Learn-and-Adapt Stochastic Dual Gradients for Network Resource Allocation

Tianyi Chen, *Student Member, IEEE*, Qing Ling, *Senior Member, IEEE*, and Georgios B. Giannakis, *Fellow, IEEE*

***Abstract*—Network resource allocation shows revived popularity in the era of data deluge and information explosion. Existing stochastic optimization approaches fall short in attaining a desirable cost-delay tradeoff. Recognizing the central role of Lagrange multipliers in a network resource allocation, a novel learn-and-adapt stochastic dual gradient (LA-SDG) method is developed in this paper to learn the sample-optimal Lagrange multiplier from historical data, and accordingly adapt the upcoming resource allocation strategy. Remarkably, an LA-SDG method only requires just an extra sample (gradient) evaluation relative to the celebrated stochastic dual gradient method. LA-SDG can be interpreted as a foresighted learning scheme with *an eye on the future*, or, a modified heavy-ball iteration from an optimization viewpoint. It has been established—both theoretically and empirically—that LA-SDG markedly improves the cost-delay tradeoff over state-of-the-art allocation schemes.**

***Index Terms*—First-order method, network resource allocation, statistical learning, stochastic approximation.**

I. Introduction

IN THE era of big data analytics, cloud computing and Internet of Things, the growing demand for massive data processing challenges existing resource allocation approaches. Huge volumes of data acquired by distributed sensors in the presence of operational uncertainties caused by, for example, renewable energy, call for scalable and adaptive network control schemes. Scalability of a desired approach refers to low complexity and amenability to distributed implementation, while adaptivity implies the capability of online adjustment to dynamic environments.

Allocation of network resources can be traced back to the seminal work of [1]. Since then, popular allocation algorithms operating in the dual domain are first-order methods based on dual gradient ascent, either deterministic [2] or stochastic [3], [4]. Thanks to their simple computation and implementation, these approaches have attracted a great deal of recent interest, and have been successfully applied to cloud, transportation, and power grid networks, for example, [5]–[8]. However, their major limitation is *slow convergence*, which results in high *network delay*. Depending on the application domain, the delay can be viewed as workload queuing time in a cloud network, traffic congestion in a transportation network, or energy level of batteries in a power network. To address this delay issue, recent attempts aim at accelerating first- and second-order optimization algorithms [9]–[12]. Specifically, momentum-based accelerations over first-order methods were investigated using Nesterov [9] or heavy-ball iterations [10]. Though these approaches work well in static settings, their performance degrades with online scheduling, as evidenced by the increase in accumulated steady-state error [13]. On the other hand, second-order methods such as the decentralized quasi-Newton approach and its dynamic variant developed in [11] and [12], incur high overhead to compute and communicate the decentralized Hessian approximations.

Capturing prices of resources, Lagrange multipliers play a central role in stochastic resource allocation algorithms [14]. Given abundant historical data in an online optimization setting, a natural question arises: *Is it possible to learn the optimal prices from past data, so as to improve the performance of online resource allocation strategies?* The rationale here is that past data contain statistics of network states, and learning from them can aid coping with the stochasticity of future resource allocation. A recent work in this direction is [15], which considers resource allocation with a *finite* number of possible network states and allocation actions. The learning procedure, however, involves constructing a histogram to estimate the underlying distribution of the network states, and explicitly solves an empirical dual problem. While constructing a histogram is feasible for a probability distribution with finite support, quantization errors and prohibitively high complexity are inevitable for a continuous distribution with infinite support.

In this context, this paper aims to design a novel online resource allocation algorithm that leverages online learning from historical data for stochastic optimization of the ensuing allocation stage. The resulting approach, which we call "learn-and-adapt" stochastic dual gradient (LA-SDG) method, only doubles computational complexity of the classic stochastic dual gradient (SDG) method. With this minimal cost, LA-SDG mitigates

Manuscript received July 10, 2017; revised September 12, 2017; accepted October 31, 2017. Date of publication November 15, 2017; date of current version December 14, 2018. This work was supported in part by NSF 1509040, 1508993, 1509005; in part by NSF China 61573331; in part by NSF Anhui 1608085QF130; and in part by CAS-XDA06040602. Recommended for publication by Associate Editor Fabio Fagnani. *(Corresponding author: Georgios B. Giannakis.)*

T. Chen and G. B. Giannakis are with the Department of Electrical and Computer Engineering and the Digital Technology Center, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: chen3827@umn.edu; georgios@umn.edu).

Q. Ling is with the School of Data and Computer Science, Sun Yat-Sen University, Guangzhou 510006, China (e-mail: qingling@ieee.org).

Digital Object Identifier 10.1109/TCNS.2017.2774043

steady-state oscillation, which is common in stochastic first-order acceleration methods [10], [13], while avoiding computation of the Hessian approximations present in the second-order methods [11], [12]. Specifically, LA-SDG only requires one more past sample to compute an extra SDG, in contrast to constructing costly histograms and solving the resulting large-scale problem [15].

The main contributions of this paper are summarized as follows.

1) Targeting a low-complexity online solution, LA-SDG only takes an additional dual gradient step relative to the classic SDG iteration. This step enables adapting the resource allocation strategy through learning from historical data. Meanwhile, LA-SDG is linked with the stochastic heavy-ball method, nicely inheriting its fast convergence in the initial stage, while reducing its steady-state oscillation.
2) The novel LA-SDG approach, parameterized by a positive constant μ, provably yields an attractive cost-delay tradeoff $[\mu, \log^2(\mu)/\sqrt{\mu}]$, which improves upon the standard tradeoff $[\mu, 1/\mu]$ of the SDG method [4]. Numerical tests further corroborate the performance gain of LA-SDG over existing resource allocation schemes.

Notation: $\mathbb{E}$ denotes the expectation operator, $\mathbb{P}$ stands for probability, $(\cdot)^\top$ stands for vector and matrix transposition, and $\|\mathbf{x}\|$ denotes the ℓ_2-norm of a vector $\mathbf{x}$. Inequalities for vectors, for example, $\mathbf{x} > \mathbf{0}$, are defined entry-wise. The positive projection operator is defined as $[a]^+ := \max\{a, 0\}$, also entry-wise.

II. Network Resource Allocation

In this section, we start with a generic network model and its resource allocation task in Section II-A, and then introduce a specific example of resource allocation in cloud networks in Section II-B. The proposed approach is applicable to more general network resource allocation tasks such as geographical load balancing in cloud networks [5], traffic control in transportation networks [7], and energy management in power networks [8].

A. Unified Resource Allocation Model

Consider discrete time $t \in \mathbb{N}$, and a network represented as a directed graph $\mathcal{G} = (\mathcal{I}, \mathcal{E})$ with nodes $\mathcal{I} := \{1, \ldots, I\}$ and edges $\mathcal{E} := \{1, \ldots, E\}$. Collect the workloads across edges $e = (i, j) \in \mathcal{E}$ in a resource allocation vector $\mathbf{x}_t \in \mathbb{R}^E$. The $I \times E$ node-incidence matrix is formed with the (i, e)th entry

$$\mathbf{A}_{(i,e)} = \begin{cases} 1, & \text{if link } e \text{ enters node } i \\ -1, & \text{if link } e \text{ leaves node } i \\ 0, & \text{else.} \end{cases} \tag{1}$$

We assume that each row of $\mathbf{A}$ has at least one -1 entry, and each column of $\mathbf{A}$ has, at most, one -1 entry, meaning that each node has at least one outgoing link, and each link has, at most, one source node. With $\mathbf{c}_t \in \mathbb{R}^I_+$ collecting the randomly arriving workloads of all nodes per slot t, the aggregate (endogenous plus exogenous) workloads of all nodes are $\mathbf{A}\mathbf{x}_t + \mathbf{c}_t$. If the ith entry of $\mathbf{A}\mathbf{x}_t + \mathbf{c}_t$ is positive, there is service residual queued at node i; otherwise, node i overserves the current arrival. With a workload queue per node, the queue length vector $\mathbf{q}_t := [q_t^1, \ldots, q_t^I]^\top \in \mathbb{R}^I_+$ obeys the recursion

$$\mathbf{q}_{t+1} = [\mathbf{q}_t + \mathbf{A}\mathbf{x}_t + \mathbf{c}_t]^+ \qquad \forall t \tag{2}$$

where $\mathbf{q}_t$ can represent the amount of user requests buffered in data queues, or energy stored in batteries, and $\mathbf{c}_t$ is the corresponding exogenously arriving workloads or harvested renewable energy of all nodes per slot t. Defining $\Psi_t(\mathbf{x}_t) := \Psi(\mathbf{x}_t; \boldsymbol{\phi}_t)$ as the aggregate network cost parameterized by the random vector $\boldsymbol{\phi}_t$, the local cost per node i is $\Psi_t^i(\mathbf{x}_t) := \Psi^i(\mathbf{x}_t; \boldsymbol{\phi}_t^i)$, and $\Psi_t(\mathbf{x}_t) = \sum_{i \in \mathcal{I}} \Psi_t^i(\mathbf{x}_t)$. The model here is quite general. The duration of time slots can vary from (micro-) seconds in cloud networks, minutes in road networks, to even hours in power networks; the nodes can present the distributed front-end mapping nodes and back-end data centers in cloud networks, intersections in traffic networks, or buses and substations in power networks; the links can model wireless/wireline channels, traffic lanes, and power transmission lines, while the resource vector $\mathbf{x}_t$ can include the size of data workloads, the number of vehicles, or the amount of energy.

Concatenating the random parameters into a random state vector $\mathbf{s}_t := [\boldsymbol{\phi}_t^\top, \mathbf{c}_t^\top]^\top$, the resource allocation task is to determine the allocation $\mathbf{x}_t$ in response to the observed (realization) $\mathbf{s}_t$ "on the fly," so as to minimize the *long-term average* network cost subject to queue stability at each node, and operation feasibility at each link. Concretely, we have

$$\Psi^* := \min_{\{\mathbf{x}_t, \forall t\}} \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\Psi_t(\mathbf{x}_t)\right] \tag{3a}$$

$$\text{s.t.} \quad \mathbf{q}_{t+1} = [\mathbf{q}_t + \mathbf{A}\mathbf{x}_t + \mathbf{c}_t]^+ \qquad \forall t \tag{3b}$$

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\mathbf{q}_t\right] < \infty \tag{3c}$$

$$\mathbf{x}_t \in \mathcal{X} := \{\mathbf{x} \,|\, \mathbf{0} \le \mathbf{x} \le \bar{\mathbf{x}}\} \qquad \forall t \tag{3d}$$

where Ψ^* is the optimal objective of problem (3), which includes also future information; $\mathbb{E}$ is taken over as $\mathbf{s}_t := [\boldsymbol{\phi}_t^\top, \mathbf{c}_t^\top]^\top$ as well as possible randomness of optimization variable $\mathbf{x}_t$; constraints (3c) ensure queue stability;[1] and (3d) confines the instantaneous allocation variables to stay within a time-invariant box constraint set $\mathcal{X}$, which is specified by, for example, link capacities or server/generator capacities.

The queue dynamics in (3b) couple the optimization variables over an infinite time horizon, which implies that the decision variable at the current slot will have an effect on all future decisions. Therefore, finding an optimal solution of (3) calls for dynamic programming [16], which is known to suffer from the "curse of dimensionality" and intractability in an online setting. In Section III-A, we will circumvent this obstacle by relaxing (3b)–(3c) to limiting average constraints, and employing dual decomposition techniques.

[1] Here, we focus on the strong stability given by [4, Def. 2.7], which requires the time-average expected queue length to be finite.

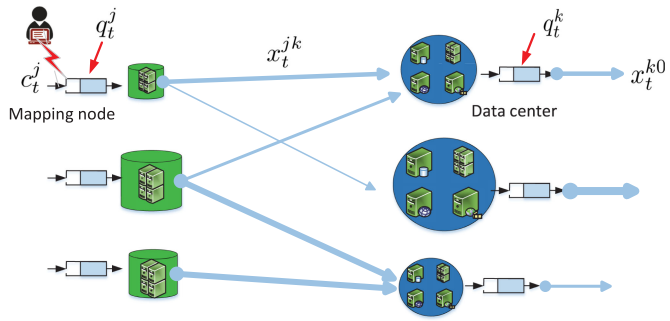


Fig. 1. Diagram of online geographical load balancing. Per time t, mapping node j has an exogenous workload c_t^j plus that stored in the queue q_t^j, and schedules workload x_t^{jk} to data center k. Data center k serves an amount of workload x_t^{k0} out of all the assigned x_t^{jk} as well as that stored in the queue q_t^k. The thickness of each edge is proportional to its capacity.

B. Motivating Setup

The geographic load balancing task in a cloud network [5], [17], [18] takes the form of (3) with J mapping nodes (e.g., DNS servers) indexed by $\mathcal{J} := \{1, \ldots, J\}$, K data centers indexed by $\mathcal{K} := \{J+1, \ldots, J+K\}$. To match the definition in Section II-A, consider a virtual outgoing node (indexed by 0) from each data center, and let $(k, 0)$ represent this outgoing link. Define further the node set $\mathcal{I} := \mathcal{J} \bigcup \mathcal{K}$ that includes all nodes except the virtual one, and the edge set $\mathcal{E} := \{(j,k), \forall j \in \mathcal{J}, k \in \mathcal{K}\} \bigcup \{(k,0), \forall k \in \mathcal{K}\}$ that contains links connecting mapping nodes with data centers, and outgoing links from data centers.

Per slot t, each mapping node j collects the amount of user data requests c_t^j, and forwards the amount x_t^{jk} on its link to data center k constrained by the bandwidth availability. Each data center k schedules workload processing x_t^{k0} according to its resource availability. The amount x_t^{k0} can be also viewed as the resource on its virtual outgoing link $(k, 0)$. The bandwidth limit of link (j, k) is $\bar{x}^{jk}$, while the resource limit of data center k (or link $(k, 0)$) is $\bar{x}_t^{k0}$. Similar to those in Section II-A, we have the optimization vector $\mathbf{x}_t := \{x_t^{ij}, \forall (i,j) \in \mathcal{E}\} \in \mathbb{R}^{|\mathcal{E}|}$, $\mathbf{c}_t := [c_t^1, \ldots, c_t^J, 0 \ldots, 0]^\top \in \mathbb{R}^{J+K}$, and $\bar{\mathbf{x}} := \{\bar{x}_t^{ij}, \forall (i,j) \in \mathcal{E}\} \in \mathbb{R}^{|\mathcal{E}|}$. With these notational conventions, we have an $|\mathcal{I}| \times |\mathcal{E}|$ node-incidence matrix $\mathbf{A}$ as in (1). At each mapping node and data center, undistributed or unprocessed workloads are buffered in queues obeying (3b) with queue length $\mathbf{q}_t \in \mathbb{R}_+^{J+K}$; see also the system diagram in Fig. 1.

Performance is characterized by the aggregate cost of power consumed at the data centers plus the bandwidth costs at the mapping nodes, namely

$$\Psi_t(\mathbf{x}_t) := \sum_{k \in \mathcal{K}} \underbrace{\Psi_t^k(x_t^{k0})}_{\text{power cost}} + \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} \underbrace{\Psi_t^{jk}(x_t^{jk})}_{\text{bandwidth cost}}. \quad (4)$$

The power cost $\Psi_t^k(x_t^{k0}) := \Psi^k(x_t^{k0}; \boldsymbol{\phi}_t^k)$, parameterized by the random vector $\boldsymbol{\phi}_t^k$, captures the local marginal price, and the renewable generation at data center k during time period t. The bandwidth cost $\Psi_t^{jk}(x_t^{jk}) := \Psi^{jk}(x_t^{jk}; \boldsymbol{\phi}_t^{jk})$, parameterized by the random vector $\boldsymbol{\phi}_t^{jk}$, characterizes the heterogeneous cost of data transmission due to spatiotemporal differences. To match the unified model in Section II-A, the local cost at data center $k \in \mathcal{K}$ is its power cost $\Psi_t^k(x_t^{k0})$, and the local cost at mapping node $j \in \mathcal{J}$ becomes $\Psi_t^j(\{x_t^{jk}\}) := \sum_{k \in \mathcal{K}} \Psi_t^{jk}(x_t^{jk})$. Hence, the cost in (4) can be also written as $\Psi_t(\mathbf{x}_t) := \sum_{i \in \mathcal{I}} \Psi_t^i(\mathbf{x}_t)$. Aiming to minimize the time-average of (4), geographical load balancing fits the formulation in (3).

III. Online Network Management via SDG

In this section, the dynamic problem (3) is reformulated to a tractable form, and the classical SDG approach is revisited, along with a brief discussion of its online performance.

A. Problem Reformulation

Recall in Section II-A that the main challenge of solving (3) resides in time-coupling constraints and unknown distribution of the underlying random processes. Regarding the first hurdle, combining (3b) with (3c), it can be shown that in the long term, workload arrival and departure rates must satisfy the following necessary condition [4, Theor. 2.8]:

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\mathbf{A}\mathbf{x}_t + \mathbf{c}_t\right] \leq \mathbf{0} \quad (5)$$

given that the initial queue length is finite, that is, $\|\mathbf{q}_1\| \leq \infty$. In other words, on average, all buffered delay-tolerant workloads should be served. Using (5), a relaxed version of (3) is

$$\tilde{\Psi}^* := \min_{\{\mathbf{x}_t, \forall t\}} \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\Psi_t(\mathbf{x}_t)\right] \quad \text{s.t. (3d) and (5)} \quad (6)$$

where $\tilde{\Psi}^*$ is the optimal objective for the relaxed problem (6).

Compared to (3), problem (6) eliminates the time coupling across variables $\{\mathbf{q}_t, \forall t\}$ by replacing (3b) and (3c) with (5). Since (6) is a relaxed version of (3) with the optimal objective $\tilde{\Psi}^* \leq \Psi^*$, if one solves (6) instead of (3), it will be prudent to derive an optimality bound on Ψ^*, provided that the sequence of solutions $\{\mathbf{x}_t, \forall t\}$ obtained by solving (6) is feasible for the relaxed constraints (3b) and (3c). Regarding the relaxed problem (6), using arguments similar to those in [4, Th. 4.5], it can be shown that if the random state $\mathbf{s}_t$ is independent and identically distributed (i.i.d.) over time t, there exists a *stationary* control policy $\boldsymbol{\chi}^*(\cdot)$, which is a pure (possibly randomized) function of the realization of random state $\mathbf{s}_t$ (or the *observed* state $\mathbf{s}_t$), that is, it satisfies (3d) and guarantees that $\mathbb{E}[\Psi_t(\boldsymbol{\chi}^*(\mathbf{s}_t))] = \tilde{\Psi}^*$ and $\mathbb{E}[\mathbf{A}\boldsymbol{\chi}^*(\mathbf{s}_t) + \mathbf{c}_t] \leq \mathbf{0}$. Since the optimal policy $\boldsymbol{\chi}^*(\cdot)$ is time invariant, it implies that the *dynamic* problem (6) is equivalent to the following time-invariant *ensemble* program:

$$\tilde{\Psi}^* := \min_{\boldsymbol{\chi}(\cdot)} \ \mathbb{E}\left[\Psi\big(\boldsymbol{\chi}(\mathbf{s}_t); \mathbf{s}_t\big)\right] \quad (7a)$$

$$\text{s.t.} \quad \mathbb{E}[\mathbf{A}\boldsymbol{\chi}(\mathbf{s}_t) + \mathbf{c}(\mathbf{s}_t)] \leq \mathbf{0} \quad (7b)$$

$$\boldsymbol{\chi}(\mathbf{s}_t) \in \mathcal{X} \quad \forall \mathbf{s}_t \in \mathcal{S} \quad (7c)$$

where $\boldsymbol{\chi}(\mathbf{s}_t) := \mathbf{x}_t$, $\mathbf{c}(\mathbf{s}_t) = \mathbf{c}_t$, and $\Psi\big(\boldsymbol{\chi}(\mathbf{s}_t); \mathbf{s}_t\big) := \Psi_t(\mathbf{x}_t)$; set $\mathcal{S}$ is the sample space of $\mathbf{s}_t$, and the constraint (7c) holds almost surely. Observe that the index t in (7) can be dropped,

since the expectation is taken over the distribution of random variable $\mathbf{s}_t$, which is time-invariant. Leveraging the equivalent form (7), the remaining task boils down to finding the optimal policy that achieves the minimal objective in (7a) and obeys the constraints (7b) and (7c).[2] Note that the optimization in (7) is with respect to a stationary policy $\boldsymbol{\chi}(\cdot)$, which is an infinite dimensional problem in the primal domain. However, there is a finite number of expected constraints [cf., (7b)]. Thus, the dual problem contains a finite number of variables, hinting at the effect that solving (7) is tractable in the dual domain [19], [20].

B. Lagrange Dual and Optimal Policy

With $\boldsymbol{\lambda} \in \mathbb{R}^I_+$ denoting the Lagrange multipliers associated with (7b), the Lagrangian of (7) is

$$\mathcal{L}(\boldsymbol{\chi}, \boldsymbol{\lambda}) := \mathbb{E}\left[\mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda})\right] \tag{8}$$

with $\boldsymbol{\lambda} \geq \mathbf{0}$, and the instantaneous Lagrangian is

$$\mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}) := \Psi_t(\mathbf{x}_t) + \boldsymbol{\lambda}^\top(\mathbf{A}\mathbf{x}_t + \mathbf{c}_t) \tag{9}$$

where constraint (7c) remains implicit. Notice that the instantaneous objective $\Psi_t(\mathbf{x}_t)$ and the instantaneous constraint $\mathbf{A}\mathbf{x}_t + \mathbf{c}_t$ are both parameterized by the observed state $\mathbf{s}_t := [\boldsymbol{\phi}_t^\top, \mathbf{c}_t^\top]^\top$ at time t, that is, $\mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}) = \mathcal{L}(\boldsymbol{\chi}(\mathbf{s}_t), \boldsymbol{\lambda}; \mathbf{s}_t)$.

Correspondingly, the Lagrange dual function is defined as the minimum of the Lagrangian over all feasible primal variables [21], given by

$$\begin{aligned}\mathcal{D}(\boldsymbol{\lambda}) &:= \min_{\{\boldsymbol{\chi}(\mathbf{s}_t)\in\mathcal{X},\quad \forall \mathbf{s}_t\in\mathcal{S}\}} \mathcal{L}(\boldsymbol{\chi}, \boldsymbol{\lambda}) \\ &= \min_{\{\boldsymbol{\chi}(\mathbf{s}_t)\in\mathcal{X},\quad \forall \mathbf{s}_t\in\mathcal{S}\}} \mathbb{E}\left[\mathcal{L}(\boldsymbol{\chi}(\mathbf{s}_t), \boldsymbol{\lambda}; \mathbf{s}_t)\right]. \end{aligned} \tag{10a}$$

Note that the optimization in (10a) is still in regards to a function. To facilitate the optimization, we rewrite (10a), relying on the so-termed *interchangeability principle* [22, Theor. 7.80].

Lemma 1: Let $\boldsymbol{\xi}$ denote a random variable on $\boldsymbol{\Xi}$, and $\mathcal{H} := \{h(\cdot) : \boldsymbol{\Xi} \to \mathbb{R}^n\}$ denote the function space of all functions on $\boldsymbol{\Xi}$. For any $\boldsymbol{\xi} \in \boldsymbol{\Xi}$, if $f(\cdot, \boldsymbol{\xi}) : \mathbb{R}^n \to \mathbb{R}$ is a proper and lower semicontinuous convex function, then it follows that:

$$\min_{h(\cdot)\in\mathcal{H}} \mathbb{E}\left[f(h(\boldsymbol{\xi}), \boldsymbol{\xi})\right] = \mathbb{E}\left[\min_{\mathbf{h}\in\mathbb{R}^n} f(\mathbf{h}, \boldsymbol{\xi})\right]. \tag{10b}$$

Lemma 1 implies that, under mild conditions, we can replace the optimization over a function space with (infinitely many) point-wise optimization problems. In the context here, we assume that $\Psi_t(\mathbf{x}_t)$ is proper, lower semicontinuous, and strongly convex (cf., Assumption 2 in Section V). Thus, for given finite $\boldsymbol{\lambda}$ and $\mathbf{s}_t$, $\mathcal{L}(\cdot, \boldsymbol{\lambda}; \mathbf{s}_t)$ is also strongly convex, proper, and lower semicontinuous. Therefore, applying Lemma 1 yields

$$\min_{\{\boldsymbol{\chi}(\cdot):\mathcal{S}\to\mathcal{X}\}} \mathbb{E}\left[\mathcal{L}(\boldsymbol{\chi}(\mathbf{s}_t), \boldsymbol{\lambda}; \mathbf{s}_t)\right] = \mathbb{E}\left[\min_{\boldsymbol{\chi}(\mathbf{s}_t)\in\mathcal{X}} \mathcal{L}(\boldsymbol{\chi}(\mathbf{s}_t), \boldsymbol{\lambda}; \mathbf{s}_t)\right] \tag{10c}$$

where the minimization and the expectation are interchanged. Accordingly, we rewrite (10a) in the following form:

$$\mathcal{D}(\boldsymbol{\lambda}) = \mathbb{E}\left[\min_{\boldsymbol{\chi}(\mathbf{s}_t)\in\mathcal{X}} \mathcal{L}(\boldsymbol{\chi}(\mathbf{s}_t), \boldsymbol{\lambda}; \mathbf{s}_t)\right] = \mathbb{E}\left[\min_{\mathbf{x}_t\in\mathcal{X}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda})\right]. \tag{10d}$$

Likewise, for the instantaneous dual function $\mathcal{D}_t(\boldsymbol{\lambda}) = \mathcal{D}(\boldsymbol{\lambda}; \mathbf{s}_t) := \min_{\mathbf{x}_t\in\mathcal{X}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda})$, the dual problem of (7) is

$$\max_{\boldsymbol{\lambda}\geq\mathbf{0}} \mathcal{D}(\boldsymbol{\lambda}) := \mathbb{E}\left[\mathcal{D}_t(\boldsymbol{\lambda})\right]. \tag{11}$$

In accordance with the ensemble primal problem (7), we will henceforth refer to (11) as the *ensemble* dual problem.

If the optimal Lagrange multiplier $\boldsymbol{\lambda}^*$ associated with (7b) was known, then optimizing (7) and consequently (6) would be equivalent to minimizing the Lagrangian $\mathcal{L}(\boldsymbol{\chi}, \boldsymbol{\lambda}^*)$ or infinitely many instantaneous $\{\mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}^*)\}$, over the set $\mathcal{X}$ [16]. We restate this assertion as follows.

Proposition 1: Consider the optimization problem in (7). Given a realization $\mathbf{s}_t$, and the optimal Lagrange multiplier $\boldsymbol{\lambda}^*$ associated with the constraints (7b), the optimal instantaneous resource allocation decision is

$$\mathbf{x}_t^* = \boldsymbol{\chi}^*(\mathbf{s}_t) \in \arg\min_{\boldsymbol{\chi}(\mathbf{s}_t)\in\mathcal{X}} \mathcal{L}(\mathbf{x}_t, \boldsymbol{\lambda}^*; \mathbf{s}_t) \tag{12}$$

where $\in$ accounts for possibly multiple minimizers of $\mathcal{L}_t$.

When the realizations $\{\mathbf{s}_t\}$ are obtained sequentially, one can generate a sequence of optimal solutions $\{\mathbf{x}_t^*\}$ correspondingly for the dynamic problem (6). To obtain the optimal allocation in (12), however, $\boldsymbol{\lambda}^*$ must be known. This fact motivates our novel LA-SDG method in Section IV. To this end, we will first outline the celebrated SDG iteration (a.k.a. Lyapunov optimization).

C. Revisiting Stochastic Dual (Sub)Gradient

To solve (11), a standard gradient iteration involves sequentially taking expectations over the distribution of $\mathbf{s}_t$ to compute the gradient. Note that when the Lagrangian minimization (cf., (12)) admits possibly multiple minimizers, a subgradient iteration is employed instead of the gradient one [21]. This is challenging because the distribution of $\mathbf{s}_t$ is typically unknown in practice. But even if the joint probability distribution functions were available, finding the expectations is not scalable as the dimensionality of $\mathbf{s}_t$ grows.

A common remedy to this challenge is stochastic approximation [4], [23], which corresponds to the following SDG iteration:

$$\boldsymbol{\lambda}_{t+1} = \left[\boldsymbol{\lambda}_t + \mu\nabla\mathcal{D}_t(\boldsymbol{\lambda}_t)\right]^+ \quad \forall t \tag{13a}$$

where μ is a positive (and typically preselected constant) stepsize. The stochastic (sub)gradient $\nabla\mathcal{D}_t(\boldsymbol{\lambda}_t) = \mathbf{A}\mathbf{x}_t + \mathbf{c}_t$ is an unbiased estimate of the true (sub)gradient; that is, $\mathbb{E}[\nabla\mathcal{D}_t(\boldsymbol{\lambda}_t)] = \nabla\mathcal{D}(\boldsymbol{\lambda}_t)$. Hence, the primal $\mathbf{x}_t$ can be found by solving the following instantaneous subproblems, one per t

$$\mathbf{x}_t \in \arg\min_{\mathbf{x}_t\in\mathcal{X}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\lambda}_t). \tag{13b}$$

The iterate $\boldsymbol{\lambda}_{t+1}$ in (13a) depends only on the probability distribution of $\mathbf{s}_t$ through the stochastic (sub)gradient $\nabla\mathcal{D}_t(\boldsymbol{\lambda}_t)$.

[2]Though there may exist other time-dependent policies that generate the optimal solution to (6), our attention is restricted to the one that purely depends on the observed state $\mathbf{s} \in \mathcal{S}$, which can be time-independent [4, Theor. 4.5].

Consequently, the process $\{\boldsymbol{\lambda}_t\}$ is Markov with invariant transition probability when $\mathbf{s}_t$ is stationary. An interesting observation is that since $\nabla \mathcal{D}_t(\boldsymbol{\lambda}_t) := \mathbf{A}\mathbf{x}_t + \mathbf{c}_t$, the dual iteration can be written as [cf., (13a)]

$$\boldsymbol{\lambda}_{t+1}/\mu = [\boldsymbol{\lambda}_t/\mu + \mathbf{A}\mathbf{x}_t + \mathbf{c}_t]^+ \qquad \forall t \tag{14}$$

which coincides with (3b) for $\boldsymbol{\lambda}_t/\mu = \mathbf{q}_t$; see also [4], [14], and [17] for a virtual queue interpretation of this parallelism.

Thanks to its low complexity and robustness to nonstationary scenarios, SDG is widely used in various areas, including adaptive signal processing [24]; stochastic network optimization [4], [14], [15]; and energy management in power grids [8], [17]. For network management, in particular, this iteration entails a cost-delay tradeoff as summarized next; see for example [4].

Proposition 2: If Ψ^* is the optimal cost in (3) under any feasible control policy with the state distribution available, and if a constant stepsize μ is used in (13a), the SDG recursion (13) achieves an $\mathcal{O}(\mu)$-optimal solution in the sense that

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\left[\Psi_t\left(\mathbf{x}_t(\boldsymbol{\lambda}_t)\right)\right] \leq \Psi^* + \mathcal{O}(\mu) \tag{8a}$$

where $\mathbf{x}_t(\boldsymbol{\lambda}_t)$ denotes the decisions obtained from (13b), and it incurs a steady-state queue length $\mathcal{O}(1/\mu)$, namely

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\left[\mathbf{q}_t\right] = \mathcal{O}\left(\frac{1}{\mu}\right). \tag{15b}$$

Proposition 2 asserts that SDG with stepsize μ will asymptotically yield an $\mathcal{O}(\mu)$-optimal solution [21, Prop. 8.2.11], and it will have a steady-state queue length $\mathbf{q}_\infty$ inversely proportional to μ. This optimality gap is standard, because iteration (13a) with a constant stepsize[3] will converge to a neighborhood of the optimum $\boldsymbol{\lambda}^*$ [24]. Under mild conditions, the optimal multiplier is bounded, that is, $\boldsymbol{\lambda}^* = \mathcal{O}(1)$, so that the steady-state queue length $\mathbf{q}_\infty$ naturally scales with $\mathcal{O}(1/\mu)$ since it hovers around $\boldsymbol{\lambda}^*/\mu$; see (14). As a consequence, to achieve near optimality (sufficiently small μ), SDG incurs large average queue lengths and, thus, undesired average delay as per Little's law [4]. To overcome this limitation, we develop next an *online* approach, which can improve SDG's cost-delay tradeoff, while still preserving its affordable complexity and adaptability.

IV. Learn-and-Adapt SDG

Our main approach is derived in this section, by nicely leveraging both learning and optimization tools. Its decentralized implementation is also developed.

A. LA-SDG as a Foresighted Learning Scheme

The intuition behind our LA-SDG approach is to incrementally learn network state statistics from the observed data while adapting resource allocation driven by the learning process. A key element of LA-SDG could be called "foresighted" learning because instead of myopically learning the exact optimal argument from empirical data, LA-SDG maintains the capability to hedge against the risk of "future non-stationarities."

[3]A vanishing stepsize in the stochastic approximation iterations can ensure convergence, but necessarily implies an unbounded queue length as $\mu \to 0$ [4].

Algorithm 1: LA-SDG for Stochastic Network Optimization.

1: **Initialize:** dual iterate $\boldsymbol{\lambda}_1$, empirical dual iterate $\hat{\boldsymbol{\lambda}}_1$, queue length $\mathbf{q}_1$, control variable $\boldsymbol{\theta} = \sqrt{\mu}\log^2(\mu)\cdot\mathbf{1}$, and proper stepsizes μ and $\{\eta_t,\ \forall t\}$.
2: **for** $t = 1, 2\ldots$ **do**
3: **Resource allocation (1st gradient):**
4: Construct the effective dual variable via (17b), observe the current state $\mathbf{s}_t$, and obtain resource allocation $\mathbf{x}_t(\boldsymbol{\gamma}_t)$ by minimizing online Lagrangian (17a).
5: Update the instantaneous queue length $\mathbf{q}_{t+1}$ via

$$\mathbf{q}_{t+1} = \left[\mathbf{q}_t + \left(\mathbf{A}\mathbf{x}_t(\boldsymbol{\gamma}_t) + \mathbf{c}_t\right)\right]^+,\ \forall t. \tag{16}$$

6: **Sample recourse (2nd gradient):**
7: Obtain variable $\mathbf{x}_t(\hat{\boldsymbol{\lambda}}_t)$ by solving online Lagrangian minimization with sample $\mathbf{s}_t$ via (18b).
8: Update the empirical dual variable $\hat{\boldsymbol{\lambda}}_{t+1}$ via (18a).
9: **end for**

The proposed LA-SDG is summarized in Algorithm 1. It involves the queue length $\mathbf{q}_t$ and an empirical dual variable $\hat{\boldsymbol{\lambda}}_t$, along with a bias-control variable $\boldsymbol{\theta}$ to ensure that LA-SDG will attain near optimality in the steady state [cf., Theorems 2 and 3]. At each time slot t, LA-SDG obtains two stochastic gradients using the current $\mathbf{s}_t$: one for online resource allocation, and another one for sample learning/recourse. For the first gradient (lines 3–5), contrary to SDG that relies on the stochastic multiplier estimate $\boldsymbol{\lambda}_t$ [cf., (13b)], LA-SDG minimizes the instantaneous Lagrangian

$$\mathbf{x}_t(\boldsymbol{\gamma}_t) \in \arg\min_{\mathbf{x}_t\in\mathcal{X}} \mathcal{L}_t(\mathbf{x}_t, \boldsymbol{\gamma}_t) \tag{17a}$$

which depends on what we term *effective* multiplier, given by

$$\underbrace{\boldsymbol{\gamma}_t}_{\text{effective multiplier}} = \underbrace{\hat{\boldsymbol{\lambda}}_t}_{\text{statistical learning}} + \underbrace{\mu\mathbf{q}_t - \boldsymbol{\theta}}_{\text{online adaptation}} \qquad \forall t. \tag{17b}$$

Variable $\boldsymbol{\gamma}_t$ also captures the effective price, which is a linear combination of the empirical $\hat{\boldsymbol{\lambda}}_t$ and the queue length $\mathbf{q}_t$, where the control variable μ tunes the weights of these two factors, and $\boldsymbol{\theta}$ controls the bias of $\boldsymbol{\gamma}_t$ in the steady state [15]. As a single pass of SDG "wastes" valuable online samples, LA-SDG resolves this limitation in a learning step by evaluating a second gradient (lines 6–8); that is, LA-SDG simply finds the stochastic gradient of (11) at the previous empirical dual variable $\hat{\boldsymbol{\lambda}}_t$, and implements a gradient ascent update as

$$\hat{\boldsymbol{\lambda}}_{t+1} = \left[\hat{\boldsymbol{\lambda}}_t + \eta_t\left(\mathbf{A}\mathbf{x}_t(\hat{\boldsymbol{\lambda}}_t) + \mathbf{c}_t\right)\right]^+ \qquad \forall t \tag{18a}$$

where η_t is a proper diminishing stepsize, and the "virtual" allocation $\mathbf{x}_t(\hat{\boldsymbol{\lambda}}_t)$ can be found by solving

$$\mathbf{x}_t(\hat{\boldsymbol{\lambda}}_t) \in \arg\min_{\mathbf{x}_t\in\mathcal{X}} \mathcal{L}_t(\mathbf{x}_t, \hat{\boldsymbol{\lambda}}_t). \tag{18b}$$

Note that different from $\mathbf{x}_t(\boldsymbol{\gamma}_t)$ in (17a), the "virtual" allocation $\mathbf{x}_t(\hat{\boldsymbol{\lambda}}_t)$ will not be physically implemented. The multiplicative constant μ in (17b) controls the degree of adaptability, and allows for adaptation even in the steady state ($t \to \infty$), but the vanishing η_t is for learning, as we shall discuss next.

The key idea of LA-SDG is to empower adaptive resource allocation (via $\boldsymbol{\gamma}_t$) with the learning process (effected through $\hat{\boldsymbol{\lambda}}_t$). As a result, the construction of $\boldsymbol{\gamma}_t$ relies on $\hat{\boldsymbol{\lambda}}_t$, but not vice versa. For a better illustration of the effective price (17b), we call $\hat{\boldsymbol{\lambda}}_t$ the statistically learnt price to obtain the exact optimal argument of the expected problem (11). We also call $\mu\mathbf{q}_t$ (which is exactly $\boldsymbol{\lambda}_t$ as shown in (13a)) the online adaptation term since it can track the instantaneous change of system statistics. Intuitively, a large μ will allow the effective policy to quickly respond to instantaneous variations so that the policy gains improved control of queue lengths, while a small μ puts more weight on learning from historical samples so that the allocation strategy will incur less variance in the steady state. In this sense, LA-SDG can attain both statistical efficiency and adaptability.

Distinctly different from SDG that combines statistical learning with resource allocation into a single adaptation step [cf., (13a)], LA-SDG performs these two tasks into two intertwined steps: resource allocation (17), and statistical learning (18). The additional learning step adopts a diminishing stepsize to find the "best empirical" dual variable from all observed network states. This pair of complementary gradient steps endows LA-SDG with its attractive properties. In its transient stage, the extra gradient evaluations and empirical dual variables accelerate the convergence speed of SDG; while in the steady stage, the empirical multiplier approaches the optimal one, which significantly reduces the steady-state queue lengths.

Remark 1: Readers familiar with algorithms on statistical learning and stochastic network optimization can recognize their similarities and differences with LA-SDG.

(P1) SDG in [4] involves only the first part of LA-SDG (1st gradient), where the allocation policy purely relies on stochastic estimates of Lagrange multipliers or instantaneous queue lengths, that is, $\boldsymbol{\gamma}_t = \mu\mathbf{q}_t$. In contrast, LA-SDG further leverages statistical learning from streaming data.

(P2) Several schemes have been developed recently for statistical learning at a scale to find $\hat{\boldsymbol{\lambda}}_t$, namely, SAG in [25] and SAGA in [26]. However, directly applying $\boldsymbol{\gamma}_t = \hat{\boldsymbol{\lambda}}_t$ to allocate resources causes infeasibility. For a finite time t, $\hat{\boldsymbol{\lambda}}_t$ is δ-optimal[4] for (11), and the primal variable $\mathbf{x}_t(\hat{\boldsymbol{\lambda}}_t)$, in turn, is δ-feasible with respect to (7b) that is necessary for (3c). Since $\mathbf{q}_t$ essentially accumulates online constraint violations of (7b), it will grow linearly with t and eventually become unbounded.

B. LA-SDG as a Modified Heavy-Ball Iteration

The heavy-ball iteration belongs to the family of momentum-based first-order methods, and has well-documented acceleration merits in the deterministic setting [27]. Motivated by its convergence speed in solving deterministic problems, stochastic heavy-ball methods have been also pursued recently [10], [13].

[4]Iterate $\hat{\boldsymbol{\lambda}}_t$ is δ-optimal if $\|\hat{\boldsymbol{\lambda}}_t - \boldsymbol{\lambda}^*\| \leq \mathcal{O}(\delta)$, and likewise for δ-feasibility.

The stochastic version of the heavy-ball iteration is [13]

$$\boldsymbol{\lambda}_{t+1} = \boldsymbol{\lambda}_t + \mu\nabla\mathcal{D}_t(\boldsymbol{\lambda}_t) + \beta(\boldsymbol{\lambda}_t - \boldsymbol{\lambda}_{t-1}) \quad \forall t \tag{19}$$

where $\mu > 0$ is an appropriate constant stepsize, $\beta \in [0, 1)$ denotes the momentum factor, and the stochastic gradient $\nabla\mathcal{D}_t(\boldsymbol{\lambda}_t)$ can be found by solving (13b) using heavy-ball iterate $\boldsymbol{\lambda}_t$. This iteration exhibits an attractive convergence rate during the initial stage, but its performance degrades in the steady state. Recently, the performance of momentum iterations (heavy-ball or Nesterov) with constant stepsize μ and momentum factor β, has been proved equivalent to SDG with constant $\mu/(1-\beta)$ per iteration [13]. Since SDG with a large stepsize converges fast at the price of considerable loss in optimality, the momentum methods naturally inherit these attributes.

To see the influence of the momentum term, consider expanding the iteration (19) as

$$\begin{aligned}\boldsymbol{\lambda}_{t+1} &= \boldsymbol{\lambda}_t + \mu\nabla\mathcal{D}_t(\boldsymbol{\lambda}_t) + \beta(\boldsymbol{\lambda}_t - \boldsymbol{\lambda}_{t-1}) \\ &= \boldsymbol{\lambda}_t + \mu\nabla\mathcal{D}_t(\boldsymbol{\lambda}_t) + \beta\left[\mu\nabla\mathcal{D}_{t-1}(\boldsymbol{\lambda}_{t-1})\right. \\ &\quad \left. + \beta(\boldsymbol{\lambda}_{t-1} - \boldsymbol{\lambda}_{t-2})\right] \\ &= \boldsymbol{\lambda}_t + \underbrace{\mu\textstyle\sum_{\tau=1}^{t}\beta^{t-\tau}\nabla\mathcal{D}_\tau(\boldsymbol{\lambda}_\tau)}_{\text{accumulated gradient}} + \underbrace{\beta^t(\boldsymbol{\lambda}_1 - \boldsymbol{\lambda}_0)}_{\text{initial state}}. \end{aligned} \tag{20}$$

The stochastic heavy-ball method will accelerate convergence in the initial stage thanks to the accumulated gradients, and it will gradually forget the initial state. As t increases, however, the algorithm also incurs a worst-case oscillation $\mathcal{O}(\mu/(1-\beta))$, which degrades performance in terms of objective values when compared to SDG with stepsize μ. This is in agreement with the theoretical analysis in [13, Theor. 11].

Different from standard momentum methods, LA-SDG nicely inherits the fast convergence in the initial stage, while reducing the oscillation of stochastic momentum methods in the steady state. To see this, consider two consecutive iterations (17b)

$$\boldsymbol{\gamma}_{t+1} = \hat{\boldsymbol{\lambda}}_{t+1} + \mu\mathbf{q}_{t+1} - \boldsymbol{\theta} \tag{21a}$$

$$\boldsymbol{\gamma}_t = \hat{\boldsymbol{\lambda}}_t + \mu\mathbf{q}_t - \boldsymbol{\theta} \tag{21b}$$

and subtract them, to arrive at

$$\begin{aligned}\boldsymbol{\gamma}_{t+1} &= \boldsymbol{\gamma}_t + \mu\left(\mathbf{q}_{t+1} - \mathbf{q}_t\right) + \left(\hat{\boldsymbol{\lambda}}_{t+1} - \hat{\boldsymbol{\lambda}}_t\right) \\ &= \boldsymbol{\gamma}_t + \mu\nabla\mathcal{D}_t(\boldsymbol{\gamma}_t) + \left(\hat{\boldsymbol{\lambda}}_{t+1} - \hat{\boldsymbol{\lambda}}_t\right) \qquad \forall t. \end{aligned} \tag{22}$$

Here, the equalities in (22) follow from $\nabla\mathcal{D}_t(\boldsymbol{\gamma}_t) = \mathbf{A}\mathbf{x}_t(\boldsymbol{\gamma}_t) + \mathbf{c}_t$ in $\mathbf{q}_t$ recursion (16), and with a sufficiently large $\boldsymbol{\theta}$, the projection in (16) rarely (with sufficiently low probability) takes effect since the steady-state $\mathbf{q}_t$ will hover around $\boldsymbol{\theta}/\mu$; see the details of Theorem 2 and the proof thereof.

Comparing the LA-SDG iteration (22) with the stochastic heavy-ball iteration (19), both of them correct the iterates using the stochastic gradient $\nabla\mathcal{D}_t(\boldsymbol{\gamma}_t)$ or $\nabla\mathcal{D}_t(\boldsymbol{\lambda}_t)$. However, LA-SDG incorporates the variation of a learning sequence (also known as a reference sequence) $\{\hat{\boldsymbol{\lambda}}_t\}$ into the recursion of the main iterate $\boldsymbol{\gamma}_t$, other than the heavy-ball's momentum term $\beta(\boldsymbol{\lambda}_t - \boldsymbol{\lambda}_{t-1})$. Since the variation of learning iterate $\hat{\boldsymbol{\lambda}}_t$ eventually diminishes as t increases, keeping the learning sequence enables LA-SDG to enjoy accelerated convergence in the initial

(transient) stage compared to SDG, while avoiding large oscillation in the steady state compared to the stochastic heavy-ball method. We formally remark on this observation next.

Remark 2: LA-SDG offers a fresh approach to designing stochastic optimization algorithms in a dynamic environment. While directly applying the momentum-based iteration to a stochastic setting may lead to unsatisfactory steady-state performance, it is promising to carefully design a reference sequence that exactly converges to the optimal argument. Therefore, algorithms with improved convergence (e.g., the second-order method in [12]) can also be incorporated as a reference sequence to further enhance the performance of LA-SDG.

C. Complexity and Distributed Implementation of LA-SDG

This section introduces a fully distributed implementation of LA-SDG by exploiting the problem structure of network resource allocation. For notational brevity, collect the variables representing outgoing links from node i in $\mathbf{x}_t^i := \{x_t^{ij}, \forall j \in \mathcal{N}_i\}$ with $\mathcal{N}_i$ denoting the index set of outgoing neighbors of node i. Let also $\mathbf{s}_t^i := [\boldsymbol{\phi}_t^i; c_t^i]$ denote the random state at node i. It will be shown that the learning and allocation decision per time slot t is processed locally per node i based on its local state $\mathbf{s}_t^i$.

To this end, rewrite the Lagrangian minimization for a general dual variable $\boldsymbol{\lambda} \in \mathbb{R}_+^I$ at time t as [cf., (17a) and (18b)]

$$\min_{\mathbf{x}_t \in \mathcal{X}} \sum_{i \in \mathcal{I}} \Psi^i(\mathbf{x}_t^i; \boldsymbol{\phi}_t^i) + \sum_{i \in \mathcal{I}} \lambda^i (\mathbf{A}_{(i,:)} \mathbf{x}_t + c_t^i) \tag{23}$$

where λ^i is the ith entry of vector $\boldsymbol{\lambda}$, and $\mathbf{A}_{(i,:)}$ denotes the ith row of the node-incidence matrix $\mathbf{A}$. Clearly, $\mathbf{A}_{(i,:)}$ selects entries of $\mathbf{x}_t$ associated with the in- and out-links of node i. Therefore, the subproblem at node i is

$$\min_{\mathbf{x}_t^i \in \mathcal{X}^i} \Psi^i(\mathbf{x}_t^i; \boldsymbol{\phi}_t^i) + \sum_{j \in \mathcal{N}_i} (\lambda^j - \lambda^i) x_t^{ji} \tag{24}$$

where $\mathcal{X}^i$ is the feasible set of primal variable $\mathbf{x}_t^i$. In the case of (3d), the feasible set $\mathcal{X}$ can be written as a Cartesian product of sets $\{\mathcal{X}^i, \forall i\}$, so that the projection of $\mathbf{x}_t$ to $\mathcal{X}$ is equivalent to separate projections of $\mathbf{x}_t^i$ onto $\mathcal{X}^i$. Note that $\{\lambda^j, \forall j \in \mathcal{N}_i\}$ will be available at node i by exchanging information with the neighbors per time t. Hence, given the effective multipliers γ_t^j (jth entry of $\boldsymbol{\gamma}_t$) from its outgoing neighbors in $j \in \mathcal{N}_i$, node i is able to form an allocation decision $\mathbf{x}_t^i(\boldsymbol{\gamma}_t)$ by solving the convex programs (24) with $\lambda^j = \gamma_t^j$; see also (17a). Needless to mention, q_t^i can be locally updated via (16), that is

$$q_{t+1}^i = \left[q_t^i + \left(\sum_{j: i \in \mathcal{N}_j} x_t^{ji}(\boldsymbol{\gamma}_t) - \sum_{j \in \mathcal{N}_i} x_t^{ij}(\boldsymbol{\gamma}_t) + c_t^i \right) \right]^+ \tag{25}$$

where $\{x_t^{ji}(\boldsymbol{\gamma}_t)\}$ are the local measurements of arrival (departure) workloads from (to) its neighbors.

Likewise, the tentative primal variable $\mathbf{x}_t^i(\hat{\boldsymbol{\lambda}}_t)$ can be obtained at each node locally by solving (24) using the current sample $\mathbf{s}_t^i$ again with $\lambda^i = \hat{\lambda}_t^i$. By sending $\mathbf{x}_t^i(\hat{\boldsymbol{\lambda}}_t)$ to its outgoing neighbors, node i can update the empirical multiplier $\hat{\lambda}_{t+1}^i$ via

$$\hat{\lambda}_{t+1}^i = \left[\hat{\lambda}_t^i + \eta_t \left(\sum_{j: i \in \mathcal{N}_j} x_t^{ji}(\hat{\boldsymbol{\lambda}}_t) - \sum_{j \in \mathcal{N}_i} x_t^{ij}(\hat{\boldsymbol{\lambda}}_t) + c_t^i \right) \right]^+ \tag{26}$$

which, together with the local queue length q_{t+1}^i, also implies that the next γ_{t+1}^i can be obtained locally.

Compared with the classic SDG recursion (13a)–(13b), the distributed implementation of LA-SDG incurs only a factor of 2 increase in computational complexity. Next, we will further analytically establish that it can improve the delay of SDG by an order of magnitude with the same order of the optimality gap.

V. Optimality and Stability of LA-SDG

This section presents the performance analysis of LA-SDG, which will rely on the following four assumptions.

Assumption 1: The state $\mathbf{s}_t$ is bounded and i.i.d. over time t.

Assumption 2: $\Psi_t(\mathbf{x}_t)$ is proper, σ-strongly convex, lower semicontinuous, and has L_{p}-Lipschitz continuous gradient. Also, $\Psi_t(\mathbf{x}_t)$ is nondecreasing w.r.t. all entries of $\mathbf{x}_t$ over $\mathcal{X}$.

Assumption 3: There exists a stationary policy $\boldsymbol{\chi}(\cdot)$ satisfying $\boldsymbol{\chi}(\mathbf{s}_t) \in \mathcal{X}$ for all $\mathbf{s}_t$, and $\mathbb{E}[\mathbf{A}\boldsymbol{\chi}(\mathbf{s}_t) + \mathbf{c}_t] \leq -\boldsymbol{\zeta}$, where $\boldsymbol{\zeta} > \mathbf{0}$ is a slack vector constant.

Assumption 4: For any time t, the magnitude of the constraint is bounded, that is, $\|\mathbf{A}\mathbf{x}_t + \mathbf{c}_t\| \leq M,\ \forall \mathbf{x}_t \in \mathcal{X}$.

Assumption 1 is typical in stochastic network resource allocation [14], [15], [28], and can be relaxed to an ergodic and stationary setting following [20], [29]. Assumption 2 requires the primal objective to be well behaved, meaning that it is bounded from below and has a unique optimal solution. Note that nondecreasing costs with increased resources are easily guaranteed with, e.g., exponential and quadratic functions in our simulations. In addition, Assumption 2 ensures that the dual function has favorable properties, which are important for the ensuring stability analysis. Assumption 3 is Slater's condition, which guarantees the existence of a bounded optimal Lagrange multiplier [21], and is also necessary for queue stability [4]. Assumption 4 guarantees boundedness of the gradient of the instantaneous dual function, which is common in performance analysis of stochastic gradient-type algorithms [30].

Building upon the desirable properties of the primal problem, we next show that the corresponding dual function satisfies both smoothness and quadratic growth properties [31], [32], which will be critical to the subsequent analysis.

Lemma 2: Under Assumption 2, the dual function $\mathcal{D}(\boldsymbol{\lambda})$ in (11) is L_{d}-smooth, where $L_{\mathrm{d}} = \rho(\mathbf{A}^\top \mathbf{A})/\sigma$, and $\rho(\mathbf{A}^\top \mathbf{A})$ denotes the spectral radius of $\mathbf{A}^\top \mathbf{A}$. In addition, if $\boldsymbol{\lambda}$ lies in a compact set, there always exists a constant ϵ such that $\mathcal{D}(\boldsymbol{\lambda})$ satisfies the following quadratic growth property:

$$\mathcal{D}(\boldsymbol{\lambda}^*) - \mathcal{D}(\boldsymbol{\lambda}) \geq \frac{\epsilon}{2} \|\boldsymbol{\lambda}^* - \boldsymbol{\lambda}\|^2 \tag{27}$$

where $\boldsymbol{\lambda}^*$ is the optimal multiplier for the dual problem (11).

Proof: See Appendix A in the online version [33]. ■

We start with the convergence of the empirical dual variables $\hat{\boldsymbol{\lambda}}_t$. Note that the update of $\hat{\boldsymbol{\lambda}}_t$ is a standard learning iteration from historical data, and it is not affected by future resource allocation decisions. Therefore, the theoretical result on SDG with diminishing stepsize is directly applicable [30, Sec. 2.2].

Lemma 3: Let $\hat{\boldsymbol{\lambda}}_t$ denote the empirical dual variable in Algorithm 1, and $\boldsymbol{\lambda}^*$ the optimal argument for the dual problem (11). If the stepsize is chosen as $\eta_t = \frac{\alpha D}{M\sqrt{t}}$, $\forall t$, with a constant $\alpha > 0$, a sufficient large constant $D > 0$, and M as in Assumption 4, then it holds that

$$\mathbb{E}\left[\mathcal{D}(\boldsymbol{\lambda}^*) - \mathcal{D}(\hat{\boldsymbol{\lambda}}_t)\right] \leq \max\{\alpha, \alpha^{-1}\}\frac{DM}{\sqrt{t}} \tag{28}$$

where the expectation is over all the random states $\mathbf{s}_t$ up to t.

Lemma 3 asserts that using a diminishing stepsize, the dual function value converges sublinearly to the optimal value in expectation. In principle, D is the radius of the feasible set for the dual variable $\boldsymbol{\lambda}$ [30, Sec. 2.2]. However, as the optimal multiplier $\boldsymbol{\lambda}^*$ is bounded according to Assumption 3, one can always estimate a large enough D, and the estimation error will only affect the constant of the suboptimality bound (28) through the scalar α. The suboptimality bound in Lemma 3 holds in expectation, which averages over all possible sample paths $\{\mathbf{s}_1, \ldots, \mathbf{s}_t\}$.

As a complement to Lemma 3, the almost sure convergence of the empirical dual variables is established next to characterize the performance of each individual sample path.

Theorem 1: For the sequence of empirical multipliers $\{\hat{\boldsymbol{\lambda}}_t\}$ in Algorithm 1, if the stepsizes are chosen as $\eta_t = \frac{\alpha D}{M\sqrt{t}}$, $\forall t$, with constants α, M, D defined in Lemma 3, it holds that

$$\lim_{t\to\infty} \hat{\boldsymbol{\lambda}}_t = \boldsymbol{\lambda}^*, \quad \text{w.p.1} \tag{29}$$

where $\boldsymbol{\lambda}^*$ is the optimal dual variable for the expected dual function minimization (11).

Proof: The proof follows the steps in [21, Proposition 8.2.13], which is omitted here.

Building upon the asymptotic convergence of empirical dual variables for statistical learning, it becomes possible to analyze the online performance of LA-SDG. Clearly, the online resource allocation $\mathbf{x}_t$ is a function of the effective dual variable $\boldsymbol{\gamma}_t$ and the instantaneous network state $\mathbf{s}_t$ [cf. (17a)]. Therefore, the next step is to show that the effective dual variable $\boldsymbol{\gamma}_t$ also converges to the optimal argument of the expected problem (11), which would establish that the online resource allocation $\mathbf{x}_t$ is asymptotically optimal. However, directly analyzing the trajectory of $\boldsymbol{\gamma}_t$ is nontrivial, because the queue length $\{\mathbf{q}_t\}$ is coupled with the reference sequence $\{\hat{\boldsymbol{\lambda}}_t\}$ in $\boldsymbol{\gamma}_t$. To address this issue, rewrite the recursion of $\boldsymbol{\gamma}_t$ as

$$\boldsymbol{\gamma}_{t+1} = \boldsymbol{\gamma}_t + (\hat{\boldsymbol{\lambda}}_{t+1} - \hat{\boldsymbol{\lambda}}_t) + \mu(\mathbf{q}_{t+1} - \mathbf{q}_t) \quad \forall t \tag{30}$$

where the update of $\boldsymbol{\gamma}_t$ depends on the variations of $\hat{\boldsymbol{\lambda}}_t$ and $\mathbf{q}_t$. We will first study the asymptotic behavior of queue lengths $\mathbf{q}_t$, and then derive the analysis of $\boldsymbol{\gamma}_t$ using the convergence of $\hat{\boldsymbol{\lambda}}_t$ in (29), and the recursion (30).

Define the time-varying target $\tilde{\boldsymbol{\theta}}_t = \boldsymbol{\lambda}^* - \hat{\boldsymbol{\lambda}}_t + \boldsymbol{\theta}$, which is the optimality residual of statistical learning $\boldsymbol{\lambda}^* - \hat{\boldsymbol{\lambda}}_t$ plus the bias-control variable $\boldsymbol{\theta}$. Per Theorem 1, it readily follows that $\lim_{t\to\infty} \tilde{\boldsymbol{\theta}}_t = \boldsymbol{\theta}$, w.p.1. By showing that $\mathbf{q}_t$ is attracted towards the time-varying target $\tilde{\boldsymbol{\theta}}_t/\mu$, we will further derive the stability of queue lengths.

Lemma 4: With $\mathbf{q}_t$ and μ denoting queue length and stepsize, there exists a constant $B = \Theta(1/\sqrt{\mu})$, and a finite time $T_B < \infty$, such that for all $t \geq T_B$, if $\|\mathbf{q}_t - \tilde{\boldsymbol{\theta}}_t/\mu\| > B$, it holds in LA-SDG that

$$\mathbb{E}\left[\left\|\mathbf{q}_{t+1} - \tilde{\boldsymbol{\theta}}_t/\mu\right\| \middle| \mathbf{q}_t\right] \leq \left\|\mathbf{q}_t - \tilde{\boldsymbol{\theta}}_t/\mu\right\| - \sqrt{\mu}, \text{ w.p.1.} \tag{31}$$

Proof: See Appendix B in the online version [33].

Lemma 4 reveals that when $\mathbf{q}_t$ is large and deviates from the time-varying target $\tilde{\boldsymbol{\theta}}_t/\mu$, it will be bounced back toward the target in the next time slot. Upon establishing this drift behavior of queues, we are on track to establish queue stability.

Theorem 2: With $\mathbf{q}_t, \boldsymbol{\theta}$, and μ defined in (17b), there exists a constant $\tilde{B} = \Theta(1/\sqrt{\mu})$ such that the queue length under LA-SDG converges to a neighborhood of $\boldsymbol{\theta}/\mu$ as

$$\liminf_{t\to\infty} \|\mathbf{q}_t - \boldsymbol{\theta}/\mu\| \leq \tilde{B}, \quad \text{w.p.1.} \tag{32a}$$

In addition, if we choose $\boldsymbol{\theta} = \mathcal{O}(\sqrt{\mu}\log^2(\mu))$, the long-term average expected queue length satisfies

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\left[\mathbf{q}_t\right] = \mathcal{O}\left(\frac{\log^2(\mu)}{\sqrt{\mu}}\right), \text{ w.p.1.} \tag{32b}$$

Proof: See Appendix C in the online version [33]. ■

Theorem 2 in (32a) asserts that the sequence of queue iterates converges (in the infimum sense) to a neighborhood of $\boldsymbol{\theta}/\mu$, where the radius of neighborhood region scales as $1/\sqrt{\mu}$. In addition to the sample path result, (32b) demonstrates that with a specific choice of $\boldsymbol{\theta}$, the queue length averaged over all sample paths will be $\mathcal{O}\left(\log^2(\mu)/\sqrt{\mu}\right)$. Together with Theorem 1, it suffices to have the effective dual variable converge to a neighborhood of the optimal multiplier $\boldsymbol{\lambda}^*$; that is, $\liminf_{t\to\infty} \boldsymbol{\gamma}_t = \boldsymbol{\lambda}^* + \mu\mathbf{q}_t - \boldsymbol{\theta} = \boldsymbol{\lambda}^* + \mathcal{O}(\sqrt{\mu})$, w.p.1. Notice that the SDG iterate $\boldsymbol{\lambda}_t$ in (13a) will also converge to a neighborhood of $\boldsymbol{\lambda}^*$. Therefore, intuitively LA-SDG will behave similar to SDG in the steady state, and its asymptotic performance follows from that of SDG. However, the difference is that through a careful choice of $\boldsymbol{\theta}$, for a sufficiently small μ, LA-SDG can improve the queue length $\mathcal{O}(1/\mu)$ under SDG by an order of magnitude.

In addition to feasibility, we formally establish in the next theorem that LA-SDG is asymptotically near-optimal.

Theorem 3: Let Ψ^* be the optimal objective value of (3) under any feasible policy with distribution information about the state fully available. If the control variable is chosen as $\boldsymbol{\theta} = \mathcal{O}(\sqrt{\mu}\log^2(\mu))$, then with a sufficiently small μ, LA-SDG yields a near-optimal solution for (3) in the sense that

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\left[\Psi_t\left(\mathbf{x}_t(\boldsymbol{\gamma}_t)\right)\right] \leq \Psi^* + \mathcal{O}(\mu), \text{ w.p.1} \tag{33}$$

where $\mathbf{x}_t(\boldsymbol{\gamma}_t)$ denotes the real-time operations obtained from the Lagrangian minimization (17a).

Proof: See Appendix D in the online version [33]. ■

Combining Theorems 2 and 3, we are ready to state that by setting $\boldsymbol{\theta} = \mathcal{O}(\sqrt{\mu}\log^2(\mu))$, LA-SDG is asymptotically $\mathcal{O}(\mu)$-optimal with an average queue length $\mathcal{O}(\log^2(\mu)/\sqrt{\mu})$. This result implies that LA-SDG is able to achieve a near-optimal cost-delay tradeoff $[\mu, \log^2(\mu)/\sqrt{\mu}]$; see [4], [19]. Comparing with the standard tradeoff $[\mu, 1/\mu]$ under SDG, the learn-and-adapt design of LA-SDG markedly improves the online performance in terms of delay. Note that a better tradeoff $[\mu, \log^2(\mu)]$ has been derived in [15] under the so-termed local polyhedral assumption. Observe though, that the considered setting in [15] is different from the one here. While the network state set $\mathcal{S}$ and the action set $\mathcal{X}$ in [15] are discrete and countable, LA-SDG allows continuous $\mathcal{S}$ and $\mathcal{X}$ with possibly infinite elements, and still be amenable to efficient and scalable online operations.

VI. Numerical Tests

This section presents numerical tests to confirm the analytical claims and demonstrate the merits of the proposed approach. We consider the geographical load balancing network of Section II-B with $K = 10$ data centers, and $J = 10$ mapping nodes. Performance is tested in terms of the time-averaged instantaneous network cost in (4), namely

$$\Psi_t(\mathbf{x}_t) := \sum_{k\in\mathcal{K}} p_t^k \left((x_t^{k0})^2 - e_t^k\right) + \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} b_t^{jk}(x_t^{jk})^2 \quad (34)$$

where the energy price p_t^k is uniformly distributed over $[10, 30]$; samples of the renewable supply $\{e_t^k\}$ are generated uniformly over $[10, 100]$; and the per-unit bandwidth cost is set to $b_t^{jk} = 40/\bar{x}^{jk}, \forall k, j$, with bandwidth limits $\{\bar{x}^{jk}\}$ generated from a uniform distribution within $[100, 200]$. The capacities at data centers $\{\bar{x}_t^{k0}\}$ are uniformly generated from $[100, 200]$. The delay-tolerant workloads $\{c_t^j\}$ arrive at each mapping node j according to a uniform distribution over $[10, 100]$. Clearly, the cost (34) and the state $\mathbf{s}_t$ here satisfy Assumptions 1 and 2. Finally, the stepsize is $\eta_t = 1/\sqrt{t}, \forall t$, the tradeoff variable is $\mu = 0.2$, and the bias correction vector is chosen as $\boldsymbol{\theta} = 100\sqrt{\mu}\log^2(\mu)\mathbf{1}$ by default, but manually tuned in Figs. 5–6. We introduce two benchmarks: SDG in (13a) (see, e.g., [4]), and the projected stochastic heavy-ball in (19) and $\beta = 0.5$ by default (see, e.g., [10]). Unless otherwise stated, all simulated results were averaged over 50 Monte Carlo realizations.

Performance is first compared in terms of the time-averaged cost, and the instantaneous queue length in Figs. 2 and 3. For the network cost, SDG, LA-SDG, and the heavy-ball iteration with $\beta = 0.5$ converge to almost the same value, while the heavy-ball method with a larger momentum factor $\beta = 0.99$ exhibits a pronounced optimality loss. LA-SDG and heavy-ball exhibit faster convergence than SDG as their running-average costs quickly arrive at the optimal operating phase by leveraging the learning process or the momentum acceleration. In this test, LA-SDG exhibits a much lower delay as its aggregated queue length is only 10% of that for the heavy-ball method with $\beta = 0.5$ and 4% of that for SDG. By using a larger β, the heavy-ball method incurs a much lower queue length relative to that of SDG, but still slightly higher than that of LA-SDG. Clearly, our learn-and-adapt procedure improves the delay performance.

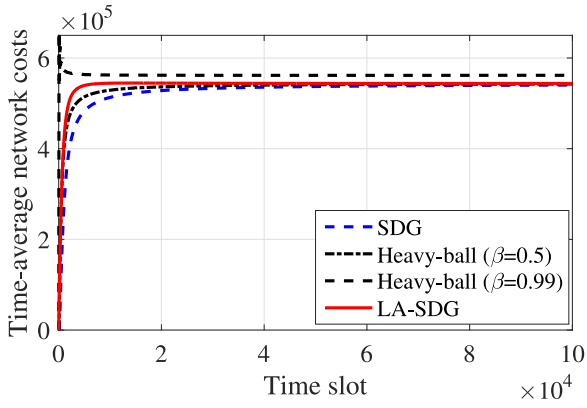


Fig. 2. Comparison of time-averaged network costs.

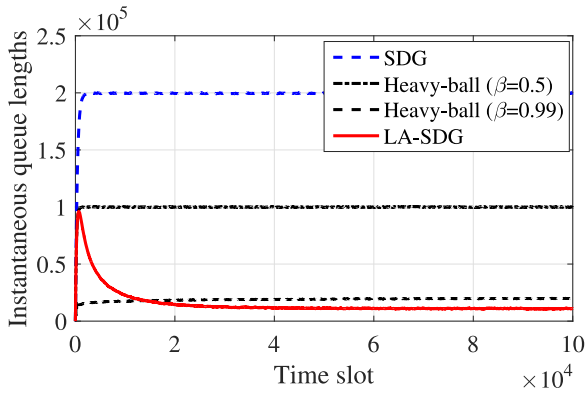


Fig. 3. Instantaneous queue lengths summed over all nodes.

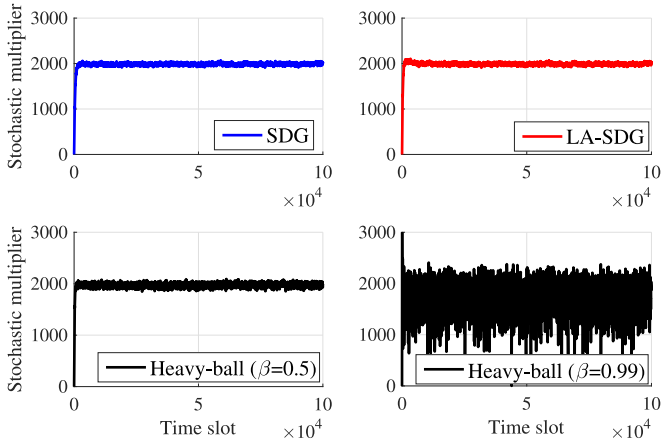


Fig. 4. Evolution of stochastic multipliers at mapping node 1 ($\mu = 0.2$).

Recall that the instantaneous resource allocation can be viewed as a function of the dual variable; see Proposition 1. Hence, the performance differences in Figs. 2–3 can be also anticipated by the different behavior of dual variables. In Fig. 4, the evolution of stochastic dual variables is plotted for a single Monte Carlo realization; that is, the dual iterate in (13a) for SDG, the momentum iteration in (19) for the heavy-ball method, and the effective multiplier in (17b) for LA-SDG. As illustrated in (20), the performance of momentum iterations is similar to SDG with larger stepsize $\mu/(1-\beta)$. This is corroborated by

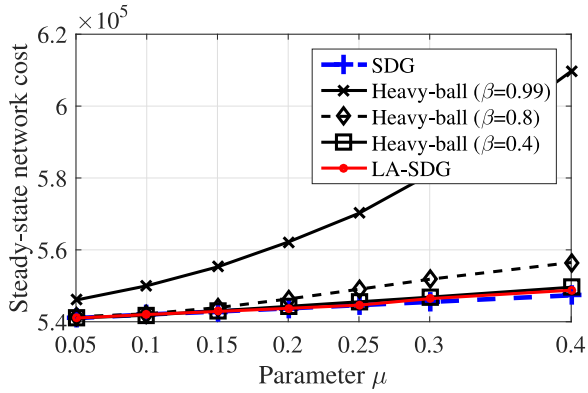


Fig. 5. Comparison of steady-state network costs (after 10^6 slots).

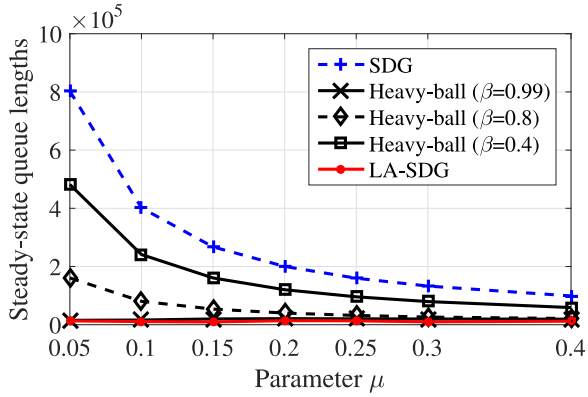


Fig. 6. Steady-state queue lengths summed over all nodes (after 10^6 slots).

Fig. 4, where the stochastic momentum iterate with $\beta = 0.5$ behaves similar to the dual iterates of SDG and LA-SDG, but its oscillation becomes prohibitively high with a larger factor $\beta = 0.99$, which nicely explains the higher cost in Fig. 2.

Since the cost-delay performance is sensitive to the choice of parameters μ and β, extensive experiments are further conducted among three algorithms using different values of μ and β in Figs. 5 and 6. The steady-state performance is evaluated by running algorithms for sufficiently long time, up to 10^6 slots. The steady-state costs of all three algorithms increase as μ becomes larger, and the costs of LA-SDG and the heavy-ball with small momentum factor $\beta = 0.4$ are close to that of SDG, while the costs of the heavy-ball with larger momentum factors $\beta = 0.8$ and $\beta = 0.99$ are much larger than that of SDG. Considering steady-state queue lengths (network delay), LA-SDG exhibits an order of magnitude lower amount than those of SDG and the heavy-ball with small β, under all choices of μ. Note that the heavy-ball with a sufficiently large factor $\beta = 0.99$ also has a very low queue length, but it incurs a higher cost than LA-SDG in Fig. 5 due to higher steady-state oscillation in Fig. 4.

VII. Concluding Remarks

Fast convergent resource allocation and low service delay are highly desirable attributes of stochastic network management approaches. Leveraging recent advances in online learning and momentum-based optimization, a novel online approach termed LA-SDG was developed in this paper. LA-SDG learns the network state statistics through an additional sample recourse procedure. The associated novel iteration can be nicely interpreted as a modified heavy-ball recursion with an extra correction step to mitigate steady-state oscillations. It was analytically established that LA-SDG achieves a near-optimal cost-delay tradeoff $[\mu, \log^2(\mu)/\sqrt{\mu}]$, which is better than $[\mu, 1/\mu]$ of SDG, at the cost of only one extra gradient evaluation per new datum. Our future research agenda includes novel approaches to further hedge against nonstationarity, and improved learning schemes to uncover other valuable statistical patterns from historical data.

Acknowledgment

The authors would like to thank Profs. Xin Wang, Longbo Huang, and Jia Liu for helpful discussions.

References

[1] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Autom. Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.

[2] S. H. Low and D. E. Lapsley, "Optimization flow control-I: Basic algorithm and convergence," *IEEE/ACM Trans. Netw.*, vol. 7, no. 6, pp. 861–874, Dec. 1999.

[3] L. Georgiadis, M. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Found. Trends Netw.*, vol. 1, pp. 1–144, 2006.

[4] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures Commun. Netw.*, vol. 3, no. 1, pp. 1–211, 2010.

[5] T. Chen, X. Wang, and G. B. Giannakis, "Cooling-aware energy and workload management in data centers via stochastic optimization," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 2, pp. 402–415, Mar. 2016.

[6] T. Chen, Y. Zhang, X. Wang, and G. B. Giannakis, "Robust workload and energy management for sustainable data centers," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 651–664, Mar. 2016.

[7] J. Gregoire, X. Qian, E. Frazzoli, A. de La Fortelle, and T. Wongpiromsarn, "Capacity-aware backpressure traffic signal control," *IEEE Trans. Control Netw. Syst.*, vol. 2, no. 2, pp. 164–173, Jun. 2015.

[8] S. Sun, M. Dong, and B. Liang, "Distributed real-time power balancing in renewable-integrated power grids with storage and flexible loads," *IEEE Trans. Smart Grid*, vol. 7, no. 5, pp. 2337–2349, Sep. 2016.

[9] A. Beck, A. Nedic, A. Ozdaglar, and M. Teboulle, "An $\mathcal{O}(1/k)$ gradient method for network resource allocation problems," *IEEE Trans. Control Netw. Syst.*, vol. 1, no. 1, pp. 64–73, Mar. 2014.

[10] J. Liu, A. Eryilmaz, N. B. Shroff, and E. S. Bentley, "Heavy-ball: A new approach to tame delay and convergence in wireless network optimization," in *Proc. IEEE INFOCOM*, San Francisco, CA, USA, Apr. 2016, pp. 1–9.

[11] E. Wei, A. Ozdaglar, and A. Jadbabaie, "A distributed Newton method for network utility maximization-I: Algorithm," *IEEE Trans. Autom. Control*, vol. 58, no. 9, pp. 2162–2175, Sep. 2013.

[12] M. Zargham, A. Ribeiro, and A. Jadbabaie, "Accelerated backpressure algorithm," Feb. 2013. [Online]. Available https://arxiv.org/abs/1302.1475

[13] K. Yuan, B. Ying, and A. H. Sayed, "On the influence of momentum acceleration on online learning," *J. Mach. Learning Res.*, vol. 17, no. 192, pp. 1–66, 2016.

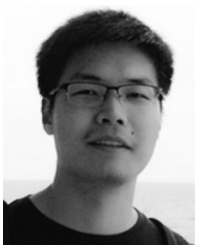
[14] L. Huang and M. J. Neely, "Delay reduction via Lagrange multipliers in stochastic network optimization," *IEEE Trans. Autom. Control*, vol. 56, no. 4, pp. 842–857, Apr. 2011.

[15] L. Huang, X. Liu, and X. Hao, "The power of online learning in stochastic network optimization," *ACM SIGMETRICS*, vol. 42, no. 1, pp. 153–165, Jun. 2014.

[16] V. S. Borkar, "Convex analytic methods in Markov decision processes," in *Handbook of Markov Decision Processes*. New York, NY, USA: Springer, 2002, pp. 347–375.

[17] R. Urgaonkar, B. Urgaonkar, M. Neely, and A. Sivasubramaniam, "Optimal power cost management using stored energy in data centers," in *Proc. ACM SIGMETRICS*, San Jose, CA, USA, Jun. 2011, pp. 221–232.

[18] T. Chen, A. G. Marques, and G. B. Giannakis, "DGLB: Distributed stochastic geographical load balancing over cloud networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 7, pp. 1866–1880, Jul. 2017.
[19] A. G. Marques, L. M. Lopez-Ramos, G. B. Giannakis, J. Ramos, and A. J. Caamaño, "Optimal cross-layer resource allocation in cellular networks using channel-and queue-state information," *IEEE Trans. Veh. Technol.*, vol. 61, no. 6, pp. 2789–2807, Jul. 2012.
[20] A. Ribeiro, "Ergodic stochastic optimization algorithms for wireless communication and networking," *IEEE Trans. Signal Process.*, vol. 58, no. 12, pp. 6369–6386, Dec. 2010.
[21] D. P. Bertsekas, A. Nedic, and A. Ozdaglar, *Convex Analysis and Optimization*. Belmont, MA, USA: Athena Sci., 2003.
[22] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on Stochastic Programming: Modeling and Theory*. Philadelphia, PA, USA: SIAM, 2009.
[23] H. Robbins and S. Monro, "A stochastic approximation method," *Annals Math. Stat.*, vol. 22, no. 3, pp. 400–407, Sep. 1951.
[24] V. Kong and X. Solo, *Adaptive Signal Processing Algorithms*. Upper Saddle River, NJ, USA: Prentice-Hall, 1995.
[25] N. L. Roux, M. Schmidt, and F. R. Bach, "A stochastic gradient method with an exponential convergence rate for finite training sets," in *Proc. Adv. Neural Inform. Process. Syst.*, Lake Tahoe, NV, USA, Dec. 2012, pp. 2663–2671.
[26] A. Defazio, F. Bach, and S. Lacoste-Julien, "SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives," in *Proc. Adv. Neural Info. Process. Syst.*, Montreal, QC, Canada, Dec. 2014, pp. 1646–1654.
[27] B. T. Polyak, *Introduction to Optimization*. New York, NY, USA: Optimization Software, 1987.
[28] A. Eryilmaz and R. Srikant, "Joint congestion control, routing, and MAC for stability and fairness in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1514–1524, Aug. 2006.
[29] J. C. Duchi, A. Agarwal, M. Johansson, and M. I. Jordan, "Ergodic mirror descent," *SIAM J. Optim.*, vol. 22, no. 4, pp. 1549–1578, 2012.
[30] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, "Robust stochastic approximation approach to stochastic programming," *SIAM J. Optim.*, vol. 19, no. 4, pp. 1574–1609, 2009.
[31] M. Hong and Z.-Q. Luo, "On the linear convergence of the alternating direction method of multipliers," *Math. Program.*, vol. 162, pp. 165–199, 2017.
[32] H. Karimi, J. Nutini, and M. Schmidt, "Linear convergence of gradient and proximal-gradient methods under the Polyak-Łojasiewicz condition," *Proc. Eur. Conf. Mach. Learn.*, Riva del Garda, Italy, Sep. 2016, pp. 795–811.
[33] T. Chen, L. Qing, and G. B. Giannakis "Learn-and-adapt stochastic dual gradients for network resource allocation," Mar. 2017. [Online]. Available: https://arxiv.org/pdf/1703.01673.pdf



Tianyi Chen (S'14) received the B.Eng. degree (with highest hons.) in communication science and engineering from Fudan University, Shanghai, China, in 2014, and the M.Sc. degree in electrical and computer engineering (ECE) from the University of Minnesota (UMN), Minneapolis, MN, USA, in 2016, where he has been working toward the Ph.D. degree in electrical engineering.

His research interests include online learning, online convex optimization, and stochastic network optimization with applications to smart grids, sustainable cloud networks, and Internet-of-Things.

Dr. Chen was one of the Best Student Paper Award finalists of the Asilomar Conference on Signals, Systems, and Computers. He received the National Scholarship from China in 2013, UMN ECE Department Fellowship in 2014, and the UMN Doctoral Dissertation Fellowship in 2017.

Qing Ling (SM'15) received the B.E. degree in automation and the Ph.D. degree in control theory and control engineering from the University of Science and Technology of China, Hefei, Anhui, China, in 2001 and 2006, respectively.

He was a Postdoctoral Research Fellow with the Department of Electrical and Computer Engineering, Michigan Technological University, Houghton, MI, USA, from 2006 to 2009, and an Associate Professor with the Department of Automation, University of Science and Technology of China, Hefei, from 2009 to 2017. He is currently a Professor with the School of Data and Computer Science, Sun Yat-Sen University, Guangzhou, Guangdong, China. His research interests include decentralized network optimization and its applications.

Dr. Ling received the 2017 IEEE Signal Processing Society Young Author Best Paper Award as a supervisor, and the 2017 International Consortium of Chinese Mathematicians Distinguished Paper Award. He is an Associate Editor of IEEE SIGNAL PROCESSING LETTERS.

Georgios B. Giannakis (F'97) received the Diploma in Electrical Engineering degree from the National Technical University of Athens, Athens, Greece, in 1981. He received the M.Sc. degree in electrical engineering in 1983, the M.Sc. degree in mathematics in 1986, and the Ph.D. degree in electrical engineering in 1986, all from the University of Southern California Los Angeles, CA, USA.

He was with the University of Virginia, Charlottesville, VA, USA, from 1987 to 1998, and since 1999 he has been a Professor with the University of Minnesota, Minneapolis, MN, USA, and serves as the Director of the Digital Technology Center. His general interests span the areas of communications, networking, and statistical signal processing—subjects on which he has published more than 400 journal papers, 700 conference papers, 25 book chapters, two edited books, and two research monographs (h-index 128). Current research focuses on learning from Big Data, wireless cognitive radios, and network science with applications to social, brain, and power networks with renewables. He is the (co-) inventor of 30 patents issued.

Dr. Giannakis holds an Endowed Chair in Wireless Telecommunications as well as a University of Minnesota McKnight Presidential Chair, and the (co-)recipient of nine best journal paper awards from the IEEE Signal Processing and Communications Societies, including the G. Marconi Prize Paper Award in Wireless Communications. He also received Technical Achievement Awards from the SP Society (2000), from EURASIP (2005), a Young Faculty Teaching Award, the G. W. Taylor Award for Distinguished Research from the University of Minnesota, and the IEEE Fourier Technical Field Award (2015). He is a Fellow of EURASIP, and has served the IEEE in a number of posts, including that of a Distinguished Lecturer for the IEEE-SP Society.