Increasing Opportunistic Gain in Small Cells Through Base Station-Driven Traffic Spreading

Qing Wang[†‡] Balaji Rengarajan[†] Joerg Widmer[†]
[†]IMDEA Networks Institute, Madrid, Spain [‡]University Carlos III of Madrid, Spain
Email: {qing.wang, balaji.rengarajan, joerg.widmer}@imdea.org

Abstract—**Dense deployment of small cells is an important, emerging trend to enable future cellular networks to cope with growing traffic demand. However, this reduces the number of users per cell and thus opportunistic scheduling gain. We propose a base station-driven energy-aware approach to exploit user-user communication to increase the opportunistic gain. We use tools from stochastic Lyapunov optimization to determine the optimal scheduling policy subject to a constraint on energy consumption for user-user communication. Our simulation results show that with a large energy budget, packet transfer delay is reduced by up to 70%. The bulk of the performance improvement can be achieved with only a small increase in energy consumption, where 60% of the improvement is achieved at only 20% of the additional energy consumption. Further, we evaluate our algorithm using realistic video traffic traces and show that frame loss ratio is reduced by 90% and PSNR is improved by 4dB.**

I. Introduction

Opportunistic scheduling [1, 2] was proposed for multiuser wireless networks and has been widely adopted in practical cellular systems [3, 4]. An opportunistic scheduler exploits the time-varying channels between the base station (BS) and users, to improve the overall system performance.

In this paper, we consider a setting where a BS serves a set of users with stochastic traffic loads, similar to [5–10]. In such a scenario, the BS at times has no data to transmit to some users and multi-user diversity is reduced. This does not affect performance in large cells (with large user populations) since opportunistic gain scales as a concave function of the number of available users. However, as smaller and smaller cells are deployed more and more densely to increase wireless capacity and meet increasing traffic demands [11–13], the average number of users in a cell will decrease significantly and a time-varying user population may greatly affect the performance of opportunistic scheduling.

We propose a BS-drIven Traffic Spreading (BITS) algorithm that can increase the opportunistic scheduling gain in small cells. We consider downlink communication that accounts for most of the traffic in a cellular network [14]. The BITS algorithm benefits applications whose performance is sensitive to delay (distribution) of received packets. Such delay-sensitive applications include live streaming, video-conferencing, etc. For instance, in live streaming the video frames which arrive late will be dropped by the video player at the user, thus reducing video quality.

To exploit the user-user communication, BITS leverages the multiple radio interfaces (e.g., 3G, WiFi) available in most smartphones. The criteria used by BITS are the users' current channel conditions and queue backlogs. Each user measures its perceived channels to other users and shares channel information with the BS. BITS takes into account users' backlogs as well as the BS-user and user-user channels, to maximize its scheduling options and hence increase opportunistic gain.

To illustrate the traffic spreading mechanism of BITS, let us consider the example shown in Fig. 1, where two users (U_1 and U_2) are served by a BS. The queues Q_1 and Q_2 depict the number of packets waiting to be sent to each user. The users perceive similar channel statistics and have similar traffic loads. During each time slot, BITS determines from which queue packets are served and to which user packets are sent (i.e., when and how to spread traffic). In Fig. 1 (a), the queues are balanced. From Little's law, the average packet delays of Q_1 and Q_2 are similar (below/above the delay threshold of user's player). Thus traffic spreading is not beneficial and BITS sends packets to the corresponding users directly. In Fig. 1 (b), there are more packets in Q_2 than in Q_1 which is nearly empty. This implies the average packet delay of Q_2 is higher than that of Q_1. Thus in the near future, (more and more) packets in Q_2 are likely to become useless when they arrive at the user. BITS reacts to this imbalance in queues and if the channel of U_1 is better than U_2, BITS sends packets from Q_2 to U_1, who then forwards them to U_2 through the user-user link.

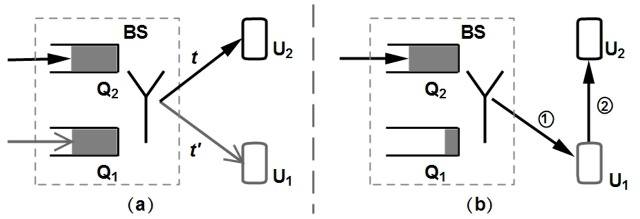


Fig. 1: An example of BITS: (a) no spreading; (b) with spreading.

The proposed algorithm incurs additional energy consumption due to forwarding traffic among users. As mobile devices have limited energy resources, excessive traffic spreading can result in high penalties in terms of energy consumption. At the same time, balancing energy consumption among the users is an important consideration. Our energy-aware BITS algorithm optimizes the degree of spreading for a given energy constraint.We summarize our main contributions as follows:

1) We propose an energy-aware scheduling policy (BITS) to increase opportunistic gain by taking into account

users' backlogs and the BS-user and user-user channels.

2) We model the BITS policy with the objective to maximize delay-sensitive utility under an energy constraint. Using stochastic Lyapunov optimization, we develop an online algorithm and study its properties.
3) We evaluate BITS using realistic Rayleigh fading channels. Simulation results show that under BITS: *i)* utility is increased greatly and average packet transfer delay is reduced by up to 70% even in homogeneous scenarios; *ii)* in the energy-constrained case (i.e., small energy budget), significant gains (up to 60% of the gain) are typically achieved at only 20% of the energy consumption of performance-centric case (i.e., with a sufficiently large energy budget); *iii)* excellent *fairness* on additional energy consumption among users can be achieved in heterogeneous scenarios; *iv)* the performance can be improved greatly in scenarios where overall system performance under the proportionally fair scheduling policy used in current 3G and 4G systems is very poor.
4) We also evaluate BITS using realistic video traffic traces. We provide results showing that with BITS, the average Peak Signal-to-Noise Ratio (PSNR) of the received video can be improved by up to 4dB and the frame loss ratio is reduced by up to 90%. Moreover, the quality of the received video varies much more slowly with fluctuations of the wireless channel.

The rest of this paper is organized as follows: the related work is summarized in Section II, followed by the system model and stochastic Lyapunov optimization in Sections III and IV, respectively. The properties of BITS are discussed in Section V. Performance evaluation in a multi-channel system and evaluation using realistic video traces are presented in Section VI. Finally, conclusions and possible directions for future work are provided in Section VII.

II. Related Work

Many scheduling algorithms considering both users' backlogged queues and channel states have been proposed [7–9]. Among these, [7, 8] propose throughput-optimal MaxWeight and Exponential rules, respectively. Authors in [9] propose the log rule to improve delay performance. All these algorithms react to imbalance in users' queues by sacrificing opportunistic gain in order to balance queues. In contrast, BITS can balance users' backlogs without losing instantaneous gain, by opportunistically exploiting the BS-user and user-user channels.

Another class of scheduling algorithms aims to maximize delay/time-sensitive utility [15, 16], as BITS does. Among these, [16] proposes to maximize delay-sensitive utility to provide delay QoS for each user, while [15] aims to maximize the time average utility to enforce fairness. Compared to our work, the utilities in [15, 16] are defined as concave functions of packet queueing delay, while the utility used in BITS (cf. Section III) is a function of throughput, where throughput is sensitive to packet delay.

Wang et al. [17] propose a downlink BS-transparent dispatching policy where users spread traffic requests among each other to balance their backlogs. This increases the BS scheduling options and hence improves the performance. Compared to BITS, the dispatching policy is user-initiated and on a per-file basis, while BITS is BS-driven and operates on a per-packet basis. Further, the dynamic programming is used in [17] to determine the optimal dispatching policy and the complexity in large systems is reduced by aggregating users. In BITS, the algorithm is derived from Lyapunov optimization and the complexity is low even in large systems.

Another approach to exploit both the BS-user and user-user channels is opportunistic relaying [10, 18–20]. Among these, [18] proposes the idea of opportunistic relaying and an approach of choosing the best relay that maximizes the minimal quality of BS-relay and relay-user channels. In [10, 19, 20] mobile users themselves, instead of particular relay nodes are used as relays. The work in [20] considers relaying traffic to areas without cellular coverage and proposes an approach where a user with the best channel to the destination is chosen as the relay. Authors in [10, 19] propose scheduling algorithms to improve the system capacity and fairness. Compared to BITS, [18–20] assume users have infinitely backlogged queues, which is different from BITS and [10] that consider stochastic traffic loads. Moreover, the delay-sensitive utility as well as delay-energy tradeoff have not been investigated in neither [10] nor the other works discussed above.

III. System Model

We consider a time-slotted system with N users attached to a single BS, where the set of users is denoted as $\mathcal{I} = \{1, 2, ..., N\}$. The BS maintains a separate queue for each user, and we denote by $\boldsymbol{Q}(t) \equiv (Q_i(t), i \in \mathcal{I}) \in \mathbb{N}^N$, the number of packets waiting to be sent to each user at slot t. Without loss of generality, packet sizes are fixed. The number of packets that can be sent during a slot depends on the modulation scheme. The arrival rates of packets to the BS are modelled through the vector $\boldsymbol{\lambda}(t) = \{\lambda_i(t), i \in \mathcal{I}\}$ where $\lambda_i(t)$ denotes the number of packets that arrive to queue i during slot t and $\lambda_i(t)$ can be arbitrarily bursty. The arrival processes are assumed to be independent across users.

Channel model: The channel is time-varying. The channel instances of the BS-user channel at slot t are denoted as $\boldsymbol{c}(t) = \{c_i(t), i \in \mathcal{I}\}$, where $c_i(t)$ is the maximal number of packets that can be sent to user i if the BS chooses to serve user i. We assume the BS is aware of users' channel states. The set of all the possible channel instances is $\mathcal{S} = \{c_1^*, c_2^*, \cdots, c_K^*\}$.

Scheduling policy: In each slot t, the BS scheduler decides from which queue packets are served and to which user the packets are sent. This decision takes into account current queue states and channel states. The scheduling policy is defined through a binary indicator, $\forall i, j \in \mathcal{I}$ and $\forall t$:

$$\sigma_i^j(t) = \begin{cases} 1, & \text{if serving user } j \text{ with packets from queue } i \\ 0, & \text{otherwise} \end{cases}$$

The set of all the possible scheduling policies is defined as

$$\mathcal{C} \equiv \left\{ \boldsymbol{\sigma}(t) : \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} \sigma_i^j(t) = 1, \ \sigma_i^j(t) \in \{0, 1\} \right\} .$$

We assume packets in the same queue are served according to a first-come-first-served discipline and the BS can only serve one user during a slot. The departure rate $\nu_i(t)$ of queue i can be written as

$$\nu_i(t) = \begin{cases} \min[Q_i(t), c_j(t)], & \text{if } \sigma_i^j(t) = 1 \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Finally, the queueing dynamics of the system are

$$Q_i(t+1) = Q_i(t) - \nu_i(t) + \lambda_i(t), \quad \forall i \in \mathcal{I}, \forall t. \quad (2)$$

A. Delay-sensitive utility

Let $M_i^t(d_x, d_y)$ denote the number of packets received by user i in slot t, with packet delay between d_x and d_y. For each user i, let $\boldsymbol{d}_i^* = \{d_{i,l}^*, l \in \{1, 2, ..., L\}\}$ be the delay thresholds that determine the utility of a packet. The delay-sensitive utility of user i is defined as

$$\phi(\overline{\mu}_i) = \log(\overline{\mu}_i) = \log \sum_{l=1}^{L-1} w_i^l \lim_{t\to\infty} \frac{\sum_t M_i^t(d_{i,l}^*, d_{i,l+1}^*)}{t} \quad (3)$$

where $\overline{\mu}_i$ is the weighted average throughput, $\boldsymbol{w}_i = \{w_i^l, l \in \{1, 2, ...L-1\}\}$ is a weight vector. We use the non-decreasing and concave logarithmic utility function to provide fairness among users. The delay thresholds of different applications may be different. For instance, in live streaming packets that arrive late are dropped at the player and become useless. Under this case, the weight vector can be modelled as $\boldsymbol{w}_i = \{1, 0\}$. In other interactive applications such as gaming, the weights can be modelled by quantizing packet delay and setting the weights in a piecewise constant manner with respect to delay.

Performance metrics: The metrics we use are the delay-sensitive utility and *re-routing cost*, i.e., the additional energy consumption induced by traffic spreading. We define $\boldsymbol{p}^* = \{p_i^*, i \in \mathcal{I}\}$ as the energy budget per slot (i.e., power) and $\boldsymbol{p}(t) = \{p_i(t), i \in \mathcal{I}\}$ as the re-routing cost in slot t, which depends on the scheduling policy $\boldsymbol{\sigma}(t)$.

Our objective is to maximize the sum of users' utilities, subject to users' energy budget:

$$\max_{\boldsymbol{\sigma}(t)} \quad \sum_{i\in\mathcal{I}} \phi(\overline{\mu}_i) \quad (4)$$
$$\text{s.t.} \quad \overline{p}_i \leq p_i^*, \ \forall i \in \mathcal{I}$$

where $\phi(\overline{\mu}_i)$ is given in (3) and $\overline{p}_i$ is the time average of $p_i(t)$ of user i.

IV. Stochastic Lyapunov Optimization

We use stochastic Lyapunov optimization to solve the problem given in (4). Since $\phi(\cdot)$ is a concave function, we first transform the above problem (with functions of time averages) to a problem including only time averages, then use the drift-plus-penalty framework proposed in [21] to solve it.

A. Problem transformation

The original problem (4) is transformed by adding a rectangle constraint and auxiliary variables. Define a rectangle constraint $\mathcal{R} \equiv \{(\mu_1, ..., \mu_N) \in \mathbb{R}^N | 0 \leq \mu_i \leq \gamma_i^{max}, \ \forall i \in \mathcal{I}\}$ where γ_i^{max} is a finite constant. Further, denote by ϕ^* the maximum utility of problem (4), augmented with the constraint $\mathcal{R}$. For each slot t, denote by $\boldsymbol{\gamma}(t) = \{\gamma_i(t), i \in \mathcal{I}\}$ a vector of auxiliary variables within the rectangle constraint set $\mathcal{R}$ and assume $\overline{\gamma}_i \leq \overline{\mu}_i, \forall i \in \mathcal{I}$, where $\overline{\gamma}_i$ is the average of $\gamma_i(t)$. According to [21], we can consider the following transformed problem with only time averages instead of the problem including functions of time averages in (4):

$$\max_{\boldsymbol{\sigma}(t)} \quad \sum_{i\in\mathcal{I}} \overline{\phi(\gamma_i)} \quad (5)$$
$$\text{s.t.} \quad \overline{p}_i - p_i^* \leq 0, \ \forall i \in \mathcal{I} \quad (6)$$
$$\overline{\gamma}_i \leq \overline{\mu}_i, \ \forall i \in \mathcal{I} \quad (7)$$
$$\boldsymbol{\gamma}(t) \in \mathcal{R}, \ \forall t \in \{0, 1, 2, ...\} \quad (8)$$

where $\overline{\phi(\cdot)}$ is the time average of the utility function $\phi(\cdot)$.

B. Lyapunov optimization

We introduce two virtual queues $\boldsymbol{Z}(t)$ and $\boldsymbol{G}(t)$ as follows:

$$Z_m(t+1) = \max[Z_m(t) + p_m(t) - p_m^*, 0], \ \forall m \in \mathcal{I} \quad (9)$$
$$G_s(t+1) = \max[G_s(t) + \gamma_s(t) - \mu_s(t), 0], \ \forall s \in \mathcal{I} \quad (10)$$

We assume $Z_m(0) = 0$, $G_s(0) = 0, \forall m, s \in \mathcal{I}$. The $p_m(t)$ and $\gamma_s(t)$ in (9) and (10) can be viewed as arrival rates, while p_m^* and $\mu_s(t)$ as departure rates. If the queues $Z_m(t)$ and $G_s(t)$ are stable, i.e., $\lim_{t\to\infty} \mathbb{E}\{Z_m(t)\}/t = 0$ and $\lim_{t\to\infty} \mathbb{E}\{G_s(t)\}/t = 0$, then the constraints in (6) and (7) can be satisfied [21]. Thus, to ensure that users' additional energy consumption $\overline{p}_s$ is below the energy budget p_s^* and the auxiliary variable $\overline{\gamma}_m$ is within the rectangle constraint set $\mathcal{R}$, the virtual queues $Z_m(t)$ and $G_s(t)$ have to be stable over time. Therefore, define a quadratic Lyapunov function as:

$$L(\boldsymbol{\Theta}(t)) \equiv \frac{1}{2} \left[\sum_{i\in\mathcal{I}} Q_i(t)^2 + \sum_{m\in\mathcal{I}} Z_m(t)^2 + \sum_{s\in\mathcal{I}} G_s(t)^2 \right],$$

where $\boldsymbol{\Theta}(t) \equiv [\boldsymbol{Q}(t), \boldsymbol{Z}(t), \boldsymbol{G}(t)]$. For any non-negative constant V, define the one-slot Lyapunov drift-plus-penalty as

$$\Delta(\boldsymbol{\Theta}(t)) - V\mathbb{E}\{\phi(\boldsymbol{\gamma}(t))\} \equiv \mathbb{E}\{L(\boldsymbol{\Theta}(t+1)) - L(\boldsymbol{\Theta}(t))\} - V\mathbb{E}\{\phi(\boldsymbol{\gamma}(t))\} \quad (11)$$

To solve the transformed problem (5)-(8), we only need to minimize (11) in each slot [21]. The intuitions behind this are $i)$ by minimizing the drift $\Delta(\boldsymbol{\Theta}(t))$, the virtual queues $Z_m(t)$ and $G_s(t)$ will be stable and thus the constraints in (6) and (7) are satisfied; $ii)$ similarly, (5) is solved by minimizing the penalty $-V\mathbb{E}\{\phi(\boldsymbol{\gamma}(t))\}$. This relationship is given by Theorem 1 presented in Sec. IV-D. For the property of the drift-plus-penalty defined in (11), we have the following lemma:

Lemma 1: For all possible values of $\boldsymbol{\Theta}(t)$ and under any scheduling policy $\boldsymbol{\sigma}(t) \in \mathcal{C}$, the drift-plus-penalty has the following upper bound for all slots t:

$$\begin{aligned} \Delta(\boldsymbol{\Theta}(t)) - V\mathbb{E}\{\phi(\boldsymbol{\gamma}(t)) | \boldsymbol{\Theta}(t)\} \leq D - V\mathbb{E}\{\phi(\boldsymbol{\gamma}(t)) | \boldsymbol{\Theta}(t)\} \\ + \sum_{m\in\mathcal{I}} Z_m(t)\mathbb{E}\{(p_m(t) - p_m^*) | \boldsymbol{\Theta}(t)\} \\ + \sum_{s\in\mathcal{I}} G_s(t)\mathbb{E}\{(\gamma_s(t) - \mu_s(t)) | \boldsymbol{\Theta}(t)\} \\ + \sum_{i\in\mathcal{I}} Q_i(t)\mathbb{E}\{(\lambda_i(t) - \nu_i(t)) | \boldsymbol{\Theta}(t)\} \end{aligned} \quad (12)$$

where $D \equiv \frac{1}{2}\sum_{i\in\mathcal{I}}[(\overline{\lambda}_i)^2+(\overline{\nu}_i)^2]+\frac{1}{2}\sum_{s\in\mathcal{I}}[(\overline{\gamma}_s)^2+(\overline{\mu}_s)^2]+\frac{1}{2}\sum_{m\in\mathcal{I}}[(\overline{p}_m)^2+(p_m^*)^2]$.

Proof: From the queueing dynamics (2), we have

$$\begin{aligned} Q_i(t+1)^2-Q_i(t)^2 &= (Q_i(t)-\nu_i(t)+\lambda_i(t))^2-Q_i(t)^2 \\ &\leq\ \nu_i(t)^2+\lambda_i(t)^2+2Q_i(t)(\lambda_i(t)-\nu_i(t)),\ \forall i\in\mathcal{I} \end{aligned}$$

Similarly, we have

$$\begin{aligned} Z_m(t+1)^2-Z_m(t)^2 &\leq p_m(t)^2+(p_m^*)^2 \\ &\quad +2Z_m(t)(p_m(t)-p_m^*),\ \forall m\in\mathcal{I} \\ G_s(t+1)^2-G_s(t)^2 &\leq \gamma_s(t)^2+\mu_s(t)^2 \\ &\quad +2G_s(t)(\gamma_s(t)-\mu_s(t)),\ \forall s\in\mathcal{I} \end{aligned}$$

Taking conditional expectations of the above three equations and summing over $i, m, s \in \mathcal{I}$, we get a bound on $\Delta(\boldsymbol{\Theta}(t))$. The lemma is then proved by subtracting $V\mathbb{E}\{\phi(\boldsymbol{\gamma}(t))|\boldsymbol{\Theta}(t)\}$ from both sides. ■

C. Proposed BITS Algorithm

The BITS algorithm seeks to minimize the upper bound of (12) instead of directly minimizing the drift-plus-penalty itself. As shown in Sec. IV-D, this does not affect the optimality of the solution. The algorithm works as follows:

1) Auxiliary variables: Based on $\boldsymbol{G}(t)$, choose $\boldsymbol{\gamma}(t)$ in each slot t such that the following function is maximized

$$\max_{\boldsymbol{\gamma}(t)}\quad V\sum_{s\in\mathcal{I}}\phi(\gamma_s(t))-\sum_{s\in\mathcal{I}}G_s(t)\gamma_s(t) \tag{13}$$

$$\text{s.t.}\quad 0\leq\gamma_s(t)\leq\gamma_s^{max},\ \forall s\in\mathcal{I} \tag{14}$$

Since the auxiliary variables $\boldsymbol{\gamma}(t)$ are independent, the above maximization can be decoupled as maximizing $V\phi(\gamma_s(t)) - G_s(t)\gamma_s(t), \forall s \in \mathcal{I}$, subject to $0 \leq \gamma_s(t) \leq \gamma_s^{max}$. The peak value of this objective function is obtained when $\gamma_s(t) = V/G_s(t)$ for $G_s(t) > 0$. Therefore, by taking into account the constraint of $\gamma_s(t)$ in (14), we have the following optimal solution to the above problem:

$$\gamma_s(t)=\begin{cases}\frac{V}{G_s(t)}, & G_s(t)\geq\frac{V}{\gamma_s^{max}}\\ \gamma_s^{max}, & G_s(t)<\frac{V}{\gamma_s^{max}}\end{cases}$$

The value of $G_s(t)$ directly affects the value of $\gamma_s(t)$. If the value of $G_s(t)$ is small, this implies that the time average of $\gamma_s(t)$ is very close to that of $\mu_s(t)$, which enforces the stability of virtual queue $G_s(t)$. If the value of $G_s(t)$ is large, then a small $\gamma_s(t)$ should be chosen to enforce queue stability. The complexity of (13) is $O(N)$.

2) Scheduling policy: Based on $\boldsymbol{Q}(t)$, $\boldsymbol{d}(t)$, $\boldsymbol{G}(t)$, $\boldsymbol{Z}(t)$ and $\boldsymbol{c}(t)$, choose $\boldsymbol{\sigma}(t) \in \mathcal{C}$ in each slot t to minimize

$$\begin{aligned}\sum_{m\in\mathcal{I}}Z_m(t)p_m(\boldsymbol{\sigma}(t))-\sum_{i\in\mathcal{I}}Q_i(t)\nu_i(\boldsymbol{\sigma}(t))\\ -\sum_{s\in\mathcal{I}}G_s(t)\mu_s(\boldsymbol{\sigma}(t))\end{aligned} \tag{15}$$

The $Z_m(t), Q_i(t)$ and $G_s(t)$ in (15) can be interpreted as the weights of re-routing cost, service rate and delay-sensitive throughput, respectively. Under a large energy budget p_i^*, the value of $Z_m(t)$ is always equal to zero according to (9). Thus, in each slot the scheduling policy is to balance the queues $\boldsymbol{Q}(t)$ and $\boldsymbol{G}(t)$ among users. Under a small energy budget, the value of $Z_m(t)$ is no longer always equal to zero. The scheduling policy starts to trade off between the opportunistic gain and the re-routing energy cost. The complexity of this policy is $O(N^2)$ because it chooses $\boldsymbol{\sigma}(t) \in \mathcal{C}$ to minimize (15) in each slot and there are N^2 different $\boldsymbol{\sigma}(t)$.

3) Queue updates: Update the virtual queues $\boldsymbol{Z}(t)$ and $\boldsymbol{G}(t)$ according to (9) and (10) and the $\boldsymbol{\gamma}(t), \boldsymbol{p}(t)$ determined from previous two steps. The queue $\boldsymbol{Q}(t)$ is updated according to (2) and the scheduling decision $\boldsymbol{\sigma}(t)$.

D. Optimality analysis

To prove the optimality of BITS, we show that the difference between the utility under BITS and the optimal utility can be made arbitrarily small:

Theorem 1: Assume initially all the queues $\boldsymbol{Q}(t)$, $\boldsymbol{Z}(t)$ and $\boldsymbol{G}(t)$ are empty. For a particular constant $V > 0$, the achieved time-average utility under the BITS algorithm satisfies:

$$\lim_{t\to\infty}\inf\sum_{i\in\mathcal{I}}\phi(\overline{\mu}_i)\geq\phi^*-\frac{D}{V} \tag{16}$$

where ϕ^* is the maximal achievable utility under all possible scheduling policies $\boldsymbol{\sigma}(t) \in \mathcal{C}$.

Theorem 1 indicates that by increasing V, the utility under BITS can be made arbitrary close to the optimal utility. Note that a large V also results in large average backlogs of the virtual queues $\boldsymbol{Q}(t)$ and $\boldsymbol{Z}(t)$. The theorem can be proved by first proving the existence of a stationary queue-state-unaware scheduling decision, followed by inserting the decision into (12) to remove the dependence on $\boldsymbol{\Theta}(t)$ in the expectations. After that, the theorem can be proved by applying iteration over time slots, using the property of Jensen's inequality and then by taking the limit (similar to the proof the Theorem 5.1 in [21]).

V. Property of the BITS Scheduling Policy

In this section, we study the properties of BITS in a two-user two-channel-state system. The channel is a Markovian channel with *on* and *off* states. We consider a homogeneous scenario where the transition probabilities of the BS-user link are $p^i_{\mathit{on2on}} = p^i_{\mathit{off2off}} = 0.8;\ p^i_{\mathit{on2off}} = p^i_{\mathit{off2on}} = 0.2, \forall i \in \mathcal{I}$. The packet size is fixed to 1. We assume one packet can be sent in a slot (1ms) if the channel is *on* and zero packets is sent if the channel is *off*. The arrivals are according to a Poisson process with average arrival rates $\boldsymbol{\lambda} = \{0.3, 0.3\}$ packets/ms. For each user i, the delay threshold vector $\boldsymbol{d}_i^* = \{0, 20, +\infty\}$ms and the weight $\boldsymbol{w}_i = \{1, 0\}$. The additional energy cost $p_i(t)$ is 1 mJ for re-routing one packet, $\forall i \in \mathcal{I}$. For simplicity, here we assume that the user-user link has no forwarding delay. (*In Section VI, however, forwarding delay is considered.*)

Let h_i denote the queueing delay of head-of-line packet of $Q_i, \forall i \in \mathcal{I}$. Further, we define the *combined queue* as follows:

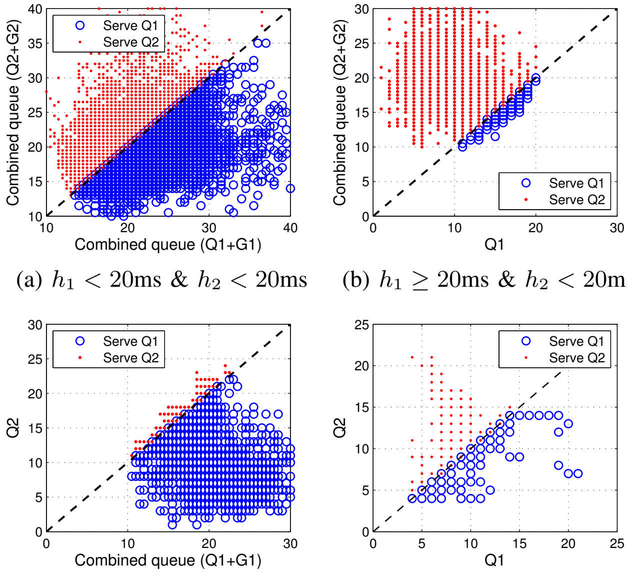
$$U_i\text{'s combined queue}=\begin{cases}Q_i+G_i, & h_i<d_i^*\\ Q_i, & \text{otherwise}\end{cases} \tag{17}$$

We show the scheduling decisions of BITS in Fig. 2 and 3. The axes represent the lengths of combined queues and the figures depict scheduling decisions at each queue state (except for Fig. 3(d)). To help distinguish between different scheduling decisions, we draw the diagonal where the lengths of combined queues are equal. We observe the following properties:

Instantaneous gain vs. queue balancing: BITS can exploit instantaneous gain and/or balance queues based on current channel states, packet delays, actual and virtual queue states. Exploiting instantaneous gain is achieved by sending packets that contribute the most to the utility, while balancing queues aims to reduce the future number of packets that have no contribution to the utility. We present this property in detail by considering different combinations of channels states.

1) Both channels are on: There is no re-routing under this case, i.e., BITS always serves users with their own packets. Thus there are two scheduling decisions, i.e., to serve Q_1 or Q_2, as shown in Fig. 2. We observe that the scheduling policies are separated by the diagonal, which implies that BITS always balances the combined queues. This can be derived from (15) where $p_l(\boldsymbol{\sigma}(t))$ is always equal to zero, $\forall l \in \mathcal{I}$. We can further observe for different values of h_1 and h_2:

- $h_1, h_2 < 20\text{ms}$: as shown in Fig. 2 (a), BITS balances the combined queues, aiming to keep in the future as many packets that contribute to the utility as possible.
- $h_1 \geq 20\text{ms}, h_2 < 20\text{ms}$: The head-of-line packet of Q_1 under this case does not contribute to the utility, thus from (17) we know U_1's combined queue is Q_1. Again, BITS algorithm balances the combined queues, i.e., Q_1 and $Q_2 + G_2$, as shown in Fig. 2 (b). This implies even Q_2 is smaller than Q_1, the policy may still serve U_2 to exploit instantaneous gain (note that $G_2 \geq 0$).
- $h_1 < 20\text{ms}$ & $h_2 \geq 20\text{ms}$: similar to the previous case.
- $h_1, h_2 \geq 20\text{ms}$: The head-of-line packets of both users do not contribute to the utility, thus BITS only balances the actual queues Q_1 and Q_2, as shown in Fig. 2 (d).

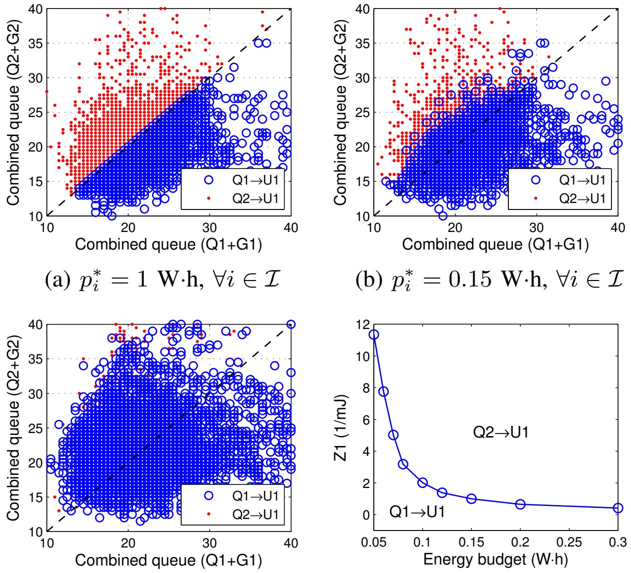


(a) $h_1 < 20$ms & $h_2 < 20$ms (b) $h_1 \geq 20$ms & $h_2 < 20$ms

(c) $h_1 < 20$ms & $h_2 \geq 20$ms (d) $h_1 \geq 20$ms & $h_2 \geq 20$ms

Fig. 2: Properties of BITS: both users' channels are *on*.

2) One user's channel is on, the other's is off: Without loss of generality, let us focus on the case where U_1's channel is *on* and U_2's channel is *off*. Note that there is re-routing from Q_2 to U_1 if the energy budget permits. We depict the scheduling decisions in Fig. 3 where we show the decisions when both h_1 and h_2 are smaller than 20ms. Scheduling decisions under other values of h_1 and h_2 are similar. We observe that under large energy budget (e.g., $p_i^* = 1\text{W}\cdot\text{h}$ (over a second),$\forall i \in \mathcal{I}$), BITS still balances the combined queues. However, when the energy budget decreases, BITS reacts less to the imbalance in combined queues. We explain this in depth through another property of BITS.



(a) $p_i^* = 1$ W·h, $\forall i \in \mathcal{I}$ (b) $p_i^* = 0.15$ W·h, $\forall i \in \mathcal{I}$

(c) $p_i^* = 0.05$ W·h, $\forall i \in \mathcal{I}$ (d) Different energy budget

Fig. 3: Properties of BITS: U_1's channel is *on*, U_2's channel is *off*, $h_1 < 20$ms and $h_2 < 20$ms

Performance vs. energy consumption: The tradeoff between performance and energy consumption can be seen clearly from Fig. 3. We depict the scheduling decisions when both h_1 and h_2 are smaller than 20ms, U_1's channel is *on* and U_2's channel is *off*. If BITS chooses to serve Q_2, there will be re-routing from Q_2 to U_1. Given a large energy budget, e.g., $p_i^* = 1\text{W}\cdot\text{h}$, BITS behaves exactly the same as in Fig. 2 (a), i.e., balancing combined queues. The re-routing area ($Q_2 \to U_1$) diminishes progressively with the decrease of energy budget, as shown in Fig. 3 (a)-(c). This implies that performance is sacrificed in order to reduce the energy cost. When the energy budget is very small, re-routing only occurs when the imbalance between users' combined queues is very large. Furthermore, from (15) we can derive that if the difference between the combined queues of U_2 and U_1 at time slot t is larger than the length of virtual queue $Z_1(t)$, BITS will choose to send packets from Q_2. Based on this derivation, we depict the scheduling decisions in Fig. 3 (d) where the axes are energy budget and length of Z_1, respectively. In this figure, the scheduling decision above the curve is re-routing from Q_2 to U_1, and below the curve is no re-routing. We observe clearly that the re-routing degree increases with the increase of energy budget, which shows the tradeoff between performance and energy consumption. Another very interesting observation is that the chances of re-routing from Q_2 to U_1 can be increased greatly with even a small increase in energy budget.

VI. Performance Evaluation

To evaluate BITS, we compare it with following policies:

1) *Queue-unaware, proportionally fair scheduling (PF)*: At any slot t, a PF scheduler chooses to serve the user i with the maximum $R_i(t)/R'_i(t)$, where $R_i(t)$ is U_i's instantaneous data rate and $R'_i(t)$ is the exponentially smoothed average service rate of U_i [6].
2) *Queue-aware, log rule scheduling*: Queue-aware means the scheduler is aware of the queue length. At time t, a log rule scheduler makes decisions based on current channel state and the logarithm of queue length [9].
3) *Queue-aware, maximal re-routing (MaxRR)*: At time t, a MaxRR scheduler chooses to serve user i that has the largest instantaneous data rate with packets from the longest queue. If the packets do not belong to user i, they will be forwarded by user i to the corresponding user through user-user link.

Note that among the above scheduling policies, re-routing only occurs under MaxRR.

A. Simulation setup

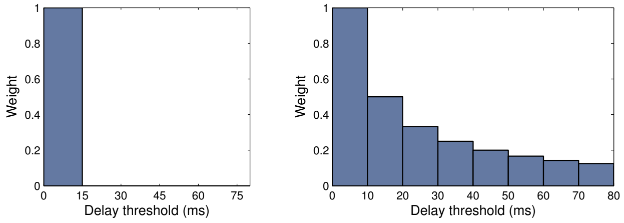
BS-user link: Time is slotted and each slot lasts for 1ms. The BS-user channel is a Rayleigh fading channel where the Signal-to-Noise-Ratio (SNR) is assumed to be constant during each slot. Other channel settings are listed in Table I (note that the specific choice of parameters has no significant impact on the fundamental tradeoff between performance improvement and re-routing energy consumption). We adopt modulation scheme to SNR according to the mechanism specified in [22]. The setting of the parameters in the PF and log rule scheduling policies are taken from [6] and [9], respectively.

User-user link: The user-user channels are also Rayleigh fading channels where the path loss exponent and Doppler shift are set according to Table I. The bandwidth is 20MHz. The durations of aSlotTime, SIFS and DIFS are 9μs, 10μs and 28μs, respectively. The minimal and maximal contention window are set to 15 and 1023, respectively. The RTS/CTS mechanism is disabled. The lengths of PHY header, MAC header and ACK are 192bits, 256bits and 304bits. The setting of data rate with respect to SNR is according to [23].

TABLE I: Channel parameters of BS-user link

Parameters	Value
Bandwidth	5 MHz
BS Tx power	0.1/5 W/MHz
Noise spectral density	10^{-8}/5 W/MHz
Path loss exponent (Urban Area)	3
Doppler shift (ITU Pedestrian A)	5 Hz

Two different types of delay thresholds (Type I and Type II) are evaluated. In Type I, the thresholds and corresponding weights are shown in Fig. 4 (a). Type I can represent a type of applications where packets are dropped if their delays exceed certain value, e.g., live video streaming. The relationship between thresholds and the corresponding weights of Type II is shown in Fig. 4 (b). Type II can represent a type of interactive applications, e.g., gaming.



(a) Type I

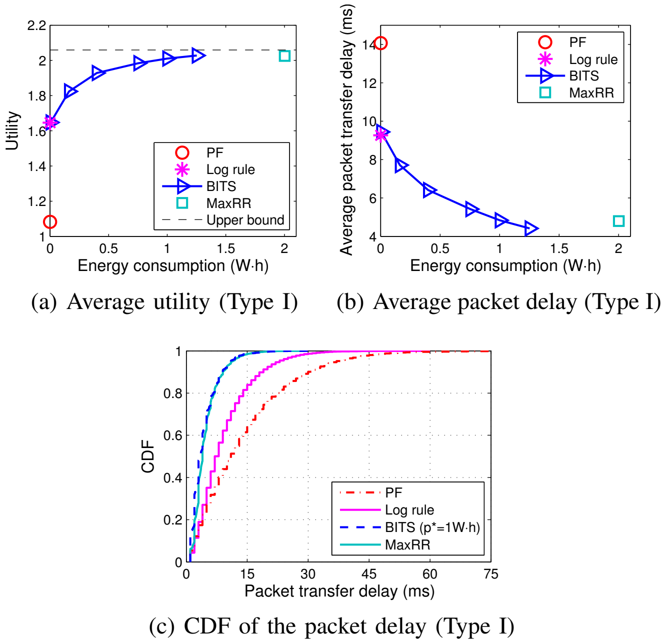
(b) Type II

Fig. 4: Delay thresholds and the corresponding weights.

Without loss of generality, the packet size is fixed to 175 bytes unless otherwise specified. We assume the additional energy cost $p_i(t)$ for re-routing a packet is 1 mJ, $\forall i \in \mathcal{I}$. The constant V in (11) is set to 100. We assume during each slot, $M(t) \in \mathbb{Z}_+$ packets are transmitted, depending on the value of instantaneous SNR. In the simulation results, the values of energy budget and energy consumption of users are given within a second (i.e., power).

B. The two-user scenarios

Homogeneous scenarios: The simulation results are shown in Fig. 5 (due to space limitation, the results under the Type II threshold are not shown). The x-axes are energy consumption while the y-axes are utility and average packet transfer delay, respectively. The tradeoff between performance and energy budget under BITS can be clearly seen in both Fig. 5 (a) and (b). Note that the upper bound in Fig. 5 (a) is obtained by assuming all the packets are transmitted without delay. Under large energy budgets, the performance of BITS can be as good as that under maximal re-routing. Compared to PF, the utility can be increased by 1 under both BITS and MaxRR. The average packet transfer delay with BITS can be reduced by up to 72%. To provide further insights into these results, we plot in Fig. 5 (c) the Cumulative Distribution Function (CDF) of packet delay. We observe that under BITS (with a large energy budget) and MaxRR, almost all packets are served with a delay below 15ms, while under the log rule and PF, around 20% and 40% of the packets have delays exceeding 15ms. Under small energy budget, the re-routing energy consumption decreases rapidly along with the decrease of utility and the increase of average packet transfer delay, as shown in Fig. 5 (a) and (b). Further, we observe that BITS is able to achieve the same performance as MaxRR while consuming less energy (65%). This is because BITS can optimize the degree of re-routing under a specified energy budget. An another interesting observation is that most of the performance gain under BITS can be achieved at small increase in energy consumption, e.g., 70% of the utility gain can be achieved at only 30% of the maximal energy consumption under BITS.



(a) Average utility (Type I)

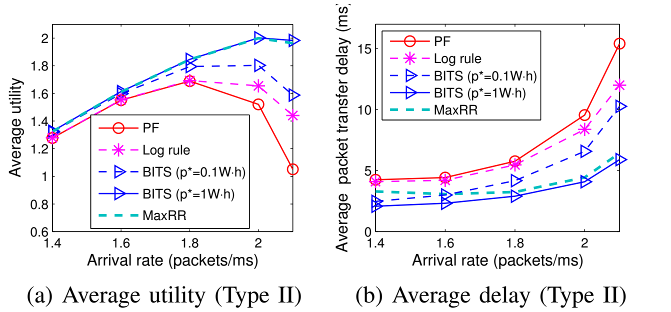
(b) Average packet delay (Type I)

(c) CDF of the packet delay (Type I)

Fig. 5: Two-user homogeneous scenario: the average $\mathbf{SNR}^{\text{bs-user}} = \{9,9\}$dB, $\text{SNR}_{ij}^{\text{user-user}} = 9$dB, $\boldsymbol{\lambda} = \{2,2\}$packets/ms, $i \neq j$.

The impact of arrival rates on the utility and delay performance under Type II threshold is shown in Fig. 6, where we scale $\boldsymbol{\lambda}$ while keeping other parameters unchanged. As expected, the average transfer delay increases with the increase of arrival rates, as indicated in Fig. 6 (b). We also observe that when the arrival rates are low (e.g., 1.4 packets/ms), the utilities of all the policies are almost the same. This is because almost all the packets can be served with low queueing delay at the BS. As the packet arrival rates increase, packet queueing delays increase and more and more packets are going to exceed the delay threshold. This is why the utilities of all the scheduling policies first increase and then decrease with the increase of packet arrival rate. However, since BITS and MaxRR exploit the local user-user communication to spread traffic, the utilities under these policies decrease much slower than those under the log rule and PF. Note that for a large energy budget, the performance of BITS is even better than MaxRR while consuming less energy for re-routing. This is because BITS is aware of the packet delays, and thus can make better scheduling decisions than MaxRR.

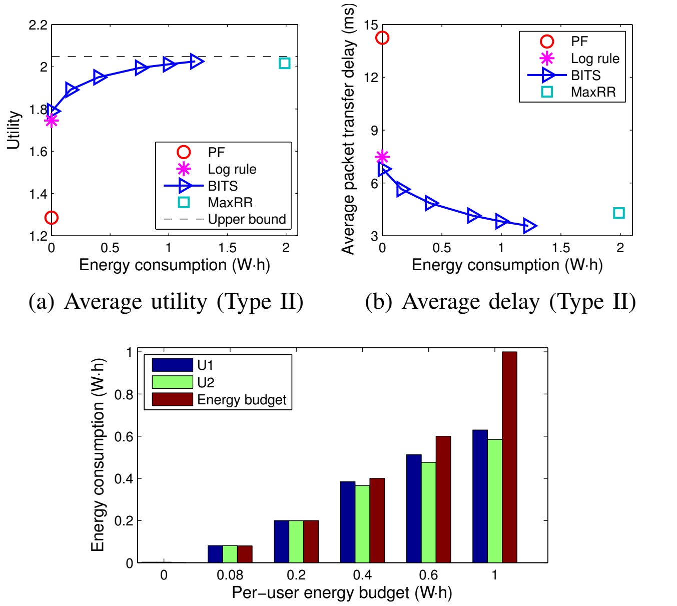
Heterogeneous scenarios: Fig. 7 depicts the tradeoff between performance and energy consumption achieved by BITS under a scenario where one user has a lower traffic load as well as a worse average BS-user channel quality. The upper bound also comes from the assumption as in the homogeneous scenarios that all the packets are transmitted without delay. The maximal utility and delay improvements under BITS are 0.7 and 75%, respectively, compared to PF. When the energy budget is large, the delay performance of BITS is even better than MaxRR, while requiring only 65% of the energy consumption. This is because BITS introduces less additional delay than MaxRR, where the delay comes from packet forwarding among users. Thus, in Fig. 7 (b) we can observe that the average packet transfer delay under BITS (with a large energy budget) is lower than that under MaxRR. Similar to the homogeneous scenario, the re-routing energy consumption reduces rapidly as the energy budget decreases, while the performance decreases slowly.



(a) Average utility (Type II)

(b) Average delay (Type II)

Fig. 6: Two-user homogeneous scenarios with different arrival rates: the average $\mathbf{SNR}^{\text{bs-user}} = \{9,9\}$dB, $\text{SNR}_{ij}^{\text{uses-user}} = 9$dB, $i \neq j$.



(a) Average utility (Type II)

(b) Average delay (Type II)

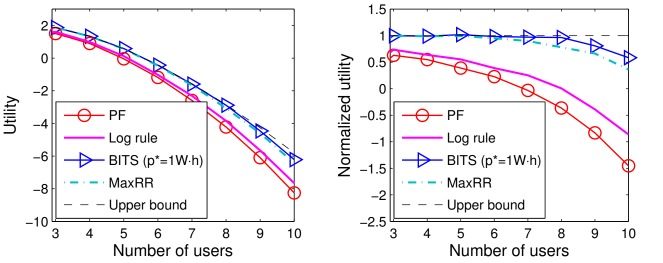
(c) Per-user energy consumption under BITS (Type II)

Fig. 7: Two-user heterogeneous scenario: the average $\mathbf{SNR}^{\text{bs-user}} = \{10,9\}$dB, $\text{SNR}_{ij}^{\text{uses-user}} = 9$dB, $\boldsymbol{\lambda} = \{2.2, 1.8\}$packets/ms.

The overall re-routing energy consumption as well as the split between users under BITS is shown in Fig. 7 (c). We observe that both of the two users contribute to the performance improvement, i.e., even the user with a worse channel (U_2) forwards packets (thus consuming energy) to the user that has a good channel (U_1). Another observation is that the re-routing energy consumption is always constrained by the energy budget, as expected from the theoretical analysis. Further, when energy budget is below 0.4W·h, the energy consumptions of the two users are the same, both are equal to the budget. This implies that BITS can balance the energy consumption among users when given the same energy budget for the users. We also observe that when the energy budget is equal to or above 0.4W·h, the re-routing energy consumption of U_1 is slightly higher than that of U_2, while both of them are below the energy budget. This is because the average channel quality of U_1 is better than that of U_2, so U_1 has more opportunities to forward packets for U_2. Since the energy budget is large, BITS will exploit as much as possible these opportunities to maximize the performance improvement.

C. Multi-user scenarios

Homogeneous scenarios (performance scaling with number of users): The scaling of the utility with the number of users is shown in Fig. 8. Here, all users have the same traffic load, and the sum of arrival rates across all users is fixed to 4 packets/ms. The average SNRs of the BS-user and user-user channels are the same. All users can communicate with each other and each user has a sufficiently large energy budget. An interesting observation from Fig. 8 (a) is that the utility decreases when the number of users increases. Another observation in Fig. 8 (a) is that the difference between utilities under BITS and PF increases with an increasing number of users. The reason behind this is as the user population increases, the BS have more and more choices to spread traffic among users. This can be seen more clearly from Fig. 8 (b) where we fix the upper bound of the utility to 1 for all numbers of users and depict the differences between the upper bound and utilities under all the scheduling algorithms.

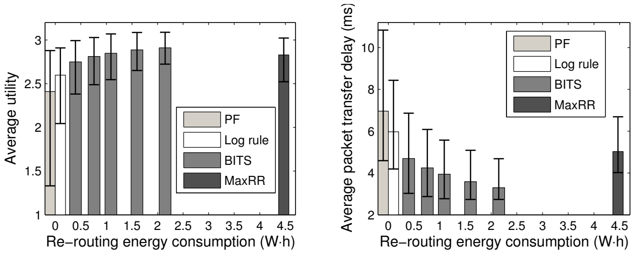


(a) Utility (Type II) (b) Normalized utility (Type II)

Fig. 8: Homogeneous scenarios with different number of users: the average $\mathbf{SNR}^{\text{bs-user}} = \{9, 9\}$dB, $\text{SNR}_{ij}^{\text{uses-user}} = 9$dB, $i \neq j$.

Heterogeneous scenarios (users randomly distributed in a cell): We present results where users are served by a BS with a service range of 100m. We consider instances with four users, all of them randomly distributed at distances ranging from 50m to 100m to the BS (corresponding to average SNRs between 19dB and 10dB). Note that if the channel between two users is very poor, the scheduler will not spread traffic among them. Users' arrival rates are also heterogeneous and the arrival rate for each user is chosen randomly in a range of $1.25 \pm 10\%$ packets/ms. We evaluate 100 instances for each chosen re-routing energy budget. The average results as well as the 95^{th} and 5^{th} percentile of the utility and average packet transfer delay are shown in Fig. 9.

We observe that under BITS, average utility can be improved by 0.4 and the average packet transfer delay can be reduced by up to 50%, compared to PF. Even compared to MaxRR, the delay under BITS can be improved by up to 35% while having only half of the energy consumption. This is due to the smart control of traffic spreading of BITS as well as the large difference of channel conditions among users. Further, from the 5^{th} percentile of utility in Fig. 9 (a) we observe that the utility can be increased by 1.3, and from the 95^{th} percentile of delay in Fig. 9 (b) that the average packet transfer delay can be be reduced by around 55%. This demonstrate that BITS is capable of significantly improving user performance in instances where the overall system performance is poor under PF (used in current 3G and 4G systems), which is a very important practical consideration.



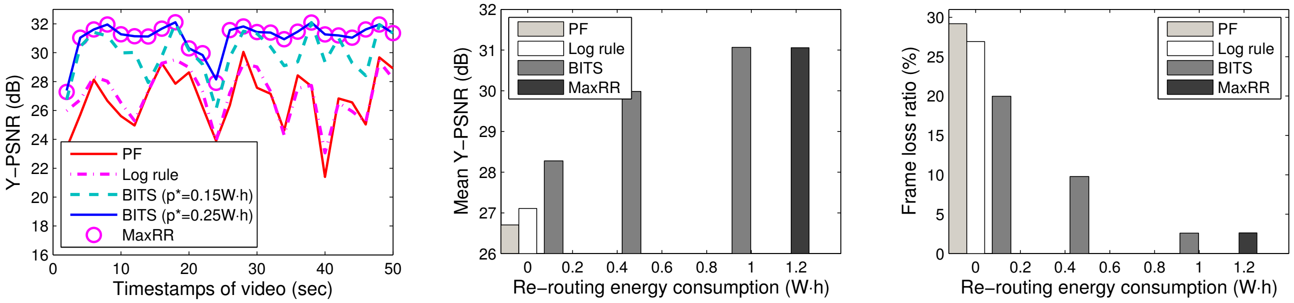
(a) Average utility (Type II) (b) Average delay (Type II)

Fig. 9: Performance of BITS under four-user heterogeneous scenarios where $\lambda_i = 1.25 \pm 10\%$ packets/ms, $i \in \mathcal{I}$.

D. Simulation of live video streaming

Finally, we evaluate BITS using realistic traces of video traffic. We consider a four user homogeneous scenario, where the average $\mathbf{SNR}^{\text{bs-user}} = \{6, 6, 6, 6\}$dB and $\text{SNR}_{ij}^{\text{user-user}} = 9$dB, $\forall i, j \in \mathcal{I}$. The other wireless settings are the same as in the previous subsections. The video stream is a 50-second soccer game (4CIF YUV video with 30 frames per second) used in [24, 25]. The source YUV video is encoded into single layer H.264 Advanced Video Coding (AVC) format through the Joint Scalable Video Model (JSVM) software [25]. The encoded video is then extracted and we use its trace as the input to our simulation. We reproduce the H.264 AVC video based on the received trace, convert it to YUV format and calculate its Peak Signal-to-Noise Ratio (PSNR) with respect to the source of YUV video. Note that in the video reproduction, if the delay of a frame exceeds 200ms, we consider it lost and substitute it with the previous frame. The metrics we use are the Y-PSNR of each frame and the frame loss ratio of the received video.

Results are shown in Fig. 10. The Y-PSNR in Fig. 10 (a) is averaged across all users over two second intervals. Fig. 10 (a) shows that under a large energy budget, BITS can achieve performance as good as that of maximal re-routing, both of which are not sensitive to channel fluctuations most of the time (except at the beginning and several seconds around 25s where the channels between all users and the BS are bad). In contrast, the received PSNR under PF reacts greatly to the channel fluctuations, which affects the quality of experience of the users. Furthermore, we observe that with a lower energy



(a) Y-PSNR versus time.

(b) Y-PSNR vs. energy consumption

(c) Frame loss ratio vs. energy consumption

Fig. 10: Performance of the BITS policy in the application of live video streaming.

budget, BITS still performs well and its PSNR changes much more slowly than that under PF.

As for the average Y-PSNR, Fig. 10 (b) shows that it can be improved by up to 4dB under the BITS algorithm. Similar to previous scenarios, the gain decreases with, but not as fast as, the decrease of energy budget. The frame loss ratio can be reduced greatly under BITS as shown in Fig. 10 (c), e.g., up to 90% when compared to both PF and the log rule.

VII. Conclusion and Future Work

In this paper, we presented a BS-driven traffic spreading policy (BITS) to increase downlink user performance in small cells, by exploiting both BS-user and user-user communication. We formulated the problem to maximize delay-sensitive utility under a re-routing energy budget, and solved it through stochastic Lyapunov optimization. We designed the BITS algorithm, studied its properties and then evaluated it in a range of scenarios. We found that BITS can greatly increase the utility and reduce the average packet transfer delay, as well as balance the additional energy consumption for traffic spreading among users. Finally, we evaluated BITS with a realistic video trace, and showed that it can increase the average Y-PSNR of the received video and reduce the frame loss ratio significantly. For future work, we plan to extend our work by considering in-band user-user communication [26].

References

[1] P. Bender and etc., “CDMA/HDR: a bandwidth efficient high speed wireless data service for nomadic users,” *IEEE Communications Magazine*, vol. 38, no. 7, pp. 70–77, Jul. 2000.

[2] X. Liu, E. K. P. Chong, and N. B. Shroff, “A framework for opportunistic scheduling in wireless networks,” *Computer Networks*, vol. 41, pp. 451–474, 2003.

[3] 3GPP2, “CDMA2000 high rate packet data air interface specication,” *C.S20024-A v. 1.0*, 2004.

[4] ——, “Dual-cell high speed downlink packet access (HSDPA) operation,” *TR 25.825 v.1.0.0*, 2008.

[5] S. Borst, “User-level performance of channel-aware scheduling algorithms in wireless data networks,” *IEEE/ACM Transaction on Networking*, vol. 13, no. 3, pp. 636–647, Jun. 2005.

[6] M. Andrews, “Instability of the proportional fair scheduling algorithm for HDR,” *IEEE Transactions on Wireless Communications*, vol. 3, no. 5, pp. 1422–1426, 2004.

[7] M. Andrews, K. Kumaran, K. Ramanan, and etc., “Scheduling in a queuing system with asynchronously varying service rates,” *Probab. Eng. Inf. Sci.*, vol. 18, no. 2, pp. 191–217, Apr. 2004.

[8] S. Shakkottai and A. L. Stolyar, “Scheduling for multiple flows sharing a time-varying channel: The exponential rule,” *American Mathematical Society Translations*, vol. 2, 2000.

[9] B. Sadiq, S. J. Baek, and G. de Veciana, “Delay-optimal opportunistic scheduling and approximations: the log rule,” in *IEEE INFOCOM*, 2009.

[10] H. Zhou, P. Fan, and H.-C. Yang, “Cross-layer scheduling for multiuser downlink transmissions with opportunistic relaying,” in *ICCCN*, 2009.

[11] *Cisco visual networking index: Global mobile data traffic forecast update, 20102015.* Cisco whitepaper, 2011.

[12] *Femtocell Market Status.* Femtoforum whitepaper, 2011.

[13] J. Andrews, H. Claussen, M. Dohler, S. Rangan, and M. Reed, “Femtocells: Past, present, and future,” *IEEE JSAC*, vol. 30, no. 3, pp. 497–508, 2012.

[14] H. Falaki, D. Lymberopoulos, R. Mahajan, and et al., “A first look at traffic on smartphones,” in *ACM IMC*, 2010.

[15] P. Liu, R. Berry, and M. Honig, “Delay-sensitive packet scheduling in wireless networks,” in *IEEE WCNC*, 2003.

[16] G. Song, Y. Li, L. Cimini, and H. Zheng, “Joint channel-aware and queue-aware data scheduling in multiple shared wireless channels,” in *IEEE WCNC*, 2004.

[17] Q. Wang and B. Rengarajan, “Recouping opportunistic gain in dense base station layouts through energy-aware user cooperation,” in *IEEE WoWMoM*, 2013.

[18] A. Bletsas, A. Khisti, D. Reed, and A. Lippman, “A simple cooperative diversity method based on network path selection,” *IEEE JSAC*, vol. 24, no. 3, pp. 659–672, 2006.

[19] S. Song, K. Son, H.-W. Lee, and S. Chong, “Opportunistic relaying in cellular network for capacity and fairness improvement,” in *IEEE GLOBECOM*, 2007.

[20] R. Ganti and M. Haenggi, “Spatial analysis of opportunistic downlink relaying in a two-hop cellular system,” *IEEE Transactions on Communications*, vol. 60, no. 5, pp. 1443–1450, 2012.

[21] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems.* Morgan & Claypool, 2010.

[22] S. Sesia, I. Toufik, and M. Baker, *LTE: the UMTS long term evolution.* Wiley, 2011.

[23] T.-S. Kim, H. Lim, and J. C. Hou, “Improving spatial reuse through tuning transmit power, carrier sense threshold, and data rate in multihop wireless networks,” in *ACM Mobicom*, 2006.

[24] A. Detti, G. Bianchi, C. Pisa, F. Proto, P. Loreti, W. Kellerer, S. Thakolsri, and J. Widmer, “SVEF: an open-source experimental evaluation framework,” in *IEEE MediaWIN*, 2009.

[25] JSVM. http://www.hhi.fraunhofer.de/de/kompetenzfelder/image-processing/research-groups/image-video-coding/svc-extension-of-h264avc/jsvm-reference-software.html.

[26] A. Asadi, Q. Wang, and V. Mancuso, “A survey on device-to-device communication in cellular networks,” 2013, arxiv.org/abs/1310.0720.