HARNESSING BANDIT ONLINE LEARNING TO LOW-LATENCY FOG COMPUTING

Tianyi Chen and Georgios B. Giannakis

Dept. of ECE and DTC, University of Minnesota, USA

ABSTRACT

This paper focuses on the online fog computing tasks in the Internet-of-Things (IoT), where online decisions must flexibly adapt to the changing user preferences (loss functions), and the temporally unpredictable availability of resources (constraints). Tailored for such human-in-the-loop systems where the loss functions are hard to model, a family of bandit online saddle-point (BanSP) schemes are developed, which adaptively adjust the online operations based on (possibly multiple) bandit feedback of the loss functions, and the changing environment. Performance here is assessed by: i) *dynamic regret* that generalizes the widely used static regret; and, ii) *fit* that captures the accumulated amount of constraint violations. Specifically, BanSP is proved to simultaneously yield sub-linear dynamic regret and fit, provided that the best dynamic solutions vary slowly over time. Numerical tests on fog computing tasks corroborate that BanSP offers desired performance under such limited information.

***Index Terms*—** Computation offloading, fog computing, Internet of Things, online learning, bandit convex optimization.

1. INTRODUCTION

Internet-of-Things (IoT) envisions an intelligent infrastructure of networked smart devices offering task-specific monitoring and control services [1]. Leveraging advances in embedded systems, contemporary IoT devices are featured with *small-size* and *low-power* designs, but their computation and communication capabilities are limited. Along with other features of IoT, such as *extreme heterogeneity* and *unpredictable dynamics*, the need arises for innovations in network design and management to allow for adaptive online service provisioning, subject to stringent delay constraints.

From the network design vantage point, *fog* is viewed as a promising architecture for IoT that distributes computation and communication closer to the IoT users, along the cloud-to-things continuum [2]. In the fog computing paradigm, service provisioning starts at the network edge, e.g., smartphones, and high-tech routers, and only a portion of tasks will be offloaded to the cloud for further processing (a.k.a. computation offloading) [3]. Existing approaches for computation offloading mainly focus on static or stationary settings [4, 5]; see also [4, 6]. Nevertheless, static settings cannot capture the changing IoT environment, and the stationarity commonly assumed in stochastic optimization may not hold in practice, especially when involving human participation as in IoT.

Indeed, the *primary goal* of this paper is an algorithmic pursuit of online network optimization suitable for emerging tasks in IoT. Focusing on such algorithmic challenges, online convex optimization (OCO) is a promising methodology for sequential tasks with well-documented merits, especially when the sequence of costs varies in a possibly adversarial manner [7]. Tailored for fog computing setups, OCO with time-varying constraints was first studied in [8], along with its adaptive variant in [9], and the optimal regret bound in this setting was first established in [10]. Yet, the approaches in [8–10] remain operational under the premise that the loss functions are *explicitly known*, or, their gradients are readily available. Clearly, none of these two assumptions can be satisfied, because i) the loss function capturing user dissatisfaction, e.g., service latency or reliability, is hard to model; and, ii) even if modeling is possible in theory, the low-power IoT devices may not afford the complexity of running learning tools such as deep neural networks "on-the-fly."

In this context, alternative online schemes have been advocated leveraging point-wise values of loss functions (partial-information feedback) rather than their gradients (full-information feedback). They are termed bandit convex optimization (BCO) in machine learning [11, 12]. Building on full-information precursors [8–10], the present paper broadens the scope of BCO to the regime with *time-varying constraints*, and proposes a class of online algorithms termed online bandit saddle-point (BanSP) approaches.

In a nutshell, relative to existing works, the main contributions of the present paper are summarized as follows.

c1) We formulate the fog computing tasks as a BCO problem, by generalizing the standard BCO framework with only time-varying costs [11], to account for both time-varying costs and constraints.

c2) We develop a novel BanSP algorithm to tackle this BCO problem, and analytically establish that the BanSP solver yields simultaneously sub-linear dynamic regret and fit, given that the variation of per-slot minimizers grows sub-linearly with time.

c3) Simulations demonstrate that the BanSP solvers have comparable performance relative to full-information alternatives.

2. FOG COMPUTING WITH BANDIT FEEDBACK

In this section, we introduce the fog computing setting, and formulate its computation offloading task as a bandit learning problem.

2.1. Fog Computing Setting

Consider a mobile network with a sensor layer, a fog layer, and a cloud layer [2, 13]. The sensor layer contains heterogeneous low-power IoT devices (e.g., wearable watches and smart cameras), which do not have enough computational capability, and usually offload their collected data to the local fog nodes (e.g., smartphones and high-tech routers) in the fog layer for further processing [14]. The fog layer consists of N nodes in the set $\mathcal{N} := \{1, \ldots, N\}$ with moderate processing capability; thus, part of workloads will be collaboratively processed by the local fog servers to meet the stringent latency requirement, and the rest will be offloaded to the remote data center in the cloud layer [4]; also see Fig. 1.

Per time t, each fog node n collects data requests b_t^n from all its nearby sensors. Once receiving these requests, node n has *three options*: i) offloading the amount z_t^n to the remote data center; ii) offloading the amount y_t^{nk} to each of its nearby

This work was supported by NSF 1509040, 1508993, and 1711471.

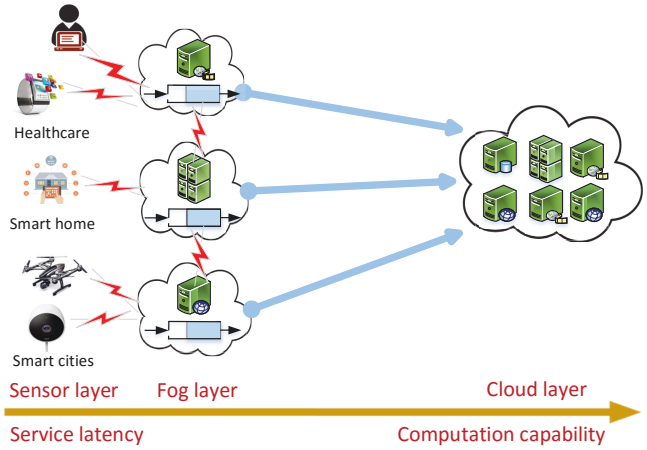

Fig. 1. A diagram of hierarchical fog computing framework.

node k for collaborative computing; and, iii) locally processing the amount y_t^{nn} according to its resource availability. The optimization variable $\mathbf{x}_t$ in this case consists of the cloud offloading, local offloading, and local processing amounts; i.e., $\mathbf{x}_t := [z_t^1, \ldots, z_t^N, y_t^{11}, \ldots, y_t^{1N}, \ldots, y_t^{N1}, \ldots, y_t^{NN}]^\top$. Assuming that each fog node has a data queue to buffer unserved workloads, the instantaneously served workloads (offloading plus processing) is not necessarily equal to the data arrival rate. Instead, a long-term constraint is common to ensure that the cumulative amount of served workloads is no less than the arrived amount over time

$$\sum_{t=1}^{T} g_t^n(\mathbf{x}_t) := \sum_{t=1}^{T} \left(b_t^n + \sum_{k\in\mathcal{N}_n^{\text{in}}} y_t^{kn} - \sum_{k\in\mathcal{N}_n^{\text{out}}} y_t^{nk} - z_t^n - y_t^{nn} \right) \leq 0 \tag{1}$$

where $\mathcal{N}_n^{\text{in}}$ and $\mathcal{N}_n^{\text{out}}$ represent the sets of fog nodes with in-coming links to node n and those with out-going links from node n, respectively. The bandwidth limit of communication link (e.g., wireline) from fog node n to the remote cloud is $\bar{z}^n$; the limit of the transmission link (e.g., wireless) from node n to its neighbor k is $\bar{y}^{nk}$, and the computation capability of node n is $\bar{y}^{nn}$. With $\bar{\mathbf{x}}$ collecting all the aforementioned limits, the feasible region can be expressed by $\mathbf{x}_t \in \mathcal{X} := \{\mathbf{0} \leq \mathbf{x}_t \leq \bar{\mathbf{x}}\}$.

Performance is assessed by the user dissatisfaction of the online processing and offloading decisions, e.g., aggregate delay [1, 13]. Specifically, as the computation delay is usually negligible for data centers with thousands of high-performance servers, the latency for cloud offloading amount z_t^n is mainly due to the communication delay, which is denoted as a time-varying cost $c_t^n(z_t^n)$ depending on the unpredictable network congestion during slot t. Likewise, the communication delay of the local offloading decision y_t^{nk} from node n to a nearby node k is denoted as $c_t^{nk}(y_t^{nk})$, but its magnitude is much lower than that of cloud offloading. Regarding the processing amount y_t^{nn}, its latency comes from the computation delay due to its limited computational capability, which is presented as a time-varying function $h_t^n(y_t^{nn})$ capturing the dynamic CPU capability during the computing processes. Per slot t, the network delay $f_t(\mathbf{x}_t)$ aggregates the computation delay at all nodes plus the communication delay at all links, namely

$$f_t(\mathbf{x}_t) := \sum_{n\in\mathcal{N}} \Big(\underbrace{c_t^n(z_t^n) + \textstyle\sum_{k\in\mathcal{N}_n^{\text{out}}} c_t^{nk}(y_t^{nk})}_{\text{communication}} + \underbrace{h_t^n(y_t^{nn})}_{\text{computation}} \Big). \tag{2}$$

Clearly, the explicit form of functions $c_t^n(\cdot)$, $c_t^{nk}(\cdot)$, and $h_t^n(\cdot)$ is *unknown* to the network operator due to the unpredictable traffic patterns [15]; but they are convex (thus $f_t(\mathbf{x}_t)$ is convex) with respect to their arguments, which implies that the marginal computation/communication latency is increasing as the offloading/processing amount grows.

Aiming to minimize the accumulated network delay while serving all the IoT workloads in the long term, the optimal offloading strategy in this mobile network is the solution of the following online optimization problem (cf. (2))

$$\min_{\{\mathbf{x}_t\in\mathcal{X},\forall t\}} \sum_{t=1}^{T} f_t(\mathbf{x}_t), \quad \text{s. to} \quad (1), \quad \text{for } n = 1, \ldots, N. \tag{3}$$

Comparing to the generic form (4), we consider an online fog computing problem in (3), where the loss (network latency) function $f_t(\cdot)$ and the data requests $\{b_t^n\}$ within slot t are not known when making the offloading and local processing decision $\mathbf{x}_t$; after performing $\mathbf{x}_t$, only the value of $f_t(\mathbf{x}_t)$ (a.k.a. loss) as well as the measurements $\{b_t^n\}$ are revealed to the network operator.

2.2. Optimal Computation Offloading via Bandit Learning

Targeting a light-weight online solution to the fog computing task (3) with only limited information, our idea is to leverage emerging bandit online learning tools to design algorithms with provable performance guarantees. Akin to its full-information counterpart [7], BCO can be viewed as a repeated game between a learner and nature. To abstract the task, we consider that per slot t, a learner selects an action $\mathbf{x}_t$ from a known and fixed convex set $\mathcal{X} \subseteq \mathbb{R}^d$, and then nature chooses not only a loss function $f_t(\cdot) : \mathbb{R}^d \to \mathbb{R}$, but also a time-varying penalty function $\mathbf{g}_t(\cdot) : \mathbb{R}^d \to \mathbb{R}^N$. The later gives rise to the time-varying constraint $\mathbf{g}_t(\mathbf{x}) \leq \mathbf{0}$, which is driven by the unknown application-specific dynamics. Similar to the standard BCO setting, only the value of $f_t(\mathbf{x}_t)$ at the queried point $\mathbf{x}_t$ is revealed to the learner here; but different from the standard BCO setting, besides $\mathcal{X}$, the constraint $\mathbf{g}_t(\mathbf{x}) \leq \mathbf{0}$ needs to be carefully taken care of. And the fact that $\mathbf{g}_t$ is unknown to the learner when performing her/his decision, makes it impossible to satisfy in every time slot. Hence, a more realistic goal here is to find a sequence of solutions $\{\mathbf{x}_t\}$ that minimizes the aggregate loss, and ensures that the constraints $\{\mathbf{g}_t(\mathbf{x}_t) \leq \mathbf{0}\}$ are satisfied in the long term on average. Specifically, extending the BCO framework [11, 16] to accommodate such time-varying constraints, we consider the following online problem

$$\boxed{\min_{\{\mathbf{x}_t\in\mathcal{X},\forall t\}} \sum_{t=1}^{T} f_t(\mathbf{x}_t) \quad \text{s. to} \quad \sum_{t=1}^{T} \mathbf{g}_t(\mathbf{x}_t) \leq \mathbf{0}.} \tag{4}$$

In the current setting, we assume that only the values of loss function are available at queried points since e.g., its complete form related to user experience is hard to approximate, but the constraint function is revealed to the learner as it represents measurable physical requirements, e.g., data flow conservation constraints in (1).

3. ONLINE BANDIT SADDLE-POINT METHODS

To solve the problem in Section 2, a bandit online saddle-point method will be developed, along with its performance guarantees.

3.1. Fog Computing with Partial Feedback

The key idea behind BCO is to construct (possibly stochastic) gradient estimates using the limited *function value* information [12, 17].

To begin with, we consider the case where the learner can only observe the function value of $f_t(\mathbf{x})$ at two points per slot t. The task here remains to construct a (possibly unbiased) estimate of the gradient using two feedbacks. The intuition can be revealed from the one-dimensional case ($d = 1$): For a small constant $\delta > 0$, the idea of forward differentiation implies that the derivative f_t' at x can be approximated by

$$f_t'(x) \approx \frac{f_t(x+\delta) - f_t(x-\delta)}{2\delta} \tag{5}$$

where the approximation is due to $\delta > 0$. Generalizing this approximation to high dimensions, with a random vector $\mathbf{u}$ drawn from the unit sphere (a.k.a. the surface of a unit ball), the scaled function evaluations at perturbed points $\mathbf{x}+\delta\mathbf{u}$ and $\mathbf{x}-\delta\mathbf{u}$ yield an estimate of the gradient $\nabla f_t(\mathbf{x})$, given by [11]

$$\nabla f_t(\mathbf{x}) \approx \mathbb{E}_{\mathbf{u}}\left[\frac{d\mathbf{u}}{2\delta}\Big(f_t(\mathbf{x}+\delta\mathbf{u}) - f_t(\mathbf{x}-\delta\mathbf{u})\Big)\right]. \tag{6}$$

Hence, $\frac{d\mathbf{u}}{2\delta}\big(f_t(\mathbf{x}+\delta\mathbf{u}) - f_t(\mathbf{x}-\delta\mathbf{u})\big)$ constructed by two function evaluations can serve as a stochastic estimator of $\nabla f_t(\mathbf{x})$. Compared with (5), the random vector $\mathbf{u}$ introduced in (6) intuitively allows the gradient estimator to evaluate the directional derivative along every possible direction on average.

Building upon this intuition, consider a bandit version of the online saddle-point iteration, for which the primal update becomes

$$\hat{\mathbf{x}}_{t+1} = \mathcal{P}_{(1-\gamma)\mathcal{X}}\Big(\hat{\mathbf{x}}_t - \alpha\hat{\nabla}^2_{\mathbf{x}}\mathcal{L}_t(\hat{\mathbf{x}}_t, \boldsymbol{\lambda}_t)\Big) \tag{7}$$

where $(1-\gamma)\mathcal{X} := \{(1-\gamma)\mathbf{x} : \mathbf{x} \in \mathcal{X}\}$ is a subset of $\mathcal{X}$, $\gamma \in [0,1)$ is a pre-selected constant depending on δ, and the two-point Langragian gradient is given by

$$\hat{\nabla}^2_{\mathbf{x}}\mathcal{L}_t(\hat{\mathbf{x}}_t, \boldsymbol{\lambda}_t) := \hat{\nabla}^2 f_t(\hat{\mathbf{x}}_t) + \nabla\mathbf{g}_t(\hat{\mathbf{x}}_t)^\top\boldsymbol{\lambda}_t. \tag{8}$$

At slot t, the stochastic gradient is $\hat{\nabla}^2 f_t(\hat{\mathbf{x}}_t) := \frac{d\mathbf{u}_t}{2\delta}\big(f_t(\hat{\mathbf{x}}_t+\delta\mathbf{u}_t) - f_t(\hat{\mathbf{x}}_t-\delta\mathbf{u}_t)\big)$, where the function values are evaluated on two points around the learning iterate $\hat{\mathbf{x}}_t$, namely, $\mathbf{x}_{1,t} := \hat{\mathbf{x}}_t + \delta\mathbf{u}_t$ and $\mathbf{x}_{2,t} := \hat{\mathbf{x}}_t - \delta\mathbf{u}_t$ with $\mathbf{u}_t$ again drawn uniformly from the unit sphere $\mathbb{S} := \{\mathbf{u} \in \mathbb{R}^d : \|\mathbf{u}\| = 1\}$. Furthermore, the projection is performed on a smaller convex set $(1-\gamma)\mathcal{X}$ in (7), which ensures feasibility of the perturbed $\mathbf{x}_{1,t}, \mathbf{x}_{2,t} \in \mathcal{X}$. The dual update of BanSP is given by

$$\boldsymbol{\lambda}_{t+1} = \Big[\boldsymbol{\lambda}_t + \mu(\mathbf{g}_t(\hat{\mathbf{x}}_t) + \nabla^\top\mathbf{g}_t(\hat{\mathbf{x}}_t)(\hat{\mathbf{x}}_{t+1} - \hat{\mathbf{x}}_t))\Big]^+ \tag{9}$$

where μ is again the stepsize, and the learning iterate $\hat{\mathbf{x}}_t$ rather than the actual decision $\mathbf{x}_t$ is used in this update. Here (9) is updated along the first-order approximation of $\mathbf{g}_t(\hat{\mathbf{x}}_{t+1})$ at $\hat{\mathbf{x}}_t$ [10, 18].

With the insights gained so far, the next step is to endow the BanSP with more than two function evaluations [11]. With $M > 2$ points, the gradient estimator is obtained by querying the function values over M points near $\hat{\mathbf{x}}_t$. These points include $\mathbf{x}_{m,t} := \hat{\mathbf{x}}_t + \delta\mathbf{u}_{m,t}$, $1 \le m \le M-1$, and the learning iterate $\mathbf{x}_{m,t} := \hat{\mathbf{x}}_t$, where $\mathbf{u}_{m,t}$ is drawn from $\mathbb{S}$. Specifically, the gradient becomes (cf. (7))

$$\hat{\nabla}^M_{\mathbf{x}}\mathcal{L}_t(\hat{\mathbf{x}}_t, \boldsymbol{\lambda}_t) := \tag{10}$$
$$\frac{d}{\delta(M-1)}\sum_{m=1}^{M-1}\big(f_t(\hat{\mathbf{x}}_t+\delta\mathbf{u}_{m,t}) - f_t(\hat{\mathbf{x}}_t)\big)\mathbf{u}_{m,t} + \nabla\mathbf{g}_t(\hat{\mathbf{x}}_t)^\top\boldsymbol{\lambda}_t$$

where we define the M-point stochastic gradient as $\hat{\nabla}^M f_t(\hat{\mathbf{x}}_t) := \frac{d}{\delta(M-1)}\sum_{m=1}^{M-1}\big(f_t(\hat{\mathbf{x}}_t+\delta\mathbf{u}_{m,t}) - f_t(\hat{\mathbf{x}}_t)\big)\mathbf{u}_{m,t}$. At the price of extra computations, simulations will validate that the BanSP with multi-point feedback enjoys improved performance. The BanSP approach is summarized in Algorithm 1.

Algorithm 1 BanSP for low-latency fog computing

1: **Initialize:** primal iterate $\hat{\mathbf{x}}_1$, dual iterate $\boldsymbol{\lambda}_1$, parameters δ and γ, and proper stepsizes α and μ.
2: **for** $t = 1, 2\ldots$ **do**
3: The learner plays the perturbed actions $\{\mathbf{x}_{m,t}\}_{m=1}^M$ based on the learning iterate $\hat{\mathbf{x}}_t$.
4: The nature reveals the losses $\{f_t(\mathbf{x}_{m,t})\}_{m=1}^M$ at queried points, and the constraint function $\mathbf{g}_t(\mathbf{x})$.
5: The learner updates the primal variable $\hat{\mathbf{x}}_{t+1}$ by (7) with the gradient estimated by (8) for $M = 2$, or, (10) for $M > 2$.
6: The learner updates the dual variable $\boldsymbol{\lambda}_{t+1}$ via (9).
7: **end for**

3.2. Optimality and Feasibility of BanSP

In response to the quest for improved benchmarks in this dynamic setup with constraints, two metrics are considered here: *dynamic regret* and *dynamic fit*. The notion of dynamic regret has been recently adopted in [19, 20] to assess performance of online algorithms under time-invariant constraints. For our BCO setting of (4), we adopt

$$\mathrm{Reg}^{\mathrm{d}}_T := \frac{1}{M}\sum_{t=1}^T\sum_{m=1}^M \mathbb{E}\left[f_t(\mathbf{x}_{m,t})\right] - \sum_{t=1}^T f_t(\mathbf{x}_t^*) \tag{11}$$

where the actual loss per slot is averaged over the losses of M actions (queried points), $\mathbb{E}$ is taken over the sequence of random actions (due to $\delta\mathbf{u}$ perturbations), and the benchmark is now formed via a sequence of best dynamic solutions $\{\mathbf{x}_t^*\}$ for the instantaneous cost minimization problem subject to the instantaneous constraint, namely, $\mathbf{x}_t^* \in \arg\min_{\mathbf{x}\in\mathcal{X}}\ f_t(\mathbf{x})\ \ \text{s. to}\ \ \mathbf{g}_t(\mathbf{x}) \le \mathbf{0}$.

Regarding feasibility of decisions generated by a BCO algorithm, the notion of *dynamic fit* will be used to measure the accumulated violation of constraints [21], that is

$$\mathrm{Fit}^{\mathrm{d}}_T := \left\|\left[\frac{1}{M}\sum_{t=1}^T\sum_{m=1}^M \mathbf{g}_t(\mathbf{x}_{m,t})\right]^+\right\|. \tag{12}$$

Note that the dynamic fit is zero if the accumulated violation $\frac{1}{M}\sum_{t=1}^T\sum_{m=1}^M \mathbf{g}_t(\mathbf{x}_{m,t})$ is entry-wise less than zero. The long-term constraint implicitly assumes that the instantaneous constraint violations can be compensated by the later strictly feasible decisions, and thus allows adaptation of online decisions to the unknown dynamics. Under this broader BCO setup, an ideal online algorithm is the one that achieves both sub-linear dynamic regret and fit. A sub-linear dynamic regret implies "no-regret" relative to the dynamic solution on the long-term average; i.e., $\lim_{T\to\infty}\mathrm{Reg}^{\mathrm{d}}_T/T = 0$; and a sub-linear dynamic fit indicates that the online strategy is also feasible on average; i.e., $\lim_{T\to\infty}\mathrm{Fit}^{\mathrm{d}}_T/T = 0$.

(as1) *Functions $f_t(\mathbf{x})$ and $\mathbf{g}_t(\mathbf{x})$ are convex, $f_t(\mathbf{x})$ is bounded by F over $\mathcal{X}$, and $f_t(\mathbf{x})$ and $g_t^n(\mathbf{x})$ have bounded gradients; that is, $\|\nabla f_t(\mathbf{x})\| \le G$, and $\max_n\|\nabla g_t^n(\mathbf{x})\| \le G$.*

(as2) *For a small constant γ, there exists a constant $\eta > 0$, and an interior point $\tilde{\mathbf{x}} \in (1-\gamma)\mathcal{X}$ such that $\mathbf{g}_t(\tilde{\mathbf{x}}) \le -\eta\mathbf{1}$, $\forall t$.*

(as3) *With $\mathbb{B} := \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| \le 1\}$ denoting the unit ball, there exist constants $0 < r \le R$ such that $r\mathbb{B} \subseteq \mathcal{X} \subseteq R\mathbb{B}$.*

Assumption (as1) is typical in OCO with both partial-information feedback [21–23]; (as2) is Slater's condition modified for our BCO setting, which guarantees the existence of a Lagrange multiplier [24]; and, (as3) requires the action set to be bounded within a ball that contains the origin. When (as3) appears to be restrictive, it is tantamount to assuming $\mathcal{X}$ is compact and has a nonempty interior, because one can always apply an affine transformation (a.k.a. reshaping) on $\mathcal{X}$ to satisfy (as3); see [23, Section 3.2].

Under these assumptions, we are on track to provide upper bounds for the dynamic regret and fit of the BanSP solver. Proofs of results in this paper are available in the online version [18].

Theorem 1 *Suppose (as1)-(as3) are satisfied, and consider α, μ, δ, γ defined in* (7)-(9)*, and constants F, G, r, R defined in (as1)-(as3). Then BanSP with two-point feedback has dynamic regret bounded by*

$$\mathrm{Reg}_T^{\mathrm{d}} \leq \frac{R}{\alpha} V(\mathbf{x}_{1:T}^*) + \frac{R^2}{2\alpha} + 2\mu G^2 R^2 T + \alpha d^2 G^2 T + \gamma G R T (1 + \|\bar{\boldsymbol{\lambda}}\|) + 2\delta G T \quad (13)$$

where $\|\bar{\boldsymbol{\lambda}}\| := \max_t \|\boldsymbol{\lambda}_t\|$, and the accumulated variation of the per-slot minimizers $\mathbf{x}_t^*$ in (11) *is given by $V(\mathbf{x}_{1:T}^*) := \sum_{t=1}^T \|\mathbf{x}_t^* - \mathbf{x}_{t-1}^*\|$. In addition, the dynamic fit in* (12) *is bounded by*

$$\mathrm{Fit}_T^{\mathrm{d}} \leq \frac{\|\bar{\boldsymbol{\lambda}}\|}{\mu} + \frac{G^2\sqrt{N}T}{2\beta} + \delta G\sqrt{N}T + \beta\sqrt{N}T\left(\alpha^2 d^2 G^2 + \alpha^2 G^2 \|\bar{\boldsymbol{\lambda}}\|^2\right). \quad (14)$$

In this case, if we choose the stepsizes as $\alpha = \mu = \mathcal{O}(T^{-\frac{1}{2}})$, and set the parameters as $\beta = T^{\frac{1}{2}}$, $\delta = \mathcal{O}(T^{-1})$, and $\gamma = \delta/r$, then the BanSP solvers yield dynamic regret and fit bounded by

$$\boxed{\mathrm{Reg}_T^{\mathrm{d}} = \mathcal{O}\left(V(\mathbf{x}_{1:T}^*) T^{\frac{1}{2}}\right) \text{ and } \mathrm{Fit}_T^{\mathrm{d}} = \mathcal{O}\left(T^{\frac{1}{2}}\right)} \quad (15)$$

where $V(\mathbf{x}_{1:T}^*)$ is the variation of the per-slot minimizers $\mathbf{x}_t^*$.

Theorem 1 establishes that the dynamic regret and fit are sub-linear if $V(\mathbf{x}_{1:T}^*) = \mathbf{o}(T^{\frac{1}{2}})$. As a special case of Theorem 1, by confining $\mathbf{x}_1^* = \cdots = \mathbf{x}_T^*$ so that $V(\mathbf{x}_{1:T}^*) = 0$, the dynamic regret (15) reduces to the static ones, which correspond to $\mathcal{O}(\sqrt{T})$ in the two-point case. This pair of bounds markedly improves the *regret versus fit tradeoff* in [21], and matches the order of regret in [11,12], which are the best possible ones that can be achieved by *efficient* algorithms even in the BCO setup without the long-term constraints. For BanSP with $M > 2$, improved bounds can be proved without changing the order of regret and fit, but they are omitted for brevity.

4. NUMERICAL EVALUATION

Consider the fog computing task with $N = 10$ nodes and a cloud center. Each fog node has an outgoing link to the cloud, and two outgoing links to two nearby fog nodes for local collaborative computing. For a communication link offloading loads from node n to k, the offloading limit is $\bar{y}^{nk} = 10$, the local computation limit at node n is $\bar{y}^{nn} = 50$, and the fog-cloud offloading limits $\{\bar{z}^n\}$ are all set to 100. The online cost (a.k.a. service latency) in (2) is specified by

$$f_t(\mathbf{x}_t) := \sum_{n\in\mathcal{N}} \left(e^{p_t^n z_t^n} + \sum_{k\in\mathcal{N}_n^{\mathrm{out}}} l^{nk} y_t^{nk} + l^{nn} (y_t^{nn})^2 \right) \quad (16)$$

where $p_t^n = 0.015\sin(\pi t/96) + 0.05$, $n \in \mathcal{N}\backslash\{4,5\}$, $p_t^n = 0.045\sin(\pi t/96) + 0.15$, $n \in \{4,5\}$, and the local coefficients are set to $l^{nk} = 8/\bar{y}^{nk}$ and $l^{nn} = 8/\bar{y}^{nn}$. Regarding the data arrival rate b_t^n, it is generated according to $b_t^n = q^n \sin(\pi t/96) + \nu_t^n$, with q^n and ν_t^n uniformly distributed over $[40, 50]$ and $[45, 55]$ for $n \in \mathcal{N}\backslash\{1,2,3\}\bigcup\{4,5\}$, and $q^n \in [32, 40], \nu_t^n \in [36, 44]$, $n \in \{1,2,3\}$, and $q^n \in [20, 25], \nu_t^n \in [22.5, 27.5]$, $n \in \{4,5\}$. Notice that the scales of p_t^n and b_t^n vary between nodes, mimicking heterogeneity of real IoT systems; and the periods of p_t^n and b_t^n correspond to a 24-hour interval with slot duration 7.5 minutes. When the parameters of BanSP need to be adjusted in each test, they are set to $\gamma = 0.05$, and $\delta = 0.05$ for $M \geq 2$. Finally, BanSP is benchmarked by: i) the *full-information* MOSP method in [8] that takes gradient-based update; ii) the *heuristic* cloud-only approach that offloads all data requests to the cloud; and, iii) the *heuristic* fog-only approach that processes all requests locally without collaboration.

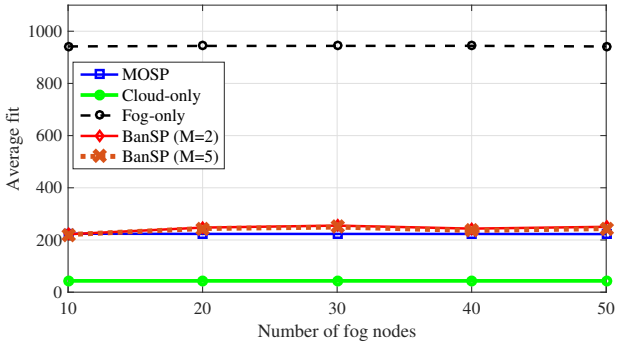

Fig. 2. Impact of network size on dynamic fit per fog node.

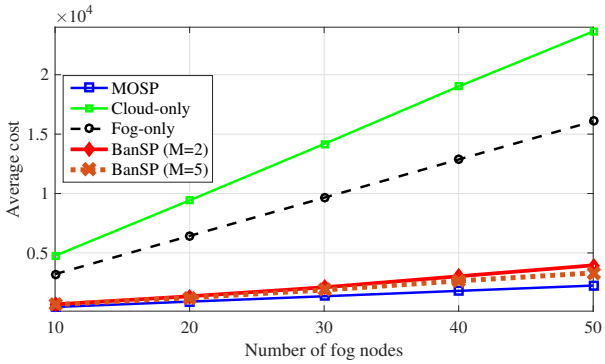

Fig. 3. Impact of network size on average network cost.

The performance of all schemes is evaluated under different number of fog nodes (i.e., network size). For each algorithm, the fit averaged over all fog nodes and time is presented in Fig. 2, and the cost (i.e., network delay) averaged over the time is shown in Fig. 3. Clearly, without collaborative computing between fog and cloud, the cloud- and fog-only schemes have much higher network delays. The average delay and fit of BanSP variants with multiple function evaluations are comparable to those of the full-information MOSP as the network size grows. An interesting observation here is that as the number of fog nodes increases, the performance gain of the BanSP solver with a large M becomes more evident; see e.g., Fig. 3. This implies that for a larger network, BanSP benefits from more bandit information to learn and track the network dynamics.

5. CONCLUSIONS

In this paper, fog computation offloading under partial information was formulated as an online bandit learning task with both adversarial costs and constraints. Different from existing work in bandit settings, the focus was on a broader setting where part of the constraints are revealed after taking actions, and are also tolerable to instantaneous violations but have to be satisfied on average. An online bandit saddle-point (BanSP) approach was developed, and its online performance was rigorously analyzed. It was shown that the BanSP solver can simultaneously yield sub-linear dynamic regret and fit, if the dynamic fog offloading solutions vary slowly over time.

6. REFERENCES

[1] Farzad Samie, Vasileios Tsoutsouras, Sotirios Xydis, Lars Bauer, Dimitrios Soudris, and Jörg Henkel, "Distributed QoS management for Internet of Things under resource constraints," in *Proc. Intl. Conf. on Hardware/Software Codesign and System Synthesis*, Pittsburgh, PA, Oct. 2016, pp. 1–10.

[2] Mung Chiang and Tao Zhang, "Fog and IoT: An overview of research opportunities," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 854–864, 2016.

[3] Farzad Samie, Vasileios Tsoutsouras, Lars Bauer, Sotirios Xydis, Dimitrios Soudris, and Jorg Henkel, "Computation offloading and resource allocation for low-power IoT edge devices," in *Proc. World Forum Internet Things*, Dec. 2016, pp. 7–12.

[4] Pavel Mach and Zdenek Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Comm. Surveys & Tutorials*, 2017, to appear.

[5] Feng Wang, Jie Xu, Xin Wang, and Shuguang Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, Feb. 2017, submitted.

[6] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B Letaief, "Mobile edge computing: Survey and research outlook," *arXiv preprint:1701.01090*, Jan. 2017.

[7] Martin Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proc. Intl. Conf. on Machine Learning*, Washington D.C., Aug. 2003.

[8] T. Chen, Q. Ling, and G. B. Giannakis, "An online convex optimization approach to proactive network resource allocation," *IEEE Trans. Signal Processing*, Jan. 2017 (revised), Available: https://arxiv.org/abs/1701.03974.

[9] T. Chen, Y. Shen, Q. Ling, and G. B. Giannakis, "Online learning for "thing-adaptive" fog computing in IoT," in *Proc. of Asilomar Conf.*, Pacific Grove, CA, Oct. 2017.

[10] Michael J Neely and Hao Yu, "Online convex optimization with time-varying constraints," *arXiv preprint:1702.04783*, Feb. 2017.

[11] Alekh Agarwal, Ofer Dekel, and Lin Xiao, "Optimal algorithms for online convex optimization with multi-point bandit feedback.," in *Proc. Annual Conf. on Learning Theory*, Haifa, Israel, 2010, pp. 28–40.

[12] John C Duchi, Michael I Jordan, Martin J Wainwright, and Andre Wibisono, "Optimal rates for zero-order convex optimization: The power of two function evaluations," *IEEE Trans. Inform. Theory*, vol. 61, no. 5, pp. 2788–2806, May 2015.

[13] Gilsoo Lee, Walid Saad, and Mehdi Bennis, "An online secretary framework for fog network formation with minimal latency," *arXiv:1702.05569*, Apr. 2017.

[14] Houfeng Huang, Qing Ling, Wei Shi, and Jinlin Wang, "Collaborative resource allocation over a hybrid cloud center and edge server network," *Journal of Computational Mathematics*, 2017, to appear.

[15] Baruch Awerbuch and Robert D Kleinberg, "Adaptive routing with end-to-end feedback: Distributed learning and geometric approaches," in *Proc. ACM Symp. on Theory of Computing*, Chicago, IL, June 2004, pp. 45–53.

[16] Ohad Shamir, "An optimal algorithm for bandit and zero-order convex optimization with two-point feedback," *Journal of Machine Learning Research*, vol. 18, no. 52, pp. 1–11, 2017.

[17] Yurii Nesterov and Vladimir Spokoiny, "Random gradient-free minimization of convex functions," *Foundations of Computational Mathematics*, vol. 17, no. 2, pp. 527–566, Apr. 2017.

[18] T. Chen and G. B. Giannakis, "An online convex optimization approach to proactive network resource allocation," *IEEE J. Internet-of-Things*, July 2017 (submitted), Available: https://arxiv.org/pdf/1707.09060.

[19] Ali Jadbabaie, Alexander Rakhlin, Shahin Shahrampour, and Karthik Sridharan, "Online optimization: Competing with dynamic comparators," in *Intl. Conf. on Artificial Intelligence and Statistics*, San Diego, CA, May 2015.

[20] Eric C Hall and Rebecca M Willett, "Online convex optimization in dynamic environments," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 4, pp. 647–662, June 2015.

[21] Mehrdad Mahdavi, Rong Jin, and Tianbao Yang, "Trading regret for efficiency: Online convex optimization with long term constraints," *Journal of Machine Learning Research*, vol. 13, pp. 2503–2528, Sep 2012.

[22] Elad Hazan, Amit Agarwal, and Satyen Kale, "Logarithmic regret algorithms for online convex optimization," *Machine Learning*, vol. 69, no. 2-3, pp. 169–192, Dec. 2007.

[23] Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan, "Online convex optimization in the bandit setting: gradient descent without a gradient," in *Proc. of ACM SODA*, Vancouver, Canada, Jan. 2005, pp. 385–394.

[24] Dimitri P Bertsekas, *Nonlinear Programming*, Athena scientific, Belmont, MA, 1999.