Online User-AP Association with Predictive Scheduling in Wireless Caching Networks

Shuang Zhao[1,4], Ziyu Shao[2], Hua Qian[3,4], Yang Yang[1,4]
[1]Shanghai Institute of Microsystem and Information Technology, CAS,
[2]ShanghaiTech University, [3]Shanghai Advanced Research Institute, CAS,
[4]University of Chinese Academy of Sciences
Email: shuang.zhao@wico.sh, shaozy@shanghaitech.edu.cn, hua.qian@wico.sh, yang.yang@wico.sh

***Abstract*—Caching is a promising way to alleviate the capacity bottleneck of wireless networks. To fully exploit the potential of wireless caching networks, joint optimization of user-AP(Access Points) association and caching placement is necessary. However, such optimization problem is very challenging. Most of existing work focus on the caching placement problem by assuming a fixed user-AP association, which is impractical in a dense network with intermittent and random request arrivals. In this paper, we focus on the user-AP association and resource allocation problem in dense and resource-limited network under the specific caching placement. We first design a *Joint User-AP association and Resource Allocation* (JUARA) algorithm to maximize the average network utility based on Lyapunov optimization technique, which is an online algorithm with strong theoretical performance guarantee and does not require any statistical information of the system dynamics. Given the NP-hard user-AP association and bandwidth allocation problem in JUARA, we reformulate it and prove the transferred problem falls in the category of modular maximization over two matroids. A time efficient greedy algorithm is proposed which achieves $\frac{1}{2}$ of optimal value. Then we incorporate the prediction of arriving traffic into the User-AP association, which means user's file requests with a limited future time window can be estimated accurately. We propose a *Predictive User-AP association and Resource Allocation* (P-UARA) algorithm to utilize such future information. Numerical results validates the effectiveness of our proposed predictive scheduling algorithm and shows that it performs better in high workload environment.**

I. Introduction

Recent years have seen the explosive growth of global mobile data traffic, which is expected to increase by more than 53% annually to 30.6 Exabytes per month by 2020 [1]. It is well understood that the current trend of cellular technology (e.g.,LTE) cannot cope with such traffic increase. In order to support the increasing data traffic, one main effort is to adopt network densification to boost the network capacity, which is expected to increase the capacity significantly in future 5G networks [2]–[4]. However, this increase impose a heavy burden on the backhaul links for the wireless access at the edge, leading to significantly degradation of user QoE (quality of experience). In order to tackle such problem, one promising approach is deploying cache at the wireless network edges, such as small-cell wireless access points (APs) [5]–[8] and user terminals (UTs) [9]–[11], where backhaul capacity is replaced with computation and storage capacity. In this way, users can directly download the frequently requested data (e.g., videos) or called popular contents from the local cache without increasing the burden over backhaul links. Different from conventional cloud computing, where remote public clouds are utilized, such caching networks offers computation capability and storage capability at the wireless edges, which would relieve the burden over backhaul links greatly.

Meanwhile, to provide high-level quality service, it is important to understand human behavior features and utilize such information for system scheduling policy design in caching networks. Therefore, lots of studies have been conducted for learning human behavior patterns, such as data mining [12] and online social networks [13].

A. Related works

There are two critical phases in wireless caching networks [14]. The first phase, *content placement phase*, is to cache the contents efficiently on the limited cache memories. The second phase, *content delivery phase*, is to determine user-AP association and then delivery the requested files to the users over the wireless channel. In this phase, if the network is dense and the resource is fiercely competitive, the rate required for serving requested file would be the main limitation. To fully exploit the potential of wireless caching networks, it is necessary to conduct joint optimization of caching placement and content delivery. However, such optimization problem is very challenging [5].

Most existing work of wireless caching networks focus on the caching placement problem while assuming fixed network topologies [15]–[17]. For example, Shanmugam *et al.* in [5] studied the caching placement problem given a certain network topology (users APs connectivity) to obtain the minimum expected downloading time for files. A belief propagation based distributed algorithm is proposed in [6] to solve the caching placement problem. Peng *et al.* in [7] proposed caching placement strategy to minimize the average download delay of files, subject to the caching capacity constraint of each BS. Song *et al.* in [18] proposed caching placement algorithms to minimize the average bit error rate (BER) in a wireless network with Rayleigh flat fading channel model. Psaras *et al.* in [19] introduced the resource management for in-network caching environments. However, such network with fixed topology does not adapt to the resource limited or dense network. A more efficient user-AP association policy or resource scheduling policy needs to be design aiming at content delivery phase.

There are also some preliminary studies focus on content delivery in caching networks. In [20], the author designed a policy which allow the user to dynamically select helper nodes in wireless caching networks. However, such caching scheme failed to take advantage of human behavior features, which is beneficial for system scheduling policy design.

B. Contributions

In this paper, we investigate a caching network formed by densely deployed fixed nodes (denoted as APs hereafter), serving multiple stationary or low-mobility users. We focus on dynamic user-AP association and resource allocation under online policy, which relies on limited or no prediction information (future information). In each time slot, users' request arriving at the network will be queued at the operator center, which controls all the APs through wired backhaul links. Then the operator center would decide the scheduling policy including user-AP association and bandwidth allocation for serving the requests.

To the best of our knowledge, this is the first work that proposes the user-AP association and resource allocation algorithms in wireless caching networks. The main contributions of this paper are summarized as follows:

- ***Online scheduling policy****:* We consider stochastic network where users' locations, traffic demands and channel conditions change over time, which require the operator center to design online algorithm which determine the dynamic user-AP association and resource allocation for users. Based on Lyapunov optimization techniques, we first propose the low-complexity joint user-AP association and resource allocation (JUARA) algorithm, which dose not require any prior statistic information about network dynamics.
- ***Efficient approximation algorithm****:* To solve the NP-hard subproblem in JUARA, which is a complicated nonlinear integer programming problem, we exploit the structure information and reformulate it to an equivalent modular maximization problem over two matroid constraints. With the reformulated form, an efficient greedy algorithm is developed, which achieves at least $\frac{1}{2}$ of the optimal value.
- ***Online predictive scheduling policy****:* Furthermore, we design the online predictive scheduling policy with predictive information (future information) on users' file request. Different from the model of JUARA algorithm, a prediction window is introduced and the network have the future information about the users' file request within the prediction window in advance. With such predictive scheduling model, the operator can serve the upcoming requests and pre-push the files to the users beforehand when the link condition is good, which helps to relieve the latency pressure when the network is busy. The predictive user-AP association and resource allocation algorithm is proposed in this part to solve such a problem.
- ***Performance improvement with prediction****:* Performance analysis for JUARA and P-UARA algorithm are both conducted. Both algorithms do not require statistical information of the system dynamics, and have strong theoretical performance guarantee. It is shown that there is a trade off between the average network throughput and queueing delay theoretically through a control parameter V. Moreover, we also prove that P-UARA achieves an average delay reduction that is roughly linear in the prediction window size compared with JUARA. Simulation results show that the predictive information improves the average network throughput and-delay performance significantly. In addition, P-UARA in high workload scenario achieves a higher average queueing delay reduction performance when increasing the prediction window size.

C. Organization

The rest of this paper is organized as follows. In section II, we present our system model and formulate the problem. In section III and section IV, we study the non-predictive and predictive user-AP association and resource allocation, respectively. In section V, we present the simulation results and conclude the paper in section VI.

II. System Model and Problem Formulation

We consider a cache-enabled content-centric wireless network (CCWN) with several fixed wireless access points (APs) and multiple randomly distributed users in the considered region. Each AP is equipped with a cache to store files with different types, which come from a finite file library. Through the backhaul links, APs connect to an operator center, which is also the server of file library. Users request to download files from APs through wireless links. The whole system is shown in Fig.1. We denote the set of APs as $\mathcal{H}$ and its size being H, the set of users as $\mathcal{U}$ and its size being U, the file library as $\mathcal{F}$ and the size of file types being F. We assume that the size of file types cached on each AP is N.

We assume that this wireless caching network operates in a slotted system, indexed by $t \in \{0, 1, 2, \ldots\}$ and the time slot length is $\mathcal{T}$. During every slot, each user u broadcasts its content requests for file with only one type $f \in \mathcal{F}$ to all the APs. The operator center makes decision and associates the users with corresponding APs. Due to backhaul constraints or caching constraints, the AP may not have access to the whole file library. Hence, user u requesting file f can only be downloaded from the AP which have cached it.

A. User-AP association and Cache Placement

We define the user-AP association as a bipartite graph $\mathcal{G} = (\mathcal{U}, \mathcal{H}, \mathcal{E})$, where $\mathcal{E}$ contains edges for all pairs (u, h) such that there exists a potential transmission link between AP $h \in \mathcal{H}$ and user $u \in \mathcal{U}$. We assume $\mathcal{G}$ varies in different time slots. Let $\boldsymbol{X}(t)$ denote the $U \times H$ association matrix of $\mathcal{G}$ between users and APs in time slot t. Then $x_{uh}(t) = 1$ if $(u, h) \in \mathcal{E}$, and 0 otherwise. We denote $\mathcal{N}(h, t) \subseteq \mathcal{U}$ as the set of users that associated with AP $h, \forall h \in \mathcal{H}$ in time slot t.

We also define the cache placement (AP-File Association) as a bipartite graph $\widetilde{\mathcal{G}} = (\mathcal{H}, \mathcal{F}, \widetilde{\mathcal{E}})$, where edges $(h, f) \in \widetilde{\mathcal{E}}$ indicates that files with type f are cached in AP h. For ease of reference, we list the key notation of our system model in TABLE I.

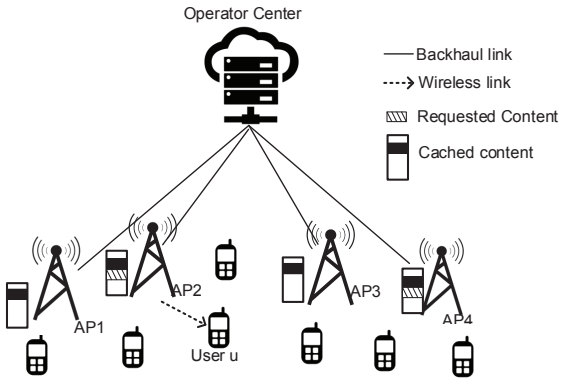


Fig. 1: A sample cache-enabled CCWN. In each time slot, the user $u \in \mathcal{U}$ will request to download the file from APs. If the requested file is not cached on the AP that the user u associating with in this time slot, the downloading rate would be 0.

TABLE I: Summary of key notations

Notation	Description
$\mathcal{U}$	Index set of the users
$\mathcal{H}$	Index set of the APs
$\mathcal{F}$	Index set of the files
$\mathcal{E}(\tilde{\mathcal{E}})$	user-AP (AP-File) association set
M	Maximum connectable users for each AP in one time slot
t	Index set of the time slots
$\mathcal{T}$	The length of one time slot
$x_{uh}(t)$	Association indicator for user u and AP h in time slot t
y_{hf}	Association indicator AP h and file f
$A_u(t)$	Requested files amount by user u in time slot t
$I_{uf}(t)$	File f requested by user u indicator in time slot t
$C_{uh}(t)$	Maximum achievable rate over link (u,h) in time slot t
$\nu_{uh}(t)$	Allocated bandwidth proportion for user u in time slot t
$\mu_u(t)$	Achievable service rate for user u in time slot t
$\mu_{uf}(t)$	Allocated service rate for requested file f in time slot t
$\tilde{\mu}^d_{uf}(t)$	Allocated service rate for arriving request for file f in time slot $t+d$

In this paper, we focus on dynamic user-AP association and source allocation, and assume that the cache placement AP-File Association is fixed during all time slots. Let $\boldsymbol{Y}$ denote the $H \times F$ file placement matrix of $\widetilde{\mathcal{G}}$. $y_{hf} = 1$ if $(h,f) \in \widetilde{\mathcal{E}}$, and 0 otherwise.

We assume that each user can associate with at most one AP and each AP can associate with at most M users in one time slot. Thus we have

$$\sum_{u \in \mathcal{U}} x_{uh}(t) \leq M \ \forall h \in \mathcal{H} \tag{1}$$

$$\sum_{h \in \mathcal{H}} x_{uh}(t) \leq 1 \ \forall u \in \mathcal{U} \tag{2}$$

$$\boldsymbol{X}(t) \in \{0,1\}^{U \times H} \tag{3}$$

B. User Traffic Model

We assume that all users generate file request traffic randomly in each time slot, and such traffic generation is independent of the operator center's operation.

Let $\boldsymbol{A}(t)$ denote the request arriving vector in time slot t and $\boldsymbol{A}^{\mathrm{T}}(t) \triangleq [A_1(t), \cdots, A_U(t)]$. Specially, denote the random variable $A_u(t)$ (with the unit kbits) the requested amount in time slot t. Here we assume that $A_u(t)$ is $i.i.d.$ with $\mathbb{E}\{A_u(t)\} = \lambda_u$, and there exist a positive constant $A_{\max}$ such that:

$$0 \leq A_u(t) \leq A_{\max}. \tag{4}$$

Let $\boldsymbol{I}(t) \triangleq \{I_{uf}(t), u \in \mathcal{U}, f \in \mathcal{F}\}$ denote the $U \times F$ requested file type matrix at time slot t. $I_{uf}(t) = 1$ if user u requests a file with type f in time slot t, and 0 otherwise. We assume that each user can request at most only one type of file in a time slot, which means that the row weight of $\boldsymbol{I}(t)$ is at most 1. The requested probability of each file $f \in \mathcal{F}$ is subject to Zipf distribution [21].

C. The Transmission Model

We assume the wireless channels between users and APs are flat fading channels [22], and all APs transmit at constant powers. Then the maximum backlog that can be served over link $(u,h) \in \mathcal{E}$ in time slot t is given by

$$C_{uh}(t) = \mathcal{T} B_h(t) \cdot \mathbb{E}\left[\log\left(1 + \frac{P_h g_{uh}(t)|s_{uh}(t)|^2}{1 + \sum_{h' \in \mathcal{H} \setminus h} P_{h'} g_{h'u}(t)|s_{uh'}(t)|^2}\right)\right] \tag{5}$$

where $B_h(t)$ is the total bandwidth of AP h in time slot t, P_h is the transmit power of AP h, $g_{uh}(t)$ is the large scale fading from user u to AP h which contains pathloss and shadow, and $s_{uh}(t)$ is the small scale fading gain which follows Rayleigh distribution. Consistently with currently implemented rate adaption schemes [23] [24], we also assume that each AP $h, h \in \mathcal{H}$ is aware of slowly varying pathloss coefficient $g_{uh}(t)$ for all $u \in \mathcal{U}$.

We also assume that each AP h serves for the users $u \in \mathcal{N}(h,t)$ that connected to it by using orthogonal FDMA/TDMA, which is consistent with most current wireless standards. Denote $\boldsymbol{\nu}(t) \triangleq \{\nu_{uh}(t), u \in \mathcal{U}, h \in \mathcal{H}\}$ as the proportion of bandwidth allocated matrix. Specially, $\nu_{uh}(t)$ denote the proportion of bandwidth allocated to the user u by AP h, and satisfies $0 < \nu_{uh} \leq 1$ when $x_{uh}(t) = 1$ otherwise $\nu_{uh} = 0$. Thus we have

$$\sum_{u \in \mathcal{U}} \nu_{uh}(t) x_{uh}(t) \leq 1 \ \forall h \in \mathcal{H}. \tag{6}$$

Let $\mu_u(t)$ denote the amount of backlog that can be served for user u in time slot t, which is called as service rate hereafter. Define $\boldsymbol{\mu}^{\mathrm{T}}(t) \triangleq [\mu_1(t), \cdots, \mu_U(t)]$. Note that each user can associate with at most one AP in a time slot, thus $\mu_u(t)$ can be expressed as below:

$$\mu_u(t) = \sum_{h \in \mathcal{H}} C_{uh}(t) \nu_{uh}(t) x_{uh}(t), \forall u \in \mathcal{U}. \tag{7}$$

D. Definitions of Random Event and Scheduling Policy

Assuming that the channel pathloss coefficient $g_{uh}(t), \ \forall (u,h) \in \mathcal{E}$ changes slowly in time slot t. Then we define the random event observed in time slot t and the feasible scheduling policy for our system.

Definition 1: The random event $\omega(t)$ observed in time slot t contains the slowly-varying channel pathloss coefficients, the amount of new request arrivals and corresponding requested file types. Therefore, we have:

$$\omega(t) = \{g_{uh}(t), A_u(t), I_{uf}(t),\ \forall u \in \mathcal{U}, h \in \mathcal{H}, f \in \mathcal{F}\} \quad (8)$$

Definition 2: The scheduling policy $\{\alpha(t)\}_{t=0}^{\infty}$ is a sequence of control actions $\alpha(t)$ which comprises the user-AP association $\boldsymbol{X}(t)$and bandwidth allocation $\boldsymbol{\nu}(t)$:

$$\alpha(t) = \{\boldsymbol{X}(t), \boldsymbol{\nu}(t)\}. \quad (9)$$

Definition 3: The feasible set of control actions $\mathcal{A}_{\omega(t)}$ in time slot t includes all control actions $\alpha(t)$ such that the constraints (1),(2),(3) and (6) are satisfied simultaneously.

E. Queueing

We assume each user $u \in \mathcal{U}$ has F data queues $\{Q_{uf}(t), f \in \mathcal{F}\}$, where $Q_{uf}(t)$ records the amount of user u's unserved request for files with type f. Define $Q_u^{\text{sum}}(t) \triangleq \sum_{f\in\mathcal{F}} Q_{uf}(t)$ and denote $\boldsymbol{Q}^{\mathrm{T}}(t) = [Q_u^{\text{sum}}(t), \cdots, QU^{\text{sum}}(t)$ the queue length vector. We assume that all queues are initially empty, *i.e.*,

$$Q_{uf}(0) = 0, \forall u \in \mathcal{U}, f \in \mathcal{F} \quad (10)$$

Let $\mu_{uf}(t)$ denote the service rate for requested file f scheduled by the operator center according to a certain rate allocation discipline [25], such as FIFO, LIFO and Random discipline. We adopt fully-efficient scheduling policy for queues, which means:

$$\sum_{f\in\mathcal{F}} \mu_{uf}(t) = \mu(t), \quad (11)$$

where $\mu_u(t)$ is defined in (7) and $\mu_{uf}(t) = 0$ if $y_{hf} = 0$.

We use the notation $[x]^+ = \max\{x, 0\}$. The queue length $Q_{uf}(t)$ is updated in every time slot t according to the following rules:

$$Q_{uf}(t+1) = \left[Q_{uf}(t) - \mu_{uf}(t)\right]^+ + A_u(t) \cdot I_{uf}(t). \quad (12)$$

F. Problem Formulation

Throughout this paper, we use the following standard notation for the time-averaged expectation for any quantity x:

$$\overline{x} = \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[x(\tau)]. \quad (13)$$

Let $\overline{Q}_{uf}$ to be the time-averaged length of the queue. i.e.,

$$\overline{Q}_{uf} = \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[Q_{uf}(\tau)] \quad (14)$$

Let $\overline{\mu}_u$ be the time-averaged expected service rate of user u and define $\overline{\boldsymbol{\mu}} \triangleq [\overline{\mu}_1, \ldots, \overline{\mu}_U]$. Specially,

$$\overline{\mu}_u = \lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[\mu_u(\alpha(\tau), \tau)] \quad (15)$$

Let $\phi(\overline{\boldsymbol{\mu}})$ be the time-averaged network throughput function and $\phi(\overline{\boldsymbol{\mu}}) = \sum_{u\in\mathcal{U}} \overline{\mu}_u$. Thus we have $\phi(\overline{\boldsymbol{\mu}}) = \overline{\phi(\boldsymbol{\mu})}$

The systems objective is to find a feasible scheduling policy $\alpha(t)$ for the system to maximize the average network throughput. Then, the scheduling policy is the solution of the following problem $\mathcal{P}1$:

$$\mathcal{P}1: \max \quad \overline{\phi(\boldsymbol{\mu})} \quad (16)$$

$$\text{s.t.} \quad \overline{Q}_{uf} < \infty\ \forall u \in \mathcal{U}, f \in \mathcal{F} \quad (17)$$

$$\alpha(t) \in \mathcal{A}_{\omega(t)}\ \forall t, \quad (18)$$

where requirement of finite $\overline{Q}_{uf}$ corresponds to the strong stability condition for all the queues [26].

Define ϕ^{opt} as the optimal average network throughput associated with the above problem $\mathcal{P}1$, augmented with the following rectangle constraint:

$$\overline{\boldsymbol{\mu}} \in \mathcal{R} \quad (19)$$

where $\mathcal{R}$ is chosen large enough to contain a time average service rate vector $\overline{\boldsymbol{\mu}}$ that is optimal for the original problem $\mathcal{P}1$

Next, we focus on solving the problem $\mathcal{P}1$ by using the Lyapunov optimization techniques [26], in which a deterministic problem needs to be solved in each time slot.

III. User-AP association and resource allocation without Prediction

In this section, we study the situation where the operator center has only complete information within the current slot and cannot predict the users' behavior for the future time slots. In subsection III-A, we propose the online JUARA algorithm which can be pushed arbitrarily close to the suboptimal value of problem $\mathcal{P}1$. In subsection III-B, we transfer the subproblem of $\mathcal{P}1$ into an equivalent modular maximization problem and provide its solution. In subsection III-C, we analyze the performance of JUARA algorithm. In subsection III-D, we provide the computation complexity analysis for JUARA algorithm.

A. Joint User-AP Association with Resource Allocation (JUARA) algorithm

We define a quadratic Lyapunov function as follows:

$$L(\boldsymbol{Q}(t)) \triangleq \frac{1}{2}\boldsymbol{Q}^{\mathrm{T}}(t)\boldsymbol{Q}(t) = \frac{1}{2}\sum_{u\in\mathcal{U}} (Q_u^{\text{sum}}(t))^2. \quad (20)$$

Then the one-slot Lyapunov drift function can be written as:

$$\begin{aligned} & L(\boldsymbol{Q}(t+1)) - L(\boldsymbol{Q}(t)) \\ = \ & \frac{1}{2}\Big[\boldsymbol{Q}^{\mathrm{T}}(t+1)\boldsymbol{Q}(t+1) - \boldsymbol{Q}^{\mathrm{T}}(t)\boldsymbol{Q}(t)\Big] \end{aligned} \quad (21)$$

Adopt the queue evolution (12), we have:

$$\begin{aligned} & L(\boldsymbol{Q}(t+1)) - L(\boldsymbol{Q}(t)) \\ = \ & \frac{1}{2}\Big\{\big([\boldsymbol{Q}(t) - \boldsymbol{\mu}(t)]^+ + \boldsymbol{A}(t)\big)^{\mathrm{T}}\big([\boldsymbol{Q}(t) - \boldsymbol{\mu}(t)]^+ \\ & + \boldsymbol{A}(t)\big) - \boldsymbol{Q}^{\mathrm{T}}(t)\boldsymbol{Q}(t)\Big\} \end{aligned} \quad (22)$$

We define one-slot conditional Lyapunov drift $\Delta(\boldsymbol{Q}(t))$ as follows:

$$\Delta(\boldsymbol{Q}(t)) \triangleq \mathbb{E}\{L(\boldsymbol{Q}(t+1)) - L(\boldsymbol{Q}(t))|\boldsymbol{Q}(t)\} \tag{23}$$

Thus the one-slot conditional Lyapunov *drift-plus-penalty* function is shown as follows:

$$\Delta_V(\boldsymbol{Q}(t)) = \Delta(\boldsymbol{Q}(t)) - V\mathbb{E}\{\phi(\boldsymbol{\mu}(t))|\boldsymbol{Q}(t)\}, \tag{24}$$

where $V \geq 0$ is the policy control parameter.

Lemma 1: For any feasible control decision $\alpha(t)$ for $\mathcal{P}1$, $\Delta_V(\boldsymbol{Q}(t))$ is upper bounded by

$$\begin{aligned}\Delta_V(\boldsymbol{Q}(t)) \quad &\leq \quad \mathcal{K} - V\mathbb{E}\{\phi(\boldsymbol{\mu}(t))|\boldsymbol{Q}(t)\} \\ &+ \quad \mathbb{E}\{\big(\boldsymbol{A}(t) - \boldsymbol{\mu}(t)\big)^{\mathrm{T}}\boldsymbol{Q}(t)|\boldsymbol{Q}(t)\}\end{aligned} \tag{25}$$

where $\mathcal{K} = \frac{U}{2}[\mu_{\max}^2 + A_{\max}^2]$.

Proof: See Appendix A

Lemma 1 provide the upper bound of conditional Lyapunov *drift-plus-penalty* function $\Delta_V(\boldsymbol{Q}(t))$, which plays a significant role in JUARA algorithm. The control policy in JUARA algorithm is to acquire the information about $\boldsymbol{Q}(t)$ and random event $\boldsymbol{\omega}(t)$ in every slot and make a control decision $\boldsymbol{\alpha}(t) \in \boldsymbol{\mathcal{A}}_{\omega}(t)$ to minimize upper bound of $\Delta_V(\boldsymbol{Q}(t))$, which is shown in $\mathcal{P}2$ (26).

By Lyapunov optimization method [26], to solve Problem $\mathcal{P}1$, it needs an algorithm to minimize the upper bound of *Lyapunov drift-plus-penalty* term at each time slot, as is shown in $\mathcal{P}2$.

$$\begin{aligned}\mathcal{P}2: \min \quad & -V\sum_{u\in\mathcal{U}}\mu_u(t) - \sum_{u\in\mathcal{U}}Q_u^{\text{sum}}(t)\mu_u(t) \\ \text{s.t.} \quad & (1),(2),(3),(6),\end{aligned} \tag{26}$$

where $V(V > 0)$ is the control parameter that trades off the average queue lengths with the accuracy with which the policy is able to approach the optimum of the problem $\mathcal{P}1$.

Rearrange the objective function (26) of $\mathcal{P}2$, we have:

$$-\sum_{u\in\mathcal{U}}\Big[V + Q_u^{\text{sum}}(t)\Big]\mu_u(t) \tag{27}$$

Note that $Q_u^{\text{sum}}(t)$ are observed at the beginning of each time slot, which can be viewed as constants during per-slot. Thus the objective function (27) of $\mathcal{P}2$ only depends on $\mu_u(t)$, which involves the network User-AP association and bandwidth allocation. To simplify the notation, we define $\mathcal{M}_{uh}(t) = [V + Q_u^{\text{sum}}(t)]C_{uh}(t)$, which is a constant during per-slot. By the definition of $\mu_u(t)$ in (7), $\mathcal{P}2$ can be transferred to the following equivalent subproblem:

$$\begin{aligned}\mathcal{P}_{\text{AR}}: \max_{\boldsymbol{\nu}(t),\boldsymbol{X}(t)} \quad & \sum_{h\in\mathcal{H}}\sum_{u\in\mathcal{U}}\mathcal{M}_{uh}(t)\nu_{uh}(t)x_{uh}(t) \\ \text{s.t.} \quad & (1),(2),(3),(6),\end{aligned} \tag{28}$$

The main idea of proposed joint user-AP association and resource allocation (JUARA) algorithm is to solve the $\mathcal{P}_{\text{AR}}$ in each time slot. The solution for $\mathcal{P}_{\text{AR}}$ will be present in the next subsection. By doing so, the amount of request waiting in the queues can be maintained at a small level and the network throughput can be maximized at the same time. The JUARA algorithm is illustrated in Figure 2 and present in Algorithm 1 .

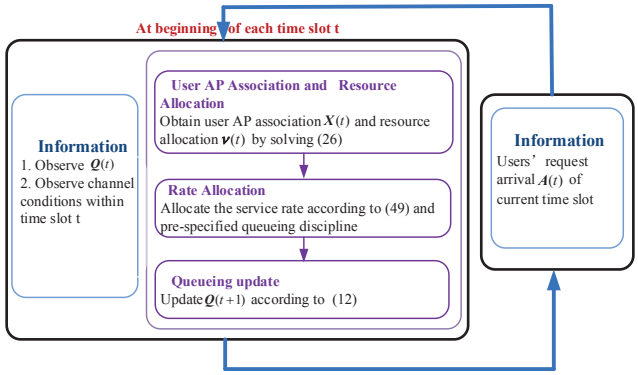


Fig. 2: Flowchart of JUARA algorithm.

Algorithm 1 Joint User-AP association and Resource Allocation (JUARA) algorithm

1: Set t = 0, $\boldsymbol{Q}(0) = 0$;
2: **While** t $< t_{\text{end}}$, **do**
3: At beginning of the tth time slot, observe random events $\omega(t)$ and queues $Q_{uf}(t)$;
4: Obtain $\boldsymbol{X}(t)$ and $\boldsymbol{\nu}(t)$ through solving $\mathcal{P}_{\text{AR}}$;
5: Allocate the service rates $\mu_{uf}(t)$ to the queues $Q_{uf}(t)$ according to (49) and any pre-specified queueing discipline;
6: Update $Q_{uf}(t+1)$ according to (12);
7: t $\leftarrow$ t + 1.
8: **end While**

B. Problem Transformation and Solution

$\mathcal{P}_{\text{AR}}$ is an integer programming problem containing two optimization variables: user-AP association $\boldsymbol{X}(t)$ and bandwidth allocation $\boldsymbol{\nu}(t)$. It is not hard to see that the computational complexity of the brute-force search is prohibitive.

By exploiting the structure information of $\mathcal{P}_{\text{AR}}$, we transfer the problem $\mathcal{P}_{\text{AR}}$ to the following problem $\mathcal{P}'_{\text{AR}}$, which is proved to be equivalent with $\mathcal{P}_{\text{AR}}$.

$$\mathcal{P}'_{\text{AR}}: \max_{\boldsymbol{X}(t)} \quad \sum_{h\in\mathcal{H}}\sum_{u\in\mathcal{U}}\mathcal{M}_{uh}(t)x_{u,h}(t) \tag{29}$$

$$\text{s.t.} \quad \sum_{u\in\mathcal{U}}x_{u,h}(t) \leq 1 \ \forall h \in \mathcal{H}, \tag{30}$$

$$(2),(3),$$

and the corresponding bandwidth allocation $\nu_{uh}(t) = 1$ if $x_{uh} = 1$, and $\nu_{uh}(t) = 0$ otherwise.

Lemma 2 : Problem $\mathcal{P}_{\text{AR}}$ and Problem $\mathcal{P}'_{\text{AR}}$ are equivalent.

Proof: See Appendix B.

Notice that the maximum connectable users is M in problem $\mathcal{P}_{\text{AR}}$ while it becomes 1 in $\mathcal{P}'_{\text{AR}}$.

Next, we show that problem $\mathcal{P}'_{\text{AR}}$ is a modular maximization problem over two matroid constraints. This structure can be exploited to design computationally efficient algorithms

for Problem $\mathcal{P}'_{\text{AR}}$ with provable approximation gaps. The definitions of matroid and submodular function are given in Appendix J.

First, we define the following ground set $\mathcal{E}$ which consists of all possible pairs of user-AP association

$$\mathcal{E} = \{\mathcal{X}_1^1, \mathcal{X}_2^1, \ldots, \mathcal{X}_U^1, \ldots, \mathcal{X}_1^H, \mathcal{X}_2^H, \ldots, \mathcal{X}_U^H\},$$

where $\mathcal{X}_u^h$ denotes the association between user u and AP h.

Lemma 3: The constraint of problem $\mathcal{P}'_{\text{AR}}$ can be written as intersection of two partition matroids $\mathcal{A}_1 = (\mathcal{E}, \mathcal{I}_1)$ and $\mathcal{A}_2 = (\mathcal{E}, \mathcal{I}_2)$ on ground set $\mathcal{E}$.

Proof: See Appendix C.

Next, we let $|\underline{\mathcal{E}}|$ denote the cardinality of the set $\underline{\mathcal{E}}$ for all $\underline{\mathcal{E}} \subseteq \mathcal{E}$. Then we rewrite the objective function in problem $\mathcal{P}'_{\text{AR}}$ as:

$$g(\underline{\mathcal{E}}) = \sum_{u\in\mathcal{U}} \sum_{h\in\mathcal{H}} \mathcal{M}_{uh}(t)\mathcal{X}_u^h, \tag{31}$$

where $\mathcal{X}_u^h \in \underline{\mathcal{E}}$.

Lemma 4 verifies that $g(\underline{\mathcal{E}})$ is a modular function.

Lemma 4: Let $\mathcal{E}_1 \subset \mathcal{E}_2 \subset \mathcal{E}$ and $\mathcal{X}_u^h \in \mathcal{E} - \mathcal{E}_2$, then,

$$g(\mathcal{X}_1 \cup \mathcal{X}_u^h) - g(\mathcal{X}_1) = g(\mathcal{X}_2 \cup \mathcal{X}_u^h) - g(\mathcal{X}_2) \tag{32}$$

Proof: See Appendix D.

Normally, the greedy algorithm provides an effective solution for such modular maximization problem. It is optimal for modular function maximization subject to one matroid constraint [27], and achieves a tight $\frac{1}{p}$-approximation for intersection of p matroid constraints [28]–[30].

The greedy algorithm is described in algorithm 1. Denote $\mathcal{C}$ as the feasible region that meets the constraints (2),(3) and (30) and denote the iteration number as $i = 0, 1, 2, \ldots$. At the ith iteration, $\boldsymbol{X}^{[i]} \subseteq \mathcal{C}$ shall denote the set of user-AP associations, and $\mathcal{V}^{[i]}$ denotes the remaining set, including elements that may be added into the user-AP association $\boldsymbol{X}^{[i+1]}$. We initialize by setting $\boldsymbol{X}^{[0]} = \emptyset$ and $\mathcal{V}^{[0]} = \mathcal{E}$. The greedy algorithm iteratively adds the element $x_{uh} \in \mathcal{V}^{[i]}$ with the highest marginal value while satisfying the matroid constraints at each step. The marginal value of an element $\mathcal{X}_u^h$ is defined as the gain of adding $\mathcal{X}_u^h$ into the user-AP association $\boldsymbol{X}^{[i]}$, given by $\mathcal{J}(\mathcal{X}_u^h) = \mathcal{M}_{uh}(t)$, when $\boldsymbol{X}^{[i]} \in \mathcal{I}_1 \cap \mathcal{I}_2$, $\mathcal{X}_u^h \in \mathcal{E}\backslash\boldsymbol{X}^{[i]}$ and $\boldsymbol{X}^{[i]} \cup \{\mathcal{X}_u^h\} \in \mathcal{I}_1 \cap \mathcal{I}_2$. In our algorithm, once the element $\mathcal{X}_u^h$ is added to $\boldsymbol{X}^{[i]}$, it should be deleted from $\mathcal{V}^{[i]}$. The greedy algorithm stops until $\mathcal{V}^{[i]}$ becomes $\emptyset$ during the iteration.

Algorithm 2 The Greedy Algorithm

1: At beginning of time slot t, observe maximum achievable rate $C_{uh}(t)$, queues $Q_{uf}(t)$, $\forall u \in \mathcal{U}, h \in \mathcal{H}, f \in \mathcal{F}$. Then compute $\mathcal{M}_{uh}(t)$;
2: Initialize $\boldsymbol{X}^{[0]} = \emptyset, \mathcal{V}^{[0]} = \mathcal{E}$ and $i = 0$;
3: **while** $|\mathcal{V}^{[i]}| \neq 0$ **do**
4: $\quad \mathcal{X}_{u^\star}^{h^\star} = \underset{\substack{\mathcal{X}_u^h \in \mathcal{V}^{[i]}, \\ \{\boldsymbol{X}^{[i]} \cup \mathcal{X}_u^h\} \subseteq \mathcal{C}}}{\text{argmax}} \mathcal{J}(\mathcal{X}_u^h)$;
5: $\quad$ Update $\mathcal{V}^{[i+1]} = \mathcal{V}^{[i]} \backslash \mathcal{X}_{u^\star}^{h^\star}$;
6: $\quad$ Update $\boldsymbol{X}^{[i+1]} = \boldsymbol{X}^{[i]} \cup \mathcal{X}_{u^\star}^{h^\star}$;
7: $\quad$ Set $i = i + 1$;
8: **end while**

C. Performance Analysis for JUARA

The greedy algorithm is efficient and guarantees a tight $\frac{1}{2}$-approximation for $\mathcal{P}'_{\text{AR}}$, i.e., the worst case is at least 50% of the optimal solution. The user-AP association $\boldsymbol{X}(t)$ and bandwidth allocation $\boldsymbol{\nu}(t)$ obtained through greedy algorithm is the suboptimal solution to the scheduling problem $\mathcal{P}'_{\text{AR}}$, which we refer to as *imperfect scheduling* [31], [32]. Therefore, at each time slot, the user-AP association policy of $\mathcal{P}_{\text{AR}}$ yields a transmission rate $\boldsymbol{\mu}(t) \in \mathcal{R}$ that satisfies:

$$\begin{aligned} &\sum_{u\in\mathcal{U}} \Big[V + Q_u^{\text{sum}}(t)\Big]\mu_u(t) \\ \geq\ & \beta \max_{\boldsymbol{\mu}(t)\in\mathcal{R}} \Big\{ \sum_{u\in\mathcal{U}} \Big[V + Q_u^{\text{sum}}(t)\Big]\mu_u(t)\Big\}, \end{aligned} \tag{33}$$

for some fixed constant $\beta = \frac{1}{2}$.

The parameter β in (33) can be viewed as a tuning parameter indicating the degree of precision of imperfect scheduling. Notice that when $\beta = 1$ it reduce to the case with perfect scheduling of $\mathcal{P}_{\text{AR}}$. Let $\boldsymbol{\mu}^{*,0}(t)$ denote the optimal solution to $\mathcal{P}_2$. The following problem turns out to be a good reference point for studying the imperfect scheduling.

β-reduced problem:

$$\begin{aligned} \max \quad & \phi(\overline{\boldsymbol{\mu}}) && (34)\\ \text{s.t.} \quad & \boldsymbol{\mu}(t) \in \beta\mathcal{R} && (35)\\ & \overline{Q}_{uf} < \infty,\ \forall u \in \mathcal{U}, f \in \mathcal{F} && (36)\\ & \alpha(t) \in \mathcal{A}_{\omega(t)},\ \forall t. && (37) \end{aligned}$$

Let $\boldsymbol{\mu}^{*,\beta}(t)$ denote the optimal solution to the β-reduced problem above.

Lemma 5: Let $\boldsymbol{\mu}^{*,0}(t)$ be the optimal solution of the $\mathcal{P}1$. Then the solution to the β-reduced problem is

$$\boldsymbol{\mu}^{*,\beta}(t) = \beta\boldsymbol{\mu}^{*,0}(t) \tag{38}$$

Proof: See Appendix E.

Next, we analyze the performance of JUARA under the assumption that the random event $\omega(t)$ is independent and identically distributed (i.i.d) over slots. We denote the state space of $\omega(t)$ by $\overline{\Omega} = \{\omega_1, \omega_2, \cdots, \omega_J\}$. Let π_{ω_j} be the probability that $\omega(t) = \omega_j, j = 1, \cdots, J$. Denote the control action under the $\omega_j \in \overline{\Omega}$ by $\alpha_m^{\omega_j}$ with probability $\varphi_m^{\omega_j}$, where $\sum_m \varphi_m^{\omega_j} = 1$ and $\varphi_m^{\omega_j} \geq 0$. Then we assume that there exist a constant $\epsilon > 0$ in the following *slater-type condition* holds to ensure that JUARA algorithm satisfies the system stability constraints (17), such that:

$$\lambda_u - \sum_{\omega_j} \pi_{\omega_j} \sum_m \varphi_m^{\omega_j} \mu_u(\alpha_m^{\omega_j}) \leq -\epsilon,\ \forall u \in \mathcal{U} \tag{39}$$

$$\phi(\boldsymbol{\mu}(\alpha_m^{\omega_j})) = \phi_\epsilon \tag{40}$$

where ϕ_ϵ is a finite constant. Constraints (39) and (40) are commonly assumed in the network stability problems [26], [25].

***Lemma** 6*: For any alternative policy $\alpha^{\omega_j} \in \mathcal{A}_\omega$, we have:

$$\Delta_V(\boldsymbol{Q}(t)) \leq \mathcal{K} - V\phi_\epsilon - \beta\epsilon \sum_{u\in\mathcal{U}} Q_u^{\text{sum}}(t) \tag{41}$$

$Proof$: See Appendix F

We denote ϕ_{av}^{JUARA} and Q_{av}^{JUARA} as the long term average network throughput and average queue length of JUARA, respectively. Then the performance of JUARA is shown as follows.

Theorem1 : For the system defined in section II, the dynamic scheduling policy obtained through JUARA algorithm, the average network throughput achieves following:

$$\phi_{av}^{\text{JUARA}} \triangleq \liminf_{t\to\infty} \phi(\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{\boldsymbol{\mu}(\tau)\}) \geq \phi_\beta^{\text{opt}} - \frac{\mathcal{K}}{V}, \tag{42}$$

with bounded queue backlog:

$$\begin{aligned} Q_{av}^{\text{JUARA}} &\triangleq \limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{u}\mathbb{E}\{Q_u^{\text{sum}}(t)\} \\ &\leq \frac{\mathcal{K} + V(\phi^{\max} - \phi_\epsilon)}{\beta\epsilon}, \end{aligned} \tag{43}$$

where ϕ_β^{opt} is the optimal expected average network throughput for the $\beta-$ reduced problem in (34) and $\beta = \frac{1}{2}$.

Proof: See Appendix G.

Theorem 1 shows that under the proposed JUARA algorithm, the lower bound of average network throughput increases proportional to V, while the upper bound of average queues increases linearly with V. Hence there exists an $[O(1/V, O(V))]$ tradeoff between these two objects. Through adjusting V, we can balance the network throughput and average delay.

D. Complexity Analysis of JUARA

In each time slot, the computation complexity for JUARA mainly remains in the greedy algorithm. According to Alogrithm1, the computation complexity of the greedy algorithm is

$$\mathcal{O}(UHT_m), \tag{44}$$

where T_m denotes the computation complexity for searching for the element $\mathcal{X}_{u^\star}^{h^\star}$. According to Algorithm 1, greedy algorithm starts with an empty set and at each step it adds one element with highest marginal value to the set while maintaining the feasibility of the solution. Since the objective function is modular, the marginal value of the elements decreases as we add more elements to the set $\boldsymbol{X}$. Thus, if at one iteration, if the largest marginal value is zero, then the algorithm should stop. In our case, it would be at most UH iterations. Each iteration would involve evaluating marginal value of at most UH elements. Then, the computation complexity for searching for the element $\mathcal{X}_{u^\star}^{h^\star}$ is $\mathcal{O}(T_m) = \mathcal{O}(UH)$. Thus the overall computation complexity for the greedy algorithm is

$$\mathcal{O}(U^2H^2). \tag{45}$$

IV. User-AP association and resource allocation with prediction

In this section, we study the scenario where the operator can predict the users' request information within a limited future time window (lookahead window). With such predictive information, the operator can serve the upcoming requests and pre-push the file to the users beforehand when the link condition is good, instead of waiting for them to submit their requests at the same time which may lead to a large service latency.

In subsection IV-A, we introduce the predictive service model based on prediction window. In subsection IV-B, we design P-UARA algorithm. In IV-C and IV-D, we analyze its performance and compare it with JUARA.

A. The Predictive Scheduling Model

We introduce a *lookahead window* [25] in our predictive scheduling model and assume that operator center has accesses to future arrival information $\{A_u(t), \ldots, A_u(t+D_u-1)\}$ and $\{I_{uf}(t), \ldots, I_{uf}(t+D_u-1)\}, \forall u \in \mathcal{U}$ in the *lookahead window*. Here D_u is the prediction window size of user u with $D_u \geq 1$. $\{\tilde{\mu}_{uf}^d(t)\}_{d=0}^{D_u-1}$ is introduced to denote the service rate scheduled by the operator center in time slot t serving for arriving request in time slot $t+d$. Here $d = \{0, 1, \ldots D_u - 1\}$ is the predictive phase. Let $\tilde{\mu}_{uf}^{-1}(t)$ denote the service rate allocated for the file requests that already in the system queues in time slot t.

Next, we introduce prediction queues, which record the residual requests for different type of files in the lookahead window $[t, t+D_u-1]$. Specifically, $\tilde{Q}_{uf}^{-1}(t)$ denote the amount of file request queue already in the system at the beginning of time slot t. $\tilde{Q}_{uf}^d(t)(d = 0, \ldots, D_u - 1)$ denote the amount of file request queue in future slot $t+d$. Define $\tilde{Q}_u^{\text{sum}}(t) \triangleq \sum_{f\in\mathcal{F}}\sum_{d=-1}^{D_u-1}\tilde{Q}_{uf}^d(t)$ and $\tilde{\boldsymbol{Q}}^{\text{T}}(t) \triangleq [\tilde{Q}_1^{\text{sum}}(t), \ldots, \tilde{Q}_U^{\text{sum}}(t)]$. Note that $\tilde{Q}_{uf}^{-1}(t)$ is the only actual backlog in the network and the network is stable if and only if $\tilde{Q}_{uf}^{-1}(t)$ is stable. $\{\tilde{Q}_{uf}^d(t)\}_{d=0}^{D_u-1}$ are virtual queues which simply record the residual arrivals in lookahead window.

The prediction queues $\{\tilde{Q}_{uf}^d\}_{d=-1}^{D_u-1}(u \in \mathcal{U}, f \in \mathcal{F})$ are updated according to the following rules [25]:

1) If $d = D_u - 1$, then:

$$\tilde{Q}_{uf}^d(t+1) = A_u(t+D_u)\cdot I_{uf}(t+D_u) \tag{46}$$

2) If $0 \leq d \leq D_u - 2$, then:

$$\tilde{Q}_{uf}^d(t+1) = \left[\tilde{Q}_{uf}^{d+1}(t) - \tilde{\mu}_{uf}^{d+1}(t)\right]^+ \tag{47}$$

3) If $d = -1$, then:

$$\begin{aligned} \tilde{Q}_{uf}^{-1}(t+1) = &\left[\tilde{Q}_{uf}^{-1}(t) - \tilde{\mu}_{uf}^{-1}(t)\right]^+ \\ &+ \left[\tilde{Q}_{uf}^0(t) - \tilde{\mu}_{uf}^0(t)\right]^+, \end{aligned} \tag{48}$$

where $\tilde{Q}_{uf}^{-1}(0) = 0$ and $\tilde{Q}_{uf}^d(0) = A_u(0+d)\cdot I_{uf}(0+d)$.

Same with JUARA algorithm, predictive scheduling use all the queues $\tilde{Q}_u^{\text{sum}}(t)$ to make a decision including user-AP

association and resource allocation. Different from the system without prediction, in predictive scheduling system allocates rate $\{\tilde{\mu}_{uf}^{d}(t)\}_{d=-1}^{D_u-1}$ serving for the request that already or will arrive in the system according to a certain rate allocation discipline. Fully-efficient scheduling policy is also employed here, thus we have:

$$\sum_{f\in\mathcal{F}}\sum_{d=-1}^{D_u-1}\tilde{\mu}_{uf}^{d}(t)=\mu_u(t), \tag{49}$$

where $\mu_u(t)$ is defined in (7) and $\tilde{\mu}_{uf}^{d}(t)=0$ if $y_{hf}=0$.

B. Predictive User AP association and Resource Allocation (P-UARA) algorithm

Recall that, the basic idea of JUARA in section III is to minimize the upper bound of *Lyapunov drift-plus-penalty* for each time slot *t*. With such predictive scheduling model, we propose P-UARA algorithm to solve the following problem in $\mathcal{P}3$:

$$\begin{aligned}\mathcal{P}3:\min\quad & -\sum_{u\in\mathcal{U}}\Big[V+\tilde{Q}_u^{\text{sum}}(t)\Big]\mu_u(t) \\ \text{s.t.}\quad & (1),(2),(3),(6),\end{aligned} \tag{50}$$

which has the same structure information with $\mathcal{P}2$. Thus the solution of $\mathcal{P}3$ can be obtained through greedy algorithm described in Algorithm 2. The P-UARA algorithm is present in Algorithm 3.

Algorithm 3 Predictive User-AP association and Resource Allocation (P-UARA) algorithm

1: Set t = 0, $\tilde{\boldsymbol{Q}}(0)=0$;
2: **While** t $<t_{\text{end}}$, **do**
3: At beginning of the tth time slot, observe random events $\omega(t)$ and prediction queues $\{\tilde{Q}_{uf}^{d}(t)\}_{d=-1}^{D_u-1}$;
4: Obtain $\boldsymbol{X}(t)$ and $\boldsymbol{\nu}(t)$ through solving $\mathcal{P}3$;
5: Allocate the service rates $\{\tilde{\mu}_{uf}^{d}(t)\}_{d=-1}^{D_u-1}$ to the queues $\{\tilde{Q}_{uf}^{d}(t)\}_{d=-1}^{D_u-1}$ according to (49) and any pre-specified queueing discipline;
6: Update $\tilde{Q}_{uf}^{d}(t+1)$ according to (46)-(48);
7: t $\leftarrow$ t + 1.
8: **end While**

C. Performance Analysis of P-UARA

Similar with JUARA, we characterize the performance of P-UARA algorithm under the i.i.d. system randomness and assume that

$$\lambda_u-\sum_{\omega_j}\pi_{\omega_j}\sum_{m}\varphi_m^{\omega_j}\mu_u(\alpha_m^{\omega_j})\le-\theta,\ \forall u\in\mathcal{U} \tag{51}$$

$$\phi(\boldsymbol{\mu}(\alpha_m^{\omega_j}))=\phi_\theta \tag{52}$$

where $\theta\in(0,\epsilon]$ and $\theta\to0$ if $D_u\to\infty$, ϕ_θ is a finite constant.

Naturally, we have the following relation:

$$\lim_{\theta\to0}\phi_\theta=\phi^{\text{opt}}. \tag{53}$$

***Lemma** 7* For any alternative policy $\alpha^{\omega_j}\in\mathcal{A}_\omega$, we have:

$$\Delta_V^p(\boldsymbol{Q}(t))\le\mathcal{K}-V\phi_\theta-\beta\theta\sum_{u\in\mathcal{U}}\sum_{f\in\mathcal{F}}\sum_{d=-1}^{D_u-1}\tilde{Q}_{uf}^{d}(t). \tag{54}$$

Proof: See Appendix H.

Theorem2 : The P-UARA achieves the following average network throughput:

$$\phi_{av}^{\text{P-UARA}}\ \triangleq\ \liminf_{t\to\infty}\phi\Big(\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{\boldsymbol{\mu}(\tau)\}\Big)\ge\phi_\beta^{\text{opt}}-\frac{\mathcal{K}}{V}, \tag{55}$$

with bounded queue backlog:

$$\begin{aligned}Q_{av}^{\text{P-UARA}}\ &\triangleq\ \limsup_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{u}\mathbb{E}\{\tilde{Q}_u^{\text{sum}}(t)\}\\ &\le\frac{\mathcal{K}+V(\phi^{\text{opt}}-\phi_\theta)}{\beta\theta},\end{aligned} \tag{56}$$

where $\mathcal{K}=\frac{U}{2}(\mu_{\max}^2+A_{\max}^2),\beta=\frac{1}{2}$.

Proof: See Appendix I.

Notice that the system average queue size $Q_{av}^{\text{P-UARA}}$ includes the true backlog $\tilde{Q}_{uf}^{-1}(t)$ and prediction queues $\{\tilde{Q}_{uf}^{d}(t)\}_{d=0}^{D_u-1}$. Theorem 2 also demonstrates the tradeoff between average network throughput $\phi_{av}^{\text{P-UARA}}$ and average system backlog $Q_{av}^{\text{P-UARA}}$ in P-UARA algorithm.

In the following, we will show the average backlog reduction due to prediction in P-UARA. To do so, we will use a theorem from [33], which show that the queue vector of the network is within $O(\log(V))$ distance from a fixed point. Firstly, we define the following optimization problem:

$$\max:\quad g(\boldsymbol{\ell}),\qquad \text{s.t. }\boldsymbol{\ell}\succeq0, \tag{57}$$

where $g(\boldsymbol{\ell})$ is called the dual function with scaled objective (by V) of original problem that maximize the average network throughput. $\boldsymbol{\ell}=[l_1,\ldots,l_U]$ is the Lagrange multiplier. $g(\boldsymbol{\ell})$ is defined as below:

$$\begin{aligned}g(\boldsymbol{\ell})\ =\ &\sum_{\omega_j}\pi_{\omega_j}\inf_{\boldsymbol{\mu}(\alpha_m^{\omega_j})}\Big\{V\phi\big(\sum_m\varphi_m^{\omega_j}\boldsymbol{\mu}(\alpha_m^{\omega_j})\big)\\ &+\sum_{u\in\mathcal{U}}l_u[\lambda_u-\sum_m\varphi_m^{\omega_j}\mu_u(\alpha_m^{\omega_j})]\Big\}\end{aligned}$$

Let $\boldsymbol{\ell}^*$ denote the optimal solution of (57) and $\boldsymbol{\ell}^*$ is either $\Gamma(V)$ or 0 according to [33]. Now we have the following Theorem 3, which is Theorem 1 in [33].

Theorem3 : Suppose that (i) $\boldsymbol{\ell}^*$ is unique and dual function $g(\boldsymbol{\ell})$ satisfies:

$$g(\boldsymbol{\ell}^*)\ge g(\boldsymbol{\ell})+L\parallel\boldsymbol{\ell}^*-\boldsymbol{\ell}\parallel,\qquad\forall\boldsymbol{\ell}\succeq0, \tag{58}$$

for some constant $L>0$ independent of V,(ii) the $\theta-$ slack condition (51) is satisfied with $\theta>0$. Then, there exist constants G, K, c, such that for any $m\in\mathbb{R}_+$,

$$\mathcal{P}_r(G,Km)\le ce^{-m}, \tag{59}$$

where $\mathcal{P}_r(G,Km)$ is defined as

$$\begin{aligned}&\mathcal{P}_r(G,Km)\triangleq\\ &\limsup_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}Pr\{\exists u|\tilde{Q}_u^{\text{sum}}(\tau)-l_u^*|>G+Km\}.\end{aligned} \tag{60}$$

Proof: See [33].

Next, we state Theorem 4 below regarding the average backlog reduction due to predictive scheduling.

Theorem4 : Suppose that (i) the assumption in Theorem 3 hold, (ii) there exists a steady-state distribution of $\tilde{\boldsymbol{Q}}(t)$ under P-UARA, (v) $D_u = O\big(\frac{1}{A_{\max}}[l^*_{uf} - G - K(\log(V))^2 - \mu_{\max}]^+\big)$ for all $u \in \mathcal{U}$, (iv) FIFO is used in P-UARA. Then P-UARA achieves the following with a sufficiently large V:

$$\tilde{Q}_{av}^{-1} \leq Q_{av}^{\text{JUARA}} - \sum_{u\in\mathcal{U}} D_u\big[\lambda_u - O(\frac{1}{V^{log(V)}})\big]^+. \tag{61}$$

Proof: See [25].

Theorem 4 shows that the average system true queue length is roughly reduced by $\sum_{u\in\mathcal{U}} \lambda_u D_u$.

D. Comparison between JUARA and P-UARA

Comparing Theorem 1 and Theorem 2, we find P-UARA in (42) achieves similar performance bounds as JUARA. In particular:

- The bound for network throughput by P-UARA equals that of JUARA in (42). That is,

$$\phi_{av}^{\text{P-UARA}} = \phi_{av}^{\text{JUARA}} \tag{62}$$

  This is interesting and shows that predictive scheduling does not improve the average network throughput for the system. Instead, this conclusion, together with Theorem 4 above, deliver an important message that predictive scheduling improves the system delay given the same average network throughput.

- when $\theta = \epsilon$, the average queue length of P-UARA achieves the same bound as that of JUARA:

$$\begin{aligned} Q_{av}^{\text{P-UARA}} &\leq \frac{\mathcal{K} + V(\phi^{\text{opt}} - \phi_\theta)}{\beta\theta} \\ &= \frac{\mathcal{K} + V(\phi^{\text{opt}} - \phi_\epsilon)}{\beta\epsilon} \end{aligned} \tag{63}$$

  where the right hand bound is the same as that specified in (43) for JUARA.

V. Simulation Results

In subsection V-A, we explain the simulation settings. In subsection V-B, we simulate the average network throughput and average system backlog performance for JUARA and P-UARA algorithm, respectively.

A. Simulation Settings

Consider a network with $H = 9$ fixed APs and $U = 100$ randomly deployed users. Each AP is associated with at most $M = 12$ users. The system region has a size of 15×15 m^2. Each user requests files with type $f \in \mathcal{F}$ according to the Zipf distribution with parameter η_r, i.e., $p_f = \frac{f^{-\eta_r}}{\sum_{i\in\mathcal{F}} i^{-\eta_r}}$ [10], [21], [34]. In our simulation, we set $\eta_r = 0.56$. We set the length of lookahead window $D_u \in \{0, 5, 10, 15\}$. The simulation results are averaged over 1000 constant time slots (over $\mathcal{T} = 10$ milliseconds intervals).

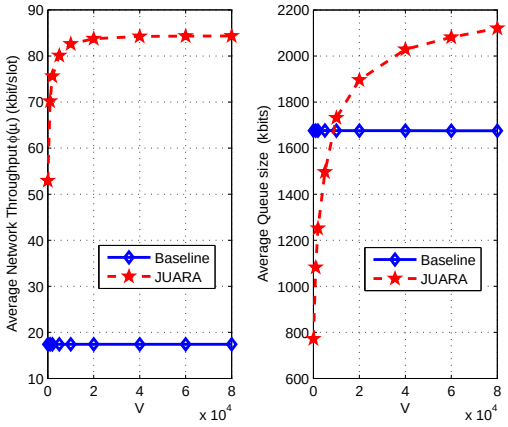


Fig. 3: Average network throughput/average queue size per users vs. the control parameter V, $A_{\max} = 100$kbits.

We assume that each AP operates on a 18 MHz band (100 source block with 180 kHz of each source block) and transmits at a fixed power level $P = 10^8$. Basing on the WINNER II channel model with small-cell scenario in [35], the path loss coefficients between AP h and user u is defined by:

$$g_{uh}(t) = 10^{-\frac{\text{PL}(d_{uh}(t))}{10}}, \tag{64}$$

where d_{uh} is the distance from user u to AP h at time slot t, and

$$PL(d) = Alog(d) + B + Clog(f_0/5) + \mathcal{X}_{dB} \tag{65}$$

where f_0 is the carrier frequency, $\mathcal{X}_{dB}$ is a shadowing log-normal variable with variance σ^2_{dB}; A = 18.7, B = 46.8, C = 20, and $\sigma^2_{dB} = 9$ in line-of-sight (LOS) condition; A = 36.8, B = 43.8, C = 20, and $\sigma^2_{dB} = 16$ in non-line-of-sight (NLOS) condition. Each link is in LOS or NLOS independently and randomly, with probability $p_l(d)$ and $1 - p_l(d)$, respectively, where

$$p_l(d) = \begin{cases} 1 & d \leq 2.5m \\ 1 - 0.9(1 - (1.24 - 0.6\log(d))^3)^{1/3} & \text{otherwise} \end{cases} \tag{66}$$

B. Simulation Results

1) Performance comparison between JUARA algorithm and Baseline algorithm: In this part, we compare JUARA with the following baseline case. We set $A_{\max} = 100$kbits in both case.

Baseline case: At beginning of the algorithm, each user connects to one AP in the network randomly. The maximum number of connected users for each AP is $M = 12$. In each time slot, the network operates on fixed User-AP association and each AP allocates bandwidth to its connected users equally.

In Fig.3, we compare the average network throughput/average queue size performance of JUARA under different parameter V with the baseline case. We see from the left plot that average network throughput of JUARA increases as V increases and converges when V is sufficiently large. According to (42), the lower bound of ϕ_{av}^{JUARA} increases with

the increasing of V, which is consistent with our observation here. The left plot also shows that the average network throughput of baseline case, which is independent of V. It can be noticed that JUARA obtains larger average network throughput than baseline case for any V.

The right plot illustrates the network average queue backlog against V under JUARA and baseline algorithm. It is not hard to see that average network queue backlog of JUARA scales as $O(V)$. Compared with the baseline case, JUARA generates less queue for any $V < 9.9\times10^3$. These observations verify the $[O(1/V), O(V)]$ tradeoff between average network throughput and average queue backlog as shown in Theorem 1.

2) Performance comparison between JUARA and P-UARA algorithm: In Fig.4, we compare the performance of JUARA and P-UARA. FIFO rate allocation discipline is adopted and the network workload is medium with $A_{\max} = 100$kbits in both algorithms. Fig.4(a) demonstrates the average network throughput against V for JUARA and P-UARA with different prediction window size $D \in \{5, 10, 15\}$. We observe from Fig.4(a) that JUARA and P-UARA with different prediction window size achieves the same average network throughput, which is consistent with (62) that the predictive scheduling doesn't improve the average network throughput.

In Fig.4(b), we plot the average queue backlog against V for both algorithms. We observe that P-UARA always generates a smaller average queue backlog than JUARA under the same traffic. Fig.4(b) also shows that as the predictive window size D increases, the average queue size decreases almost linearly in D. We also observe that the performance improvement increases with the size of the prediction window. The reason is that the predictive information helps the operator design a better user-AP association and resource allocation strategy, which utilizes the network more efficiently.According to (61) in Theorem 4, average system actual queue length of P-UARA is roughly reduced by $\sum_{u\in\mathcal{U}} \lambda_u D_u$.

In Fig.4(c), we compare the performance of JUARA with P-UARA under different prediction errors. For example, P-UARA with 20% prediction error means that the information (i.e., file request amount and type) for future time slots is predicted with 0.8 accuracy rate and the operator center may make a decision with the incorrect value. We investigate the average network throughput-delay tradeoff for both algorithms. We observe that the average network throughput-delay performance for P-UARA declines as the prediction error rate increases. However, P-UARA with 20% prediction error still achieves a better average network throughput-delay tradeoff, which shows the robustness of P-UARA against the prediction errors.

3) P-UARA's performance under different workload: In Fig.5, we investigate the P-UARA's performance under low, medium, and high workloads ($A_{\max} = 50$kbits, 100kbits and 200kbits respectively). FIFO policy is applied in P-UARA.

Fig.5(a) shows the average network throughput for P-UARA with prediction window size $D = 5$. We observe that when V is large enough, the P-UARA algorithm with low workloads achieves the highest average network throughput, and P-UARA with high workload achieves the lowest average network throughput by contrast. According to (55), the low bound of average network throughput $\phi_{av}^{\text{P-UARA}}$ is determined by parameters V and $\mathcal{K}$, where $\mathcal{K} = \frac{U}{2}(\mu_{\max}^2 + A_{\max}^2)$. Larger workload leads to larger $\mathcal{K}$, thus a smaller average network throughput is obtained.

Fig.5(c) investigates the average actual queue backlog of P-UARA under different workload and different prediction window size $(D = 5, 10, 15)$. We observe that P-UARA generates larger average queue size under a higher workload. Fig.5(c) also illustrates that P-UARA algorithm performs better in high workload scenarios. For example, when $V = 4 \times 10^4$, the average queue size for P-UARA under high workload decreases about 18% when prediction window increases from $D = 5$ to $D = 15$, whereas it decreases 11% under low workload. The size of prediction window (lookahead window) has the largest impacts on the average queue size under high workload. In contrast, the size of prediction window (lookahead window) has nearly no impact on the average queue size under low workload. Hence, the P-UARA policy can be used to relieve the network average delay in dense or resource limited networks.

VI. Conclusions

In this paper, we focused on online user-AP association problem with fixed cache placement in wireless caching network. We exploited the structure information of the problem and first proposed JUARA algorithm. Such algorithm is an efficient online algorithm that does not rely on the statistics of users' file requests. The core of JUARA is an efficient greedy approximation algorithm that solves the subproblem of user-AP association and bandwidth allocation. Then given the availability of limited future (prediction) information, we introduced the predictive service model and proposed the P-UARA algorithm. Performance analysis was conducted for the proposed algorithms, which both explicitly characterize the $[O(1/V, O(V))]$ tradeoff between the average network throughput and average delay. Simulation results validated the theoretical analysis and showed that the future (prediction) information helps the operator achieves a much better performance of average network delay.

Appendix A

Notice that for any $Q \geq 0, \mu \geq 0, A \geq 0$, we have:

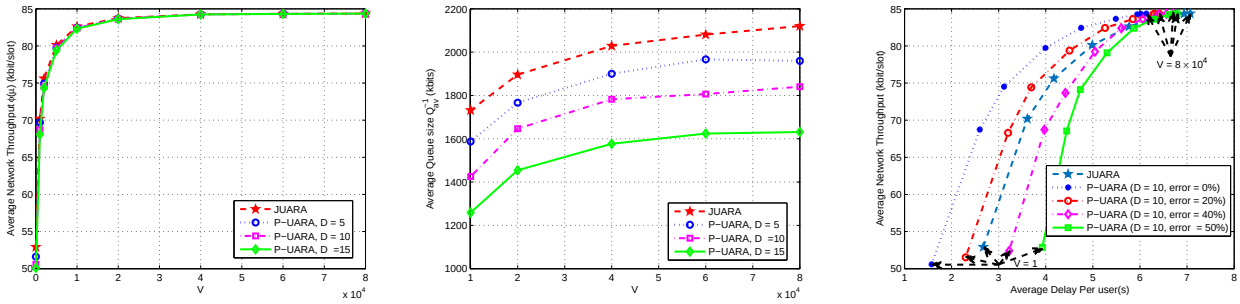
$$\big([Q-\mu]^+ + A\big)^2 \leq Q^2 + A^2 + \mu^2 + 2Q(A-\mu). \quad (67)$$

Plugging the inequations (67) into the Lyapunov drift function (22), we have

$$\begin{aligned} &L(\boldsymbol{Q}(t+1)) - L(\boldsymbol{Q}(t)) \\ &\leq \frac{1}{2}\boldsymbol{A}(t)^{\mathrm{T}}\boldsymbol{A}(t) + \big(\boldsymbol{A}(t) - \boldsymbol{\mu}(t)\big)^{\mathrm{T}}\boldsymbol{Q}(t) + \frac{1}{2}\boldsymbol{\mu}(t)^{\mathrm{T}}\boldsymbol{\mu}(t) \\ &\leq \mathcal{K} + \big(\boldsymbol{A}(t) - \boldsymbol{\mu}(t)\big)^{\mathrm{T}}\boldsymbol{Q}(t) \end{aligned} \quad (68)$$

where $\mathcal{K} = \frac{U}{2}[\mu_{\max}^2 + A_{\max}^2]$.

Taking conditional expectation of (68) and adding on both side the penalty term $-V\mathbb{E}\{\phi(\boldsymbol{\mu}(t))|\boldsymbol{Q}(t)\}$, we can proof the *lemma 1*.

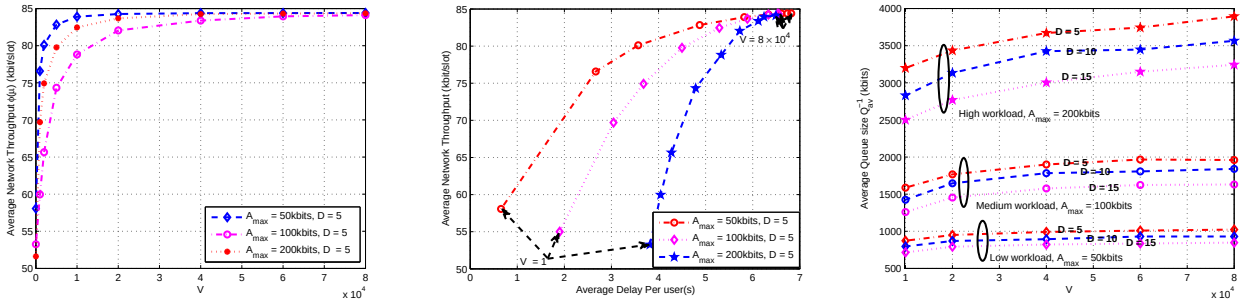


(a) Average Network Throughput vs. Parameter V.

(b) Average Queue size vs. Parameter V.

(c) Network Throughput-Delay Tradeoff under Prediction Errors

Fig. 4: Comparison of JUARA and P-UARA algorithm, $A_{\max} = 100$kbits.



(a) Average Network Throughput vs. Parameter V.

(b) Network Throughput-Delay Tradeoff under different workload.

(c) Average Queue size performance under different workload and prediction window.

Fig. 5: P-UARA's performance under High, Medium, Low workloads.

APPENDIX B

According to the constraints that $0 < \nu_{uh}(t) \leq 1$ when $x_{uh}(t) = 1$ and $\nu_{uh}(t) = 0$ if $x_{uh}(t) = 0$, we know that if and only if user u associates with the AP h ($x_{u,h} = 1$), it will be allocated bandwidth by the AP h

Through proof by contradiction, we prove that each AP $h \in \mathcal{H}$ associates with only one user in time slot t for the optimal user-AP association $\boldsymbol{X}^{\star}(t)$.

We first assume that under the optimal user-AP association $\boldsymbol{X}^{\star}(t)$, the AP h associates with more than one user, i.e., $x^{\star}_{u_1h} = 1$, $x^{\star}_{u_2h} = 1$, $x^{\star}_{uh} = 0$ where $u \in \mathcal{U}\backslash\{u_1, u_2\}$. Thus we have $\nu^{\star}_{u_1h}(t) \geq 0$, $\nu^{\star}_{u_2h}(t) \geq 0$, $\nu^{\star}_{u_1h}(t) + \nu^{\star}_{u_2h}(t) = 1$ and $\nu^{\star}_{uh}(t) = 0$ where $u \in \mathcal{U}\backslash\{u_1, u_2\}$.

We also assume that $\mathcal{M}_{u_1h}(t) \geq \mathcal{M}_{u_2h}(t)$.

Let $x'_{u_1h}(t) = 1$, $x'_{u_2h}(t) = 0$ and $\nu'_{u_1h}(t) = \nu^{\star}_{u_1h}(t) + \nu^{\star}_{u_2h}(t) = 1$, $\nu'_{u_2h}(t) = 0$, then we have that:

$$\begin{aligned} & \mathcal{M}_{u_1h}(t) = \mathcal{M}_{u_1h}(t)\nu'_{u_1h}(t) \\ \geq \quad & \mathcal{M}_{u_1h}(t)\nu^{\star}_{u_1h}(t) + \mathcal{M}_{u_2h}(t)\nu^{\star}_{u_2h}(t), \end{aligned}$$

Thus we can conclude that $x'_{u_1h}(t) = 1$, $x'_{u_2h}(t) = 0$ and $\nu'_{u_1h}(t) = 1, \nu'_{u_2h}(t) = 0$ would be the optimal solution of $\mathcal{P}'_{\text{AR}}$, where the AP h associates with the user u with the maximum $\mathcal{M}_{uh}(t)$, and allocates the whole bandwidth to it, which is contradictory with the assumption before.

Extend this to all the AP $h \in \mathcal{H}$, in which case the optimal user-AP association can be obtained through solving the optimization problem $\mathcal{P}'_{\text{AR}}$.

APPENDIX C

The ground set $\mathcal{E}$ can be partitioned into H disjoint subsets: $S_1, \ldots, S_H$, where $S_h = \{\mathcal{X}^h_1, \ldots, \mathcal{X}^h_U\}$ is the set of all users that might associate with AP h.

The user-AP association is expressed by the matrix $\boldsymbol{X}(t)$. We define the user-AP association set $X \subseteq \mathcal{E}$ such that $\mathcal{X}^h_u \in X$ if and only if $x_{uh}(t) = 1$. Notice that the nonzero elements of the hth column of matrix $\boldsymbol{X}(t)$ equals to the elements in $X \subseteq S_h$. Thus the constraint of the column of matrix $\boldsymbol{X}(t)$ can be expressed as $X \subseteq \mathcal{I}_1$, where

$$\mathcal{I}_1 = \{X \subseteq \mathcal{E} : |X \cap S_h| \leq 1, \forall h = 1, \ldots, H\}. \tag{69}$$

Comparing $\mathcal{I}_1$ with the definition of partition matroid in (90), we can see that the constraints (30) form a partition matroid with $l = H$ and $k_i = 1$ for $i = 1, \ldots, H$. Denote this partition matroid by $\mathcal{A}_1 = (\mathcal{E}, \mathcal{I}_1)$.

Similarly, the ground set $\mathcal{E}$ can also be partitioned into U disjoint subsets: $S'_1, \ldots, S'_U$, where $S'_u = \{\mathcal{X}^1_u, \ldots, \mathcal{X}^H_u\}$ is the set of all AP that might associate with user u. The constraint of the row of matrix $\boldsymbol{X}(t)$ can be expressed as $X \subseteq \mathcal{I}_2$, where

$$\mathcal{I}_2 = \{X \subseteq \mathcal{E} : |X \cap S'_u| \leq 1, \forall u = 1, \ldots, U\}. \tag{70}$$

Hence the constraint (2) form a partition matroid with $l = U$ and $k_i = 1$ for $i = 1, \ldots, U$. This partition matroid is denoted by $\mathcal{A}_2 = (\mathcal{E}, \mathcal{I}_2)$.

To sum up, the constraint of problem $\mathcal{P}'_{\text{AR}}$ can be expressed as two partition matroids on a ground set $\mathcal{X}$.

APPENDIX D

It's easily to be verified that the function $h(x) = x$ is strictly increasing and linear for all $x > 0$. This means that we have

$$h(x+1) - h(x) = h(y+1) - h(y),\ \forall 0 < x < y \tag{71}$$

Combining (71) with the facts that $h(0) = 0$ and $f(\underline{\mathcal{E}}) = h(|\underline{\mathcal{E}}|)$ (so that $f(\emptyset) = 0$, where $\emptyset$ is empty set), we can conclude that

$$\begin{aligned} f(\underline{\mathcal{E}} \cup (u,h)) - f(\underline{\mathcal{E}}) = f(\underline{\mathcal{E}}^{'} \cup (u,b)) - f(\underline{\mathcal{E}}^{'}), \\ \forall \underline{\mathcal{E}} \subseteq \underline{\mathcal{E}}^{'} \subseteq \mathcal{E} \& (u,b) \in \mathcal{E} \backslash \underline{\mathcal{E}}^{'} \end{aligned} \tag{72}$$

which yields the desired result. Then we know that the objective function in Problem $\mathcal{P}_{\text{AR}}$ is a modular function.

APPENDIX E

In the β-reduced problem (34), do a change of variables $\boldsymbol{\mu}'(t) = \boldsymbol{\mu}(t)/\beta$. Using the fact that

$$\phi(\boldsymbol{\mu}(t)) = \sum_{u \in \mathcal{U}} \mu_u(t), \tag{73}$$

we have

$$\phi(\mu_u(t)) = \beta \mu'_u(t). \tag{74}$$

one can show that β-reduced problem becomes equivalent to the problem $\mathcal{P}1$. Hence, $\boldsymbol{\mu}^{*,\beta}(t) = \beta \boldsymbol{\mu}^{*,0}(t)$.

APPENDIX F

Because the JUARA algorithm comes from the minimization of the right-hand-side of (25), for any alternative (possibly randomized) imperfect scheduling policy $\alpha^{\omega_j} \in \mathcal{A}_\omega$, we have

$$\begin{aligned} &\Delta_V(\boldsymbol{Q}(t)) \leq \mathcal{K} - V\phi(\boldsymbol{\mu}^{*,\beta}(t)) \\ &+ \sum_{u \in \mathcal{U}} Q_u^{\text{sum}}(t) \mathbb{E}\{A_u(t)(t) - \mu_u^{*,\beta}(t) | \boldsymbol{Q}(t)\} \end{aligned} \tag{75}$$

where $\boldsymbol{\mu}^{*,\beta}(t) = [\mu_1^{*,\beta}(t), \ldots, \mu_U^{*,\beta}(t)]$ is from the imperfect scheduling policy $\alpha^{\omega_j} \in \mathcal{A}_\omega$.

Plugging (38) into last term of right-hand-side of (75), and rearranging its terms, we have:

$$\begin{aligned} &\Delta_V(\boldsymbol{Q}(t)) \leq \mathcal{K} - V\phi(\boldsymbol{\mu}^{*,\beta}(t)) \\ &+ \beta \sum_{u \in \mathcal{U}} Q_u^{\text{sum}}(t) \mathbb{E}\{A_u(t)(t) - \mu_u^{*,0}(t) | \boldsymbol{Q}(t)\} \end{aligned} \tag{76}$$

Plugging the *slater-type conditions* (39) and (40), using $\boldsymbol{\mu}^{*,\beta}(t) = \beta \mathbb{E}\{\boldsymbol{\mu}^{*,0}(\alpha^{\omega_j})\}$ into the right-hand-side of (75), we have

$$\begin{aligned} \Delta(\boldsymbol{Q}(t)) \quad &- \quad V\mathbb{E}\{\phi(\boldsymbol{\mu}(t)) | \boldsymbol{Q}(t)\} \leq \mathcal{K} - V\phi_\epsilon \\ &- \quad \beta\epsilon \sum_{u \in \mathcal{U}} Q_u^{\text{sum}}(t) \end{aligned} \tag{77}$$

which prove the lemma 6.

APPENDIX G

Taking expectation of (77) and using the law of iterated expectations yields:

$$\begin{aligned} &\mathbb{E}\{L(\boldsymbol{Q}(\tau+1))\} - \mathbb{E}\{L(\boldsymbol{Q}(\tau))\} - V\mathbb{E}\{\phi(\boldsymbol{\mu}(\tau))\} \leq \mathcal{K} \\ &- V\phi_\epsilon - \epsilon \sum_{u \in \mathcal{U}} \mathbb{E}\{Q_u^{\text{sum}}(\tau)\} \end{aligned} \tag{78}$$

Summing the above over $\tau \in \{0, 1, \cdots, t-1\}$ for some slot $t > 0$ and using the law of telescoping sums yields:

$$\begin{aligned} &\mathbb{E}\{L(\boldsymbol{Q}(t))\} - \mathbb{E}\{L(\boldsymbol{Q}(0))\} - V \sum_{\tau=0}^{t-1} \mathbb{E}\{\phi(\boldsymbol{\mu}(\tau))\} \\ &\leq \mathcal{K}t - Vt\phi_\epsilon - \beta\epsilon \sum_{\tau=0}^{t-1} \sum_{u \in \mathcal{U}} \mathbb{E}\{Q_u^{\text{sum}}(\tau)\} \end{aligned} \tag{79}$$

Note that $\epsilon > 0$, then dividing (79) by $t\epsilon$, rearranging terms, and using the fact $\mathbb{E}\{L(\boldsymbol{Q}(t))\} > 0$ yields:

$$\begin{aligned} \frac{1}{t} \sum_{\tau=0}^{t-1} \sum_{u \in \mathcal{U}} \sum_{f \in \mathcal{F}} \mathbb{E}\{Q_{uf}(\tau)\} \quad &\leq \quad \frac{\mathcal{K} + V\left[\overline{\phi(\boldsymbol{\mu}(t))} - \phi_\epsilon\right]}{\beta\epsilon} \\ &- \quad \frac{\mathbb{E}\{L(\boldsymbol{Q}(0))\}}{\beta\epsilon t} \end{aligned} \tag{80}$$

where $\overline{\phi(\boldsymbol{\mu}(t))} = \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{\phi(\boldsymbol{\mu}(\tau))\}$. Note that $\overline{\phi(\boldsymbol{\mu}(t))} \leq \phi^{\text{opt}}$, and taking a lim sup of both sides of (80), we have for all t:

$$\limsup_{t \to \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \sum_{u} \{Q_u^{\text{sum}}(t)\} \leq \frac{\mathcal{K} + V(\phi^{\text{opt}} - \phi_\epsilon)}{\beta\epsilon}, \tag{81}$$

Such that proof (43) in Theorem 1.

Now we consider the policy $\alpha^{\omega_j}(t)$ which achieves the optimal solution ϕ_β^{opt} to the $\beta-$reduced problem $\mathcal{P}1$. Then we have

$$\begin{aligned} &\mathbb{E}\{L(\boldsymbol{Q}(\tau+1))\} - \mathbb{E}\{L(\boldsymbol{Q}(\tau))\} - V\mathbb{E}\{\phi(\boldsymbol{\mu}(\tau))\} \\ &\leq \mathcal{K} - V\phi_\beta^{\text{opt}} \end{aligned} \tag{82}$$

Summing the above over $\tau \in \{0, 1, \cdots, t-1\}$ for some slot $t > 0$ and using the law of telescoping sums yields:

$$\begin{aligned} &\mathbb{E}\{L(\boldsymbol{Q}(t))\} - \mathbb{E}\{L(\boldsymbol{Q}(0))\} - V \sum_{\tau=0}^{t-1} \mathbb{E}\{\phi(\boldsymbol{\mu}(\tau))\} \\ &\leq \mathcal{K}t - V\phi_\beta^{\text{opt}} t \end{aligned} \tag{83}$$

Dividing by tV, rearranging terms, we have:

$$\begin{aligned} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\{\phi(\boldsymbol{\mu}(\tau))\} \quad &\geq \quad \phi_\beta^{\text{opt}} - \frac{\mathcal{K}}{V} + \frac{\mathbb{E}\{L(\boldsymbol{Q}(t))\}}{Vt} \\ &- \quad \frac{\mathbb{E}\{L(\boldsymbol{Q}(0))\}}{Vt} \qquad (84) \\ &\geq \quad \phi_\beta^{\text{opt}} - \frac{\mathcal{K}}{V} - \frac{\mathbb{E}\{L(\boldsymbol{Q}(0))\}}{Vt} \qquad (85) \end{aligned}$$

where the fact that $\mathbb{E}\{L(\boldsymbol{Q}(t))\}$ is used in (84) to get (85).

Taking the lim inf as $t \to \infty$ of both side of (85) proves (42) in *Theorem 1*.

APPENDIX H

Applying the upper bound of *drift-plus-penalty* term for each time slot in (50), we can easily prove Lemma 7 under the *slater-type condition* (51) and (52).

APPENDIX I

According to Lemma 7, P-UARA guarantees that

$$\Delta(\boldsymbol{Q}(t)) - V\mathbb{E}\{\phi(\boldsymbol{\mu}(t))|\boldsymbol{Q}(t)\} \leq \mathcal{K} - V\phi_\theta - \beta\theta\sum_{u\in\mathcal{U}}\tilde{Q}_u^{\text{sum}}(t). \tag{86}$$

Taking expectation of both side of (86) and using the law of iterated expectations, we have:

$$\mathbb{E}\{L(\boldsymbol{Q}(\tau+1))\} - \mathbb{E}\{L(\boldsymbol{Q}(\tau))\} - V\mathbb{E}\{\phi(\boldsymbol{\mu}(\tau))\} \leq \mathcal{K} - V\phi_\theta - \beta\theta\sum_{u\in\mathcal{U}}\tilde{Q}_u^{\text{sum}}(\tau) \tag{87}$$

Using the law of telescoping sums of both sides of (87) yields:

$$\mathbb{E}\{L(\boldsymbol{\Theta}(t))\} - \mathbb{E}\{L(\boldsymbol{\Theta}(0))\} - V\sum_{\tau=0}^{t-1}\mathbb{E}\{\phi(\boldsymbol{\gamma}(\tau))\} \leq \mathcal{K}t - V\phi_\theta t - \beta\theta\sum_{\tau=0}^{t-1}\sum_{u\in\mathcal{U}}\tilde{Q}_u^{\text{sum}}(\tau) \tag{88}$$

Similar with the proof of Theorem 1, we eventually proof the (56) in Theorem 2 taking a lim up of both sides of above inequality.

Now we consider the policy $\alpha^{\omega_j}(t)$ which achieves the optimal solution ϕ_β^{opt} to the $\beta-$reduced problem $\mathcal{P}1$. Then we have

$$\mathbb{E}\{L(\boldsymbol{Q}(\tau+1))\} - \mathbb{E}\{L(\boldsymbol{Q}(\tau))\} - V\sum_{\tau=0}^{t-1}\mathbb{E}\{\phi(\boldsymbol{\mu}(\tau))\} \leq \mathcal{K}t - V\phi_\beta^{\text{opt}}t \tag{89}$$

Dividing by tV, rearranging terms and taking limits for the both sides of inequality as $t \to 0$, we prove (55) in Theorem 2.

APPENDIX J
BASIC DEFINITIONS

Matroids: Matroids are structures that generalize the concept from linear algebra, to general sets. Formally, we have the definition below: A matroid $\mathcal{A}$ is a tuple $\mathcal{A} = (\mathcal{S}, \mathcal{I})$, where $\mathcal{S}$ is a finite ground set and $\mathcal{I} \subseteq 2^{\mathcal{S}}$ (the power set of $\mathcal{S}$) is a collection of independent sets, such that:

1) $\mathcal{I}$ is nonempty, in particular, $\emptyset \in \mathcal{I}$,
2) $\mathcal{I}$ is downward closed: i.e., if $Y \in \mathcal{I}$ and $X \subseteq Y$, then $X \in \mathcal{I}$.
3) If $X, Y \in \mathcal{I}$, and $|X| < |Y|$, then $\exists y \in Y\backslash X$ such that $X \cup \{y\} \in \mathcal{I}$.

Partition matroid: The matroid $\mathcal{A} = (\mathcal{S}, \mathcal{I})$ is also a partition matroid when the ground set $\mathcal{S}$ is partitioned into (disjoint) sets $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_l$ and

$$\mathcal{I} = \{\boldsymbol{X} \subseteq \mathcal{S} : |\boldsymbol{X} \cap \mathcal{S}_i| \leq k_i \text{ for all } i = 1, \ldots, l\}, \tag{90}$$

for some given parameters $k_1, k_2, \ldots, k_l$.

Submodular functions: Let $\mathcal{S}$ be a finite ground set. A set function $f : 2^{\mathcal{I}} \to \mathbb{R}$ is submodular if for all sets $A, B \subseteq \mathcal{S}$,

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B). \tag{91}$$

Equivalently, submodularity can be defined by the following condition. Let $f_A(i) = f(A + i) - f(A)$ denote the marginal value of an element $i \in \mathcal{S}$ with respect to a subset $A \subseteq \mathcal{S}$. Then, f is a submodular if for all $A \subseteq B \subseteq \mathcal{S}$ and for all $i \in \mathcal{S}\backslash B$ we have:

$$f_A(i) \geq f_B(i). \tag{92}$$

REFERENCES

[1] *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2015-2020*, 2016.

[2] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "What will 5g be?" *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1065–1082, 2014.

[3] Y. Shi, J. Zhang, and K. B. Letaief, "Group sparse beamforming for green cloud-ran," *IEEE Transactions on Wireless Communications*, vol. 13, no. 5, pp. 2809–2823, 2014.

[4] R. Wang, X. Peng, J. Zhang, and K. Letaief, "Mobility-aware caching for content-centric wireless networks: Modeling and methodology," *arXiv preprint arXiv:1605.03709*, 2016.

[5] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless content delivery through distributed caching helpers," *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

[6] J. Liu, B. Bai, J. Zhang, and K. B. Letaief, "Content caching at the wireless network edge: A distributed algorithm via belief propagation," in *2016 IEEE International Conference on Communications (ICC)*, June 2016, pp. 1–6.

[7] X. Peng, J. C. Shen, J. Zhang, and K. B. Letaief, "Backhaul-aware caching placement for wireless networks," in *2015 IEEE Global Communications Conference (GLOBECOM)*, December 2015, pp. 1–6.

[8] X. Peng, J. Zhang, S. H. Song, and K. B. Letaief, "Cache size allocation in backhaul limited wireless networks," in *2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6.

[9] N. Golrezaei, A. Molisch, A. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution," *Communications Magazine, IEEE*, vol. 51, no. 4, pp. 142–149, April 2013.

[10] M. Ji, G. Caire, and A. F. Molisch, "Wireless device-to-device caching networks: Basic principles and system performance," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 1, pp. 176–189, 2016.

[11] R. Wang, J. Zhang, S. Song, and K. B. Letaief, "Mobility-aware caching in d2d networks," *arXiv preprint arXiv:1606.05282*, 2016.

[12] V. Dhar, "Data science and prediction," *Communications of the ACM*, vol. 56, no. 12, pp. 64–73, 2013.

[13] A. Johansson, "Clustering user-behavior in a collaborative online social network: A case study on quantitative user-behavior classification," 2016.

[14] M.-A. M. Ali and U. Niesen, "Fundamental limits of caching," *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 2856–2867, 2014.

[15] M. Ji, G. Caire, and A. F. Molisch, "Optimal throughput-outage trade-off in wireless one-hop caching networks," in *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*. IEEE, 2013, pp. 1461–1465.

[16] M.Ji, G.Caire, and A.F.Molisch, "The throughput-outage tradeoff of wireless one-hop caching networks," *IEEE Transactions on Information Theory*, vol. 61, no. 12, pp. 6833–6859, 2015.

[17] G. C. Mingyue Ji and M. F. Andreas, "Fundamental limits of caching in wireless d2d networks," *IEEE Transactions on Information Theory*, vol. 62, no. 2, pp. 849–869, 2016.

[18] J. Song, H. Song, and W. Choi, "Optimal caching placement of caching system with helpers," in *2015 IEEE International Conference onCommunications (ICC)*, June 2015, pp. 1825–1830.

[19] I. Psaras, W. K. Chai, and G. Pavlou, "In-network cache management and resource allocation for information-centric networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 11, pp. 2920–2931, November 2014.

[20] D. Bethanabhotla, G. Caire, and M. J. Neely, "Adaptive video streaming for wireless networks with multiple users and helpers," *IEEE Transactions on Communications*, vol. 63, no. 1, pp. 268–285, January 2015.

[21] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Characteristics of youtube network traffic at a campus network–measurements, models, and implications," *Computer Networks*, vol. 53, no. 4, pp. 501–514, 2009.

[22] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge university press, 2005.

[23] E. Biglieri, J. Proakis, and S. Shamai, "Fading channels: Information-theoretic and communications aspects," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2619–2692, 1998.

[24] E. H. Ong, J. Kneckt, O. Alanen, Z. Chang, T. Huovinen, and T. Nihtilä, "Ieee 802.11 ac: Enhancements for very high throughput wlans," in *2011 IEEE 22nd International Symposium on Personal, Indoor and Mobile Radio Communications*, 2011, pp. 849–853.

[25] L. Huang, S. Zhang, M. Chen, and X. Liu, "When backpressure meets predictive scheduling," in *Proceedings of the 15th ACM international symposium on Mobile ad hoc networking and computing*, 2014, pp. 33–42.

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

[27] J. Edmonds, "Matroids and the greedy algorithm," *Mathematical programming*, vol. 1, no. 1, pp. 127–136, 1971.

[28] T. A. Jenkyns, "The efficacy of the greedy algorithm," in *Proceedings of the 7th Southeastern Conference on Combinatorics, Graph Theory, and Computing, Utilitas Mathematica, Winnipeg*, 1976, pp. 341–350.

[29] B. Korte and D. Hausmann, "An analysis of the greedy heuristic for independence systems," *Algorithmic aspects of combinatorics*, vol. 2, pp. 65–74, 1978.

[30] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, "Maximizing a submodular set function subject to a matroid constraint," in *International Conference on Integer Programming and Combinatorial Optimization*. Springer, 2007, pp. 182–196.

[31] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, pp. 302–315, April 2006.

[32] L. Georgiadis, M. J. Neely, and L. Tassiulas, *Resource allocation and cross-layer control in wireless networks*. Now Publishers Inc, 2006.

[33] L. Huang and M. J. Neely, "Delay reduction via lagrange multipliers in stochastic network optimization," *IEEE Transactions on Automatic Control*, vol. 56, no. 4, pp. 842–857, April 2011.

[34] N. Golrezaei, A. G. Dimakis, and A. F. Molisch, "Scaling behavior for device-to-device communications with distributed caching," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 4286–4298, 2014.

[35] N. Khan and C. Oestges, "Impact of transmit antenna beamwidth for fixed relay links using ray-tracing and winner ii channel models," in *Antennas and Propagation (EUCAP), Proceedings of the 5th European Conference on*, 2011, pp. 2938–2941.