Information Sciences 577 (2021) 140–161

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Dynamic online convex optimization with long-term constraints via virtual queue

Xiaofeng Ding [a], Lin Chen [a], Pan Zhou [b,*], Zichuan Xu [c], Shiping Wen [d], John C.S. Lui [e], Hai Jin [a]

[a] National Engineering Research Center for Big Data Technology and System Lab, Services Computing Technology and System Lab, Cluster and Grid Computing Lab, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China
[b] Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China
[c] Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116621, China
[d] Australian AI Institute, Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW 2007, Australia
[e] Department of Computer Science and Engineering, The Chinese University of HongKong, Shatin, N.T., Hong Kong

ARTICLE INFO

Article history:
Received 22 January 2021
Received in revised form 4 June 2021
Accepted 22 June 2021
Available online 24 June 2021

Keywords:
Constrained online convex optimization
Dynamic regret
Virtual queues
Resource allocation
Cloud computing

ABSTRACT

In this paper, we investigate the online convex optimization (OCO) with long-term constraints which is widely used in various resource allocations and recommendation systems. Different from the most existing works, our work adopts a dynamic benchmark to analyze the optimization performance since the dynamic benchmark is more common than the static benchmark in practical applications. Moreover, compared with many constrained OCO works ignoring the Slater condition, we study the effect of the Slater condition on the constraint violation bounds and obtain the better performance of the constraint violations when the Slater condition holds. More importantly, we propose a novel iterative optimization algorithm based on the virtual queues to achieve sublinear regret and constraint violations. Finally, we apply our dynamic OCO model to a resource allocation problem in cloud computing and the results of the experiments validate the effectiveness of our algorithm.

1. Introduction

Online convex optimization (OCO) is a kind of common framework to model the various real-time problems in the practical applications including network resource allocation [1–3] and spam filtering [4,5] and therefore it increasingly becomes a popular topic in the optimization community. Unlike the other methods based on the time-variant optimization [6–8], OCO can be understood as a multi-round learning procedure between a learner and the unknown environment. At every round, the learner comes to a decision $\boldsymbol{a}_t$ from the convex set $\mathcal{A} \subset \mathbb{R}^d$ and until the end of this round, the environment will divulge a time-varying convex loss function $l_t(\cdot) : \mathbb{R}^d \to \mathbb{R}$ to the learner who sustains a loss $l_t(\boldsymbol{a}_t)$. The learner aims to keep the total loss at a minimum over the time horizon T. To tackle the minimization problem, the widely adopted performance metric is the *regret* defined as $\text{Regret}_T = \sum_{t=1}^{T} l_t(\boldsymbol{a}_t) - \sum_{t=1}^{T} l_t(\boldsymbol{a}^*)$ where $\boldsymbol{a}^*$ is the optimal solution to the cumulative loss $\sum_{t=1}^{T} l_t(\boldsymbol{a}_t)$. Obviously, when the regret is sublinear then the performance of the sequence $\{\boldsymbol{a}_t\}$ is not inferior to that of the optimal sequence $\{\boldsymbol{a}^*\}$ as the round tends to infinity.

* Corresponding author.
E-mail addresses: xfding@hust.edu.cn (X. Ding), chenl@hust.edu.cn (L. Chen), panzhou@hust.edu.cn (P. Zhou), z.xu@dlut.edu.cn (Z. Xu), shiping.wen@uts.edu.cn (S. Wen), cslui@cse.cuhk.edu.hk (J.C.S. Lui), hjin@hust.edu.cn (H. Jin).

https://doi.org/10.1016/j.ins.2021.06.072
0020-0255/

Many early works choose the static sequence $\{\boldsymbol{a}^*\}$ as the optimal benchmark to solve $\min_{\boldsymbol{a}\in\mathcal{A}}\sum_{t=1}^{T} l_t(\boldsymbol{a})$. Although the static regret is suitable for many real scenarios, such as training the SVM (support vector machine) for security vulnerability detection [9] and estimating the static vector by sensor network [10–12], it cannot be used to solve the time-varying and dynamic problems like keeping track of the setpoint in online demand response [13–15] and locating the objects in motion [16,17]. Besides, as a typical dynamic scenario, the network nodes in cloud computing usually move dynamically and therefore they could affect the distribution of the link loss functions such as link delays. It means that the route selections of the optimal communication links will become dynamic eventually. As to these problems, it is required to utilize the dynamic regret for analyzing the optimization performance where the dynamic sequence $\{\boldsymbol{a}_t^*\}_{t=1}^{T}$ is chosen as the optimal sequence to minimize $\sum_{t=1}^{T} l_t(\boldsymbol{a}_t)$.

The above OCO model belongs to the unconstrained OCO frame, i.e., there is no constraint on decision $\boldsymbol{a}_t$, but numerous realistic OCO situations include some constraints $\{\boldsymbol{h}(\boldsymbol{a}_t) : \mathbb{R}^d \to \mathbb{R}^m | \boldsymbol{h}(\boldsymbol{a}_t) \leqslant \boldsymbol{0}_m\}$ which are resulted from the undetermined dynamic demands in diverse application scenes such as the access requirements from the users in data centers [18]. Moreover, we take the aforementioned cloud computing problem as an example again. In cloud computing, for a link optimization problem, it is required for the decision-maker to simultaneously focus on the optimal link selection and the constraints on the cumulative power consumption. In general, the optimal communication link keeps changing and the power consumption is usually presumed as the i.i.d. function. More importantly, for this kind of power consumption produced from different links, it tends to satisfy the physical power limits of mobile devices that conform to some power allocation policies. The existing constrained OCO literature mainly focuses on the OCO problems with time-varying constraint $\boldsymbol{h}_t(\boldsymbol{a}_t)$ but a small number of works discusses the effect of the time-invariant constraint $\boldsymbol{h}(\boldsymbol{a}_t)$. For plenty of OCO problems with time-varying constraint, if the constraint is satisfied at each round t, a projection step by the round is required to make every single $\boldsymbol{a}_t$ more feasible. As to general convex set $\mathcal{A}$, it is probably computationally expensive for the projection step to resolve an auxiliary optimization problem. What's more, in real-world settings, the learner may only care about meeting long-term constraints. In other words, the sequence $\{\boldsymbol{a}_t\}_{t=1}^{T}$ should give rise to the bounded accumulative constraint violations after T rounds. Therefore, the aim of OCO problems with long-term constraints is to minimize $\sum_{t=1}^{T} l_t(\boldsymbol{a}_t)$ and satisfy $\sum_{t=1}^{T} \boldsymbol{h}_t(\boldsymbol{a}_t) \leqslant 0$ at the same time.

To further illustrate the broad potential application value of our dynamic OCO model subject to the long-term constraints, we consider an optimal resource allocation for mobile edge computing. Because the computation traffic is random, in terms of users, the rates of task arrival could have a drastic fluctuation in time and space. Besides, considering that the wireless channel is equipped with randomness, it is also possible for the wireless energy acquired from the access point to vary notably with the time between different users. Therefore, it is required to study the dynamic computation requirements to satisfy the computation and communication resource allocation at the users. More importantly, the operators may want to maximize the vendor revenue by deploying finite computing resources such as memory, CPU, or even bandwidth so they need to guarantee that the dynamic resource violations will not exceed a pre-defined threshold. To address this challenge, we can cast the problem of resource scheduling in the setting of dynamic OCO subject to the long-term constraints.

For the dynamic OCO problems with long-term constraints, we aim to minimize the cumulative loss while satisfying the constraints in a long-term manner, i.e., the long-term constraint is not violated too much. It means that we need to guarantee that the time-averaged constraint violation approaches zero when the time slot goes to infinity, which can be understood as a sublinear cumulative constraint violation. Lyapunov optimization [19] is a widely used method in network systems whose advantage lies in using the system state and network queueing information to implicitly learn and adapt to changes in the system even not knowing the statistical information of the system. The goal of this method is to satisfy the queue stability constraints while optimizing some performance objectives like minimization of the average energy. Comparing our optimization goal with Lyapunov optimization, it is straightforward to introduce the "*virtual queues*" to successively record the cumulative constraint violations in our proposed OCO model and then the sublinear constraint violation can be connected to the stability of the virtual queues. More importantly, we can also view the virtual queues as the penalizations to the violations of the constraints and by applying the key drift-plus-penalty (DPP) technology [19] of Lyapunov optimization, we can transform our constrained OCO problem into the DPP minimization problem. According to the above illustration, in this paper, we combine the virtual queue technique with the dynamic OCO subject to long-term constraints to achieve the desired optimization performance, i.e., sublinear regret and constraint violation.

Compared with the conventional constrained optimization algorithms like the dual gradient method, the virtual queue based approach owns a faster convergence rate. Besides, different from the common saddle point method whose step size parameter is decided by the time frame T, the step size selection for the virtual queue based method is unrelated to T, which guarantees that for any time including the unknown time point, the online optimization task can terminate. Furthermore, it means that the performance criterion of the virtual queue based method can be satisfied at the arbitrary time point even though the online optimization procedure is still running. More importantly, the other methods are just from the perspective of optimization theory but our method is more meaningful in real scenario, especially in various network systems. To better exhibit the practical application value and explain how to apply our ideas, we present a few representative examples of virtual-queue-based dynamic OCO with long-term constraints.

1.1. Motivating examples

Livecast Services with Cloud-Edge-Crowd Integration. Consider the crowdsourced live cast services (CLS) which leverage the cooperation of cloud, edge and crowd technologies. Since resource provision and network conditions are highly nonstationary and unpredictable, the resource allocation of transmission and online transcoding is dynamic. Moreover, due to the limited transcoding and link bandwidth resource, the transcoding and transmission rate are subject to the node and link capacity in the long-time term. Then, we can build a queue model by converting transcoding task allocation to the virtual queue control problem and the longer the task queue implies the more transcoding workloads. Similarly, we can use another kind of virtual queue to record the transmission link in the edge servers. Therefore, for this kind of CLS system, we can use our dynamic constrained OCO model based on the virtual queues to jointly minimizes the usage of transmission resources like bandwidth and transcoding resources like CPU.

Energy Management for Multiuser Mobile-Edge Computing Systems. Take the power consumption in a multiuser MEC system with energy harvesting devices as an example. Considering that the data arrivals are not static and the energy harvesting is intermittent, the data transmission power consumption in mobile users is dynamic. Besides, when the computation data produced by the mobile user transmits to the MEC server, the QoS constraints require that the proportion of unprocessed data in the total arrival data is less than a prescribed threshold and apparently, these constraints should be satisfied in a long time. For this dynamic constrained problem, we can utilize a virtual queue to track the uncompleted data request of the current time slot and execute this request in the next slot. In this way, we can satisfy the QoS constraint by stabilizing the virtual queue. Finally, we can utilize the drift-plus-penalty of virtual queue technique to solve the dynamic power consumption minimization while satisfying the long-term QoS constraints.

Profit maximization problem of the wireless network operator. Consider a network operator that provides wireless communications services to its own secondary users by acquiring spectrum resources from some frequency spectrum owners and it aims to maximize revenue with pricing and market control and minimize the cost with proper resource investment and allocation. Considering that the downlink communication system has various network characteristics including random user demand, uncertain sensing spectrum resources and fluctuating spectrum prices, the optimization problem is dynamic. In addition, when the secondary users detect the currently available spectrum resource, missed detections will lead to transmission collisions with the primary users. To protect primary users transmissions, the operator needs to ensure that the average collision in each channel does not exceed a tolerable level specified by the spectrum owner, i.e., collision constraint should be satisfied in long term. For this kind of collision constraint, we can use virtual queues to track the number of collisions happening in the sensing channel and if the virtual queue is stable, it implies that the average incoming rate is no larger than the average serving rate. This exactly avoids the collision. Therefore, according to our proposed virtual-queue-based method, we can achieve the optimization goal while satisfying the long-term collision constraints.

1.2. Contributions

In this work, we exploit the virtual queue based approach to tackle the OCO problem with long-term constraints. Compared with recent researches, our work differs in the following two aspects: (a) We bring forward a dynamic OCO algorithm utilizing the virtual queues which greatly improves the optimization performance. It should be noted that we choose the dynamic benchmark sequence to describe the regret since static benchmark is not appropriate to study the non-stationary system whose potential optimum typically changes. Moreover, our results are functions of the trade-off parameter δ which can guarantee that the regret and constraint violations have the adjustable upper bound. (b) We provide a formal analysis of the Slater condition. While considering OCO with constraints, many papers (e.g.,[22,29,23]) drew their conclusion by ignoring the Slater condition. However, it is necessary to consider the Slater condition because it is a common assumption in constrained convex optimization problem and it has an impact on the analysis of convergence time and performance [36,45]. Specifically, the slater condition requires that there exists at least one interior point in the feasible region to make all inequality constraints are strictly satisfied, i.e., for the constraint function $\boldsymbol{h}(\boldsymbol{a}_t)$, it is required to exist at least one point satisfying $\boldsymbol{h}(\boldsymbol{a}_t) < \boldsymbol{0}_m$ rather than $\boldsymbol{h}(\boldsymbol{a}_t) \leqslant \boldsymbol{0}_m$. Since the Slater condition can ensure that the strictly feasible solution always exists, it plays a significant role in many practical problems related to the constrained convex optimization. For example, in engineering, consider an inequality constraint to limit the force that usually satisfies the Slater condition. If the Slater condition does not hold then for the inequality constraint $F_1^2/F_2 \leq 0$ with $F_2 > 0$, it is possible to have a force $F_1 = 0$ and it is obviously a unique solution. Considering that this solution is on the boundary of constraint function, it means that there does not exist the interior point in the feasible region and then the slater condition is not satisfied. However, from the perspective of the practical requirement, it is meaningless for a force with 0 Newton. Therefore, the slater condition is natural for the analysis of convex optimization problems with constraints. More application scenarios including low latency communication in wireless networks [20] and online advertising [21] also assume that the Slater condition holds. In [20], the slater condition guarantees that the timely throughput constraints cannot be set arbitrarily and must be feasible under some condition. In [21], the slater condition gives the chance for the investors not to bid for the advertising, i.e., money spent is zero and therefore they can maintain their budget. Taking all of the above into consideration, we discuss the effect of the Slater condition on the proposed virtual-queue-based dynamic OCO algorithm and make a comparison. The detailed information is shown in Section 4. We sum up the major contributions of this work as follows:

1. We propose a novel and efficient virtual-queue-based approach to solve the OCO problem subject to time-variant and long-term constraints. The proposed approach possesses a better convergence rate and can guarantee that the sublinear optimization performance is not affected by the time horizon. More importantly, our approach is applicable to the dynamic OCO problem and can achieve dynamic regret bound $\mathcal{O}\left(\max\left(T^{\delta}\mathcal{D}(\boldsymbol{a}), T^{1-\delta}\right)\right)$ and constraint violation bound $\mathcal{O}\left(\max\left(T^{1-\delta}, T^{\delta}\right)\right)$, where $\mathcal{D}(\boldsymbol{a})$ records the dynamic accumulative change about the optimal benchmark $\boldsymbol{a}^*$ and δ acts as a trade-off parameter to make the upper bound of regret and constraint violations adjustable. The analyses and experiments illustrate that our algorithm can obtain sublinear regret and the constraint violations as long as the dynamic accumulative change $\mathcal{D}(\boldsymbol{a})$ grows sublinearly.
2. Considering that the Slater condition is common in many practical OCO scenarios and can ensure the existence of a feasible solution, we discuss the algorithm under the assumption that the Slater condition holds or not and analyze the impact of the Slater condition on the optimization performance. We find that the dynamic regrets have the same bound in both cases. However, the bound of constraint violation is remarkably improved if the Slater condition is satisfied. Specifically, it can be reduced from $\mathcal{O}\left(T^{1-\frac{\delta}{2}}\right)$ to $\mathcal{O}\left(\max\left(T^{\delta}, T^{1-\delta}\right)\right)$ when the Slater condition holds.
3. We implement our OCO model to the resource allocation problem in cloud computing and conduct the performance analysis. The experiment results exhibit that the upper bound of constraint violation is smaller when the Slater condition holds. More importantly, to compare our virtual-queue-based algorithm with state-of-the-art algorithms, we perform extensive experiments on our synthetic dataset used in the cloud computing problem and two real datasets including the social network ads dataset and adult dataset. The experiment results validate that our algorithm owns the faster convergence characteristic and the better performance of regret and constraint violation.

We arrange the rest of this paper as following sections. We present the related work and comparisons in Section 2. In Section 3, we state the OCO model subject to long-term constraints and propose an iterative algorithm assisted by the virtual queue for studying the optimization performance of this constrained model. We analyze the optimization performance and discuss the effect of the Slater condition in Section 4. Subsequently, we conduct experiment on resource allocation in cloud computing to validate the feasibility of our algorithms in Section 5. Lastly, we state the conclusion of our work in Section 6.

2. Related work

Although the classical unconstrained OCO formulations are useful in some situations, e.g., signal processing and control problems [4,24,27,26], it cannot deal with problems with constraints [25,28,32]. Therefore, the constrained OCO problem has been actively explored recently. The early OCO problems with constraints [31,30] required that the constraints should be satisfied all the time and did not permit instantaneous constraint violations. However, there always exists a projection operator in constrained OCO to obtain the feasible variable update at each step, but it could cause too much computational cost for performing the projection operation if the constraints are complex. Therefore, Mahdavi et al. [29] attempted to solve the OCO problems with long-term constraints where the constraint is not required to satisfy at each round but in a long-term manner and they utilized an online gradient descent algorithm to achieve sublinear bound of regret and constraint violations. Following their results, [23] proposed an adaptive version where they used two adaptive step sizes instead of a single step size to update dual variables. Besides, they introduced a trade-off parameter δ to regulate the bound of performance. Furthermore, [25] considered OCO problem with a squared long-term constraint and put forward methods to minimize the convex or strongly convex regret function. For the above works, the constraint functions are all invariant while it is more common for the optimization problem to be subject to the time-varying constraints in the real life. Therefore, Cao et al. [22] studied the OCO problem coupled with the time-variant constraints and proposed two algorithms for the gradient and bandit feedback, respectively. Similar works such as [32,43] also attached importance to the time-varying constrained OCO.

The traditional method to solve the constrained OCO problem is the online gradient descent method. The initial work [33] proposed an algorithm that combined gradient descent with projection under the unconstrained OCO setting and achieved $\mathcal{O}\left(\sqrt{T}\right)$ regret. Motivated by this work, Mahdavi et al. [29] modified the gradient descent algorithm to be suitable for the scene under long-term constraint. But for the gradient-descent-based algorithms [23,25,29,33], it usually requires a projection step at each iteration to return to the feasible region, which results in inefficient computation performance and low convergence rate of the optimization algorithm. Therefore, many constrained OCO literature [35,1] solved the equality or inequality constraint by adopting the saddle point method which was essentially a primal-dual approach. By alternating between updates of decision variables (primal variables) and Lagrange multipliers (dual variables), the optimization could be achieved whiling the constraint was satisfied. However, the step size parameter of the saddle point method is related to the time horizon, which results in the bounds of optimization performance only hold at the iteration step where the online procedure terminates.

Different from the above methods, some recent researches [37,38,41,39] designed the OCO algorithm utilizing the virtual queue approach to tackle time-varying constraints. Virtual queue based technique is originated from the Lyapunov optimization in the wireless communication and queueing systems and then extended to OCO by Neely et al. [34]. In virtual-queue-based method, virtual queues can be regarded as the penalizations to the violations of the constraints and by applying the key drift-plus-penalty (DPP) technology [19] of Lyapunov optimization, the constrained OCO problem can be transformed into the DPP minimization problem. Motivated by these works, our work also takes advantage of the virtual queues but our performance criterion is different from theirs. In [37,38,41], the regret was defined about the static and off-line optimum which is probably not a suitable optimal solution if the system to be optimized is essentially non-stationary. In our work, instead of utilizing static offline benchmark, we adopt a dynamic optimal baseline $\{\boldsymbol{a}_t^*\}$ which is more widely used in real life. Although in [39], they also chose the dynamic benchmark, our virtual queue update is totally different from their update method. In the latest work [40], the authors also presented the virtual queue based algorithm, but their constraint is time invariant and benchmark is static, which is more simple than our work.

Comparing the performance guarantees of our algorithm with those of the saddle-point-typed method including modified online saddle-point (MOSP) approach [1], although we both study the constrained OCO problems with dynamic benchmark sequence, there are two main differences between our virtual queue technique and their saddle point methods. The first is the dual variables updating rule and the other is the means of integrating constraints into the Lagrangian. Specifically, MOSP updates the dual variable to record the variation of the constraint and the step size of the update is dependent on the time horizon. However, it is common for the practical system to have an unknown time horizon and therefore the online optimization procedure may terminate at some unknown time in many real scenarios. It means that MOSP has a limited scope of application since its optimization performance can only be guaranteed at the time point when the optimization process terminates. Whereas our virtual-queue-based method tracks the constraint violations utilizing the virtual queues and it does not rely on the time horizon. It implies our method can achieve sublinear optimization performance at an arbitrary time point even when the online procedure is running. Besides, for MOSP, combining the constraint with a dual variable is a well-worn approach in the saddle-point-typed method but our virtual queue-based method is more challenging because we transform the time average constraint violation into the stability of virtual queues which is important for many practical optimization systems. Moreover, compared to the conditions needed in MOSP ($\mathcal{D}(\boldsymbol{a}) = \mathcal{O}\left(T^{\frac{2}{3}}\right)$ and $\mathcal{D}(\boldsymbol{h}(\boldsymbol{a})) = \mathcal{O}\left(T^{\frac{2}{3}}\right)$), our requirement is easier to be satisfied. In a word, these differences make our virtual queue algorithm surpass the saddle point methods in performance guarantees.

Last but not least, we compare our work with [39] which discussed the constrained OCO with or without the Slater condition. Noting that although our results are similar to theirs, they focus on the distributed network while we mainly consider the centralized architecture. Because of the distinct benefits of low construction cost and simple deployment architecture, the centralized system plays an increasing role in addressing optimization problems like resource allocation for the data centers. Therefore, we consider the centralized optimization in this paper and refer the readers interested in the distributed OCO problems to [10,11,14,16]. Additionally, the specific update approach in [42] is the gradient-descent-based method whose convergence characteristic is inferior to our virtual-queue-based algorithm.

To conclude, we make a comparison of our result with the main related works in Table 2 and give some comparison of results against these alternative approaches. For [37,39,40], these methods are all related to the virtual queue based method. In [37,40], the results of regret and constraint violations are all fixed while our bounds are all adjustable and therefore we can always get the better bounds of performance. In [39], the bounds of the regret and constraint violation depend on the cumulative variation of action and constraint function so our condition for achieving the sublinear results is easier. The results of [43] are obtained from the primal–dual online algorithm in a distributed setting which is different from our centralized structure and the results of constraint are obviously worse than ours. Compared with [29] which is based on the online gradient descent, our virtual-queue-based method is better than theirs. For example, when $\delta = \frac{1}{2}$, our constraint violation bound is $\sqrt{T}$ which is smaller than that in [29]. It is consistent with the analysis that OGD needs to perform projection which results in more constraint violations. [1] is the saddle point typed method and we can see that our condition is easier to satisfy to obtain the sublinear bounds since the result of regret in [1] is dependent on the accumulative variation of action and constraint function.

3. Problem statement and new algorithm

In this part, we firstly establish the OCO problem with time-varying and long-term constraint functions and then propose an online iterative algorithm grounded on the virtual queues to handle this problem.

3.1. OCO problem with long-term constraints

The common framework for OCO with long-term constraints is represented as follows: At every single round t, the incoming learner chooses a vector $\boldsymbol{a}_t \in \mathbb{R}^d$ in the convex set $\mathcal{A} \subset \mathbb{R}^d$. After the selection of the decision $\boldsymbol{a}_t$, the environment reports the loss function $l_t : \mathbb{R}^d \mapsto \mathbb{R}$ and the constraint function on $\boldsymbol{a}_t$ to the learner. In this paper, we consider multiple constraints and define the constraint function in vector form as $\boldsymbol{h}_t : \mathbb{R}^d \mapsto \mathbb{R}^m$. We denote $h_{t,k}(\cdot)$ as the k-th element of constraint vector function $\boldsymbol{h}_t(\cdot)$, that is, $\boldsymbol{h}_t(\boldsymbol{a}) = [h_{t,1}(\boldsymbol{a}), \dots, h_{t,m}(\boldsymbol{a})]^T$. Both $l_t(\boldsymbol{a})$ and $h_{t,k}(\boldsymbol{a})$ are determined via the environment and can change within every round, whereas they are presumed as the convex functions. Moreover, the function $l_t(\boldsymbol{a})$ and $h_{t,k}(\boldsymbol{a})$ are presumed to possess subgradients $l'_t(\boldsymbol{a}) : \mathbb{R}^d \mapsto \mathbb{R}^d$ and $h'_{t,k}(\boldsymbol{a}) : \mathbb{R}^d \mapsto \mathbb{R}^d, \forall \boldsymbol{a} \in \mathcal{A}$.

Table 1
Summary of Main Notations.

$\boldsymbol{a}_t$	an adaptive parameter at time t
$\boldsymbol{a}_t^*$	the per-slot minimizer at time t
V	a trade-off parameter used in Lagrangian function
$\boldsymbol{Q}(t)$	the virtual queue introduced in this paper
d	the dimensionality of vector parameters
m	the number of constraint functions
$l_t(\boldsymbol{a})$	loss function
$\boldsymbol{h}_t(\boldsymbol{a})$	vector-valued constraint function
α	learning rate or stepsize
U	the upperbound of $\|\boldsymbol{a} - \boldsymbol{b}\|$
C	the upperbound of $\|l_t(\boldsymbol{a})|$ and $|h_{t,k}(\boldsymbol{a})|$, respectively
S	the upperbound of $\|l_t'(\boldsymbol{a})\|$ and $\|h_{t,k}'(\boldsymbol{a})\|$, respectively
$\mathcal{D}(\boldsymbol{a})$	$\sum_{t=1}^{T}\|\boldsymbol{a}_t^* - \boldsymbol{a}_{t-1}^*\|$
$\mathcal{D}(\boldsymbol{h}(\boldsymbol{a}))$	$\sum_{t=1}^{T}\|\boldsymbol{h}_t(\boldsymbol{a}) - \boldsymbol{h}_{t-1}(\boldsymbol{a})\|$

Table 2
A Summary of Related Work on Constrained OCO.

Ref.	Benchmark	VQ	Slater Condition	Regret	Constraint Violation
[29]	Static	No	No	$\mathcal{O}\left(T^{\frac{1}{2}}\right)$	$\mathcal{O}\left(T^{\frac{3}{4}}\right)$
[43]	Dynamic	No	No	$\mathcal{O}\left(\max\left(T^{1-\delta}, T^{\delta}\right)\right)$	$\mathcal{O}\left(max\left(T^{\frac{1}{2}+\frac{\delta}{2}}, T^{1-\frac{\delta}{2}}\right)\right)$
[37]	Static	Yes	Yes	$\mathcal{O}\left(T^{\frac{1}{2}}\right)$	$\mathcal{O}\left(T^{\frac{1}{2}}\right)$
[39]	Dynamic	Yes	No	$\mathcal{O}(max(\mathcal{D}(\boldsymbol{a}), \mathcal{D}(h(\boldsymbol{a}))))$	$\mathcal{O}(max(\mathcal{D}(\boldsymbol{a}), \mathcal{D}(h(\boldsymbol{a}))))$
[1]	Dynamic	No	Yes	$\mathcal{O}\left(\max\left(T^{\frac{1}{3}}\mathcal{D}(\boldsymbol{a}), T^{\frac{1}{3}}\mathcal{D}(h(\boldsymbol{a})), T^{\frac{2}{3}}\right)\right)$	$\mathcal{O}\left(T^{\frac{2}{3}}\right)$
[40]	Static	Yes	Yes	$\mathcal{O}\left(T^{\frac{1}{2}}\right)$	$\mathcal{O}(1)$
			No		$\mathcal{O}\left(T^{\frac{1}{4}}\right)$
[42]	Dynamic	No	Yes	$\mathcal{O}\left(\max\left(T^{\delta}\mathcal{D}(\boldsymbol{a}), T^{1-\delta}\right)\right)$	$\mathcal{O}\left(\max\left(T^{1-\delta}, T^{\delta}\right)\right)$
			No		$\mathcal{O}\left(T^{1-\frac{\delta}{2}}\right)$
our work	Dynamic	Yes	Yes	$\mathcal{O}\left(\max\left(T^{\delta}\mathcal{D}(\boldsymbol{a}), T^{1-\delta}\right)\right)$	$\mathcal{O}\left(\max\left(T^{1-\delta}, T^{\delta}\right)\right)$
			No		$\mathcal{O}\left(T^{1-\frac{\delta}{2}}\right)$

We define the cumulative regret as well as the constraint violation for a type of online learning algorithm by: $Reg(T) = \sum_{t=1}^{T} l_t(\boldsymbol{a}_t) - \sum_{t=1}^{T} l_t(\boldsymbol{a}_t^*)$ and $Vio(T) = \sum_{t=1}^{T} \boldsymbol{h}_t(\boldsymbol{a}_t)$, wherein $\{\boldsymbol{a}^*\}$ representing the optimal benchmark for minimizing $\sum_{t=1}^{T} l_t(\boldsymbol{a}_t)$. Herein, unlike the static sequence chosen by [37,38,40], our work adopts $\{\boldsymbol{a}_t^*\}_{t=1}^{\infty}$ to be a dynamic baseline. It can be seen that compared with the static benchmark, the dynamic benchmark is more appropriate for describing the underlying optimum of a dynamic system such as tracking a moving target.

In summary, we aim at efficiently generating the vector sequence $\{\boldsymbol{a}_t\}_{t=1}^{T}$ which minimize the accumulative regret while satisfying the long-term constraints, namely,

$$\min Reg(T) = \min \left[\sum_{t=1}^{T} l_t(\boldsymbol{a}_t) - \sum_{t=1}^{T} l_t(\boldsymbol{a}_t^*) \right], \tag{1}$$

$$s.t. \quad Vio_k(T) = \sum_{t=1}^{T} h_{t,k}(\boldsymbol{a}_t) \leqslant 0, \quad k = 1, \ldots, m. \tag{2}$$

Here, the action benchmark $\{\boldsymbol{a}_t^*\}$ is dynamic and the loss function $l_t(\cdot)$ is time-varying. It should be noted that $h_{t,k}(\boldsymbol{a}_t)$ is the k-th element of constraint vector function $\boldsymbol{h}_t(\boldsymbol{a}_t)$ and $\boldsymbol{h}_t(\boldsymbol{a}_t) \preceq 0$ means $h_{t,k}(\boldsymbol{a}_t) \leqslant 0, \ \forall k$. Clearly, the long-term constraint $Vio(T) = \sum_{t=1}^{T} \boldsymbol{h}_t(\boldsymbol{a}_t) \preceq \boldsymbol{0}$ can be transformed to $Vio_k(T) \leqslant 0$. Therefore, our goal is to make both regret and constraint violation sub-linear about T, namely, regarding all $k \in \{1, \ldots, m\}$, we can obtain $Reg(T) < \mathcal{O}(T)$ and $Vio_k(T) < \mathcal{O}(T)$. Then, we can finally get $\frac{Reg(T)}{T} < \mathcal{O}(1)$as well as $\frac{Vio_k(T)}{T} < \mathcal{O}(1)$ with the increase of horizon T. It means the time-average regret and constraint violation will approach zero when T tends to infinity, which guarantees that in the aspect of the optimization performance, the decision sequence $\{\boldsymbol{a}_t\}$ is not inferior to the benchmark $\{\boldsymbol{a}_t^*\}$ asymptotically.

3.2. Dynamic OCO algorithm based on virtual queue

It is important to introduce some technological assumptions before formally proposing our algorithm since they are useful for the subsequent performance analysis.

Assumption 1. (Fundamental Assumption).

a) The decision set $\mathcal{A}$ is closed convex subset of $\mathbb{R}^d$, where d is the dimensionality of decision vector.
b) For any $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{A}$, there exists a constant $U > 0$ guaranteeing $\|\boldsymbol{a} - \boldsymbol{b}\| \leqslant U$.
c) All loss function l_t and the constraint function $h_{t,k}$ are convex and uniformly bounded continuous functions, that is, there exists a constant $C > 0$ guaranteeing $|l_t(\boldsymbol{a})| \leqslant C, |h_{t,k}(\boldsymbol{a})| \leqslant C, \forall \boldsymbol{a} \in \mathcal{A}, t \in \{1, \ldots, T\}$ and $k = 1, \ldots, m$.
d) The subgradients of l_t and $h_{t,k}$ are uniformly bounded, that is, there is a constant $S > 0$ guaranteeing $\|l'_t(\boldsymbol{a})\| \leqslant S$ and $\|h'_{t,k}(\boldsymbol{a})\| \leqslant S, \forall \boldsymbol{a} \in \mathcal{A}, t \in \{1, \ldots, T\}$ and $k = 1, \ldots, m$.

Assumption 2. *(Slater Condition).* There exist $\lambda > 0$ and an interior point $\boldsymbol{a} \in \mathcal{A}$ making $\boldsymbol{h}_t(\boldsymbol{a}) \preceq -\boldsymbol{\lambda}_m$ for any round t.

From Assumption 2, we can get $h_{t,k}(\boldsymbol{a}) \leqslant -\lambda, \forall k \in \{1, \ldots, m\}$. The Slater condition is innate under numerous circumstances even when the constraint functions are given offline, i.e., $h_{t,k} = h_k$. Moreover, we can guarantee the existence of bounded Lagrange multiplier $\{\boldsymbol{Q}(t)\}$ when the Slater condition holds [44].

In this paper, the main idea of tackling the constrained optimization problem is the virtual-queue based method that is essentially a kind of Lagrange multiplier method:

$$L_t(\boldsymbol{a}) = Vl_t(\boldsymbol{a}) + \boldsymbol{Q}(t)^T \boldsymbol{h}_t(\boldsymbol{a}) + \alpha\|\boldsymbol{a} - \boldsymbol{a}_{t-1}\|^2, \tag{3}$$

It means $\boldsymbol{a}_t : \arg_{\boldsymbol{a} \in \mathcal{A}} \min L_t(\boldsymbol{a})$. With the preceding decision $\boldsymbol{a}_{t-1}$ and the current virtual queue $\boldsymbol{Q}(t)$, this problem can be transformed into the following online version:

$$\begin{aligned} \boldsymbol{a}_t &= argmin\left\{Vl_t(\boldsymbol{a}) + \boldsymbol{Q}(t)^T \boldsymbol{h}_t(\boldsymbol{a}) + \alpha\|\boldsymbol{a} - \boldsymbol{a}_{t-1}\|^2\right\} \\ &= \arg\min\left\{\left[Vl'_{t-1}(\boldsymbol{a}_{t-1})^T + \sum_{k=1}^{m} Q_k(t) h'_{t-1,k}(\boldsymbol{a}_{t-1})\right]^T (\boldsymbol{a} - \boldsymbol{a}_{t-1}) + \alpha\|\boldsymbol{a} - \boldsymbol{a}_{t-1}\|^2\right\}. \end{aligned}$$

Besides, it is required to update the virtual queue $\boldsymbol{Q}(t+1)$, that is, for $k = 1, \ldots, m$, it has

$$Q_k(t+1) = \max\left\{Q_k(t) + h_{t-1,k}(\boldsymbol{a}_{t-1}) + h'_{t-1,k}(\boldsymbol{a}_{t-1})^T (\boldsymbol{a}_t - \boldsymbol{a}_{t-1}), 0\right\}.$$

Note that the above update strategy is much the same as the ordinary update of the virtual queue with an additional item $\boldsymbol{h}_{t-1}(\boldsymbol{a}_t)$ and therefore the virtual queues $\boldsymbol{Q}(t)$ can represent the accumulative constraint violations, that is, it is simple to transform the bound of the virtual queues into that of constraint violations.

Based on the above analysis, we put forward Algorithm 1 which utilizes the virtual queues. In this algorithm, we want to maintain and update the dynamic decision sequence $\{\boldsymbol{a}_t\}$ as well as the virtual queues $\{\boldsymbol{Q}(t)\}$. The procedure of Algorithm 1 can be summarized as: With the previous iteration $\boldsymbol{a}_{t-1}$ and current virtual queue $\boldsymbol{Q}(t)$, we update $\boldsymbol{a}_t$ at each round according to the below update expression:

$$\arg\min\left\{\left[Vl'_{t-1}(\boldsymbol{a}_{t-1})^T + \sum_{k=1}^{m} Q_k(t) h'_{t-1,k}(\boldsymbol{a}_{t-1})\right]^T (\boldsymbol{a} - \boldsymbol{a}_{t-1}) + \alpha\|\boldsymbol{a} - \boldsymbol{a}_{t-1}\|^2\right\} \tag{4}$$

and at the same time, update every single virtual queue $Q_k(t+1)$ for all $k = 1, \ldots, m$:

$$Q_k(t+1) = \max\left\{Q_k(t) + h_{t-1,k}(\boldsymbol{a}_{t-1}) + h'_{t-1,k}(\boldsymbol{a}_{t-1})^T (\boldsymbol{a}_t - \boldsymbol{a}_{t-1}), 0\right\}. \tag{5}$$

To facilitate the analysis, we introduce scalar $\tilde{h}_{t-1,k}(\boldsymbol{a}) : \mathbb{R}^d \mapsto \mathbb{R}$ to depict the variation of the convex function $h_{t-1,k}(\boldsymbol{a})$ and define

$$\begin{aligned} \tilde{h}_{t-1,k}(\boldsymbol{a}) &= h'_{t-1,k}(\boldsymbol{a}_{t-1})^T (\boldsymbol{a} - \boldsymbol{a}_{t-1}), \\ \tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{a}) &= \left[\tilde{h}_{t-1,1}(\boldsymbol{a}), \ldots, \tilde{h}_{t-1,m}(\boldsymbol{a})\right]^T. \end{aligned}$$

Substituting $\tilde{h}_{t-1,k}(\boldsymbol{a})$ into (4) and (5), we can obtain the more simple update expression:

$$\boldsymbol{a}_t = \arg\min\Big\{Vl'_{t-1}(\boldsymbol{a}_{t-1})^T(\boldsymbol{a}-\boldsymbol{a}_{t-1}) + \boldsymbol{Q}(t)^T\tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{a}) + \alpha\|\boldsymbol{a}-\boldsymbol{a}_{t-1}\|^2\Big\}, \tag{6}$$

$$Q_k(t+1) = max\Big\{Q_k(t) + h_{t-1,k}(\boldsymbol{a}_{t-1}) + \tilde{h}_{t-1,k}(\boldsymbol{a}_t), 0\Big\}. \tag{7}$$

Many works [40,39] solved the minimization expression based on Eq. (3) to study the constrained OCO problems. In their work, $l_t()$ and $\boldsymbol{h}_t()$ were used to describe loss function and constraint function, respectively. However, in this paper, we introduce the subgradient $l'_{t-1}()$ and $h'_{t-1,k}()$ to minimize the online optimization problem (4) and (5). This slight discrepancy plays a vital role in analyzing the later performance.

Algorithm 1: Dynamic OCO Algorithm Based on the Virtual Queue

1: **Input:** trade-off parameter V and stepsize parameter α; primal iterate $\boldsymbol{a}_0 \in \mathcal{A}$; initial value $Q_k(1) = 0, k = 1, \ldots, m$; maximum iteration T. When each round t ends, report loss function $\{l_t\}_{t=1}^{\infty}$ and constraint function sequences $\{\boldsymbol{h}_t\}_{t=1}^{\infty}$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: obtain $\boldsymbol{a}_t$ via:

$$\arg\min\Big\{Vl'_{t-1}(\boldsymbol{a}_{t-1})^T(\boldsymbol{a}-\boldsymbol{a}_{t-1}) + \boldsymbol{Q}(t)^T\tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{a}) + \alpha\|\boldsymbol{a}-\boldsymbol{a}_{t-1}\|^2\Big\}$$

4: Calculate each virtual queue $Q_k(t+1)$:

$$Q_k(t+1) = max\Big\{Q_k(t) + h_{t-1,k}(\boldsymbol{a}_{t-1}) + \tilde{h}_{t-1,k}(\boldsymbol{a}_t), 0\Big\}$$

5: **end for**

Remark 1. Some recent works [37,38] also made use of the virtual queue based approach for solving the constrained OCO problem. Howerver, the authors of these works chose the static offline optimal sequence $\{\boldsymbol{a}^*\}$ to study the regret. As illustrated earlier, when the system is non-stationary, it is improper to use static optimum to investigate the optimal point of the system. Hence, considering the broader application prospect, we choose dynamic benchmark $\{\boldsymbol{a}_t^*\}$ to discuss the regret.

Remark 2. In [39], Cao et al. also focused on the dynamic OCO constrained problem assisted by the virtual queue method. Similarly, the dynamic sequence $\{\boldsymbol{a}_t^*\}$ was used to analyze the minimization problem of regret in their work. However, compared with their method, our optimization algorithm differs in the following two aspects. First, we introduce the subgradient $l'_t(\cdot)$ and $\boldsymbol{h}'_t(\cdot)$ to better analyze the loss function and constraint function, while in [39], they adopted the minimization expression based on (3) which only contains the loss function $l_t(\cdot)$ and constraint function $\boldsymbol{h}_t(\cdot)$. Secondly, our virtual queue updating method (5) is completely different from theirs.

4. Dynamic OCO algorithm analysis

In this section, under basic Assumption 1, we mainly investigate the dynamic OCO problem with cumulative constraints. Firstly, we evaluate the performance for Algorithm 1 with the Slater condition and give the upper bounds of regret as well as constraint violation. Then, we discuss the corresponding results without the Slater condition. By comparison, we summarize the impact of Slater condition on the constraint violations bound. Finally, according to the aforementioned bounds of performance, we give the sufficient conditions to ensure that Algorithm 1 can make both regret and constraint violations sublinear.

Firstly, we present a definition and some valuable lemmas for our later performance analysis.

Definition 1. *(Lyapunov Drift).* For virtual queue $\boldsymbol{Q}_t$, a Lyapunov function $\mathcal{F}_t = \frac{1}{2}\|\boldsymbol{Q}_t\|_2^2$ is defined for quantifying the size of the queue backlog at time t. Moreover, a Lyapunov drift $\Delta(t)$ is defined as follows for describing the change of the queue backlog:

$$\Delta(t) := \mathcal{F}_{t+1} - \mathcal{F}_t = \frac{1}{2}\Big[\|\boldsymbol{Q}(t+1)\|^2 - \|\boldsymbol{Q}(t)\|^2\Big].$$

From [37,38], it can be known that for this type of virtual-queue-related problem, drift-plus-penalty (DPP) technique as a classical tool can be used to analyze the regret but also the constraint violations:

$$\underbrace{\Delta(t)}_{drift} + \underbrace{\alpha\|\boldsymbol{a}_t-\boldsymbol{a}_{t-1}\|^2 + Vl'_{t-1}(\boldsymbol{a}_{t-1})^T(\boldsymbol{a}_t-\boldsymbol{a}_{t-1})}_{weighted\ \ penalty}, \tag{8}$$

where $\Delta(t)$ is used to evaluate the constraint violation and is closely related to the virtual queues. The weighted penalty includes the regularization term $\|\boldsymbol{a}_t - \boldsymbol{a}_{t-1}\|^2$ which is used to smoothen the difference between the coherent actions. The remaining terms is used to describe the optimization problem. α is the step size parameter. V is acted as the trade-off parameter to balance the optimization and the penalty of constraint violations. Moreover, as we all know, for the OCO with long-term constraints, it is a core work to analyze the DPP expression so as to discuss the bounds of the regret as well as the constraint violations. Therefore, we first give some useful lemmas about drift and penalty for our algorithm analysis.

Lemma 1. *(Simple upper bound of drift). From the definition of the Lyapunov drift, at the t-th iteration, it has*

$$\Delta(t) \leqslant A + \boldsymbol{Q}(t)^T\left[\boldsymbol{h}_{t-1}(\boldsymbol{a}_{t-1}) + \tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{a}_t)\right],$$

where $A = \frac{m}{2}(C + SU)^2$ *and* m, C, S, U *are defined in Table 1.*

Proof. Considering that $\max[j, 0] \leqslant j^2, \forall j \in \mathbb{R}$ and then for the virtual queue update expression (5), it has:

$$Q_k(t+1)^2 \leqslant \left[Q_k(t) + h_{t-1,k}(\boldsymbol{a}_{t-1}) + \tilde{h}_{t-1,k}(\boldsymbol{a}_t)\right]^2 = Q_k(t)^2 + \left(h_{t-1,k}(\boldsymbol{a}_{t-1}) + \tilde{h}_{t-1,k}(\boldsymbol{a}_t)\right)^2 + 2Q_k(t)\left[h_{t-1,k}(\boldsymbol{a}_{t-1}) + \tilde{h}_{t-1,k}(\boldsymbol{a}_t)\right].$$

Define $\rho_k(t) = h_{t-1,k}(\boldsymbol{a}_{t-1}) + \tilde{h}_{t-1,k}(\boldsymbol{a}_t)$ and then get:

$$Q_k(t+1)^2 - Q_k(t)^2 \leqslant \rho_k(t)^2 + 2Q_k(t)\left[h_{t-1,k}(\boldsymbol{a}_{t-1}) + \tilde{h}_{t-1,k}(\boldsymbol{a}_t)\right].$$

Sum up the above inequality for $k = 1, \ldots, m$ and utilizing the Definition 1, it has:

$$\Delta(t) \leqslant \frac{1}{2}\sum_{k=1}^{m}\rho_k(t)^2 + \sum_{k=1}^{m}Q_k(t)\left[h_{t-1,k}(\boldsymbol{a}_{t-1}) + \tilde{h}_{t-1,k}(\boldsymbol{a}_t)\right].$$

Note that $|\rho_k(t)| \leqslant C + SU$ and write in vector-valued form, we can get the result in Lemma 1. □

Next, we start to present the lemma about the penalty. Because of the $\|\boldsymbol{a}_t - \boldsymbol{a}_{t-1}\|^2$ term, the discussion in penalty is related to minimizing a strongly convex function. Therefore, we introduce the following lemma.

Lemma 2. *(Strong convexity analysis). Suppose that* $\boldsymbol{z} \in \mathcal{A}$ *is an arbitrary vector variable. Then, it has*

$$\begin{aligned}&Vl'_{t-1}(\boldsymbol{a}_{t-1})^T(\boldsymbol{a}_t - \boldsymbol{a}_{t-1}) + \boldsymbol{Q}(t)^T\tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{a}_t) + \alpha\|\boldsymbol{a}_t - \boldsymbol{a}_{t-1}\|^2\\ &\leqslant Vl'_{t-1}(\boldsymbol{a}_{t-1})^T(\boldsymbol{z} - \boldsymbol{a}_{t-1}) + \boldsymbol{Q}(t)^T\tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{z}) + \alpha\|\boldsymbol{z} - \boldsymbol{a}_{t-1}\|^2 - \alpha\|\boldsymbol{z} - \boldsymbol{a}_t\|^2.\end{aligned}$$

Proof. As we all know, for c-strongly convex function $q : \mathcal{A} \to \mathbb{R}$ which has a minimization point $\boldsymbol{a}_{min} \in \mathcal{A}_0$ then we have $q(\boldsymbol{a}_{min}) \leqslant q(\boldsymbol{a}) - \frac{c}{2}\|\boldsymbol{a} - \boldsymbol{a}_{\boldsymbol{min}}\|^2$. Then, considering that the minimization update expression (4) is strongly convex with modulus 2α and $\boldsymbol{a}_t$ is a minimization value, therefore we can get the above lemma. □

Besides, in the previous work [37], they considered the regret and constraint violation problem which was partially similar to ours. Therefore, we also conclude their results about constraint violation and regret in the following lemmas. Note that although we present the following lemmas about regret and constraint violations in the related constrained OCO work, it is still not trivial to analyze the regret and constraint violation bound for our algorithm since our work considers the dynamic benchmark and the effect of the Slater condition.

Lemma 3. *For* $k = 1, \ldots, m$ *and* $T \geqslant 1$*, it has*

$$\sum_{t=0}^{T-1}h_{t,k}(\boldsymbol{a}_t) \leqslant \|\boldsymbol{Q}(T+1)\| + \frac{VS^2T}{2\alpha} + \frac{\sqrt{m}S^2}{2\alpha}\sum_{t=1}^{T}\|\boldsymbol{Q}(t)\|.$$

The above lemma is directly from Corollary 2 in [37]. Moreover, we can get the useful lemma about regret from the proof of Theorem 1 in [37].

Lemma 4. *(The drift bound directly connected to regret [38])..*

4.1. Performance based on the Slater condition

In this subsection, we aim to bound the regret and constraint violation of Algorithm 1 under the Slater condition based on the above lemmas. Now, we begin to discuss the regret function. In this paper, we choose the dynamic benchmark $\{\boldsymbol{a}_t\}_{t=1}^{\infty}$ rather than the static benchmark $\boldsymbol{a}^*$ in [37,40,38] to bound the regret function. To quantify the temporal variations of function sequences, we define $\mathcal{D}(\boldsymbol{a})$ as the accumulated variation of the optimal sequence $\boldsymbol{a}_t^*$, i.e., $\mathcal{D}(\boldsymbol{a}) = \sum_{t=1}^{T}\|\boldsymbol{a}_t^* - \boldsymbol{a}_{t-1}^*\|$. Next, we give the main result regarding the regret of Algorithm 1 in the following theorem.

Theorem 1. *(Regret). Supposing $t \in \{0, 1, \ldots, T-1\}$, it has*

$$\sum_{t=0}^{T-1} l_t(\boldsymbol{a}_t) \leqslant \sum_{t=0}^{T-1} l_t(\boldsymbol{a}_t^*) + \frac{\alpha U^2}{V} + \frac{2\alpha U \mathcal{D}(\boldsymbol{a})}{V} + \frac{TVS^2}{4\alpha} + \frac{TA}{V},$$

where $\mathcal{D}(\boldsymbol{a})$ represents the accumulative change of the optimal dynamic sequence and A is defined in Lemma 1.

Proof. From Lemma 4, let $\boldsymbol{z} = \boldsymbol{a}_{t-1}^*$ and we can get, for $t = \{1, 2, \ldots T\}$,

$$Vl_{t-1}(\boldsymbol{a}_{t-1}) \leqslant Vl_{t-1}(\boldsymbol{a}_{t-1}^*) - \Delta(t) + \alpha\|\boldsymbol{a}_{t-1}^* - \boldsymbol{a}_{t-1}\|^2 - \alpha\|\boldsymbol{a}_{t-1}^* - \boldsymbol{a}_t\|^2 + \frac{V^2S^2}{4\alpha} + A.$$

To simplify, we rewrite it as, for $t = \{0, 1, \ldots, T-1\}$

$$Vl_t(\boldsymbol{a}_t) \leqslant Vl_t(\boldsymbol{a}_t^*) - \Delta(t+1) + \alpha\|\boldsymbol{a}_t^* - \boldsymbol{a}_t\|^2 - \alpha\|\boldsymbol{a}_t^* - \boldsymbol{a}_{t+1}\|^2 + \frac{V^2S^2}{4\alpha} + A.$$

Sum it and we can get

$$V\sum_{t=0}^{T-1} l_t(\boldsymbol{a}_t) \leqslant V\sum_{t=0}^{T-1} l_t(\boldsymbol{a}_t^*) - \sum_{t=0}^{T-1}\Delta(t+1) + \alpha\sum_{t=0}^{T-1}\left(\|\boldsymbol{a}_t^* - \boldsymbol{a}_t\|^2 - \|\boldsymbol{a}_t^* - \boldsymbol{a}_{t+1}\|^2\right) + T\left(\frac{V^2S^2}{4\alpha} + A\right). \tag{9}$$

We note $\sum_{t=0}^{T-1}\Delta(t+1) = \mathcal{F}(T+1) - \mathcal{F}(1)$ and $\mathcal{F}(1) = \frac{1}{2}\|\boldsymbol{Q}_1\|^2 = 0, \mathcal{F}(T+1) = \frac{1}{2}\|\boldsymbol{Q}(T+1)\|^2 \geqslant 0$. Therefore,

$$-\sum_{t=0}^{T-1}\Delta(t+1) \leq 0. \tag{10}$$

Besides, we can see that

$$\begin{aligned}
&\|\boldsymbol{a}_t^* - \boldsymbol{a}_t\|^2 - \|\boldsymbol{a}_t^* - \boldsymbol{a}_{t+1}\|^2 = \|\boldsymbol{a}_t - \boldsymbol{a}_t^*\|^2 - \|\boldsymbol{a}_{t+1} - \boldsymbol{a}_{t+1}^* + \boldsymbol{a}_{t+1}^* - \boldsymbol{a}_t^*\|^2 \\
&\overset{(a)}{\leqslant} \|\boldsymbol{a}_t - \boldsymbol{a}_t^*\|^2 - \left(\|\boldsymbol{a}_{t+1} - \boldsymbol{a}_{t+1}^*\|^2 - 2\|\boldsymbol{a}_{t+1} - \boldsymbol{a}_{t+1}^*\|\|\boldsymbol{a}_{t+1}^* - \boldsymbol{a}_t^*\| + \|\boldsymbol{a}_{t+1}^* - \boldsymbol{a}_t^*\|^2\right) \\
&\overset{(b)}{\leqslant} \|\boldsymbol{a}_t - \boldsymbol{a}_t^*\|^2 - \|\boldsymbol{a}_{t+1} - \boldsymbol{a}_{t+1}^*\|^2 + 2U\|\boldsymbol{a}_{t+1}^* - \boldsymbol{a}_t^*\|,
\end{aligned}$$

where (a) is from $\|\xi_1 + \xi_2\|^2 \geqslant \|\xi_1\|^2 - 2\|\xi_1\|\|\xi_2\| + \|\xi_2\|^2, \forall\xi_1, \xi_2$; (b) is from Assumption 1 and drop the positive term $\|\boldsymbol{a}_{t+1}^* - \boldsymbol{a}_t^*\|^2$. Next, we sum above inequality from $t = 0$ to $T-1$, then

$$\sum_{t=0}^{T-1}\left(\|\boldsymbol{a}_t^* - \boldsymbol{a}_t\|^2 - \|\boldsymbol{a}_t^* - \boldsymbol{a}_{t+1}\|^2\right) \leqslant \|\boldsymbol{a}_0 - \boldsymbol{a}_0^*\|^2 - \|\boldsymbol{a}_T - \boldsymbol{a}_T^*\|^2 + 2U\sum_{t=0}^{T-1}\|\boldsymbol{a}_{t+1}^* - \boldsymbol{a}_t^*\| \overset{(a)}{\leqslant} U^2 + 2U\sum_{t=0}^{T-1}\|\boldsymbol{a}_{t+1}^* - \boldsymbol{a}_t^*\|, \tag{11}$$

where (a) is due to $\|\boldsymbol{a}_0 - \boldsymbol{a}_0^*\|^2 - \|\boldsymbol{a}_T - \boldsymbol{a}_T^*\|^2 \leqslant U^2$ by Assumption 1-b. The residual term $\sum_{t=0}^{T-1}\|\boldsymbol{a}_{t+1}^* - \boldsymbol{a}_t^*\|$ reflects the influence of our dynamic benchmark in regret function. From the definition $\mathcal{D}(\boldsymbol{a}) = \sum_{t=0}^{T-1}\|\boldsymbol{a}_{t+1}^* - \boldsymbol{a}_t^*\|$ and according to (11), it has

$$\sum_{t=0}^{T-1}\left(\|\boldsymbol{a}_t^* - \boldsymbol{a}_t\|^2 - \|\boldsymbol{a}_t^* - \boldsymbol{a}_{t+1}\|^2\right) \leqslant U^2 + 2U\mathcal{D}(\boldsymbol{a}). \tag{12}$$

Substituting (10) and (12) into (9), we can get the bound of regret:

$$\sum_{t=0}^{T-1} l_t(\boldsymbol{a}_t) \leqslant \sum_{t=0}^{T-1} l_t(\boldsymbol{a}_t^*) + \frac{\alpha U^2}{V} + \frac{2\alpha U \mathcal{D}(\boldsymbol{a})}{V} + \frac{TVS^2}{4\alpha} + \frac{TA}{V}.$$

□

Next, we start to investigate the bound of constraint violations. In fact, according to (5), the virtual queue is closely related to the constraint violation. To this end, we first discuss the upper bound of $\boldsymbol{Q}(t)$.

Lemma 5. *Suppose $Q_k(1) = 0, \forall k \in \{1, \ldots, m\}$. When the Slater condition hold, it has for each round t:*

$$\|\boldsymbol{Q}(t)\| \leqslant \frac{A + VSU}{\lambda} + \frac{\alpha U^2}{\lambda V} + \frac{(V+1)\sqrt{m}(C + SU)}{2}, \tag{13}$$

where A is given in Lemma 1.

Proof. According to Lemma 2, then

$$V l'_{t-1}(\boldsymbol{a}_{t-1})^T(\boldsymbol{a}_t-\boldsymbol{a}_{t-1})+\boldsymbol{Q}(t)^T\tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{a}_t)+\alpha\|\boldsymbol{a}_t-\boldsymbol{a}_{t-1}\|^2$$
$$\leqslant V l'_{t-1}(\boldsymbol{a}_{t-1})^T(\boldsymbol{z}-\boldsymbol{a}_{t-1})+\boldsymbol{Q}(t)^T\tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{z})+\alpha\|\boldsymbol{z}-\boldsymbol{a}_{t-1}\|^2-\alpha\|\boldsymbol{z}-\boldsymbol{a}_t\|^2.$$

Note that the definition of $\tilde{\boldsymbol{h}}$ represents the variation of a convex function $\boldsymbol{h}$. Then, we have $\tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{z})+\boldsymbol{h}_{t-1}(\boldsymbol{a}_{t-1})\leqslant\boldsymbol{h}_{t-1}(\boldsymbol{z})$. Adding $\boldsymbol{Q}(t)^T\boldsymbol{h}_{t-1}(\boldsymbol{a}_{t-1})$ on both side, then we have

$$V l'_{t-1}(\boldsymbol{a}_{t-1})^T(\boldsymbol{a}_t-\boldsymbol{a}_{t-1})+\boldsymbol{Q}(t)^T\left[\tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{a}_t)+\boldsymbol{h}_{t-1}(\boldsymbol{a}_{t-1})\right]+\alpha\|\boldsymbol{a}_t-\boldsymbol{a}_{t-1}\|^2$$
$$\leqslant V l'_{t-1}(\boldsymbol{a}_{t-1})^T(\boldsymbol{z}-\boldsymbol{a}_{t-1})+\boldsymbol{Q}(t)^T\boldsymbol{h}_{t-1}(\boldsymbol{z})+\alpha\|\boldsymbol{z}-\boldsymbol{a}_{t-1}\|^2-\alpha\|\boldsymbol{z}-\boldsymbol{a}_t\|^2.$$

Rearranging terms yields

$$\boldsymbol{Q}(t)^T\left[\tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{a}_t)+\boldsymbol{h}_{t-1}(\boldsymbol{a}_{t-1})\right]$$
$$\leqslant V l'_{t-1}(\boldsymbol{a}_{t-1})^T[(\boldsymbol{z}-\boldsymbol{a}_{t-1})-(\boldsymbol{a}_t-\boldsymbol{a}_{t-1})]+\alpha\|\boldsymbol{z}-\boldsymbol{a}_{t-1}\|^2-\alpha\|\boldsymbol{z}-\boldsymbol{a}_t\|^2-\alpha\|\boldsymbol{a}_t-\boldsymbol{a}_{t-1}\|^2+\boldsymbol{Q}(t)^T\boldsymbol{h}_{t-1}(\boldsymbol{z})$$
$$\overset{(a)}{\leqslant} V l'_{t-1}(\boldsymbol{a}_{t-1})^T(\boldsymbol{z}-\boldsymbol{a}_t)+\alpha\|\boldsymbol{z}-\boldsymbol{a}_{t-1}\|^2-\alpha\|\boldsymbol{z}-\boldsymbol{a}_t\|^2+\boldsymbol{Q}(t)^T\boldsymbol{h}_{t-1}(\boldsymbol{z})$$
$$\overset{(b)}{\leqslant} V\|l'_{t-1}(\boldsymbol{a}_{t-1})\|\|\boldsymbol{z}-\boldsymbol{a}_t\|+\alpha\|\boldsymbol{z}-\boldsymbol{a}_{t-1}\|^2-\alpha\|\boldsymbol{z}-\boldsymbol{a}_t\|^2+\boldsymbol{Q}(t)^T\boldsymbol{h}_{t-1}(\boldsymbol{z})$$
$$\overset{(c)}{\leqslant} VSU+\alpha\|\boldsymbol{z}-\boldsymbol{a}_{t-1}\|^2-\alpha\|\boldsymbol{z}-\boldsymbol{a}_t\|^2+\boldsymbol{Q}(t)^T\boldsymbol{h}_{t-1}(\boldsymbol{z}),$$

where (a) drops the nonnegative term $\alpha\|\boldsymbol{a}_t-\boldsymbol{a}_{t-1}\|^2$; (b) uses the Cauchy–Schwarz inequality; (c) considers Assumption 1-d and 1-b. Combining the above result with Lemma 1, we obtain the drift bound directly connected to constraint violations:

$$\Delta(t)\leqslant A+\boldsymbol{Q}(t)^T\left[\boldsymbol{h}_{t-1}(\boldsymbol{a}_{t-1})+\tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{a}_t)\right]\leqslant A+VSU+\alpha\|\boldsymbol{z}-\boldsymbol{a}_{t-1}\|^2-\alpha\|\boldsymbol{z}-\boldsymbol{a}_t\|^2+\boldsymbol{Q}(t)^T\boldsymbol{h}_{t-1}(\boldsymbol{z}). \quad (14)$$

From the queue update Eq. (4), we can konw that for every single round t, the maximum variation range of $Q_k(t)$ is $C+SU$. Therefore, $\|\boldsymbol{Q}(t)\|$ can change by at most $\sqrt{m}(C+SU)$ over one slot. Suppose V is a positive integer, when round t satisfies $t\in\{0,1,\ldots,V\}$ it has $\|\boldsymbol{Q}(t)\|\leqslant\sqrt{m}(C+SU)V$. Subsequently, we use mathematical induction to find the upper bound of $\|\boldsymbol{Q}(t)\|$ denoted as MAX.

First, choose a round $T\geqslant V$ and assume that $\|\boldsymbol{Q}(t)\|\leqslant MAX$ always hold when $t\leqslant T$.

Next, it is required to ensure that $\|\boldsymbol{Q}(T+1)\|\leqslant MAX$ still holds for $t=T+1$. It should be noted that $V\geqslant 1$ then $T-V+1\leqslant T$, and thus $\|\boldsymbol{Q}(T-V+1)\|\leqslant MAX$ according to the above assumption. Here, we consider two following cases: $\|\boldsymbol{Q}(T+1)\|\leqslant\|\boldsymbol{Q}(T-V+1)\|$ and $\|\boldsymbol{Q}(T+1)\|>\|\boldsymbol{Q}(T-V+1)\|$. Therefore, it is required to ensure that $\|\boldsymbol{Q}(t+1)\|\leqslant MAX$ in both cases. As to the case 1, assume $\|\boldsymbol{Q}(T+1)\|\leqslant\|\boldsymbol{Q}(T-V+1)\|$ and due to $\|\boldsymbol{Q}(T-V+1)\|\leqslant MAX$, then we get $\|\boldsymbol{Q}(T+1)\|\leqslant MAX$. For the case 2, assume $\|\boldsymbol{Q}(T+1)\|>\|\boldsymbol{Q}(T-V+1)\|$. Sum (14) for $t=T-V+1,\ldots,T$ then:

$$\mathcal{F}(T+1)-\mathcal{F}(T-V+1)\overset{(a)}{\leqslant}(A+VSU)V+\alpha\left(\|\boldsymbol{z}-\boldsymbol{a}_{T-V}\|-\|\boldsymbol{z}-\boldsymbol{a}_T\|^2\right)-\lambda\sum_{t=T-V+1}^{T}\|\boldsymbol{Q}(t)\|$$
$$\overset{(b)}{\leqslant}(A+VSU)V+\alpha U^2-\lambda\sum_{t=T-V+1}^{T}\|\boldsymbol{Q}(t)\|,$$

where (a) follows $\boldsymbol{Q}(t)^T\boldsymbol{h}_{t-1}(\boldsymbol{z})=\sum_{k=1}^{m}Q_k(t)h_{t-1,k}(\boldsymbol{z})$, the Slater condition $h_{t-1,k}(\boldsymbol{z})\leqslant-\lambda$ and $\sum_{k=1}^{m}Q_k(t)>\|\boldsymbol{Q}(t)\|$; (b) follows $\|\boldsymbol{z}-\boldsymbol{a}_{T-V}\|-\|\boldsymbol{z}-\boldsymbol{a}_T\|\leqslant U^2$ by Assumption 1-b. Considering that for case 2, $\|\boldsymbol{Q}(T+1)\|>\|\boldsymbol{Q}(T-V+1)\|$ holds and then $\mathcal{F}(T+1)-\mathcal{F}(T-V+1)>0$. Hence, it has

$$\lambda\sum_{t=T-V+1}^{T}\|\boldsymbol{Q}(t)\|\leqslant(A+VSU)V+\alpha U^2. \quad (15)$$

Because the maximum variation range of the queue norm is $\sqrt{m}(C+SU)$ and then $\|\boldsymbol{Q}(T+1)\|-\|\boldsymbol{Q}(t)\|\leqslant(T+1-t)\sqrt{m}(C+SU)$. Therefore,

$$\sum_{t=T-V+1}^{T}\|\boldsymbol{Q}(T+1)\|\leqslant\sum_{t=T-V+1}^{T}\|\boldsymbol{Q}(t)\|+\sum_{t=T-V+1}^{T}(T+1-t)\sqrt{m}(C+SU).$$

Substituting (15) and our requirement $\|\boldsymbol{Q}(T+1)\|\leqslant MAX$ into the above inequality, then we have $MAX\leqslant\frac{A+VSU}{\lambda}+\frac{\alpha U^2}{\lambda V}+\frac{(V+1)\sqrt{m}(C+SU)}{2}$. Therefore, choose $MAX=\frac{A+VSU}{\lambda}+\frac{\alpha U^2}{\lambda V}+\frac{(V+1)\sqrt{m}(C+SU)}{2}$ and then we find the upper bound of $\|\boldsymbol{Q}(t)\|$. □

From the aforementioned analysis, we finally give the bound related to the constraint violations.

Theorem 2. *(Constraint Violation). Suppose the deterministic Slater condition hold and then for each constraint $k \in \{1, \ldots, m\}$ we have:*

$$\sum_{t=0}^{T-1} h_{t,k}(\boldsymbol{a}_t) \leqslant \frac{A + VSU}{\lambda} + \frac{\alpha U^2}{\lambda V} + \frac{(V+1)\sqrt{m}(C+SU)}{2} + \frac{VS^2T}{2\alpha} + \frac{\sqrt{m}S^2}{2\alpha}\frac{(A+VSU)T + \alpha U^2}{\lambda},$$

where m is the number of constraints and constant A is defined in Lemma 1.

Proof. Combining Lemma 3 with Lemma 5, we can get the constraint violation bound

$$\sum_{t=0}^{T-1} h_{t,k}(\boldsymbol{a}_t) \leqslant MAX + \frac{VS^2T}{2\alpha} + \frac{\sqrt{m}S^2}{2\alpha}\sum_{t=1}^{T}\|\boldsymbol{Q}(t)\|, \tag{16}$$

where MAX denotes the upper bound of $\|\boldsymbol{Q}(t)\|$. Then, we start to discuss the upper bound of $\sum_{t=1}^{T}\|\boldsymbol{Q}(t)\|$. From (14), we can get

$$\Delta(t) \leqslant A + \boldsymbol{Q}(t)^T\left[\boldsymbol{h}_{t-1}(\boldsymbol{a}_{t-1}) + \tilde{\boldsymbol{h}}_{t-1}(\boldsymbol{a}_t)\right] \leqslant A + VSU + \alpha\|\boldsymbol{z} - \boldsymbol{a}_{t-1}\|^2 - \alpha\|\boldsymbol{z} - \boldsymbol{a}_t\|^2 + \boldsymbol{Q}(t)^T\boldsymbol{h}_{t-1}(\boldsymbol{z}), \tag{17}$$

and under Slater condition, we can get

$$\Delta(t) \leqslant A + VSU + \alpha\|\boldsymbol{z} - \boldsymbol{a}_{t-1}\|^2 - \alpha\|\boldsymbol{z} - \boldsymbol{a}_t\|^2 - \lambda\|\boldsymbol{Q}(t)\|. \tag{18}$$

On both sides, summing from 1 to T, we can get

$$\mathcal{F}(T+1) - \mathcal{F}(1) \leqslant (A + VSU)T + \alpha\left(\|\boldsymbol{z} - \boldsymbol{a}_0\|^2 - \|\boldsymbol{z} - \boldsymbol{a}_T\|^2\right) - \lambda\sum_{t=1}^{T}\|\boldsymbol{Q}(t)\|.$$

Note that $\|\boldsymbol{z} - \boldsymbol{a}_0\|^2 - \|\boldsymbol{z} - \boldsymbol{a}_T\|^2 \leqslant U^2$ by Assumption 1-b and $\mathcal{F}(1) = 0, \mathcal{F}(T+1) > 0$ by Definition 1, then

$$\sum_{t=1}^{T}\|\boldsymbol{Q}(t)\| \leqslant \frac{(A+VSU)T + \alpha U^2}{\lambda}.$$

Substituting the above inequality and MAX into (16), we get the upper bound of constraint violation:

$$\sum_{t=0}^{T-1} h_{t,k}(\boldsymbol{a}_t) \leqslant \frac{A + VSU}{\lambda} + \frac{\alpha U^2}{\lambda V} + \frac{(V+1)\sqrt{m}(C+SU)}{2} + \frac{VS^2T}{2\alpha} + \frac{\sqrt{m}S^2}{2\alpha}\frac{(A+VSU)T + \alpha U^2}{\lambda}.$$

□

Remark 3. Different from [38], in our analysis of constraint violation, we rediscuss the drift bound for the constraint violation. That is to say, we use (17) rather than Lemma 4 to study the constraint violation. However, in [38], the authors used the same drift bound in both regret and constraint violation analysis. Therefore, it brings more terms about V and α in the constraint violation bound, which is inconvenient to analyze the upper bound of constraint violation. The method similar to ours is also adopted in Appendix B of [37].

Remark 4. It should be noted that in the proof of Theorem 2, we already get the upper bound for every $\|\boldsymbol{Q}(t)\|,\ t \in \{1, \ldots, T\}$ according to (13). In [37,38], they directly used this upper bound to calculate the upper bound of $\sum_{t=1}^{T}\|\boldsymbol{Q}(t)\|$, which equals $T \cdot MAX$. However, in this paper, we recalculate the upper bound for $\sum_{t=1}^{T}\|\boldsymbol{Q}(t)\|$ and it is helpful to get tighter upper bound. Thus, we can get tighter upper bound of constraint violation.

From the above proof process, we realize that the regret bound is independent of the parameter λ, i.e., the Slater condition does not affect the regret bound. Hence, in Section 4.2, we will not discuss the regret bound without the Slater condition.

4.2. Performance without the Slater condition

.

Theorem 3. *For each constraint $k \in \{1, \ldots, m\}$, we obtain the constraint violation bound without the Slater condition:*

$$\sum_{t=0}^{T-1} h_{t,k}(\boldsymbol{a}_t) \leqslant \left(1 + \frac{\sqrt{m}S^2T}{2\alpha}\right)\sqrt{(2A + 2VSU)T + 2\alpha U^2} + \frac{VS^2T}{2\alpha},$$

where m is the number of constraints and constant A is defined in Lemma 1.

Proof. Without the Slater condition, the upper bound of $\|\boldsymbol{Q}(t)\|$ changes, i.e., Lemma 5 is not satisfied anymore. However, as shown in (17), the drift bound directly connected to constraint violation still holds, i.e.,

$$\Delta(t) \leqslant A + VSU + \alpha\|\boldsymbol{z} - \boldsymbol{a}_{t-1}\|^2 - \alpha\|\boldsymbol{z} - \boldsymbol{a}_t\|^2 + \boldsymbol{Q}(t)^T\boldsymbol{h}_{t-1}(\boldsymbol{z}).$$

It should be noted that without the Slater condition, we still have $\boldsymbol{h}_t(\boldsymbol{a}) \leq 0, \forall \boldsymbol{a} \in \mathcal{A}$ when $t \in \{0, 1, 2, \ldots\}$. Then, we can transform the above inequality into

$$\Delta(t) \leqslant A + VSU + \alpha\|\boldsymbol{z} - \boldsymbol{a}_{t-1}\|^2 - \alpha\|\boldsymbol{z} - \boldsymbol{a}_t\|^2.$$

Summing up the above inequality from 1 to t ($t \leqslant T$), it has

$$\mathcal{F}(t+1) - \mathcal{F}(1) \leqslant (A + VSU)t + \alpha\left(\|\boldsymbol{z} - \boldsymbol{a}_0\|^2 - \|\boldsymbol{z} - \boldsymbol{a}_t\|^2\right) \leqslant (A + VSU)T + \alpha U^2.$$

According to Definition 1, we can get $\mathcal{F}(t+1) = \frac{1}{2}\|\boldsymbol{Q}(t+1)\|^2$ and $\mathcal{F}(1) = 0$. Then we have

$$\|\boldsymbol{Q}(t+1)\| \leqslant \sqrt{(2A + 2VSU)T + 2\alpha U^2}. \tag{19}$$

Based on the above queue bound, we can give the upper bound of constraint violation without Slater condition. Recall that in Lemma 3, it has

$$\sum_{t=0}^{T-1} h_{t,k}(\boldsymbol{a}_t) \leqslant \|\boldsymbol{Q}(T+1)\| + \frac{VS^2T}{2\alpha} + \frac{\sqrt{m}S^2}{2\alpha}\sum_{t=1}^{T}\|\boldsymbol{Q}(t)\|,$$

then substituting (19) into the above inequality, we can get

$$\sum_{t=0}^{T-1} h_{t,k}(\boldsymbol{a}_t) \leqslant \left(1 + \frac{\sqrt{m}S^2T}{2\alpha}\right)\sqrt{(2A + 2VSU)T + 2\alpha U^2} + \frac{VS^2T}{2\alpha}.$$

□

4.3. Performance characterisation

In the above two subsections, we obtain the upper bounds with variable V and α. Now, let $\alpha = T, V = T^{1-\delta}$ where $\delta \in (0, 1)$ and then we further simplify the above results.

Corollary 1. :

$$\sum_{t=0}^{T-1} h_{t,k}(\boldsymbol{a}_t) \sim \mathcal{O}\left(max\left(T^{1-\delta}, T^{\delta}\right)\right). \tag{20}$$

Corollary 2. :

$$\sum_{t=0}^{T-1} h_{t,k}(\boldsymbol{a}_t) \sim \mathcal{O}\left(T^{1-\frac{\delta}{2}}\right). \tag{21}$$

Remark 5. is always sublinear. Comparing (20) and (21), we can easily sum up the impact of the Slater condition on the constraint violation. When the Slater condition holds, the constraint violation bound reduces from $\mathcal{O}\left(T^{1-\frac{\delta}{2}}\right)$ to $\mathcal{O}\left(max\left(T^{1-\delta}, T^{\delta}\right)\right)$.

Corollary 3. :

$$\sum_{t=0}^{T-1} [l_t(\boldsymbol{a}_t) - l_t(\boldsymbol{a}_t^*)] \sim \mathcal{O}\left(max\left(T^{\delta}\mathcal{D}(\boldsymbol{a}), T^{1-\delta}\right)\right).$$

Remark 6. Note that the regret upper bound is closely linked to the drift of the benchmark sequence $\mathcal{D}(\boldsymbol{a})$ but independent of the Slater condition. The dynamic regret can achieve sublinear bound if the order $T^{\delta}\mathcal{D}(\boldsymbol{a})$ is *sublinear*, i.e., the drift of the benchmark grows sublinearly with a given upper bound. Moreover, there exists a lower bound $\mathcal{O}(1)$ for the cumulative drift $\mathcal{D}(\boldsymbol{a})$. If there is a constant $c \in [0, 1)$ satisfying $\mathcal{D}(\boldsymbol{a}) = \mathcal{O}(T^c)$, we shall set $\delta \in (0, 1-c)$in order to achieve a sublinear regret bound. Compared to the conditions needed in [1] ($\mathcal{D}(\boldsymbol{a}) = \mathcal{O}\left(T^{\frac{2}{3}}\right)$ and $\mathcal{D}(h(\boldsymbol{a})) = \mathcal{O}\left(T^{\frac{2}{3}}\right)$), our requirement is easier to be satisfied.

Besides, although the saddle point methods such as the modified saddle-point method (MOSP) in [1] can also guarantee that the regret and constraint violations grow sublinearly, the performance assurances for these methods cannot be achieved unless the online process comes to an end. Additionally, the step-size parameters of various saddle-point methods are all dependent on the time horizon T, but for our Algorithm 1, the setting of the step-size parameter is unrelated to T, which allows the online optimization procedure to terminate at any time. It means that we can achieve the performance guarantees of our algorithm at an arbitrary time point even for those before the ending of the online optimization process.

5. Numerical experiments

In this part, we investigate the optimization performance of Algorithm 1 under a resource allocation problem assisted by cloud computing. Specifically, we firstly discuss the effect of the drift of the benchmark sequence $\mathcal{D}(\boldsymbol{a})$ as well as the trade-off parameter δ on the regret and constraint violations. Then, we analyze the effect of the Slater condition when assuming the Slater condition hold or not. Finally, we compare our algorithm with state-of-the-art optimization algorithms to validate the effectiveness of our algorithm.

5.1. Experiment setup

Let us consider a resource allocation problem in cloud computing. In general, resource allocation can be regarded as a kind of OCO application scenario with QoS requirements from users and it can be tackled with instant scene information in real time. Through adopting cloud computing, massive historical scenario data is sampled to calculate similarities between the different scenarios using online learning which essentially belongs to the OCO algorithm. By analyzing these similarities, it is effective to utilize the remedies of resource allocation under historical scenarios to strengthen the resource allocation under the current scene. Concretely, when the measured data under a scenario reaches, it is worth making a comparison between the current scenario and the historical scenario to get the closest result. After that, the best scheme in the most similar historical scenario is exploited for allocating the radio resources of the current scenario.

The optimization problem of the resource allocation performed at the base station can be modeled as:

$$\begin{aligned} &\min_{\boldsymbol{a}\in\mathbb{R}^d}\sum_{t=1}^{T}\log\left(1+\exp\left(-y_t\boldsymbol{a}^T\boldsymbol{x}_t\right)\right),\\ &\text{subject to } \|\boldsymbol{a}\|_1 \leqslant \gamma_t, \end{aligned} \tag{22}$$

where $\{y_t|y_t \in \{-1,1\}\}$ represents a sequence of label vector for feature selection. To reduce the dimensionality of feature vectors, the feature selection is only conducted on the information useful for resource allocation. All useful information can be classified as time-varying or time-invariant information. Some constant elements, such as antenna number, subcarrier number and maximum transmit power should be labelled as time-invariant parameters. Other fast-changing elements like interference level, user number and channel state information of all users should be labelled as time-varying parameters since it is necessary to keep on measuring and feeding back this information to the resource allocator. $\boldsymbol{a}$ represents the weight vector which explains the allocation scheme of the radio resources and each element of $\boldsymbol{a}$ is used for describing the configuration or allocated amount of radio resources, such as the assigned subcarrier index and the transmit power level. $\left\{\boldsymbol{x}_t|\boldsymbol{x}_t \in \mathbb{R}^d\right\}$ represents a sequence of training vectors. The log-loss function expresses the characteristics of possible optimal solutions and exhibits the key performance indicators for resource allocation. Besides, $\gamma_t > 0$ is a qualifying parameter for describing the particular scenario or limitation in resource allocation like the QoS requirements from the users or the available amount of radio resources. This optimization problem (22) can be cast into our formulation (1) and (2) by setting $l_t(\boldsymbol{a}) = \log\left(1+\exp\left(-y_t\boldsymbol{a}^T\boldsymbol{x}_t\right)\right)$ and $h_t(\boldsymbol{a}) = \|\boldsymbol{a}\|_1 - \gamma_t$. The best scheme of resource allocation denoted by $\boldsymbol{a}^*$ aims to get the best value from the loss function and simultaneously meets all constraints.

We can generate the time-variant sequences $\{\boldsymbol{x}_t,\gamma_t\}_{t=1}^T$ as follows. First, in the current scenario, we have known $\boldsymbol{x}_t$ and γ_t at time t so we can update $\boldsymbol{x}_{t+1}$ by $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{\theta}_t$ where each element of vector $\boldsymbol{\theta}_t \in \mathbb{R}^d$ is uniformly distributed over $\left[-\frac{1}{2t^2},\frac{1}{2t^2}\right]$. Likewise, we can also update γ_t by $\gamma_{t+1} = \max\{\gamma_t+\eta_t,0\}$ where η_t is an uniformly distributed function over $\left[-\frac{1}{2t^2},\frac{1}{2t^2}\right]$. Secondly, we can generate $\boldsymbol{a}_{t+1}$ by following the steps below. With $\boldsymbol{a}_t$ in hand, we define an auxiliary vector $\hat{a}_{t+1} = \prod_{\mathcal{A}}(\boldsymbol{a}_t+\boldsymbol{v}_t)$ where each element of vector $\boldsymbol{v}_t$ is uniformly distributed over $\left[-\frac{1}{2t^2},\frac{1}{2t^2}\right]$. When the constraint condition is satisfied, i.e., $\|\hat{a}_{t+1}\|_1 \leqslant \gamma_{t+1}$, we let $\boldsymbol{a}_{t+1} = \hat{a}_{t+1}$. Otherwise, we set $\boldsymbol{a}_{t+1} = \gamma_{t+1}\frac{\hat{a}_{t+1}}{\|\hat{a}_{t+1}\|_1}$. By this way, we can ensure that the all the constraint have at least one feasible solution. As discussed in former sections, we set the default parameter values of our problem as: $d=3, m=1, T=1000, C=4, S=8, U=2, \lambda=2, \alpha=T$ and $V=\sqrt{T}$.

5.2. Performance analysis

In this part, we investigate the effect of the drift of benchmark sequence $\mathcal{D}(\boldsymbol{a})$, the trade-off parameter δ and the Slater condition on the performance of Algorithm 1.

Drift of the benchmark sequence $\mathcal{D}(\boldsymbol{a})$. From our above setups, we can obtain that the update rate of the vector-valued decision $\boldsymbol{a}_t$ is $\frac{1}{t^2}$ so the order in the difference of neighboring benchmark sequence is also $\frac{1}{t^2}$, that is, $\|\boldsymbol{a}_t^* - \boldsymbol{a}_{t-1}^*\|_2 = \mathcal{O}\left(\frac{1}{t^2}\right)$. Then, when T tends to infinity, the drift $\mathcal{D}(\boldsymbol{a}) = \sum_{t=1}^{T} \|\boldsymbol{a}_t^* - \boldsymbol{a}_{t-1}^*\| < \mathcal{O}(1)$ which approaches zero. Therefore, from Corollary 3, we can get the optimal sublinear regret $\mathrm{Reg}(T) \sim \mathcal{O}\left(\sqrt{T}\right)$ when the drift is sublinear and $\delta = \frac{1}{2}$. Similarly, under the same condition, we can get the optimal sublinear constraint violations $\mathrm{Vio}_k(T) \sim \mathcal{O}\left(\sqrt{T}\right)$ from Corollary 1.

To study the effect of $\mathcal{D}(\boldsymbol{a})$ on the performance of Algorithm 1, we increase the update rate of $\boldsymbol{a}_t$ to $\frac{1}{t}$, i.e., we set the update rate of $\boldsymbol{v}_t$ in the auxiliary vector $\hat{\boldsymbol{a}}_t$ to be an uniformly distributed function over $\left[-\frac{1}{t}, \frac{1}{t}\right]$. Thus, $\|\boldsymbol{a}_t^* - \boldsymbol{a}_{t-1}^*\|_2 = \mathcal{O}\left(\frac{1}{t}\right), \mathcal{D}(\boldsymbol{a}) = \sum_{t=1}^{T} \|\boldsymbol{a}_t^* - \boldsymbol{a}_{t-1}^*\| = \mathcal{O}(\log T)$ so we can get the regret bound $\mathrm{Reg}(T) \sim \mathcal{O}\left(\sqrt{T}\log T\right)$ and the constraint violation bound $\mathrm{Vio}_k(T) \sim \mathcal{O}\left(\sqrt{T}\right)$, both of which are sublinear. We apply Algorithm 1 to the optimization problem (22). Fig. 1 shows the results of the time-average regret $\frac{Reg(t)}{t}$ and the time-average constraint violation $\frac{\|Vio_k(t)\|}{t}$. As we see, the time-average regret and time-average constraint violation both converge to zero when the iteration approaches infinity, which implies that our algorithm can achieve the desirable sublinear optimization performance. Moreover, from Fig. 1, it shows the time-average regrets with $\mathcal{D}(\boldsymbol{a}) = \mathcal{O}(\log(T))$ are larger than that with $\mathcal{D}(\boldsymbol{a}) = \mathcal{O}(1)$ and the constraint violations barely change given the different values of $\mathcal{D}(\boldsymbol{a})$, which validate our analysis in Remark 5.

Trade-off parameter δ. To study the effect of the trade-off parameter δ on the dynamic regret bound and constraint violations bound, we set the different δ as $\delta = \frac{1}{4}, \frac{1}{3}, \frac{1}{2}$ and substitute them into Corollary 1 and 3. We show the results of the time average regret $\frac{Reg}{T}$ but also constraint violations $\frac{\|Vio_k\|}{T}$ under the different δ in Fig. 2. It can be seen that the convergence rates of the time average regret and constraint violations decrease faster as δ increases. Besides, we can see that the variation trends of these two performance metrics are the same when δ varies. Notably, the above results are acquired under the Slater condition and we omit the corresponding simulation results without the Slater condition since in Section 4, we have pointed out the optimization performance is better when the Slater condition hold.Fig. 3.

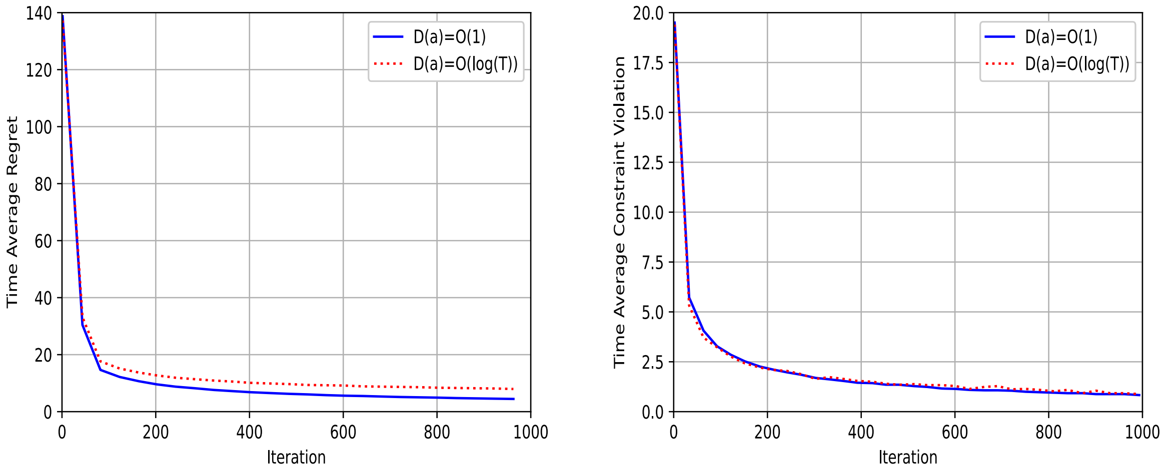
Slater condition. To investigate the effect of the Slater condition on the constraint violations, we run Algorithm 1 with or without the Slater condition. Set $\delta = 0.5, \mathcal{D}(\boldsymbol{a}) = \mathcal{O}(1)$ and then according to Corollary 1 and 2, we can get the constraint violation bound is $\mathcal{O}\left(T^{\frac{1}{2}}\right)$ when the Slater condition hold and it becomes $\mathcal{O}\left(T^{\frac{3}{4}}\right)$ when the Slater condition does not hold. As we can see, in Fig. 3(b), when the iteration arrives at 963, the time average constraint violation $\frac{\|Vio_k\|}{T}$ is 0.856 when the Slater condition hold but this value becomes 2.37 if we do not consider the slater condition. It implies that the performance of constraint violations enhances significantly when the Slater condition is guaranteed, which validates the theoretical analysis in Remark 5. Moreover, from Fig. 3(a), we can see that the time average regret $\frac{Reg}{T}$ is the same when the slater condition holds or not. Actually, with the prescribed parameter, the bound of regret is $\mathcal{O}\left(\sqrt{T}\right)$ which is a constant independent of the Slater condition. Therefore, this result validates the correctness of the analysis in Remark 6.

5.3. Comparing with state-of-the-art algorithms

To illustrate the advantages of the proposed virtual-queue-based algorithm, we compare Algorithm 1 with some latest algorithms related to the constrained OCO problems. According to the optimization technique, we divide them into the categories including the Lagrangian descent method [41], the online gradient descent (OGD) method [23,25] and saddle point method [1]. Next, we give a brief description of these algorithms as following.

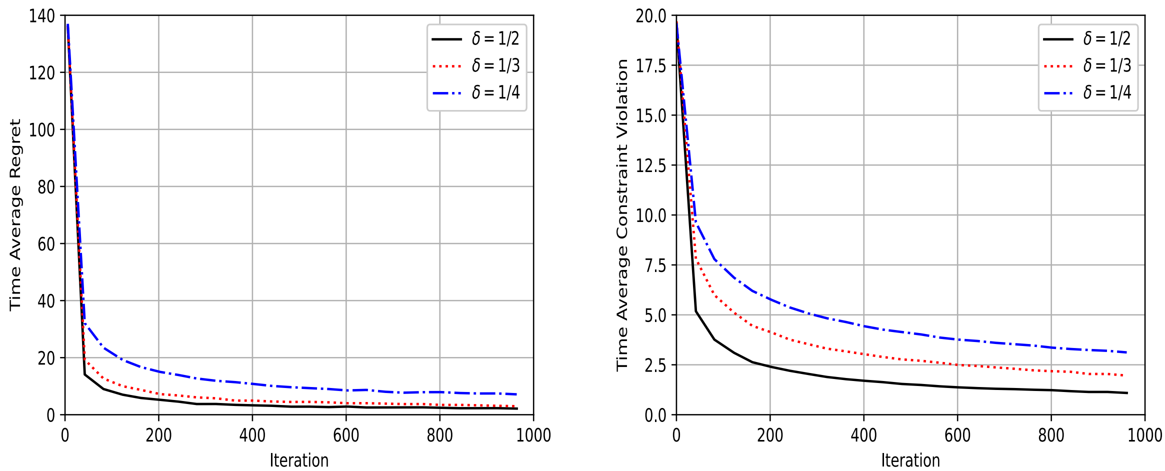
- COLD [41] is a kind of Lagrangian descent algorithm which computes a stationary point of the optimization problem in each round and then performs the projection whiling updating the constraint by the Lagrangian multiplier.
- Adap [23] adopts the online gradient descent method to transform the OCO problem with the constraint condition into the primary-descent and dual-descent and then solves this problem by the gradient descent method in an online manner.
- Clipped [25] is an extension of the OGD method which aims to reduce the cumulative constraint violation by clipping the dual-gradient.
- MOSP [1] is a modified online saddle point approach which takes a modified saddle-point recursion for updating the optimization problem and conducts a dual ascent step at each time slot in a Gauss–Seidel manner.

To compare all the algorithms with our algorithm on the same criteria, we set the following default parameters: Algorithm 1 with $\delta = 0.5$ and $\mathcal{D}(\boldsymbol{a}) = \mathcal{O}(1)$, Adap in [23] with $\delta = 0.5$, Clipped in [25] with $\delta = 0.5$, COLD in [41] with $K = T^{\frac{1}{2}}, V = T^{\frac{3}{4}}$ and MOSP in [1] with $\alpha = \mu = T^{-\frac{1}{3}}$. Considering that the constraint violations in the other algorithms could be always satisfied, i.e., the constraint violation could be always negative, we choose $Vio_k(T)$ rather than $\|Vio_k(T)\|$ as the met-



(a) Time average regret for $\mathcal{D}(\boldsymbol{a}) = O(1), O(logT)$ (b) Time average constraint violation for $\mathcal{D}(\boldsymbol{a}) = O(1), O(logT)$

Fig. 1. The effect of the drift of benchmark sequence $\mathcal{D}(\boldsymbol{a})$.

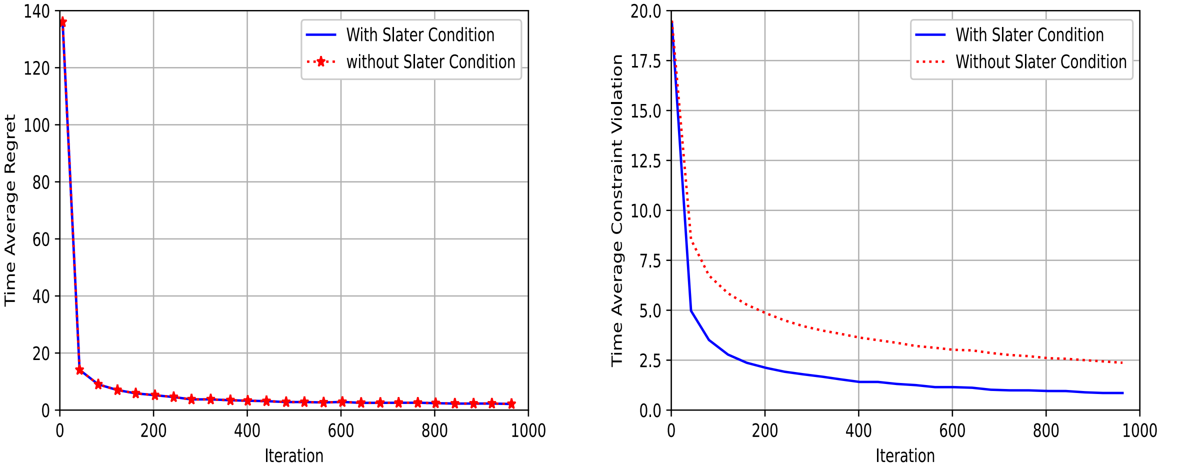


(a) Time average regret for $\delta = \frac{1}{2}, \frac{1}{3}, \frac{1}{4}$ (b) Time average constraint violation for $\delta = \frac{1}{2}, \frac{1}{3}, \frac{1}{4}$

Fig. 2. The effect of the trade-off parameter δ.

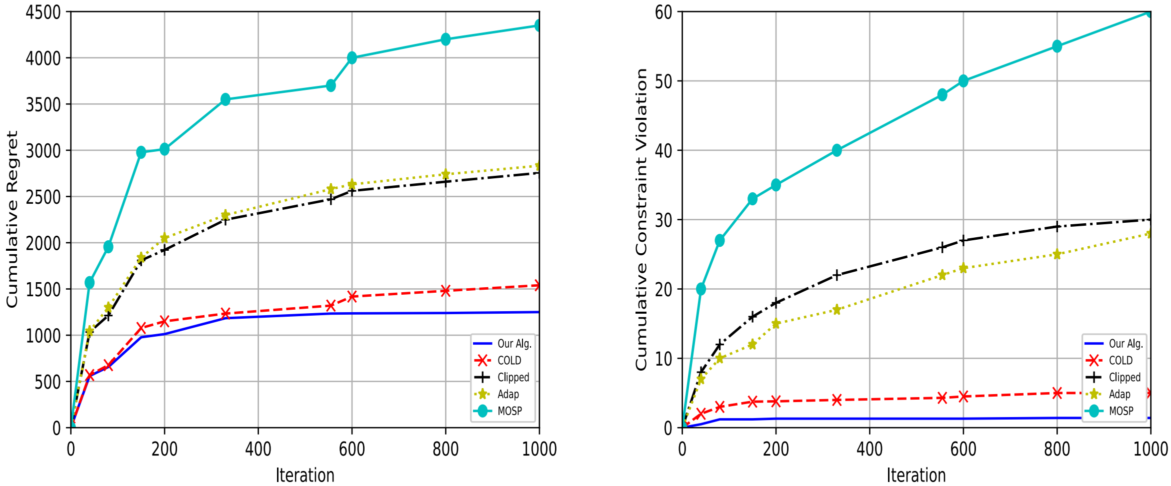
ric to describe the constraint violation. For a clearer comparison, we firstly investigate the evolution of the cumulative regret $Reg(T) = \sum_{t=0}^{T-1} l_t(\boldsymbol{a}_t) - \sum_{t=0}^{T-1} l_t(\boldsymbol{a}_t^*)$ and cumulative constraint violation $Vio_k(T) = \sum_{t=0}^{T-1} h_{t,k}(\boldsymbol{a}_t)$ for all the algorithms including our algorithm. Fig. 4 shows the comparative results of different algorithms about $Reg(t)$ and $Vio_k(t)$, respectively. As shown in Fig. 4(a), we observe that when iteration round comes to 1000, our algorithm generates the cumulative regret as 1250, but compared with our algorithm, this value for the other algorithms is: COLD with 1500 (increased by 20%), Clipped with 2750 (increased by 120%), Adap with 2800 (increased by 124%) and MOSP with 4400 (increased by 252%). It is obvious that our algorithm remarkably improves the regret performance because our algorithm has the smallest cumulative regret. Moreover, we can also see that the cumulative regret of our algorithm reaches stable in the shortest time slot, which means that our virtual-queue-based method has better convergence characteristics. From Fig. 4(b), we can see that when the round arrives at 1000, the cumulative constraint violation is 2 and this value for COLD is 5 which increases by 150% compared with our result. The other three results are much bigger than that of our algorithm and it exhibits that our algorithm has the better performance of constraint violation because our upper bound of the cumulative constraint violation is the lowest.

Next, we want to investigate the regret and constraint violation in a time averaged manner because it is beneficial to evaluate the final optimization performance. Fig. 5 shows the comparative results of these algorithms and our algorithm. From Fig. 5(a), we can see that the time average regret of all the optimization algorithms has a downward trend and it can be estimated that all the optimization algorithms will achieve sublinear regret because all the curves are approaching zero when



(a) Time average regret with or without Slater condition
(b) Time average constraint violation with or without Slater condition

Fig. 3. The effect of the Slater condition.

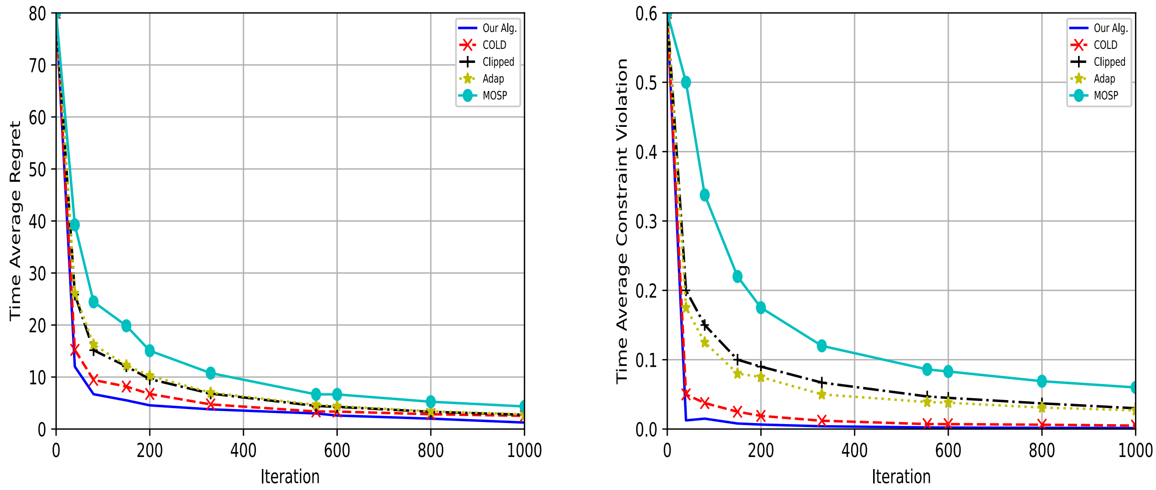


(a) Cumulative regret of different algorithms
(b) Cumulative constraint violation of different algorithms

Fig. 4. Comparing our algorithm with the state-of-the-art algorithms in a cumulative manner.

the time slot increases to infinity. Moreover, it is apparent that the result of our algorithm has the fastest rate to decrease to zero and therefore it validates that the convergence characteristic of our regret is better than that of the other method. Fig. 5 (b) shows the time average constraint violation of all the algorithms also has a downward trend, which implies that all the algorithms can achieve sublinear constraint violation. But from the rate of decrease, we can see that our time average constraint violation decrease to zero in a short time slot and therefore our algorithm can better satisfy the long-term constraint.

Finally, we want to analyze the reason for these results in more detail. Firstly, we can see that COLD has the closest result with our result since the COLD is a Lagrangian multiplier method and our virtual-queue-based method is essentially a kind of Lagrangian method. But compared with our method, the performance or rate of COLD degrades as the horizon increases. Secondly, Clipped and Adap are both based on the online gradient descent technique and as we discussed before, the gradient-descent-based algorithms usually require a projection step at each iteration to return to the feasible region, which results in inefficient computation performance and a low convergence rate of the optimization algorithm. Thirdly, as for the saddle point methods like MOSP, we have pointed that the step size parameter of the saddle point method is related to the time horizon, which results in the bounds of optimization performance only hold at the iteration step where the online procedure terminates. Moreover, compared to the conditions needed in MOSP ($\mathcal{D}(\boldsymbol{a}) = \mathcal{O}\left(T^{\frac{2}{3}}\right)$ and $\mathcal{D}(g(\boldsymbol{a})) = \mathcal{O}\left(T^{\frac{2}{3}}\right)$), our requirement is easier to be satisfied which make our result better.



(a) Time average regret of different algorithms (b) Time average constraint violation of different algorithms

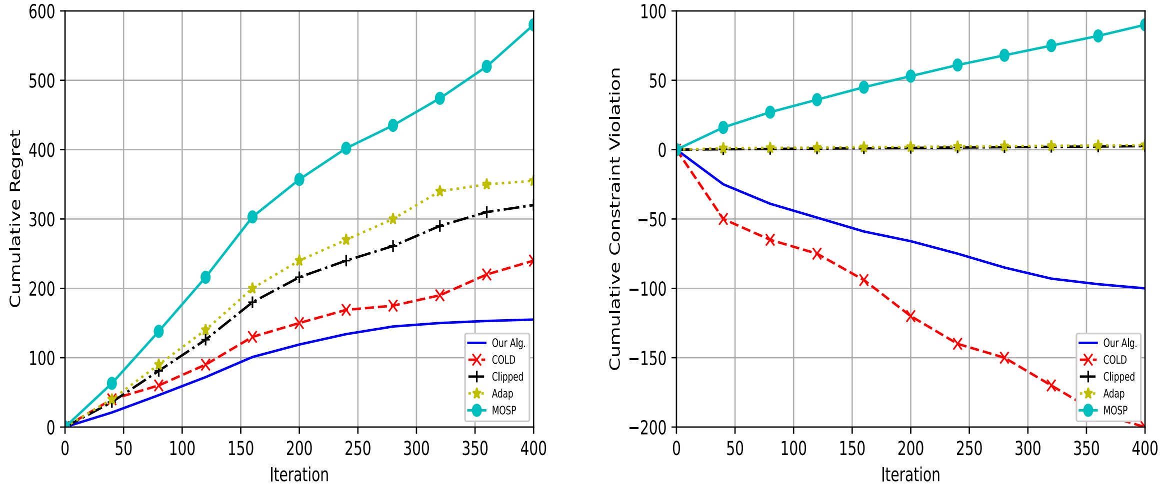
Fig. 5. Comparing our algorithm with the state-of-the-art algorithms in a time average manner.

To further illustrate that our algorithm possesses the better optimization performance, we run our algorithm on two real datasets including social network ads dataset[1] and adult dataset.[2] In the above experiment, the label y_t and training data $\boldsymbol{x}_t$ in (44) is the synthetic data under the scenario of the cloud computing. But in the real social network ads dataset, it has 400 training data and each data is with four attributes containing user id, gender, age and estimated salary. In our experiment, we only use the age as the training attribute and therefore the dimension of $\boldsymbol{x}_t$ is 1. The label is chosen from $\{0,1\}$ and it is used to indicate whether the user purchases the product. Adult dataset contains 32,500 training data and each data includes 14 attributes, such as age, eduction and occupation. The label is chosen from $\{0,1\}$ and it implies whether a person makes over 50,000 a year. Likewise, the performance of the optimization algorithm is also evaluated by the regret and constraint violation of the weight vector $\boldsymbol{a}_t$, i.e., we aim to obtain the sublinear regret and constraint violation. For comparison, we also perform the aforementioned latest algorithms on these two datasets. The default parameter is same as that in the above experiment setup but the number of iterations T is equal to the number of samples in each datase. Specifically, we set $d = 1, T = 400$ in social network ads dataset and give $d = 14, T = 32,500$ in the adult dataset.

Fig. 6(a) shows the cumulative regret for the different optimization algorithms on the social network ads dataset. From this figure, we can see that the cumulative regret of our algorithm is also the smallest one and reaches stable in the shortest time slot. For the other algorithms, the closest result is also COLD since this method is based on the Lagrangian multiplier technique but the other three algorithms still have the larger cumulative regret. Fig. 6(b) exhibits the cumulative constraint violation for different algorithms on the social network ads dataset. Different from the former results on the synthetic dataset, the cumulative constraint violation of our algorithm and COLD is always negative, which implies that our algorithm and COLD always satisfy the constraint. Besides, the absolute cumulative constraint violation $\|Vio_k(t)\|$ of our algorithm is smaller than that of COLD, which contributes to generating less loss when the constraint has been satisfied. For the online gradient descent method like Clipped and Adap, we can see that they all keep the cumulative constraint violation around zero and it means that they can also satisfy the constraint in the long term. For MOSP, it causes the largest result. To present the optimization performance more intuitively, we also plot the regret and constraint violation in time average manner in Fig. 7. Fig. 7(a) shows that the time average regret for all the algorithms are around zero and it can be estimated that all the algorithms could achieve the sublinear regret. But we can still observe that our algorithm has the lowest result and can tend to zero in the shortest time slot, which validates that our algorithm has the best regret performance. From Fig. 7(b), we can observe that the time average constraint violation of our algorithm and COLD are always negative so the constraint is always satisfied. The results of Clipped and Adap are both around zero so the long-term constraint is satisfied. For MOSP, the time average constraint violation is approaching zero so MOSP might achieve the sublinear constraint violation eventually. Apparently, all the observations on the social network ads dataset are consistent with the former analysis for the synthetic dataset, which further validates the advantages of our algorithm.

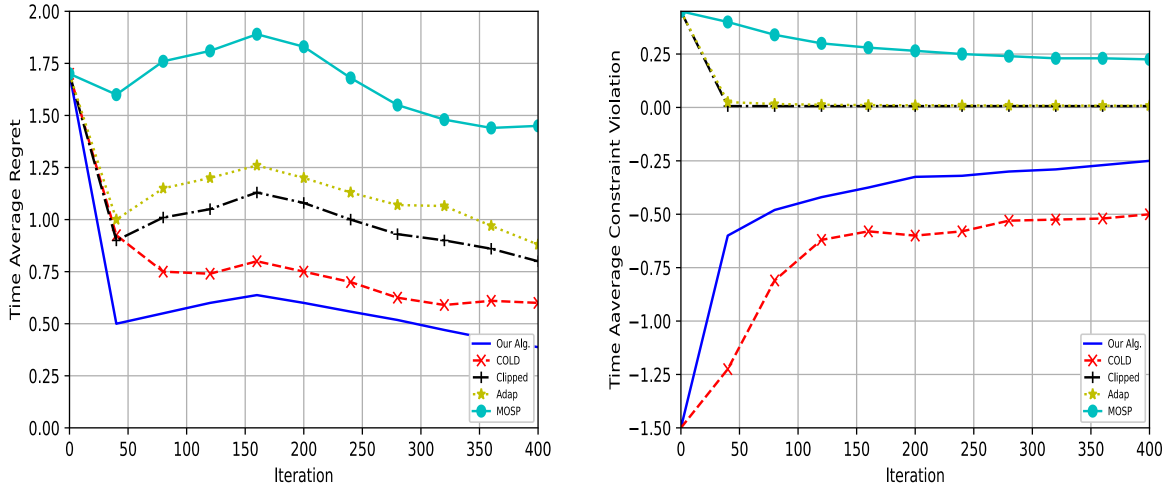
Fig. 8 exhibits the cumulative regret and constraint violation for the different algorithms on the adult dataset. The results are similar to that on the social network ads dataset and also illustrate that our algorithm achieves the better performance of regret and constraint violation. However, it should be noted that compared with the synthetic dataset and the social network ads dataset, the training data and its dimension in the adult dataset are both much more than the corresponding value in the

[1] https://www.kaggle.com/rakeshrau/social-network-ads.
[2] https://archive.ics.uci.edu/ml/datasets/adult.



(a) Cumulative regret of different algorithms (b) Cumulative constraint violation of different algorithms

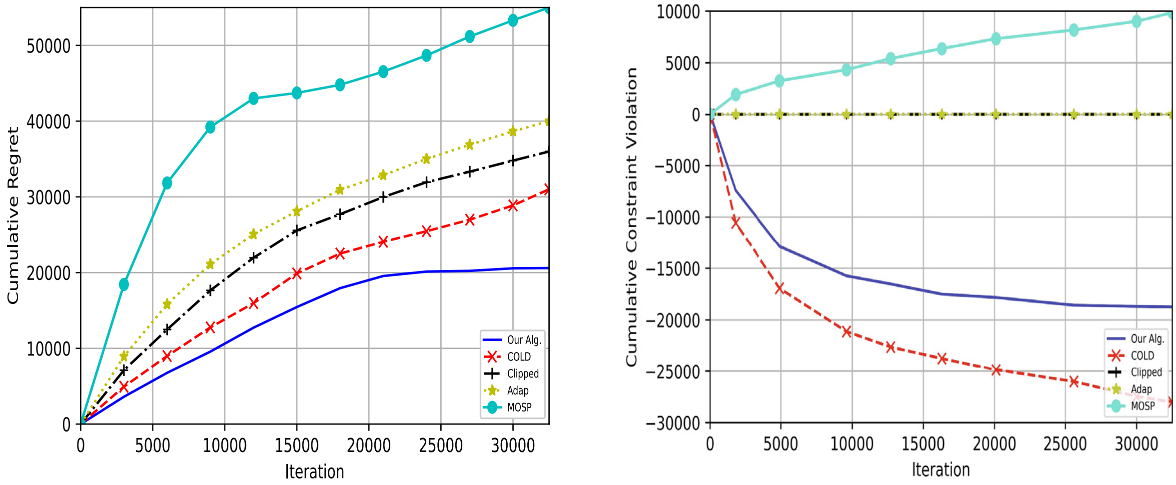
Fig. 6. Cumulative performance of different algorithms on the social network ads dataset.



(a) Time average regret of different algorithms (b) Time average constraint violation of different algorithms

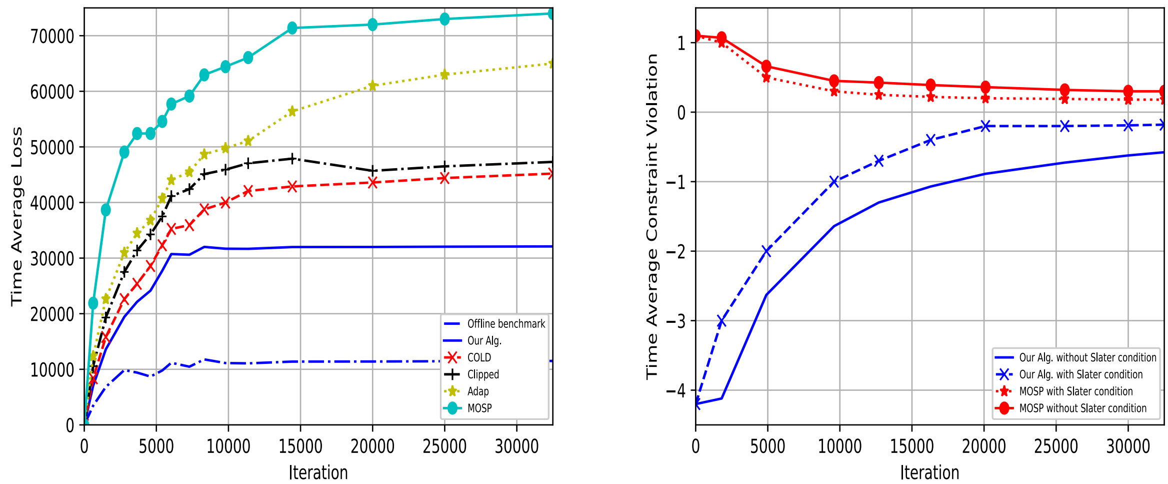
Fig. 7. Time average performance of different algorithms on the social network ads dataset.

other two datasets so our loss is also much more than that of the other two datasets. More importantly, we adopt a new way to evaluate the performance of regret. According to the definition of regret, we can consider the loss gap between the optimization method and the offline benchmark to track the evolution of the regret. Therefore, in Fig. 9(a), we present the time average loss of all the algorithms compared with the offline benchmark which is solved by CVXPY [46]. From this figure, we can see that the time average loss of our algorithm is really closest to that of the offline benchmark, which implies that our algorithm has the best regret performance. At last, when it comes to the constraint violation, among all the state-of-the-art algorithms, only MOSP discusses the Slater condition so we make a comparison with MOSP under the assumption that the Slater condition holds or not. From Fig. 9(b), whether the Slater condition holds or not, our algorithm has negative results but MOSP always has positive results, which validates that our algorithm achieves the better performance of constraint violation. Besides, our absolute time average constraint violation under the Slater condition hold is 0.2 but this value is 0.6 (increased by 200%) if the Slater condition is not assumed. Similarly, the corresponding values in MOSP are 0.1 and 0.2 (increased by 100%). From the observation, we validate the importance of the Slater condition for the constraint violation again.



(a) Cumulative regret of different algorithms (b) Cumulative constraint violation of different algorithms

Fig. 8. Cumulative performance of different algorithms on the adult dataset.



(a) Time average loss of different algorithms (b) Time average constraint violation of Alg. 1 and MOSP

Fig. 9. Time average performance of different algorithms on the adult dataset.

6. Conclusion

In this paper, we investigate the OCO problem subject to time-varying as well as long-term constraints and present an iterative OCO algorithm that makes use of the virtual queues. Compared with the traditional optimization like the dual gradient method, our method possesses better algorithm performance and a faster convergence rate. Besides, our approach is suitable for solving the dynamic OCO problem and can not only achieve the dynamic regret bound $\mathcal{O}\left(\max\left(T^{\delta}\mathcal{D}(\boldsymbol{a}), T^{1-\delta}\right)\right)$ but also the constraint violation bound $\mathcal{O}\left(\max\left(T^{1-\delta}, T^{\delta}\right)\right)$, where trade-off parameter δ can make the upper bound of regret and constraint violations adjustable. We can achieve the sublinear regret and constraint violations when the accumulated variation of optimal dynamic benchmark sequence grows sublinearly. Additionally, we discuss the algorithm under the assumption that the Slater condition holds or not and analyze the impact of the Slater condition on the optimization performance. We find that the dynamic regrets have the same bound in both cases but the bound of constraint violation is remarkably improved if the Slater condition is satisfied. Finally, we present some numerical simulations of resource allocation in cloud computing to conduct the performance analysis and compare the proposed virtual-queue-based method with the online-gradient-descent and saddle-point-typed methods. To further validate that our algorithm possesses a faster convergence rate and the lower upper bound of regret and constraint violation, we run our algorithm on two real datasets containing social network ads dataset and adult dataset to make more convincing experiment results.

Nowadays, dynamic OCO problems and constrained OCO problems are both attracting more and more attention from the optimization community. We prepare to investigate the more general dynamic situation for the OCO problem and it means that we aim to extend the dynamic model to the adaptive model in future work. Specifically, in a real-world situation, many scenes impose constraints and need to consider constraints violations under the adaptive framework. Different from the existing OCO algorithms with constraints, the adaptive algorithm needs to consider the status of the different decisions in the different time slots. Therefore, it a meaningful future work to search the decision in each time interval and then make the final decision by considering the weight possibility. Secondly, as there is two updating process in the adaptive algorithm so the loss functions and constraint violations need to be optimized together. It brings in a great challenge to the theoretical analysis, which is also a valuable future work. Finally, the trade-off between the optimization problem and the long-term constraints plays an important role in improving the algorithm performance and it is also an interesting future research topic.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61472148, 61972448 and 61772215 and the National Key Research and Development Program of China under Grant No. 2016QY02D0302. We would like to thank the anonymous reviewers for their insightful comments that have helped improve this paper.

References

[1] T. Chen, Q. Ling, G.B. Giannakis, An online convex optimization approach to proactive network resource allocation, IEEE Trans. Signal Process. 65 (24) (2017) 6350–6364.
[2] J. Kumar, A.K. Singh, R. Buyya, Self directed learning based workload forecasting model for cloud resource management, Inf. Sci. 543 (2020) 345–366.
[3] Z. Gharibsha, X. Zhu, A. Hainline, M. Conway, Deep learning for user interest and response prediction in online display advertising, Data Sci. Eng. 5 (1) (2020) 12–26.
[4] E. Hazan et al, Introduction to online convex optimization, Found. Trends Optim. 2 (3–4) (2016) 157–325.
[5] S. Kumar, X. Gao, I. Welch, M. Mansoori, "A machine learning based web spam filtering approach, in: Proc of 30th IEEE International Conference on Advanced Information Networking and Applications (AINA), 2016, pp. 973–980.
[6] O.L. Mangasarain, Nonlinear programming, Classics in Applied Mathematics, SIAM, Philadelphia, PA, 1994.
[7] S. Byod, S.P. Byod, L. Vandenberghe, Convex optimization, Cambridge University Press, Cambrideg, 2004.
[8] D.A. Gianni, P. Francesco, GGA: a modified genetic algorithm with gradient-based local search for solving constrained optimization problems, Inf. Sci. 547 (2020) 136–162.
[9] A. Koppel, F.Y. Jakubiec, A. Ribeiro, A saddle point algorithm for networked online convex optimizaiton, IEEE Trans. Signal Process. 63 (19) (2015) 5149–5164.
[10] S. Hosseini, A. Chapman, M. Mesbahi, Online distributed convex optimization on dynamic networks, IEEE Trans. Autom. Control 61 (11) (2016) 3545–3550.
[11] M. Akbari, B. Gharesifard, T. Linder, Distributed online convex optimization on time-varying directed graphs, IEEE Trans. Control Network Syst. 4 (3) (2016) 417–428.
[12] P.C. Zhu, B.Q. Xu, J.M. Li, et al, Joint utility optimization for wireless sensor networks with energy harvesting and cooperation, Sci. China Inf. Sci. 62 (2) (2020) 122302.
[13] A. Lesage-Landry, I. Shames, J.A. Taylor, Predictive online convex optimization, Automatica 113 (2020) 108771.
[14] A. Lesage-Landry, D.S. Callway, Dynamic and distributed online convex optimiaztion for demand response of commercial buildings, IEEE Control Syst. Lett. 4 (3) (2020) 632–637.
[15] J. Mandal, A. Goswami, J. Wang, M.K. Tiwari, Optimization of vehicle speed for batches to minimize supply chain cost under uncertain demand, Inf. Sci. 515 (2020) 26–43.
[16] S. Shahrampour, A. Jadbabaie, Distributed online optimization in dynamic environments using mirror descent, IEEE Trans. Autom. Control 63 (3) (2018) 714–725.
[17] A. Lesgae-Landry, J.A. Taylor, I. Shames, Second-order online nonconvex optimization, IEEE Trans. Autom. Control early access, 2019..
[18] Y. Fan et al, Data placement in distributed data centers for improved SLA and network cost, J. Parallel Distrib. Comput. 146 (2020) 189–200.
[19] M.J. Neely, Stochastic network optimization with application to communication and queueing systems, Synthesis Lectures on Communication Networks 3 (1) (2010) 1–211.
[20] S. Elazzouni, Algorithm design for low latency communication in wireless networks (Ph.D. disseration), The Ohio State University, 2020.
[21] V. Valls, G. Iosifidis, D. Leith, L. Tassiulas, Online convex optimization with perturbed constraints: optimal rates against stronger benchmarks, in: Proc. of International Conference on Artificial Intelligence and Statistics (AISTATS), 2020, pp. 2885–2895.
[22] X. Cao, K.R. Liu, Online convex optimization with time-varying constraints and bandit feedback, IEEE Trans. Autom. Control 64 (7) (2018) 2665–2680.
[23] R. Jenatton, J. Huang, C. Archambeau, Adaptive algorithms for online convex optimization with long-term constraints, in: Proc. of the 33rd International Conference on Machine Learning (ICML), 2015, pp. 402–411.
[24] S. Shalev-Shwartz et al, Online learning and online convex optimization, Found. Trends Mach. Learn. 4 (2) (2012) 107–194.
[25] J. Yuan, A. Lamperski, Online convex optimization for cumulative constraints, Proc. of Neur-IPS (2018) 6137–6146.
[26] C. Wang, W. Zheng, E. Bertino, Provenance for wireless sensor networks: a survey, Data Sci. Eng. 1 (3) (2016) 189–200.
[27] S. Wu, Y. Zhang, C. Gao, et al, GARG: anonymous recommendation of point-of-interest in mobile networks by graph convolution network, Data Sci. Eng. 5 (4) (2020) 433–447.
[28] S. Lee, M.M. Zavlanos, On the sublinear regret of distributed primal-dual algorithms for online constrained optimization, arXiv preprint arXiv:1705.11128, 2017.

[29] M. Mahdavi, R. Jin, T. Yang, Trading regret for efficiency: online convex optimization with long term constraints, J. Mach. Learn. Res. 13 (1) (2012) 2503–2528.
[30] S. Mannor, J.N. Tsitsiklis, J. Yu, Online learning with sample path constraints, J. Mach. Learn. Res. 10 (3) (2009) 569–590.
[31] S. Mannor, J.N. Tsitsiklis, Online learning with constraints, in: Proc. of International Conference on Computational Learning Theory, 2006, pp. 529–543.
[32] S. Paternain, A. Ribeiro, Online learning of feasible strategies in unknown environments, IEEE Trans. Autom. Control 62 (6) (2017) 2807–2822.
[33] M. Zinkevich, Online convex programming and generalized infinitesimal gradient ascent, in: Proc. of 20th International Conference on Machine Learning (ICML), 2003, pp. 928–936.
[34] C. Li, M.J. Neely, Delay and rate-optimal control in a multi-class priority queue with adjustable service rates, in: Proc. of 2012 IEEE INFOCOM, 2012, pp. 2976–2980.
[35] S. Lee, A. Ribeiro, M. Alejandro, Distributed continuous-time online optimization using saddle-point methods, in: Proc. of 55th IEEE Conference on Decision and Control (CDC), 2016, pp. 4314–4319.
[36] W. Stephan, S.M. Gunther, An introduction to duality in convex optimization, in: Proc. of Seminar Innovative Internet Technologies and Mobile Communications (IITMC), 2011, pp. 153–162.
[37] H. Yu, M. Neely, X. Wei, Online convex optimization with stochastic constraints, Proc. of Neur-IPS (2017) 1428–1438.
[38] M.J. Neely, H. Yu, Online convex optimization with time-varying constraints, arXiv preprint arXiv:1702.04783, 2017.
[39] X. Cao, J. Zhang, H.V. Poor, A virtual-queue-based algorithm for constrained online convex optimization with applications to data center resource allocation, IEEE J. Sel. Top. Signal Process. 12 (4) (2018) 703–716.
[40] H. Yu, M.J. Neely, A low complexity algorithm with $O\left(\sqrt{T}\right)$ regret and O(1) constraint violations for online convex optimization with long term constraints, J. Mach. Learn. Res. 21 (1) (2020) 1–24.
[41] N. Liakopoulos, A. Destounis, G. Paschos, T. Spyropoulos, P. Mertikopoulos, Cautious regret minimization: online optimization with long-term budget constraints, in: Proc. of ICML, 2019, pp. 3944–3952.
[42] X. Yi, X. Li, L. Xie, K. Johansson, Distributed online convex optimization with time-varying coupled inequality constraints, IEEE Trans. Signal Process. 68 (2020) 731–746.
[43] J. Li, C. Gu, Z. Wu, T. Huang, Online learning algorithm for distributed convex optimization with time-varying coupled constraints and bandit feedback, IEEE Trans. Cybern. (2020).
[44] D.P. Bertsekas, Nonlinear programming, Athena Scientific, Belmont, Massachusetts, MA, 1999.
[45] X. Wei, H. Yu, M.J. Neely, Online learning in weakly coupled markov decision processes: a convergence time study, in: Proc. of the ACM on Measurement and Analysis of Computing Systems, 2018, pp. 1–38.
[46] S. Diamond, S. Boyd, CVXPY: a Python-embedded modeling language for convex optimization, J. Mach. Learn. Res. 17 (1) (2016) 2909–2913.