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

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

***Abstract*— We consider an online version of the well-studied network utility maximization problem, where users arrive one by one and a network operator makes irrevocable rate allocation 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 logarithmic in the maximum number of links requested by a user. Extensive simulations are conducted to demonstrate the performance advantage of our proposed algorithm in comparison with two state-of-the-art algorithms. In addition, we devise an adaptive implementation of our algorithm with online learning.**

***Index Terms*— Online algorithms, networked control systems, communication networks, adaptive control.**

I. Introduction

NETWORK utility maximization (NUM) is a general optimization paradigm of vital importance in the field of networking. It has been widely applied since the seminal work by Kelly *et al.* [1]. For example, it often serves as the underlying model to draw insights into 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 communication network management [4]. Utility-based maximization has also been shown to play a significant role in areas beyond traditional communication networks, such as the demand response in power systems [5], electric vehicle charging control [6], information dissemination in vehicular ad-hoc networks [7], and many other applications [8]–[11].

Existing research on NUM usually assumes users are static. Therefore, it is focused on designing distributed algorithms and corresponding convergence issues. However, users usually arrive in an online manner. For example, hosts come sequentially when requesting network access in public networks, and cloud service requests arrive one by one at 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 irrevocable decisions depending only on 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.

Efforts exist in the literature that deal with online resource allocation with constraints. Some works, such as [12] and [13], use online learning to solve the allocation problem under the regret minimization framework and usually assume that utility functions are drawn i.i.d. from an unknown distribution. However, the i.i.d. assumption may not be valid, and in [13], constraints are allowed to be violated. Bandit feedback in online resource allocation has also attracted attention in recent years [14], [15], where only the function value of the action taken will be observed. In the online learning framework, the algorithms make decisions before observing the environment, and one usually uses regret as the performance metric, assuming a static or a weakly dynamic benchmark. For example, [14] uses the instantaneous optimal up to the current round as the benchmark and [15] uses the optimal solution to an expectation of an optimization problem.

In contrast, another stream of study assumes that the utility functions are generated adversarially and designs algorithms under the competitive analysis framework. This framework allows 1-lookahead, i.e., the algorithm can observe the information in the current round before making a decision, and compare algorithms with a strong benchmark, the optimal algorithm in the hindsight. [16] and [17] consider the online linear programming problem and the online packing problem, respectively. However, both allow violation of the constraints. One special case in [17] can ensure no constraints are violated when all constraints coefficients are either 0 or 1; however, the work just considers a simple linear objective with no uncertainty. [18] and [19] consider non-linear uncertain objectives and are the closest work to ours. However, the non-linearity source is different from ours. They maximize linear utilities of all users minus the convex cost of using the resource. The concavity comes from the convex cost function and is separable over resources. On the contrary, we aim to maximize the total utility, where the individual utility functions

Manuscript received January 26, 2021; revised October 23, 2021 and March 23, 2022; accepted June 5, 2022. Date of publication: xx; date of current version: xx. This work was supported by Hong Kong Research Grants Councils General Research Fund under Project 16211220. Paper no. TCNS-21-0611.

Y. Cao and B. Sun are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong SAR, China (emails: ycaoan@connect.ust.hk; bsunaa@connect.ust.hk).

D.H.K. Tsang is with the Internet of Things Trust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, Guangdong 511400, China, and also with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong SAR, China (e-mail: eetsang@ust.hk).

are concave, and thus our objective is separable in users.

In this paper, we design online algorithms for the general ONUM under the competitive analysis framework and ensure no violation of the constraints. We propose an online threshold-based algorithm that makes an allocation based on a threshold function. This function is an increasing function of the link utilization level and an estimate of the pseudo cost of using resources. Then the algorithm decides the allocation by solving an optimization problem that maximizes the total utility minus the pseudo cost.

One classical 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 to several online problems, such as online covering and packing [17] [20], online matching [21] and weighted paging [22], etc. The threshold function used in the proposed algorithm is also related to the dual variable, but we bypass the online primal-dual analysis and adopt a different analysis method in this paper, which directly bounds the offline optimal value by a multiple of the online counterpart and gives an order-optimal competitive ratio.

The contributions of this paper are three-fold:

- **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 takes a certain form, the algorithm yields a competitive ratio that is logarithmic in the maximum number of links requested, which matches the lower bound of the ONUM problem.
- **Applications.** We apply the algorithm to the online bandwidth allocation problem based on a real network topology and show that our algorithm is competitive against two state-of-the-art algorithms.

This paper is organized as follows. In Section II, we present the system model for ONUM and show applications that fit the model. In Section III, we present the threshold-based algorithm, give a rigorous competitive analysis of the algorithm and show that the proposed algorithm achieves the optimal competitive ratio. In Section IV, we first conduct simulations using the Abilene network topology to show its performance advantages under different arrival patterns. Then, worst-case scenarios are constructed for validating the theoretical analysis. Throughout the simulation section, two state-of-the-art online algorithms are used as benchmarks. In Section V, we conclude the paper and discuss promising future directions.

II. Online Network Utility Maximization

We describe the general system model for ONUM and provide three exemplary applications in this section. A network with a link set $\mathcal{L}$ is considered, where each link $\ell \in \mathcal{L}$ corresponds to an edge in the network graph and has a capacity of c_ℓ. Each user comes to the network one by one.

Let $\mathcal{N}$ denote the set of users and let N be the total number of users. Each user requests the access to a set of links $\mathcal{L}_i$, whose cardinality is denoted by L_i. In our problem, we assume that the link set is determined before the arrival of the user and the maximum link set size is L. The network operator decides how much capacity should be allocated to the arriving user. The allocation is equal for each link in the link set request of a user. At any time, the total allocation on any link cannot exceed its capacity. The ith user also comes with a utility function $g_i(\cdot)$ and a budget b_i.

The utility is a function of the capacity allocated and represents the revenue gained by the user. The budget is an upper bound on the allocation. We denote the information coming with user i by the tuple $A_i = \{g_i(\cdot), \mathcal{L}_i, b_i\}$.

The goal is to design an online algorithm that determines the allocation y_i at the time of the ith arrival and maximizes the total utility of all users. The difficulty of such algorithms is the lack of future information at the time of decision making; that is, y_i is irrevocably determined without knowing the tuples of future users, i.e., $\{A_k\}_{k>i}$. The performance of an online algorithm is usually evaluated by the *competitive analysis framework* [23], and we briefly introduce the basics here. Given an arrival sequence $\mathcal{I}= \{A_1, \dots, A_N\}$, let ALG($\mathcal{I}$) be the objective value achieved by an online algorithm. If the full information of the future (the full arrival sequence) $\mathcal{I}$ is disclosed at the beginning, our problem is formulated as follows:

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

$$\text{s.t.} \quad \sum_{i:\ell\in\mathcal{L}_i} y_i \le c_\ell, \forall \ell \in \mathcal{L}, \tag{1b}$$

$$0 \le y_i \le b_i, \forall i \in \mathcal{N}. \tag{1c}$$

An off-the-shelf convex program solver can solve Problem (1) optimally, and we denote the optimal objective value as OPT($\mathcal{I}$). The performance of the online algorithm is typically evaluated by the *competitive ratio,* defined as

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

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, 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., $\frac{g_i'(y)}{L_i} \in [m, M]$.*

Assumption 1 is standard for NUM problems (e.g., [1], [3], [5]). Typically, the user utility is increasing in the resource allocated and concave because of the diminishing returns property. Some commonly used utility functions are logarithmic and polynomial functions, and thus the differentiability assumption is usually satisfied. Assumption 2 requires the first-order derivatives of the utility functions to be bounded, which is essential to achieve a bounded competitive ratio. Similar assumptions appear in other online optimization literature, e.g., the online knapsack problem [24]–[26] and one-way trading problem [27], [28].

We now provide three exemplary applications, among others, that fit into the ONUM model.

Online bandwidth allocation for video streaming. Available bit rate (ABR) service is proposed in asynchronous transfer mode (ATM) networks to accommodate video streaming and other applications with a variable bit rate. The ABR service provides a minimum cell rate (MCR) guarantee for each user, and the remaining capacity will be dynamically allocated to users on demand. Consider an ATM network where application requests/users come sequentially. Each application request is binded with a source and a destination node, and the MCR for the ith request, denoted by y_i, needs to be decided before establishing the connection. The routing path is a set of links that connect its source and destination. For the ith user, denote its routing path as $\mathcal{L}_i$ and the number of links on the routing path as $L_i = |\mathcal{L}_i|$. Each user has a peak rate b_i, which is the highest rate at which the user can transmit. The user utility, $g_i(y_i)$, is modeled as a function of its allocated MCR. The goal of the network operator is to allocate the MCR in such a way that maximizes the total utility of all users. It is not difficult to see that the problem of online bandwidth allocation for variable bit rate (VBR) video streaming using ABR service fits the ONUM framework well.

Online routing of virtual circuits [29]. In the simplest version, requests $r_i = (s_i, d_i)$ come online with a predetermined routing path between source s_i and destination d_i. The algorithm will determine whether 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 ith 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 ith 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 $\sum_{\ell' \in IS_\ell} \sum_{i:\ell' \in \mathcal{L}_i} y_i \le c_\ell$ for each link $\ell \in \mathcal{L}$ and the energy constraints $(e^t + e^r) \sum_{i:v \in \mathcal{V}_i} y_i \le e_v/T - e^d$ 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.

Remark *Usually it is more realistic for applications that the allocation for each user is only valid for a time duration. This can be captured in our framework by treating time as another resource, increasing the dimension of the decision space by one. Moreover, this will not affect the logic of the analysis in Section III, and only the competitive ratio will have an additional* $\log(D)$ *term added, where* D *is the longest time duration. In our presentation, we do not emphasize the time dimension because we want to focus on the core problem of how to allocate multiple capacitated resources against the uncertainty.*

III. Competitive Online Algorithms for ONUM

In this section, an online threshold-style algorithm for ONUM is presented, followed by a competitive analysis for the algorithm, where the competitive ratio is shown to be logarithmic in the maximum number of links in a request.

Before turning to the algorithm, we first introduce an idea that is common in the distributed optimization field. To solve an optimization problem in a distributed manner, dual variables are usually viewed as the shadow prices, which reflect the costs of allocating additional units of resources at the current state. Nodes individually solve a local optimization problem that captures its own concern. The classical algorithm for congestion control [1] also falls within this kind of studies.

A standard dual-based algorithm for the offline NUM problem is as follows, where each link is associated with a dual price $\lambda_\ell(t)$ at the beginning of the tth iteration and user i determines its rate y_i at the tth iteration by solving the following maximization problem:

$$y_i(\lambda^i(t)) = \arg\max_{y \ge 0} \left(g_i(y) - \lambda^i(t) y \right), \forall i, \tag{2}$$

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

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

where t is the iteration index and γ is the step size at the tth iteration. The ONUM and the offline NUM are different in several ways. The ONUM can only access the causal information (i.e., the past and the present information), while the offline NUM has access to all information. Moreover, decisions made by the ONUM cannot be revoked, but the decision of the offline NUM is updated in each iteration. The dual update above can also take the form of a function as the algorithm in [1]. By iterating between the primal and dual updates, y_i and λ_ℓ will converge to the optimal rate allocation and dual prices for the optimization problem [30].

In the design of our algorithm, we also maximize the term that is the utility function minus the pseudo-cost as in the primal update (2). However, the aim of our problem is different, and thus the estimation of the dual price or the pseudo-cost differs. Distributed algorithms aim for the convergence of the primal and dual solutions for a static optimization problem, in contrast, our algorithm strives to reserve the right amount of resources for the future and makes decision online based on only causal information. Thus, the key is how to estimate the pseudo-cost, so that the worst-case performance is guaranteed in the face of uncertainty. The following subsections focus on answering this question.

A. Online Threshold-Based Algorithm Design

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 (Threshold Function) *A threshold function for link ℓ, $\phi_\ell(\omega) : [0, c_\ell] \to \mathbb{R}^+$, is a non-decreasing continuous function that evaluates the marginal utility of the resource at the utilization level ω.*

Contrary to equation (3) where the dual price of each link will be iteratively updated based on the utilization, the threshold function in our problem is a one-off value evaluation of the remaining capacity because the decisions are irrevocable. In essence, the threshold function characterizes the attitude towards the uncertain future; that is, how much should we reserve for the possible future arrivals with higher utility? This part is missing in the offline problems [1], [30].

Next, we present our proposed algorithm in Algorithm 1 and term it as the online algorithm with threshold functions $\phi := \{\phi_\ell\}_{\ell\in\mathcal{L}}$ (OAVF$_\phi$). In detail, OAVF$_\phi$ uses the threshold functions $\phi_\ell(\cdot)$ to evaluate the scarcity of the remaining capacity of link ℓ. We denote by ω_ℓ^i the total consumption of link ℓ after making decision for the ith user. At the time when the utilization of link ℓ is s, the cost of using an infinitesimal amount of capacity, ds, is estimated by $\phi_\ell(s)ds$, and thus the total charging cost of user i for link ℓ is $\int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1}+y_i} \phi_\ell(s)ds$. OAVF$_\phi$ then determines y_i by solving a pseudo-utility maximization problem (4), where the user's pseudo-utility is defined as its utility $g_i(y_i)$ minus the total pseudo-cost charged.

In the sequel, a brief comment on the time complexity of the proposed algorithm is provided. The problem in Equation (4) is a one-dimensional convex optimization problem with a box constraint. Define the first-order derivative of the objective function as $\Delta_i(y) = g_i'(y) - \sum_{\ell\in\mathcal{L}_i} \phi_\ell(\omega_\ell^{i-1}+y)$. The problem in Equation (4) can be solved as follows: (i) If $\Delta_i(0) \le 0$, $y_i = 0$; (ii) if $\Delta_i(b_i) \ge 0$, $y_i = b_i$; (iii) otherwise, solve $\Delta_i(y_i) = 0$ by the bisection algorithm since $\Delta_i(y)$ is a non-increasing function with $\Delta_i(0) > 0$ and $\Delta_i(b_i) < 0$. The bisection algorithm is with logarithmic time complexity, and thus the problem can be solved efficiently.

Since all the design freedom of OAVF$_\phi$ lies in the threshold function, the remainder of this section will be centered around the following question: what form should ϕ_ℓ take such that OAVF$_\phi$ can yield a bounded competitive ratio?

B. Competitive Analysis

The following theorem provides a form of the threshold function ϕ_ℓ for OAVF$_\phi$ to have a bounded competitive ratio.

Algorithm 1 Online Algorithm with threshold functions $\{\phi_\ell\}_{\ell\in\mathcal{L}}$ (OAVF$_\phi$)

```
Initialize: threshold functions $\{\phi_\ell\}_{\ell\in\mathcal{L}}$, 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
        $y_i = \arg\max_{0\le y\le b_i} \left( g_i(y) - \sum_{\ell\in\mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1}+y} \phi_\ell(s)ds \right);$   (4)
    Update for links $\ell \in \mathcal{L}_i$: $\omega_\ell^i = \omega_\ell^{i-1} + y_i$;
end for
```

The competitive ratio is logarithmic in the maximum number of links that an arrival can request.

Theorem 1 *OAVF$_\phi$ is $(\alpha+1)$-competitive if ϕ_ℓ is given by*

$$\begin{aligned}\phi_\ell(\omega) &= m\left(\exp\left(\alpha\omega/2c_\ell\right) - 1\right)\\ &= m\left(L\theta+1\right)^{\omega/c_\ell} - m, \omega \in [0, c_\ell],\end{aligned} \tag{5}$$

where $\alpha = 2\ln(L\theta + 1)$ and $\theta = M/m$.

The threshold function estimates the pseudo charging cost for making decisions. When the links are all vacant, the total charging cost at the beginning is zero, and this increases exponentially with the rate allocated. The capacity constraint is automatically observed because the charging cost for any single link gets close to LM when the utilization approaches the capacity, higher than the marginal utility of any user. This helps the algorithm act very conservatively and reserve enough capacity for future arrivals.

The algorithm in [25] is proposed to solve an online 0/1 knapsack problem, which can also be solved by a threshold-based algorithm. We extend this algorithm to continuous decision variables and term it as posted-pricing because [25] studies a posted-pricing mechanism. We use it as a benchmark algorithm for the ONUM problem. The key difference between the posted-pricing algorithm and ours lies in the threshold functions. The one for the posted-pricing algorithm is defined as follows:

$$\phi_\ell(\omega) = \begin{cases} m, & \omega \in [0, \frac{c_\ell}{\ln\theta+1}],\\ m\exp\left((\ln\theta+1)\omega/c_\ell - 1\right), & \omega \in [\frac{c_\ell}{\ln\theta+1}, c_\ell],\\ \infty, & \omega \in (c_\ell, \infty). \end{cases} \tag{6}$$

The two threshold functions are plotted for comparison in Figure 1.

Next, some intuitions for the logarithmic competitive ratio of our algorithm are provided. Consider an arrival sequence containing two arrivals, the first arrival requests link ℓ, the second arrival requests every link in $\mathcal{L}$ and the marginal utilities per link of both arrivals are $\frac{g_i'}{L_i} = M$, the highest possible. Consider all links have the same capacity 1 and both arrivals can occupy the requested links up to their capacity. The optimal choice in hindsight is to accept only the second arrival because it provides higher utility for each unit capacity of link

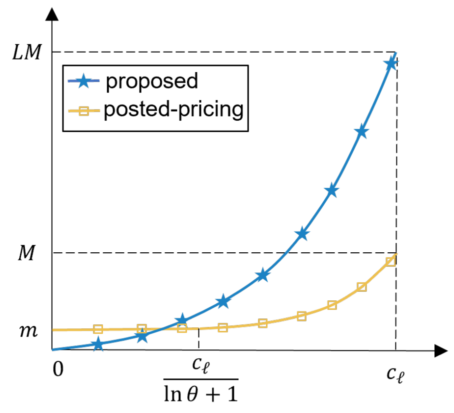


Fig. 1: Threshold functions

ℓ. The optimal return is LM. The posted-pricing algorithm will only allocate capacity to the first arrival until link ℓ is fully occupied and block the second arrival. The total utility is M, and the competitive ratio is at least $LM/M = L$. On the contrary, our proposed algorithm will stop allocating to the first arrival when the resource utilization level is at most $\phi_\ell^{-1}(M)$ and reserve more for the second arrival, leading to a higher utility. Being conservative means to be more alert to scenarios similar to the one described above, where some links become the bottleneck too early, blocking arrivals of higher utility, which is the main reason for the logarithmic competitive ratio of our algorithm.

The formal proof of Theorem 1 is constituted by the following three lemmas. The first lemma shows that OAVF$_\phi$ will observe the capacity constraints automatically.

Lemma 1 OAVF$_\phi$ *will not violate the capacity constraints.*

Proof: Assume that the first capacity violation happens after making a decision for the ith arrival. Denote the set of violated links as $\mathcal{L}_o$. Let us consider an alternative decision $\hat{y}_i = \left(c_{\ell_0} - \omega_{\ell_0}^{i-1}\right)$, where ℓ_0 is the link closest to its capacity in $\mathcal{L}_o$, i.e., $\ell_0 = \arg\min_{\ell \in \mathcal{L}_o} \left(c_{\ell_0} - \omega_\ell^{i-1}\right)$. It is clear that $\hat{y}_i < y_i$. By the mean value theorem and Assumption 2,

$$\begin{aligned} g_i(y_i) - g_i(\hat{y}_i) &= g_i'(\xi)(y_i - \hat{y}_i), \\ &= g_i'(\xi)(\omega_{\ell_0}^i - c_{\ell_0}), \xi \in [\hat{y}_i, y_i] \\ &\le L_i M(\omega_{\ell_0}^i - c_{\ell_0}). \end{aligned} \tag{7}$$

The following also holds:

$$\begin{aligned} L_i M(\omega_{\ell_0}^i - c_{\ell_0}) &\le LM(\omega_{\ell_0}^i - c_{\ell_0}) \\ &< \int_{c_{\ell_0}}^{\omega_{\ell_0}^i} \phi_{\ell_0}(s)ds \\ &\le \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}+\hat{y}_i}^{\omega_\ell^{i-1}+y_i} \phi_\ell(s)ds. \end{aligned} \tag{8}$$

By combining (7) and (8), along with the equation $\int_{\omega_\ell^{i-1}+\hat{y}_i}^{\omega_\ell^{i-1}+y_i} \phi_\ell(s)ds = \int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1}+y_i} \phi_\ell(s)ds - \int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1}+\hat{y}_i} \phi_\ell(s)ds$, we have

$$\begin{aligned} g_i(y_i) - &\sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1}+y_i} \phi_\ell(s)ds \\ &< g_i(\hat{y}_i) - \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1}+\hat{y}_i} \phi_\ell(s)ds, \end{aligned}$$

which contradicts the algorithm decision rule (4). Thus, the assumption does not hold, and the OAVF$_\phi$ algorithm will not violate the capacity constraints. ■

Lemma 2 *The objective value of* OAVF$_\phi$ *is lower bounded by the final dual prices as follows:*

$$\alpha \text{ALG} \ge \sum_{\ell \in \mathcal{L}} c_\ell \phi(\omega_\ell^N). \tag{9}$$

Proof: By the decision rule (4), we have for each i,

$$g_i(y_i) \ge \sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1}+y_i} \phi_\ell(s)ds.$$

Recall Assumption 2, $g_i'/L_i \in [m, M]$ and $g_i(y_i) = g_i(0) + g_i'(\xi)y_i = g_i'(\xi)y_i, \xi \in [0, y_i]$. Then we have $g_i(y_i) \ge mL_i y_i$. Thus, the return of the online algorithm, ALG, is

$$\begin{aligned} \sum_{i=1}^N g_i(y_i) &\ge \frac{1}{2}\sum_{i \in \mathcal{N}} g_i(y_i) + \frac{1}{2}\sum_{i \in \mathcal{N}}\sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1}+y_i} \phi_\ell(s)ds \\ &\ge \frac{1}{2}\sum_{i \in \mathcal{N}} mL_i y_i + \frac{1}{2}\sum_{i \in \mathcal{N}}\sum_{\ell \in \mathcal{L}_i} \int_{\omega_\ell^{i-1}}^{\omega_\ell^{i-1}+y_i} \phi_\ell(s)ds \\ &= \frac{m}{2}\sum_{\ell \in \mathcal{L}} \omega_\ell^N + \frac{1}{2}\sum_{\ell \in \mathcal{L}} \int_0^{\omega_\ell^N} \phi_\ell(s)ds. \end{aligned} \tag{10}$$

Note that the derivative of ϕ_ℓ can be represented by ϕ_ℓ itself, i.e., $\phi_\ell(\omega) = \frac{2c_\ell}{\alpha}\phi_\ell'(\omega) - m$. Plugging it into (10),

$$\frac{1}{2}\sum_{\ell \in \mathcal{L}} \int_0^{\omega_\ell^N} \phi_\ell(s)ds = \frac{1}{\alpha}\sum_{\ell \in \mathcal{L}} c_\ell \phi(\omega_\ell^N) - \frac{m}{2}\sum_{\ell \in \mathcal{L}} \omega_\ell^N. \tag{11}$$

Adding equations (10) and (11), we have $\alpha\text{ALG} \ge \sum_{\ell \in \mathcal{L}} c_\ell \phi(\omega_\ell^N)$. ■

Lemma 3 *The offline optimal objective value is upper bounded by the final dual prices as follows:*

$$\text{OPT} \le \text{ALG} + \sum_{\ell \in \mathcal{L}} c_\ell \phi_\ell(\omega_\ell^N). \tag{12}$$

Proof: Denote the offline optimal decision for the ith arrival as y_i^* and the set of arrivals to which the offline optimal allocates more than OAVF$_\phi$, i.e., $y_i < y_i^*$ as $\mathcal{N}^1$.

$$\begin{aligned} \text{OPT} &= \sum_{i \in \mathcal{N}} g_i(y_i^*) \\ &= \sum_{i \in \mathcal{N}} (g_i(y_i^*) - g_i(y_i)) + \sum_{i \in \mathcal{N}} g_i(y_i) \\ &\le \sum_{i \in \mathcal{N}^1} (g_i(y_i^*) - g_i(y_i)) + \sum_{i \in \mathcal{N}} g_i(y_i). \end{aligned} \tag{13}$$

Now it remains to bound the first term, the difference between the offline optimal algorithm and the online algorithm for $\mathcal{N}^1$. We make the following claim:

Claim 1 *The arrivals in $\mathcal{N}^1$ have $g_i'(y_i) \le \sum_{\ell\in\mathcal{L}_i} \phi_\ell(\omega_\ell^N)$.*

Proof: For arrivals in $\mathcal{N}^1$, we have $y_i \in [0, b_i)$, because when $y_i = b_i$, $y_i^* \le y_i$, violating the definition of $\mathcal{N}^1$. If $y_i = 0$, according to the decision rule (4) and the concave property in Assumption (1), we have $\sum_{\ell\in\mathcal{L}_i} \phi_\ell(\omega_\ell^{i-1}) \ge g_i'(0)$, thus, $g_i'(y_i) \le \sum_{\ell\in\mathcal{L}_i} \phi_\ell(\omega_\ell^N)$ holds when $y_i = 0$. For $y_i \in (0, b_i)$, by (4), the following holds:

$$g_i'(y_i) = \sum_{\ell\in\mathcal{L}_i} \phi_\ell(\omega_\ell^i) \le \sum_{\ell\in\mathcal{L}_i} \phi_\ell(\omega_\ell^N).$$

Thus, we have shown that for arrivals in $\mathcal{N}^1$, $g_i'(y_i) \le \sum_{\ell\in\mathcal{L}_i} \phi_\ell(\omega_\ell^N)$ holds. ■

Based on Claim 1, we can proceed to bound the offline optimal return, following (13).

$$\begin{aligned}\sum_{i\in\mathcal{N}^1} (g_i(y_i^*) - g_i(y_i)) &\le \sum_{i\in\mathcal{N}^1} g_i'(y_i)(y_i^* - y_i)\\ &\le \sum_{i\in\mathcal{N}^1}\sum_{\ell\in\mathcal{L}_i} \phi_\ell(\omega_\ell^N)(y_i^* - y_i)\\ &= \sum_{\ell\in\mathcal{L}} \phi_\ell(\omega_\ell^N) \sum_{i\in\mathcal{N}^1:\ell\in\mathcal{L}_i} (y_i^* - y_i)\\ &\le \sum_{\ell\in\mathcal{L}} c_\ell\phi_\ell(\omega_\ell^N), \end{aligned} \tag{14}$$

where the last inequality is due to the fact that the offline optimal algorithm will not violate the capacity constraints, i.e., for any ℓ, $\sum_{i:\ell\in\mathcal{L}_i} y_i^* \le c_\ell$. Thus, by combining (13) and (14), we have OPT $\le \sum_{i\in\mathcal{N}} g_i(y_i) + \sum_{\ell\in\mathcal{L}} c_\ell\phi_\ell(\omega_\ell^N) =$ ALG $+ \sum_{\ell\in\mathcal{L}} c_\ell\phi_\ell(\omega_\ell^N)$. ■

To recap what we have, Lemma 1 ensures that the online algorithm OAVF$_\phi$ produces a feasible solution, and then combining Lemma 2 and Lemma 3 gives OPT $\le (1+\alpha)$ALG, which proves Theorem 1.

Now we show that the competitive ratio of our algorithm matches the lower bound of the problem.

Lemma 4 *There exists no online algorithm for the online network utility maximization problem that can achieve a competitive ratio smaller than $\Omega(\log\theta L)$.*

Proof: We first show that the classic one-way trading problem and the online fractional packing problem are two special cases of the ONUM problem. The ONUM problem reduces to the one-way trading problem when (i) the objective function is linear and its derivative is bounded in $[m, M]$ with $\theta = M/m$, (ii) there is only one resource (i.e., $L = 1$), and (iii) there is no individual budget constraint $b_i = c, \forall i$, with c being the resource capacity. The ONUM problem reduces to the online fractional packing problem when (i) the objective function is $g_i(y_i) = y_i$ and (ii) there is no individual budget constraint $b_i = \max_\ell c_\ell, \forall i$. Existing works have shown that the competitive ratios of the one-way trading problem and the online fractional packing problem are lower bounded by $\Omega(\log\theta)$ [26] and $\Omega(\log L)$ (Lemma 3.2 of [17]), respectively. Thus, the competitive ratio of the ONUM problem must be lower bounded by

$$\begin{aligned}\max\{\Omega(\log\theta), \Omega(\log L)\} &\ge \frac{1}{2}\Omega(\log\theta) + \frac{1}{2}\Omega(\log L)\\ &= \Omega(\log\theta L).\end{aligned}$$

■

IV. Simulation Results

A. Simulation Settings

We use the network trace of the Abilene network collected from December 8, 2003, to December 28, 2003 [31], which contains the routing information and network topology but not the time stamps, to demonstrate the performance of our algorithm. Abilene network is the most widely used academic network created by the Internet2 community. The data set includes 11 backbone routers, 41 links and 121 source-destination pairs. 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 2016×121, where the component on the ith column and the jth row represents the traffic volume of the ith source-destination pair measured in the jth 5-min slot. The routing matrix is a binary matrix whose dimension is 121×41, where the ith row vector denotes the routing path of the corresponding source-destination pair. We know from the routing matrix that the maximum request link set size L is 6. We remove the 11 self-pairs that do not need routing decisions and use the remaining traffic matrix of size 2016×110 in the simulation.

(Link Set Arrival Type) Each arrival instance contains 300 arrivals. To generate the arrival order, we draw samples from 110 source-destination pairs in the following ways to represent different types of the request link sets arrivals.

- *Random Sample (RS).* Samples are drawn uniformly from the 110 source-destination pairs.
- *Increasing Request Link Set (IRLS).* The first half of the instance is uniformly sampled from the set of source-destination pairs that request less than or equal to 3 links (half of the maximum request link set size), and the second half of the instance is uniformly sampled from the set of source-destination pairs that request more than 3 links.
- *Decreasing Request Link Set (DRLS).* Contrary to the IRLS case, the first half requests more than 3 links and the second half requests less than or equal to 3 links. The sampling method inside each half is the same as for the IRLS type.

Note that the three types above may not be a complete characterization of any real-life arrival instance. The reason for evaluating the performance under these three special cases is that the RS type captures the possibility that every source-destination pair is equally important. The IRLS type and DRLS type are to test the influence of the resource request type, or more specifically, the arrival sequence of the request link set, on the algorithmic performance for ONUM, a multi-resource online allocation problem. In particular, our interest is to find out how to allocate capacity for a potential bottleneck link. The routing information in the Abilene network data set shows that there exist links serving as the backbone links and appearing in many routing paths, which supports our use of this data set. Next, utility functions in an instance are introduced.

(**Utility Type**) Utilities and rate limits are not included in the data set. We first discuss the properties of the utility function. From the perspective of the network operator, a utility function $g_i(y) = L_i y$ allows the number of links requested by each user to be considered in the network utility. However, for individual users who care more about their own throughput, a utility function $g_i(y) = y$ is more realistic. To accommodate the concerns of both the network operator and the individual users, we allow g'_i to be bounded in $[1, L_i]$ and propose three types of utility as follows:

- *Throughput-based.* $g'_i = 1$, capturing the user throughput.
- *Unit-density.* $g'_i = L_i$, representing the concern of the network operator.
- *Balanced.* $g'_i \in [1, L]$, which generalizes the aforementioned two types.

All three types will be investigated in simulations. Following the convention in NUM, the utility functions of the balanced type are $g_i(y) = a_i \log(1 + y)$, noting the decreasing marginal utility property for many real-life utility functions. The throughput-based and unit-density types consider utility functions $g_i(y) = y$ and $g_i(y) = L_i y$, respectively. Note that the balanced type is the only type whose utility functions contain a random variable, a_i, which is constructed as follows:

1) *Uniform Case*. In this case, the sequences of a_i in an instance are uniformly distributed within $[1, L]$.
2) *Robust Case*. This case is to evaluate the robustness of online algorithms. The first half of the a_i,s in an instance are uniformly distributed within $[1, (L + 1)/2]$, and the second half is uniformly distributed within $[(L+1)/2, L]$.

We assume that the rate limits of arrivals $b_i = C/k$, where C is the capacity of the links and k is set to be 20. We use the uniform capacity case to illustrate and set $C = 1$.

(Performance Metric) Given any arrival instance $\mathcal{I}$, we define the empirical ratio by

$$\text{ER}(\mathcal{I}) = \frac{\text{OPT}(\mathcal{I})}{\text{ALG}(\mathcal{I})},$$

where OPT($\mathcal{I}$) is the offline optimal value and ALG($\mathcal{I}$) is the return of the online algorithm for $\mathcal{I}$, respectively. We use the Matlab API of the Sedumi solver to solve the offline optimal value. For each meaningful combination of the link set arrival type and the utility type, we run 1000 instances, generate the cumulative distribution function (CDF) of the empirical ratio for a given algorithm and compare their performance. Specifically, for the balanced utility type, we further divide the results into the uniform case and the robust case, as mentioned before.

(Benchmarks) We compare our proposed algorithm with two benchmarks, which we introduce as follows:

- *Primal-dual.* This benchmark has been shown to be optimal for the throughput-based utility in [29], where $g_i(y) = y$. For the throughput-based utility type, we expect our algorithm to achieve performance at the same level as the primal-dual algorithm, if possible, and to beat it for the other two utility types. We use **primal-dual** to denote this algorithm in the simulations.
- *Posted-pricing.* To show that our design of the threshold function is vital to the algorithmic performance, we use in Algorithm 1 the threshold function defined in Equation (6). This threshold function is shown to be optimal for the single-resource online allocation problem in the discrete decision case [25]. We use **posted-pricing** to denote the algorithm with the above threshold function in the simulations.

In the following, we assess the performance under the combination of the Abilene network topology and the synthetic arrival instances described above. Then we validate our theoretical results, showing the performance of our proposed algorithm in the worst case.

B. Results

1) Simulations over the Abilene network: The results presented here are obtained based on simulations over the Abilene network topology and the routing information provided in the Abilene data set. We compare the algorithmic performance for all three types of utility: throughput-based, unit-density and balanced. Note that both our algorithm and the posted-pricing algorithm require knowledge of the lower and upper bounds of g', and we assume they are true in this section and the next. In Section IV-C, we will show that the lower and upper bounds can be learned adaptively.

Figure 2 shows the CDFs of the empirical ratio for the throughput-based utility. For this utility, if the early arrivals have larger request link sets, i.e., they are the DRLS type, any online algorithm should perform worse than in the case that the request link set size is growing and the other variables are kept the same. Because the throughput-based utility does not reward the user by the number of links she/he requests, the early arrivals with larger request link sets will not be beneficial enough since she/he uses more resource to return the same utility. This is validated by the simulation results shown in Figure 2, where the range of the x axis is larger in Figure 2c than in Figure 2b. The primal-dual algorithm is proven to be order-optimal for the throughput-based utility, and thus our algorithm is also optimal. Actually, the RS type can be viewed as a combination of the IRLS and DRLS types, which explains that the performance of each algorithm for the RS type lies between that for the IRLS and DRLS types.

Figure 3 shows the CDFs of the empirical ratio of different algorithms for the unit-density utility and the IRLS link set arrival type. Contrary to the result for the throughput-based utility, the IRLS case is the most difficult to deal with for this utility, because for a potential bottleneck link, the arrivals requesting it, but also other links, will return a higher utility value than those requesting only it and no other links when the allocation amount is the same. This requires an algorithm to reserve enough of the capacity of any potential bottleneck link for the future arrivals with a larger link request set. The unit-density utility is amenable to greedy algorithms, since any algorithm will return exactly the same amount

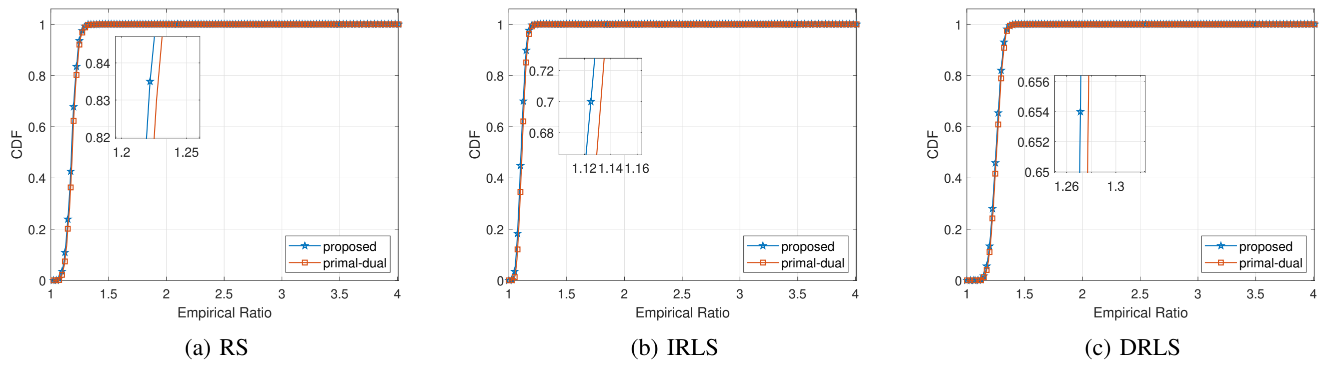


Fig. 2: CDFs of the empirical ratio of algorithms for the throughput-based utility. The topology and the routing information are from the Abilene network data set. The three subfigures show the performance under different link set arrival types.

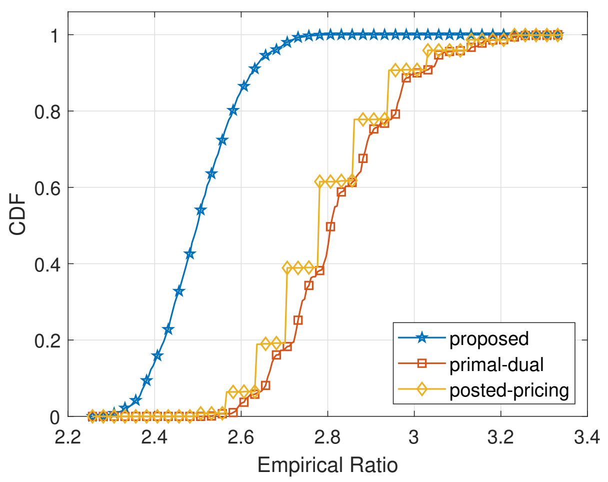


Fig. 3: CDFs of the empirical ratio of algorithms for the unit-density utility and the IRLS link set arrival type.

of utility as the total link capacity it consumes. Thus, we show only the performance of the most difficult link set arrival type, the IRLS type, for the unit-density utility. The primal-dual algorithm performs the worst for the unit-density utility, because it is designed for the throughput-based utility and cannot differentiate users requesting different numbers of links, which just misses meeting the essential requirement that we discussed before.

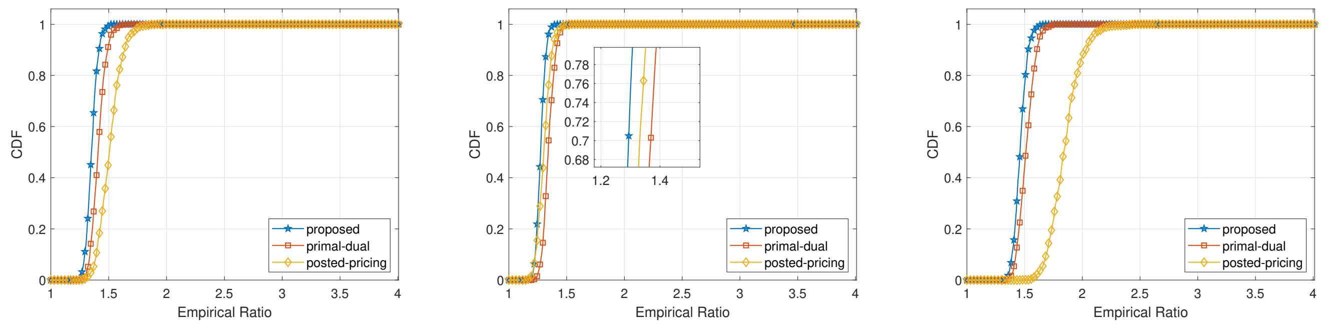
Note that the posted pricing algorithm behaves like a greedy algorithm for this case, because both m and M equal 1 for the unit-density case. This explains the step effect in Figure 3, because the rate limits are fixed to $1/20$ of the link capacity, and a greedy algorithm will allocate all the remaining capacity, if needed, to fulfill the request. The advantage of our algorithm over the posted-pricing algorithm is due to the setting of the threshold function at the capacity, i.e., $\phi_\ell(c_\ell)$. It sets the value to be M, while we set the value to be LM. A higher $\phi_\ell(c_\ell)$ means more conservative behavior, allocating less to the earlier arrivals and reserving more for the future arrivals, which is especially useful for the IRLS case with the unit-density utility type.

Next, we show the performance under the balanced utility type. For this type, the utility coefficients a_i have two possible settings, the uniform and the robust. Figure 4 and Figure 5 show the distribution of the empirical competitive ratio under the uniform and the robust utility, respectively. We see from these figures that our proposed algorithm achieves the smallest empirical ratio for all combinations of link set arrival types and utility cases when the utility is of the balanced type. This shows that our algorithm can be applied to more general scenarios.

When comparing Figure 4 and Figure 5, it is obvious that the robust utility case is more difficult since all algorithms return larger empirical ratios. Notice that the performance of the primal-dual algorithm is quite stable over the link set arrival types, both in Figure 4 and Figure 5, which again corroborates our understanding that the design of the primal-dual algorithm relies on the throughput-based utility and does not consider the effect of the link set arrivals.

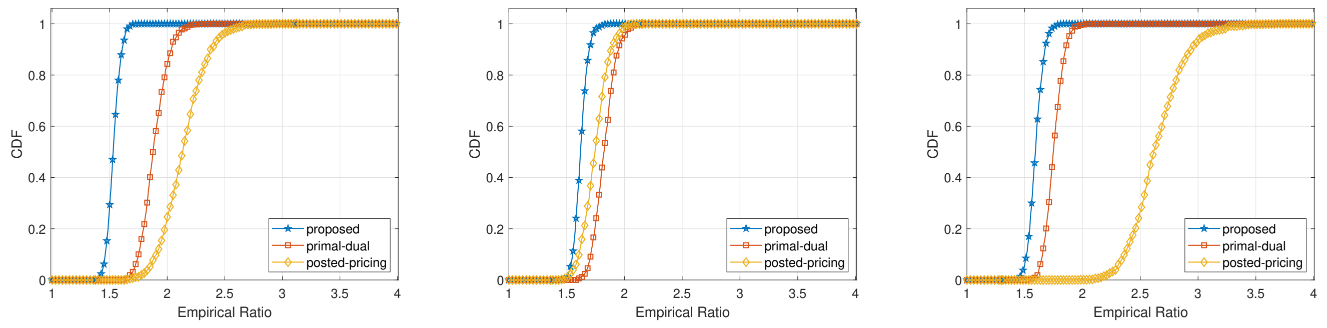
In sharp contrast, the performance of the posted-pricing algorithm varies significantly when the link set arrival type changes. The reason is that the designed threshold function is directly borrowed from the single-resource case and cannot take the possibly different link set size into consideration. The posted-pricing algorithm makes the allocation decision as if different links are of the same importance. However, the reality is that some links are more popular than others and require more conservative allocation, like our algorithm provides. We think this is the key to designing a successful online algorithm for the multiple-resource online allocation problem.

2) Simulations on the worst-case input: We validate our theoretical results in this section. The worst-case instance of our algorithm is constructed following the worst-case input in the lower bound proof in [29]: Consider a line network with L links of capacity 1, and the links are indexed by ℓ. L is chosen such that $\log(L)$ is an integer. For all arrivals, the rate limits are uniformly 1, equal to the capacity. The first group of arrivals contains only 1 user, who requests all L links; the second group of arrivals consists of two users, the first of whom requests the first half of the links and the second the remaining half; and the ith group of arrivals contains 2^{i-1} users, indexed by $j \in [0, 2^{i-1} - 1]$. The utility is throughput-based, i.e., $g_i(y) = y$. The difficulty of this arrival instance is that the request link set size decreases by a factor of two



(a) RS (b) IRLS (c) DRLS

Fig. 4: CDFs of the empirical ratio under the balanced utility case. The utilities of the instances are drawn from the uniform case. The three subfigures show the performance under different arrival instances.



(a) RS (b) IRLS (c) DRLS

Fig. 5: CDFs of the empirical ratio under the balanced utility case. The utilities of the instances are drawn from the robust case. The three subfigures show the performance under different arrival instances.

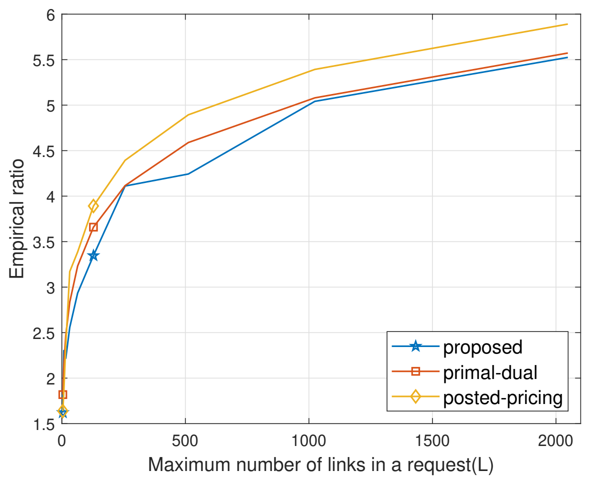


Fig. 6: Performance of all algorithms under the worst-case input specified in the lower bound proof of the problem.

for subsequent groups, so the capacity consumed in a group to gain a unit of utility also decreases by a factor of two.

For this arrival instance, it has been proven that no algorithm will achieve a competitive ratio better than the order of $\Omega(\log L)$, and the primal-dual algorithm is shown to be order optimal for this arrival instance. The performance of the three algorithms considered is shown in Fig. 6. We observe that our algorithm returns a higher utility (a smaller empirical ratio) than both the primal-dual algorithm and the posted-pricing algorithm, while all of them exhibit logarithmic behavior. Therefore, it achieves the order optimality for this worst-case instance, and is better than the two existing state-of-the-art algorithms in the worst case.

C. Adaptive Implementation

Note that the results presented up to this point are based on the knowledge of the lower and upper bounds, but there are many possible approaches to overcome this limitation. One way is to get estimates based on the past history and use them in the algorithm. However, sometimes such history will be inaccessible to a system operator. In this section, we present one viable approach to overcome this limitation. We show that without prior information of the bounds, we can still get good performance by applying online learning techniques and learning the bounds along the way.

In this subsection, we view the lower and upper bounds as changeable algorithmic parameters, and thus denote them as $(\tilde{m}, \tilde{M})$ to differentiate them from those in the previous algorithms. How to intelligently choose the parameters? One way is to model the algorithmic learning 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

Algorithm 2 Online Adaptive Implementation of Algorithm 1

```
1: Initialize: Initial weights $w_{(\tilde{m},\tilde{M})} = w_0$, for $\forall(\tilde{m},\tilde{M}) \in \mathbb{A}$. Initial probability of choosing arm $(\tilde{m},\tilde{M})$ is $p_{(\tilde{m},\tilde{M})} = w_0/(\sum_{(\tilde{m},\tilde{M})} w_0) = 1/|\mathbb{A}|$. Stepsize parameter $\eta$.
2: for episode $i = 1, \ldots, N$ do
3:     Select the arm $(\hat{m}_i, \hat{M}_i)$ according to the probability distribution $\mathbf{p}$.
4:     Run Algorithm 1 in parallel for all $(\tilde{m},\tilde{M}) \in \mathbb{A}$, with parameter $(m, M) = (\tilde{m},\tilde{M})$ to execute the arrivals in episode $i$.
       Obtain utility vector $\mathbf{u} = (u_{(\tilde{m},\tilde{M})})_{(\tilde{m},\tilde{M})\in\mathbb{A}}$.
5:     Update the weight of arm $(\tilde{m},\tilde{M}), \forall(\tilde{m},\tilde{M}) \in \mathbb{A}$:
           $$w_{(\tilde{m},\tilde{M})} \leftarrow w_{(\tilde{m},\tilde{M})} \exp\left(\frac{\eta u_{(\tilde{m},\tilde{M})}}{u^*}\right),$$
       where $u^* = \max_{(\tilde{m},\tilde{M})\in\mathbb{A}} u_{(\tilde{m},\tilde{M})}$.
6:     Update the probability of choosing arm $(\tilde{m},\tilde{M}), \forall(\tilde{m},\tilde{M}) \in \mathbb{A}$:
           $$p_{(\tilde{m},\tilde{M})} \leftarrow w_{(\tilde{m},\tilde{M})} / \sum_{(\bar{m},\bar{M})\in\mathbb{A}} w_{(\bar{m},\bar{M})}.$$
7: end for
8: Collect the total utility $\sum_{i=1}^{N} u_{(\hat{m}_i,\hat{M}_i)}$.
```

gained by choosing $(\tilde{m},\tilde{M})$ as the parameters in the threshold function $\phi_\ell(y)$ in one episode. The algorithm keeps a record of a distribution over arms based on the utilities observed from each arm, and then chooses an arm according to the distribution. In our problem, full feedback (i.e., the knowledge of the utilities of all arms in each step) is available when we run the algorithms with different parameters in parallel. Thus, we model the tuning process as an expert problem instead. We adopt the classic Hedge algorithm [32] in online learning and show the complete algorithm in Algorithm 2.

In Algorithm 2, we consider N episodes, each of which consists of an instance with E arrivals. We set $E = 100$ and $N = 4000$ in the simulation. The stepsize parameter η is set by following the convention [32]. In our problem, the total number of possible parameters is $|\mathbb{A}| = 400$, and thus we set $\eta = \sqrt{\frac{2\log|\mathbb{A}|}{N}} = 0.055$ in the simulation. We investigate the balanced utility type of arrivals, for which the sequences of a_is are generated from the truncated Gaussian distribution with mean = $\frac{1+L}{2}$ and variance = 1.

For a fair comparison, we also implement an adaptive version of the posted-pricing algorithm. Fig. 7 compares the average cumulative utility of the adaptive algorithm and the algorithm with the best possible parameter selected offline. It shows that the average utility converges as the learning process moves forward and the proposed Algorithm 2 outperforms the posted-pricing benchmark in this adaptive implementation. Compared with the previous method of choosing the parameter 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 a wider range of application scenarios.

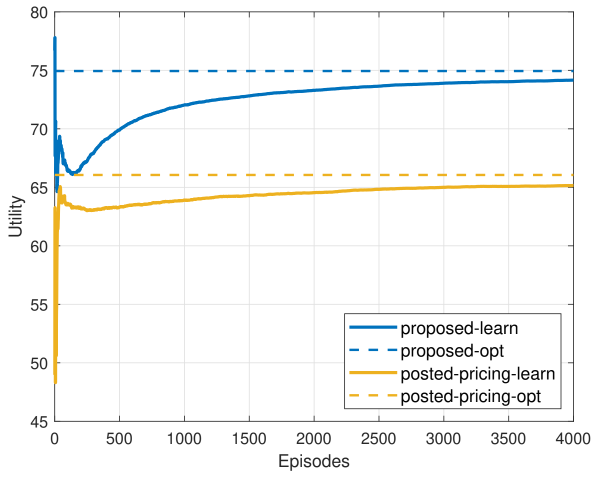


Fig. 7: Evolution of the average cumulative utility. **Alg-opt** represents the algorithm with the best possible parameters and **Alg-learn** is the algorithm with learned parameters, where **Alg** is either the posted-pricing algorithm or the proposed Algorithm 2.

V. Conclusions and Future Directions

In this paper, we consider the online network utility maximization problem and develop an algorithm that makes an allocation based on the utilization level. We show that the proposed algorithm achieves the optimal competitive ratio, which is logarithmic in the number of links in a request. Extensive simulations are conducted to show the empirical performance advantages of the proposed algorithm, compared with two state-of-the-art algorithms. For practical use, we devise an adaptive implementation of the algorithm, employing online learning techniques.

The ONUM problem shows a superior performance for the bandwidth allocation problem. It is compelling to look for other real-life applications that can be modeled by the ONUM. For example, the problem in this paper does not explicitly consider the fairness issue, which may be of interest, and the problem considered in this paper is also under the full-information setting, but it would be interesting to explore the bandit setting. Moreover, a combination of ideas from online learning and online algorithm design, such as utilizing the learning to predict crucial information in online algorithms, is also worth more attention.

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.
[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] S. R. Balseiro, H. Lu, and V. Mirrokni, "The best of many worlds: Dual mirror descent for online allocation problems," *Operations Research*, 2022.
[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] T. Chen and G. B. Giannakis, "Bandit convex optimization for scalable and dynamic iot management," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 1276–1286, 2018.
[15] X. Fu and E. Modiano, "Learning-NUM: Network utility maximization with unknown utility functions and queueing delay," in *Proceedings of the Twenty-second International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2021, pp. 21–30.
[16] 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.
[17] 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.
[18] 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.
[19] Z. Huang and A. Kim, "Welfare maximization with production costs: A primal dual approach," *Games and Economic Behavior*, vol. 118, pp. 648–667, 2019.
[20] 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.
[21] 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.
[22] 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.
[23] A. Borodin and R. El-Yaniv, *Online Computation and Competitive Analysis*. Cambridge University Press, 2005.
[24] 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.
[25] 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.
[26] B. Sun, A. Zeynali, T. Li, M. Hajiesmaili, A. Wierman, and D. H. Tsang, "Competitive algorithms for the online multiple knapsack problem with application to electric vehicle charging," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 4, no. 3, pp. 1–32, 2020.
[27] 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.
[28] Y. Cao, B. Sun, and D. H. Tsang, "Optimal online algorithms for one-way trading and online knapsack problems: A unified competitive analysis," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 1064–1069.
[29] B. Awerbuch, Y. Azar, and S. Plotkin, "Throughput-competitive on-line routing," in *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*. IEEE, 1993, pp. 32–40.
[30] D. P. Palomar and M. 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.
[31] Statistical analysis of network data. [Online]. Available: https://math.bu.edu/people/kolaczyk/datasets.html
[32] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

Ying Cao (Student Member, IEEE) received the B.Eng. degree from Department of Electronic Engineering and Information Science, University of Science and Technology of China in 2018. She is currently working toward the Ph.D. degree with Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology. Her research interests include online algorithms and online learning in networked systems.

Bo Sun (Member, IEEE) is a Postdoctoral Fellow at The Chinese University of Hong Kong. He received his B.E. degree from Harbin Institute of Technology, Harbin, China, in 2013, and his Ph.D. degree from The Hong Kong University of Science and Technology in 2018. His research focuses on optimization and decision-making under uncertainty with applications to real-world networked systems.

Danny H.K. Tsang (M'82-SM'00-F'12) received the Ph.D. degree in electrical engineering from the Moore School of Electrical Engineering at the University of Pennsylvania, U.S.A., in 1989. Upon graduation, he joined the Department of Computer Science at Dalhousie University in Canada. He later joined the Department of Electronic and Computer Engineering at the Hong Kong University of Science and Technology in 1992 and is now a professor in the department. He has also served as professor and the Founding Acting Thrust Head of the Internet of Things Thrust of the HKUST (Guangzhou) since March 2020. He was a Guest Editor for IEEE Journal on Selected Areas in Communications' special issue on Advances in P2P Streaming Systems, an Associate Editor for Journal of Optical Networking published by the Optical Society of America, and a special issue Guest Editor for IEEE Systems Journal, Transactions on Industrial Informatics, Communications Magazine, and Network Magazine. He currently serves as Technical Editor for IEEE Communications Magazine. He was nominated to become an IEEE Fellow in 2012 and an HKIE Fellow in 2013. During his leave from HKUST in 2000-2001, Dr. Tsang assumed the role of Principal Architect at Sycamore Networks in the United States. He was responsible for the network architecture design of Ethernet MAN/WAN over SONET/DWDM networks. He invented the 64B/65B encoding (US Patent No.: US 6,952,405 B2) and contributed it to the proposal for Transparent GFP in the T1X1.5 standard that was advanced to become the ITU G.GFP standard. The coding scheme has now been adopted by International Telecommunication Union (ITU)'s Generic Framing Procedure recommendation GFP-T (ITU-T G.7041/Y.1303)) and Interfaces for the Optical Transport Network (ITU-T G.709/Y1331). His current research interests include cloud computing, edge computing, NOMA networks, online algorithm design, and smart grids.