Learning Unknown Service Rates in Queues: A MAB Approach

Haricharan

IITM

Our model (single-queue)

- Consider inputs as well as services to be Bernoulli iid's
- We have an input queue, with arrival probability some λ
- We have K servers, each with some service probability $\vec{\mu}$ **not known a-priori**. $\lambda < \mu_{max}$
- In each timestep, we have to choose one server

Some random variables

- Consider the random variable for arrival $A(t)$, random variable for service by the chosen server $S(t)$
- A queue builds up here, of length $Q(t)$ (reward function)
- $\boxed{Q(t) = \max(0, Q(t-1) + A(t) - S(t))}$
- We attempt to minimize the queue-regret: $\Psi(t) = E[Q(t) - Q^*(t)]$
- As usual, we observe the exploration-exploitation tradeoff

Two stages

Characterized by a *phase transition*

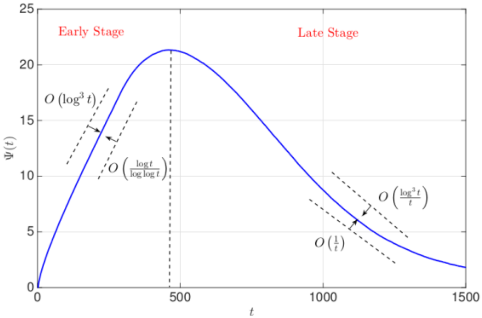

Figure 1: Plot of Early and Late Stages

- Early Stage: characterized by *forced exploration*, needed to achieve stability (i.e. enter cyclic stage)
- Late Stage: characterized by regenerative cycles in which queue regret goes to zero cyclically

Characterizing the Late Stage

$\alpha-$consistent policy: Amount of time we pick sub-optimal server should be sublinear

- Upper bound: $\boxed{O\left(\frac{\log^3(t)}{t}\right)}$, by forced exploration, i.e. Q-UCB, Q-ThS
- Lower bound: $\boxed{O\left(\frac{1}{t}\right)}$, by $\alpha-$consistency

Characterizing the Early Stage

- Upper bound: $\boxed{O(\log^3(t))}$
- Lower bound: $\boxed{O\left(\frac{\log(t)}{\log(\log(t))}\right)}$, by $\alpha-$consistency

Q-UCB

Algorithm 1 Q-UCB

At time $t \geq 1$,

Let $\mathsf{E}(t)$ be an independent Bernoulli sample of mean $\min\{1, 3K\frac{\log^2 t}{t}\}$.

if $\mathsf{E}(t) = 1$ **then**

Explore:

Schedule a matching from $\mathcal{X}$ uniformly at random.

else

Exploit:

Compute for all $u \in [U]$

$$\hat{k}_u(t) := \arg\max_{k \in [K]} \hat{\mu}_{uk}(t) + \sqrt{\frac{\log^2 t}{2T_{uk}(t-1)}}.$$

Schedule a matching $\boldsymbol{\kappa}(t)$ such that

$\hat{\kappa}$

$$\boldsymbol{\kappa}(t) \in \arg\min_{\boldsymbol{\kappa} \in \mathcal{M}} \sum_{u \in [U]} \mathbb{1}\left\{\kappa_u \neq \hat{k}_u(t)\right\},$$

end if

Multiple Queues

- U $(U \leq K)$ input queues, with arrival parameters $\vec{\lambda}$
- We have to find the optimal matching in the graph
- It is guaranteed that optimal server for each queue is distinct
- We have U many queue lengths: $E[\vec{Q}(t)]$

Future ideas

- Answer the question when optimal servers are not distinct, and $U \geq K$
- Some thoughts: depends on if we minimize $E[\vec{Q}(t)]$, one-norm, two-norm, etc.
- How does the steady state look in this case?
- Late stage: can the $\log^3$ factor be reduced? Can we avoid forced exploration? (suggested in the paper)

Quantum Queue-Channel

- Application of Crowdsourcing mentioned in the paper
- Try GI/GI/1 or M/G/k Queues
- Quantum Codes?
- Additive Quantum Code
- Characterize when upper bound will depend on knowledge on waiting time (done already?)