

Online Network Utility Maximization: Algorithm, Competitive Analysis, and Applications

Ying Cao, Bo Sun, and Danny H.K. Tsang, *IEEE Fellow*

Abstract—We consider an online version of the well-studied network utility maximization problem, where users arrive one by one and an operator makes irrevocable decisions for each user without knowing the details of future arrivals. We propose a threshold-based algorithm and analyze its worst-case performance. We prove that the competitive ratio of the proposed algorithm is linearly increasing in the number of links in a network and show this competitive analysis is tight. Extensive trace-driven simulations are conducted to demonstrate the performance of our proposed algorithm. In addition, since worst-case scenarios rarely occur in practice, we devise an adaptive implementation of our algorithm to improve its average-case performance and validate its effectiveness via simulations.

I. INTRODUCTION

NETWORK utility maximization (NUM) is a general optimization paradigm of vital importance in the field of networking. It has been widely applied after the seminal work by Kelly *et al.* [1]. For example, it often serves as the underlying model to draw insights on understanding and designing congestion control mechanisms in computer networks [1] and media access control protocols in wireless networks [2], [3]. In addition, the utility-based models are shown to be effective for the power-aware load balancing and queueing in the communication network management [4]. Utility-based maximization has also been shown to play an important role in areas outside traditional communication networks, such as the demand response in power systems [5], the electric vehicle charging control [6], information dissemination in the vehicular ad-hoc networks [7] and many other applications [8]–[11].

Existing research on NUM usually assumes users are static. Therefore, they mainly focus on designing distributed algorithms to solve the offline problem. In reality, users usually arrive one by one in an online manner. For example, hosts come sequentially requesting network access in public networks. In addition, cloud service requests arrive one by one to a cloud data center. Thus, we consider an online version of NUM and term it as online network utility maximization (ONUM). Compared with its offline counterpart, the difficulty of ONUM originates from the fact that the information about the problem, e.g., the utility function, is revealed piece by piece. We need to make irrecoverable decisions that only depend on the causal information (i.e., the past and current information), and the

aim is to be as close as possible to the offline optimum that can be obtained if all information is given from the start.

There exist some efforts in the literature that deal with the ONUM problem. Some works assume that utility functions are drawn i.i.d. from an unknown distribution and use online learning to solve ONUM under the regret minimization framework, such as [12], [13]. However, the i.i.d. assumption may not be valid in reality, and it is worth noting that the capacity constraints will be violated in those algorithms. Another line of work assumes that the utility functions are generated adversarially and designs algorithms under the competitive analysis framework. For instance, [14] considers the online packing problem and [15] considers the online linear programming problem. However, both of them allow violation of the constraints. One special case in [14] can ensure no constraints are violated when all constraints coefficients are either 0 or 1; however, they just consider a simple linear objective without uncertainty. [16] and [17] consider non-linear uncertain objectives but the non-linearity source is different from ours, which will be clear in the next section. In this paper, we consider designing online algorithms for the general ONUM under the competitive analysis framework and can ensure no violation of the constraints.

A systematic way to design and analyze competitive online algorithms is to follow the online primal-dual framework, which is based on the weak duality and has been applied successfully to several classic online problems, such as online covering and packing [14] [18], online matching [19] and weighted paging [20], etc. The algorithm proposed in this paper is also related to the dual, however, the dual problem for the ONUM problem with general concave utility functions is not able to be expressed as explicitly as the counterpart problems mentioned before. Thus, we bypass the online primal-dual analysis method in this paper and adopt a different analysis method, which directly bounds the competitive ratio by identifying the worst-case instances.

The contributions of this paper consist of three parts:

- **Algorithm.** We design an online *threshold-based algorithm* for ONUM with general concave utility functions and hard capacity constraints. The threshold is an increasing function of the resource utilization level, whose curvature tunes the behavior of the algorithm, balancing between being aggressive and conservative.
- **Competitive Analysis.** We show under the umbrella of the competitive analysis framework, that when the threshold function satisfies certain *sufficient conditions*, the algorithm yields a bounded competitive ratio. We

The authors are with Department of Electrical and Computer Engineering, Hong Kong University of Science and Technology. (e-mail: ycaoan, bsunaa, eetsang@ust.hk).

adopt a direct analysis method which does not rely on the dual and prove the tightness of the analysis by characterizing the worst-case instances.

- **Applications.** We apply the algorithm to real network traces and propose a practical adaptive implementation based on *online learning* that can tune the parameters of the threshold-based algorithm, yielding much better empirical results.

This paper is organized as follows. In Section II, we present the system model for ONUM and show some applications that fit the model. In Section III, we present the threshold-based algorithm, explain the intuition, and give a tight competitive analysis of the algorithm rigorously. In Section IV, we first conduct simulations to demonstrate the proposed algorithm's performance, compare it with two heuristic algorithms under various arrival patterns, and propose viable adaptive implementation methods that strategically adjust the algorithm parameters to further improve the empirical results. In Section V, we conclude the paper and discuss about promising future directions.

II. ONLINE NETWORK UTILITY MAXIMIZATION

We describe the general system model for ONUM and show some exemplary applications in this section. A network with a link set \mathcal{L} is considered, where each link $\ell \in \mathcal{L}$ has a capacity of c_ℓ . Let L be the total number of links. We consider the homogeneous capacity case, i.e., $c_\ell = 1$ for any link. We let a user represent a traffic flow with a source and a destination. Users come to the network one by one. The transmission rate of the user needs to be determined upon its arrival and the allocation is irrevocable. The routing path of a user is a set of links that connect its source and destination. For the i th user, denote its routing path as \mathcal{L}_i . In our problem, we assume that the routing path is determined before the arrival of the user. The capacity consumption at each link on the routing path of a user is equal to its rate allocation. The total rate allocation on any link cannot exceed its capacity. In addition, the user also comes with a utility function $g_i(\cdot)$ and a budget b_i . The utility function is a function of the rate allocated to the user and the budget is the highest rate at which the user is able to transmit. After allocating an amount of rate y_i to the i th user, $g_i(y_i)$ amount of utility can be gained. We denote user i by the tuple $A_i = \{g_i(\cdot), \mathcal{L}_i, b_i\}$. Let \mathcal{N} denote the set of users and let N be the total number of users. The goal is to design an online algorithm that determines the rate allocation y_i by the time of i th arrival, maximizing the total utility of all users. Note that y_i is determined without knowing the utility functions of future users, i.e., $\{g_k(\cdot)\}_{k>i}$. If the full sequence of arrivals $\mathcal{I} = \{A_1, \dots, A_N\}$ is disclosed at the beginning, our problem is formulated as follows:

$$\max_{y_i} \sum_{i \in \mathcal{N}} g_i(y_i) \quad (1a)$$

$$\text{s.t.} \quad \sum_{i: \ell \in \mathcal{L}_i} y_i \leq 1, \forall \ell \in \mathcal{L}, \quad (1b)$$

$$0 \leq y_i \leq b_i, \forall i \in \mathcal{N}. \quad (1c)$$

A convex program solver can solve Problem (1) optimally. We denote the optimal objective as $\text{OPT}(\mathcal{I})$. The performance of an online algorithm is usually evaluated by the *competitive analysis* [21]. Given a request sequence \mathcal{I} , let $\text{ALG}(\mathcal{I})$ be the objective value achieved by an online algorithm. The performance of the online algorithm is evaluated by the competitive ratio defined as

$$\alpha = \max_{\forall \mathcal{I}} \frac{\text{OPT}(\mathcal{I})}{\text{ALG}(\mathcal{I})}.$$

The competitive ratio is desired to be as small as possible, such that the algorithm performs as close as possible to the offline optimum even under the worst case.

We make the following assumptions on the utility functions.

Assumption 1 $g_i(y)$ is increasing, strictly concave and continuously differentiable over $[0, b_i]$ and $g_i(0) = 0$.

Assumption 2 the marginal utility averaged over the number of links is bounded from above and below, i.e., $\frac{g'_i(y)}{|\mathcal{L}_i|} \in [m, M]$.

Assumption 1 is standard for NUM problems (e.g., [1], [3], [5]). It means that the user utility is usually increasing in the rate allocated and is concave because of the diminishing returns property. The commonly used utility functions are logarithm functions and polynomial functions, and thus the differentiability assumption is usually satisfied. Assumption 2 requires the first-order derivatives of utility functions to be bounded, which is essential to achieve a bounded competitive ratio. Similar assumptions appear in other online optimization literature, e.g., online knapsack problem [22], [23] and one-way trading problem [24].

We now provide some exemplary applications that fit into the ONUM model.

Online routing of virtual circuits [25]. In the simplest version, requests $r_i = (s_i, t_i)$ come online with a predetermined routing path between source s_i and destination t_i . The algorithm will determine whether or not the request can be accepted. If the request is accepted, the algorithm then decides how much bandwidth y_i should be allocated to the request and establish a virtual circuit with the requested routing path. The aggregate throughput is $\sum_i y_i$, and thus a throughput-maximizing objective is linear in the allocated bandwidth. The methods and results developed in this paper can be easily applied to this case.

Online flow control for wireless sensor networks [11]. A sensor network is modeled as a connected graph $G(\mathcal{V}, \mathcal{L})$, where \mathcal{V} denotes the sensor nodes and \mathcal{L} denotes the logical bidirectional communication links between the sensor nodes. Due to the broadcast nature of sensors, each link $\ell \in \mathcal{L}$ has an interference link set IS_ℓ . Each sensor node $v \in \mathcal{V}$ has an energy capacity e_v and each link $\ell \in \mathcal{L}$ has an interference margin level c_ℓ , which guarantees the transmission rate of the flow on a link if the margin level is observed by all flows in the link's interference link set. Traffic flows are generated online. For the i th flow, it goes through a set of sensors \mathcal{V}_i and a set of links \mathcal{L}_i with the transmission rate y_i to be determined. Also, the i th flow is characterized by a utility function $g_i(y_i)$ that is strictly concave in y_i .

There are two sets of constraints, the link capacity constraints $\sum_{\ell' \in \mathcal{L}_\ell} \sum_{i: \ell' \in \mathcal{L}_i} y_i \leq c_\ell$ for each link $\ell \in \mathcal{L}$ and the energy constraints $(e^t + e^r) \sum_{i: v \in \mathcal{V}_i} y_i \leq e_v/T - e^d$ for each sensor $v \in \mathcal{V}$, where T is the pre-specified sensor lifetime, e^t, e^r and e^d are the energy consumption per unit data during the transmission, reception and idle state, respectively. The goal is to maximize the sum of utility $\sum_i g_i(y_i)$ subject to the two sets of constraints. It can be studied under the ONUM model by simply normalizing the parameters.

Online rate control of elastic traffic [26]. Consider a network with a set \mathcal{L} of resources, and let c_ℓ be the finite capacity of resource ℓ . Users arrive online to generate traffic along its chosen route, which is a subset of \mathcal{L} . The same route can be taken by multiple users. Let the route chosen by the i th user be \mathcal{L}_i . Suppose that if rate y_i is allocated to the traffic of the i th user, a utility of $g_i(y_i)$ is gained by the user. The traffic is called elastic when the utility function is increasing, concave and continuously differentiable function of y_i over $y_i \geq 0$. The problem is to find the optimal rate allocation for sequentially-arriving elastic traffic users to maximize the aggregate utility of rate $\sum_i g_i(y_i)$ subject to the resource capacity constraints.

III. COMPETITIVE ONLINE ALGORITHMS FOR ONUM

In this section, we present an online threshold-based algorithm for ONUM, establish sufficient conditions for the algorithm to have a bounded competitive ratio, and show the tightness of the competitive analysis. Before introducing the algorithm, we first introduce an idea that is not uncommon for algorithms that rely on the dual problems. In the algorithmic design based on an optimization problem, dual variables are usually viewed as the prices, reflecting the system state. One standard distributed algorithm to solve the classic NUM problem [27] is based on this idea, where each link is attributed a dual price λ_ℓ and user i determines its rate y_i by solving the following maximization problem:

$$y_i^*(\lambda^i) = \arg \max_{y_i \geq 0} (g_i(y_i) - \lambda^i y_i), \forall i, \quad (2)$$

where $\lambda^i = \sum_{\ell \in \mathcal{L}_i} \lambda_\ell$ is the total price on the path of user i . The dual prices of links will be updated as follows:

$$\lambda_\ell(t+1) = \left[\lambda_\ell(t) - \gamma \left(c_\ell - \sum_{i: \ell \in \mathcal{L}_i} y_i^*(\lambda^i(t)) \right) \right]^+, \forall \ell, \quad (3)$$

where t is the iteration index and γ is the step size. By iterating between (2) and (3), y_i^* and λ_ℓ converge to the optimal rate allocation and dual prices [27]. In the design of online algorithms, we also aim to determine the online rate allocation based on equation (2). However, in the online setting, the rate allocation is irrevocable, making it impossible to iteratively update the prices based on equation (3). Thus, the key is to design an approach to update prices in an online manner. In our online algorithm, we design the link dual price as a function of the link utilization level and introduce the following definition:

Definition 1 (Value Function) A value function for link ℓ , $\phi_\ell(\omega_\ell) : [0, 1] \rightarrow \mathbb{R}^+$, is a non-decreasing continuous function

that evaluates the marginal utility of the resource at the utilization level ω_ℓ .

Contrary to equation (3), where the dual price of each link is determined based on the total utilization of the link, the value function provides a way of estimating the link price based on the currently observed utilization level. In general, our algorithm takes the value function $\phi := \{\phi_\ell\}_{\ell \in \mathcal{L}}$ as input and sequentially determines the rate allocation y_i for each user upon observing its request tuple A_i . Next, we present our proposed algorithm in Algorithm 1 and term it as the online algorithm with value function ϕ (OA_ϕ). In detail, OA_ϕ uses the value function $\phi_\ell(\cdot)$ to evaluate the scarcity of the remaining capacity of link ℓ by the link price. We denote by ω_ℓ^i the total capacity of link ℓ consumed by the previous i users. The cost of using an infinitesimal amount of capacity, ds , of link ℓ when its utilization is s can be estimated by $\phi_\ell(s)ds$, and thus the cost of using link ℓ by user i can be shown as $\int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1} + y_i} \phi_\ell(s)ds$. OA_ϕ then determines y_i by solving a pseudo-utility maximization problem (4), where the user's pseudo-utility is defined as its utility of being allocated rate y_i , i.e., $g_i(y_i)$, minus the total cost of using links. Since all the design space of OA_ϕ lies in the value function, the following part of this section will be centered around the key question: what conditions should ϕ follow such that OA_ϕ can yield a bounded competitive ratio?

Algorithm 1 Online Algorithm with Value Functions ϕ (OA_ϕ)

Initialize: value function ϕ , initial utilization $\omega_\ell^0 = 0, \forall \ell$;
for the i th user **do**
 Observe user i 's request $A_i = \{g_i(\cdot), \mathcal{L}_i, b_i\}$;
 Determine y_i by solving the problem

$$y_i = \arg \max_{0 \leq y \leq b_i} g_i(y) - \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1} + y} \phi_\ell(s)ds; \quad (4)$$

Update for links $\ell \in \mathcal{L}_i$: $\omega_\ell^i = \omega_\ell^{i-1} + y_i$;
end for

A. Sufficient Conditions for Being Competitive

The following theorem provides a sufficient condition on the value function ϕ to ensure OA_ϕ to be competitive.

Theorem 1 OA_ϕ is $L\alpha$ -competitive if ϕ_ℓ is given by

$$\phi_\ell(y) = \begin{cases} m, & y \in [0, \beta], \\ \varphi_\ell(y), & y \in [\beta, 1], \\ \infty, & y \in (1, \infty), \end{cases} \quad (5)$$

where

(i) $\alpha \geq \ln(M/m) + 1$. $\beta \in [0, 1]$ is a utilization threshold and satisfies

$$\beta \geq 1/\alpha, \quad (6)$$

(ii) φ_ℓ is a non-decreasing function that satisfies

$$\begin{cases} \varphi_\ell(y) \geq \frac{1}{\alpha} \varphi_\ell'(y), y \in [\beta, 1], \\ \varphi_\ell(\beta) = m, \varphi_\ell(1) = M. \end{cases} \quad (7)$$

Before the detailed proof, we provide intuitions on the sufficient conditions above. We note that when all users request for the same single link ℓ with $g'_i(y) = m$, i.e., all users are with sufficiently low utility, if $\phi_\ell(0) > m$, all of them will be rejected. To ensure a bounded competitive ratio for this case, the value function needs a flat segment with $\phi_\ell(y) = m$ at the beginning. After that, when the remaining resource decreases, the link price increases so as to preserve capacity for possible high-utility users. The exponential form is the same as the single link case in our previous work [28]. We next show the proof of Theorem 1.

Proof: Because OA_ϕ is deterministic, for any arrival instance \mathcal{I} , OA_ϕ will produce a deterministic final utilization level for each link. Also, different arrival instances can induce the same set of final link utilization levels. Denote the final utilization level of link ℓ by ω_ℓ^N , i.e., $\omega_\ell^N = \sum_{i:\ell \in \mathcal{L}_i} y_i$. We are then able to classify the set of all possible arrival instances \mathcal{I} into 3 cases according to their induced final utilization levels:

- **Case 1:** $\forall \ell \in \mathcal{L}, \omega_\ell^N < \beta$.
- **Case 2:** $\forall \ell \in \mathcal{L}, \omega_\ell^N < 1$; $\exists \ell \in \mathcal{L}, \omega_\ell^N \geq \beta$.
- **Case 3:** $\exists \ell \in \mathcal{L}, \omega_\ell^N = 1$.

In Case 1, we make the observation that any user will transmit at its maximal rate b_i . Because for any arrival i , $\omega_\ell^i \leq \omega_\ell^N < \beta$, $\phi(\omega_\ell^i) = m \leq g'_i(b_i) \leq g'_i(y)$, then $g_i(y) - \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1} + y} \phi_\ell(s) ds$ is non-decreasing in y over $[0, b_i]$ and $y_i = b_i$ is the maximizing solution. In this case, the offline optimal decisions are exactly the same as those made by the online algorithm, and thus the offline optimal utility is equal to the online utility, i.e., $\text{OPT}(\mathcal{I}) = \text{ALG}(\mathcal{I}) \leq L\alpha \text{ALG}(\mathcal{I})$, where both the number of links L and parameter α are no less than 1.

In Case 2, there exists a link whose final utilization level exceeds β , which makes the exponential segment take effect. Moreover, no link has reached the capacity limit in Case 2. Given the final utilization $\{\omega_\ell^N\}_{\ell \in \mathcal{L}}$, we next find the worst case of \mathcal{I} , for which the ratio $\frac{\text{OPT}(\mathcal{I})}{\text{ALG}(\mathcal{I})}$ is maximized over all possible instances in Case 2.

For an arrival instance \mathcal{I} , denote the last user with $y_i > 0$ by τ . We group the users that come before τ into 4 sets, based on their utility functions and requested link set $\{g_i(\cdot), \mathcal{L}_i\}$. Let \mathcal{N}_h denote the set of *high-valuation* users whose minimal marginal utility $g'_i(b_i) \geq \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N)$, and thus for $i \in \mathcal{N}_h$, $y_i = b_i$; Let \mathcal{N}_m denote the set of *medium-valuation* users that $\exists y \in (0, b_i)$ such that $g'_i(y) = \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N)$, and thus for $i \in \mathcal{N}_m$, $y_i > 0$; Let \mathcal{N}_l denote the set of *low-valuation* users whose largest marginal utility $g'_i(0) \leq \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N)$ and $g'_i(b_i) < \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^i) < g'_i(0)$, and thus for $i \in \mathcal{N}_l$, $y_i > 0$; Let \mathcal{N}_z denote the set of *very-low-valuation* users that $g'_i(0) \leq \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^i) \leq \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N)$, and thus $y_i = 0$ for $i \in \mathcal{N}_z$. Note that either of the aforementioned four user sets is an empty set if there do not exist such users in the arrival instance \mathcal{I} . We use \tilde{y}_i to denote the offline optimal rate allocation decision for user i , and define \hat{y}_i as the value such that $g'_i(\hat{y}_i) = \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N)$ for $i \in \mathcal{N}_m$. The offline optimal

total utility for any \mathcal{I} is

$$\begin{aligned} \text{OPT}(\mathcal{I}) &= \sum_{i=1}^{\tau} g_i(\tilde{y}_i) + \sum_{i=\tau+1}^N g_i(\tilde{y}_i) \\ &\stackrel{(i)}{=} \sum_{i \in \mathcal{N}_h} g_i(b_i) + \sum_{i \in \mathcal{N}_m} g_i(\tilde{y}_i) + \sum_{i \in \mathcal{N}_l} g_i(\tilde{y}_i) \\ &\quad + \sum_{i \in \mathcal{N}_z} g_i(\tilde{y}_i) + \sum_{i=\tau+1}^N \int_0^{\tilde{y}_i} g'_i(s) ds \\ &\stackrel{(ii)}{\leq} \sum_{i \in \mathcal{N}_h} g_i(b_i) + \sum_{i \in \mathcal{N}_m} [g_i(\hat{y}_i) + \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N)(\tilde{y}_i - \hat{y}_i)] \\ &\quad + \sum_{i \in \mathcal{N}_l \cup \mathcal{N}_z} \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N) \tilde{y}_i + \sum_{i=\tau+1}^N \int_0^{\tilde{y}_i} g'_i(s) ds \\ &\stackrel{(iii)}{\leq} \sum_{i \in \mathcal{N}_h} [g_i(b_i) - \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N) b_i] \\ &\quad + \sum_{i \in \mathcal{N}_m} \left[g_i(\hat{y}_i) - \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N) \hat{y}_i \right] + \sum_{\ell \in \mathcal{L}} \phi_\ell(\omega_\ell^N). \end{aligned} \quad (8)$$

Equality (i) holds due to the observations that, for high-valuation users $i \in \mathcal{N}_h$, if any, they will be allocated b_i in the offline optimal decisions. Also, with $g_i(0) = 0$, $g_i(y) = \int_0^y g'_i(s) ds$. Thus, equality (i) holds. For inequality (ii), it can be observed that the marginal utility of any user in \mathcal{N}_l and \mathcal{N}_z (if any) is bounded by $\sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N)$ and the marginal utility when allocating more than \hat{y}_i to user i in \mathcal{N}_m (if any) is also bounded by $\sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N)$. Thus, for $i \in \mathcal{N}_l \cup \mathcal{N}_z$, $g_i(\tilde{y}_i) \leq \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N) \tilde{y}_i$; for $i \in \mathcal{N}_m$, $g_i(\tilde{y}_i) - g_i(\hat{y}_i) \leq \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N)(\tilde{y}_i - \hat{y}_i)$. Inequality (ii) then holds. The correctness of inequality (iii) is based on the following observations:

- For users coming after τ , if any, they will be allocated nothing in the online algorithm ($y_i = 0, i > \tau$), and thus their marginal utility functions shall satisfy $g'_i(y) \leq \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^{i-1}) = \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N)$.
- The offline optimal utility reaches the largest if the users have enough budgets to occupy all the capacity left in each link and the utility functions of those users coming after τ are the largest possible $g'_i(y) = \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N) - \epsilon$, where ϵ is small.

The total utility of the online algorithm under the same request sequence \mathcal{I} is

$$\begin{aligned} \text{ALG}(\mathcal{I}) &= \sum_{i=1}^N g_i(y_i) = \sum_{i=1}^{\tau} g_i(y_i) \\ &= \sum_{i=1}^{\tau} \left[g_i(y_i) - \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^i} \phi_\ell(s) ds \right] \\ &\quad + \sum_{i=1}^{\tau} \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^i} \phi_\ell(s) ds. \end{aligned} \quad (9)$$

Let $\Delta_i = g_i(y_i) - \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^i} \phi_\ell(s) ds$. Thus, Δ_i is the optimal objective value for (4) and $\Delta_i \geq 0$. Define $\hat{\Delta}_i$ as

follows: For $i \in \mathcal{N}_h$, $\hat{\Delta}_i = g_i(b_i) - \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N) b_i$; for $i \in \mathcal{N}_m$, $\hat{\Delta}_i = g_i(\hat{y}_i) - \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N) \hat{y}_i$; for $i \in \mathcal{N}_l \cup \mathcal{N}_z$, $\hat{\Delta}_i = 0$. Combining (8) and (9), we have

$$\begin{aligned} \frac{\text{OPT}(\mathcal{I})}{\text{ALG}(\mathcal{I})} &\leq \frac{\sum_{i=1}^{\tau} \hat{\Delta}_i + \sum_{\ell \in \mathcal{L}} \phi_\ell(\omega_\ell^N)}{\sum_{i=1}^{\tau} \Delta_i + \sum_{i=1}^{\tau} \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^N} \phi_\ell(s) ds} \\ &\stackrel{(iv)}{\leq} \frac{\sum_{\ell \in \mathcal{L}} \phi_\ell(\omega_\ell^N)}{\sum_{i=1}^{\tau} \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^N} \phi_\ell(s) ds} \\ &= \frac{\sum_{\ell \in \mathcal{L}} \phi_\ell(\omega_\ell^N)}{\sum_{\ell \in \mathcal{L}} \int_0^{\omega_\ell^N} \phi_\ell(s) ds}. \end{aligned} \quad (10)$$

To show (iv) holds, let $h_i(\lambda)$ be the conjugate function of $g_i(y)$, i.e., $h_i(\lambda) = \max_{0 \leq y \leq b_i} (g_i(y) - \lambda y)$. It is easy to verify that $h_i(\lambda)$ is non-increasing in λ . Notice that when $y_i > 0$, i.e., $i \in \mathcal{N}_h \cup \mathcal{N}_m \cup \mathcal{N}_l$,

$$\begin{aligned} \hat{\Delta}_i &\stackrel{(v)}{=} h_i\left(\sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^N)\right) \stackrel{(vi)}{\leq} h_i\left(\sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^i)\right) \\ &\stackrel{(vii)}{=} \max_{0 \leq y \leq b_i} (g_i(y) - \sum_{\ell \in \mathcal{L}_i} \phi_\ell(\omega_\ell^i) y) \\ &\stackrel{(viii)}{\leq} \max_{0 \leq y \leq b_i} (g_i(y) - \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1}+y} \phi_\ell(s) ds) \stackrel{(ix)}{=} \Delta_i \end{aligned}$$

where (v) can be verified based on definitions of $\hat{\Delta}_i$ and $h_i(\cdot)$; (vii) and (ix) are based on definitions of $h_i(\cdot)$ and Δ_i ; (vi) and (viii) hold because $\phi_\ell(\omega)$ is increasing in ω and $h_i(\lambda)$ is non-increasing in λ . When $i \in \mathcal{N}_z$, $\Delta_i = \hat{\Delta}_i = 0$. Thus, for any $i \leq \tau$, $\hat{\Delta}_i \leq \Delta_i$, inequality (iv) holds.

Denote the set of links where $\omega_\ell^N < \beta$ by \mathcal{L}^1 and the links where $\beta \leq \omega_\ell^N < 1$ by \mathcal{L}^2 , we can express (10) as the following:

$$\frac{\text{OPT}(\mathcal{I})}{\text{ALG}(\mathcal{I})} \leq \frac{\sum_{\ell \in \mathcal{L}^1} m + \sum_{\ell \in \mathcal{L}^2} \phi_\ell(\omega_\ell^N)}{\sum_{\ell \in \mathcal{L}^1} m \omega_\ell^N + \sum_{\ell \in \mathcal{L}^2} \int_0^{\omega_\ell^N} \phi_\ell(s) ds}. \quad (11)$$

By the sufficient conditions (6) and (7), when $\omega_\ell^N \in [\beta, 1]$, we have

$$\int_0^{\omega_\ell^N} \phi_\ell(s) ds \geq \frac{1}{\alpha} \phi_\ell(\omega_\ell^N). \quad (12)$$

Thus, (11) can be further upper bounded as follows:

$$\frac{\text{OPT}(\mathcal{I})}{\text{ALG}(\mathcal{I})} \leq \frac{\sum_{\ell \in \mathcal{L}^1} m + \sum_{\ell \in \mathcal{L}^2} \phi_\ell(\omega_\ell^N)}{\sum_{\ell \in \mathcal{L}^1} m \omega_\ell^N + \sum_{\ell \in \mathcal{L}^2} \frac{1}{\alpha} \phi_\ell(\omega_\ell^N)}. \quad (13)$$

From (13), the upper bound depends on ω_ℓ^N , and we have

$$\begin{aligned} \frac{\text{OPT}(\mathcal{I})}{\text{ALG}(\mathcal{I})} &\leq \frac{\sum_{\ell \in \mathcal{L}^1} m + \sum_{\ell \in \mathcal{L}^2} \phi_\ell(\omega_\ell^N)}{\sum_{\ell \in \mathcal{L}^2} \frac{1}{\alpha} \phi_\ell(\omega_\ell^N)} \\ &\leq \alpha \left(\frac{|\mathcal{L}^1|}{|\mathcal{L}^2|} + 1 \right) \\ &\leq |\mathcal{L}| \alpha = L\alpha. \end{aligned}$$

From the above inequalities, we observe that, in the worst case, $\omega_\ell^N = 0$ for $\ell \in \mathcal{L}^1$ or $|\mathcal{L}^1| = 0$, and there is only one link in \mathcal{L}^2 with $\phi_\ell(\omega_\ell^N) = m$. In summary, the worst case of arrivals in Case 2 can just contain two arrivals. The first

user will only request one single link ℓ with utility satisfying $g'_1(y) = m$, followed by the second user requesting all links with utility satisfying $g'_2(y) = m|\mathcal{L}|$. Both of their budgets are greater than β . The online algorithm will only accept the first user and allocate β of link ℓ to him, producing a total utility of βm . The offline optimal will accept the second user with the total utility being $m|\mathcal{L}|$.

In Case 3, there exist links with the capacity limit reached. \mathcal{L}^1 and \mathcal{L}^2 are defined in the same way as that in Case 2 and the links with $\omega_\ell^N = 1$ are denoted by \mathcal{L}^3 . We do not need to group users by their utility functions as in Case 2, because for any user i , $g'_i(y) \leq |\mathcal{L}_i|M$, and each link carries a capacity limit of 1, we can directly bound the offline optimal as follows:

$$\text{OPT}(\mathcal{I}) \leq |\mathcal{L}|M.$$

The utility of the online algorithm is lower bounded as follows:

$$\begin{aligned} \text{ALG}(\mathcal{I}) &\geq \sum_{\ell \in \mathcal{L}^1} m \omega_\ell^N + \sum_{\ell \in \mathcal{L}^2 \cup \mathcal{L}^3} \int_0^{\omega_\ell^N} \phi_\ell(s) ds \\ &\stackrel{(x)}{\geq} \sum_{\ell \in \mathcal{L}^1} m \omega_\ell^N + \frac{1}{\alpha} \sum_{\ell \in \mathcal{L}^2} \phi_\ell(\omega_\ell^N) + \frac{1}{\alpha} \sum_{\ell \in \mathcal{L}^3} M, \end{aligned}$$

where (x) follows from equation (12). Then we have

$$\begin{aligned} \frac{\text{OPT}(\mathcal{I})}{\text{ALG}(\mathcal{I})} &\leq \frac{|\mathcal{L}|M}{\sum_{\ell \in \mathcal{L}^1} m \omega_\ell^N + \frac{1}{\alpha} \sum_{\ell \in \mathcal{L}^2} \phi_\ell(\omega_\ell^N) + \frac{1}{\alpha} \sum_{\ell \in \mathcal{L}^3} M} \\ &\stackrel{(xi)}{\leq} \alpha |\mathcal{L}| = L\alpha. \end{aligned}$$

We observe that for inequality (xi) to be equal, we must have $|\mathcal{L}^1| = 0$ and $|\mathcal{L}^2| = 0$.

Thus, combining results in Cases 1-3, the online algorithm we proposed is $L\alpha$ -competitive, linear in the number of links. ■

In the sequel, we show that our analysis above is tight by explicitly presenting the worst-case instances for the algorithm to reach the competitive ratio claimed. We have presented a special worst-case instance with two arrivals for Case 2 in the proof of Theorem 1. A more general property of the worst-case instances for OA_ϕ is shown in the following lemma.

Lemma 2 Any worst-case instance for OA_ϕ must possess the property: for each arrival $i < \tau$, $g_i(y_i) = \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^N} \phi_\ell(s) ds + \epsilon$.

Proof: Notice that the inequality (iv) in equation (10) becomes equal only if the arrival instance \mathcal{I} owns the property above. We prove the lemma by constructing a general arrival sequence \mathcal{I} such that $\frac{\text{OPT}(\mathcal{I})}{\text{ALG}(\mathcal{I})} = L\alpha$.

The arrival instance is constructed as follows: there comes a group of t_1 arrivals requesting a specific link ℓ , whose utility functions satisfy

$$\forall y \in [0, b_i], g'_i(y)/|\mathcal{L}_i| = g'_i(y) = m,$$

and the budgets satisfy $b_i = \frac{1}{\alpha t_1}$, $i = 1, \dots, t_1$; followed by a second group of t_2 arrivals requesting the same link ℓ , whose utility functions satisfy

$$\exists y_i \text{ close to } 0, g'_i(y_i) = g'_{i-1}(y_i) + \frac{\epsilon}{t_2},$$

and the budgets $b_i = 1$, $i = t_1 + 1, \dots, t_1 + t_2$; at last, an arrival comes and requests all the links, whose utility function satisfies

$$g'_{t_1+t_2+1}(y) = mL + \frac{\epsilon}{2},$$

and $b_{t_1+t_2+1} = 1$.

For the first group of arrivals, the online algorithm will allocate rate $y_i = b_i$ and the utilization of link ℓ reaches $\frac{1}{\alpha}$ before the second group. In the sequel, for arrival $i = t_1 + 1$, the online algorithm will allocate rate

$$y_i = \phi_\ell^{-1}\left(m + \frac{(i - t_1)\epsilon}{t_2}\right) - \frac{1}{\alpha}.$$

For arrivals $i = t_1 + 2, \dots, t_1 + t_2$, the online algorithm will allocate rate

$$\begin{aligned} y_i &= \phi_\ell^{-1}\left(m + \frac{(i - t_1)\epsilon}{t_2}\right) - \phi_\ell^{-1}\left(m + \frac{(i - t_1 - 1)\epsilon}{t_2}\right) \\ &\approx \phi_\ell'^{-1}\left(m + \frac{(i - t_1)\epsilon}{t_2}\right) \frac{\epsilon}{t_2}. \end{aligned}$$

For arrival $i = t_1 + t_2 + 1$, namely, the last arrival comes, we have

$$\sum_{\ell \in \mathcal{L}} \phi_\ell(\omega_\ell^{t_1+t_2}) = mL + \epsilon > g'_{t_1+t_2+1}(0),$$

and thus the solution to (4) is $y_{t_1+t_2+1} = 0$, i.e., the online algorithm will reject the last arrival.

However, as long as the number of links $L > 1$, in terms of the capacity allocation of the frequently requested link ℓ , rates assigned to the last arrival obviously yields more utility than those assigned to the first two groups since the last arrival requests all the links. Thus, the offline optimal will reject the first two groups and allocate everything to the last arrival. This being the case, we have

$$\begin{aligned} \frac{\text{OPT}(\mathcal{I})}{\text{ALG}(\mathcal{I})} &= \frac{mL + \frac{\epsilon}{2}}{\sum_{i=1}^{t_1} g_i\left(\frac{1}{\alpha t_1}\right) + \sum_{i=t_1+1}^{t_1+t_2} g_i(y_i)} \\ &= \frac{mL + \frac{\epsilon}{2}}{\frac{m}{\alpha} + \sum_{i=t_1+1}^{t_1+t_2} \int_0^{y_i} g'_i(y) dy}. \end{aligned}$$

Thus, let $t_1, t_2 \rightarrow \infty, \epsilon \rightarrow 0$, we have

$$\frac{\text{OPT}(\mathcal{I})}{\text{ALG}(\mathcal{I})} \rightarrow \frac{mL}{\frac{m}{\alpha} + m \cdot \phi_\ell'^{-1}(m) \cdot 0} = L\alpha,$$

which shows that our analysis is tight. ■

IV. SIMULATION RESULTS

A. Simulation Settings

We use the network trace of the Abilene network collected from December 8, 2003 to December 28, 2003 to demonstrate the performance of our algorithm [29]. There are 11 nodes, 41 links and 110 source-destination pairs in the Abilene network. The network trace contains the traffic matrix and the routing matrix. The traffic matrix given in the data set is a real-valued matrix with dimension 2016×110 , where the component on the i th column and the j th row represents the traffic volume of the i th source-destination pair measured in the j th 5-min slot. The routing matrix is a binary matrix whose dimension

is 110×41 , where the i th row vector denotes the routing path of the corresponding source-destination pair.

The arrival order is obtained based on the traffic matrix. We order the link by the traffic measured in each 5-min slot and assume that the arrivals on a specific link in a 5-min slot are generated by a Poisson process, the mean arrival rate of which is proportional to the traffic amount of that link. Note that the Poisson assumption is not necessary to our theoretical analysis but only adopted for generating the synthetic arrival data. We obtain the requested link set of each arrival based on the routing matrix. Utilities and budgets are not included in the data set. We assume that the utility functions of arrivals are in the form of $g_i(y) = a_i |\mathcal{L}_i| \log(1 + y)$, where \mathcal{L}_i is the link set requested by the i th arrival. We consider two cases for a_i : the average case and the worst case. In the average case, a_i s are randomly generated from a truncated Gaussian distribution, $a_i \sim \text{TruncGaussian}(\mu, \sigma, LB, UB)$, where μ, σ, LB, UB are the mean, variance, lower bound and upper bound, respectively. The mean and variance are fixed for each instance and the bounds are properly set so that $\frac{g'_i(y)}{|\mathcal{L}_i|} \in [m, M]$. In the worst case, $a_i \sim \text{TruncGaussian}(\mu_i, \sigma_i, LB, UB)$, with means μ_i slowly increasing or decreasing with i . We assume that the budgets of arrivals are uniformly distributed $b_i \sim U(0, 1)$.

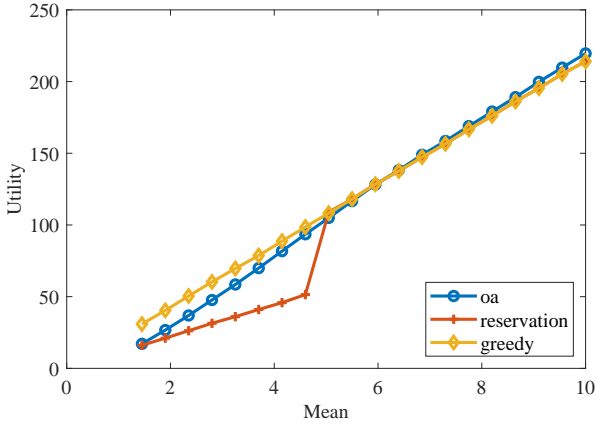
Before this work, there is no algorithm designed for our ONUM problem. Thus, we compare our proposed algorithm with two heuristics, which are greedy and reservation-based algorithm. The descriptions of the heuristics are as follows:

Greedy. $y_i = b_i$ as long as the capacity permits, otherwise $y_i = \min_{\ell \in \mathcal{L}_i} (1 - \omega_\ell^{i-1})$, where ω_ℓ^i is the utilization level of link ℓ after the i th allocation y_i is finished. The implementation of this algorithm will be referred to as **greedy** hereinafter.

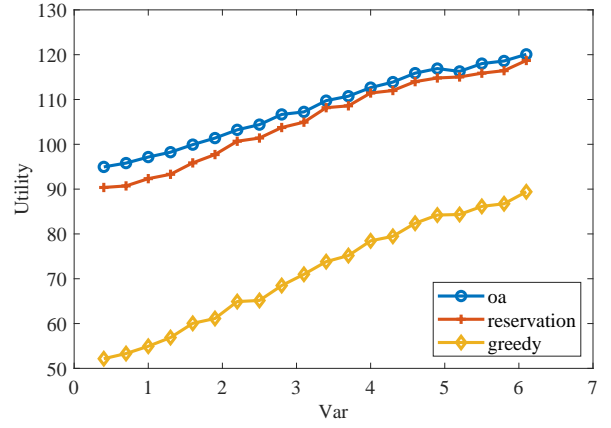
Reservation-based algorithm. Reserve p of the capacity for arrivals with $\frac{g'_i(y)}{|\mathcal{L}_i|} \geq qM$, where $p, q \in [0, 1]$ are parameters to be determined. It is plain to see that it is beneficial for high-utility arrivals if α, β are closer to 1 and beneficial for low-utility arrivals otherwise. This class of algorithms are widely used in network control problems [30]. The implementations of this algorithm will be referred to as **res** or **reservation** hereinafter.

B. Results of Plain Implementations

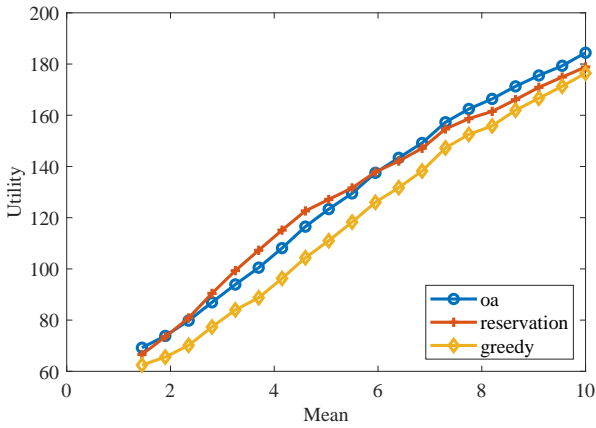
Fig. 1 shows the total utility achieved by the aforementioned three algorithms w.r.t. the mean or variance of arrival instances. From Fig. 1a, when the utilities are densely centered around a certain value, i.e., the variance is small, our algorithm performs stable with the mean while we observe a sharp twist from the reservation-based algorithm. The position of the twist in reservation-based algorithm relies badly on the parameter chosen and our algorithm avoids such instability by the virtue of the smooth price function. Fig. 1b shows the case when the utilities expose higher uncertainty, our algorithm performs the best among the three when high-utility arrivals are the mainstream. Meanwhile, when the arrivals are coming with a low utility on average, i.e., the mean is small, our algorithm is not so competitive in both cases because we are too conservative to accept enough low-utility arrivals. It leaves



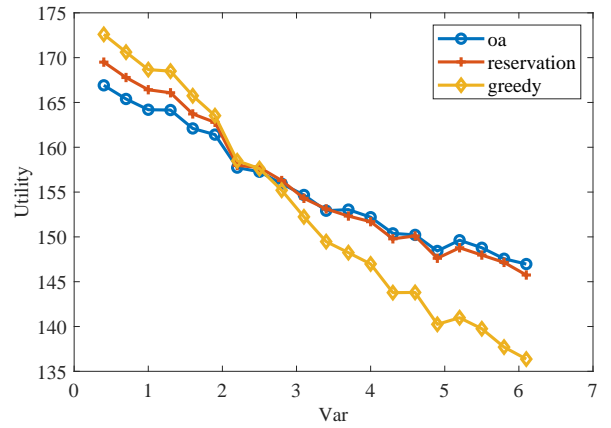
(a) Utility vs. Mean (Var=0.1)



(a) Utility vs. Var (increasing mean)



(b) Utility vs. Mean (Var=3)



(b) Utility vs. Var (decreasing mean)

Fig. 1: Performance of different algorithms with plain implementation in the average case. The utilities of arrivals are generated by a truncated Gaussian distribution. The y -axis is the average utility produced by algorithms and the x -axis is the mean of the distribution. Two sub-figures show the performance under arrivals with low and high uncertainties.

Fig. 2: Performance of different algorithms with plain implementation in the extreme case. The utilities of arrivals are either increasing or decreasing on average. The x -axis is the uncertainty of the arrivals. The larger Var, the more uncertain the utilities are. Two sub-figures show the performance when the utilities are in general in an increasing or decreasing trend.

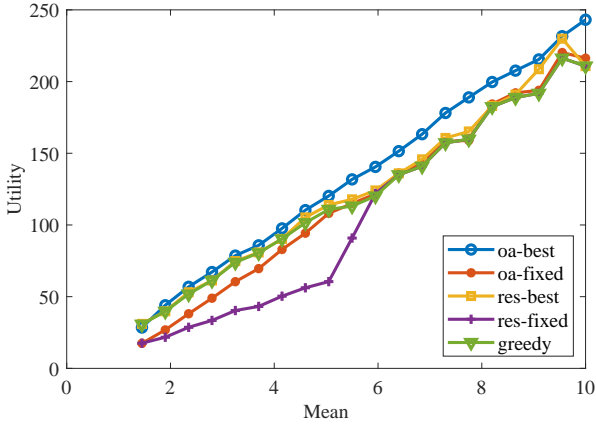
us the space to further improve our algorithm that will be shown in Section IV-C.

Fig. 2 shows the total utility of the three algorithms w.r.t. the variance while the utilities are slowly increasing/decreasing on average with time. Our algorithm always produces a higher utility than the other two no matter of the variance when the utilities are increasing, which is the hard case for online algorithm in general, because decisions must be made more conservatively so as to reserve capacity for future users. Fig. 2a validates our conjecture, where our algorithm yields the highest utility. Fig. 2b shows that, when the utilities are decreasing on average, if the utility realizations are in fact decreasing, i.e., the variance is small, it is more beneficial to be aggressive, as shown by the advantage of the greedy algorithm. Actually, all three algorithms allocate most of the capacity in the beginning; however, our algorithm is the most pessimistic because of its worst-case nature. Thus, it always waits for better deals in the future, which is impossible in this extreme case, and thus it is sub-optimal when the variance is

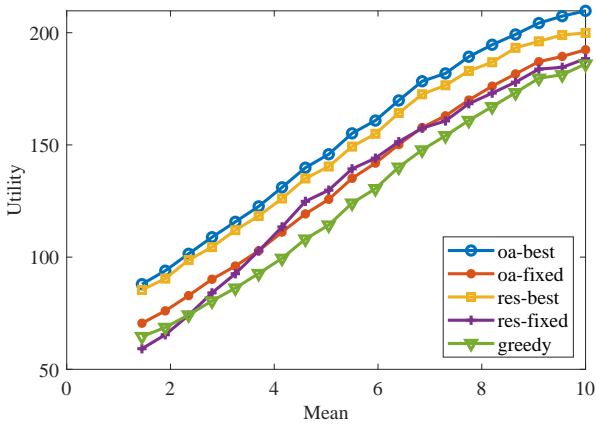
small in Fig. 2b. In the sequel, we will show that if introducing a proper parameter-choosing mechanism, our algorithm will perform the best across all possible cases.

C. Results of Adaptive Implementations

In Section IV-B, our algorithm is at a disadvantage under certain arrival instances. Actually, this is not uncommon for algorithms designed by the worst-case analysis. Those algorithms are completely agnostic to the input patterns, and thus, if the input indeed poses a pattern such as some stochasticity, algorithms designed for taking care of the worst-case scenarios will fail pathetically. However, we are going to show that with the right parameters, our algorithm is able to be both theoretically and practically competitive. To accomplish that, we need to understand what information our algorithm fails to capture in the instances. We observe that the common feature of the instances under which our algorithm is disadvantageous is that either m or M serves as a loose bound of



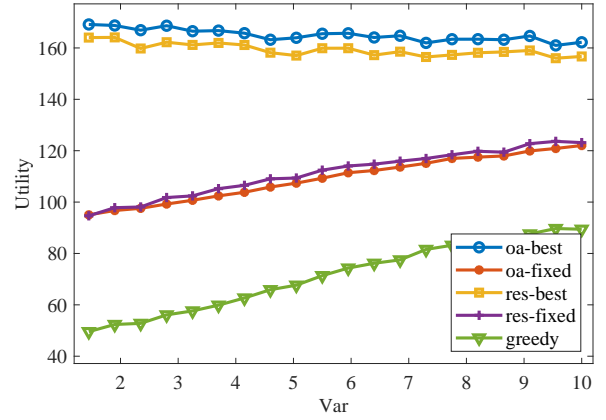
(a) Utility vs. Mean (Var=0.1)



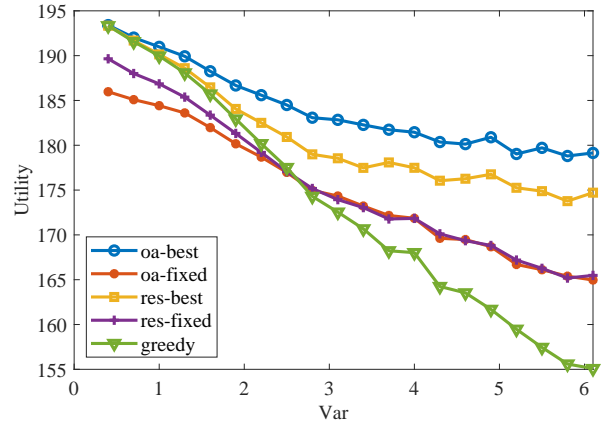
(b) Utility vs. Mean (Var=3)

Fig. 3: Performance improvement of algorithms in the average case when the parameters are chosen offline. The arrivals are generated in a similar way to Fig. 1. **Alg-best** denotes the performance with the best set of parameters and **Alg-fixed** denotes the performance with the same set of parameters in Fig. 1.

$\frac{g'_i(y)}{|\mathcal{L}_i|}$; in other words, the utility bounds that our algorithm uses in Algorithm 1 are not tight enough. A conjecture is that if such pattern is incorporated into the algorithm, our algorithm will be able to overcome its intrinsic worst-case nature and yield competitive performance even under average cases. Therefore, we first confirm our conjecture by doing a simple experiment. We replace the original (m, M) by (\tilde{m}, \tilde{M}) to indicate that they are variables to be chosen. A group of arrival instances are generated by a certain distribution. We run parallel implementations of our algorithm with different (\tilde{m}, \tilde{M}) under those arrival instances generated, and apply our algorithm with the (\tilde{m}, \tilde{M}) pair producing the most utility on average to instances newly generated by the same distribution. If our algorithm with the parameters learned offline is able to perform better over other algorithms, then our conjecture is proved. The utilities in Fig. 3 are generated from the same distribution used in Fig. 1. The possible parameter space is a convex region $\{(\tilde{m}, \tilde{M}) | m \leq \tilde{m} \leq \tilde{M} \leq M\}$. We discretize the space into grids with grid size $\frac{M-m}{20} \times \frac{M-m}{20}$. Although the



(a) Utility vs. Var (increasing mean)

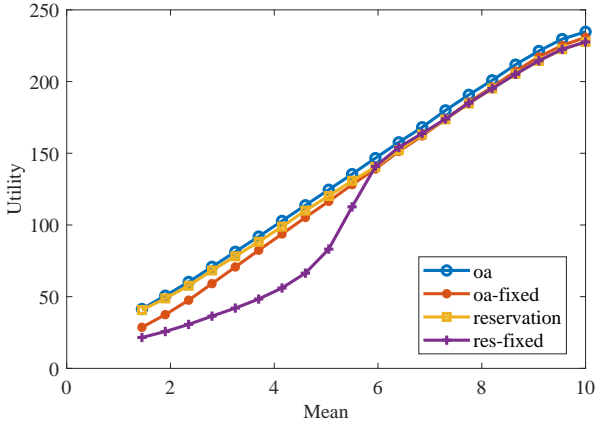


(b) Utility vs. Var (decreasing mean)

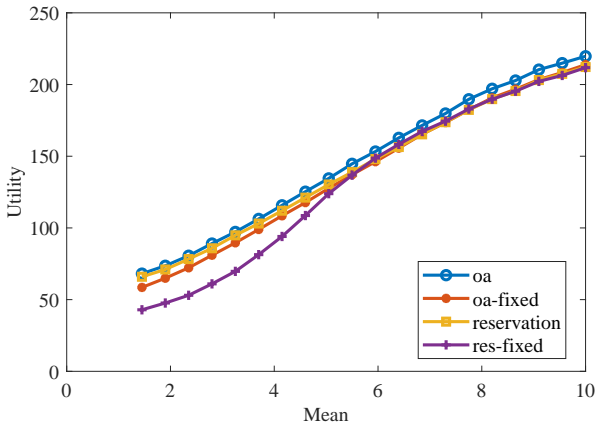
Fig. 4: Performance improvement of algorithms in the extreme case when the parameters are chosen offline. The arrivals are generated in a similar way to Fig. 2. **Alg-best** denotes the performance with the best set of parameters and **Alg-fixed** denotes the performance with the same set of parameters in Fig. 2.

discretization can lead to suboptimality in the parameters, a smaller search space enables a faster exhaustive search. Since we are expecting to observe the performance gain from a better parameter choice instead of optimizing the parameters, the upside dominates the downside. As shown in Fig. 3 and Fig. 4, if parameters (\tilde{m}, \tilde{M}) are chosen offline by exhaustive search, our algorithm yields the most utility among all candidate algorithms.

Though the method above is effective, we may not have enough offline data to learn the parameters accurately. An alternate idea is to adaptively tune m and M as the algorithm runs. One way is to model the tuning process as a multi-armed bandit problem and treat different (\tilde{m}, \tilde{M}) as potential actions. The expected reward of action (\tilde{m}, \tilde{M}) is the utility gained by choosing (\tilde{m}, \tilde{M}) as the parameters in the value function $\phi_\ell(y)$ in one episode. The algorithm then chooses arm (\tilde{m}, \tilde{M}) based on the utility observed. However, for our problem, full feedback is possible if parallel computing is employed. Thus, we model the tuning process as an expert



(a) Utility vs. Mean (Var=0.1)



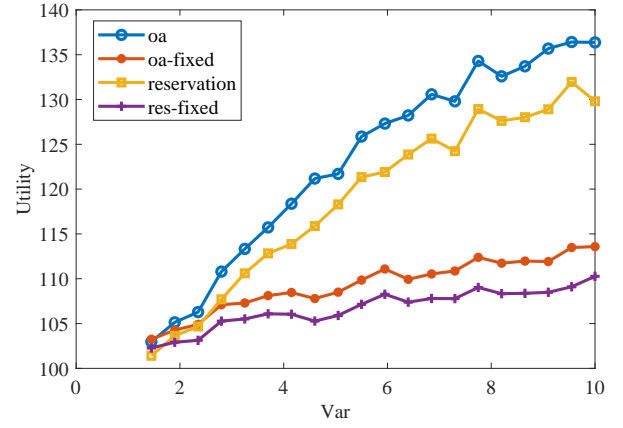
(b) Utility vs. Mean (Var=3)

Fig. 5: Performance improvement of algorithms with learned parameters in the average case. The arrivals are generated in a similar way to Fig.1. **Alg** denotes the performance with the learned parameters and **Alg-fixed** denotes the performance with the same set of parameters in Fig. 1.

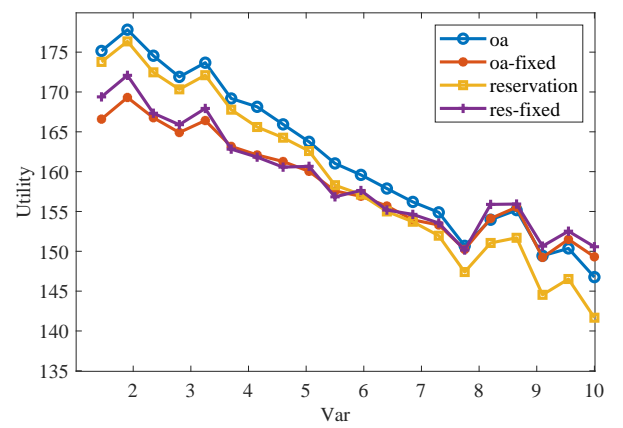
problem instead, which enjoys a faster convergence. We adopt the exponential weight algorithm proposed in [31] and show the complete algorithm in Algorithm 2.

The action space is the same as the discretized parameter space mentioned before. In Algorithm 2, each episode consists of arrivals in 24 hours, which are generated by 288 rows in the traffic matrix, since each row represents the traffic amount measured in 5 min. We denote by e_i the number of arrivals in the i th episode. We set $E = 30$ and $\eta = 10$ in the simulation. We show that the adaptive implementation indeed improves the performance under the previous disadvantageous instances.

For a fair comparison, we also implement the adaptive version for the reservation-based algorithm. Fig. 5 and Fig. 6 show the utilities of our algorithm and its adaptive version, the utilities of reservation-based algorithm and its adaptive version, under the average cases and the extreme cases. We see that the adaptive implementation always improves the performance of both Algorithm 1 and the reservation-based compared to the fixed parameter chosen in Section IV-B. It is due to that in the adaptive implementation, the parameters are



(a) Utility vs. Var (increasing mean)



(b) Utility vs. Var (decreasing mean)

Fig. 6: Performance improvement of algorithms with learned parameters in the extreme case. The arrivals are generated in a similar way to Fig.2. **Alg** denotes the performance with the learned parameters and **Alg-fixed** denotes the performance with the same set of parameters in Fig. 2.

dynamically chosen in accordance with the input pattern. Fig. 7 shows the convergence of the average utility produced by Algorithm 2 to that produced by the static optimal parameter pair with the increase of episodes. Compared with the previous method of choosing the parameter offline based on past traces, the online learning algorithm does not need prior information on the input or past history, and thus can be applied in wider application scenarios.

V. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we consider the online network utility maximization problem, develop an algorithm that makes the allocation based on the utilization levels, and give a tight competitive analysis of the proposed algorithm. We find that the competitive ratio is linear in the number of links in the network. Extensive trace-driven simulations are conducted to show the empirical performance of the proposed algorithm. We confirm that the algorithm suffers from the common weakness for algorithms designed with the worst-case performance guarantee, whose performance under typical cases is usually mediocre.

Algorithm 2 Online Adaptive Implementation of Algorithm 1

- 1: **Initialize:** Initial utilization $\omega_\ell^0 = 0$. Weights $w_{(\tilde{m}, \tilde{M})} = w_0, (\tilde{m}, \tilde{M}) \in \mathbb{A}$, initial probability of choosing arm (\tilde{m}, \tilde{M}) is $p_{(\tilde{m}, \tilde{M})} = w_0 / (\sum_{(\tilde{m}, \tilde{M})} w_0) = 1/|\mathbb{A}|$. Episode length e_i of episode i .
- 2: **for** the i th episode **do**
- 3: Choose $(\tilde{m}_i, \tilde{M}_i)$ according to the probability distribution \mathbf{p} .
- 4: Initialize the link utilization levels to 0.
- 5: **for** the t th arrival ($t = (i-1)e_i + 1 : ie_i$) **do**
- 6: Observe $A_i = \{g_i(\cdot), \mathcal{L}_i, b_i\}$.
- 7: Determine y_i by solving problem (4) with $(\tilde{m}_i, \tilde{M}_i)$ replacing (m, M) in ϕ and accumulate utility $u_{(\tilde{m}, \tilde{M})} = u_{(\tilde{m}, \tilde{M})} + g_i(y_i)$.
- 8: Update the link utilization levels: $\omega_\ell^t = \omega_\ell^{t-1} + y_i, \ell \in \mathcal{L}_i$.
- 9: **end for**
- 10: Simultaneously observe the utilities produced by the other arms $\mathbf{u} = (u_{(\tilde{m}, \tilde{M})})_{(\tilde{m}, \tilde{M}) \in \mathbb{A}}$ by repeating lines 4-9.
- 11: Update the weight of arm $(\tilde{m}, \tilde{M}), \forall (\tilde{m}, \tilde{M}) \in \mathbb{A}$:

$$w_{(\tilde{m}, \tilde{M})} \leftarrow w_{(\tilde{m}, \tilde{M})} \exp \left(\frac{\eta(u_{(\tilde{m}, \tilde{M})} - u^*)}{u^*} \right),$$

where $u^* = \max_{(\tilde{m}, \tilde{M}) \in \mathbb{A}} u_{(\tilde{m}, \tilde{M})}$.

- 12: Update the probability of choosing arm (\tilde{m}, \tilde{M}) :

$$p_{(\tilde{m}, \tilde{M})} = w_{(\tilde{m}, \tilde{M})} / \sum_{(\tilde{m}, \tilde{M}) \in \mathbb{A}} w_{(\tilde{m}, \tilde{M})}.$$

- 13: **end for**

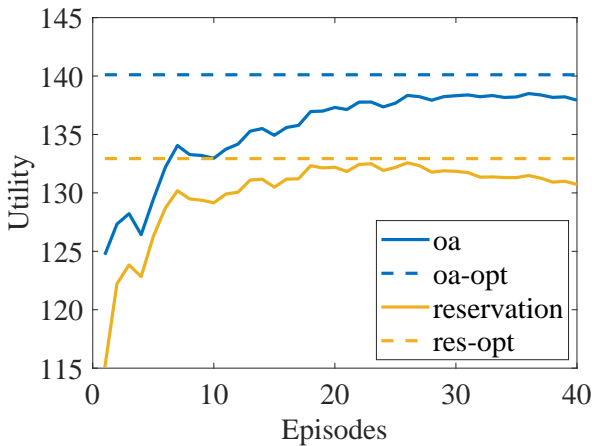


Fig. 7: Convergence of the learning algorithm. The y -axis is the average value of 100 independent runs of utility produced by Algorithm 2. **Alg-opt** is the utility of the algorithm with the best set of parameters. **Alg** is the performance of the algorithm with learned parameters. The arrivals generating this figure are from the truncated Gaussian distribution with mean $= \frac{m+M}{2}$, var = 1.

To further improve the performance, we first demonstrate the performance improvement when the algorithm parameters are chosen offline. Then we cast the parameter selection process as an online learning problem and show the performance gain by applying an off-the-shelf learning algorithm.

The ONUM problem is a relatively new problem for online algorithm design. We list some of the promising future directions that could be explored. First, we consider homogeneous capacities in this paper for the technical simplicity. It is more practical and intriguing to deal with heterogeneous capacities. We speculate that it is doable by a more fine-grained analysis. Second, the competitive ratio of our proposed algorithm is linear in terms of the number of links, but we make the conjecture that it is possible to further improve the competitive ratio by redesigning the threshold function and making more assumptions, such as utilizing the estimate of the demand information in [23]. Third, it is interesting to look for other real-life applications that can be modeled by the ONUM.

REFERENCES

- [1] F. P. Kelly, A. K. Maulloo, and D. K. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research society*, vol. 49, no. 3, pp. 237–252, 1998.
- [2] Q. Pham and W. Hwang, "Network utility maximization-based congestion control over wireless networks: A survey and potential directives," *IEEE Communications Surveys Tutorials*, vol. 19, no. 2, pp. 1173–1200, 2017.
- [3] J.-W. Lee, M. Chiang, and R. Calderbank, "Optimal mac design based on utility maximization: Reverse and forward engineering," in *Proc. IEEE Infocom*, 2006, pp. 1–13.
- [4] E. Meshkova, J. Riihijärvi, A. Achtzehn, and P. Mähönen, "On utility-based network management," in *2010 IEEE Globecom Workshops*, 2010, pp. 600–605.
- [5] N. Li, L. Chen, and S. H. Low, "Optimal demand response based on utility maximization in power networks," in *2011 IEEE power and energy society general meeting*. IEEE, 2011, pp. 1–8.
- [6] J. Rivera and H. Jacobsen, "A distributed anytime algorithm for network utility maximization with application to real-time ev charging control," in *53rd IEEE Conference on Decision and Control*, 2014, pp. 947–952.
- [7] M. Xing, J. He, and L. Cai, "Utility maximization for multimedia data dissemination in large-scale vanets," *IEEE Transactions on Mobile Computing*, vol. 16, no. 4, pp. 1188–1198, 2017.
- [8] M. H. Hajiesmaili, A. Khonsari, A. Sehati, and M. S. Talebi, "Content-aware rate allocation for efficient video streaming via dynamic network utility maximization," *Journal of Network and Computer Applications*, vol. 35, no. 6, pp. 2016 – 2027, 2012. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1084804512001762>
- [9] J. Verdyck, C. Blondia, and M. Moonen, "Network utility maximization for adaptive resource allocation in dsl systems," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 787–791.
- [10] J. Hao, R. Wang, Y. Zhuang, and B. Zhang, "A flexible network utility optimization approach for energy harvesting sensor networks," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 206–212.
- [11] J. Chen, W. Xu, S. He, Y. Sun, P. Thulasiraman, and X. Shen, "Utility-based asynchronous flow control algorithm for wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 7, pp. 1116–1126, 2010.
- [12] H. Lu, S. Balseiro, and V. Mirrokni, "Dual mirror descent for online allocation problems," *arXiv preprint arXiv:2002.10421*, 2020.
- [13] S. Agrawal and N. R. Devanur, "Fast algorithms for online stochastic convex programming," in *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, 2014, pp. 1405–1424.
- [14] N. Buchbinder and J. Naor, "Online primal-dual algorithms for covering and packing," *Mathematics of Operations Research*, vol. 34, no. 2, pp. 270–286, 2009.
- [15] S. Agrawal, Z. Wang, and Y. Ye, "A dynamic near-optimal algorithm for online linear programming," *Operations Research*, vol. 62, no. 4, pp. 876–890, 2014.

- [16] Y. Azar, N. Buchbinder, T. H. Chan, S. Chen, I. R. Cohen, A. Gupta, Z. Huang, N. Kang, V. Nagarajan, J. Naor, and D. Panigrahi, "Online algorithms for covering and packing problems with convex objectives," in *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, 2016, pp. 148–157.
- [17] Z. Huang and A. Kim, "Welfare maximization with production costs: A primal dual approach," *Games and Economic Behavior*, vol. 118, pp. 648–667, 2019.
- [18] N. Buchbinder, K. Jain, and J. S. Naor, "Online primal-dual algorithms for maximizing ad-auctions revenue," in *European Symposium on Algorithms*. Springer, 2007, pp. 253–264.
- [19] Z. Huang and Q. Zhang, "Online primal dual meets online matching with stochastic rewards: configuration lp to the rescue," in *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, 2020, pp. 1153–1164.
- [20] N. Bansal, N. Buchbinder, and J. Naor, "A primal-dual randomized algorithm for weighted paging," *Journal of the ACM (JACM)*, vol. 59, no. 4, pp. 1–24, 2012.
- [21] A. Borodin and R. El-Yaniv, *Online computation and competitive analysis*. Cambridge University Press, 2005.
- [22] Y. Zhou, D. Chakrabarty, and R. Lukose, "Budget constrained bidding in keyword auctions and online knapsack problems," in *International Workshop on Internet and Network Economics*. Springer, 2008, pp. 566–576.
- [23] Z. Zhang, Z. Li, and C. Wu, "Optimal posted prices for online cloud resource allocation," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 1, pp. 1–26, 2017.
- [24] R. El-Yaniv, A. Fiat, R. M. Karp, and G. Turpin, "Optimal search and one-way trading online algorithms," *Algorithmica*, vol. 30, no. 1, pp. 101–139, 2001.
- [25] B. Awerbuch, Y. Azar, and S. Plotkin, "Throughput-competitive online routing," in *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*. IEEE, 1993, pp. 32–40.
- [26] R. J. La and V. Anantharam, "Utility-based rate control in the internet for elastic traffic," *IEEE/ACM Transactions on Networking*, vol. 10, no. 2, pp. 272–286, 2002.
- [27] D. P. Palomar and Mung Chiang, "A tutorial on decomposition methods for network utility maximization," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1439–1451, 2006.
- [28] Y. Cao, B. Sun, and D. H. K. Tsang, "Optimal online algorithms for one-way trading and online knapsack problems: A unified competitive analysis," *arXiv preprint arXiv:2004.10358*, 2020.
- [29] "Statistical analysis of network data," <https://math.bu.edu/people/kolaczyk/datasets.html>.
- [30] P. B. Key, "Optimal control and trunk reservation in loss networks," *Probability in the Engineering and Informational Sciences*, vol. 4, no. 2, p. 203–242, 1990.
- [31] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The non-stochastic multiarmed bandit problem," *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2002.

This figure "pre-ext-inc.png" is available in "png" format from:

<http://arxiv.org/ps/2101.10898v1>