Receding Learning-aided Control in Stochastic Networks

Longbo Huang

longbohuang@tsinghua.edu.cn

IIIS, Tsinghua University *

Abstract

In this paper, we develop the *Receding Learning-aided Control* algorithm (`RLC`) for solving optimization problems in general stochastic networks with potentially *non-stationary* system dynamics. `RLC` is a low-complexity online algorithm that requires *zero* a-priori statistical knowledge. It has three main functionalities. First, it detects changes of the underlying distribution of system dynamics via receding sampling. Then, it carefully selects the sampled information and estimates a Lagrange multiplier of an underlying optimization problem via dual-learning. Lastly, it incorporates the multiplier into an online system controller via drift-augmentation. We show that `RLC` achieves near-optimal utility-delay tradeoffs for stationary systems, while ensuring an effeicient distribution-change detection and a fast convergence speed when applied to non-stationary networks. The results in this paper provide a general framework for designing joint detection-learning-control algorithms and provide new understanding about the role-of-information and the power-of-online-learning in network control.

1 Introduction

We consider the following constrained network optimization problem. We are given a stochastic network with a dynamic system state that evolves according to some *potentially non-stationary* probability law. Under each system state, a control action is chosen and implemented. The action generates traffic into the network queues but also serves workload from them. The action also results in a system cost due to resource expenditure. The traffic, service, and cost are jointly determined by the action and the system state. The objective is to minimize the expected cost given traffic/service constraints. This is a general framework that models many practical scenarios, for instance, computer networks, supply chains, mobile networks, and smart grids. Hence, it is one of the central problems in network research to develop efficient control techniques for this framework. In particular, it is most desirable for the techniques to (i) provide strong explicit utility and

*This work was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61361136003, 61303195, Tsinghua Initiative Research Grant, Microsoft Research Asia Collaborative Research Award, and the China Youth 1000-talent Grant.

delay guarantees, (ii) possess fast convergence speed, and (iii) be able to quickly detect changes and adapt to the dynamic environment.

However, this is a very challenging problem. First of all, statistical information of the system dynamics is often unknown a-priori. Hence, in order to achieve optimal performance, algorithms must be able to efficiently learn certain sufficient statistics of the dynamics. Second, since every time only a single random state will appear, algorithms must be able to handle individual realization of the system randomness, which often requires algorithms to be incremental. Third, to provide explicit delay guarantees, algorithms usually need to have explicit queue-like interpretations of the control steps. However, such algorithms often suffer from a slow convergence speed.

There has been continuous effort in developing algorithms that can achieve good utility and delay performance for various networks, for instance, wireless networks, [1], [2], [3], processing networks, [4], [5], cognitive radio, [6], and the smart grid, [7], [8]. However, we notice that most existing algorithms focus on dynamic systems with stationary distributions, and they either assume full system statistical information beforehand, or rely on stochastic approximation techniques to avoid the need of such information. Thus, both approaches ignore the information and system observation aspects. As a result, they fail to provide deep understanding about the *role-of-information* in network control and do not explore the *power-of-learning*. This ignorance inevitably results in a *mismatch* between many control algorithms developed in the literature and control schemes in practical systems, under which system dynamics are often constantly monitored and such information is explicitly incorporated into control.

In this work, we develop the *receding learning-aided control* algorithm (`RLC`). `RLC` is an online algorithm which requires *zero* a-priori statistical knowledge of the system. Instead, it continuously updates its estimates of the underlying distribution of the dynamics via receding sampling of the observed system history, and efficiently incorporates the information into the control component via learning an optimal Lagrange multiplier of a carefully constructed optimization problem and drift-augmentation [9]. The receding learning feature enables a quick self-adaptation to the changing dynamics, while the learning and augmentation steps explicitly explore the benefits of utilizing historic information in control. By carefully integrating the components, `RLC` strikes a good balance among detection speed, learning accuracy, and algorithm performance. Specifically, we show that `RLC` guarantees the near-optimal $[O(\epsilon), O(\log(1/\epsilon)^2)]$ utility-delay tradeoff $(0 < \epsilon < 1)$ when applied to stationary systems with discrete action space, while guaranteeing a detection speed of $O(\epsilon^{-2/3})$ and a convergence time of $O(\epsilon^{-2/3}\log(1/\epsilon)^2)$ (the time it takes for the algorithm to converge to its optimal operating point) when the system is non-stationary. For systems with continuous action sets, we similarly show that `RLC` achieves the near-optimal $[O(\epsilon), O(\sqrt{1/\epsilon}\log(1/\epsilon)^2)]$ utility-delay tradeoff for stationary systems, and ensures an $O(\epsilon^{-4/3})$ detection time and an $O(\epsilon^{-4/3}\log(1/\epsilon)^2)$ algorithm convergence

time for the non-stationary counterparts. In both cases, RLC's convergence times are better than previous known results, i.e., $O(\epsilon^{-1})$ for discrete systems and $O(\epsilon^{-3/2})$ for continuous systems, respectively, and it offers explicit distribution change detection results for non-stationary systems.

Closest to our work are two recent works [9] and [10]. Specifically, [9] explores the possibility of joint learning and control, but the resulting algorithms only apply to stationary systems. [10] considers an online optimization setting very different from ours, and proposes a prediction-based algorithm to jointly optimize algorithm competitive ratio and regret. Our work explicitly considers the non-stationarity in dynamic systems and proposes a learning-aided control algorithm that quickly learns the underlying distribution and adapts its actions. The results in this paper provide a systematic approach for designing joint detection-learning-control algorithms for dynamic systems, and offer new insight into understanding the role-of-information and learning in network control.

The main contributions of the paper are summarized as follows:

- We propose a general framework for joint detection-learning-control algorithm design, and develop the *receding learning-aided control* (RLC) algorithm for potentially non-stationary dynamic systems. RLC is an online algorithm that requires zero a-priori statistical knowledge. It quickly detects changes in system dynamic statistics via receding sampling, and efficiently incorporates learned system information into network control via dual learning and drift-augmentation.
- For stationary systems, we show that RLC guarantees a near-optimal $[O(\epsilon), O(\log(1/\epsilon)^2)]$ utility-delay tradeoff when applied to systems with discrete action space ($0 < \epsilon < 1$). For systems with continuous action sets, we show an achievement of the near-optimal $[O(\epsilon), O(\epsilon^{-1/2}\log(1/\epsilon)^2)]$ utility-delay tradeoff.
- For non-stationary systems, we show that RLC guarantees a detection time of $O(\epsilon^{-2/3})$ and an algorithm convergence time of $O(\epsilon^{-2/3}\log(1/\epsilon)^2)$ for discrete systems, and achieves an $O(\epsilon^{-4/3})$ detection time and an $O(\epsilon^{-4/3}\log(1/\epsilon)^2)$ convergence time for continuous systems. In both cases, RLC offers novel distribution change detection results and its convergence times are better than previous known results, i.e., $O(\epsilon^{-1})$ for discrete systems and $O(\epsilon^{-3/2})$ for continuous systems, respectively.

The rest of the paper is organized as follows. In Section 2, we discuss a few motivating examples in diverse application fields and the related works. We set up our notations in Section 3, and present the system model and problem formulation in Section 4. Background information is provided in Section 5. Then, we present RLC in Section 6, and prove its performance in Section 7 and convergence in Section 8. Simulation results are presented in Section 9, followed by conclusions in Section 10.

2 Motivating Examples

In this section, we present a few interesting practical scenarios where our framework can be applied.

Mobile Networks: Consider a mobile user sending data to a base-station (BS). The channel condition (state) between the user and the BS is time-varying, which requires a different amount of power for packet transmission (cost) at different time. Due to requirements from higher layer applications, the user has to deliver a set of flows at given rates. The objective of the user is to find a joint power allocation and scheduling policy, so as to minimize the average energy consumption, while meeting the rate constraints, e.g., [3], [6]. This example can be generalized to include other factors such as modulation/coding, or other objectives.

Crowdsourcing: In a crowdsourcing application, e.g., crowdsourcing map construction [11], tasks enter the system and are assigned to crowd-workers by a server. Depending on workers' qualifications, types of jobs, and job requestors' current requirements, i.e, whether a requestor is in a hurry due to job deadline (state), task requestors receive reward (utility), e.g., satisfaction, upon job completion, and workers receive payments (cost). The objective of the server is to find an assignment scheme to maximize the average system utility minus cost.

Smart Grids: Consider a system operator trying to fulfill a set of flexible users' power demand, e.g., charging an EV, by allocating available renewable energy, e.g., solar power, or by purchasing power from the grid (cost) if the renewable is not sufficient. The available renewable energy evolves according to some time-varying process (state), and the power prices change over time (state). The operator's objective is to design a joint power procurement and scheduling scheme to minimize the average expenditure while meeting the average power demand of the users. Other important components in smart grid, e.g., demand response [12], scheduling of deferrable load [13], and storage management [8], can also be included in this example.

Cloud Computing: Consider an operator, e.g., a dispatcher, trying to assign jobs to servers for processing. The job arrival process is time-varying (state), and available processing capacities at servers may also be dynamic (state), e.g., due to background processing. Completing user's job requests brings the operator reward (utility). The goal is to allocate resources and balance the loads in such a way that the system utility is maximized. This example can be extended to capture other factors such as rate scaling [14] and data locality constraints [15].

In all aforementioned examples and related works, we note that the state statistics are typically assumed to be given and fixed (may be unknown), e.g., [3] and [8]. In practice, however, they are often time-varying and may even be non-stationary. For instance, in cellular networks, a user's location may change due to mobility, which affects his channel statistics. In crowdsourcing, popularity levels of certain types of jobs affect the satisfaction statistics of task completion. Yet this time-varying statistics aspect has been largely ignored, resulting in a *mismatch* between many control algorithms developed in the literature and control schemes in practical system, which often constantly monitor the system dynamics and adapt their actions when a change is detected.

3 Notations

$\mathbb{R}^n$ denotes the n-dimensional Euclidean space. $\mathbb{R}^n_+$ and $\mathbb{R}^n_-$ denote the non-negative and non-positive orthant. Bold symbols $\boldsymbol{x} = (x_1, ..., x_n)$ denote vectors in $\mathbb{R}^n$. The notion $w.p.1$ denotes "with probability 1." $\|\cdot\|$ denotes the Euclidean norm. For a sequence of variables $\{y(t)\}_{t=0}^{\infty}$, we also use $\overline{y} = \lim_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{y(\tau)\}$ to denote its average (when exists). $\boldsymbol{x} \succeq \boldsymbol{y}$ means that $x_j \geq y_j$ for all j.

4 System Model and Problem Formulation

In this section, we specify the general network model. We consider a network controller that operates a network with the goal of minimizing the time average cost, subject to the queue stability constraint. The network is assumed to operate in slotted time, i.e., $t \in \{0, 1, 2, ...\}$. We assume there are $r \geq 1$ queues in the network (e.g., the amount of data to be transmitted in cellular networks or the amount of flexible load to be scheduled in a smart grid).

4.1 Network state

In every slot t, we use $S(t)$ to denote the current network state, which indicates the current network parameters, such as a vector of conditions for each network link, or a collection of other relevant information about the current network channels and arrivals. We assume that $S(t)$ is i.i.d. over time given the state distribution and takes M different random network states denoted as $\mathcal{S} = \{s_1, s_2, \ldots, s_M\}$.[1] We denote $\pi_i(t) = \Pr\{S(t) = s_i\}$ the probability of being in state s_i at time t and denote $\boldsymbol{\pi}(t) = (\pi_1(t), ..., \pi_M(t))$ the state distribution at time t. We assume that the network controller can observe $S(t)$ at the beginning of every slot t, but the $\pi_i(t)$ probabilities are unknown.

4.2 The cost, traffic, and service

At each time t, after observing $S(t) = s_i$, the controller chooses an action $x(t)$ from a set $\mathcal{X}^{(s_i)}$, i.e., $x(t) = x^{(s_i)}$ for some $x^{(s_i)} \in \mathcal{X}^{(s_i)}$. The set $\mathcal{X}^{(s_i)}$ is called the feasible action set for network state s_i and is assumed to be time-invariant and compact for all $s_i \in \mathcal{S}$. The cost, traffic, and service generated by the chosen action $x(t) = x^{(s_i)}$ are as follows:

(a) The chosen action has an associated cost given by the cost function $f(t) = f(s_i, x^{(s_i)}) : \mathcal{X}^{(s_i)} \mapsto \mathbb{R}_+$ (or $\mathcal{X}^{(s_i)} \mapsto \mathbb{R}_-$ in reward maximization problems);

(b) The amount of traffic generated by the action to queue j is determined by the traffic function $A_j(t) = A_j(s_i, x^{(s_i)}) : \mathcal{X}^{(s_i)} \mapsto \mathbb{R}_+$, in units of packets;

(c) The amount of service allocated to queue j is given by the rate function $\mu_j(t) = \mu_j(s_i, x^{(s_i)}) : \mathcal{X}^{(s_i)} \mapsto \mathbb{R}_+$, in units of packets.

[1] The results in this paper can likely be generalized to systems where $S(t)$ evolves according to general time inhomogeneous Markovian dynamics.

Note that $A_j(t)$ includes both the exogenous arrivals from outside the network to queue j, and the endogenous arrivals from other queues, i.e., the transmitted packets from other queues, to queue j. We assume the functions $-f(s_i,\cdot)$, $\mu_j(s_i,\cdot)$ and $A_j(s_i,\cdot)$ are time-invariant, their magnitudes are uniformly upper bounded by some constant $\delta_{\max} \in (0,\infty)$ for all s_i, j, and they are known to the network operator.

4.3 Problem formulation

Let $\boldsymbol{q}(t) = (q_1(t),...,q_r(t))^T \in \mathbb{R}_+^r$, $t = 0,1,2,...$ be the queue backlog vector process of the network, in units of packets. We assume the following queueing dynamics:

$$q_j(t+1) = \max\big[q_j(t) - \mu_j(t) + A_j(t), 0\big], \quad \forall j, \tag{1}$$

and $\boldsymbol{q}(0) = \mathbf{0}$. By using (1), we assume that when a queue does not have enough packets to send, null packets are transmitted, so that the number of packets entering $q_j(t)$ is equal to $A_j(t)$. In this paper, we adopt the following notion of queue stability [16]:

$$\overline{q}_{\text{av}} \triangleq \limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{j=1}^{r}\mathbb{E}\big\{q_j(\tau)\big\} < \infty. \tag{2}$$

We use Π to denote an action-choosing policy, and use f_{av}^{Π} to denote its time average cost, i.e.,

$$f_{\text{av}}^{\Pi} \triangleq \limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\big\{f^{\Pi}(\tau)\big\}, \tag{3}$$

where $f^{\Pi}(\tau)$ is the cost incurred at time τ under policy Π. We call an action-choosing policy *feasible* if at every time slot t it only chooses actions from the feasible action set $\mathcal{X}^{(S(t))}$. We then call a feasible action-choosing policy under which (2) holds a *stable* policy.

In every slot, the network controller observes the current network state and chooses a control action, with the goal of minimizing the time average cost subject to network stability. This goal can be mathematically stated as:[2]

$$\textbf{(P1)}\ \ \textbf{min :}\ \boldsymbol{f_{\text{av}}^{\Pi}},\ \text{s.t.}\ \textbf{(2)}.$$

In the following, we call **(P1)** *the stochastic problem*. It can be seen that the examples in Section 2 can all be modeled by the stochastic problem framework, which is the problem formulation we focus on in this paper.

4.4 Discussion of the model

The key difference between our model and those in previous works is that $\boldsymbol{\pi}(t)$ itself can be time-varying. This is an important extension, as practical systems often have time-varying distributions for system dynamics.

[2]When $\boldsymbol{\pi}(t)$ is time-varying, the optimal system utility needs to be defined carefully in general. We will specify it when discussing corresponding results.

Thus, it is important to develop efficient techniques to handle network control in this case. Moreover, adopting a time-varying distribution model allows us to explicitly investigate the convergence property and robustness of the resulting algorithms, an aspect often missing in prior works, where mostly stochastic models with fixed parameters are considered. Focusing on this model also motivates us to explicitly consider the roles of information and learning in control.

5 Deterministic Problem and Backpressure

In this section, we review some known results in the literature that will be useful for our algorithm presentation and analysis later. We first define the *deterministic problem* and its dual problem. Then, we review the Backpressure algorithm (BP) developed in [16] for solving the stochastic problem **(P1)** with fixed $\boldsymbol{\pi}(t) = \boldsymbol{\pi}$.

5.1 The deterministic problem

The *deterministic problem* is defined as follows [17]:[3]

$$\begin{aligned} \min: F(\boldsymbol{x}, \boldsymbol{\pi}) &\triangleq V \sum_{s_i} \pi_i f(s_i, x^{(s_i)}) \\ \text{s.t. } H_j(\boldsymbol{x}, \boldsymbol{\pi}) &\triangleq \sum_{s_i} \pi_i [A_j(s_i, x^{(s_i)}) - \mu_j(s_i, x^{(s_i)})] \leq 0, \ \forall j, \\ x^{(s_i)} &\in \mathcal{X}^{(s_i)}, \quad \forall i = 1, 2, ..., M. \end{aligned} \tag{4}$$

The minimization in (4) is taken over $\boldsymbol{x} \in \prod_i \mathcal{X}^{(s_i)}$, where $\boldsymbol{x} = (x^{(s_1)}, ..., x^{(s_M)})^T$, and $V \geq 1$ is a positive constant introduced for algorithm design and analysis later. The dual problem of (4) can be obtained as follows:

$$\max: \ g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}), \quad \text{s.t. } \boldsymbol{\gamma} \succeq \mathbf{0}, \tag{5}$$

where $g_{\boldsymbol{\pi}}(\boldsymbol{\gamma})$ is the dual function and is defined as:

$$g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}) = \inf_{x^{(s_i)} \in \mathcal{X}^{(s_i)}} \sum_{s_i} \pi_i \Big\{ V f(s_i, x^{(s_i)}) + \sum_j \gamma_j \big[A_j(s_i, x^{(s_i)}) - \mu_j(s_i, x^{(s_i)}) \big] \Big\}. \tag{6}$$

Here $\boldsymbol{\gamma} = (\gamma_1, ..., \gamma_r)^T$ is the *Lagrange multiplier* of (4) and the subscript $\boldsymbol{\pi}$ is for specifying the distribution under which the dual function is defined. It is well known that $g_{\boldsymbol{\pi}}(\boldsymbol{\gamma})$ in (6) is concave in the vector $\boldsymbol{\gamma}$ for all $\boldsymbol{\gamma} \in \mathbb{R}^r$ [19]. Below, we use $\boldsymbol{\gamma}^*_{\boldsymbol{\pi}} = (\gamma^*_{\boldsymbol{\pi},1}, \gamma^*_{\boldsymbol{\pi},2}, ..., \gamma^*_{\boldsymbol{\pi},r})^T$ to denote an optimal solution of the problem (5) for a given distribution $\boldsymbol{\pi}$. For our later analysis, we also define:

$$g_i(\boldsymbol{\gamma}) = \inf_{x^{(s_i)} \in \mathcal{X}^{(s_i)}} \Big\{ V f(s_i, x^{(s_i)}) + \sum_j \gamma_j \big[A_j(s_i, x^{(s_i)}) - \mu_j(s_i, x^{(s_i)}) \big] \Big\}, \tag{7}$$

[3] In stationary systems, it can be shown that if one has all the statistical information, then a "convexified" version of (4) can be solved to obtain the optimal control policy [18].

to be the dual function for state s_i, i.e., when there is only a single state s_i. It is clear from equations (6) and (7) that:

$$g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}) = \sum_{s_i} \pi_i g_i(\boldsymbol{\gamma}). \tag{8}$$

In the following, we use $f^*_{\boldsymbol{\pi}}$ and $g^*_{\boldsymbol{\pi}}$ to denote the minimum average cost of the system and the optimal dual value of (5) under distribution $\boldsymbol{\pi}$.[4] It has been shown in [18] that:

$$f^*_{\boldsymbol{\pi}} = g^*_{\boldsymbol{\pi}}. \tag{9}$$

That is, $g^*_{\boldsymbol{\pi}}$ captures the optimal time average cost of the stochastic problem.

5.2 The Backpressure algorithm

Among the many techniques developed for solving the stochastic problem, the Backpressure algorithm has received much attention because (i) it does not require any statistical information of the changing network conditions, (ii) it has low implementation complexity, and (iii) it has strong provable performance guarantees. The Backpressure algorithm works as follows [16].[5]

Backpressure (BP): At every time slot t, observe the current network state $S(t)$ and the backlog $\boldsymbol{q}(t)$. If $S(t) = s_i$, choose $x^{(s_i)} \in \mathcal{X}^{(s_i)}$ that solves the following:

$$\begin{aligned} \max: \quad & -Vf(s_i, x) + \sum_{j=1}^{r} q_j(t)\big[\mu_j(s_i, x) - A_j(s_i, x)\big] \\ \text{s.t.} \quad & x \in \mathcal{X}^{(s_i)}. \qquad \diamond \end{aligned} \tag{10}$$

Here V is a control parameter offered by BP for trading off system utility and delay. In many problems, (14) can usually be decomposed into separate parts that are easier to solve, e.g., [21], [6]. Also, when the network state process $S(t)$ is i.i.d., it has been shown in [16] that,

$$f^{\mathtt{BP}}_{\text{av}} = f^*_{\text{av}} + O(1/V), \quad \overline{q}^{\mathtt{BP}} = O(V), \tag{11}$$

where $f^{\mathtt{BP}}_{\text{av}}$ and $\overline{q}^{\mathtt{BP}}$ are the expected average cost and the expected average network backlog size under Backpressure, respectively. The performance results in (11) hold under BP with *any* queueing discipline for choosing which packets to serve and for any V. However, we note from (14) that BP completely discards the historic information from system observations and ignores the potential benefits of learning.

[4] If a distribution $\boldsymbol{\pi}$ is such that there is no feasible solution for (4), in which case one can show that it is impossible to ensure queue stability (2) and the primal optimal is infinity, we define $f^*_{\boldsymbol{\pi}} = g^*_{\boldsymbol{\pi}} = \infty$.

[5] A similar definition of Backpressure based on fluid model was also given in Section 4.8 of [20].

6 Receding Learning-Aided Control

In this section, we present the *receding learning-aided control* technique (RLC), which allows us to simultaneously handle time-varying distributions of system dynamics and achieve good performance.

Our algorithm works as follows. First, we choose a *sampling window* size $w \triangleq V^c$ (c to be specified) and divide time into frames of w slots each, denoted by $\{\mathcal{T}_k\}_{k=0}^{\infty}$ where $\mathcal{T}_k = [kw, ..., (k+1)w - 1]$. Then, we fix a *detection threshold* $\alpha \triangleq \frac{8\log(V)}{V^{c/2}}$ and choose a *reference starting time* t_c. We also choose a *deviation factor* $\boldsymbol{\theta}$ and a *queue reference point* $\boldsymbol{q}_{\text{ref}}$ (values to be specified). After that, we periodically compare the recent samples with the historic samples to identify changes of the distribution, and adjust the sampling window's starting time once a change is identified. Then, we carry out a step called *dual learning* [9] to efficiently incorporate the learned information about the underlying distribution into a Backpressure controller. Fig. 1 shows how RLC works. The formal algorithm is given after the demonstration.

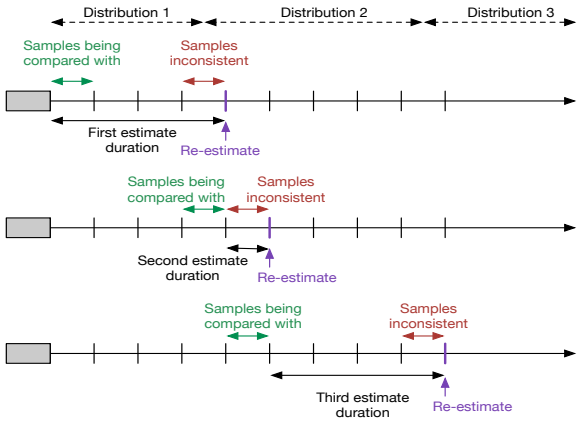

Figure 1: Demonstration of RLC. The solid rectangle represents the first frame for initialization. Once RLC detects that the sample distribution is not consistent with the reference distribution, it moves the reference point forward and re-estimates the Lagrange multiplier that will be used in the controller.

Receding Learning-aided Control (RLC): Initialize $t_c = 0$ and $\boldsymbol{\gamma}^*[0] = \boldsymbol{\theta}$. Implement:

- **Receding Dual learning** (performed every frame): At the k-th frame with $k \geq 1$, let $N_{ci}[k]$ be the number of slots in $[t_c, t_c + w - 1]$ during which $S(t) = s_i$,[6] and let $N_{si}[k]$ be the number of slots in $[(k-1)w, kw - 1]$ during which $S(t) = s_i$, respectively. Form the *reference* distribution $\hat{\boldsymbol{\pi}}_c[k]$ and the *sampling* distribution $\hat{\boldsymbol{\pi}}_s[k]$ for frame k by having $\hat{\pi}_{ci}[k] = N_{ci}[k]/w$ and $\hat{\pi}_{si}[k] = N_{si}[k]/w$. Then, perform *distribution change detection* by checking if the following condition holds:[7]

$$\max_i |\hat{\pi}_{ci}[k] - \hat{\pi}_{si}[k]| \leq \alpha/4. \tag{12}$$

(i) If yes, complete the learning step and set $\boldsymbol{\gamma}^*[k] = \boldsymbol{\gamma}^*[k-1]$.

[6] In actually implementation, one can use all samples in $[0, t_c + w]$ for estimating the empirical distribution if it can be assured that the distribution has not yet changed.

[7] Note that $\hat{\boldsymbol{\pi}}_c[k]$ stays the same if t_c remains unchanged. Note here that other methods for detecting distribution changes and estimating underlying distributions can also be applied.

(ii) If not, set $t_c = (k-1)w$ and let $\hat{\boldsymbol{\pi}}_c[k] = \hat{\boldsymbol{\pi}}_s[k]$. Then, carry out the *dual learning* step by solving the following optimization problem and obtain the optimal solution $\boldsymbol{\gamma}^*[k]$:

$$\max: \ g_{\hat{\boldsymbol{\pi}}_c[k]}(\boldsymbol{\gamma}) \triangleq \sum_{s_i} \hat{\pi}_{ci}[k] g_i(\boldsymbol{\gamma}), \quad \text{s.t. } \boldsymbol{\gamma} \succeq \mathbf{0}. \tag{13}$$

If the resulting optimal $\boldsymbol{\gamma}^*[k]$ is infinite, set $\boldsymbol{\gamma}^*[k] = V\log(V)\cdot\mathbf{1}$. Moreover, set $\boldsymbol{q}(kw) = \boldsymbol{q}_{\text{ref}}$.

- **Online control**: At every time $t \in [kw, (k+1)w-1]$, observe the current network state $S(t)$ and the backlog $\boldsymbol{q}(t)$. If $S(t) = s_i$, choose $x^{(s_i)} \in \mathcal{X}^{(s_i)}$ that solves the following:

$$\begin{aligned} \max: \ & -Vf(s_i,x) + \sum_{j=1}^{r} Q_j(t)\big[\mu_j(s_i,x) - A_j(s_i,x)\big] \\ \text{s.t.} \quad & x \in \mathcal{X}^{(s_i)}. \end{aligned} \tag{14}$$

  Here $Q_j(t) \triangleq q_j(t) + \beta_j(t)$ with $\beta_j(t) \triangleq \gamma_j^*[k] - \theta_j$ is the *effective* size for queue j at time t.
- **Queueing update**: Update the queues according to (1). Use Last-In-First-Out (LIFO) for packet scheduling. $\diamond$

Remarks: (i) `RLC` does not require any statistical information of the system and retains the low-complexity feature of `BP`. By setting $w = \infty$, `RLC` reduces to the online learning-aided control 2 (`OLAC2`) technique developed in [9] for systems with fixed distributions. (ii) The dual learning step (13) only has to be solved roughly once per distribution change. This significantly saves computational resources in practice compared to continuous computing, and can avoid reacting too quickly to temporal random changes. (iii) `RLC` tries to only utilize the recent historic information by adjusting t_c when it believes there is a change in the distribution. This feature is important and allows `RLC` to quickly adapt to distribution changes of the system dynamics. (iv) `RLC` sets the queue vector to the queue reference value $\boldsymbol{q}_{\text{ref}}$. This step readjusts the backlog starting point to further improve convergence time of the algorithm. It requires adding dummy packets when the original queue size is smaller and dropping redundant packets when there is more. We will see later that dropping rarely happens and delay is almost not affected. (v) The reason for using (12) is for screening out estimations $\hat{\boldsymbol{\pi}}_c[k]$ that are not accurate enough. This is an important step, as the quality of $\hat{\boldsymbol{\pi}}_c[k]$ affects the estimation accuracy of $\boldsymbol{\gamma}^*[k]$, which in turn has a direct impact on the algorithm performance.

7 Performance Analysis

In this section, we carry out the analysis for the `RLC` algorithm. For ease of presentation, we introduce the *distribution switching time*. Specifically, we use $\{t_d, d = 0, 1, ...\}$ to denote the starting point of the d-th distribution, i.e., $\boldsymbol{\pi}(t) = \boldsymbol{\pi}_d$ for all $t \in \mathcal{D}_d \triangleq \{t_d, t_{d+1} - 1\}$, where $\mathcal{D}_d$ is called the d-th interval. We also use $D_d \triangleq t_{d+1} - t_d$ to denote the length of $\mathcal{D}_d$.

We now present the assumptions made throughout our analysis. These assumptions are not restrictive and can typically be satisfied in network optimization problems.[8]

Assumption 1. *For every system distribution $\boldsymbol{\pi}_d$, there exists a constant $\epsilon_d = \Theta(1) > 0$ such that for any valid state distribution $\boldsymbol{\pi}' = (\pi'_1, ..., \pi'_M)$ with $\|\boldsymbol{\pi}' - \boldsymbol{\pi}_d\| \leq \epsilon_d$, there exist a set of actions $\{x_z^{(s_i)}\}_{i=1,...,M}^{z=1,2,...,\infty}$ with $x_z^{(s_i)} \in \mathcal{X}^{(s_i)}$ and some variables $\vartheta_z^{(s_i)} \geq 0$ for all s_i and z with $\sum_z \vartheta_z^{(s_i)} = 1$ for all s_i (possibly depending on $\boldsymbol{\pi}'$), such that:*

$$\sum_{s_i} \pi_{di} \Big\{ \sum_z \vartheta_z^{(s_i)} [A_j(s_i, x_z^{(s_i)}) - \mu_j(s_i, x_z^{(s_i)})] \Big\} \leq -\eta, \ \forall\, j, \tag{15}$$

where $\eta = \Theta(1) > 0$ is independent of $\boldsymbol{\pi}'$. $\diamond$

Assumption 2. *For every system distribution $\boldsymbol{\pi}_d$, $g_{\boldsymbol{\pi}_d}(\boldsymbol{\gamma})$ has a unique optimal solution $\boldsymbol{\gamma}_d^* \neq \mathbf{0}$ in $\mathbb{R}^r$.* $\diamond$

In the existing literature, Assumption 1 is mostly assumed with $\epsilon_d = 0$, e.g., [22], [23], and is known as the "slack" condition that is necessary for queue stability. Under this assumption, one can show that there exists a stationary randomized policy that stabilizes all the queues in the network (where $\vartheta_z^{(s_i)}$ represents the probability of choosing action $x_z^{(s_i)}$ when $S(t) = s_i$) [16]. Here with $\epsilon_d > 0$, we assume that when two systems are relatively "similar," they can both be stabilized by some randomized control policy (the policies may be different) that results in the same slack. Assumption 2 holds for many network utility optimization problems, e.g., [3], [17].

Below, we first have two preliminary lemmas. To present the lemmas, we define a Lyapunov function $L(t) \triangleq \frac{1}{2}\sum_{j=1}^r q_j(t)^2$ and the one-slot instant Lyapunov drift $\Delta(t) \triangleq \mathbb{E}_{\boldsymbol{\pi}(t)}\{L(t+1) - L(t) \mid \boldsymbol{q}(t)\}$, where the expectation is taken over the distribution $\boldsymbol{\pi}(t)$. Then, we define:

$$\Delta_V(t) \triangleq \Delta(t) + V\mathbb{E}_{\boldsymbol{\pi}(t)}\{f(t) \mid \boldsymbol{q}(t)\}. \tag{16}$$

We then have the following lemma.

Lemma 1. *At every time t, we have:*

$$\Delta_V(t) - \Delta_A(t) \leq B + V\mathbb{E}_{\boldsymbol{\pi}(t)}\{f(t) \mid \boldsymbol{q}(t)\} - \sum_{j=1}^r Q_j(t)\mathbb{E}_{\boldsymbol{\pi}(t)}\{\mu_j(t) - A_j(t) \mid \boldsymbol{q}(t)\}. \tag{17}$$

Here $\Delta_A(t) \triangleq \sum_j \mathbb{E}_{\boldsymbol{\pi}(t)}\{\beta_j(t)[\mu_j(t) - A_j(t)] \mid \boldsymbol{q}(t)\}$ is the drift-augmentation term, $B \triangleq 2r\delta_{\max}^2$ is independent of V, and the expectation is taken over the distribution $\boldsymbol{\pi}(t)$ of the states at time t and the possible randomness in the control policy. $\diamond$

Proof. See Appendix A. $\square$

[8] Our results can likely be extended to also handle the case where the assumptions do not hold under some distributions that only last for some finite time (Indeed, if this is also violated, no algorithm can possibly stabilize the system).

Notice that Lemma 1 indeed holds under any feasible control policies. Also note in (17) that the learned information about the underlying distribution is incorporated into system control via the term $\boldsymbol{\beta}(t)$ (recall that $\boldsymbol{Q}(t) = \boldsymbol{\beta}(t) + \boldsymbol{q}(t) - \boldsymbol{\theta}$). Based on this lemma, we obtain the following result regarding the utility performance of RLC.

Lemma 2. *For every $\mathcal{D}_d$, we have:*

$$\frac{1}{D_d}\sum_{t=t_d}^{t_{d+1}-1}\mathbb{E}_{\boldsymbol{\pi}_d}\{f(t)\} \leq f^*_{\boldsymbol{\pi}_d} + \frac{B}{V} + \frac{\Delta_A^{\mathcal{D}_d}}{VD_d} + \frac{\sum_j \mathbb{E}_{\boldsymbol{\pi}_d}\{q_j(t_d)^2\}}{2VD_d}. \tag{18}$$

Here $\Delta_A^{\mathcal{D}_d} \triangleq \sum_{t=t_d}^{t_{d+1}-1}\mathbb{E}_{\boldsymbol{\pi}_d}\{\Delta_A(t)\}$ and $f^*_{\boldsymbol{\pi}_d}$ denotes the optimal system utility when $\boldsymbol{\pi}(t) = \boldsymbol{\pi}_d$ for all t. $\diamond$

Proof. See Appendix B. □

From Lemma 2, we see that the key in proving the performance of RLC lies in bounding the term $\Delta_A^{\mathcal{D}_d}/D_d$, which can be viewed as the temporal price to pay for the inherent inaccuracy in learning due to the finite sample window size and receding sampling. This is a very challenging task. The main challenges come from the interdependency between control and learning, i.e., $\beta_j(t)$ and $\mu_j(t) - A_j(t)$, the inaccuracy in learning and estimation, and the fact that $\boldsymbol{\beta}(t)$ changes from time to time, which requires a non-asymptotic analysis.

In the following, we carry out our analysis for two system structures that are common in practice. These two structures were first introduced in [17].

7.1 The polyhedral case

We first consider the case when the system in consideration satisfies the following polyhedral condition:

Definition 1. *A system is polyhedral with parameter $\rho > 0$ under distribution $\boldsymbol{\pi}$ if the dual function $g_{\boldsymbol{\pi}}(\boldsymbol{\gamma})$ satisfies:*

$$g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}^*) \geq g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}) + \rho\|\boldsymbol{\gamma}^*_{\boldsymbol{\pi}} - \boldsymbol{\gamma}\|.\ \diamond \tag{19}$$

Condition (19) typically holds for systems where control actions are discrete (see [17] for more discussions). In this case, we first consider the performance of RLC when the distribution is time-invariant. Note that being able to perform well in stationary systems is an important requirement for any efficient adaptive algorithm. The following theorem shows that RLC achieves almost the best performance among existing algorithms designed for systems with fixed distributions.

Theorem 1. *(Polyhedral Stationary) Suppose (i) $\boldsymbol{\pi}(t) = \boldsymbol{\pi}$ for all t and (ii) $g_{\boldsymbol{\pi}}(\boldsymbol{\gamma})$ is polyhedral with $\rho = \Theta(1) > 0$. Then, under RLC with $w = V^c$, $\theta_j = q_{ref,j} = 2V^{1-c/2}\log(V)^2$, $\boldsymbol{q}(0) = \mathbf{0}$, $c \in [0,1]$, and a sufficiently large V, we have w.p.1 that:*

- *(Utility) The average cost satisfies:*

$$f_{av}^{\texttt{RLC}} \leq f_{\boldsymbol{\pi}}^* + \frac{B+O(1)}{V}. \tag{20}$$

- *(Delay) For each queue j with an average arrival rate $\lambda_j > 0$, there exist a set of packets with rate $\tilde{\lambda}_j \geq (\lambda_j - O(\log(V)/V^2))^+$ that experience only $O(\log(V)^2)$ delay.*
- *(Packet dropping) The average rate of the dropped packets during queue adjustment is $O(1/V^4)$.* $\diamond$

Proof. See Appendix C. □

Theorem 1 shows that RLC achieves the near-optimal $[O(1/V),\ O(\log(V)^2)]$ utility-delay tradeoff (only a log-factor from the optimal), which is the same compared to the previous algorithms, e.g., OLAC [9] and LIFO-BP [24]. We emphasize that the analysis of RLC is very different from previous algorithms. The main challenge lies in the fact that the $\boldsymbol{\beta}(t)$ value is obtained from fixed size samples. Hence, it changes during algorithm implementation, making it difficult to analyze the $\Delta_A^{\mathcal{D}_d}$ term.

Next, we consider the case when $\boldsymbol{\pi}(t)$ is time-varying. The following figure shows how RLC detects distribution changes. The formal statement is given in the following lemma.

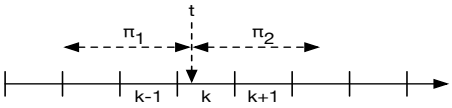

Figure 2: RLC detects distribution changes in at most $2w$ slots.

Lemma 3. *(Efficient Detection) Suppose for some time t, $\boldsymbol{\pi}(\tau) = \boldsymbol{\pi}_1$ for $\tau \in [t-2w, t-1]$ and $\boldsymbol{\pi}(\tau) = \boldsymbol{\pi}_2$ for $\tau \in [t, t+2w]$, where $\boldsymbol{\pi}_1 \neq \boldsymbol{\pi}_2$.*[9] *Then, under* RLC *with a sufficiently large V, this distribution change will be detected by time $t+2w$ with probability at least $1 - O(\frac{2M}{V^{\log(V)}})$.* $\diamond$

Proof. See Appendix D. □

Lemma 3 shows that RLC guarantees the detection of distribution changes within $2w$ timeslots with very high probability (In many cases, it will be detected within w time), which contributes to guaranteeing a fast convergence speed compared to existing algorithms (see Theorem 5 and discussions below). This property is particularly useful for non-stationary systems, as shown in the following theorem.

Theorem 2. *(Polyhedral Non-Stationary) Suppose condition (ii) in Theorem 1 holds. Then, under* RLC *with $w = V^c$, $\theta_j = q_{ref,j} = 2V^{1-c/2}\log(V)^2$, $\boldsymbol{q}(0) = \mathbf{0}$, and a sufficiently large V, for each interval $\mathcal{D}_d$ with $D_d = \Theta(V^{2+\epsilon-c/2})$ for $\epsilon > 0$ and $c \leq \frac{2+2\epsilon}{3}$, we have with probability $1 - O(\frac{M}{V^{\log(V)/2}})$ that:*

- *(Utility)* RLC *achieves:*

$$\frac{1}{D_d}\sum_{t=t_d}^{t_{d+1}-1}\mathbb{E}\{f(t)\} \leq f_{\boldsymbol{\pi}_d}^* + \frac{B+O(1)}{V}. \tag{21}$$

[9] It should be pointed out that the distribution detection component of RLC really only requires the "frame distributions" to be different, i.e., $\boldsymbol{\pi}[k] \triangleq (\boldsymbol{\pi}(kw), ..., \boldsymbol{\pi}((k+1)w-1))$ is different from $\boldsymbol{\pi}[k+1]$, which is weaker than the condition used in the lemma. The condition in the lemma is for convenience in presentation.

- *(Queueing)* $\overline{q}_{av} = O(V^{1-c/2}\log(V)^2)$. $\diamond$

Proof. See Appendix E. $\square$

Theorem 2 provides a *non-asymptotic* result for RLC's utility and delay performance. It is important to note that (21) only requires $D_d = \Theta(V^{2+\epsilon-c/2})$ for some small $\epsilon > 0$. Guaranteeing a similar performance result under BP will need an $\Theta(V^2)$ time. Also, the $O(V^{1-c/2}\log(V)^2)$ average queue size applies to general D_d sizes. For D_d values larger than $\Theta(V^{2+\epsilon-c/2})$, most packets will experience only $O(\log(V)^2)$ delay as in Theorem 1. To further appreciate the result, note that for smaller D_d values, it is much harder to guarantee results similar to (21), as in this case algorithms will mostly be running at their transient stages. On the other hand, if $\boldsymbol{\pi}(t)$ varies in a way that the aggregate frame distribution exhibits stationarity, i.e., $\boldsymbol{\pi}[k] \triangleq (\boldsymbol{\pi}(kw), ..., \boldsymbol{\pi}((k+1)w-1))$ is statistically the same as $\boldsymbol{\pi}[k+1]$ for all k, then it can be shown that RLC achieves an $O(1/V)$ close-to-optimal utility of the system under the frame-based distribution.

7.2 The locally-smooth case

We now consider the case when the system has a *locally-smooth* structure defined as follows [17].

Definition 2. *A system is locally-smooth with parameters ρ and δ if there exist $\rho > 0$ and $\delta > 0$, such that when $V = 1$, for all $\boldsymbol{\gamma}$ with $\|\boldsymbol{\gamma} - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| \leq \delta$, the dual function $g_{\boldsymbol{\pi}}(\boldsymbol{\gamma})$ satisfies:*

$$g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}^*_{\boldsymbol{\pi}}) \geq g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}) + \rho\|\boldsymbol{\gamma}^*_{\boldsymbol{\pi}} - \boldsymbol{\gamma}\|^2. \ \diamond \tag{22}$$

This structural property is different from the locally polyhedral case and we only require that all the vectors $\boldsymbol{\gamma}$ with $\|\boldsymbol{\gamma} - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| \leq \delta$ satisfy condition (22). This is due to the fact that $f(s_i, \cdot)$, $\mu_j(s_i, \cdot)$ and $A_j(s_i, \cdot)$ are all upper bounded, which eventually leads to:

$$g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}_1) - g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}_2) \leq \sqrt{B}\|\boldsymbol{\gamma}_1 - \boldsymbol{\gamma}_2\|. \tag{23}$$

That is, this condition indeed only holds *locally*. In contrast to the polyhedral condition in (19), (22) is typically satisfied when the action set is continuous, in which case the dual function is mostly continuously differentiable [17]. Similar to the polyhedral case, we first consider the performance of RLC in locally-smooth systems with fixed distributions.

Theorem 3. *(Locally-smooth Stationary) Suppose (i) $\boldsymbol{\pi}(t) = \boldsymbol{\pi}$ for all t and (ii) $g_{\boldsymbol{\pi}}(\boldsymbol{\gamma})$ is locally-smooth with $\delta, \rho = \Theta(1) > 0$. Then, under* RLC *with $w = V^c$, $\theta_j = q_{ref,j} = 2V^{3/2-c/2}\log(V)^2$, $\boldsymbol{q}(0) = \mathbf{0}$, $c \in [0, 2]$, and a sufficiently large V, we have w.p.1 that:*

- *(Utility)* RLC *achieves $f^{\texttt{RLC}}_{av} \leq f^*_{\boldsymbol{\pi}} + O(\frac{1}{V})$.*

- *(Delay) For each queue j with an average arrival rate λ_j, there exists a set of packets with rate $\tilde{\lambda}_j \geq (\lambda_j - O(1/V))^+$ that experience $O(\sqrt{V}\log(V)^2)$ delay.*
- *(Packet dropping) The average rate of packets that can potentially be dropped is $O(1/V^4)$.*

Proof. See Appendix F. □

Compared to Theorem 1, we notice that the values of $\boldsymbol{\theta}$, $\boldsymbol{q}_{\text{ref}}$, and packet delay are all different. This is because under the smooth structure, drift-based algorithms have loose control when $\boldsymbol{Q}(t)$ gets close to $\boldsymbol{\gamma}^*_{\boldsymbol{\pi}}$, resulting in a larger queue deviation and delay. The following theorem considers the non-stationary case for smooth systems and is similar to Theorem 2.

Theorem 4. *(Locally-smooth Non-Stationary) Suppose condition (ii) in Theorem 3 holds. Then, under* `RLC` *with $w = V^c$, $\theta_j = q_{ref,j} = 2V^{3/2-c/2}\log(V)^2$, $\boldsymbol{q}(0) = \mathbf{0}$, and a sufficiently large V, for each $\mathcal{D}_d$ with $D_d = \Theta(V^{5/2+\epsilon-c/2})$ for $\epsilon > 0$ and $c \leq 1 + 2\epsilon/3$, we have with probability $1 - O(\frac{M}{V^{\log(V)/2}})$ that:*

- *(Utility)* `RLC` *achieves:*

$$\frac{1}{D_d}\sum_{t=t_d}^{t_{d+1}-1}\mathbb{E}\{f(t)\} \leq f^*_{\boldsymbol{\pi}_d} + \frac{B+O(1)}{V}. \tag{24}$$

- *(Queueing) $\overline{q}_{av} = O(V^{3/2-c/2}\log(V)^2)$.* ◇

Proof. It can be proven almost identically as Theorem 2. Omitted for brevity. □

Similar to the polyhedral case, here `BP` needs an $\Theta(V^2)$ time for achieving the same performance. Hence, by choosing $c > 1$, `RLC` guarantees performance for intervals of similar sizes with much better queue size guarantee (`BP` needs $\Theta(V)$). Also, for D_d values that are larger, most packets will experience only $O(\sqrt{V}\log(V))$ delay as in Theorem 3.

8 Convergence Time Analysis

In this section, we look at the convergence time of our algorithms. Convergence time measures how fast an algorithm reaches its steady-state. Hence, it is an important indicator of the robustness and efficiency of the technique. To formally state our results, we adopt the following definition of convergence time from [9].

Definition 3. *Let $\zeta > 0$ be a given constant and let $\boldsymbol{\pi}$ be a system distribution. The ζ-convergence time of a control algorithm, denoted by T_ζ, is the time it takes for the effective queue vector $\boldsymbol{Q}(t)$ to get to within ζ distance of $\boldsymbol{\gamma}^*_{\boldsymbol{\pi}}$, i.e.,*

$$T_\zeta \triangleq \inf\{t \,|\, ||\boldsymbol{Q}(t) - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}|| \leq \zeta\}. \ \diamond \tag{25}$$

Our definition of convergence time is different from the that in [25] and [26], where convergence time relates to how fast the time-average rates converge to the optimal values. Our convergence time definition (25) is motivated by the fact that both BP and RLC (and many other drift-based algorithms) use the effective queue vector to track $\boldsymbol{\gamma}^*_{\boldsymbol{\pi}}$, which is the key for determining the optimal control actions. Hence, the faster the algorithm learns $\boldsymbol{\gamma}^*_{\boldsymbol{\pi}}$, the faster the system enters the optimal operating zone.

In the following, we present the convergence results of RLC.

Theorem 5. *(Polyhedral Convergence) Suppose condition (ii) in Theorem 1 holds. Then, under* RLC *with* $w = V^c$, $\theta_j = q_{ref,j} = 2V^{1-c/2}\log(V)^2$, *and a sufficiently large* V, *for each* $\mathcal{D}_d$ *with* $D_d = \Omega(V^{1-c/2}\log(V)^2 + V^c)$ *and* $D_{d-1} \geq 2V^c$, *we have with probability* $1 - O(\frac{M}{V^{\log(V)}})$ *that:*

$$\mathbb{E}\{T_{G_p}\} = O(V^{1-c/2}\log(V)^2 + V^c). \tag{26}$$

Here $G_p = \Theta(1)$ *is a system-dependent constant.* $\diamond$

Proof. See Appendix G. □

Choosing $c = \frac{2}{3}$, we see that $\mathbb{E}\{T_{G_p}\} = \Theta(V^{2/3}\log(V)^2)$. This result is of the same order as the OLAC algorithm's convergence time, with the key difference that OLAC does not apply to non-stationary systems. This convergence time is much faster compared to the $\Theta(V)$ time of BP (also see simulation).

We then also have the following convergence time result for the locally-smooth case.

Theorem 6. *(Locally-smooth Convergence) Suppose condition (ii) in Theorem 3 holds. Then, under* RLC *with* $w = V^c$, $\theta_j = q_{ref,j} = 2V^{3/2-c/2}\log(V)^2$, $\boldsymbol{q}(0) = \mathbf{0}$, *and a sufficiently large* V, *for each* $\mathcal{D}_d$ *with* $D_d = \Omega(V^{2-c/2}\log(V)^2 + V^c)$ *and* $D_{d-1} \geq 2V^c$, *we have with probability at least* $1 - O(\frac{M}{V^{\log(V)}})$ *that:*

$$\mathbb{E}\{T^{\texttt{RLC}}_{G_s}\} = O(V^{2-c/2}\log(V)^2 + V^c), \tag{27}$$

where $G_s = \Theta(\sqrt{V})$. *Also, in this case,*

$$\mathbb{E}\{T^{\texttt{BP}}_{G_s}\} = O(V^{3/2}), \tag{28}$$

where $T^{\texttt{BP}}_{G_s}$ *denotes the convergence time of* BP. $\diamond$

Proof. See Appendix H. □

Optimizing the c value for the locally-smooth case, we see that choosing $c = 4/3$ leads to $T^{\texttt{RLC}}_{G_s} = \Theta(V^{4/3})$, which is strictly better compared to the $T^{\texttt{BP}}_{G_s} = \Theta(V^{3/2})$ convergence time of BP. Here, it is also important to notice the different convergence times and proximities in the polyhedral case and the locally-smooth case, i.e.,

$G_p = \Theta(1)$ while $G_s = \Theta(\sqrt{V})$. This difference is due to the structural properties. In the locally-smooth case, the drift towards $\boldsymbol{\gamma}^*_{\boldsymbol{\pi}}$ decreases as the distance $\|\boldsymbol{Q}(t) - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\|$ decreases, and G_s is just enough to guarantee an $O(1/\sqrt{V})$ drift. In the polyhedral case, the drift remains constant as long as $\boldsymbol{Q}(t) \neq \boldsymbol{\gamma}^*$. Hence, a deviation of G_p is enough for guaranteeing a good concentration result.

9 Simulation

We provide simulation results for `RLC` to demonstrate both the utility-delay performance and the detection and convergence behavior. We consider a two-queue system depicted in Fig. 3.

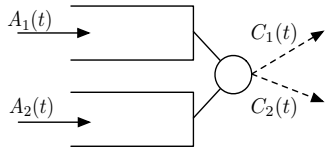

Figure 3: A two-queue system. In this system, the queues receive exogenous arrivals and the server allocates power for packet transmission over the time-varying channels.

We use $A_i(t)$ to denote the number of arriving packets to $q_i(t)$ at time t. We assume that $A_i(t)$ is i.i.d. and takes value either 2 or 0. We use $p_i = \Pr\{A_i(t) = 2\}$ and set $p_1 = 0.15$ and $p_2 = 0.3$. We assume that the channel is time-varying and denote its state at time t by $C_i(t)$, which takes values in $\mathcal{C}_1 = \{0, 1\}$ and $\mathcal{C}_2 = \{1, 2\}$. Each channel condition is equally likely for both channels. At each time t, the queue operator decides how much power to allocate for transmission. We denote $P_i(t)$ the power allocated at time t. Then, the instantaneous service rate is given by:

$$\mu_i(t) = \log(1 + C_i(t)P_i(t)). \tag{29}$$

The feasible power allocation set $\mathcal{P} = \{0, 1, 2\}$ for the discrete case and $\mathcal{P} = [0, 2]$ for the continuous case. The operator's objective is to stabilize the queues with minimum average power. We note that though the setting considered here is simple, it can indeed be used to model many problems in various contexts, e.g., CPU scheduling or mobile user transmission. Also, it can be verified that Assumptions 1 and 2 both hold for this example.

For comparison, we also simulate the `BP` algorithm. We choose $V = \{10, 30, 50, 100, 150\}$ and run each simulation instance for $T = 5 \times 10^5$ slots. The left two plots in Fig. 4 present the power and delay performance of `RLC` compared to the `BP` algorithm for the polyhedral case. It can be seen that `RLC` achieves a much better delay performance compared to `BP`, as shown in Theorem 1. The right two plots present the performance of `RLC` in the locally-smooth case. Similar to the polyhedral case, `RLC` achieves a better power-delay tradeoff. Here the empirical delay is much better than $O(\sqrt{V}\log(V)^2)$. This can be due to the structure of the particular setting. With general settings, the $O(\sqrt{V}\log(V)^2)$ will likely be observed.

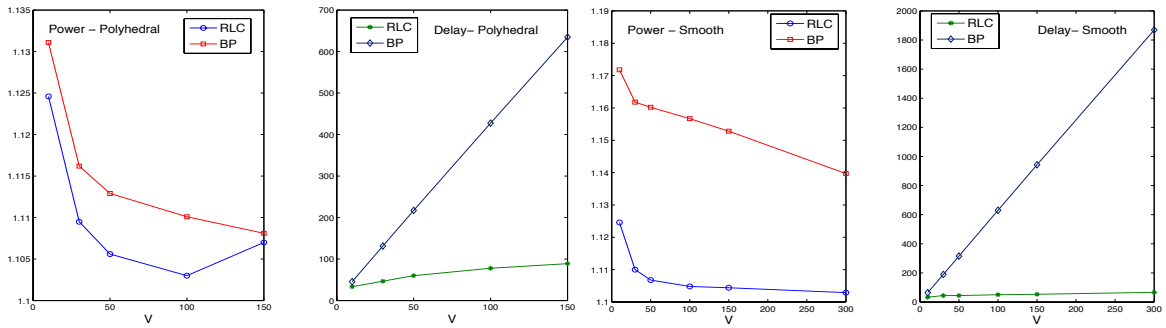

Figure 4: (Left-two) Power-delay performance of RLC and BP for the polyhedral case. We see that RLC achieves a similar power performance, while guaranteeing a much better delay. (Right-two) Performance of RLC in the locally-smooth case (we also simulate $V = 300$ since BP's power performance does not seem to converge at $V = 150$). The results are similar to those in the polyhedral case.

Fig. 5 then also shows a convergence example of RLC in the polyhedral case with $V = 500$. In this case, the packet arrival probabilities change to $p_1 = 0.1$ and $p_2 = 0.1$ at 1/3 of the simulation time. Throughout the simulation, RLC changes t_c three times, two before the rate change and one after. This shows that RLC is efficient in detecting distribution changes. It can be seen from the plots that RLC adapts much faster compared to BP. Indeed, the first convergence of RLC happens at time 1300 while BP converges at time 6000 (4500+ slots faster). The second convergence of RLC is around time 34000 while BP converges at 36000 (2000+ slots faster, change happens at 33333). We also observe that the actual queue size under RLC, i.e., $q_1(t)$ and $q_2(t)$, remain stable during the simulation, with only small fluctuations when RLC adjusts t_c and $\boldsymbol{\gamma}^*[k]$.

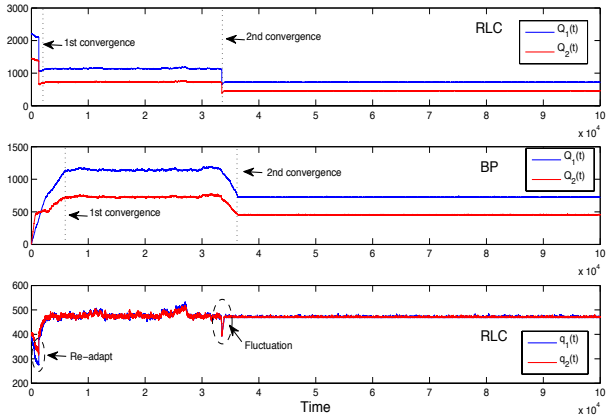

Figure 5: (Top) Convergence of RLC. (Middle) Convergence of BP. (Bottom) The actual queue sizes under RLC. We see that RLC converges much faster and ensures better smoothness of the actual queue sizes.

10 Conclusion

In this paper, we develop the *Receding Learning-aided Control* algorithm (RLC). RLC is a low-complexity online algorithm that requires *zero* a-priori statistical knowledge. It efficiently detects distribution changes via receding sampling and incorporates learned information into system controller via dual learning and drift-augmentation. We show that RLC achieves near-optimal utility-delay tradeoffs for stationary systems,

while ensuring efficient distribution change detection and fast convergence when applied to non-stationary networks. The results in this paper provide a general framework for designing joint detection-learning-control algorithms and provide new understanding about the role-of-information and the power-of-online-learning in network control.

References

[1] M. Gatzianas, L. Georgiadis, and L. Tassiulas. Control of wireless networks with rechargeable batteries. *IEEE Trans. on Wireless Communications, Vol. 9, No. 2*, Feb. 2010.

[2] D. I. Shuman and M. Liu. Energy-efficient transmission scheduling for wireless media streaming with strict underflow constraints. *WiOpt*, 2008.

[3] A. Eryilmaz and R. Srikant. Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control. *IEEE/ACM Trans. Netw.*, 15(6):1333–1344, 2007.

[4] H. Zhao, C. H. Xia, Z. Liu, and D. Towsley. A unified modeling framework for distributed resource allocation of general fork and join processing networks. *Proc. of ACM Sigmetrics*, 2010.

[5] L. Jiang and J. Walrand. Stable and utility-maximizing scheduling for stochastic processing networks. *Allerton Conference on Communication, Control, and Computing*, 2009.

[6] R. Urgaonkar and M. J. Neely. Opportunistic scheduling with reliability guarantees in cognitive radio networks. *IEEE Transactions on Mobile Computing*, 8(6):766–777, June 2009.

[7] H. Su and A. El Gamal. Modeling and analysis of the role of fast-response energy storage in the smart grid. *Proc. of Allerton*, 2011.

[8] M. J. Neely R. Urgaonkar, B. Urgaonkar and A. Sivasubramaniam. Optimal power cost management using stored energy in data centers. *Proceedings of ACM Sigmetrics*, June 2011.

[9] L. Huang, X. Liu, and X. Hao. The power of online learning in stochastic network optimization. *Proceedings of ACM Sigmetrics*, 2014.

[10] N. Chen, A. Agarwal, A. Wierman, S. Barman, and L. L. H. Andrew. Online convex optimization using predictions. *Proceedings of ACM Sigmetrics*, 2015.

[11] Waze. https://www.waze.com/.

[12] L. Huang, J. Walrand, and K. Ramchandran. Optimal smart grid tariff. *Information Theory and Applications Workshop (ITA) (Invited), San Diego*, Feb 2012.

[13] M. J. Neely, A. S. Tehrani, and A. G. Dimakis. Effficient algorithms for renewable energy allocation to delay tolerant consumers. *Proceedings of IEEE SmartGridComm*, Oct 2010.

[14] Y. Yao, L. Huang, A. Sharma, L. Golubchik, and M. J. Neely. Data centers power reduction: A two time scale approach for delay tolerant workloads. *IEEE Transactions on Parallel and Distributed Systems (TPDS), vol. 25, no. 1, pp. 200-211*, Jan 2014.

[15] W. Wang, K. Zhu, Lei Ying, J. Tan, and L. Zhang. Map task scheduling in mapreduce with data locality: Throughput and heavy-traffic optimality. *IEEE/ACM Transactions on Networking*, to appear.

[16] L. Georgiadis, M. J. Neely, and L. Tassiulas. *Resource Allocation and Cross-Layer Control in Wireless Networks*. Foundations and Trends in Networking Vol. 1, no. 1, pp. 1-144, 2006.

[17] L. Huang and M. J. Neely. Delay reduction via Lagrange multipliers in stochastic network optimization. *IEEE Trans. on Automatic Control*, 56(4):842–857, April 2011.

[18] L. Huang and M. J. Neely. Max-weight achieves the exact $[O(1/V), O(V)]$ utility-delay tradeoff under Markov dynamics. *arXiv:1008.0200v1*, 2010.

[19] D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar. *Convex Analysis and Optimization*. Boston: Athena Scientific, 2003.

[20] Sean Meyn. *Control Techniques for Complex Networks*. Cambridge University Press, 2007.

[21] L. Huang and M. J. Neely. The optimality of two prices: Maximizing revenue in a stochastic network. *IEEE/ACM Transactions on Networking*, 18(2):406–419, April 2010.

[22] L. Ying, S. Shakkottai, and A. Reddy. On combining shortest-path and back-pressure routing over multihop wireless networks. *Proceedings of IEEE INFOCOM*, April 2009.

[23] L. Bui, R. Srikant, and A. Stolyar. Novel architectures and algorithms for delay reduction in back-pressure scheduling and routing. *Proceedings of IEEE INFOCOM Mini-Conference*, April 2009.

[24] L. Huang, S. Moeller, M. J. Neely, and B. Krishnamachari. LIFO-backpressure achieves near optimal utility-delay tradeoff. *IEEE/ACM Transactions on Networking*, 21(3):831–844, June 2013.

[25] B. Li, A. Eryilmaz, and R. Li. Wireless scheduling for utility maximization with optimal convergence speed. *Proceedings of IEEE INFOCOM, Turin, Italy*, April 2013.

[26] M. Neely. Energy-aware wireless scheduling with near optimal backlog and convergence time tradeoffs. *Proceedings of IEEE INFOCOM*, 2015.

Appendix A – Proof of Lemma 1

Proof. (Lemma 1) Squaring both sides of the queueing dynamics (1), we obtain:

$$q_j(t+1)^2 \leq q_j(t)^2 + A_j(t)^2 + \mu_j(t)^2 - 2q_j(t)[\mu_j(t) - A_j(t)].$$

Summing it over $j = 1, ..., r$, defining $B \triangleq 2r\delta_{\max}^2$, and taking an expectation over $S(t)$ with distribution $\boldsymbol{\pi}(t)$ conditioning on $\boldsymbol{q}(t)$, we have:

$$\Delta(t) \leq B - \sum_j q_j(t)\mathbb{E}_{\boldsymbol{\pi}(t)}\{[\mu_j(t) - A_j(t)] \mid \boldsymbol{q}(t)\}. \tag{30}$$

Then, by adding to both sides the term $V\mathbb{E}_{\boldsymbol{\pi}(t)}\{f(t) \mid \boldsymbol{q}(t)\} - \Delta_A(t)$, and using the definition of $\Delta_A(t)$, we see that the lemma follows. □

Appendix B – Proof of Lemma 2

Proof. (Lemma 2) First, by comparing (14) and (7), one sees that (17) can indeed be rewritten as:

$$\Delta_V(t) - \Delta_A(t) \leq B + Vg_{\boldsymbol{\pi}(t)}(\boldsymbol{Q}(t)). \tag{31}$$

Using (9) and the fact that $g_{\boldsymbol{\pi}(t)}(\boldsymbol{Q}(t)) \leq g^*_{\boldsymbol{\pi}(t)}$, we have:

$$\Delta_V(t) - \Delta_A(t) \leq B + Vf^*_{\boldsymbol{\pi}(t)}. \tag{32}$$

Thus, by taking an expectation over $\boldsymbol{q}(t)$ and carrying out a telescoping sum over $t \in \mathcal{D}_d$, and by dividing both sides by VD_d, we obtain:

$$\frac{1}{D_d}\sum_{t=t_d}^{t_{d+1}-1}\mathbb{E}_{\boldsymbol{\pi}_d}\{f(t)\} \leq f^*_{\boldsymbol{\pi}_d} + \frac{B}{V} + \frac{1}{VD_d}\sum_{t=t_d}^{t_{d+1}-1}\mathbb{E}_{\boldsymbol{\pi}_d}\{\Delta_A(t)\} + \frac{\sum_j \mathbb{E}_{\boldsymbol{\pi}_d}\{q_j(t_d)^2\}}{2VD_d}.$$

Using the definition of $\Delta_A^{\mathcal{D}_d}$ proves the lemma. □

Appendix C – Proof of Theorem 1

We first have the following lemmas, which will be used in proving the theorem.

Lemma 4. *Suppose $\boldsymbol{\pi}(t) = \boldsymbol{\pi}$. Then, for a large V, at every frame k, we have with probability at least $p_s \triangleq 1 - \frac{M}{V^{\log(V)}}$ that:*

$$\max_i |\hat{\pi}_{ci}[k] - \pi_i| \leq \alpha/2, \tag{33}$$

where $\alpha \triangleq \frac{8\log(V)}{V^{c/2}}$ is the detection threshold. ◇

Proof. (Lemma 4) Using the proof of Theorem 5 in [9], we see that with probability at least $p_s = 1 - \frac{M}{V^{\log(V)}}$, $\max_i |\hat{\pi}_{si}[k] - \pi_i| \leq \alpha/4$. According to the detection rules of RLC, this ensures (33), i.e., either $\max_i |\hat{\pi}_{ci}[k] - \hat{\pi}_{si}[k]| \leq \alpha/4$, which guarantees $\max_i |\hat{\pi}_{ci}[k] - \pi_i| \leq \alpha/2$, or $\hat{\boldsymbol{\pi}}_c[k]$ will be replaced by $\hat{\boldsymbol{\pi}}_s[k]$. □

Lemma 5. *When the system is polyhedral, for a large V, at every frame k, if $\max_i |\hat{\pi}_{ci}[k] - \pi_i| \leq \alpha/2$, then:*

$$\|\boldsymbol{\gamma}^*[k] - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| \leq e^p_{\max} \triangleq b_0 V^{1-c/2} \log(V), \tag{34}$$

where $b_0 = \Theta(1)$ is a system dependent parameter. ◇

Proof. (Lemma 5) It follows directly from the proof of Theorem 5 in [9]. □

Lemma 6. *[17] Suppose the conditions in Theorem 1 hold. Then, under* BP*, there exist constants $G_p, \epsilon = \Theta(1)$, i.e., both independent of V, such that whenever $\|\boldsymbol{q}(t) - \boldsymbol{\gamma}^*\| > G_p$,*

$$\mathbb{E}_{\boldsymbol{\pi}}\{\|\boldsymbol{q}(t+1) - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| \mid \boldsymbol{q}(t)\} \leq \|\boldsymbol{q}(t) - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| - \epsilon. \quad \diamond \tag{35}$$

We also need the following technical lemmas, whose proofs are given in Appendix I.

Lemma 7. *Suppose $\boldsymbol{\pi}(t) = \boldsymbol{\pi}$. Then, for a large V, (i) if $\max_i |\hat{\pi}_{ci}[k] - \pi_i| \leq \alpha/2$,* RLC *declares distribution change with probability $O(\frac{M}{V^{\log(V)/4}})$, and (ii) if $\max_i |\hat{\pi}_{ci}[k] - \pi_i| > \alpha/2$,* RLC *declares distribution change with probability at least $1 - \frac{M}{V^{\log(V)}}$.* ◇

Lemma 8. *For any $t_1 < t_2$, we have for each queue $q_j(t)$ that:*

$$\sum_{t=t_1}^{t_2-1} [\mu_j(t) - A_j(t)] \leq q_j(t_1) - q_j(t_2) + \delta_{\max}(1 + \sum_{t=t_1}^{t_2-1} I_{[q_j(t)\leq 0]}). \quad \diamond \tag{36}$$

We now begin the proof for Theorem 1.

Proof. (Theorem 1) (**Utility**) We first prove the utility performance. By taking a limit as $D_d \to \infty$ in (18) and using $\boldsymbol{\pi}(t) = \boldsymbol{\pi}$ and $\boldsymbol{q}(0) < \infty$, we get:

$$f^{\texttt{RLC}}_{\text{av}} \leq f^*_{\boldsymbol{\pi}} + \frac{B + \overline{\Delta}^{\infty}_A}{V}. \tag{37}$$

Here $\overline{\Delta}^{\infty}_A = \lim_{D_d\to\infty} \Delta^{\mathcal{D}_d}_A / D_d$. It remains to show that $\overline{\Delta}^{\infty}_A = O(1)$, i.e.,

$$\lim_{D_0\to\infty} \frac{1}{D_0} \sum_{t=0}^{D_0-1} \sum_j \mathbb{E}_{\boldsymbol{\pi}}\{\beta_j(t)[\mu_j(t) - A_j(t)]\} = O(1), \tag{38}$$

where we use D_0 as we only have one distribution throughout.

Under Assumption 1, we see that for a large enough V, with probability 1, we have $\boldsymbol{\gamma}^*[k] = \Theta(V)$ for each k (Lemma 1 in [17]). Now let us divide the frames into disjoint intervals such that during each interval

$\hat{\boldsymbol{\pi}}_c$ remains unchanged (hence $\boldsymbol{\beta}(t)$ stays constant). Then, we say that (i) the reference distribution $\hat{\boldsymbol{\pi}}_c[k]$ is *correct* if $\max_i |\hat{\pi}_{ci}[k] - \pi_i| \leq \alpha/2$ (event denoted by $\mathcal{E}[k]$), or (ii) the reference estimation $\hat{\boldsymbol{\pi}}_c[k_1]$ is *incorrect* if $\max_i |\hat{\pi}_{ci}[k_1] - \pi_i| > \alpha/2$ (event denoted by $\mathcal{E}[k]^c$). This is shown in Fig. 6. Note that in both cases, $\boldsymbol{\beta}(t)$ remains constant throughout the interval.

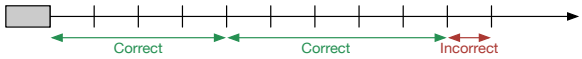

Figure 6: Under RLC, the timeline consists of intervals that possess correct ($\max_i |\hat{\pi}_{ci}[k] - \pi_i| \leq \alpha/2$) and incorrect ($\max_i |\hat{\pi}_{ci}[k] - \pi_i| > \alpha/2$) estimates of the distribution.

Denote $\mathcal{I}_l \triangleq [k_l w, k_{l+1} w - 1]$ the l-th interval during which $\boldsymbol{\beta}(t)$ stays constant. We rewrite Δ_A^∞ as:

$$\Delta_A^\infty \triangleq \sum_l \sum_{t \in \mathcal{I}_l} \sum_j \mathbb{E}_{\boldsymbol{\pi}}\{\beta_j(t)[\mu_j(t) - A_j(t)]\} = \sum_{l=0}^{\infty} \sum_{t=k_l w}^{k_{l+1} w - 1} \sum_j \mathbb{E}_{\boldsymbol{\pi}}\{\beta_j(t)[\mu_j(t) - A_j(t)]\}.$$

Note that by the rules of RLC, we always have $|\beta_j(t)| \leq b_1 V \log(V)$ for some constant $b_1 = \Theta(1)$.

Consider one interval $[k_1 w, k_2 w - 1]$ with $k_2 > k_1$ and look at $\sum_{t=k_1 w}^{k_2 w - 1} \mathbb{E}_{\boldsymbol{\pi}}\{\beta_j(t)[\mu_j(t) - A_j(t)]\}$. We start with the first case where the reference is incorrect, i.e., $\mathcal{E}[k_1]^c$ happens. In this case, using Lemma 7, we see that at most with probability $\frac{M}{V^{\log(V)}}$ a change will not be declared. Therefore,

$$\Pr\{k_2 - k_1 \geq l\} \leq (\frac{2M}{V^{\log(V)}})^{l-1}, \forall\, l \geq 1. \tag{39}$$

Hence, conditioning on $\mathcal{E}[k_1]^c$ and using the fact that $|\beta_j(t)| \leq b_1 V \log(V)$, we have:

$$\mathbb{E}\{\sum_{t=k_1 w}^{k_2 w - 1} \mathbb{E}_{\boldsymbol{\pi}}\{\beta_j(t)[\mu_j(t) - A_j(t)]\} \mid \mathcal{E}[k_1]^c\} \leq w \frac{b_1 V \log(V) \delta_{\max}}{(1 - \frac{2M}{V^{\log(V)}})^2} \leq 2 b_1 V^{1+c} \log(V) \delta_{\max}. \tag{40}$$

We also note that the probability for $\mathcal{E}[k_1]^c$ to happen is $O(\frac{M}{V^{\log(V)}})$.

Now consider the second case when $\mathcal{E}[k_1]$ takes place, i.e., the reference is correct. Using Lemma 7 again, we have that:

$$\Pr\{k_2 - k_1 > V^2\} \geq 1 - \frac{2MV^2}{V^{\log(V)/4}} \geq 1 - \frac{2M}{V^4}. \tag{41}$$

Conditioning on $\mathcal{E}_1 = \{k_2 - k_1 \leq V^2\}$, which happens with probability at most $\frac{2M}{V^4}$, one has:

$$\mathbb{E}\{\sum_{t=k_1 w}^{k_2 w - 1} \mathbb{E}_{\boldsymbol{\pi}}\{\beta_j(t)[\mu_j(t) - A_j(t)]\} \mid \mathcal{E}_1, \mathcal{E}[k_1]\} \leq b_1 V^{3+c} \log(V) \delta_{\max}. \tag{42}$$

On the other hand, when $k_2 - k_1 > V^2$, which happens with probability at least $1 - \frac{2M}{V^4}$, $\beta_j(t)$ remains constant for at least V^{2+c} slots. Lemma 9 at the end of this appendix shows that, during this period,

$$\sum_{t=k_1 w}^{k_2 w - 1} \Pr\{q_j(t) \leq \delta_{\max}\} \leq b_2 [k_2 w - k_1 w + 1] / V^{\log V}. \tag{43}$$

Here $b_2 = \Theta(1)$ is some system-dependent parameter. Thus, using Lemma 8 and (43), we get:

$$\mathbb{E}\Big\{\sum_{t=k_1w}^{k_2w-1}\mathbb{E}_{\boldsymbol{\pi}}\{[\mu_j(t)-A_j(t)]\}\mid\mathcal{E}_1^c,\mathcal{E}[k_1]\Big\} \tag{44}$$
$$\leq\mathbb{E}\big\{q_j(k_1w)-q_j(k_2w-1)\mid\mathcal{E}_1^c,\mathcal{E}[k_1]\big\}+\delta_{\max}+b_2\delta_{\max}\mathbb{E}\big\{k_2w-k_1w+1\mid\mathcal{E}_1^c,\mathcal{E}[k_1]\big\}/V^{\log V}.$$

Lastly, note that we set $\boldsymbol{q}(k_1w)=\boldsymbol{q}_{\text{ref}}=2V^{1-c/2}\log(V)^2$. Thus, combining (40), (42) and (44), we have:

$$\sum_{t=k_1w}^{k_2w-1}\sum_j\mathbb{E}_{\boldsymbol{\pi}}\{\beta_j(t)[\mu_j(t)-A_j(t)]\} \tag{45}$$
$$\leq r\Big[2b_1V^{1+c}\log(V)\delta_{\max}\cdot\frac{M}{V^{\log(V)}}+\frac{2M}{V^4}\cdot b_1V^{3+c}\log(V)\delta_{\max}+p_sb_1V\log(V)2V^{1-c/2}\log(V)^2$$
$$+p_sb_1V\log(V)\cdot b_2\delta_{\max}\mathbb{E}\big\{|\mathcal{I}|\mid\mathcal{E}_1^c,\mathcal{E}[k]\big\}/V^{\log V}+p_s\delta_{\max}V\log(V)\Big]$$
$$=O(V^{2-c/2}\log(V)^3).$$

Here $|\mathcal{I}|$ denotes the length of the interval.

Finally, to complete the analysis, recall that under RLC, the timeline is divided into intervals shown in Fig. 6. We define $\mathcal{I}_l$ as the l-th such interval and rewrite $\overline{\Delta}_A^\infty$ as follows:

$$\overline{\Delta}_A^\infty=\lim_{l\to\infty}\frac{\sum_l\sum_{t\in\mathcal{I}_l}\sum_j\mathbb{E}_{\boldsymbol{\pi}}\{\beta_j(t)[\mu_j(t)-A_j(t)]\}}{\sum_l\mathbb{E}\big\{|\mathcal{I}_l|\big\}}. \tag{46}$$

From (41), we get that for each $\mathcal{I}_l$,

$$\mathbb{E}\big\{|\mathcal{I}_l|\big\}\geq V^2/2. \tag{47}$$

Also, using the fact that $\Pr\big\{\mathcal{E}_1\big\}\leq\frac{2M}{V^4}$, it can be shown that $\mathbb{E}\big\{|\mathcal{I}|\mid\mathcal{E}_1^c,\mathcal{E}[k]\big\}\leq 2\mathbb{E}\big\{|\mathcal{I}|\big\}$. Combing this with (45), (46) and (47), we conclude that:

$$\overline{\Delta}_A^\infty=O(1).$$

Plugging this into (37), we see that the utility result follows.

(**Delay**) Now we look at the delay performance. From the argument above, we see that the frames with correct reference distribution dominate the intervals. Hence, we focus on showing that most packets experience very small delay during these intervals.

Consider one such interval $[k_1w,k_2w-1]$. Using Lemma 7 again, we can see that for a large V,

$$\Pr\big\{k_2-k_1>V^5\big\}\geq 1-\frac{2MV^5}{V^{\log(V)/4}}\geq 1-\frac{2M}{V^2}. \tag{48}$$

Indeed, using (39), we see that the expected number of packet arrivals during an interval with an incorrect reference is no more than $2V^c\delta_{\max}$, while the expected number of arrivals during a correct frame is $\Omega(\lambda_jV^5/2)$.

According to Lemma 5, Lemma 6, and (54) in the proof of Lemma 9, we see that when the distribution is correct,

$$\|\boldsymbol{\gamma}^*[k_1] - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| = O(V^{1-c/2}\log(V)). \tag{49}$$

Define $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta} + \boldsymbol{\gamma}^*_{\boldsymbol{\pi}} - \boldsymbol{\gamma}^*[k_1]$. We see from Lemma 6 that whenever $\|\boldsymbol{q}(t) - \hat{\boldsymbol{\theta}}\| > G_p$, which is equivalent to $\|\boldsymbol{Q}(t) - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| > G_p$,

$$\mathbb{E}\big\{\|\boldsymbol{q}(t+1) - \hat{\boldsymbol{\theta}}\| \mid \boldsymbol{q}(t)\big\} \leq \|\boldsymbol{q}(t) - \hat{\boldsymbol{\theta}}\| - \epsilon,$$

for the same $G_p, \epsilon = \Theta(1)$ in Lemma 6. Using (49) and $\theta = 2V^{1-c/2}\log(V)^2$, we see that $\hat{\boldsymbol{\theta}} = \Theta(V^{1-c/2}\log(V)^2)$. Therefore, by invoking Theorem 4 from [9], we have:

$$\mathbb{E}\big\{T_{G_p}(\boldsymbol{q}(t))\big\} \leq O(V^{1-c/2}\log(V)/\epsilon). \tag{50}$$

Here $T_{G_p}(\boldsymbol{q}(t)) \triangleq \inf\{t : \|\boldsymbol{q}(t) - \hat{\boldsymbol{\theta}}\| \leq G_p\}$. Thus,

$$\Pr\big\{T_{G_p}(\boldsymbol{q}(t)) > V^{3-c/2}\big\} \leq O(\log(V)/V^2). \tag{51}$$

Now focus on the event $\{T_{G_p}(\boldsymbol{q}(t)) \leq V^{3-c/2}\}$ and denote t^* the first time $Y(t) \triangleq \|\boldsymbol{q}(t) - \hat{\boldsymbol{\theta}}\| \leq G_p$. Following an argument almost identical to the proof of Theorem 1 in [17], one can show that:

$$\sum_{t=t^*}^{k_2 w - 1} \frac{\nu\epsilon}{2}\mathbb{E}\big\{e^{\nu Y(t)}\big\} \leq (k_2 w - 1 - t^*)e^{2\nu\sqrt{r}\delta_{\max}} + e^{\nu Y(t^*)}. \tag{52}$$

Here $\nu \triangleq \frac{\epsilon}{\delta^2_{\max} + \delta_{\max}\epsilon/3} = \Theta(1)$. Hence, by denoting $b_3 = 2e^{2\nu\sqrt{r}\delta_{\max}}/\nu\epsilon = \Theta(1)$ and $b_4 = e^{\nu Y(t^*)} \leq e^{\nu G_p} = \Theta(1)$ and by choosing $m = \log(V)^2$, we get from (52) that:

$$\frac{1}{(k_2 - k_1)w}\sum_{t=k_1 w}^{k_2 w - 1}\Pr\big\{Y(t) > G_p + m\big\} \leq b_3 e^{-\nu m} + (b_4 + b_3(t^* - k_1 w))/(k_2 - k_1)w = O(\frac{V^{3-c/2}}{V^5}) = O(1/V^2).$$

Thus, the fraction of time $\{\|\boldsymbol{q}(t) - \hat{\boldsymbol{\theta}}\| \geq G_p\}$ happens is only $O(\log(V)/V^2)$, implying that at most $O(\lambda_j \log(V)/V^2)$ amount of packet will enter and depart from $q_j(t)$ when $\|\boldsymbol{q}(t) - \hat{\boldsymbol{\theta}}\| > G_p$.

Summarizing the above, we see that all but an $O(\log(V)/V^2)$ fraction of the traffic (due to cases when $k_2 - k_1 < V^5$ and $T_{G_p}(\boldsymbol{q}(t)) > V^{3-c/2}$) enter and depart when $q_j(t) \in [\hat{\theta}_j - G_p - \log(V)^2, \hat{\theta}_j + G_p + \log(V)^2]$. This implies that their delay in the queue is $O(\log(V)^2)$.

(**Dropping**) We see from (48) that the frequency of dropping is no more than once every V^{5+c} slots with probability larger than 0.9. Each case we drop no more than $O(V)$ packets on average (Theorem 1 in [17]). Hence, the overall dropping rate is $O(1/V^4)$. □

Lemma 9. *Suppose the conditions in Theorem 1 hold. Then, under* `RLC`*, given $\mathcal{E}[k_1]$, we have for each j that:*

$$\sum_{t=k_1 w}^{k_2 w-1} Pr\{q_j(t) \leq \delta_{\max}\} \leq b_2[k_2 w - k_1 w + 1]/V^{\log(V)}. \tag{53}$$

Here $b_2 = \Theta(1)$ is a system-dependent constant. $\diamond$

Proof. (Lemma 9) First, we see that with a large V, $V^{1-c/2}\log(V)^2 \geq 2e^p_{\max} = 2b_0 V^{1-2/c}\log(V)$. Thus, using Lemma 5 and that $\theta_j = 2V^{1-c/2}\log(V)^2$, we have that given $\mathcal{E}[k_1]$, at every frame $k \in [k_1, k_2]$,

$$\beta_j(t) \in [\gamma^*_{\boldsymbol{\pi}j} - \frac{3}{4}\theta_j, \gamma^*_{\boldsymbol{\pi}j} - \frac{1}{4}\theta_j]. \tag{54}$$

It means that whenever $q_j(t) < q_p \triangleq \frac{1}{4}V^{1-c/2}\log(V)^2$, $Q_j(t) = q_j(t) + \beta_j(t) < \gamma^*_{\boldsymbol{\pi}j} - \frac{1}{4}V^{1-c/2}\log(V)^2$, which implies $\|\boldsymbol{\gamma}^*_{\boldsymbol{\pi}} - \boldsymbol{Q}(t)\| > G_p$ when V is large. Denote $\mathcal{E}_j(t)$ the event that $q_j(t) < q_p - \delta_{\max}$. Lemma 6 then shows that given $\mathcal{E}[k_1]$, we have for every $t \in [k_1 w, (k_1+1)w - 1]$ that:

$$\mathbb{E}_{\boldsymbol{\pi}}\{\|\boldsymbol{q}(t+1) - \tilde{\boldsymbol{\theta}}\| \mid \boldsymbol{q}(t), \mathcal{E}_j(t)\} \leq \|\boldsymbol{q}(t) - \tilde{\boldsymbol{\theta}}\| - \epsilon, \tag{55}$$

for some $\tilde{\boldsymbol{\theta}}$ with $\tilde{\theta}_j \geq \frac{1}{8}\theta$ for all j.

Having established (55), define $Y(t) = \|\boldsymbol{q}(t+1) - \tilde{\boldsymbol{\theta}}\| - \delta_{\max}$. We see then $q_j(t) < \tilde{\theta} - \delta_{\max}$ implies $Y(t) > 0$. From (55), we see then:

$$\mathbb{E}_{\boldsymbol{\pi}}\{Y(t+1) \mid Y(t) > 0\} \leq Y(t) - \epsilon. \tag{56}$$

Define an exponential Lyapunov function $\tilde{L}(t) \triangleq e^{\nu Y(t)}$ and $\tilde{\Delta}(t) \triangleq \mathbb{E}\big\{\tilde{L}(t+1) - \tilde{L}(t) \mid \boldsymbol{q}(t)\big\}$. It was shown in [17] that by choosing $\nu = \frac{\epsilon}{\delta^2_{\max} + \delta_{\max}\epsilon/3} = \Theta(1)$, we get:

$$\tilde{\Delta}(t) \leq e^{2\nu\delta_{\max}} - \frac{\nu\epsilon}{2}e^{\nu Y(t)}. \tag{57}$$

Let t^* be the first time after $k_1 w$ that $q_j(t) \leq \tilde{\theta} - \delta_{\max}$. We see then $0 \leq Y(t^*) \leq \delta_{\max}$. By taking expectation on both sides of (57) and carrying out a telescoping sum from $t = t^*$ to $k_2 w - 1$, we obtain:

$$\sum_{t=t^*}^{k_2 w-1} \frac{\nu\epsilon}{2}\mathbb{E}\big\{e^{\nu Y(t)}\big\} \leq [k_2 w - 1 - t^*]e^{2\nu\delta_{\max}} + \mathbb{E}\big\{e^{\nu Y(t^*)}\big\}.$$

Using $\mathbb{E}\big\{e^{\nu Y(t)}\big\} \geq e^{\nu m}\Pr\big\{Y(t) > m\big\}$ and the fact that $q_j(t) \leq \delta_{\max}$ implies $Y(t) \geq \tilde{\theta} - 2\delta_{\max}$, we have:

$$\begin{aligned}\sum_{t=t^*}^{k_2 w-1} \Pr\big\{q_j(t) \leq \delta_{\max}\big\} &\leq \sum_{t=t^*}^{k_2 w-1} \Pr\big\{Y(t) \geq \frac{1}{4}V^{1-c/2}\log(V)^2 - 2\delta_{\max}\big\} \\ &\leq b_2[k_2 w - t^*]e^{-\nu V^{1-c/2}\log(V)^2/4} + e^{\nu(\delta_{\max} - \frac{1}{4}V^{1-c/2}\log(V)^2 + 2\delta_{\max})}.\end{aligned}$$

Here $b_2 \triangleq \frac{2}{\nu\epsilon} = \Theta(1)$. Using $\Pr\big\{q_j(t) \leq \delta_{\max}\big\} = 0$ for $t \in [k_1 w, t^*]$, we see that the lemma follows. $\square$

Appendix D – Proof of Lemma 3

Proof. (Lemma 3) Since $\boldsymbol{\pi}_2$ is different from $\boldsymbol{\pi}_1$, there exists at least one coordinate j such that $|\pi_{1j} - \pi_{2j}| > \epsilon = \Theta(1)$.

Denote by k the frame t belongs to. We see then frame $k-1$ has $\boldsymbol{\pi}(t) = \boldsymbol{\pi}_1$. Hence, we have that with probability of at least $1 - \frac{M}{V^{\log(V)}}$, $\max |\hat{\pi}_{ci}[k-1] - \pi_{1i}| \leq \alpha/2$. Note that this holds regardless of $\hat{\boldsymbol{\pi}}_c[k-1]$ being updated at frame $k-1$ or not. Since $\boldsymbol{\pi}(\tau) = \boldsymbol{\pi}_2$ for $\tau \in [t, t+2w]$, we see that at least frame $k+1$ has a distribution very different from $\hat{\boldsymbol{\pi}}_c[k-1]$. Thus, a change will be declared at $(k+2)w$. This is so becasue with probability $1 - \frac{M}{V^{\log(V)}}$, $\max |\pi_{si}[k+1] - \pi_{2i}| \leq \alpha/2$. Thus, (12) will be violated at $(k+2)w$ with probability at least $1 - \frac{2M}{V^{\log(V)}}$, if it is not yet violated at time $(k+1)w$ (This is possible because $w = V^c$. If t is close to kw, then the sample distribution is not very different from $\boldsymbol{\pi}_1$). □

Appendix E – Proof of Theorem 2

Proof. (Theorem 2) (**Utility**) Using Lemma 4, we see that with probability at least $1 - \frac{M}{V^{\log(V)}}$, $\max_i |\hat{\pi}_{ci}[0] - \pi_i| \leq \alpha/2$. In this case, we see that when $D_d = \Theta(V^{2+\epsilon-c/2})$, with probability $1 - \frac{MV^3}{V^{\log(V)}} \geq 1 - \frac{M}{V^{\log(V)/2}}$, $\boldsymbol{\beta}(t)$ remains unchanged throughout the interval.

Conditioning on this event and using the same argument as in the utility proof of Theorem 1, we have $\Delta_A^{D_d}/D_d = O(1)$ (equation (45)). Moreover, Lemma 3 shows that the true distribution will be detected in $2w = 2V^c$ time with probability $1 - O(\frac{2M}{V^{\log(V)}})$. Thus, choosing c such that $2 + \epsilon - c/2 > c + 1$, i.e., $c \leq (2+2\epsilon)/3$, ensures that the detection period contributes only $O(1/V)$ of the cost. Plugging the above into (18) and using the fact that $q_{\text{ref},j} = 2V^{1-c/2}\log(V)^2$ prove (21).

(**Queueing**) To show the queue performance, we note that (54) also holds in this case. Thus, there exists $\tilde{\boldsymbol{\theta}}$ with $\tilde{\theta}_j \leq \frac{3}{4}\theta$, such that whenever $\|\boldsymbol{q}(t) - \tilde{\boldsymbol{\theta}}\| > G_p$,

$$\mathbb{E}\big\{\|\boldsymbol{q}(t+1) - \tilde{\boldsymbol{\theta}}\| | \boldsymbol{q}(t)\big\} \leq \|\boldsymbol{q}(t) - \tilde{\boldsymbol{\theta}}\| - \epsilon, \tag{58}$$

for some $\epsilon = \Theta(1)$. Using an argument similar to the proof of Theorem 1 in [17], one can show that $\overline{q}_{\text{av}} = O(\frac{3r}{4}V^{1-c/2}\log(V)^2)$. □

Appendix F – Proof of Theorem 3

We prove Theorem 3 here. First we have the following lemma, whose proof is given in Appendix J.

Lemma 10. *When the system is locally-smooth, for a large enough V, at every frame k, if* $\max_i |\hat{\pi}_{ci}[k] - \hat{\pi}_i| \leq \alpha/2$, *then:*

$$\|\boldsymbol{\gamma}^*[k] - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| \leq e^s_{\max} \triangleq b_5 V^{\frac{3-c}{2}}\log(V), \tag{59}$$

where $b_5 = \Theta(1)$ is a system-dependent constant. $\diamond$

We similarly have the following lemma regarding the drift of the queue vector towards $\boldsymbol{\gamma}^*_{\boldsymbol{\pi}}$ under BP.

Lemma 11. *[17] Suppose the conditions in Theorem 3 hold. Then, under* BP*, there exist a constant $G_s = \Theta(\sqrt{V})$, such that whenever $\|\boldsymbol{q}(t) - \boldsymbol{\gamma}^*\| > G_s$,*

$$\mathbb{E}_{\boldsymbol{\pi}}\{\|\boldsymbol{q}(t+1) - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| \mid \boldsymbol{q}(t)\} \leq \|\boldsymbol{q}(t) - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| - \frac{1}{\sqrt{V}}. \quad \diamond \tag{60}$$

Now we prove Theorem 3.

Proof. (Theorem 3) (**Utility**) First, one can check that Lemma 7 and Lemma 8 still hold in this case, since they only involve the underlying distribution and sample path queueing. Lemma 9 can also be verified to hold. In particular, (57) holds with $\epsilon = 1/\sqrt{V}$ and $\nu = \Theta(1/\sqrt{V})$. In this case, we can define $q_s \triangleq \frac{1}{4}V^{(3-c)/2}\log(V)^2$ and $Y_j(t) = \max[q_s - q_j(t) - \delta_{\max}, 0]$. Then, the proof of Lemma 9 for the locally-smooth case follows exactly as in the polyhedral case.

Therefore, one can verify that (40) to (44) still hold, while (45) becomes (recall $q_{\text{ref},j} = 2V^{\frac{3-c}{2}}\log(V)^2$):

$$\begin{aligned}
&\sum_{t\in\mathcal{I}_l}\sum_j \mathbb{E}_{\boldsymbol{\pi}}\{\beta_j(t)[\mu_j(t) - A_j(t)]\} \qquad (61)\\
&\leq r\Bigg[2b_1V^{1+c}\log(V)\delta_{\max}\cdot\frac{M}{V^{\log(V)}} + \frac{2M}{V^4}\cdot b_1V^{3+c}\log(V)\delta_{\max} + p_sb_1V\log(V)2V^{(3-c)/2}\log(V)^2\\
&\quad + p_sb_1V\log(V)\cdot b_2\delta_{\max}\mathbb{E}\big\{|\mathcal{I}| \mid \mathcal{E}_1^c, \mathcal{E}[k]\big\}/V^{\log V} + p_s\delta_{\max}V\log(V)\Bigg]\\
&= O(V^{\frac{5-c}{2}}\log(V)^2).
\end{aligned}$$

Having established (61), the rest of the proof goes exactly the same as in the proof of Theorem 1.

(**Delay**) We use a similar argument as in the polyhedral case. In particular, using Theorem 6, we have:

$$\mathbb{E}\big\{T_{G_s}(\boldsymbol{q}(t))\big\} \leq O(V^{2-c/2}\log(V)^2), \tag{62}$$

for some $G_s = \Theta(\sqrt{V})$, and that:

$$\Pr\big\{T_{G_s}(\boldsymbol{q}(t)) > V^{4-c/2}\big\} \leq O(\log(V)^2/V^2). \tag{63}$$

Here we have ignored the V^c term in Theorem 6 as there is only one distribution. Following the argument as in the proof of Theorem 1 and using the fact that $k_2 - k_1 \geq V^5$, we see that:

$$\frac{1}{(k_2-k_1)w}\sum_{t=k_1w}^{k_2w-1}\Pr\big\{Y(t) > G_s + m\big\} = O(1/V).$$

Using Theorem 3 in [17] and (63), we see that for each interval, with probability $1 - O(\frac{\log(V)}{V^2})$, most packets

will enter and leave the queue when $q_j(t) \in [G_s - \sqrt{V}\log(V)^2, G_s + \sqrt{V}\log(V)^2]$. Thus, all but an $O(1/V)$ fraction of the traffic only experience $O(\sqrt{V}\log(V)^2)$ delay.

(**Dropping**): The proof is the same as in Theorem 1. □

Appendix G – Proof of Theorem 5

Proof. (Theorem 5) First of all, we have from Lemma 3 that with probability at least $1 - O(\frac{2M}{V^{\log(V)}})$, the new distribution $\boldsymbol{\pi}_d$ will be detected after $2w = 2V^c$ time. Moreover, with probability $1 - O(\frac{M}{V^{\log(V)}})$, (54) holds, in which case we have:

$$\|\boldsymbol{Q}(t) - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| = \|\boldsymbol{\beta}[k] + \boldsymbol{q}(t) - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| = O(V^{1-c/2}\log(V)^2). \tag{64}$$

Lemma 5 in [9] then shows that the expected time for $\boldsymbol{Q}(t)$ to get to within G_p of $\boldsymbol{\gamma}^*_{\boldsymbol{\pi}}$ is $O(V^{1-c/2}\log(V)^2)$. Combining it with the time to detect the distribution change, we see that the theorem follows. □

Appendix H – Proof of Theorem 6

Proof. (Theorem 6) First, from Lemma 10, we have with probability of at least $1 - O(\frac{M}{V^{\log(V)}})$ that:

$$\|\boldsymbol{\gamma}^*[k] - \boldsymbol{\gamma}^*_{\boldsymbol{\pi}}\| \leq b_5 V^{\frac{3-c}{2}}\log(V). \tag{65}$$

Using Lemma 11, (65), and Lemma 5 in [9], we conclude that after getting the correct estimation of the underlying distribution, the expected time to get to within G_s distance is $O(V^{2-c/2}\log(V)^2)$. Combining it with Lemma 3, which states that $O(V^c)$ time is sufficient for detecting distribution change, we obtain (27).

(28) follows by noticing that after the distribution change, we have $\|\boldsymbol{q}(t) - \boldsymbol{\gamma}^*\| = \Theta(V)$. This is so because $\boldsymbol{q}(t)$ now has to move towards a different optimal multiplier, which has difference $\Theta(V)$ from the current one (Lemma 1 in [17]). Hence, repeating the above argument, we obtain (28). □

Appendix I – Proof of supporting lemmas

This appendix presents the proofs of supporting Lemmas 7, 8 and 10.

Proof. (Lemma 7) We first consider case (ii). Since $\max_i |\hat{\pi}_{ci}[k] - \pi_i| > \alpha/2$, `RLC` will declare distribution change if $\max_i |\hat{\pi}_{si}[k] - \pi_i| \leq \alpha/4$, because then $\max_i |\hat{\pi}_{si}[k] - \hat{\pi}_{ci}[k]| > \alpha/4$. Using Lemma 4, we see that this happens with probability at least $1 - \frac{M}{V^{\log(V)}}$.

Now consider case (i), i.e., $\max_i |\hat{\pi}_{ci}[k] - \pi_i| \leq \alpha/2$. In this case, `RLC` only declares distribution change when (12) is violated. We show that given $\max_i |\hat{\pi}_{ci}[k] - \pi_i| \leq \alpha/2$, this is very unlikely. To see this, denote t_c the reference time for frame k and let k' be the index of the frame t_c belongs to. Then, we have:

$$\Pr\Big\{ \max_i |\hat{\pi}_{ci}[k'] - \pi_i| \leq \alpha/8 \mid \max_i |\hat{\pi}_{ci}[k'] - \pi_i| \leq \alpha/2 \Big\}$$

$$
\begin{aligned}
&= \frac{\Pr\{\max_i |\hat{\pi}_{ci}[k'] - \pi_i| \leq \alpha/8\}}{\Pr\{\max_i |\hat{\pi}_{ci}[k'] - \pi_i| \leq \alpha/2\}} \\
&\geq \Pr\{\max_i |\hat{\pi}_{ci}[k'] - \pi_i| \leq \alpha/8\} \\
&\geq 1 - MV^{-\log(V)/4}.
\end{aligned}
$$

The last inequality holds since $\hat{\boldsymbol{\pi}}_c[k']$ is formed by the sample distribution in frame k'. Given this, we see that RLC declares distribution change only if $\max_i |\hat{\pi}_{si}[k] - \pi_i| > \alpha/8$, which happens only with probability $1 - MV^{-\log(V)/4}$. Thus, RLC does not claim distribution change with probability at least $1 - 2MV^{-\log(V)/4}$. □

Proof. (Lemma 8) To prove this result, let us look at a queue process example shown in Fig. 7. We see that during any busy interval $[t_{1i}, t_{2i}]$, i.e., $q_j(t) > 0$ for $t \in [t_{1i}, t_{2i}]$ but $q_j(t) = 0$ for $t = t_{1i} - 1$ and $t_{2i} + 1$, one must have $\sum_{t=t_{1i}-1}^{t_{2i}-1}[\mu_j(t) - A_j(t)] \leq 0$.

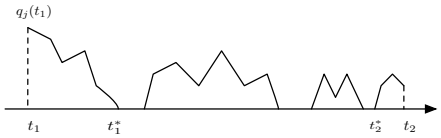

Figure 7: A queue process with busy-idle intervals.

Thus, if we start from a time t_1^* when $q_j(t_1^*) = 0$, then $\sum_{t=t_1^*}^{t_2-1}[\mu_j(t) - A_j(t)] \leq \delta_{\max} \sum_{t=t_1^*}^{t_2-1} I_{[q_j(t)\leq 0]}$. Now choose t_1^* to be the first time after t_1 such that $q_j(t) = 0$ and denote t_2^* the last time before t_2 that $q_j(t) = 0$. We see then the lemma follows as $\sum_{t=t_1}^{t_1^*-1}[\mu_j(t) - A_j(t)] \leq q_j(t_1) + \delta_{\max}$ and $\sum_{t=t_2^*}^{t_2}[\mu_j(t) - A_j(t)] \leq -q_j(t_2)$. □

Appendix J – Proof of Lemma 10

Proof. (Lemma 10) To start, we recall the following inequality from [9], which states that for all $\boldsymbol{\gamma} \neq \gamma_{\boldsymbol{\pi}}^*$,

$$
g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}_{\boldsymbol{\pi}}^*) - g_{\hat{\boldsymbol{\pi}}_c[k]}(\boldsymbol{\gamma}^*[k]) \leq 2 \max_i \delta_i[k] M(V f_{\max} + r\xi B). \tag{66}
$$

Here $\delta_i[k] \triangleq |\pi_i - \hat{\pi}_{c_i}^*[k]|$ is the distribution estimation error and $\xi = \Theta(V)$.

Given (22) and the concavity of $g_{\boldsymbol{\pi}}(\boldsymbol{\gamma})$, [17] shows that there exists $G_s = \Theta(\sqrt{V})$, such that whenever $\|\boldsymbol{\gamma}_{\boldsymbol{\pi}}^* - \boldsymbol{\gamma}\| > G_s$,

$$
g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}_{\boldsymbol{\pi}}^*) \geq g_{\boldsymbol{\pi}}(\boldsymbol{\gamma}) + \frac{1}{\sqrt{V}} \|\boldsymbol{\gamma}_{\boldsymbol{\pi}}^* - \boldsymbol{\gamma}\|. \tag{67}
$$

Combining (67) with (66), we conclude that with probability $1 - \frac{M}{V^{\log(V)}}$, $\max \delta_i[k] \leq \frac{\log(V)}{V^{c/2}}$. Hence, when $c \in [0, 2]$,

$$
\|\boldsymbol{\gamma}^*[k] - \boldsymbol{\gamma}_{\boldsymbol{\pi}}^*\| \leq b_5 V^{\frac{3-c}{2}} \log(V), \tag{68}
$$

for some constant $b_5 = \Theta(1)$. □