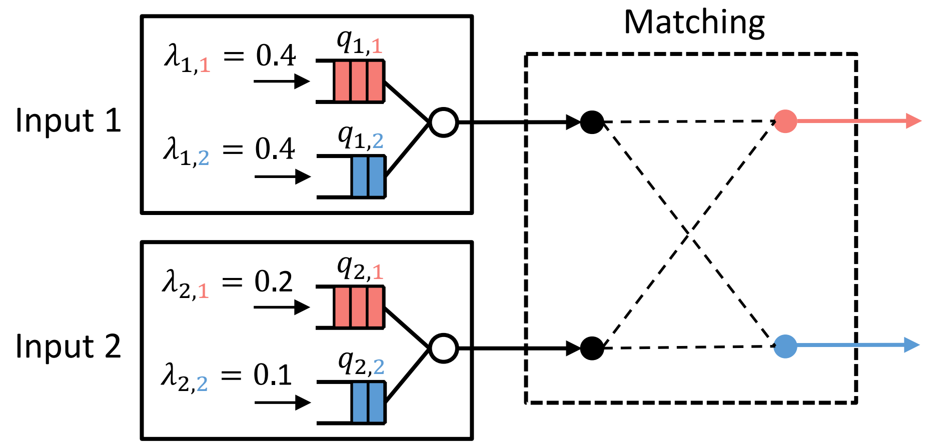


- A discrete-time system with two *infinite*-buffer queues
  - Unbounded state space: $q = (q_1, q_2) \in \mathbb{N} \times \mathbb{N}$
- Scheduling decision/action
  - $A = \{1,2\}$, i.e., which queue to serve
- Goal: minimize average total queue length (i.e., delay)

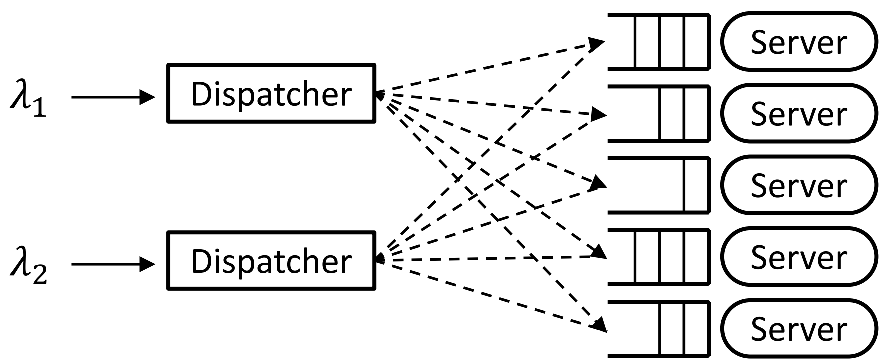
Example II: Multi-Class Parallel-Servers



Parallel server scheduling



Switch scheduling



Routing/load balancing

Challenges

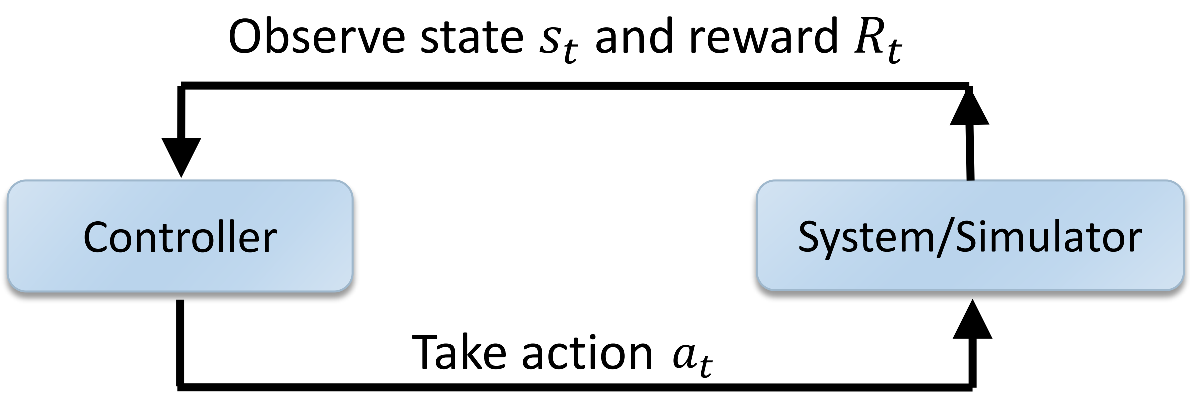
- "Model-driven" approach
  1. Accurate stochastic modeling of system
  2. Rely on intuition or a flash of genius to guess a good algorithm
  3. Test/tune the algorithm
  4. Prove performance guarantees

- Challenge 1: Lack of accurate models
  - Unknown system parameters
  - Time-varying dynamics

- Challenge 2: Optimal policies difficult to find
  - Even for simple, known models
  - More so for: jobs with multiple dependent tasks, heterogeneity of servers/jobs, general service time, etc.

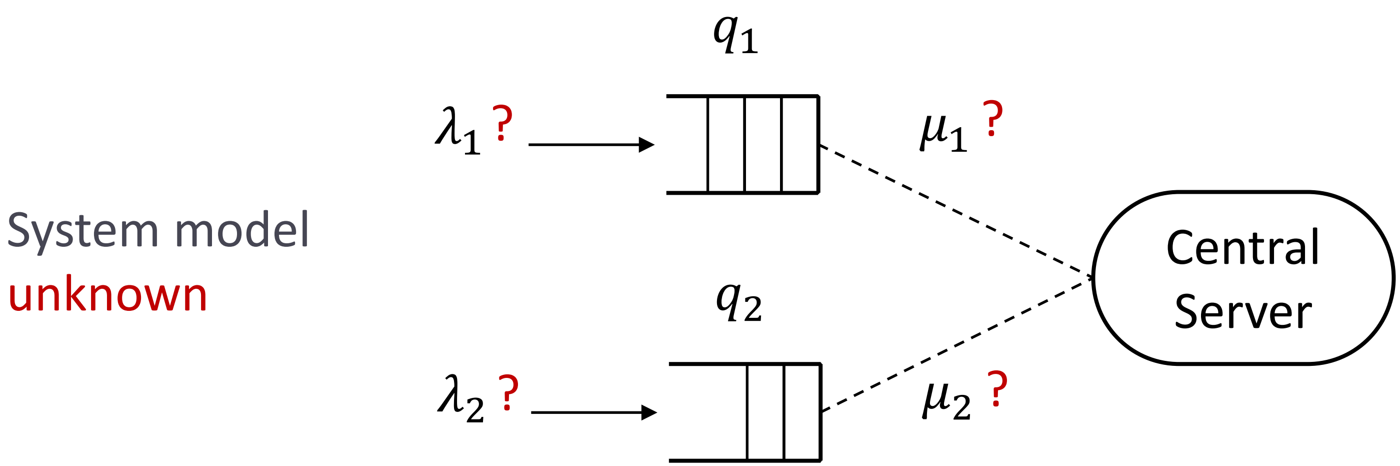
Data-Driven Approach

- Opportunity: availability of fine-grained data or system-level simulators
- A data-driven framework: **Reinforcement Learning** (RL)



- Challenge 1: Lack of accurate models
  - **Learn system dynamics from data**
- Challenge 2: Optimal policies difficult to find
  - **Discover new policies**

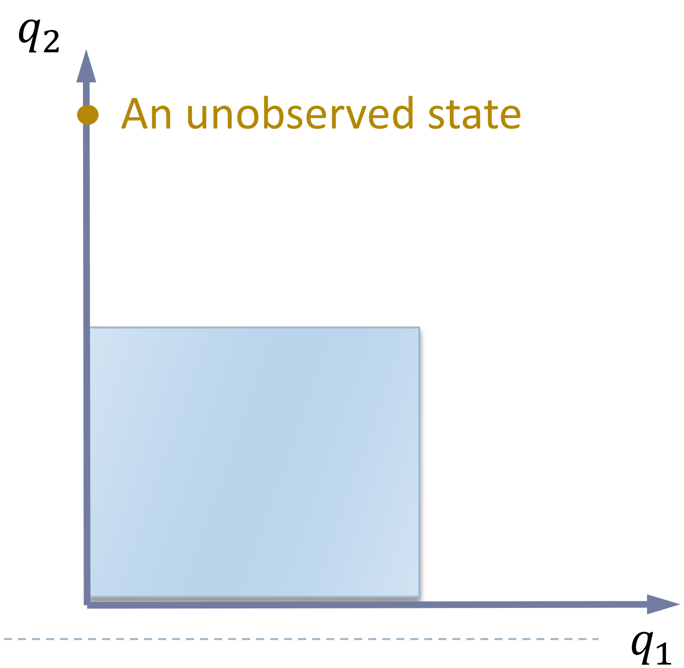
RL for Learning to Schedule



- Learn to schedule purely from data
  - System model unknown
- Optimize a given criterion
  - e.g., minimize average/discounted queue lengths
- Key characteristic: **unbounded state space**
  - e.g., $q = (q_1, q_2) \in \mathbb{N} \times \mathbb{N}$

Challenges of Unbounded State Space

- Insufficiency of ***offline***-training-then-deploy
  - Using finite samples
  - Reach a previously **unobserved state**
  - Might have undesirable behavior
    - e.g., serving an empty queue while the other queue is large; assign slow server to busy queue



- Require ***online training***: decide action when encountering new states

Summary of Our Results

- A notion of *stability* to quantify "goodness" of RL algorithm
  - Applies to general systems with unbounded state space
  - Stability provides a first-order optimality

- An *online* RL algorithm that achieves stability
  - Sample complexity bounds

- From stability to optimality

Markov Decision Process (MDP)

- Infinite horizon discounted MDP: (S, A, p, R, γ)
  - S: **unbounded** state space
  - A: **finite** action space
  - $p(s'|s, a)$: transition kernel
  - $R(s, a)$: one-stage reward
  - $\gamma \in (0,1)$: discount factor

- (Stochastic) Policy π: $S \rightarrow \Delta(A)$
- State-action value function (Q-function)

$$Q^{\pi}(s, a) = E_{\pi}\left[\sum_{t=0}^{\infty} \gamma^{t} R(s_t, a_t) \mid s_0 = s, a_0 = a\right]$$

- Optimal Q-function

$$Q^{*}(s, a) = \max_{\pi} Q^{\pi}(s, a)$$

Stability

- Minimizing queue length requires keeping queue length finite
- **Stability** is a necessary first step towards optimality

Definition (Stability)

We call a policy $\{\pi_t\}$ stable, if $\forall\theta \in (0,1)$, there exists a bounded set $S(\theta) \subset S$ s.t.

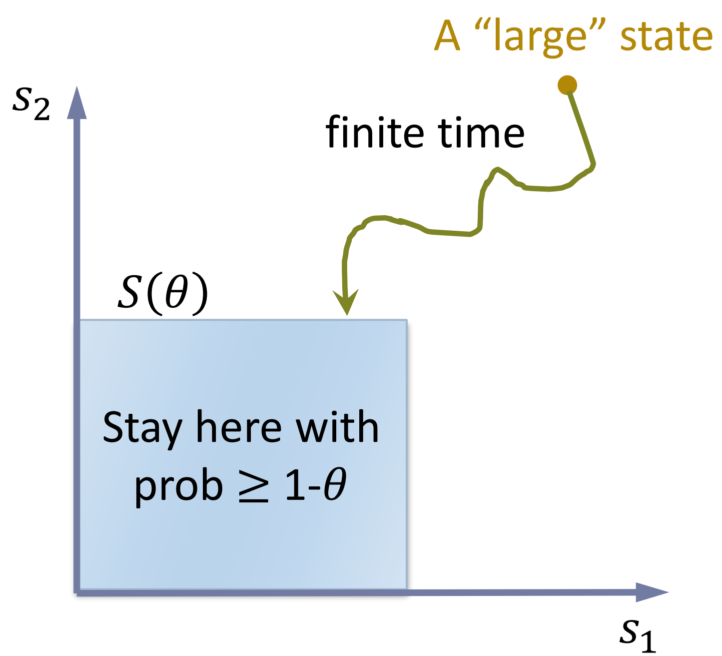
1. **Boundedness:**

$$\liminf_{t\to\infty} \mathbb{P}(s_t \in S(\theta) | s_0 = s) \geq 1 - \theta, \forall s \in S.$$

2. **Recurrence:**

Let $\boldsymbol{T(s,t,\theta)} = \min\{k \geq 0: s_{t+k} \in S(\theta) | s_t = \mathbf{s}\}$,

$$\sup_t \mathbb{E}[\boldsymbol{T(s,t,\theta)} | s_t = s] < \infty, \forall s \in S.$$

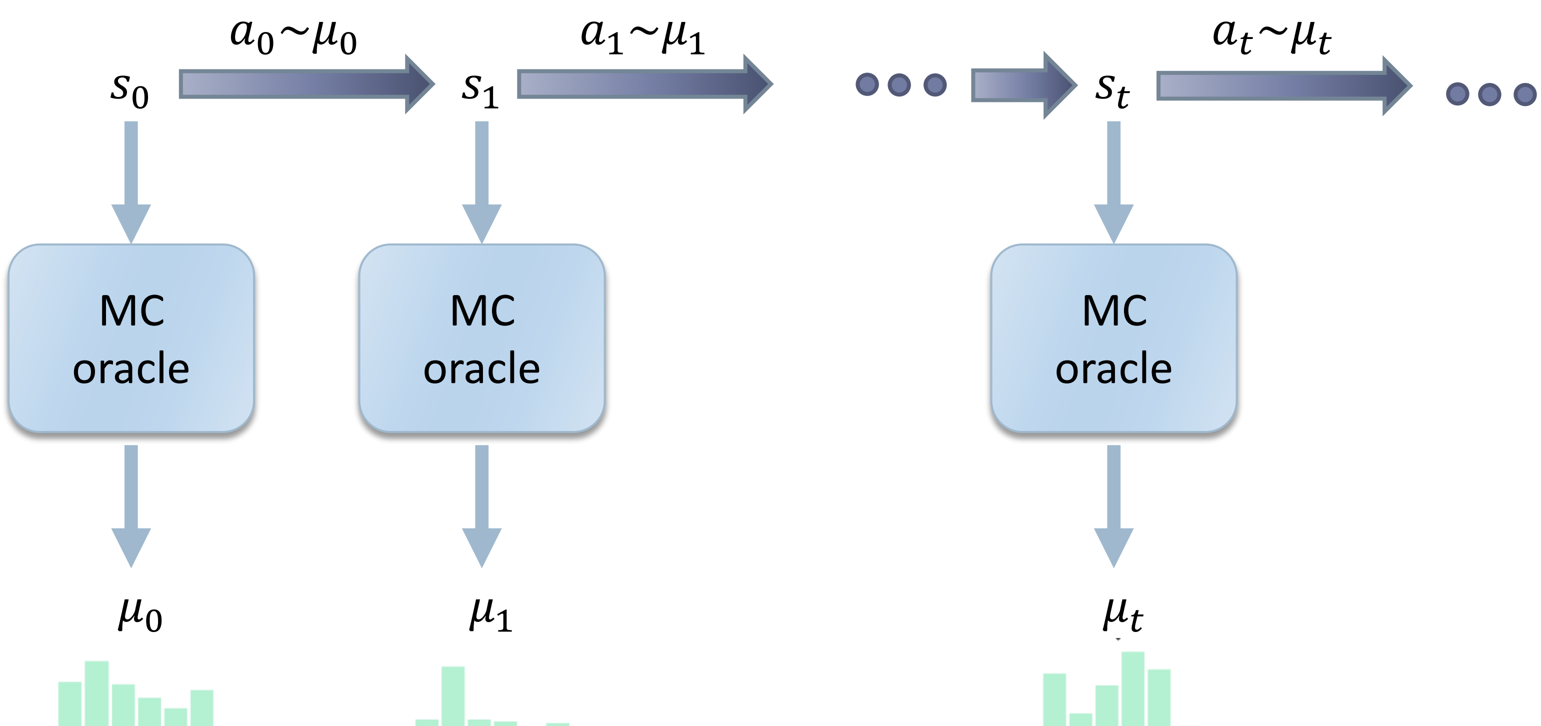


Question: For unbounded state space, how to learn a stable policy in a data-driven manner?

We use a Monte-Carlo simulation and search method.

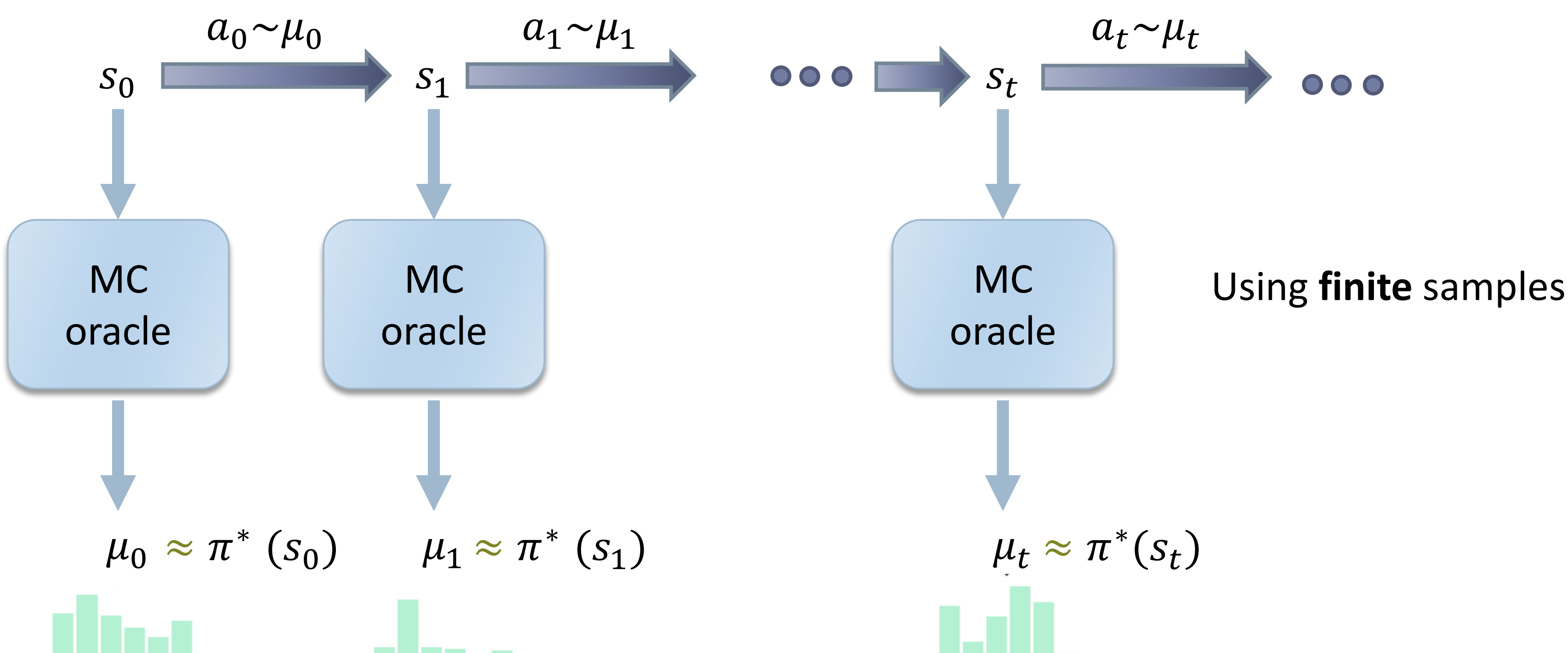
A Monte Carlo Approach

- At each time step t
  - Query a Monte Carlo (MC) oracle
    - Input: **state** s_t
    - Output: a probability distribution over actions μ_t
  - Take action $a_t \sim \mu_t$ and reach state s_{t+1}



A Monte Carlo Approach

- At each time step t
  - Query a Monte Carlo (MC) oracle
    - Input: **state** s_t
    - Output: a probability distribution over actions μ_t
  - Take action $a_t \sim \mu_t$ and reach state s_{t+1}



Monte Carlo Oracles

- Sparse-Sampling Oracle [Kearns-Mansour-Ng, '02]
- Monte Carlo Tree Search [Kocsis-Szepesvari, '06] [Shah-X-Xu, '20]

- Oracle Approximation Guarantees for MCTS

Theorem [Shah-X-Xu '20]

With appropriate parameters, with probability at least $1-\delta$,

$$\left|\hat{Q}(s,a) - Q^*(s,a)\right| \leq \varepsilon, \forall a.$$

Corollary

With softmax policy $\mu(s,a) \propto e^{\hat{Q}(s,a)/\tau}$, we have

$$\|\mu(s,\cdot) - \pi^*(s)\|_{TV} \leq c_1 \frac{e^{\varepsilon/\tau} - 1}{e^{\varepsilon/\tau} + 1} + c_2 e^{-\frac{c_3}{\tau}},$$

where $c_1, c_2, c_3 > 0$ are constants.

Can be small with small ε and τ

[Shah-X-Xu, '20] "Non-Asymptotic Analysis of Monte-Carlo Tree Search", ACM SIGMETRICS, 2020. Under revision for OR.

From Approximation to Stability

- Questions:
  - When is the policy $\{\mu_t\}$ stable?
  - What is the sample complexity of each oracle query?

- When is stability possible
  - The Markov chain M^* under the optimal policy π^* is positive recurrent
  - A *necessary* and *sufficient* condition for positive recurrence of a Markov chain is the existence of a Lyapunov function[1]
  - We assume that M^* satisfies a Lyapunov Drift Condition

[1] Jean-Franc¸ois Mertens, Ester Samuel-Cahn, and Shmuel Zamir. Necessary and sufficient conditions for recurrence and transience of Markov chains, in terms of inequalities. Journal of Applied Probability, 15(4):848–851, 1978.

Assumption: Lyapunov Function

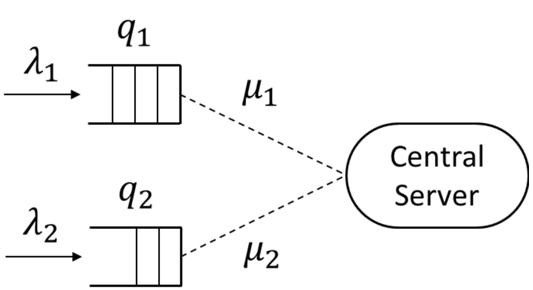
Assumption

There exists a function $L: S \to \mathbb{R}^+$ such that the Markov chain under π^* satisfies that

(1) change of L for any transition is bounded,

(2) has a negative drift $-\alpha$ when $L(s) > B$.

- *Example*: single-server two-queue system
  - Optimal policy π^*: $c\mu$ rule
  - $L(q_1, q_2) = \frac{q_1}{\mu_1} + \frac{q_2}{\mu_2}$ satisfies the assumption



- **Remark:** Algorithm *not* need to know the Lyapunov function

Main Results

Theorem (Stability)

Under the Lyapunov assumption, with proper parameters, the resulting policy $\{\mu_t\}$ sequence is stable.

Theorem (Sample complexity)

Sample complexity per time step for small α scales as

$$O\left(\left(\frac{1}{\alpha^4}\log\frac{1}{\alpha}\right)^{\log\frac{1}{\alpha}}\right)$$

Refinements

- Adaptive version
  - Automatically discover the appropriate tuning parameters ε, τ
  - Using a statistical hypothesis test for growing queue length

- Sample-efficient version
  - Small α: high load regime in queueing
  - From super-polynomial to polynomial

$$O\left(\left(\frac{1}{\alpha^4}\log\frac{1}{\alpha}\right)^{\log\frac{1}{\alpha}}\right) \to O\left(\frac{1}{\alpha^{2d+4}}\right)$$

From Stability to Optimality

- Given a stable policy, can we learn the optimal policy? Yes!

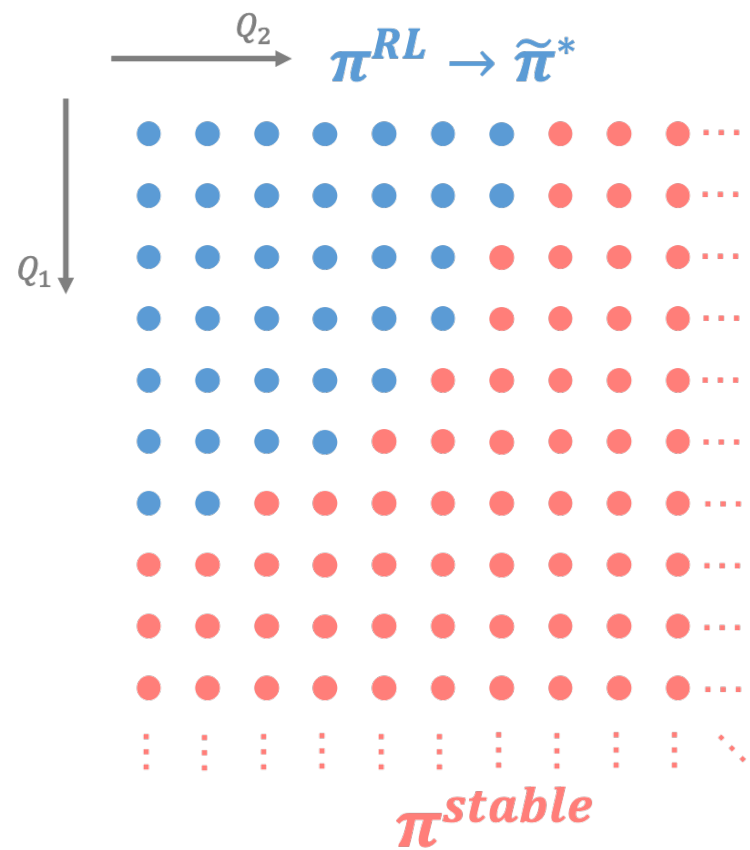
Our Approach:

For the outside states

- Apply default stable policy π^{stable}
  - e.g. policy we already know
  - Or, use stable RL
- Cost for other states can be controlled

For the truncated state space

- Apply model-based RL policy π^{RL}
- Converge to optimal policy $\tilde{\pi}^*$



Theoretical Guarantee

Theorem [Liu-X-Modiano, '19]

Under Lyapunov assumption, with state space truncated at U,

$$\lim_{T\to\infty} \frac{\mathbb{E}[\sum_{t=1}^{T} \sum_i Q_i(t)]}{T} = \rho^* + \mathcal{O}\left(\frac{1}{\exp(U)}\right)$$

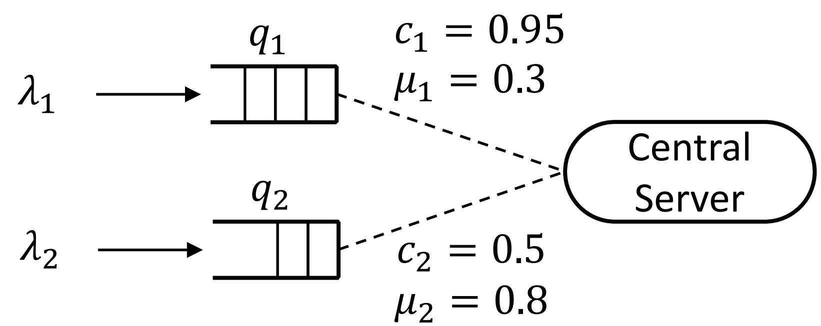
Average queue length of our algorithm

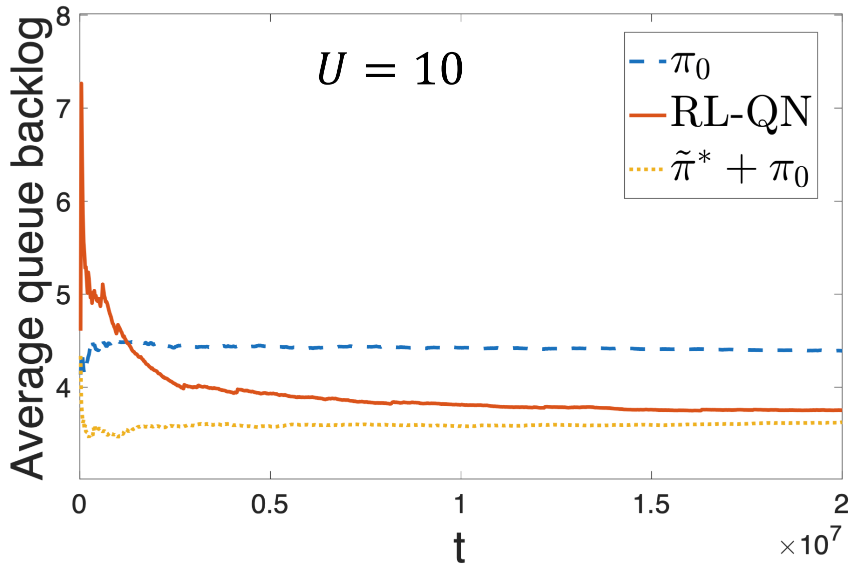
Optimal queue length

Approach optimal performance exponentially fast

B Liu, Q. Xie, E Modiano. RL-QN: A Reinforcement Learning Framework for Optimal Control of Queueing Systems. Allerton '19

Simulation: Scheduling with Connectivity





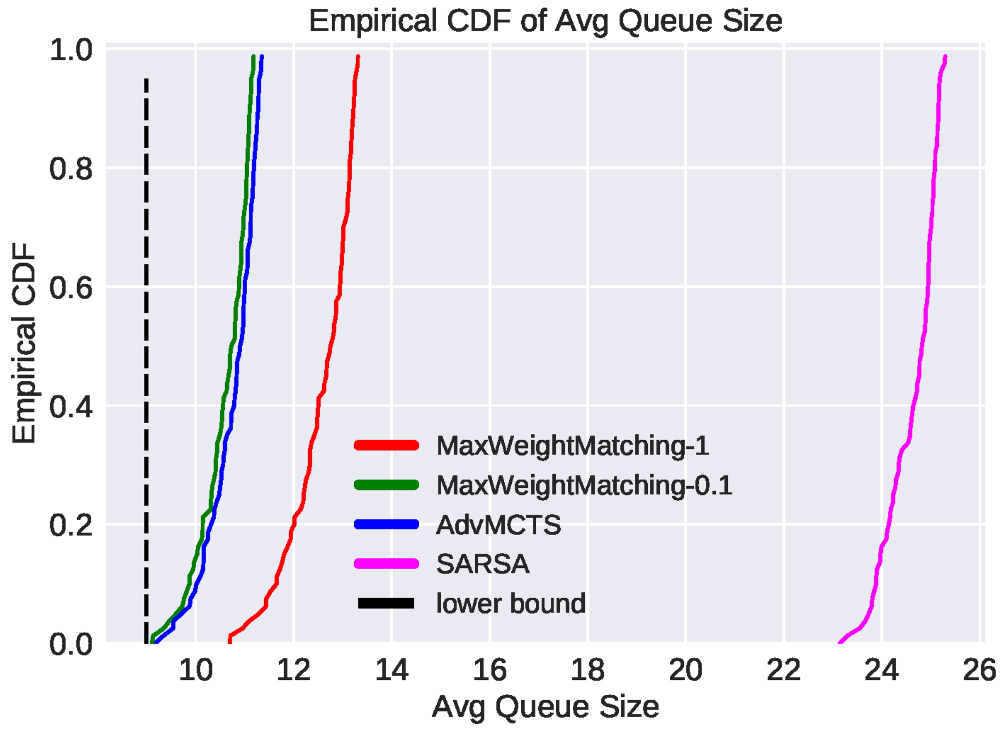
- π_0: Serve-Longest-Connected Queue
- U=5: converge to 3.93
- U=10: converge to 3.75

[1] Tassiulas, Leandros, and Anthony Ephremides. "Dynamic server allocation to parallel queues with randomly varying connectivity." *IEEE Transactions on Information Theory* 39.2 (1993): 466-478.

[2] Ganti, Anand, Eytan Modiano, and John N. Tsitsiklis. "Optimal transmission scheduling in symmetric communication models with intermittent connectivity." *IEEE Transactions on Information Theory* 53.3 (2007): 998-1008.

RL as Performance Benchmarks

- RL methods can achieve state-of-art performance for complex stochastic network control problems
- Switch Scheduling:



RL as Performance Benchmarks

- OR-Suites: OR version of OpenAI gym
  - Ongoing with S. Sinclair, C. Lee Yu and S. Banerjee
  - RL Benchmarks for operations research applications
    - Rideshare matching
    - Ambulance routing
    - Revenue management
    - Foodbank allocation
    - …

- Demo at RLNQ Workshop

Summary

- RL for stochastic network control with unbounded state space
  - A notion of stability suitable for RL setting
  - Achieve stability by Monte Carlo planning
  - From stability to optimality

- Future work
  - Combined with function approximation and policy optimization
  - Complex stochastic networks: synthesizing model-driven & data-driven approaches

Thank you!