Optimal Peer-to-Peer Scheduling for Mobile Wireless Networks with Redundantly Distributed Data

Michael J. Neely
University of Southern California

***Abstract*— This paper considers peer-to-peer scheduling for a network with multiple wireless devices. A subset of the devices are mobile users that desire specific files. Each user may already have certain popular files in its cache. The remaining devices are access points that typically have a larger set of files. Users can download packets of their requested file from an access point or from another user. A dynamic algorithm that opportunistically grabs packets from current neighbors is developed. Under a simple model where each user desires a single file with infinite length, the algorithm is shown to optimize utility while incentivizing participation. The algorithm extends as an efficient heuristic in more general cases with finite file sizes and random active and idle periods. Example simulations demonstrate the dramatic throughput gains enabled by wireless peering.**

***Index Terms*— Stochastic optimization, Queueing analysis**

I. Introduction

Consider a network with N wireless devices. Let K of these devices be identified as *users*, where $K \leq N$. The users can send packets to each other via direct peer-to-peer transmissions. Each user has a certain collection of popular files in its cache. The remaining $N - K$ devices are *access points*. The access points are connected to a larger network, such as the internet, and hence typically have access to a larger set of files. While a general network may have users that desire to upload packets to the access points, this paper focuses only on the user downloads. Thus, throughout this paper it is assumed that the access points only send packets to the users, but do not receive packets. In contrast, the users can both send and receive. The network is mobile, and so the transmission options between access points and users, and between user pairs, can change over time.

Each user only wishes to download, and does not naturally want to send any data. Users will only send data to each other if they agree to operate according to a control algorithm that schedules such transmissions. This paper assumes the users have already agreed to abide by the control algorithm, and thus focuses attention on altruistic network design that optimizes a global network utility function. Such a prior agreement is only reasonable if the algorithm provides desirable benefits to the participants. Thus, this paper includes a tit-for-tat constraint, similar to the work in [2], which is an effective mechanism for incentivizing participation. Further, while this paper develops a single algorithm for optimizing the entire network, this does not automatically require the algorithm to be centralized. Indeed, the resulting algorithm often has a distributed implementation.

The model of this paper applies to a variety of practical network situations. For example, the access points can be wireless base stations in a future cellular network that allows both base-station-to-user transmissions as well as direct user-to-user transmissions. Alternatively, some of the access points can be from smaller femtocells. To increase the capacity of wireless systems, it is essential for future networks to enable such femtocell access and/or direct user-to-user transmission. This paper considers only 1-hop communication, so that all downloads are received either directly from an access point or from another user. The possibility of a user acting as a multi-hop relay is not considered here.

The prior work [2] treats a more complex model where each user can actively download multiple files at the same time, and where the arrival process of desired files at each user is random. A key challenge in this case is the complexity explosion associated with labeling each file according to the subset of other devices that already have it. This is solved in [2] by first observing the subset information of each newly arriving file, making an immediate decision about which device in this subset should transmit the desired packets, and then placing this request in a *request queue* at that selected device. The devices are not required to transmit these packets immediately. Rather, they can satisfy the requests over time. This procedure does not sacrifice optimality, and yields an algorithm with polynomial complexity. However, while this is effective for networks with static topology, it can result in significant delays in mobile networks. That is because a device that is pre-selected for transmission may not currently be in close proximity to the intended user, and/or may move out of range before transmission occurs.

The current paper provides an alternative algorithm that is particularly suited to the mobile case. To do so, a simpler model is considered where each user desires only one file that has infinite size. This enables one to focus on scheduling to achieve optimally fair download rates. Rather than pre-selecting devices for eventual transmission, the algorithm makes opportunistic packet transmission decisions from the set of current neighbors that have the desired file. This also facilitates distributed implementation. The infinite file size assumption is an approximation that is reasonable when file sizes are large, such as for video files. A heuristic extension to the case of finite file sizes is treated in Section V. This heuristic is based on the optimality insights obtained from the infinite size case. It is analytically shown to provide the same queueing

This material was presented in part at the 46th Conference on Information Sciences and Systems (CISS), Princeton, March 2012 [1].

This material is supported in part by one or more of: the NSF Career grant CCF-0747525, the Network Science Collaborative Technology Alliance sponsored by the U.S. Army Research Laboratory W911NF-09-2-0053.

bounds as the infinite size case. It is shown in simulation to offer desirable throughput and to be adaptive to non-ergodic events. The simulations also illustrate the significant gains achievable by enabling wireless peering as opposed to receiving downloads only from a base station.

Prior work on fair scheduling in mobile ad-hoc networks has considered token-based and economics-based mechanisms to incentivize participation [3][4][5]. Incentives are also well studied in the peer-to-peer literature. For example, algorithms in [6][7][8][9] track the number of uploads and downloads for each user, and give preferential treatment to those who have helped others. Algorithms based on tokens, markets, and peer reputations are considered in [10][11][12]. Such algorithms are conceptually based on the simple "tit-for-tat" or "treat-for-treat" principle, where rewards are given in direct proportion to the amount of self-sacrificial behavior at each user. The current paper also considers a tit-for-tat mechanism. However, a key difference is that it designs a *tit-for-tat constraint* directly into the optimization problem (similar to prior work [2] that used a different network model). Remarkably, the solution of the optimization naturally results in an intuitive token-like procedure, where the number of tokens in a virtual queue at each user determines the *reputation* of the user. Peer-to-peer transmissions between user pairs are given preferential treatment according to the *differential reputation* between the users. This is similar to the *backpressure* principle for optimal network scheduling [13][14]. However, the "backpressure" in the present paper is determined by token differentials in virtual reputation queues, rather than congestion differentials.

This paper uses Lyapunov optimization, which enables optimization of time averages of general network penalties and utilities subject to queue stability [15]. Stabilizing queueing networks by minimizing Lyapunov drift was pioneered by Tassiulas and Ephremides in [13][16]. This was extended to minimizing a *drift-plus-penalty* expression in [17][18][19] for joint queue stability and time average utility optimization. This stochastic technique is related to dual subgradient techniques for static convex programs (see works in [20][21][22][23][24] for static network utility maximization). Related primal-dual stochastic algorithms are in [25][26][27][28], see also tutorials in [14][29][30][15]. Practical implementations of Lyapunov based scheduling are reported in [31][32].

II. Basic Model

Let $\mathcal{N}$ denote the set of devices, and $\mathcal{K}$ denote the set of users, where $\mathcal{K} \subseteq \mathcal{N}$. Let N and K be the sizes of these sets. The network operates in discrete time with unit time slots $t \in \{0, 1, 2, \ldots\}$. Assume that on each slot t, each user $k \in \mathcal{K}$ desires at most one file. For each $k \in \mathcal{K}$ and each slot t, define $\mathcal{F}_k(t)$ as the set of all devices in $\mathcal{N}$ that have the file currently desired by user k. The set $\mathcal{F}_k(t)$ can include both users and access points, and represents the set of devices that user k can potentially receive packets from on slot t. If user k currently does not want *any* file, or if the desired file is not contained in *any* of the devices, then $\mathcal{F}_k(t)$ is defined as the empty set $\{\}$. If user k only accepts downloads from a certain subset of devices that it identifies as its *social group*, then $\mathcal{F}_k(t)$ can be viewed as the intersection of its social group and the set of devices that have its currently desired file. It is assumed throughout that $k \notin \mathcal{F}_k(t)$ (for all $k \in \mathcal{K}$ and all t), so that no user wants a file that it already has.

In an actual network, the sets $\mathcal{F}_k(t)$ can change over time. This happens if user k desires a new file, or if there is a change in the cache of another device. The queueing bounds derived in this paper treat arbitrarily varying $\mathcal{F}_k(t)$ sets. However, optimal throughput utility is proven only in the special case when file subsets never change, so that $\mathcal{F}_k(t) = \mathcal{F}_k$ for all t, for some fixed subsets $\mathcal{F}_k \subseteq \mathcal{N}$. This corresponds to the case when each user desires a single file that consists of an infinite number of packets, and the cache contents of the devices never change. The algorithm extends as a high quality heuristic in more general cases of finite file sizes and time-varying file subsets, as explored via simulations in Section VI.

Every slot t, the network makes *transmission actions*. Let $(\mu_{nk}(t))$ be the matrix of transmission actions chosen on slot t, where each entry $\mu_{nk}(t)$ is the number of packets that device $n \in \mathcal{N}$ transmits to user $k \in \mathcal{K}$. The set of all possible matrix options to choose from is determined by the current *topology state* of the network, as in [14]. Specifically, let $\omega(t)$ represent the topology state on slot t, being a vector of parameters that affect transmission, such as current device locations and/or channel conditions. Assume $\omega(t)$ takes values in an abstract set Ω, possibly being an infinite set. The $\omega(t)$ process is assumed to be ergodic. In the case when Ω is finite or countably infinite, the steady state probabilities are represented by $\pi(\omega) = Pr[\omega(t) = \omega]$ for all $\omega \in \Omega$. Else, the steady state probabilities are represented by an appropriate probability density function. These probabilities are not necessarily known to the network controller. Extensions to the case when $\omega(t)$ is non-ergodic are explored in Sections V and VI.

For each $\omega \in \Omega$, define $\mathcal{R}(\omega)$ as the set of all transmission matrices (μ_{nk}) that are possible when $\omega(t) = \omega$. The exact structure of $\mathcal{R}(\omega)$ depends on the particular physical characteristics and interference properties of the network. One may choose the $\mathcal{R}(\omega)$ sets to constrain the transmission variables μ_{nk} to take integer values, although this is not required in the analysis of this paper. It is assumed only that each set $\mathcal{R}(\omega)$ has the following basic properties:

- Every matrix $(\mu_{nk}) \in \mathcal{R}(\omega)$ has non-negative entries.
- If $(\mu_{nk}) \in \mathcal{R}(\omega)$, then $(\tilde{\mu}_{nk}) \in \mathcal{R}(\omega)$, where $(\tilde{\mu}_{nk})$ is any matrix formed by setting one or more entries of (μ_{nk}) to zero.
- Every matrix $(\mu_{nk}) \in \mathcal{R}(\omega)$ must satisfy the constraint $0 \leq \sum_{n \in \mathcal{N}} \mu_{nk} \leq x_k^{max}$ for all $k \in \mathcal{K}$, where x_k^{max} is a given bound on the number of packets that can be delivered to user k on one slot, regardless of ω.

The next two subsections present example network models that fit into the above framework.

A. Example Network with Orthogonal Subcells

Suppose the network region is divided into C non-overlapping subcells. Let $c_n(t)$ be the current subcell of device n, so that $c_n(t) \in \{1, \ldots, C\}$. Let $\boldsymbol{c}(t) = (c_1(t), \ldots, c_N(t))$ be the vector of device locations on slot t. The access points

have fixed locations, while the mobile users can change subcells from slot to slot. Let $\boldsymbol{S}(t) = (S_{nk}(t))$ be a channel state matrix, where $S_{nk}(t)$ is the number of packets that device n can transmit to device k on slot t, provided that there are no competing transmissions (as defined below). The value $S_{nk}(t)$ can depend on the $\boldsymbol{c}(t)$ location vector. Let the topology state $\omega(t)$ be given by $\omega(t) = (\boldsymbol{c}(t), \boldsymbol{S}(t))$. Assume access point transmissions are orthogonal from all other access point transmissions and from all peer-to-peer user transmissions. For each $\omega(t)$, define $\mathcal{R}(\omega(t))$ as the set of all $(\mu_{nk}(t))$ matrices with entries that satisfy:

- $\mu_{nk}(t) \in \{0, S_{nk}(t)\}$ for all $n \in \mathcal{N}$ and $k \in \mathcal{K}$.
- Users can only transmit to other users currently in their same subcell, so that $\mu_{nk}(t) = 0$ whenever $n \in \mathcal{K}$, $k \in \mathcal{K}$, and $c_n(t) \neq c_k(t)$.
- At most one user-to-user transmission can take place per subcell on a given slot.
- Each access point can send to at most one user per slot.

This particular structure is useful because it allows user transmissions to be scheduled separately in each subcell, and access point transmissions to be scheduled separately from all other decisions. The sets $\mathcal{R}(\omega(t))$ can be defined differently for more sophisticated interference models, such as that described in the next subsection.

B. Example Network With SINR-based Transmission

Assume all devices move about a 2-dimensional network region. Let $\boldsymbol{c}(t)$ be a matrix that specifies the 2-dimensional location information for all devices $n \in \mathcal{N}$. Let $\boldsymbol{S}(t) = (S_{nk}(t))$ be a matrix of current attenuations on the links, typically dependent on $\boldsymbol{c}(t)$. The topology state process is then $\omega(t) = (\boldsymbol{c}(t), \boldsymbol{S}(t))$. Transmission proceeds in two stages, as in [33][34]: Every slot t, each device $n \in \mathcal{N}$ first chooses whether or not to transmit (it does not yet choose which user it transmits to). Let $1_n(t)$ be a binary variable that is 1 if device n chooses to transmit on slot t, and 0 else. If device n chooses to transmit, it always uses a fixed power P_n. The resulting combined signal plus noise at a given user $k \in \mathcal{K}$ is:

$$Comb_k(t) = \sigma_k^2 + \textstyle\sum_{n\in\mathcal{N}} P_n 1_n(t) S_{nk}(t)$$

where σ_k^2 is the noise power at user k. Once the $1_n(t)$ decisions are made, a pilot signal is sent so that each user k can measure $Comb_k(t)$ and report the value back to its potential transmitters. Based on this feedback, each transmitting device chooses a single user as the recipient of its transmission. Let $1_{nk}(t)$ be a binary variable that is 1 if device n decides to transmit to user k, and 0 else, with constraints:

$$\textstyle\sum_{k\in\mathcal{K}} 1_{nk}(t) \leq 1_n(t) \;\; \forall n \in \mathcal{N}$$

One may wish to impose additional constraints $(1_{nk}(t)) \in \mathcal{B}$ for some set $\mathcal{B}$. This can be used to restrict devices from transmitting to certain users that are too far away or that have relatively weak signal strength.

The transmission rates $\mu_{nk}(t)$ are then determined as a function of the resulting signal-to-noise-plus-interference ratio (SINR):

$$\mu_{nk}(t) = (1 - 1_k(t)) g_{nk}\left(\frac{1_{nk}(t) P_n S_{nk}(t)}{Comb_k(t) - 1_{nk}(t) P_n S_{nk}(t)}\right)$$

where the $(1 - 1_k(t))$ factor ensures that transmitting devices cannot receive. Each function $g_{nk}(s)$ is assumed to satisfy:

- $g_{nk}(s)$ is non-decreasing in s.
- $g_{nk}(0) = 0$.
- $0 \leq g_{nk}(s) \leq g_{max}$ for all values s that can occur as arguments, where g_{max} is some finite bound on the transmission rate.

An example function is $g_{nk}(s) = W_{nk} \ln(1 + s)$ for some bandwidth values W_{nk}, which corresponds to Shannon capacity. Alternatively, $g_{nk}(s)$ can be a discontinuous step function that corresponds to a fixed set of modulation options [33]. An example is when all transmissions use a single modulation strategy that requires SINR to be above some threshold θ, in which case $g_{nk}(s) = 0$ for all $s < \theta$, and $g_{nk}(s) = \mu$ for all $s \geq \theta$, where μ is some positive number.

This 2-stage transmission model ensures that, after the initial decisions $1_n(t)$ are made, the transmission rates $\mu_{nk}(t)$ can be selected in a distributed way at each device n using only knowledge of the $Comb_k(t)$ and $S_{nk}(t)$ values for each potential receiver $k \in \mathcal{K}$. The structure is completely distributed if one assumes the first stage decisions are made randomly, where each device n independently chooses to transmit with some pre-specified probability θ_n, and these random decisions for $1_n(t)$ are modeled as an additional part of the topology state process $\omega(t)$ [33][34].

Alternatively, one can envision a scenario where femto "helper" nodes transmit with constant power P_n on each and every slot, and user nodes never transmit. In the absence of mobility, this would produce a constant value for $Comb_k(t)$ for each user k on every slot t. This value could be measured at time $t = 0$ and no further feedback would be necessary.[1]

In all of these cases, the resulting set of transmission rate options for $(\mu_{nk}(t))$ depends only on the topology state $\omega(t)$. This fits into our general framework of an abstract set of options $\mathcal{R}(\omega(t))$. More sophisticated sets $\mathcal{R}(\omega(t))$ that require centralized scheduling can be used in cases when a single station has the capability to coordinate all link decisions.

C. Optimization Objective

For each $a \in \mathcal{N}$ and $b \in \mathcal{K}$, define $f_{ab}(t)$ as an indicator function that is 1 if and only if device a has the file currently requested by user b:

$$f_{ab}(t) \triangleq \begin{cases} 1 & \text{if } a \in \mathcal{F}_b(t) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where notation "$\triangleq$" denotes "defined to be equal to." Every slot t, the network controller observes $\omega(t)$ and chooses $(\mu_{nk}(t)) \in \mathcal{R}(\omega(t))$. For each $k \in \mathcal{K}$, define $x_k(t)$ as the total number of packets that user k receives from others on slot t, and define $y_k(t)$ as the total number of packets that user k delivers to others on slot t:

$$x_k(t) \triangleq \textstyle\sum_{a\in\mathcal{N}} \mu_{ak}(t) f_{ak}(t) \tag{2}$$

$$y_k(t) \triangleq \textstyle\sum_{b\in\mathcal{K}} \mu_{kb}(t) f_{kb}(t) \tag{3}$$

[1]Of course, a practical system would periodically update the values of $Comb_k(t)$ to track any network changes. However, in cases of low variability these updates need not take place on every slot.

The multiplication $\mu_{ak}(t)f_{ak}(t)$ in (2) and (3) formally ensures that user k can only receive a packet from another device that has the file it is requesting.

For a given control algorithm, let $\overline{x}_k$ and $\overline{y}_k$ represent the time averages of the $x_k(t)$ and $y_k(t)$ processes for all $k \in \mathcal{K}$:

$$\overline{x}_k \triangleq \lim_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} x_k(\tau) \ , \ \overline{y}_k \triangleq \lim_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} y_k(\tau)$$

These limits are temporarily assumed to exist.[2] The value $\overline{x}_k$ is the time average download rate of user k, and $\overline{y}_k$ is the time average upload rate. The goal is to develop a control algorithm that solves the following:[3]

$$\text{Maximize:} \quad \sum_{k\in\mathcal{K}} \phi_k(\overline{x}_k) \tag{4}$$

$$\text{Subject to:} \quad \alpha_k\overline{x}_k \leq \beta_k + \overline{y}_k \ \forall k \in \mathcal{K} \tag{5}$$

$$(\mu_{nk}(t)) \in \mathcal{R}(\omega(t)) \ \forall t \in \{0,1,2,\ldots\} \tag{6}$$

where for each $k \in \mathcal{K}$, $\phi_k(x)$ are given concave functions of x, and α_k, β_k are given non-negative weights. The value $\phi_k(\overline{x}_k)$ represents the *utility* associated with user k downloading at rate $\overline{x}_k$. The constraints (5) are the *tit-for-tat* constraints from [2]. These constraints incentivize participation. They allow a "free" download rate of β_k/α_k. Users can only receive rates beyond this value in proportion to the rate at which they help others. Choosing larger values of α_k (typically in the range $0 \leq \alpha_k \leq 1$) leads to more stringent requirements about helping others. These tit-for-tat constraints restrict the system operation and thus can affect overall network utility. Removing these constraints by setting $\alpha_k = 0$ for all k leads to the largest network utility, but does not embed any participation incentives into the optimization problem.

The functions $\phi_k(x)$ are assumed to be concave, continuous, and non-decreasing over the interval $x \geq 0$. They are not required to be differentiable. For example, they can be piecewise linear, such as $\phi_k(x) = \min[x, \theta_k]$, where θ_k is a given constant rate desired by user k. Alternatively, one can choose $\phi_k(x) = \ln(1+\nu x)$ for each $k \in \mathcal{K}$ and for some constant $\nu > 0$. As $\nu \to \infty$, this leads to the well known *proportional fairness utility* [21].[4]

One may want to modify the problem (4)-(6) by specifying separate utility functions for the user-to-user download rates and the access-point-to-user download rates. This is possible by creating two "virtual users" $m_1(k)$ and $m_2(k)$ for each actual user $k \in \mathcal{K}$. Channel conditions for the virtual users $m_1(k), m_2(k)$ are defined to be the same as for the actual user k, with the exception that virtual user $m_1(k)$ is restricted to receive only from other users, while virtual user $m_2(k)$ is restricted to receive only from the access points.

The above formulation does not explicitly consider energy use as part of the optimization. This simplifies exposition of the paper by focusing on the basic issues of utility optimal peering with the tit-for-tat constraint. Section VIII extends the formulation to incorporate energy constraints.

[2]This is only to simplify exposition of the optimization goal. The performance analysis in Section IV does not a-priori assume the limits exist.

[3]Note that the trivial all-zero solution is always feasible.

[4]The function $\ln(x)$ has a singularity at $x = 0$, although Lyapunov optimization theory can still be applied in this special case [15].

III. The Dynamic Algorithm

The problem (4)-(6) is solved via the stochastic network optimization theory of [15][14]. First note that problem (4)-(6) is equivalent to the following problem that uses *auxiliary variables* $\gamma_k(t)$:

$$\text{Maximize:} \quad \sum_{k\in\mathcal{K}} \overline{\phi_k(\gamma_k)} \tag{7}$$

$$\text{Subject to:} \quad \alpha_k\overline{x}_k \leq \beta_k + \overline{y}_k \ \forall k \in \mathcal{K} \tag{8}$$

$$\overline{\gamma}_k \leq \overline{x}_k \ \forall k \in \mathcal{K} \tag{9}$$

$$(\mu_{nk}(t)) \in \mathcal{R}(\omega(t)) \ \forall t \in \{0,1,2,\ldots\} \tag{10}$$

$$0 \leq \gamma_k(t) \leq x_k^{max} \ \forall t \in \{0,1,2,\ldots\} \tag{11}$$

where $\overline{\phi_k(\gamma_k)}$ is defined as the time average of the process $\phi_k(\gamma_k(t))$:

$$\overline{\phi_k(\gamma_k)} \triangleq \lim_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \phi_k(\gamma_k(\tau))$$

The auxiliary variables $\gamma_k(t)$ act as proxies for the actual download variables $x_k(t)$. This technique transforms a problem of maximizing a function of a time average into a problem of maximizing the time average of a function (compare problems (4)-(6) and (7)-(11)).

The optimal utility is the same for both problems (4)-(6) and (7)-(11). To see this, let ϕ_1^* and ϕ_2^* represent the optimal utility for problems (4)-(6) and (7)-(11), respectively. First note that $\phi_1^* \leq \phi_2^*$, since an optimal solution of (4)-(6) can be used as a feasible solution to (7)-(11), yielding the same utility, by setting $\gamma_k(t) = \overline{x}_k$ for all t. To prove $\phi_2^* \leq \phi_1^*$, consider an algorithm that solves the problem (7)-(11). Let $(\mu_{nk}(t))$, $(\gamma_k(t))$ be the decisions made over time, and let $\overline{x}_k$, $\overline{y}_k$, $\overline{\gamma}_k$, $\overline{\phi_k(\gamma_k)}$ be the corresponding time averages, all of which satisfy the constraints of the problem (7)-(11). Note that these constraints include all of the desired constraints of the original problem (4)-(6). Then:

$$\begin{aligned}
\phi_2^* &= \sum_{k\in\mathcal{K}} \overline{\phi_k(\gamma_k)} && (12)\\
&\leq \sum_{k\in\mathcal{K}} \phi_k(\overline{\gamma}_k) && (13)\\
&\leq \sum_{k\in\mathcal{K}} \phi_k(\overline{x}_k) && (14)\\
&\leq \phi_1^* && (15)
\end{aligned}$$

where (12) holds because this algorithm achieves the optimal utility ϕ_2^* for problem (7)-(11), (13) holds by Jensen's inequality, (14) holds because this algorithm must yield time averages that satisfy $\overline{\gamma}_k \leq \overline{x}_k$ for all $k \in \mathcal{K}$, and (15) holds because the transmission decisions of the algorithm satisfy all desired constraints of the original problem, and thus produce $\overline{x}_k$ values that give a utility that is less than or equal to the optimal utility of the original problem (which is ϕ_1^*). It follows that any algorithm that is optimal for (7)-(11) makes decisions that are also optimal for the original problem.

A. Virtual Queues

To facilitate satisfaction of the tit-for-tat constraints (8), for each $k \in \mathcal{K}$ define a *virtual queue* $H_k(t)$, with dynamics:

$$H_k(t+1) = \max\left[H_k(t) + \alpha_k x_k(t) - \beta_k - y_k(t), 0\right] \tag{16}$$

where $x_k(t)$, $y_k(t)$ are defined in (2)-(3). The intuition is that $\alpha_k x_k(t)$ can be viewed as the "arrivals" on slot t, and $\beta_k + y_k(t)$ can be viewed as the "offered service" on slot t. Stabilizing queue $H_k(t)$ ensures the time average of the "arrivals" is less than or equal to the time average of the "service," which ensures constraints (8).

Similarly, to satisfy the constraints (9), for each $k \in \mathcal{K}$ define another virtual queue $Q_k(t)$ with dynamics:

$$Q_k(t+1) = \max[Q_k(t) + \gamma_k(t) - x_k(t), 0] \quad (17)$$

The update (17) can be interpreted as a queueing equation where $\gamma_k(t)$ is the amount of data requested by user k on slot t, and $x_k(t)$ is the amount of service. Stabilizing $Q_k(t)$ ensures $\overline{\gamma}_k \leq \overline{x}_k$.

B. The Drift-Plus-Penalty Expression

Define the following quadratic function $L(t)$:

$$L(t) \triangleq \frac{1}{2}\sum_{k\in\mathcal{K}}[Q_k(t)^2 + H_k(t)^2]$$

Intuitively, taking actions to push $L(t)$ down tends to maintain stability of all queues. Define $\Delta(t)$ as the drift on slot t:

$$\Delta(t) \triangleq L(t+1) - L(t)$$

The algorithm is designed to observe the queues and the current $\omega(t)$ on each slot t, and to then choose $(\mu_{nk}(t)) \in \mathcal{R}(\omega(t))$ and $\gamma_k(t)$ subject to $0 \leq \gamma_k(t) \leq x_k^{max}$ to minimize a bound on the following *drift-plus-penalty expression* [15]:

$$\Delta(t) - V\sum_{k\in\mathcal{K}} \phi_k(\gamma_k(t))$$

where V is a non-negative weight that affects a performance bound. Intuitively, the value of V affects the extent to which the control action on slot t emphasizes utility optimization in comparison to drift minimization.

Lemma 1: Under any control algorithm, the drift-plus-penalty satisfies:

$$\begin{aligned}\Delta(t) - V\sum_{k\in\mathcal{K}} \phi_k(\gamma_k(t)) \leq B(t) - V\sum_{k\in\mathcal{K}}\phi_k(\gamma_k(t)) \\ +\sum_{k\in\mathcal{K}} H_k(t)[\alpha_k x_k(t) - \beta_k - y_k(t)] \\ +\sum_{k\in\mathcal{K}} Q_k(t)[\gamma_k(t) - x_k(t)] \quad (18)\end{aligned}$$

where $B(t)$ is defined:

$$\begin{aligned}B(t) \quad \triangleq \quad & \frac{1}{2}\sum_{k\in\mathcal{K}}(\alpha_k x_k(t) - \beta_k - y_k(t))^2 \\ & +\frac{1}{2}\sum_{k\in\mathcal{K}}(\gamma_k(t) - x_k(t))^2\end{aligned}$$

Proof: Squaring (16) and using $\max[y,0]^2 \leq y^2$ for any real number y yields:

$$\begin{aligned}H_k(t+1)^2 \quad \leq \quad & H_k(t)^2 + (\alpha_k x_k(t) - \beta_k - y_k(t))^2 \\ & 2H_k(t)(\alpha_k x_k(t) - \beta_k - y_k(t))\end{aligned}$$

Similarly, squaring (17) gives:

$$\begin{aligned}Q_k(t+1)^2 \quad \leq \quad & Q_k(t)^2 + (\gamma_k(t) - x_k(t))^2 \\ & +2Q_k(t)(\gamma_k(t) - x_k(t))\end{aligned}$$

Summing over $k \in \mathcal{K}$ and dividing by 2 yields the result. □

The value of $B(t)$ can be upper bounded by a finite constant B every slot, where B depends on the maximum possible values that $\mu_{nk}(t)$ and $\gamma_k(t)$ can take. The algorithm below is defined by observing $\omega(t)$ and the queue states every slot t, and choosing actions to minimize the last three terms on the right-hand-side of (18) (not including the first term $B(t)$), given these observed quantities. The algorithm is derived by identifying the factors that involve decision variables $\gamma_k(t)$ and $\mu_{nk}(t)$ in these last three terms. Isolating the $\gamma_k(t)$ variables in the last three terms on the right-hand-side of (18) gives the expression:

$$\sum_{k\in\mathcal{K}}[-V\phi_k(\gamma_k(t)) + Q_k(t)\gamma_k(t)] \quad (19)$$

Similarly, isolating the $\mu_{nk}(t)$ variables in the last three terms on the right-hand-side of (18) is done by substituting definitions of $x_k(t)$ and $y_k(t)$ from (2)-(3). This leads to the expression:

$$\sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{K}} \mu_{nk}(t)f_{nk}(t)[\alpha_k H_k(t) - 1_{\{n\in\mathcal{K}\}}H_n(t) - Q_k(t)] \quad (20)$$

where $1_{\{n\in\mathcal{K}\}}$ is an indicator function that is 1 if device n is a user, and 0 if device n is an access point.

C. The Dynamic Peering Algorithm

The dynamic algorithm observes the virtual queues and the $\omega(t)$ value on every slot t, and chooses control decision variables that greedily minimize the expressions (19) and (20). Specifically, every slot t the algorithm performs the following:

- ($\gamma_k(t)$ decisions) Each user $k \in \mathcal{K}$ observes $Q_k(t)$ and chooses $\gamma_k(t)$ to solve:

$$\begin{aligned}\text{Maximize:} \quad & V\phi_k(\gamma_k(t)) - Q_k(t)\gamma_k(t) \quad (21)\\ \text{Subject to:} \quad & 0 \leq \gamma_k(t) \leq x_k^{max} \quad (22)\end{aligned}$$

- ($\mu_{nk}(t)$ decisions) The network controller observes all queues $(\boldsymbol{Q}(t), \boldsymbol{H}(t))$ and the topology state $\omega(t)$ on slot t, and chooses matrix $(\mu_{nk}(t)) \in \mathcal{R}(\omega(t))$ to maximize the following expression:

$$\sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{K}} \mu_{nk}(t)f_{nk}(t)W_{nk}(t) \quad (23)$$

  where weights $W_{nk}(t)$ are defined:

$$W_{nk}(t) \triangleq \left[Q_k(t) + 1_{\{n\in\mathcal{K}\}}H_n(t) - \alpha_k H_k(t)\right]$$

- (Queue updates) Update virtual queues $H_k(t)$ and $Q_k(t)$ for all $k \in \mathcal{K}$ via (16) and (17).

The $\gamma_k(t)$ decisions can be viewed as flow control actions that restrict the amount of data requested from user k on each slot. They are made separately at each user k. The $(\mu_{nk}(t))$ decisions are transmission actions made at the network layer. Examples are given below.

D. Example Flow Control Decisions

Consider the following piecewise linear utility functions for all $k \in \mathcal{K}$:

$$\phi_k(x_k) = \nu_k \min[x_k, \theta_k] \quad (24)$$

where ν_k are given positive values that act as priority weights for the users, and θ_k are are given positive values that represent the maximum desired communication rate for each user.

Assume that $\theta_k \leq x_k^{max}$ for all $k \in \mathcal{K}$. Then the algorithm in the previous section chooses $\gamma_k(t)$ for each user $k \in \mathcal{K}$ as:

$$\gamma_k(t) = \begin{cases} \theta_k & \text{if } Q_k(t) \leq V\nu_k \\ 0 & \text{otherwise} \end{cases}$$

Alternatively, consider the following strictly concave utility functions for all $k \in \mathcal{K}$:

$$\phi_k(x_k) = \ln(1 + \nu_k x_k) \tag{25}$$

Then the $\gamma_k(t)$ decisions are:

$$\gamma_k(t) = \left[\frac{V}{Q_k(t)} - \frac{1}{\nu_k}\right]_0^{x_k^{max}} \tag{26}$$

where the operation $[y]_0^b$ is equal to y if $0 \leq y \leq b$, 0 if $y < 0$, and b if $y > b$. These utility functions can be viewed as an accurate approximation of the proportionally fair utility function if one selects $\nu_k = \nu$ for all k, for a large value of ν. Alternatively, using the proportionally fair utilities $\phi_k(x_k) = \ln(x_k)$ for all $k \in \mathcal{K}$ leads to $\gamma_k(t) = [V/Q_k(t)]_0^{x_k^{max}}$, which is indeed the same as (26) in the limit as $\nu_k \to \infty$.

E. Example Transmission Decisions (Cell-Partitioned Model)

Suppose the network has the special cell-partitioned structure specified in Section II-A. Let $\mathcal{A}$ be the set of access points. Let $\mathcal{K}_a(t)$ be the set of users within reach of access point a on slot t. Then each access point $a \in \mathcal{A}$ observes channels $S_{ak}(t)$ and queues $Q_k(t)$, $H_k(t)$ for all users $k \in \mathcal{K}_a(t)$ and chooses to serve the single user in $\mathcal{K}_a(t)$ with the largest non-negative value of $f_{ak}(t)S_{ak}(t)[Q_k(t) - \alpha_k H_k(t)]$ (breaking ties arbitrarily), and chooses no users if this value is negative for all $k \in \mathcal{K}_a(t)$.

Further, the user pairs in each subcell $c \in \{1, \ldots, C\}$ are observed. Amongst all users in a given subcell, the ordered user pair (a, k) with the largest non-negative value of $f_{ak}(t)S_{ak}(t)[Q_k(t) + H_a(t) - \alpha_k H_k(t)]$ is selected for peer-to-peer transmission in that subcell (breaking ties arbitrarily). No peer-to-peer transmission occurs in the subcell if this value is negative for all user pairs.

F. Example Transmission Decisions (SINR Model)

Suppose the network has the SINR structure as specified in Section II-B. Suppose the set of transmitting devices for slot t is determined in some way (perhaps randomly), so that the $1_n(t)$ decisions are made for all devices $n \in \mathcal{N}$.[5] The max-weight transmission decision rule (23) then reduces to the following: Each transmitting device n observes the $Comb_k(t)$, $S_{nk}(t)$, and $W_{nk}(t)$ values for its potential receivers k, and chooses to transmit to the single receiver k that maximizes the following quantity (breaking ties arbitrarily):

$$f_{nk}(t)W_{nk}(t)(1-1_k(t))g_{nk}\left(\frac{1_{nk}(t)P_n S_{nk}(t)}{Comb_k(t) - 1_{nk}(t)P_n S_{nk}(t)}\right)$$

[5]Of course, the optimal $1_n(t)$ decisions for each $n \in \mathcal{N}$ can be made according to the max-weight rule (23). However, this would involve a centralized decision of maximizing a weighted sum of rates over a non-convex set $\mathcal{R}(\omega(t))$. Randomizing the $1_n(t)$ decisions simplifies implementation and achieves optimality over the restricted set of algorithms that use such randomization, as discussed in Section II-B. See Chapter 6 of [15], and references therein, for a discussion of alternative approximation methods.

where $g_{nk}(s)$ is the function that maps an SINR level s to a transmission rate, as described in Section II-B.

IV. Algorithm Performance

A. Utility Optimality for Non-Varying File Subsets

In general scenarios, the $\mathcal{F}_k(t)$ sets can change over time, and these changes can be influenced by the past scheduling decisions. For example, suppose a user downloads multiple finite-sized files one after the other. Then its currently requested file and its cache of complete files depends on the rate at which it received previous downloads. An analysis of the optimal utility in this complex scenario appears to be a multi-dimensional Markov decision problem, and is beyond the scope of this paper.

However, in the special case when file subsets do not change over time, so that $\mathcal{F}_k(t) = \mathcal{F}_k$ for all t, the problem (4)-(6) exactly fits into the stochastic network optimization framework of [15]. That is because the problem (4)-(6) seeks to maximize a concave function of time averages of attribute functions $x_k(t)$ and $y_k(t)$, subject to time average constraints on these attributes, and:

- The only random events are isolated to the $\omega(t)$ process, which is ergodic and is not influenced by control actions.
- The attribute functions $x_k(t)$, $y_k(t)$ are pure functions of the observed random event $\omega(t)$ and the control actions $(\mu_{nk}(t))$, and do not depend on additional network state information.
- The attribute functions have bounded second moments $\mathbb{E}\left[x_k(t)^2\right]$, $\mathbb{E}\left[y_k(t)^2\right]$.

This leads to the following performance theorem. Assume each user $k \in \mathcal{K}$ desires a single "infinitely long" file, and define $\mathcal{F}_k \subseteq \mathcal{N}$ as the set of devices that have this file (assumed to satisfy $k \notin \mathcal{F}_k$). To model the $\omega(t)$ process, let $M(t)$ be a discrete time ergodic Markov chain with a finite state space $\mathcal{M}$. On each slot t, if $M(t) = m$, then $\omega(t)$ is chosen independently using a probability distribution $\eta_m(\omega)$ (so there is one distribution for each $m \in \mathcal{M}$).

Theorem 1: (Utility Performance [15]) Suppose $\mathcal{F}_k(t) = \mathcal{F}_k$ for all $t \in \{0, 1, 2, \ldots\}$. If $\omega(t)$ evolves according to the ergodic Markov chain as described above, then for any value of $V \geq 0$ the algorithm satisfies the desired tit-for-tat constraints, in the sense that for all $k \in \mathcal{K}$ we have:

$$\limsup_{t\to\infty}[\alpha_k \overline{x}_k(t) - \beta_k - \overline{y}_k(t)] \leq 0$$

where $\overline{x}_k(t)$ and $\overline{y}_k(t)$ are defined:

$$\begin{aligned} \overline{x}_k(t) &\triangleq \tfrac{1}{t}\textstyle\sum_{\tau=0}^{t-1}\mathbb{E}\left[x_k(\tau)\right] \\ \overline{y}_k(t) &\triangleq \tfrac{1}{t}\textstyle\sum_{\tau=0}^{t-1}\mathbb{E}\left[y_k(\tau)\right] \end{aligned}$$

Further, achieved utility differs from optimality by $O(1/V)$, in the sense that:

$$\liminf_{t\to\infty} \textstyle\sum_{k\in\mathcal{K}} \phi_k(\overline{x}_k(t)) \geq \phi^* - O(1/V)$$

where ϕ^* is the optimal utility for the problem (4)-(6).

Therefore, the achieved utility differs from the optimal ϕ^* by at most $O(1/V)$, which can be made arbitrarily small as the parameter V is increased. It turns out that the V parameter

determines a queue size tradeoff, which affects the timescales over which the tit-for-tat constraints are satisfied. This is demonstrated in the next subsections. Specifically, under mild additional assumptions, the special structure of the peer-to-peer network ensures that all queues are deterministically bounded by a constant that is proportional to V. Further, while the throughput utility results of Theorem 1 hold only when the $\omega(t)$ process is ergodic and the $\mathcal{F}_k(t)$ sets do not change over time, these assumptions are not needed in the queue bound analysis. Thus, the queue bounds derived below hold for arbitrary sample paths of $\omega(t)$ and $\mathcal{F}_k(t)$, regardless of whether or not these are ergodic and/or if these depend on past control actions.[6]

B. Bound on Data Queues $Q_k(t)$

Suppose each utility function $\phi_k(x)$ has right-derivatives that are bounded by a finite constant $\nu_k > 0$ over the interval $0 \leq x_k \leq x_k^{max}$. This holds for the example utility functions in (24) and (25), with the ν_k parameters specified there indeed being the maximum right derivatives.

Lemma 2: If utility function $\phi_k(x)$ has maximum right derivative $\nu_k > 0$, then:

$$0 \leq Q_k(t) \leq V\nu_k + x_k^{max} \ \ \forall t \in \{0, 1, 2, \ldots\}$$

provided that this inequality holds for $Q_k(0)$.

Proof: Assume that $Q_k(t) \leq V\nu_k + x_k^{max}$ for slot t (it holds by assumption on slot $t = 0$). We prove it also holds for slot $t + 1$. First consider the case $Q_k(t) \leq V\nu_k$. From the queue update equation (17), we see that this queue can increase by at most x_k^{max} on each slot, and so we have $Q_k(t + 1) \leq V\nu_k + x_k^{max}$, proving the result for this case.

Now consider the case $Q_k(t) > V\nu_k$. On slot t, the algorithm chooses $\gamma_k(t) \in [0, x_k^{max}]$ to maximize the expression:

$$V\phi_k(\gamma_k(t)) - Q_k(t)\gamma_k(t)$$

However, for any $\gamma_k(t) \geq 0$ we have:

$$\begin{aligned} &V\phi_k(\gamma_k(t)) - Q_k(t)\gamma_k(t) \\ &\quad \leq \ V\phi_k(0) + V\nu_k\gamma_k(t) - Q_k(t)\gamma_k(t) \\ &\quad = \ V\phi_k(0) + \gamma_k(t)[V\nu_k - Q_k(t)] \\ &\quad \leq \ V\phi_k(0) \end{aligned}$$

where equality holds if and only if $\gamma_k = 0$ (recall that $[V\nu_k - Q_k(t)] < 0$). It follows that the algorithm must choose $\gamma_k(t) = 0$, and so $Q_k(t)$ cannot increase on this slot. That is:

$$Q_k(t + 1) \leq Q_k(t) \leq V\nu_k + x_k^{max}$$

□

[6]Note also that if the queues are deterministically bounded and the conditions of Theorem 1 hold, then the time-average results in [35] ensure that all limiting time average expectations can be replaced by pure time averages (with probability 1).

C. Bound on Reputation Queues $H_k(t)$

The $H_k(t)$ processes act as *reputation queues* for each user $k \in \mathcal{K}$, being low if user k has a good reputation for helping others, and high otherwise (see queue dynamics in (16)). These reputations directly affect the transmission decisions via the weights $W_{nk}(t)$ in (23). To see this, define $\mathcal{A}$ as the set of access points. First consider the weight seen by an access point $a \in \mathcal{A}$ for user k on slot t:

$$W_{ak}(t) = Q_k(t) - \alpha_k H_k(t)$$

This weight is large if $H_k(t)$ is small.

Lemma 3: If $\phi_k(t)$ has maximum right-derivative $\nu_k > 0$, if initial queue backlog satisfies $0 \leq Q_k(0) \leq V\nu_k + x_k^{max}$, and if $\alpha_k > 0$, then no access point will send to user k on a given slot t if $H_k(t) > \frac{1}{\alpha_k}[V\nu_k + x_k^{max}]$.

Proof: Lemma 2 ensures $Q_k(t) \leq V\nu_k + x_k^{max}$ for all t. It follows that if $H_k(t) > \frac{1}{\alpha_k}[V\nu_k + x_x^{max}]$, then the weight seen by an access point $a \in \mathcal{A}$ for user k satisfies:

$$\begin{aligned} Q_k(t) - \alpha_k H_k(t) \ &\leq \ V\nu_k + x_k^{max} - \alpha_k H_k(t) \\ &< \ 0 \end{aligned}$$

Because the weight is negative, the max-weight functional (23) is maximized by choosing $\mu_{ak}(t) = 0$, so that access point a will not send data to user k on slot t. □

Now suppose user $u \in \mathcal{K}$ considers transmitting to another user $k \in \mathcal{K}$. User u sees the weight:

$$W_{uk}(t) = Q_k(t) + H_u(t) - \alpha_k H_k(t)$$

The value $H_u(t) - \alpha_k H_k(t)$ can be viewed as a *differential reputation*. We again see that a relatively low value of $H_k(t)$ improves the weights for user k. The next lemma shows that all queues $H_k(t)$ are deterministically bounded. For simplicity, the lemma is stated under the assumption that all initial queue backlogs are zero. Let $\mathbf{\Theta}(t) = (Q_k(t), H_k(t))|_{k\in\mathcal{K}}$ be the vector of all virtual queue values on slot t.

Lemma 4: If all utility functions have right-derivatives bounded by finite constants $\nu_k > 0$, if $\beta_k > 0$ for all $k \in \mathcal{K}$, and if initial backlog satisfies $Q_k(0) = H_k(0) = 0$ for all $k \in \mathcal{K}$, then there are finite constants C_1 and C_2, both independent of V, such that:

$$||\mathbf{\Theta}(t)|| \leq C_1 + C_2 V \ \ \forall t \in \{0, 1, 2, \ldots\}$$

where $||\mathbf{\Theta}(t)||$ is defined as the Euclidean norm:

$$||\mathbf{\Theta}(t)|| \leq \sqrt{\sum_{k\in\mathcal{K}} H_k(t)^2 + \sum_{k\in\mathcal{K}} Q_k(t)^2}$$

Proof: The algorithm makes decisions for $(\mu_{kb}(t))$ and $\gamma_k(t)$ that minimize the last three terms in the right-hand-side of (18) over all alternative feasible decisions, including the trivial decisions $\tilde{\gamma}_k(t) = \tilde{\mu}_{kb}(t) = 0$. Thus:

$$\begin{aligned} \Delta(t) - V\sum_{k\in\mathcal{K}} \phi_k(\gamma_k(t)) \ &\leq \ B(t) - V\sum_{k\in\mathcal{K}} \phi_k(0) \\ &\quad - \sum_{k\in\mathcal{K}} H_k(t)\beta_k \\ &\leq \ B - V\sum_{k\in\mathcal{K}} \phi_k(0) \\ &\quad - \beta_{min}||\boldsymbol{H}(t)|| \end{aligned}$$

where B is a constant that upper bounds $B(t)$ for all t (such a constant exists by the boundedness assumptions on $x_k(t)$, $y_k(t)$, and $\gamma_k(t)$). Further, $\beta_{min} \triangleq \min_{k\in\mathcal{K}} \beta_k$, and we have used the fact that $\sum_{k\in\mathcal{K}} H_k(t) \geq ||\boldsymbol{H}(t)||$. Now define $C_0 \triangleq \sum_{k\in\mathcal{K}}[\phi_k(x_k^{max}) - \phi_k(0)]$. We have for all t:

$$\Delta(t) \leq B + VC_0 - \beta_{min}||\boldsymbol{H}(t)||$$

By definition of $\Delta(t)$:

$$\Delta(t) = \frac{1}{2}||\boldsymbol{\Theta}(t+1)||^2 - \frac{1}{2}||\boldsymbol{\Theta}(t)||^2$$

Thus:

$$||\boldsymbol{\Theta}(t+1)||^2 - ||\boldsymbol{\Theta}(t)||^2 \leq 2(B + VC_0) - 2\beta_{min}||\boldsymbol{H}(t)|| \tag{27}$$

Note by Lemma 2 that for all slots t, we have $0 \leq Q_k(t) \leq Q_{max}$, where:

$$Q_{max} \triangleq \nu_{max}V + x_{max}$$

where $\nu_{max} \triangleq \max_{k\in\mathcal{K}} \nu_k$ and $x_{max} \triangleq \max_{k\in\mathcal{K}} x_k^{max}$. Thus, for any slot t:

$$\begin{aligned} ||\boldsymbol{\Theta}(t)|| &\leq ||\boldsymbol{Q}(t)|| + ||\boldsymbol{H}(t)|| \\ &\leq Q_{max}\sqrt{K} + ||\boldsymbol{H}(t)|| \end{aligned} \tag{28}$$

Now suppose that on slot t, we have:

$$||\boldsymbol{\Theta}(t)|| > \frac{B + VC_0}{\beta_{min}} + Q_{max}\sqrt{K} \tag{29}$$

Combining this with (28) shows that if (29) holds, then:

$$||\boldsymbol{H}(t)|| > \frac{B + VC_0}{\beta_{min}} \tag{30}$$

It follows that if (29) holds, then (by combining (27) and (30)):

$$||\boldsymbol{\Theta}(t+1)||^2 - ||\boldsymbol{\Theta}(t)||^2 < 0$$

Thus, $||\boldsymbol{\Theta}(t)||$ cannot increase if (29) holds on slot t. It follows that for all t:

$$||\boldsymbol{\Theta}(t)|| \leq \frac{B + VC_0}{\beta_{min}} + Q_{max}\sqrt{K} + g$$

where g is defined as the maximum possible increase in $||\boldsymbol{\Theta}(t)||$ in one slot. Because both $Q_k(t)$ and $H_k(t)$ can increase by at most x_{max} in one slot, we have $g \leq x_{max}\sqrt{2K}$. Thus, for all slots t we have:

$$\begin{aligned} ||\boldsymbol{\Theta}(t)|| &\leq \frac{B + VC_0}{\beta_{min}} + (V\nu_{max} + x_{max})\sqrt{K} \\ &\quad + x_{max}\sqrt{2K} \\ &= C_1 + C_2V \end{aligned}$$

where:

$$C_1 \triangleq B/\beta_{min} + x_{max}(\sqrt{K} + \sqrt{2K}) \tag{31}$$

$$C_2 \triangleq C_0/\beta_{min} + \nu_{max}\sqrt{K} \tag{32}$$

□

D. Robustness to Inexact Implementation

The proofs of Lemmas 2 and 4 reveal that the algorithm of Section III-C has the following two properties for all t:

- For each $k \in \mathcal{K}$, $\gamma_k(t)$ is chosen to be 0 whenever $Q_k(t) > V\nu_k$.
- The decisions for $(\gamma_k(t))$ and $(\mu_{nk}(t))$ for each slot t ensure the last three terms in the right-hand-side of (18) sum to a value that is less than or equal to the corresponding sum under the trivial decisions $\gamma_k(t) = 0$, $\mu_{nk}(t) = 0$ for all n, k.

These are the only properties used in the proofs of Lemmas 2 and 4. Hence, the bounded queue results hold more generally under any algorithm that satisfies the above two properties. These properties can often be hardwired into the algorithm even in cases when it is difficult to choose $\gamma_k(t)$ and $(\mu_{nk}(t))$ to exactly solve (21)-(23).

E. Behavior of the Tit-for-Tat Constraints

The deterministic queue bounds provide insight into the tit-for-tat behavior of the system. Indeed, from the queue update equation for $H_k(t)$ in (16), we have for any slot t:

$$H_k(t+1) \geq H_k(t) + \alpha_k x_k(t) - \beta_k - y_k(t)$$

Summing the above over $t \in \{t_0, t_0+1, \ldots, t_0+T-1\}$ for some initial slot $t_0 \geq 0$ and some positive integer T gives:

$$H_k(t_0+T) - H_k(t_0) \geq \sum_{t=t_0}^{t_0+T-1} [\alpha_k x_k(t) - \beta_k - y_k(t)]$$

Rearranging the above inequality and using the fact that $H_k(t_0) \geq 0$ and $H_k(t_0+T) \leq H_k^{max}$ for some finite constant H_k^{max} yields:

$$\alpha_k \frac{1}{T}\sum_{t=t_0}^{t_0+T-1} x_k(t) \leq \beta_k + \frac{1}{T}\sum_{t=t_0}^{t_0+T-1} y_k(t) + \frac{H_k^{max}}{T} \tag{33}$$

Therefore, over any interval of T slots, the average download rate by user k (multiplied by the α_k coefficient) cannot exceed β_k plus the average upload rate, plus a "fudge factor" H_k^{max}/T. The fudge factor becomes arbitrarily small with increased interval size. This is a deterministic result that holds for any sample path and any time interval. Using $t_0 = 0$ and letting $T \to \infty$ in (33) verifies that the time average tit-for-tat constraints are indeed satisfied for all users $k \in \mathcal{K}$.

V. Extension to Finite File Sizes

Now suppose the files requested by users have finite sizes. Suppose each user requests at most one file at a time. We say a user is in the *active* state if it is requesting a file, and is in the *idle* state if it does not have any file requests. For each $k \in \mathcal{K}$, define $A_k(t)$ to be 1 if user k is active on slot t, and 0 else. If $A_k(t) = 1$, define $\mathcal{F}_k(t)$ as the set of devices in $\mathcal{N}$ that have the currently requested file of user k. Define $\mathcal{F}_k(t)$ to be the empty set $\{\}$ if none of the devices have the file that user k wants on slot t, or if the user is not active on slot t. Define $D_k(t)$ as the number of additional required packets for user k to complete its file request (where $D_k(t) > 0$ if and only

if $A_k(t) = 1$). When an active user finishes downloading all packets of its requested file on some slot t, it goes to the idle state, so that $A_k(t+1) = D_k(t+1) = 0$, and $\mathcal{F}_k(t+1) = \{\}$. Further, assume the user always goes from the active state to the idle state after a certain time limit expires, representing a maximum amount of time tolerable for one file download (this allows the user to "timeout" and request a different file if none of the devices have its desired file, or if the network cannot deliver its desired file fast enough).

The algorithm of the previous section can be directly applied to this case. It proceeds exactly as before, still updating queues and making all $\gamma_k(t)$ and $\mu_{ab}(t)$ decisions the same way for all users on every slot, regardless of whether users are active or idle. Of course, the $f_{ab}(t)$ parameters in (23) and in the receive and send equations (2), (3) will remove any transmission link (a, b) from consideration if user b is idle. However, if desired, idle users can still participate in data delivery to other users. If this is not desired, the transmission link from a user u that is currently idle can be shut down by removing u from the current $\mathcal{F}_k(t)$ sets for all other users k.

Since Lemmas 2, 3, and 4 were derived for arbitrary (possibly non-ergodic) $\omega(t)$ and $\mathcal{F}_k(t)$ processes, they equally hold in this context. In particular, if utility functions have bounded right-derivatives $\nu_k > 0$, if $\beta_k > 0$, and if all queues are initially empty, then $Q_k(t) \leq V\nu_k + x_k^{max}$ for all users k and all slots t, and the tit-for-tat guarantee (33) holds for all users k and all intervals of time. However, the utility optimality theorem (Theorem 1) no longer holds in this context. Hence, the resulting algorithm should be viewed as a heuristic. Intuitively, the algorithm will behave well, with performance close to that suggested by the infinite file size assumption, when file sizes are large. This is explored via simulation in the next section.

VI. Simulation for Cell-Partitioned Networks

This section presents simulation results for a network with the cell-partitioned structure of Section II-A. There are $K = 50$ users and a single access point that is a wireless base station. The users move according to a Markov random walk on a 4×4 grid with 16 subcells. Utility functions $\phi_k(x)$ are given by (25) with $\nu_k = 1$. Each cell can support at most one user-to-user packet transmission per slot. The base station can transmit to at most one user k per slot, with transmission rate $S_k(t)$ that is independent over slots and across users with $Pr[S_k(t) = 0] = Pr[S_k(t) = 1] = Pr[S_k(t) = 2] = 1/3$. All simulations run for 10^6 slots.

A. Large Files and Non-Ergodic Changes

The first set of simulations consider a case where all users desire a single infinitely large file, and where the file availability sets change non-ergodically. Specifically, on slot $t = 0$, each user k is assigned a desired file. This file is independently in the cache of other users with probability $p = 0.05$. This establishes the $\mathcal{F}_k$ sets. These sets are held fixed for the first third of the simulation. At the end of the first third of the simulation, new sets are independently drawn using a larger probability $p = 0.1$. These sets are held fixed for the second third of the simulation. New file sets are independently drawn at the beginning of the final third of the simulation, with $p = 0.07$, and held fixed until the end.

Fig. 1 plots the resulting throughput from the base station traffic and peer-to-peer traffic separately, using $V = 10$, $\alpha_k = \alpha = 0.5$, $\beta_k = 0.05$, $x_k^{max} = 3$. Even though there are only an average of 50/16 = 3.125 users per cell, and in the first third of the simulation there is only a 5% chance that a given user has the file desired by another user, the peer-to-peer traffic is still more than twice that of the base station alone. This further increases in the middle of the simulation when the file availability probability jumps to 10%. Overall, these results demonstrate that the algorithm can quickly adapt to non-ergodic changes in the file availability. Fig. 2 shows that the value of $Q_k(t)$ never exceeds 10 packets for any user k (recall that the worst-case guarantee from Lemma 2 is $Q_k(t) \leq V + 3 = 13$ packets). All tit-for-tat constraints were satisfied, with $H_k(t) \leq 24.6$ for all $k \in \mathcal{K}$ and all t.

Figs. 3 and 4 explore the throughput-backlog tradeoff with V (one can also plot the throughput-utility with V to see a similar convergence as in Fig. 3). Fig. 3 and 4 also treat the case when the tit-for-tat constraint is made more stringent ($\alpha = 0.75$), in which case throughput is reduced.

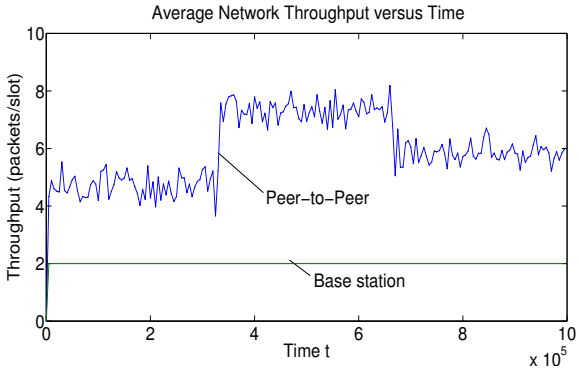

Fig. 1. Average throughput per-user versus time (16 subcells, 50 users). The base station throughput appears constant because the max-weight user that it selects almost always has a transmission rate of $S_k(t) = 2$ (note that there are an average of $50/3$ users with $S_k(t) = 2$ on every slot).

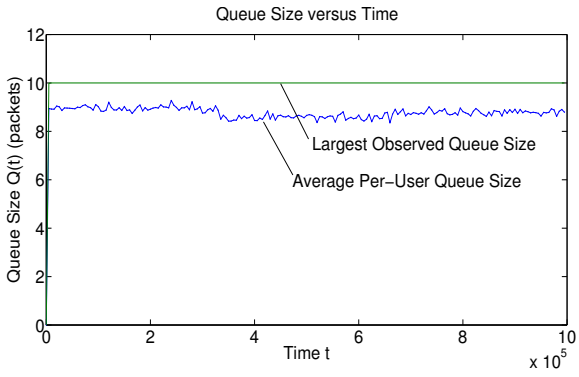

Fig. 2. Average and worst-case queue backlog per user versus time.

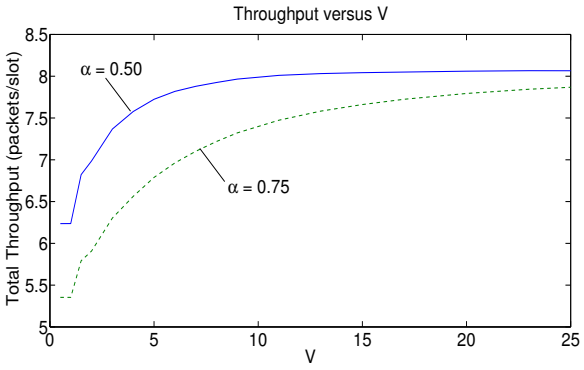

Fig. 3. Throughput versus V for $\alpha = 0.5$ and $\alpha = 0.75$.

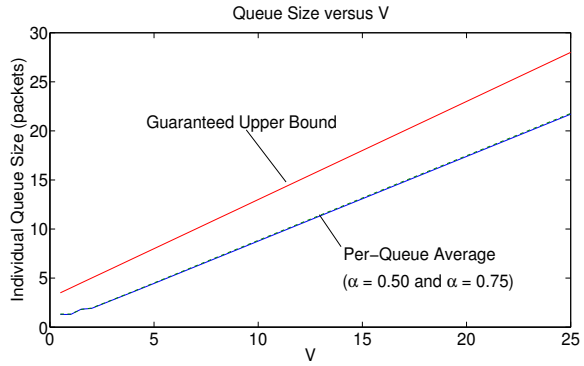

Fig. 4. Queue backlog versus V, demonstrating the $O(V)$ behavior. The results for $\alpha = 0.5$ and $\alpha = 0.75$ are right on top of each other. The upper bound $V + 3$ is also plotted.

B. Finite File Sizes

The second set of simulations consider the same network and the same parameters $V = 10$, $\alpha_k = \alpha = 0.5$, $\beta_k = 0.05$, $x_k^{max} = 3$. However, these simulations assume users transition between active and idle states, as described in Section V. At the beginning of each active period, users independently draw a new file with size that is uniformly distributed over the integers $\{50, 51, \ldots, 150\}$, so that the average file size is 100 packets. The corresponding subset $\mathcal{F}_k(t)$ is independently drawn at the beginning of each active period and held fixed for the duration of this period. The $\mathcal{F}_k(t)$ set is formed by assuming each device $n \neq k$ independently has the file desired by user k with probability p. The timeout value is set to ∞, so that a user only goes to the idle state when it finishes downloading its file. After a user spends one slot in the idle state, it transitions to the active state independently with probability q, and stays in the idle state with probability $1-q$. Thus, each idle period lasts for an average of $1/q$ slots.

Fig. 5 illustrates a sample path of throughput for the case when the file availability probability is $p = 0.05$. The idle-to-active transition probability is $q = 1/100$ so that the average duration of an idle period is 100 slots. The figure separates out the throughput due to base station traffic and peer-to-peer traffic, again illustrating the significant gains available from wireless peering. The average file delay for this experiment was 809.37 slots, where file delay is defined as the duration of time required to deliver all packets of the file (being the duration of an active period).[7] The average number of delivered files per user was 1231.9 files during the 10^6 slots.

Fig. 6 presents throughput results when q is varied between 1 and $1/400$ (so that average idle time varies between 1 and 400). Curves are given for the case $p = 0.05$ and $p = 0.025$. The base station curves are roughly the same for both cases, whereas the peer-to-peer throughput significantly increases when the file availability probability is doubled. Fig. 7 plots the resulting average file download delay.

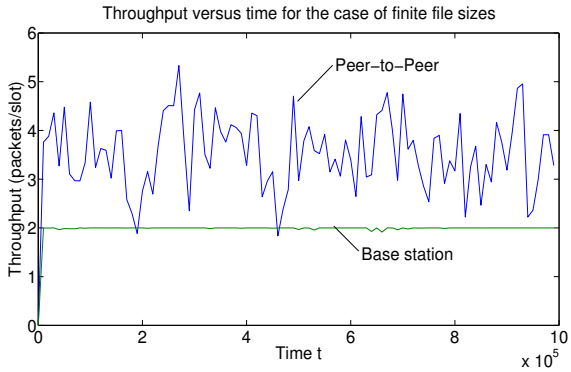

Fig. 5. A sample path of peer-to-peer throughput and base station throughput versus time for the case of finite file sizes with file availability probability $p = 0.05$. The idle-to-active transition probability is $q = 1/100$ so that the average duration of an idle period is 100 slots.

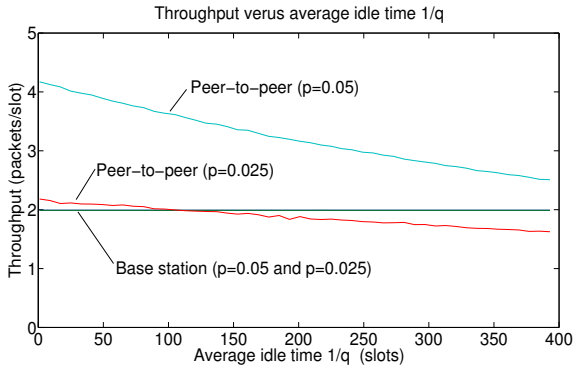

Fig. 6. Throughput versus $1/q$ for peer-to-peer traffic and base station traffic, for file availability probability $p = 0.05$ and $p = 0.025$.

C. Advantages of Scale

This subsection illustrates the dramatic throughput gains achievable by scaling the system up to allow many more users and many more subcells. The scenario of Section VI-A, Fig. 1, is repeated with the following modification:

- Previous simulation (Fig. 1): 1 base station, 50 mobile users, 16 subcells arranged in a 4×4 grid.
- Modified simulation (Fig. 8): 1 base station, 1250 mobile users, 400 subcells arranged in a 20×20 grid.

The new scenario maintains the same user/cell ratio 3.125, but allows for many more subcells and hence many more

[7]Recall that the average file size is 100 packets. Thus, an ideal non-mobile setting where there are only two users and one sends a single packet to the other on every slot would yield an average delay of 100 slots.

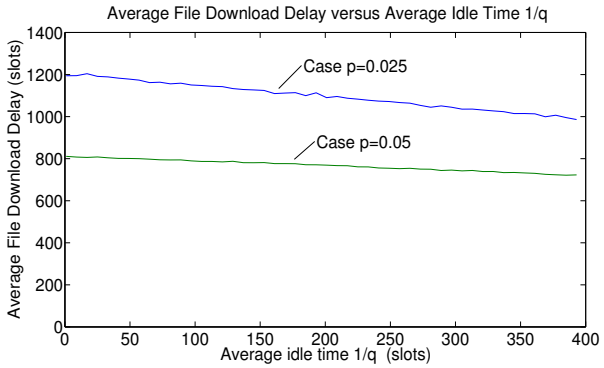

Fig. 7. Average file download delay versus $1/q$, for file availability probability $p = 0.05$ and $p = 0.025$.

simultaneous transmissions. The resulting average peer-to-peer traffic is 153.7 packets per slot while the average base station traffic is only 2 packets per slot (compare Figs. 8 and 1).

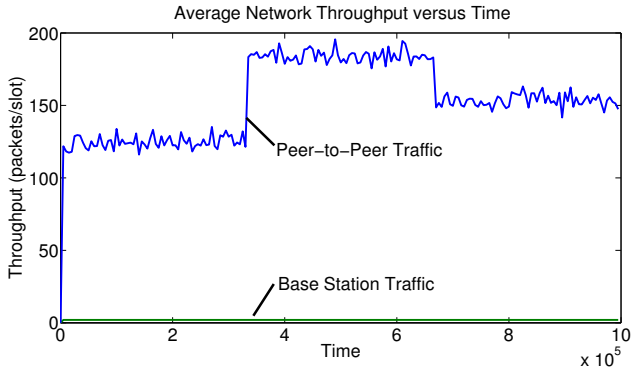

Fig. 8. Average throughput versus time for the same situation as Fig. 1, but with many more subcells and users (400 subcells, 1250 users).

VII. Simulation for SINR-Based Networks

This section presents a simulation for the SINR-based transmission model from Section II-B. To illustrate the flexibility of the model, this section also assumes there is no user-to-user communication, so the tit-for-tat constraints are removed by setting $\alpha_k = 0$ for all $k \in \mathcal{K}$ (see (8)). Instead, users can receive transmissions either from the base station or from one of multiple femto "helper" nodes.

Assume the network region is square with unit length sides. There is one base station that is located in the middle of the square. There are 40 femto nodes that are randomly and independently placed uniformly over the unit square at time 0. These femto nodes never change their locations. There are 200 users that independently move about the network as follows: The network area is divided into a 10×10 region of subcells (used only to simplify the mobility simulation). The first 100 users independently choose a subcell at time zero. These users stay in that same subcell for the duration of the simulation, but independently move to a new location in that subcell every slot (uniformly distributed over the subcell). The next 100 users take independent random walks over the subcells as follows: With probability 0.8 the user stays in the current subcell, but moves to a uniformly chosen location in that same subcell. With probability 0.2, the user moves to a neighboring subcell (uniformly chosen over all neighbor cells), and chooses a random location in that new subcell.

Let $S_{nk}(t)$ denote the attenuation between device n (either a base station or femto node) and user k, and assume:

$$S_{nk}(t) = \min\left[\frac{(1/20)^3}{dist(n,k,t)^3}, 1\right]$$

where $dist(n,k,t)$ is the distance between devices n and k on slot t, being a number between 0 and $\sqrt{2}$. The above takes a minimum with 1 to ensure $S_{nk}(t) \leq 1$ for all t. The SINR is then computed via the formula in Section II-B. For simplicity, we use a continuous rate function $g_{nk}(s) = \ln(1+s)$. We use $P_n = P$ and $\sigma_k^2 = \sigma^2$ for all transmitters n and receivers k, with a ratio $P/\sigma^2 = 100$. See [36] and references therein for details on related assumptions for SINR models.

The base station has all files, whereas the femto nodes have only a subset of the files (chosen independently for each femto node). To allow all users to have unimpeded access to the base station if needed, every 10th slot all femto nodes go idle and the base station alone transmits to the single max-weight user (as determined by the weights in Section III-F). On all other slots, the base station transmits together with all 40 femto nodes according to the max-weight rule of Section III-F.[8] The simulation mimics that of Section VI-A, with the exception that the file availability probability at the femto nodes is increased to $p = 0.1, p = 0.2, p = 0.15$ at the first, second, and third phases of the simulation. This is because such femto nodes typically have a larger set of files as compared to wireless peers. Results are shown in Fig. 9, where it is again clear that the femto node traffic is significantly larger than the base station traffic. The dashed line in the figure indicates the sum throughput that would be achieved if the base station acted alone (with no femto nodes) on each and every slot. Overall, it is evident that the femto nodes significantly boost total network throughput.

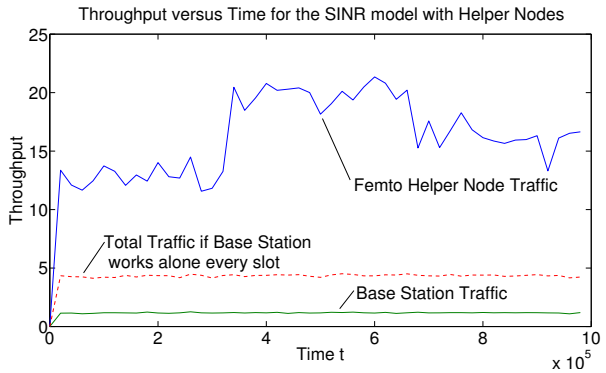

Fig. 9. A sample path of throughput for the SINR model with $V = 10$ and 200 users, 40 femto nodes, and 1 base station. The file availability probability at the femto nodes is 0.1, 0.2, 0.15 for the three phases of the simulation.

[8]Of course, one could determine the *optimal* fraction of time to have the base station transmit alone by comparing the max-weight metric for the cases when all femto nodes are idle and when all are busy, and choosing the best of the two. This modification is omitted for brevity.

VIII. Average Power Constraints

This section modifies the problem formulation to include average power constraints at each user. Assume for simplicity that a 1-slot transmission over a link (n, k) expends c units of power at the transmitting device n and d units of power at the receiving device k. Then the power expended by user k on slot t is:

$$p_k(t) = c\sum_{j\in\mathcal{K}} 1_{\{\mu_{kj}(t)>0\}} + d\sum_{n\in\mathcal{N}} 1_{\{\mu_{nk}(t)>0\}}$$

where $1_{\{\mu_{nk}(t)>0\}}$ is an indicator function that is 1 if $\mu_{nk}(t) > 0$, and 0 else. Let p_k^{av} be positive numbers that represent average power constraints for each user $k \in \mathcal{K}$. The problem becomes:

$$\text{Maximize:} \quad \textstyle\sum_{k\in\mathcal{K}} \phi_k(\overline{x}_k) \tag{34}$$

$$\text{Subject to:} \quad \alpha_k\overline{x}_k \le \beta_k + \overline{y}_k \ \ \forall k \in \mathcal{K} \tag{35}$$

$$\overline{p}_k \le p_k^{av} \ \ \forall k \in \mathcal{K} \tag{36}$$

$$(\mu_{nk}(t)) \in \mathcal{R}(\omega(t)) \ \ \forall t \in \{0, 1, 2, \ldots\} \tag{37}$$

The only difference between the above problem and the original problem (4)-(6) is the addition of the average power constraints in (36). This is easily incorporated by a virtual queue $Z_k(t)$ for each $k \in \mathcal{K}$ with the following update equation [18][15]:

$$Z_k(t+1) = \max[Z_k(t) + p_k(t) - p_k^{av}, 0] \tag{38}$$

Define the modified Lyapunov function:

$$L(t) = \frac{1}{2}\sum_{k\in\mathcal{K}}[Q_k(t)^2 + H_k(t)^2 + Z_k(t)^2]$$

This results in the following drift-plus-penalty expression (compare with (18)):

$$\begin{aligned}\Delta(t) - V\sum_{k\in\mathcal{K}}\phi_k(\gamma_k(t)) \le \tilde{B}(t) - V\sum_{k\in\mathcal{K}}\phi_k(\gamma_k(t)) \\ + \sum_{k\in\mathcal{K}} H_k(t)[\alpha_k x_k(t) - \beta_k - y_k(t)] \\ + \sum_{k\in\mathcal{K}} Q_k(t)[\gamma_k(t) - x_k(t)] \\ + \sum_{k\in\mathcal{K}} Z_k(t)\left[p_k(t) - p_k^{av}\right]\end{aligned}$$

where $\tilde{B}(t)$ is a second moment term that is similar to $B(t)$ but includes the second moments of the $Z_k(t)$ queue changes. Minimizing the last four terms on the right-hand-side of the above inequality leads to the following modified algorithm: Every slot t, the $\gamma_k(t)$ decisions are the same as (21)-(22). However, the transmission decisions are made by choosing a matrix $(\mu_{nk}(t)) \in \mathcal{R}(\omega(t))$ to maximize the following quantity (breaking ties arbitrarily):

$$\begin{aligned}\sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{K}} \mu_{nk}(t) f_{nk}(t) W_{nk}(t) \\ - \sum_{n\in\mathcal{N}}\sum_{k\in\mathcal{K}} 1_{\{\mu_{nk}(t)>0\}}[Z_k(t)d + Z_n(t)c1_{\{n\in\mathcal{K}\}}]\end{aligned}$$

where the weights $W_{nk}(t)$ are the same as before and $1_{\{n\in\mathcal{K}\}}$ is an indicator function that is 1 if $n \in \mathcal{K}$ and 0 else. The second term in the above expression effectively defines a negative bias of $Z_k(t)d + Z_n(t)c1_{\{n\in\mathcal{K}\}}$ for activating a link (n, k). Such activation will only occur if $\mu_{nk}(t)f_{nk}(t)W_{nk}(t)$ exceeds this bias. Using the Lyapunov optimization theory of [15], a modified version of Theorem 1 shows that the algorithm always satisfies the desired constraints (35)-(37) while achieving utility within $O(1/V)$ of optimal. Of course, optimal utility here is redefined over all algorithms that satisfy all of the constraints, including the new average power constraints.

IX. Conclusions

This work develops a peer-to-peer scheduling algorithm for mobile wireless networks. In the case when each user desires a single file of infinite length, the algorithm is shown to provide utility that can be pushed arbitrarily close to optimal, with a tradeoff in queue size. To incentivize participation, the algorithm embeds tit-for-tat constraints into the optimization. This results in a max-weight scheduling decision that can be viewed as a backpressure mechanism acting on the *differential reputation* between users. The resulting algorithm extends as a heuristic to more practical cases of finite file sizes and non-ergodic events. Simulations demonstrate the significant gains achievable by wireless peering and/or femto node transmissions, particularly when the file availability probability is large.

References

[1] M. J. Neely. Wireless peer-to-peer scheduling in mobile networks. *Proc. 46th Conf. on Information Sciences and Systems (CISS)*, March 2012.

[2] M. J. Neely and L. Golubchik. Utility optimization for dynamic peer-to-peer networks with tit-for-tat constraints. *Proc. IEEE INFOCOM*, 2011.

[3] L. Buttyan and J.-P. Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *ACM/Kluwer Mobile Networks and Applications (MONET)*, vol. 8, no. 5, pp. 579-592, Oct. 2003.

[4] J. Crowcroft, R. Gibbens, F. Kelly, and S. Ostring. Modeling incentives for collaboration in mobile ad-hoc networks. *presented at the 1st Int. Symp. Modeling and Optimization in Mobile, Ad-Hoc, and Wireless Networks (WiOpt '03), Sophia-Antipolis, France*, March 2003.

[5] M. J. Neely. Optimal pricing in a free market wireless network. *Wireless Networks*, vol. 15, no. 7, pp. 901-915, Oct. 2009.

[6] S. Jun and M. Ahamad. Incentives in bittorrent induce free riding. In *ACM SIGCOMM Workshop on Economics of Peer-to-Peer Systems (P2P-ECON)*, pages pp. 116–121, Philadelphia, PA, August 2005.

[7] A. R. Bharambe, C. Herley, and V. N. Padmanabhan. Analyzing and improving bittorrent performance. In *Proc. IEEE INFOCOM*, Barcelona, Catalunya, Spain, April 2006.

[8] K. Tamilmani, V. Pai, and A. E. Mohr. Swift: A system with incentives for trading. In *P2P-ECON*, 2004.

[9] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy, and A. Venkataramani. Do incentives build robustness in bittorrent? In *The 4th USENIX Symposium on Networked Systems Design & Implementation (NSDI)*, Cambridge, MA, April 2007.

[10] W.-C. Liao, F. Papadopoulos, and K. Psounis. Performance analysis of bittorrent-like systems with heterogeneous users. In *Performance*, 2007.

[11] Q. Lian, Y. Peng, M. Yang, Z. Zhang, Y. Dai, and X. Li. Robust incentives via multi-level tit-for-tat. In *Proc. 5th International Workshop on Peer-to-Peer Systems (IPTPS)*, 2006.

[12] M. J. Freedman, C. Aperjis, and R. Johari. Prices are right: Managing resources and incentives in peer-assisted content distribution. In *Proc. 7th International Workshop on Peer-to-Peer Systems (IPTPS)*, 2008.

[13] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transacations on Automatic Control*, vol. 37, no. 12, pp. 1936-1948, Dec. 1992.

[14] L. Georgiadis, M. J. Neely, and L. Tassiulas. Resource allocation and cross-layer control in wireless networks. *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1-149, 2006.

[15] M. J. Neely. *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool, 2010.
[16] L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, vol. 39, no. 2, pp. 466-478, March 1993.
[17] M. J. Neely. *Dynamic Power Allocation and Routing for Satellite and Wireless Networks with Time Varying Channels*. PhD thesis, Massachusetts Institute of Technology, LIDS, 2003.
[18] M. J. Neely. Energy optimal control for time varying wireless networks. *IEEE Transactions on Information Theory*, vol. 52, no. 7, pp. 2915-2934, July 2006.
[19] M. J. Neely, E. Modiano, and C. Li. Fairness and optimal stochastic control for heterogeneous networks. *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 396-409, April 2008.
[20] S. H. Low and D. E. Lapsley. Optimization flow control, i: Basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, vol. 7 no. 6, pp. 861-875, Dec. 1999.
[21] F. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, vol. 8, no. 1 pp. 33-37, Jan.-Feb. 1997.
[22] L. Xiao, M. Johansson, and S. P. Boyd. Simultaneous routing and resource allocation via dual decomposition. *IEEE Transactions on Communications*, vol. 52, no. 7, pp. 1136-1144, July 2004.
[23] M. Chiang. Balancing transport and physical layer in wireless multihop networks: Jointly optimal congestion control and power control. *IEEE J. on Selected Areas in Comm.*, vol. 23, no. 1, pp. 104-116, Jan. 2005.
[24] 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.
[25] A. Eryilmaz and R. Srikant. Joint congestion control, routing, and MAC for stability and fairness in wireless networks. *IEEE Journal on Selected Areas in Communications, Special Issue on Nonlinear Optimization of Communication Systems*, vol. 14, pp. 1514-1524, Aug. 2006.
[26] A. Stolyar. Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm. *Queueing Systems*, vol. 50, no. 4, pp. 401-457, 2005.
[27] H. Kushner and P. Whiting. Asymptotic properties of proportional-fair sharing algorithms. *Proc. 40th Annual Allerton Conf. on Communication, Control, and Computing, Monticello, IL*, Oct. 2002.
[28] R. Agrawal and V. Subramanian. Optimality of certain channel aware scheduling policies. *Proc. 40th Annual Allerton Conf. on Communication, Control, and Computing, Monticello, IL*, Oct. 2002.
[29] X. Lin, N. B. Shroff, and R. Srikant. A tutorial on cross-layer optimization in wireless networks. *IEEE Journal on Selected Areas in Communications, Special Issue on Nonlinear Optimization of Communication Systems*, vol. 14, no. 8, Aug. 2006.
[30] S. Shakkottai and R. Srikant. Network optimization and control. *Foundations and Trends in Networking*, vol. 2, no. 3, pp. 271-379, 2007.
[31] X. Wu, S. Tavildar, S. Shakkottai, T. Richardson, J. Li, R. Laroia, and A. Jovcic. FlashLinkQ: A synchronous distributed scheduler for peer-to-peer ad-hoc networks. *Proc. 48th Annual Allerton Conf., Moticello, IL*, 2010.
[32] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali. Routing without routes: The backpressure collection protocol. *Proc. 9th ACM/IEEE Intl. Conf. on Information Processing in Sensor Networks (IPSN)*, April 2010.
[33] M. J. Neely, E. Modiano, and C. E Rohrs. Dynamic power allocation and routing for time varying wireless networks. *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 89-103, January 2005.
[34] M. Grossglauser and D. Tse. Mobility increases the capacity of ad-hoc wireless networks. *IEEE/ACM Trans. on Networking*, vol. 10, no. 4, pp. 477-486, August 2002.
[35] M. J. Neely. Stability and probability 1 convergence for queueing networks via Lyapunov optimization. *Journal of Applied Mathematics*, vol. 2012, doi:10.1155/2012/831909, 2012.
[36] H. S. Dhillon, R. K. Ganti, F. Baccelli, and J. G. Andrews. Modeling and analysis of k-tier downlink heterogeneous cellular networks. *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 3, pp. 550-560, April 2012.