IBM Research Report

Macro-Scheduling of Base Stations for Video-on-Demand Flows in WiMAX Networks

Shubhadip Mitra
UmaMaheswari Devi
Parul Gupta
Partha Dutta
Malolan Chetlur
Shivkumar Kalyanaraman

IBM India Research Laboratory
Bangalore
India

IBM Research Division
Almaden - Austin - Beijing - Delhi - Haifa - T.J. Watson - Tokyo - Zurich

***Abstract*—Wireless broadband access network technologies, such as IEEE 802.16e WiMAX, by enabling high bandwidth applications for mobile users, necessitate mechanisms for efficiently utilizing the limited wireless resources and meeting growing user expectations. In this context, in a WiMAX network, we consider efficient utilization of base station (BS) radio resources and lifetime quality-of-experience (QoE) management of video-on-demand (VoD) users. For efficient resource utilization, we propose dynamic load balancing through coordinated scheduling among multiple BSs. To ensure that all flows of a single user are assigned to the same BS, our approach to coordinated scheduling is hierarchical. In this approach, allocations to users are determined initially (first level), followed by a distribution of users' allocations among their flows (second level). During both stages of hierarchical scheduling, a utility-maximization based approach is used. To achieve long-term *proportional fairness* (PF) and manage lifetime QoE, user and flow utilities are modeled as functions of past service rates, in addition to their current channel conditions and bandwidth needs.**

PF scheduling of users or flows across multiple BSs is known to be NP-hard. In this paper, we show that the problem is in fact NP-hard in the strong sense. We hence consider approximation algorithms proposed in prior work and design efficient heuristics for first-level scheduling. For the second level, we consider the single-BS allocation problem, and show that, despite *integrality constraints*, it can be solved *optimally* in quadratic time. Finally, we show that the hierarchical approach that we propose is optimal. The efficacy of our approaches in improving QoE and the number of satisfied users in WiMAX networks is evaluated through simulations.

I. Introduction

The IEEE 802.16e WiMAX technology promises high broadband data rates over large coverage areas (couple of miles) in mobile, wireless contexts [1]. WiMAX is also envisioned for enterprise deployments in fixed, pedestrian, and nomadic scenarios for improving coverage significantly, and supporting wide range of high data-rate applications such as mobile workforce support, video-based training, *etc.* [2].

Despite the promise of high data rates, the inherently varying and unreliable nature of the wireless channel poses significant challenges to providing guaranteed services and ensuring quality-of-experience (QoE) for users over WiMAX. In this paper, we propose a solution towards managing the lifetime QoE of video-on-demand (VoD) users in WiMAX networks. Ensuring QoE over the lifetime of a session entails managing the service across multiple BSs (due to mobility) and channel variations, and resource allocation in the presence of competing flows. This requires mechanisms based on long-term session characteristics.

The state-of-the-art resource allocation techniques in WiMAX consider single base stations (BSs) only. In general, a user, that is, subscriber station (SS), chooses its serving BS, and the scheduler at the BS performs admission control and resource allocation among the admitted SSs. This approach suffers from two limitations, namely, **(1)** SS assignments are based merely on the knowledge of signal strengths that is local to BSs and SSs, and **(2)** SS reassignments are attempted only when there is a change in the relative qualities of the channels of an SS to its neighboring BSs or when hand-offs become inevitable. Due to these limitations, the existing approaches are unlikely to efficiently utilize the network's aggregated resources, and can lead to local congestions. These ill-effects would be more pronounced when the load distribution is asymmetric.

In this paper, we address the above two limitations, while alleviating congestions, improving fairness, and enhancing user QoE. This is achieved by taking a global view of the network and *jointly* scheduling clusters of BSs for transmissions to their SSs. Regardless of changes in channel quality, scheduling is performed periodically over *scheduling epochs*. We refer to this periodic, joint scheduling technique as *macro-scheduling of base stations* (MSBS). VoD flows can use playout buffers at the SSs to store video frames before they are played and can be assumed to have backlogged queues. Therefore, to ensure long-term fairness and QoE and to improve resource utilization (even) in low-mobility scenarios, we consider assignments based on past rates of flows, in addition to their bandwidth needs and channel conditions.

One way of achieving our objective is to associate each flow with a utility function, and determine during each scheduling epoch, BS-SS assignments and allocations that maximize the sum of flow utilities. If a flow's utility function is given by a weighted logarithm of the allocation it receives, then the approach is known to yield *proportional fairness* (PF) [3]. PF scheduling across multiple access points (APs) has previously been considered by Li *et al.* in [4] in the context of multi-rate wireless LANS for improving user fairness. In our work, each flow is associated with a logarithmic utility function that depends on its past service rate in addition to current channel conditions and current allocation. The work of Li *et al.* differs from ours in the following: **(1)** Li *et al.* do not consider a flow's past service rate while modeling its utility function, and as such, their approach may not utilize the resources of a network to their fullest, **(2)** users with multiple flows are not addressed, and **(3)** allocation within a single BS under integrality constraints is not dealt with (this is important in OFDMA-based scheduling of slots).

Motivating example: To see how taking past service into account can help, consider the simple network in Fig. 1. If the assignments here are based on signal strengths only, then SSs A and B (with one flow each) would both be assigned to the first BS and C to the second. With an achievable bit rate of 15 bps for A and B each, this assignment can meet the needs of neither A nor B, and under-utilizes BS2. On the other hand, if the two BSs are jointly considered and either A or B is assigned to BS2 (instead of BS1), then the needs of at least one of them would be met (in addition to that of C). Further improvement is possible by noting that serving one of the flows, say A, exclusively from BS1 would lead to A's buffer building up.

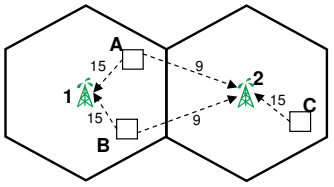

Fig. 1: An example illustrating a need for MSBS.

For instance, if A is served by BS1 in the first epoch (taken as one second, for simplicity), and B and C from BS2, then A would have a surplus of five bits, while B, a deficit of 2.5, at the end of the epoch. A can hence tolerate degraded service for a limited time in the future. Therefore, in the next epoch, A may be assigned to BS2 while B is switched over to BS1 (if past services of A and B are considered). Thus, B's deficit can be offset by the increased service it receives from BS1, and at the end of the second epoch, both A and B would have a surplus of 2.5 bits each. By periodically switching A and B betweens

BSs 1 and 2, the needs of both of them could be met, at least over reasonable time scales.

Co-scheduling related flows: In future multimedia applications, an SS can be expected to be associated with multiple video, audio, and even data flows. One complication that arises with MSBS when SSs are associated with multiple flows is ensuring that all the flows of an SS are assigned to the same BS. To deal with this problem, we propose *two-level hierarchical scheduling*. In this approach, in the first level, SS assignments and aggregate allocations to SSs are determined; in the second level, the slots allotted to an SS are distributed among its flows. We propose a SS utility function that can simplify scheduling while not sacrificing optimality.

Practical considerations: Implementing MSBS necessitates coordination among the WiMAX access network elements. In WiMAX network architecture, the ASN-Gateway, situated at the boundary of core and access-service networks, can act as a central entity that has knowledge of the quality of the channel between each BS-SS pair, and can recommend assignments and allocations with a long-term, macro view to the BSs [5].

We assume that the scheduling epochs over which MSBS is performed are long enough to balance the gains achieved over the overheads involved. Macro scheduling while accounting for the overheads, including those of handovers, is deferred for future work.

Contributions: Jointly scheduling BSs to achieve PF is known to be NP-hard if each SS or flow can be assigned to a single BS only [4]. In this paper, *we show that the problem is in fact NP-hard in the strong sense*. We therefore propose heuristics for the first level of our hierarchical scheduler and compare their efficiency and accuracy with those of the approximation algorithms proposed by Li *et al.* [4].

Second-level scheduling, which distributes the allocation of an SS among its flows, is equivalent to scheduling a set of flows within a single BS. Therefore, we consider the single-BS allocation problem. In WiMAX, since allocations at each decision time are in discrete units of slots (refer to Sec. III for more details), the resource allocation problem poses integrality constraints. We show that the problem can be *optimally solved in quadratic time while honoring the integrality constraints*.

Next, as mentioned earlier, we define the utility function of an SS based on the parameters of its flows such that it is different from the sum of the utilities of the individual flows, albeit yielding optimal allocations (that is, allocations given by sum of flow utilities) and simplifying scheduling.

Finally, through simulations, we evaluate the efficacy of our approaches in improving QoE of VoD flows in WiMAX networks.

Organization: The rest of the paper is organized as follows. Sec. II surveys related work. Relevant features of WiMAX are described in Sec. III. Our system model and notation are introduced and hierarchical scheduling is described in Sec. IV. Sec. V presents a quadratic-time algorithm for solving the single base station allocation problem, while Sec. VI shows that joint base station scheduling is NP-hard in the strong sense, and proposes heuristics. Our hierarchical approach is proved optimal in Sec. VII. Sec. VIII presents an empirical evaluation, while Sec. IX concludes.

II. Related Work

Li *et al.*'s PF scheduling for multi-rate WLANs is closely related to work in this paper [4]. As discussed in Sec. I, Li *et al.* consider maximizing the sum of logarithms of rates allocated to users in a network of access points (APs). They claim the problem can be shown to be NP-hard by adapting the reduction in [6], and propose two approximation algorithms (borrowed in this paper) with approximation ratios 5.828 and $2+\epsilon$. The differences between Li *et al.*'s work and this paper are highlighted in Sec. I. A precursor to [4] is the work in [6], which studies generalized PF in 3G data networks. In the setup there, each BS schedules SSs slot by slot. [6] illustrates that independent BS scheduling can lead to *non-Pareto optimal* bandwidth allocations and proposes optimal offline algorithms and online heuristics for the case when multi-user diversity depends only on the number of users.

Improving resource utilization in a WCDMA-like system of multiple cells with slot-by-slot scheduling model, through a distributed and cross-layer coordination framework among BSs, SSs, and a central server, is considered in [7]. Their framework consists of three optimization stages: In Stage 1, each SS chooses the BS to attach to based on signal strengths and BS loads (broadcasted by BSs). In Stage 2, during each slot, each BS opportunistically schedules an SS for service using a *weighted alpha rule*. In Stage 3, a central server chooses the value of α for all the BSs based on their loads that leads to "cell breathing" and balancing of load across BSs. This approach, however, does not take fairness into account during load balancing, and also requires specialized SSs.

In [8], dynamic load balancing and mitigating loss due to interference by powering down BSs in a coordinated manner in CDMA networks is considered. A centralized and a two-tier scheduler are proposed and compared with traditional independent schedulers deployed at BSs. The optimization algorithms proposed therein consider all possible BS combinations and are heuristic in nature with no guarantees on performance. Further, interference is less of an issue in WiMAX.

The suitability of PF in HSDPA systems for video streaming when the traffic is a mix of elastic and non-elastic applications is empirically studied in [9]. [10] proposes a scheduling algorithm for streaming services in OFDMA systems. However, allocations are considered for single cells only. The performance of weighted PF in terms of spectral efficiency, throughput, resource utilization, and fairness in WiMAX networks is studied in [11]. Several other works consider the subcarrier allocation problem in generic OFDMA systems for maximizing throughput and improving QoS subject to a power budget. Refer to [12], [13], and references therein. These do not consider WiMAX specific issues and cannot be easily extended to be used by WiMAX MAC schedulers.

III. Overview of WiMAX

WiMAX is based on the IEEE 802.16 standard [1], which defines the PHY and MAC specifications for last-mile connections in wireless metropolitan area networks. The WiMAX MAC layer is point-to-multipoint (PMP) with optional mesh support. The MAC layer can support multiple PHY specifications. This paper assumes that the underlying PHY is based on orthogonal frequency division multiple access (OFDMA).

WiMAX frames and scheduler: The basic unit of transmission in WiMAX is a *frame*. Frames are two-dimensional, with OFDM symbols (over time) in one dimension and subchannels (frequency) in the second dimension. A *subchannel*, consists of a logical collection of subcarriers, and is the

minimum frequency-resource unit of allocation. Subcarriers in a subchannel can be distributed over the entire frequency spectrum or contiguous. In this paper, we assume distributed permutation of subcarriers in a subchannel.

All frames are equal in size. The minimum time-frequency resource that can be allocated to a given flow is a *slot*. This consists of one subchannel over one, two, or three OFDM symbols [14]. A frame can be viewed as a collection of slots. WiMAX supports both time-division duplexing (TDD) and frequency-division duplexing (FDD). TDD is the preferred approach and we assume TDD in this paper. In TDD, each WiMAX frame is divided in time into two sub frames: a downlink (DL) sub frame and an uplink (UL) sub frame. A scheduler at the BS allocates the slots to the SSs both on the DL and UL. In this paper, we focus on the DL scheduler. Each SS should be allocated an integral number of slots by WiMAX schedulers.

Modulation and coding shemes: WiMAX supports *adaptive modulation and coding* (AMC). Under AMC, the channel modulation scheme and the strength of the FEC code, and in turn, the data rate, are based on the quality of the channel to the SS. WiMAX supports a (finite) number of modulation and coding schemes (MCS). To determine an appropriate MCS for an SS by the BS, SSs periodically report the quality of their channels to their neighboring BSs. It should be noted that in a single WiMAX frame, different slots can be coded using different MCSs based on the SS associated with the slots.

IV. System Model, Notation, and Utility Functions

In this section, we describe our system model, introduce needed notation and the WiMAX resource allocation problem.

Model and Notation: We consider joint allocation of resources to SSs and flows in a cluster of BSs that can be thought of as constituting a WiMAX sub-network. B denotes the number of BSs in the sub-network, N, the number of VoD of flows to be served by the BSs, and M, the number of SSs. The number of flows associated with SS i is denoted N_i. Hence, $\sum_{i=1}^{M} N_i = N$. When each SS is associated with a single flow only, $N = M$ holds. For ease of description, when unambiguous, we refer to a flow without referencing the SS it is associated with. In such cases, the index i in flow i is with respect to the universe of all flows.

The number of OFDM slots in each frame at each BS in the downlink is denoted S. (For simplicity, we assume equal number of slots in all BSs. Our results extend to the unequal case.) $m_{ij}(t)$ (resp., $\bar{m}_{ij}(t)$) denotes the rate, in terms of the number of data bytes that can be packed in a slot, for flow i (resp., SS i) at BS j at time t. $m_{ij}(t)$ depends on the modulation and coding scheme used in encoding the data for flow i, which in turn, depends on the quality of the associated channel. $x_{ij}(t)$ (resp., $\bar{x}_{ij}(t)$) denotes the number of slots allotted to flow i (resp., SS i) at BS j over all frames in a macro-scheduling epoch that begins at time t. For flow i of SS k, the rate per slot and the number of slots allocated at BS j shall be denoted m_{ij}^k and x_{ij}^k, respectively. In general, for any two flows i and ℓ of SS k, $m_{ij}^k = m_{\ell j}^k = \bar{m}_{kj}$ holds. $w_i > 0$ (resp., $w_i^k > 0$) is the weight assigned to flow i (resp., flow i of SS k), based on the bandwidth it requires (which is also the play-out rate in the case of a VoD flow).

$\bar{R}_i(t)$ (resp., $\bar{R}_i^k(t)$) denotes the average rate at which flow i (resp., flow i of SS k) is served until t. $T > 1$ is the time constant of the low-pass filter that is used to update the average rate. F is the number of frames in a scheduling epoch and t_f, the duration of each frame. Finally, $d_i = (T-1) \cdot F \cdot t_f \bar{R}_i(t) > 0$ and $d_i^k = (T-1) \cdot F \cdot t_f \bar{R}_i^k(t) > 0$. In all of the above notation, subscript j is omitted when only a single BS is considered, and for conciseness, time t is omitted.

We now define utility functions for flows and SSs, which will be used in SS and flow assignments and allocations.

Flow utility functions: Our goal is to determine an assignment of flows to BSs and an allocation of slots to flows at their assigned BSs that is in keeping with the bandwidth needs and past service rates of flows, and current channel conditions. A flow that has adequate buffer built up over the past in its lifecycle can afford being served at a lower rate in the current epoch despite its higher play-out rate. Similarly, opportunistic scheduling, which favors flows with good channel conditions and can hence improve overall system throughput, can be used to prepare for future contingency provided doing so does not adversely impact other flows. We therefore adopt *proportional fair* (PF) scheduling [3], which is known to find a balance between maximizing resource utilization and user fairness.

In single-carrier transmission systems, in which only a single flow is scheduled at each decision time, PF, which maximizes the sum of the weighted logarithms of long-term average rates for flows, *i.e.*, $\sum_i w_i \log(\bar{R}_i)$, can be achieved by scheduling that flow for which $\frac{w_i \cdot r_i(t)}{\bar{R}_i(t)}$ is maximized [15]. Here $r_i(t)$ denotes the rate achievable for flow i at time t, and $\bar{R}_i(t)$, the average rate at which i is served until t. The problem of extending PF scheduling to multi-carrier systems, in which multiple channels are scheduled at each decision time, is considered in [16]. The authors consider single-BS scheduling and derive an objective function that should be maximized for allocations to guarantee long-term proportional fairness. Using an approach similar to theirs, it can be shown that long-term PF can be achieved in single-BS OFDMA systems by allocating slots in each epoch to flows such that $\sum_{i=1}^{N} U_i(x_i)$ is maximized, where $U_i(x_i)$, which is the utility function for flow i, is given by

$$U_i(x_i) = w_i \log(1 + \frac{m_i x_i}{d_i}). \tag{1}$$

Recall that $d_i = (T-1) \cdot F \cdot t_f \bar{R}_i(t) > 0$ and $T > 1$ holds. If $T = 1$, then U_i is just $w_i \log(m_i x_i)$ since past service rate is ignored. For a flow that joins at time t, $\bar{R}_i(t)$ is initialized to αw_i, where $0 < \alpha \leq 1$.

When the number of BSs, B, exceeds 1, the utility function for flow i is given by

$$U_i(\langle x_{ij} \rangle_{j=1}^{B}) = w_i \log(1 + \frac{1}{d_i} \sum_{j=1}^{B} m_{ij} x_{ij}). \tag{2}$$

SS utility functions: One obvious choice for the utility function to associate with an SS k is the sum of the utilities of its flows, given by

$$U_k(\langle x_{1j}^k \rangle_{j=1}^{B}, \langle x_{2j}^k \rangle_{j=1}^{B}, \ldots, \langle x_{N_k j}^k \rangle_{j=1}^{B}) = \sum_{i=1}^{N_k} w_i^k \cdot \log(1 + \frac{1}{d_i^k} \sum_{j=1}^{B} \bar{m}_{kj} \cdot x_{ij}^k), \tag{3}$$

where $\bar{m}_{kj} = m_{ij}^k$, $1 \leq i \leq N_k$, is the rate per slot of SS k at BS j and is common to all its flows.

Using the above utility function for SSs will necessitate

introducing additional constraints to the optimization problem (considered later), for ensuring that all flows of an SS are assigned to the same BS, and add to its complexity.

We therefore define the following alternative utility function for SS k, which involves a single logarithmic term.

$$\bar{U}_k(\langle \bar{x}_{kj} \rangle_{j=1}^{B}) = (\sum_{i=1}^{N_k} w_i^k) \cdot \log(1 + \frac{1}{\sum_{i=1}^{N_k} d_i^k} \sum_{j=1}^{B} \bar{m}_{kj} \cdot \bar{x}_{kj}) \tag{4}$$

The above utility function may be used to determine an aggregate allocation to each SS by a first-level scheduler, which can then, at a second level, be apportioned among its flows. We will later show that allocations to individual flows using this two-level hierarchical approach, when optimal schedulers are employed at the two levels, is identical to the allocations they would receive under an optimal algorithm that uses (3) for SS utility functions.

Since the sum of flow utilities should be maximized at the second level, subject to not exceeding the number of slots allocated to the SS the flows are associated with, second level scheduling can be performed using single-BS allocation algorithms.

V. Single Base Station Allocation

In this section, we present an algorithm for the single base station resource allocation problem (SBSRA) in WiMAX. The problem is to determine at time t, the allocation x_i to flow i, $1 \leq i \leq N$, over the next scheduling epoch, by solving the following.

$$\begin{aligned} \text{SBSRA: Maximize } & U(x = \langle x_i \rangle_{i=1}^{N}) = \sum_{i=1}^{N} U_i(x_i), \\ & \text{where } U_i(x_i) = w_i \log(a_i + b_i \cdot x_i), \\ & \text{subject to } \sum_{i=1}^{N} x_i = S \\ & \qquad x_i \geq 0,\ x_i \in \mathbb{N}, \end{aligned} \tag{5}$$

where $w_i > 0$, $a_i = 1$, and $b_i = \frac{m_i}{d_i} = \frac{m_i}{(T-1) \cdot F \cdot t_f \bar{R}_i(t)} > 0$, for all i. $b_i = 0$ can hold if $m_i = 0$. For simplicity and to avoid boundary cases (when the non-negativity constraint is relaxed later), we assume that $m_i > 0$ holds for all i. This can be ensured by isolating all users with $m_i = 0$. The first constraint limits the total number of slots allocated to all the users in the BS to S. The above is a convex program with non-negativity and integrality constraints, and hence, solving it in a single step, using convex solvers, would be rather inefficient. We therefore solve it in two steps. In the first step, SBSRA-frac, which denotes a relaxed version of SBSRA, in which integrality constraints are relaxed, is solved by solving multiple instances (up to $N-1$) of SBSRA-rel. SBSRA-rel denotes the unconstrained version of the problem, in which both negativity and integrality constraints are relaxed. In the second step, the solution to SBSRA-frac is used to construct a solution to SBSRA.

A. Solution to SBSRA-rel

Since SBSRA-rel contains only an equality constraint and its objective function is concave, it can be solved using the technique of Lagrange multipliers [17]. Applying the technique to SBSRA-rel, we have the following system of linear equations, where λ is the Lagrange multiplier.

$$\frac{w_i b_i}{a_i + b_i x_i} = \lambda, \qquad \sum_{i=1}^{N} x_i = S$$

Solving the above equations, we have

$$x_i = \frac{w_i}{\sum_{j=1}^{N} w_j} (S + \sum_{j=1}^{N} \frac{a_j}{b_j}) - \frac{a_i}{b_i}. \tag{6}$$

Since $w_i > 0$, $x_i > \frac{-a_i}{b_i}$ holds for all i. Note that the optimal solution is unique and can take negative values for one or more of the unknowns. In fact, because the objective function is strictly concave and the constraints are linear, it can be shown that SBSRA-rel and SBSRA-frac have unique optimal solutions. The same need not hold for SBSRA.

B. Solution to SBSRA-frac

Recall that SBSRA-frac is obtained from SBSRA by relaxing the integrality constraints. Let $x^* = \langle x_i^* \rangle_{i=1}^{N}$, and $y^* = \langle y_i^* \rangle_{i=1}^{N}$ denote the optimal solutions to SBSRA-rel and SBSRA-frac, respectively. If $x_i^* \geq 0$, for all $1 \leq i \leq n$, then it easily follows that $y^* = x^*$. We next show that should $x_j^* < 0$ hold for some j, then $y_j^* = 0$.

Lemma 1. *Let $x^* = \langle x_i^* \rangle_{i=1}^{N}$ and $y^* = \langle y_i^* \rangle_{i=1}^{N}$ denote the optimal solutions to* SBSRA-rel *and* SBSRA-frac *respectively. Then $x_i^* < 0 \Rightarrow y_i^* = 0$ for all $1 \leq i \leq N$.*

Proof: Let $x_i^* = \alpha$, where $0 > \alpha > \frac{-a_i}{b_i}$. Suppose $y_i^* = \gamma$ where $\gamma \in \mathbb{R}^+$. Then, because $\sum_{\ell=1}^{n} x_\ell^* = S = \sum_{\ell=1}^{n} y_\ell^*$ and $x_i^* < 0 < y_i^*$, it follows that there exists some j, such that $x_j^* > y_j^*$. Let $x_j^* = \beta$, where $\beta \in \mathbb{R}^+$, and $y_j^* = \beta - \delta$, where $0 < \delta \leq \beta$. Since y^* is optimal, it follows that $U_i(\gamma) + U_j(\beta - \delta) \geq U_i(\gamma - \epsilon) + U_j(\beta - \delta + \epsilon)$, where $0 < \epsilon < \delta$ (and U_i and U_j are as defined in (1)). The above inequality may be rewritten as

$$U_i(\gamma) - U_i(\gamma - \epsilon) \geq U_j(\beta - \delta + \epsilon) - U_j(\beta - \delta). \tag{7}$$

Since $U_i(x_i)$ and $U_j(x_j)$ are strictly concave, we have the following.

$$U_i(\alpha + \epsilon) - U_i(\alpha) > U_i(\gamma) - U_i(\gamma - \epsilon) \tag{8}$$

(because $\alpha < 0$ and $\gamma > 0$)

$$U_j(\beta - \delta + \epsilon) - U_j(\beta - \delta) > U_j(\beta) - U_j(\beta - \epsilon) \tag{9}$$

From (8), (7), and (9), we have

$$\begin{aligned} & U_i(\alpha + \epsilon) - U_i(\alpha) > U_j(\beta) - U_j(\beta - \epsilon) \\ \equiv\ & U_i(\alpha + \epsilon) + U_j(\beta - \epsilon) > U_i(\alpha) + U_j(\beta), \end{aligned}$$

which is a contradiction to the optimality of x^* to SBSRA-rel. ■

We now present an algorithm for solving SBSRA-frac using SBSRA-rel. The approach is to recursively solve several instances of SBSRA-rel until all the variables in the solution are non-negative. Let SBSRA-rel^k denote a constrained version of SBSRA-rel, in which $N-k$ of the variables are specified to be zeroes. The objective function of SBSRA-rel^k is therefore the sum of the utility functions of only the k flows associated with the remaining variables not specified as zero; similarly, the constraint on the number of slots in (5) is on the sum of these variables only. The RHS of (5) however remains the same. To begin with, we consider SBSRA-rel^N, *i.e.*, SBSRA-rel in its entirety. If each x_i is non-negative in the solution to SBSRA-rel in this step, then this solution forms the solution

Algorithm 1 Algorithm $\mathcal{A}_{\text{SBSfr}}$ for solving SBSRA-frac using SBSRA-rel

1) $I = \{1, 2, .., n\}$;
2) **while** TRUE **do**
3) $\quad x_i^* := \frac{w_i}{\sum_{j \in I} w_j}(S + \sum_{j \in I} \frac{a_j}{b_j}) - \frac{a_i}{b_i}$, for all $i \in I$;
4) $\quad$ **if** $x_i^* \geq 0$, for all $i \in I$ **then**
5) $\quad\quad y_i^* := x_i^*$, for all $i \in I$;
6) $\quad\quad$ RETURN y^*
$\quad$ **fi**
7) $\quad$ **for** all $j \in I$ **do**
8) $\quad\quad$ **if** $x_j^* < 0$ **then set** $y_j^* := 0$**;** $I := I - \{j\}$ **fi**
9) $\quad$ **od**
od

to SBSRA-frac also and the algorithm terminates. Otherwise, by Lemma 1, each x_i that is negative assumes a value of zero in the solution to SBSRA-frac, and hence is set to zero. Let $r < N$ denote the number of variables remaining. To determine the values of these remaining variables, a constrained version of SBSRA-frac that considers only these r variables is solved, by considering a corresponding instance of SBSRA-rel^r. As with the previous step, the algorithm either terminates or continues to the next step with the positive variables of this step. The procedure continues until some instance of SBSRA-rel^k, $k \geq 1$, yields a non-negative solution. A complete listing is provided in Algorithm 1.

The correctness of the above algorithm follows from Lemma 1, and the fact that if non-negative, the optimal solution to any SBSRA-rel^k is also optimal for a corresponding instance of SBSRA-frac^k, for all $1 \leq k \leq N$. It is easy to see that Lemma 1 applies to the constrained versions of SBSRA-rel and SBSRA-frac alsoTherefore, the solution to SBSRA-frac can be obtained by recursively solving constrained instances.

Within the **while** loop (in line 2), lines 3–5 and the **for** loop in line 7 have $O(N)$ complexity. The **while** loop itself can be executed $O(N)$ times, and therefore, the worst-case complexity of the above algorithm is $O(N^2)$.

C. Solution to SBSRA

The final step is to convert the optimal non-negative solution, that may be fractional, obtained by solving SBSRA-frac to an optimal solution to SBSRA, which requires that all variables be non-negative and integral. Towards that end, the lemma that follows determines bounds on the values of the variables in the solutions to SBSRA-frac and SBSRA. If y and y^* are optimal solutions to SBSRA-frac and SBSRA, respectively, then according the lemma, every y_i is bounded from above by $\lfloor y_i^* \rfloor + 1$ or bounded from below by $\lfloor y_i^* \rfloor$.

Lemma 2. *Let $y = \langle y_i \rangle_{i=1}^N$ be an optimal solution to* SBSRA *and $y^* = \langle y_i^* \rangle_{i=1}^N$ the optimal solution to* SBSRA-frac*. Then there exists an i, such that $y_i < \lfloor y_i^* \rfloor$ if and only if there does not exist a j, such that $y_j > \lfloor y_j^* \rfloor + 1$.*

Proof: Suppose the lemma does not hold. Then, there exist i and j such that $0 \leq y_i \leq \lfloor y_i^* \rfloor - 1$ and $y_j \geq \lfloor y_j^* \rfloor + 2$, where $y_i \geq 0$, $y_j^* \geq 0$, $y_i^* \geq 1$, and $y_j \geq 2$. Therefore, $y_i \leq y_i^* - 1$ and $y_j > y_j^* + 1$. Because $y_i \geq 0$ and $y_j \geq 2$ (by $y_j^* \geq 0$), and y is optimal for SBSRA, we have

$$U_j(y_j) - U_j(y_j - 1) \geq U_i(y_i + 1) - U_i(y_i). \quad (10)$$

Algorithm 2 Algorithm $\mathcal{A}_{\text{SBS}}$ to solve SBSRA

1) SOLVE SBSRA-frac; let $y^* = \langle y_1^*, y_2^*, ..., y_N^* \rangle$ denote its solution
2) $y_i := \lfloor y_i^* \rfloor$, for all $1 \leq i \leq N$; $\gamma := \sum_{i=1}^N \lfloor y_i^* \rfloor$;
3) $\eta := S - \gamma$;
4) **if** $\eta = 0$ **then** exit **fi**;
5) **while** TRUE **do**
6) $\quad next_i := U_i(y_i + 1) - U_i(y_i)$, for all i;
7) $\quad prev_i := \begin{cases} \infty, & y_i = 0 \\ U_i(y_i) - U_i(y_i - 1), & y_i > 0 \end{cases}$
8) $\quad$ **if** $\max_{1 \leq i \leq n}\{next_i\} \leq \min_{1 \leq i \leq n}\{prev_i\}$ **then** BREAK **fi**
9) $\quad k := \{i | next_i \geq next_j, \forall j \neq i\}$;
10) $\quad k' := \{i | prev_i \leq prev_j, \forall j \neq i\}$;
11) $\quad y_k := y_k + 1$; $y_{k'} := y_{k'} - 1$;
$\quad$ **od**
12) **for** $r = 1, 2, ..., \eta$ **do**
13) $\quad k := \min_i \{i | U_i(y_i + 1) - U_i(y_i) \geq U_{i'}(y_{i'} + 1) - U_{i'}(y_{i'}) \, \forall i'\}$
14) $\quad y_k := y_k + 1$
$\quad$ **od**

Since U_i and U_j are strictly concave, we have

$$U_i(y_i + 1) - U_i(y_i) \geq U_i(y_i^*) - U_i(y_i^* - 1) \text{ and} \quad (11)$$
$$U_j(y_j^* + 1) - U_j(y_j^*) > U_j(y_j) - U_j(y_j - 1). \quad (12)$$

Using (12), (10), and (11), we have $U_j(y_j^* + 1) - U_j(y_j^*) > U_i(y_i^*) - U_i(y_i^* - 1)$, *i.e.*, $U_j(y_j^* + 1) + U_i(y_i^* - 1) > U_i(y_i^*) + U_j(y_j^*)$, which, because $y_i^* \geq 1$, contradicts the optimality of y^* to SBSRA-frac. ■

The corollary below easily follows.

Corollary 3. *Let Y and Y^* be as defined in Lemma 2. If for any j, $y_j > \lfloor y_j^* \rfloor + 1$, then $y_i \geq \lfloor y_i^* \rfloor$ for all i.*

Pseudo-code for an algorithm that makes use of Lemma 2 to solve SBSRA is provided in the listing in Algorithm 2.

Description of Algorithm 2: Algorithm 2 functions in three phases. In the first phase in lines 1–4, SBSRA-frac is solved and a a non-negative solution, y^*, that may be fractional is obtained. Each flow i is also assigned a base allocation of $\lfloor y_i^* \rfloor$ slots in this phase. If $\lfloor y_i^* \rfloor$ is integral for all i, then the base allocation is an optimal integral allocation, and the algorithm terminates. Otherwise, the next two phases are executed.

In the second phase (spanning lines 5–11), the base allocation of the first $\gamma = \sum_{i=1}^N \lfloor y_i^* \rfloor$ slots performed in the first phase among the N flows is converted to an optimal allocation of these γ slots. Allocation Y would be optimal, if there do not exist flows i and j such that the marginal utility of the $(y_i + 1)^{\text{st}}$ slot of i, *i.e.*, $U_i(y_i + 1) - U_i(y_i)$, is greater than $U_j(y_j) - U_j(y_j - 1)$, the marginal utility of the y_j^{th} slot of j, where $y_j > 0$. The **while** loop in line 5 executes until the optimality condition is reached (in line 8). By Lemma 2, the loop is guaranteed termination in at most $\eta + N$ iterations (proved later as part of proof of Theorem 1).

The final phase consists of optimally assigning the remaining $\eta = S - \gamma$ slots among the flows, and is carried out in the **for** loop in lines 12–14. For each slot, the flow with the largest marginal utility for an additional slot is chosen.

Correctness proof for Algorithm $\mathcal{A}_{\text{SBS}}$: A correctness proof for the algorithm claimed in Theorem 1 is provided in an appendix. Informally correctness follows from Lemma 2, which guarantees that the number of iterations of the **while** loop of lines 5–11 is at most $\eta + N = O(N)$. Correctness of the

allocation in the **for** loop follows from the strict concavity of U_i, for all i.

Theorem 1. *Algorithm $\mathcal{A}_{\text{SBS}}$ optimally solves* SBSRA *and has a worst-case time complexity of $O(N^2)$.*

VI. Macro-Scheduling among Multiple Base Stations

This section considers MSBS, which performs coordinated scheduling among multiple BSs. Aggregate allocations are determined for SSs, each of which may be associated with multiple flows. Each SS's utility is modeled using (4). Since each SS can be assigned to at most one BS in a scheduling epoch, the goal is to determine a many-to-one mapping from the set of SSs to the set of BSs such that the sum of SS utilities is maximized. The problem solved in each epoch is as follows.

$$\text{MBSRA: Maximize } \sum_{k=1}^{M} \bar{U}_k(\langle \bar{x}_{kj} \rangle_{j=1}^{B}),$$

$$\text{where } \bar{U}_k(\langle \bar{x}_{kj} \rangle_{j=1}^{B}) = w_k \log(1 + \sum_{j=1}^{B} b_{kj} \bar{x}_{kj}),$$

$$b_{kj} = \frac{\bar{m}_{kj}}{\sum_{i=1}^{N_k} (T-1) \cdot t_f \cdot F},$$

$$\text{subject to } \sum_{k=1}^{M} \bar{x}_{kj} = S, 1 \le j \le B \quad (13)$$

$$\bar{x}_{kj} \ge 0, \bar{x}_{kj} \in \mathbb{N}, \quad k = 1, \ldots, M, j = 1, \ldots, B \quad (14)$$

$$(\forall j, \ell, j \ne \ell :: \bar{x}_{kj} > 0 \Rightarrow \bar{x}_{k\ell} = 0), k = 1, \ldots, M \quad (15)$$

The constraint in (13) restricts the total number of slots allocated to all the SSs in any BS to at most S, while (14) requires the number of slots assigned to any SS to be a non-negative integer. (15) confines each SS to a single BS. (13) and (15) implicitly restrict the number of slots assigned to any SS to at most S.

A. Hardness Proof

We prove that the problem is NP-hard in the strong sense by providing a reduction from the 3-PARTITION (3-part) problem, which is NP-Complete in the strong sense. The reduction uses our solution to SBSRA. In [4], Li *et al.* claim that this problem can be shown to be NP-hard by adapting a reduction in [6] from the 3-dimensional matching problem.

3-part is a number problem [18] defined as follows.

Definition 1 (3-part)**:** Given set E of $3m$ elements, $e_1, e_2, \ldots, e_{3m}$, a bound $K \in \mathbb{Z}_+$, and a size $s(e_i) = s_i \in \mathbb{Z}_+$ for each $e_i \in E$ such that $K/4 < s_i < K/2$ and $\sum_{i=1}^{3m} s_i = mK$. The problem is to determine whether E can be partitioned into m disjoint sets $E_1, E_2, \ldots, E_m$ such that $\sum_{e \in E_i} s(e) = K$, for $1 \le i \le m$.

Theorem 2. *The multiple-base station resource allocation problem in WiMAX,* MBSRA*, is NP-hard in the strong sense.*

Proof: To prove the theorem, we show that the decision version of MBSRA is NP-complete in the strong sense. It is easy to see that the decision version is in NP. We provide a pseudo-polynomial reduction [18] from 3-part to it.

In 3-part, without loss of generality, assume that $s_1 \ge s_i$, for all $1 < i \le 3m$.

SBS-3part**:** Consider an instance of the unconstrained version of the single-BS allocation problem SBSRA-rel, denoted SBS-3part, with the following parameters: $N = 3m$, $S = mK$, $a_i = w_i = 1$, and $b_i = \frac{1}{1+s_1-s_i} > 0$ (because $s_1 \ge s_i$), for $1 \le i \le N$. As discussed in Sec. V-A, SBSRA-rel has a unique optimal solution when $b_i > 0$. By (6), the unique optimal solution to SBS-3part is given by $\langle x_i \rangle_{i=1}^{3m} = \langle s_i \rangle_{i=1}^{3m}$. Note that this solution is non-negative and the unique optimal objective value is given by

$$OPT = \sum_{i=1}^{3m} \log(1 + b_i \cdot x_i) = \sum_{i=1}^{3m} \log(1 + \frac{s_i}{1 + s_1 - s_i}). \quad (16)$$

We now reduce an instance of 3-part to MBSRA.

Reduction, MBS-3part**:** Construct an instance of MBSRA, denoted MBS-3part, from an arbitrary instance of 3-part as follows. Let $B = m$, $M = 3m$, $S = K$, and $w_i = 1$, for $1 \le i \le M$. For each i, let $b_{ij} = b_i = \frac{1}{1+s_1-s_i}$, for all $1 \le j \le B$. We claim that a solution to 3-part exists if and only if there is a solution to MBS-3part with objective value at least OPT as defined in (16).

$\Leftarrow$**:** Assume MBS-3part has a solution with objective value at least OPT. In this solution, for each i, at most one of $\bar{x}_{ij}$, for $1 \le j \le B$, that corresponds to the BS that i is assigned to, is positive. Let β_i denote this unique BS, if such a BS exists, and be undefined, otherwise (when $x_{ij} = 0$ for all j). $\bar{x}_{i\beta_i}$ is the number of slots allotted to i if β_i exists, and 0, otherwise.. Now, a feasible solution to SBS-3part can be constructed from the solution to MBS-3part as follows:

$$x_i = \begin{cases} \bar{x}_{i\beta_i}, & \beta_i \text{ is defined} \\ 0, & \text{otherwise} \end{cases}$$

This is because **(i)** b_{ij} of MBS-3part equals b_i of SBS-3part for all i, j, **(ii)** for each i, at most one of $\bar{x}_{ij}$ is positive in any solution to MBS-3part, **(iii)** the total capacity of the K BSs in MBS-3part, $K \cdot m$, is equal to the capacity of the single BS in SBS-3part, hence $\sum_{j=1}^{B} \sum_{k=1}^{3m} \bar{x}_{kj} = mK = \sum_{i=1}^{3m} x_{\beta_i}$ and **(iv)** the remaining parameters are equal in both the problems: $N = 3m$ of SBS-3part equals M of SBS-3part, and a_i's and w_i's are identical in both the problems, for $1 \le i \le 3m$. Hence, unless β_i is well defined and $\bar{x}_{i\beta_i} = s_i > 0$, for each i, the objective value of SBS-3part can be either increased or achieved with a solution different from $\langle s_i \rangle_{i=1}^{3m}$. This would contradict either the optimality of OPT for SBS-3part or the fact that its optimal solution is unique. Hence, MBS-3part has a unique solution given by for all i, $x_{ij} = 0$, for all $j \ne \beta_i$ and $x_{i\beta_i} = s_i$, and it is easy to see that a solution to 3-part is obtained by simply assigning e_i to set E_{β_i}.

$\Rightarrow$**:** Assume 3-part has a solution. Let element e_i be assigned to set E_k. Then, a solution to MBS-3part that achieves an objective value of exactly OPT can be obtained by assigning SS i to BS k.

It is easy to verify the the reduction can be performed in polynomial time. Further all numbers in MBS-3part (M, B, S, w_i, b_i, and OPT) are polynomially bounded by the numbers in 3-part. Thus, the decision version of MBSRA is also NP-complete in the strong sense. MBSRA is therefore NP-hard in the strong sense. ■

In [4], Li *et al.* propose two approximation algorithms for the problem. In the first, denoted cvap, the constraint in (15), which restricts each SS to a single BS, and the integrality

Algorithm 3 Heuristic ffrac for determining an approximate fractional solution

variables
$BSRankForSS$: **array** $[1..N]$ of **array** $[1..B]$ of **integers**

1) **for** $pass$:= 1 to B **do**
2) **for** BS := 1 to B **do**
▷ Find all SSs for which base station BS provides
▷ among the $pass$ highest bandwidths
3) $SS_{pass}[BS]$:= $\{i | BSRankForSS[i][BS] \leq pass\}$
od
4) **for** $round$:= 1 to $pass$ **do**
5) **for** BS := 1 to B **do**
6) **for** each $i \in SS_{pass}[BS]$ **do**
7) $rank$:= $BSRankForSS[i][BS]$;
8) $higher\_alloc[i]$:= sum of allocations in bytes for SS i in BSs with ranks 1 to $rank-1$
9) $U_i := w_i \log(1+\frac{higher\_alloc[i]}{d_i}+\frac{m_{i,BS}x_{i,BS}}{d_i})$
od
10) Solve SBSRA for base station BS and determine $x_{i,BS}$ for $i \in SS_{pass}[BS]$
11) $x_{i,BS} := 0$, for $i \notin SS_{pass}[BS]$
od
od
od

constraint in (14) of MBSRA are relaxed and a fractional solution to the resulting convex program in which an SS can be assigned to multiple BSs is obtained. The fractional solution is then rounded using the technique based on bi-partite graph construction proposed in [19] for the *generalized assignment problem*. Li *et al.* show that this method provides an approximation ratio of 5.828. Their second algorithm, nlap, discretizes an equivalent exact non-linear formulation of MBSRA to construct a relaxed linear program, in which an SS may be associated with multiple BSs. The solution to the discretized linear program is then rounded to obtain a solution with approximation ratio $2+\epsilon$. In this paper, we evaluate cvap, and two other heuristics that we design. (nlap is omitted due to its high running time.)

B. Heuristics

cvap involves solving a convex program, and nlap, a linear program, in which the number of variables needed depends on the desired accuracy. Hence, the running times of both these algorithms can be quite considerable. (Our simulations confirm this.) We therefore propose efficient heuristics.

Highest Bandwidth First (hbf): The first heuristic, called *highest bandwidth first* (hbf), assigns an SS to the BS from which it is likely to receive the highest bandwidth (bw). When the load distribution is asymmetric, the highest-bw BS for an SS need not be the one with the strongest signal. To determine SS assignments, the single-BS allocation algorithm, $\mathcal{A}_{\text{SBS}}$, is run at each BS for all SSs. Each SS is assigned to the BS at which the bw assigned by $\mathcal{A}_{\text{SBS}}$ is the largest. In the second step, allocations to the SS's assigned to a BS are performed using the single-BS allocation algorithm. The complexity of this heuristic can easily be seen to be $O(BN^2)$.

Approximating the fractional solution (ffrac): The convex and linear solvers of cvap and nlap are used for determining near-optimal fractional solutions (that are then rounded). To lower running time, we consider heuristics for this step, and use $\mathcal{A}_{\text{SBS}}$ to determine a fractional solution. We call the heuristic ffrac (for fast fractional), pseudo-code for which is provided in the listing in Algorithm 3. The idea is to iteratively determine allocations for SSs in successive BSs in B passes. In the first pass, in each BS k, only those flows for which BS k provides the highest bandwidth are allocated. (Ties can be resolved arbitrarily.) In general, in the i^{th} pass, in BS k, SSs for which that BS provides among the highest i bandwidths are allocated. Allocations at each BS are performed independently using $\mathcal{A}_{\text{SBS}}$. Also, in pass i, SS allocations in the BSs are revised using i rounds for better accuracy. Refer to the **for** loop beginning in line 4.

One aspect to note is that due to non-linearity, the utility function associated with an SS cannot be identical at all BSs. Let $higher\_alloc[j]$ denote the number of bytes allocated to SS j in its top i BSs. Then, the utility function to be used at the BS with the next highest bandwidth is obtained by adding $higher\_alloc[j]/d_j$ (assuming $N_j = 1$) to the log term in (1). The i rounds of pass i are needed to ensure that $higher\_alloc[j]$ used at BS i_1 is not invalidated due to reallocations later at BS i_2, where $i_2 > i_1$, and i_2 provides a higher bandwidth to SS j than i_1. It can be shown that due to the absence of circular dependencies among BSs, convergence is guaranteed in at most i rounds in the i^{th} pass.

A second aspect concerns limiting the total number of slots allotted to an SS to S. This can be ensured by not considerating an SS in poorer BSs when the total allocation to it in prior BSs reaches S. The complexity of this heuristic is $O(B^3N^2)$, and in our simulation experiments was found to be more than an order of magnitude faster than the convex solver.

VII. Hierarchical Joint Base Station Scheduling

In this section, we show that the two-level hierarchical approach proposed for ensuring that all flows of an SS are assigned to the same BS is optimal. For this, we show that the number of bytes allocated to each flow remains the same regardless of whether (3) or (4) is used as the utility function for SS k in MBSRA. One difference is that, if (3) is used as the utility function, then the solution to MBSRA would directly yield per-flow allocations, whereas flow allocations should be determined in a second step if (4) is used. For simplicity, we consider a relaxed version of MBSRA in which integrality constraints and constraints confining each SS to a single BS in (14) and (15), respectively, are relaxed. The non-negativity constraints in (14) still hold. We will refer to this relaxed version as MBSRAREL.

Before proceeding further, we claim the following.

Claim 1. *Let* $f(b) = (\sum_{j=1}^{n} w_j) \cdot \log(1 + \frac{b}{\sum_{j=1}^{n} d_j})$ *and* $f_i(b_i) = w_i \log(1 + \frac{b_i}{d_i})$, *where* $b \geq 0$, $b_i \geq 0$, $d_i > 0$, *and* $w_i > 0$ *for* $1 \leq i \leq n$. *Let* $\beta, \beta_j \in \mathbb{R}$.
(i) *If* $\beta \leq \sum_{j=1}^{n} \beta_j$, *then* $\frac{\mathrm{d}f}{\mathrm{d}b}\Big|_{b=\beta} \geq \frac{\mathrm{d}f_i}{\mathrm{d}b_i}\Big|_{b_i=\beta_i}$ *holds for some* $1 \leq i \leq n$.
(ii) *If* $\beta \geq \sum_{j=1}^{n} \beta_j$, *then* $\frac{\mathrm{d}f}{\mathrm{d}b}\Big|_{b=\beta} \leq \frac{\mathrm{d}f_i}{\mathrm{d}b_i}\Big|_{b_i=\beta_i}$ *holds for some* $1 \leq i \leq n$.

Proof: We prove part **(i)**. The proof for the other part is similar. $\frac{\mathrm{d}f}{\mathrm{d}b} = \frac{(\sum_{j=1}^{n} w_j)}{\sum_{j=1}^{n} d_j + b}$ and $\frac{\mathrm{d}f_i}{\mathrm{d}b_i} = \frac{w_i}{d_i + b_i}$. Suppose the claim does not hold. That is, $\frac{\mathrm{d}f}{\mathrm{d}b}\Big|_{b=\beta} < \frac{\mathrm{d}f_i}{\mathrm{d}b_i}\Big|_{b_i=\beta_i}$, for all i. Then,

by the above derivatives, we have $(\sum_{j=1}^{n} w_j)(d_i + \beta_i) < w_i((\sum_{j=1}^{n} d_j) + \beta)$, for all $1 \leq i \leq n$. Summing the left-hand and right-hand sides of these n inequalities, we have $(\sum_{i=1}^{n} w_i) \sum_{j=1}^{n}(d_j+\beta_j) < \sum_{i=1}^{n} w_i((\sum_{j=1}^{n} d_j)+\beta)$, which is a contradiction, since $\beta \leq \sum_{j=1}^{n} \beta_j$. ■

We next show that the hierarchical approach proposed is optimal.

Theorem 3. *The hierarchical approach described in IV is optimal for solving* MBSRAREL *if optimal algorithms are used for allocation at the first and second levels.*

Proof: Let Hier denote our hierarchical approach with optimal schedulers for both the first and second levels, and Opt, some arbitrary optimal algorithm. Let $\langle H_i \rangle_{i=1}^{M}$ and $\langle O_i \rangle_{i=1}^{M}$ denote the aggregate allocations in bytes (not slots) to the M SSs (the number of bytes to SS k at BS j is given by $\bar{m}_{kj}\bar{x}_{kj}$) by Hier and Opt, respectively. Suppose the theorem does not hold. Then, there exists an SS i, such that $H_i < O_i$. Otherwise, since Hier's second-level scheduler is optimal, for each SS i, its allocation of H_i bytes would be optimally allocated among its flows, that is, the sum of its per-flow utilities under Hier would be at least the sum under Opt (because per-flow utility functions at the second-level of Hier are the same as those in Opt), contradicting the fact that Opt is optimal. Next, because $H_i < O_i$, there must exist an SS ℓ such that $H_\ell > O_\ell$. Otherwise, a higher aggregate utility for the SSs than achieved by Hier's first level scheduler can be obtained using Opt's allocation (that is, a solution to Hier in which each SS is assigned an aggregate allocation that equals the sum of the allocations to its flows in Opt in each BS would have a higher utility than that of the the solution of Hier). This contradicts the fact that Hier's first-level scheduler is optimal.

Let $H_i = \sum_{k=1}^{B} \bar{m}_{ik}\bar{X}_{ik}$ denote the number of bytes allocated to SS i by Hier's first-level scheduler. Then, for SS i, $\bar{U}_i(H_i) = \bar{U}_i(\langle \bar{X}_{ik} \rangle_{k=1}^{B}) = (\sum_{j=1}^{N_i} w_j^i) \cdot \log(1 + \frac{H_i}{\sum_{j=1}^{N_i} d_j^i})$ (refer (4)). Note that this function is strictly concave in H_i. Let O_j^i, $1 \leq j \leq N_i$, denote the number of bytes allocated to flow j of SS i by Opt. Let $U_i(O_i) = U_i(\langle O_j^i \rangle_{j=1}^{N_i}) = \sum_{j=1}^{N_i} w_j^i \log(1 + \sum_{k=1}^{B} \frac{\bar{m}_{ik} \cdot X_{jk}^i}{d_j^i}) = \sum_{j=1}^{N_i} w_j^i \log(1 + \frac{O_j^i}{d_j^i}) = \sum_{j=1}^{N_i} U_i^j(O_i^j)$, the sum of the utilities of the flows of SS i in Opt. Let H_ℓ and O_ℓ be defined similarly to H_i and O_i, respectively. Since Hier's first-level scheduler is optimal, for any arbitrarily small ϵ, we have

$$\bar{U}_i(H_i + \epsilon) - \bar{U}_i(H_i) \leq \bar{U}_\ell(H_\ell) - \bar{U}_\ell(H_\ell - \epsilon). \quad (17)$$

Since Opt is optimal, we also have

$$(\forall 1 \leq k \leq N_\ell, 1 \leq j \leq N_i :: U_k^\ell(O_k^\ell + \epsilon) - U_k^\ell(O_k^\ell) \leq U_j^i(O_j^i) - U_j^i(O_j^i - \epsilon)). \quad (18)$$

Since $H_\ell > O_\ell$ and $\bar{U}_\ell$ is strictly concave, $\bar{U}_\ell(H_\ell) - \bar{U}_\ell(H_\ell - \epsilon) < \bar{U}_\ell(O_\ell + \epsilon) - \bar{U}_\ell(O_\ell)$, for $\epsilon < H_\ell - O_\ell$. In the limit $\epsilon \to 0$, $\frac{\bar{U}_\ell(H_\ell) - \bar{U}_\ell(H_\ell - \epsilon)}{\epsilon} = (\bar{U}_\ell)'(H_\ell - \epsilon)$, $\frac{\bar{U}_\ell(O_\ell + \epsilon) - \bar{U}_\ell(O_\ell)}{\epsilon} = (\bar{U}_\ell)'(O_\ell)$, and $\frac{U_k^\ell(O_k^\ell + \epsilon) - U_k^\ell(O_k^\ell)}{\epsilon} = (U_k^\ell)'(O_k^\ell)$. Because $O_\ell = \sum_{k=1}^{N_\ell} O_k^\ell$, by Claim 1(ii), $(\bar{U}_\ell)'(O_\ell) \leq (U_k^\ell)'(O_k^\ell)$, and hence, $(\bar{U}_\ell)'(H_\ell - \epsilon) < (U_k^\ell)'(O_k^\ell)$ holds for some k. Therefore, by (17) and (18), in the limit $\epsilon \to 0$, $\frac{\bar{U}_i(H_i + \epsilon) - \bar{U}_i(H_i)}{\epsilon} < \frac{U_j^i(O_j^i) - U_j^i(O_j^i - \epsilon)}{\epsilon}$, holds for all $1 \leq j \leq N_i$. Next, since $H_i < O_i$ and $\bar{U}_i$ is strictly concave, for $\epsilon < O_i - H_i$, $\bar{U}_i(O_i) - \bar{U}_i(O_i - \epsilon) \leq \bar{U}_i(H_i + \epsilon) - \bar{U}_i(H_i)$. Therefore, by the previous inequality, we have in the limit $\epsilon \to 0$, $\frac{\bar{U}_i(O_i) - \bar{U}_i(O_i - \epsilon)}{\epsilon} < \frac{U_j^i(O_j^i) - U_j^i(O_j^i - \epsilon)}{\epsilon}$, that is, $(\bar{U}_i)'(O_i - \epsilon) < U_j^i(O_j^i - \epsilon)$, for all $1 \leq j \leq N_i$. This contradicts Claim 1(i), since $O_i = \sum_{j=1}^{N_i} O_j^i$. The theorem follows. ■

VIII. Empirical Evaluation

This section presents results of our simulation experiments.

A. Experimental Setup

Parameter	Value
BS/SS antenna height	32 m/1.5m
Carrier Frequency	2.5 GHz
Duplexing Scheme	TDD (DL:UL Ratio= 2:1)
BS to BS distance	1 Km
Rms transmit power per carrier	43 dBm
BS/SS antenna gain	16 dBi, 0 dBi
BS,SS Hardware loss	2 dB
SS Noise Figure	7 dB
Thermal Noise Density	-174 dBm/Hz
Bandwidth	10 MHz
Frame Duration	5ms
Number of DL slots per frame	450
DL permutation type	PUSC
Penetration and other losses	10 dB

Fig. 2: WiMAX system parameters used in simulations.

We use the macro-cell model and system parameters recommended by the WiMAX Forum [20] for our experiments. We simulate 19 cells, with a frequency reuse of three, placed in a regular, three-tier hexagonal tessellation. Performance evaluation is over the serving set of the inner seven cells; the outermost tier of twelve cells model realistic interference. For the mobility experiments, we assumed pedestrian mobility at 3km/h [20]. Since the epoch durations over which the channel quality is estimated are long in comparison to channel coherence times, we model only large-scale effects, namely, path loss per the modified COST231 Hata urban propagation model and 8dB Log-Normal Shadowing. Interference only due to other BSs is modeled assuming edge users in neighboring cells can be allocated in non-overlapping subchannels. WiMAX system parameters used in our simulations are listed in Fig. 2.

SS locations are randomly generated in two modes, *uniform* and *hotspot*. The number of SSs, N, is set at 132, with playout rates set at 384 Kbps, 512 Kbps, and 1 Mbps, for every $1/3^{\text{rd}}$ of the user population. Further increase in load led to a sharp drop in the performance metrics of all the algorithms. For the uniform mode, each SS is placed randomly in a disc of radius $2R$, where R is the cell radius. For the hotspot mode, we consider two scenarios. In the first scenario (hotspot-1), $N/2$ users are placed in a disc of radius R and the remaining half are placed in an annulus of width R, whereas in the second scenario (hotspot-2), all N users are placed in a central disc of radius R. Each scheduling epoch is set to 10 secs when users are stationary and is lowered to 5 secs when users are mobile, to account for the greater variations in channel conditions under mobility. Experiments are conducted over 200 epochs.

Each SS is assumed to be associated with a single flow. All SS buffers are assumed to be empty at the start of each simulation run. Our measures of performance are as follows: *mean statlling fraction* (MSF), which is the fraction of time a user stalls in an epoch, and *percentage of satisfied users* (PSU)

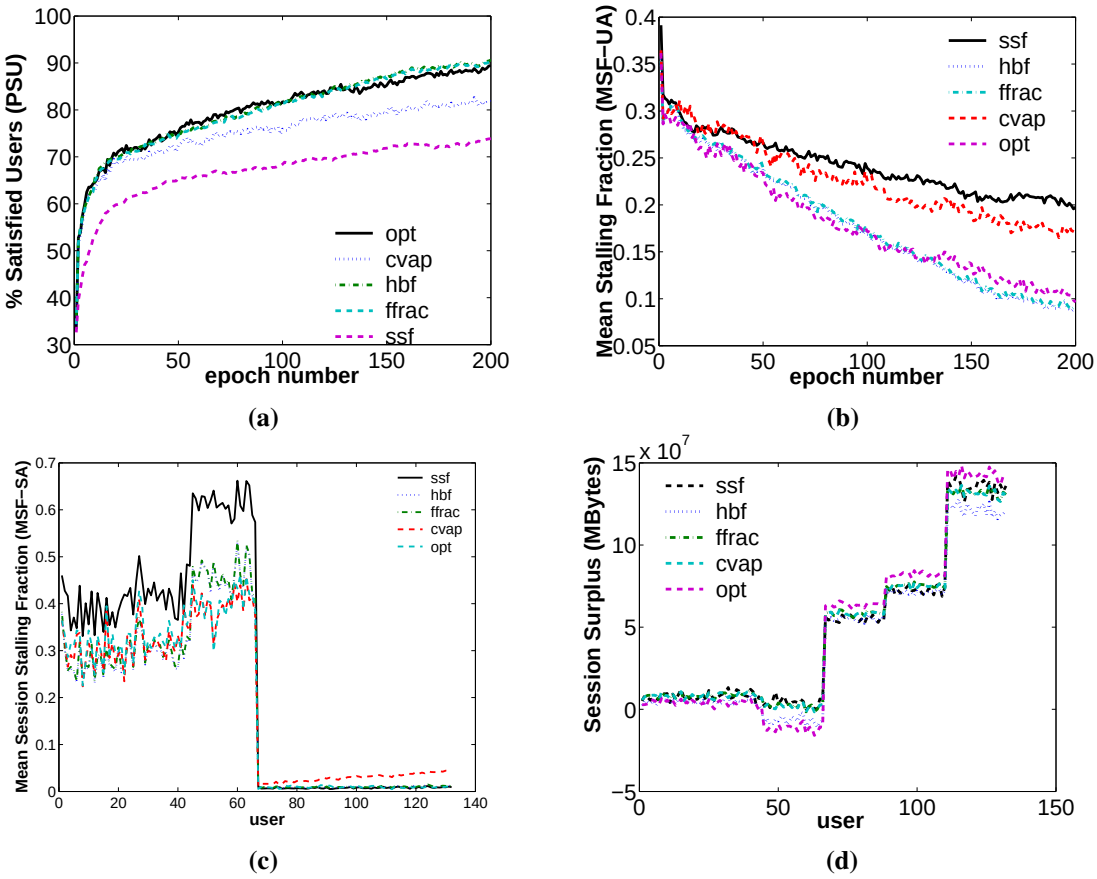

Fig. 3: Performance metrics for $T = 50$ in hotspot scenario 1 (hotspot-1) with stationary users,

per epoch. Stalling duration in an epoch is computed based on the content delivered in the epoch, SS buffer contents at the beginning of the epoch, and playout rate. A user is considered satisfied in an epoch, if their VoD flow does not stall. A flow stalls when its buffer is empty and it is served at less than its playout rate.

We compare cvap, our heuristics hbf and ffrac, and the common *strongest-signal first* (ssf) heuristic. The objective of the experiments is to evaluate the efficacy of the algorithms in balancing load across BSs and the impact of taking past service rates into account. A flow with a good channel might be served at higher than its playout rate, which can lead to its playout buffer building up at the SS. The allocation to such a flow might be lowered in a future epoch to help starved flows, if any. We compare results for $T = 50$ and $T = 1$ to study the impact of such lifetime QoE management that takes into account past history of service rates. Recall that T is the time constant used in updating average past rate. T provides a trade-off between throughput and latency. Results are averages of 15 runs. (The limitation was due to the high running time of cvap.)

B. Discussion of Results

All the algorithms exhibited comparable behavior under uniform load. This is due to the limited scope in improving performance using joint scheduling for the uniform case. Detailed results have hence been omitted for that case.

Simulation results for the first hotspot scenario (hotspot-1) for stationary users for $T = 50$ are plotted in Fig. 3. PSU and MSF averaged over all users (MSF-UA) are plotted by epoch in insets (a) and (b), respectively. In the plots, opt denotes the fractional solution returned by the convex solver used as part of cvap, and provides an upper bound on the optimal performance with respect to maximizing total utility. Both our heuristics perform as well as, and occasionally, marginally better than, opt, with respect to maximizing PSU and minimizing the stalling fraction. This suggests that additional constraints and techniques, such as limiting the allocation to flows with adequate buffers and good channels (even though PF scheduling may prefer them), to deviate slightly from PF allocation for improving QoE at the cost of system throughput may be helpful. Exploring such mechanisms is deferred for future work. cvap lags our heuristics by 10%, while ssf by around 20%. It should be noted that the performances of both the heuristics are very similar.

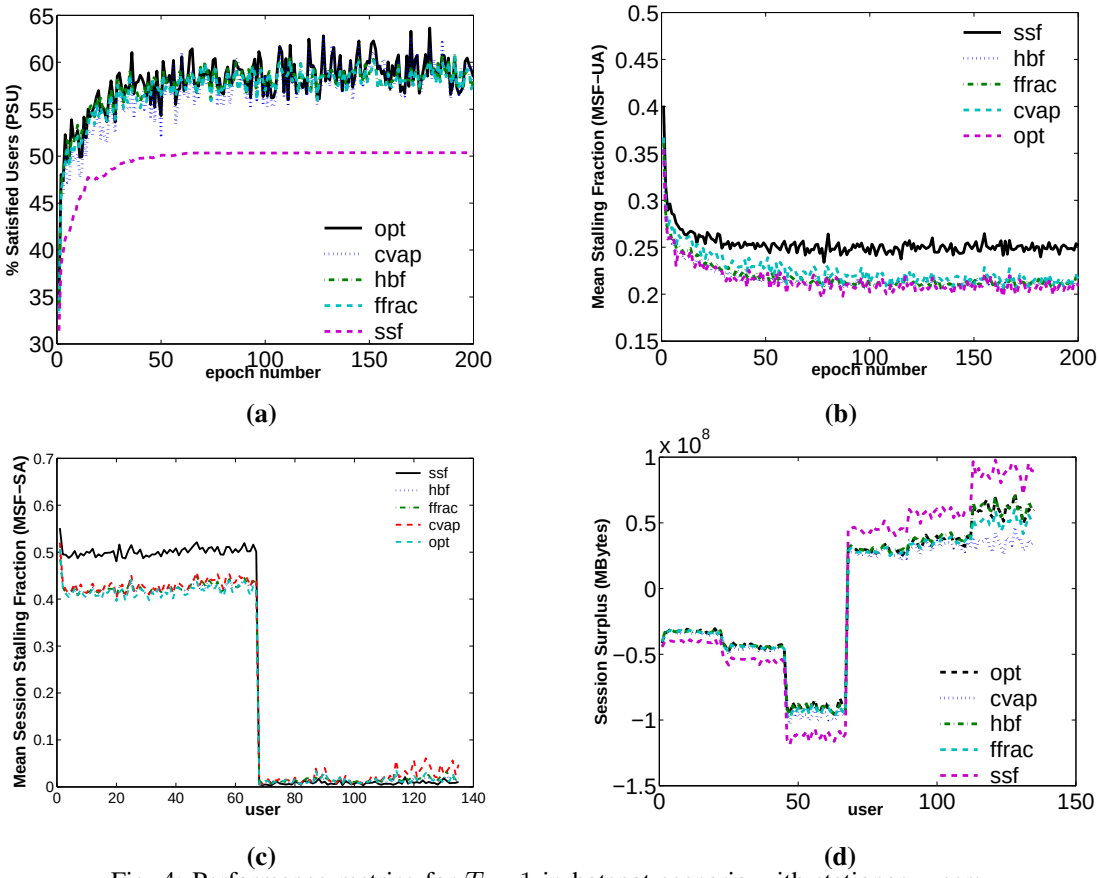

Fig. 4: Performance metrics for $T = 1$ in hotspot scenario with stationary users

Referring to inset (a), for all the algorithms, performance improves with epoch number. The "knee" of the curves is at about 20 epochs, that is, around 3.5 minutes. This suggests that, at least in our setup, pre-buffering for 3-5 minutes can be beneficial and help in improving satisfaction levels. Measuring QoE with pre-buffering is deferred for future work.

Inset (c) plots MSF averaged over the entire session by user (MSF-SA). Here, the first $N/2$ users are within the hotspot, and the remaining in the surrounding annulus. Within each class, users are arranged in an increasing order of their bandwidths. MSF-SA is negligible for the 50% of the users who are away from the central hotspot but quite substantial for the remaining users (who are in the hotspot). In the hotspot, stalling fraction is higher for users with higher bandwidth needs.

Inset (d) plots the surplus/deficit for the users at the end of 200 epochs. Users are arranged in the same order as in inset (c). Here, the first $N/2$ users, who are in the hotspot, are either underserved or have their needs met barely. The plots also indicate that there is good load balancing of the hotspot users. The remaining users have substantial surpluses in proportion to their bandwidth needs. As mentioned earlier, when meeting QoS is more important than maximizing system throughput, additional constraints may be added to the problem formulation to deviate from the PF allocation and allocate more resources to the starved flows and improve their QoE.

Fig. 4 plots the performance metrics for $T = 1$. Recall that when $T = 1$, past service history is ignored, which has the effect of lowering PSU by around 30% for all the algorithms. Note that the other metrics also deteriorate similarly to PSU. This observation underscores the importance of lifetime QoE management by taking past history into account, at least for VoD.

Insets (a)–(c) of Fig. 5 are for the second hotspot scenario, hotspot-2. In this case, all 132 users are placed in the hotspot. PSU with opt is 70%, cvap preforms close to opt, while the heuristics lag opt initially but improve gradually and catch up with opt. Plots of the surplus metric in inset (b) shows that at least under cvap and opt, load balancing of users is reasonable.

PSU for mobile users, initially distributed in the hotspot-1 mode, is shown in inset (d) of Fig. 5. Performance in this case is worse than for stationary users in hotspots by roughly 10% for all algorithms. This is because the system capacity is lowered due to mobility, whereas the load remains unchanged. It should also be noted that, in our simulations, the hotspot condition eases with time due to mobility. As mentioned earlier, since the epoch duration is halved when users are mobile, the time constant T is double to 100 to ensure that the duration of time in the past over which service is considered is the same as in stationary experiments.

IX. Conclusion

In this paper, we have considered coordinated, hierarchical scheduling in a network of WiMAX base stations, called

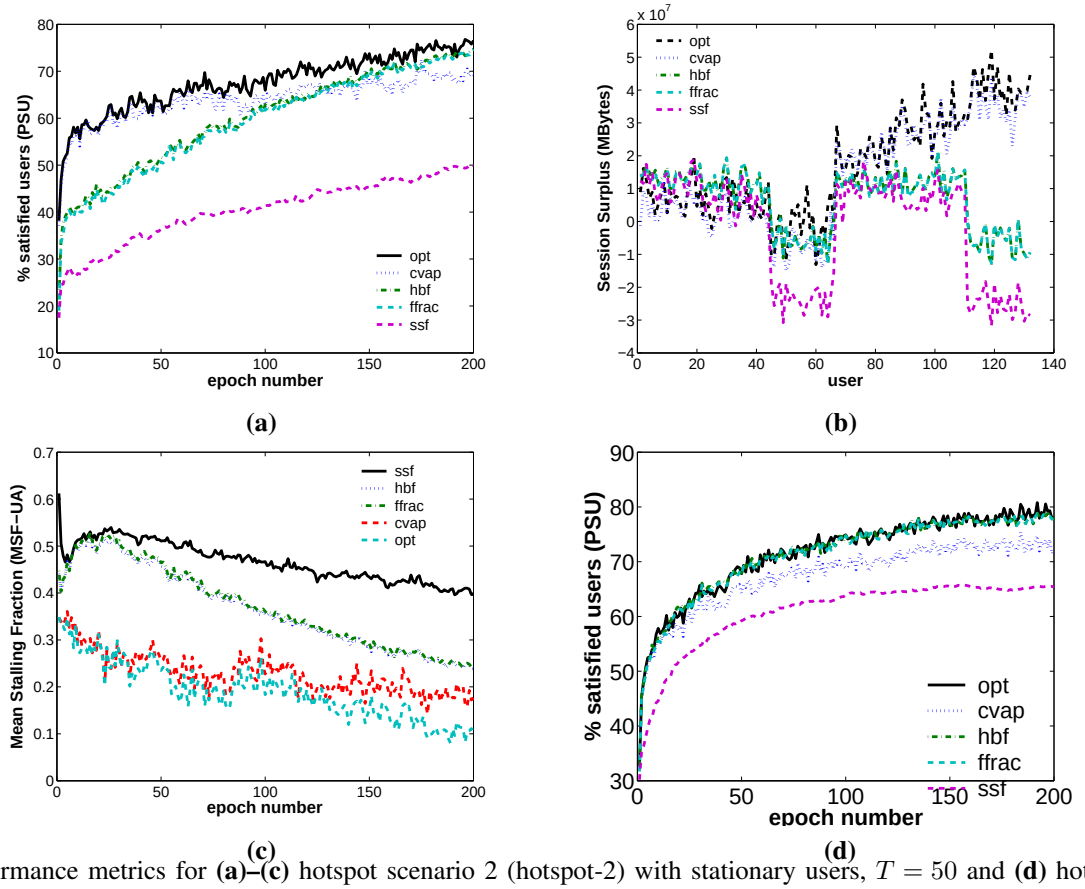

Fig. 5: Performance metrics for **(a)–(c)** hotspot scenario 2 (hotspot-2) with stationary users, $T = 50$ and **(d)** hotspot scenario 1 (hotspot-1) with mobile users, $T = 100$ (T is doubled since epoch duration is halved.

macro-scheduling of base stations (MSBS), for managing the QoE of VoD flows. Our utility maximization approach for guaranteeing long-term proportional fairness and managing lifetime QoE shows that MSBS can improve fairness for VoD flows significantly when the load distribution is asymmetric and that accounting for past service history during scheduling can increase the percentage of satisfied users, both under uniform and asymmetric loads.

As part of the QoE management problem, we showed that the network-wide PF scheduling problem is NP-hard in the strong sense. Since approximation algorithms previously proposed can require long running times, we have proposed efficient heuristics for the problem. We have also showed that the problem of assigning related flows to the same base station can be optimally solved using a hierarchical approach.

The work reported herein provides some guidelines for scheduling VoD flows. Using these guidelines to design efficient algorithms that can provide guaranteed services not only for VoD flows but real-time applications with more stringent requirements, such as video-conferencing, under WiMAX, remains a challenging next step.

REFERENCES

[1] IEEE, "802.16e-2005 IEEE standard for local and metropolitan area networks," http://standards.ieee.org/getieee802/802.16.html, 2005.

[2] WiMAX Forum, "WiMAX$^{\mathrm{TM}}$ applications for utilities: A case study of how WiMAX can enable vertical enterprise applications," http://www.m2mforum.com/it/images/stories/doc_pdf/Wirelesssrl.pdf, October 2008, WiMAX Forum White Paper.

[3] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *The J. of the Operational Research Society*, vol. 49, no. 3, pp. 237–252, March 1998.

[4] L. Li, M. Pal, and Y. Yang, "Proportional fairness in multi-rate wireless LANs," in *IEEE INFOCOM*, April 2008, pp. 1678–1686.

[5] WiMAX Forum, "WiMAX forum network architecture, stage 2, rel. 1.5, ver. 1-c," March 2009, WiMAX Forum Document Number DRAFT - T32-002-R015v01-C.

[6] T. Bu, L. Li, and R. Ramjee, "Generalized proportional fair scheduling in third generation wireless data networks," in *IEEE INFOCOM*, April 2006, pp. 1–12.

[7] A. Sang, X. Wang, M. Madihian, and R. Gitlin, "Coordinated load balancing, handoff/cell-site selection, and scheduling in multi-cell packet data systems," *Wireless Networks*, vol. 14, no. 1, pp. 103–120, 2008.

[8] S. Das, H. Viswanathan, and G. Rittenhouse, "Dynamic load balancing through coordinated scheduling in packet data systems," in *IEEE INFOCOM*, April 2003, pp. 786–796.

[9] V. Vukadinovic and G. Karlsson, "Video streaming in 3.5g: On throughput-delay performance of proportional fair scheduling," in *14th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, Sep 2006, pp. 393–400.

[10] X. Z. H. Lei, C. Fan and D. Yang, "Qos aware packet scheduling algorithm for ofdma systems," in *Vehicular Technology Conference*, Sep 2007, pp. 1877–1881.

[11] F. Hou, J. She, P. Ho, and X. Shen, "Performance analysis of weighted proportional fairness scheduling in ieee 802.16 networks," in *International Conference on Communications*, May 2008, pp. 3452–3456.

[12] M. Ergen, S. Coleri, and P. Varaiya, "Qos aware adaptive resource allocation techniques for fair scheduling in ofdma based broadband

wireless access systems," *IEEE Transactions on Broadcasting*, vol. 49, no. 4, pp. 362–370, Dec 2003.
[13] J. Gross, J. Klaue, H. Karl, and A. Wolisz, "Subcarrier allocation for variable bit rate video streams in wireless ofdm systems," in *Vehicular Technology Conference*, Oct 2003, pp. 2481–2485.
[14] J. Andrews, A. Ghosh, and R. Muhamed, *Fundamentals of WiMAX: Understanding Broadband Wireless Networking*. Prentice Hall PTR, March 2007.
[15] A. Jalali, R. Padovani, and R. Pankaj, "Data throughput of CDMA-HDR a high efficiency-high data rate personal communication wireless system," in *IEEE Vehicular Technology Conference*, 2000, pp. 1854–1858.
[16] H. Kim and Y. Han, "A proportional fair scheduling for multicarrier transmission systems," *IEEE Communications Letters*, vol. 9, no. 3, pp. 210–212, March 2005.
[17] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, March 2004.
[18] M. Garey and D. Johnson, *Computers and Intractability : a Guide to the Theory of NP-Completeness*. W. H. Freeman and company, NY, ch. 4.
[19] D. Shmoys and E. Tardos, "An approximation algorithm for the generalized assignment problem," *Mathematical Programming*, vol. 62, no. 3, pp. 461–474, 1993.
[20] WiMAX Forum, "WiMAX system evaluation methodology, ver. 2.1," July 2008.

APPENDIX
PROOF OF THEOREM 1

We begin by making two needed claims.

Claim 2. *Before the execution of the* **for** *statement in line 12 for the r^{th} time, where $1 \leq r \leq \eta+1$, the following holds. (When line (12) is executed for the $(\eta+1)^{\text{st}}$ time, the* **for** *loop terminates.)*

$$(\forall 1 \leq k \leq N :: (U_k(y_k) - U_k(y_k - 1) \geq U_\ell(y_\ell + 1) - U_\ell(y_\ell), \forall 1 \leq \ell \leq N)) \quad (19)$$

Proof: The proof is by induction on the executions r of the **for** statement in line 12.

Base Case: $r = 1$**:** The **while** loop of lines 5–11 terminates when the condition in line 8 holds. This condition implies that (19) holds when the **while** loop terminates and before the **for** statement is executed for the first time.

Induction Step: Assuming (19) holds for all $1 \leq r \leq i < \eta + 1$, we show that it holds for $r = i + 1$. For this, we show that (19) holds at the end of the i^{th} iteration of the **for** loop. Let the i^{th} allocation in the i^{th} iteration of the **for** loop be to flow p. The selection criterion for choosing p in line 13 ensures that at the end of the i^{th} iteration, (19) holds for $k = p$. Because (19) holds for all flows, *i.e.*, all k, at the beginning of the i^{th} iteration, and the allocation count is incremented for flow p only, it is easy to see that at the end of the i^{th} iteration, (19) holds for all the other flows also with respect to every flow other than perhaps p (that is, for all $k \neq p$, $\ell \neq p$ in (19)). To see that (19) holds for the other flows with respect to p as well, note that at at the end of the i^{th} iteration, by the induction hypothesis, $U_k(y_k) - U_k(y_k - 1) \geq U_p(y_p) - U_p(y_p - 1)$ holds for all $k \neq p$. Because U_p is strictly concave, $U_p(y_p) - U_p(y_p - 1) > U_p(y_p + 1) - U_p(y_p)$ holds, from which (19) follows for all $k \neq p$ with respect to p. ■

Claim 3. *If y_i is incremented (resp., decremented) in iteration p, then it is never decremented (resp., incremented) in a later iteration, q, for all i.*

Proof: We prove the claim that y_i cannot be decremented after it gets incremented in a prior iteration. The proof for the other case is similar. Suppose the claim does not hold. Let q be the first iteration after p that y_i is decremented in. Let $\hat{y}_\ell, y'_\ell$ be the number of slots allocated to each flow ℓ before the beginning of iterations p and q, respectively, *i.e.*, $y_\ell = \hat{y}_\ell$ and $y_\ell = y'_\ell$ hold at the beginning of iterations p and q, respectively. Since y_i is incremented in iteration p, by the criteria that choose k and k' (the flows whose allocations are incremented and decremented) in lines 9 and 10, respectively, we have

$$U_i(\hat{y}_i + 1) - U_i(\hat{y}_i) \geq U_\ell(\hat{y}_\ell + 1) - U_\ell(\hat{y}_\ell), \forall \ell \neq i. \quad (20)$$

In iteration q, in which y_i is decremented, let flow j's allocation, y_j, be incremented. Then, by the same argument as above, we have

$$U_j(y'_j + 1) - U_j(y'_j) \geq U_\ell(y'_\ell + 1) - U_\ell(y'_\ell), \forall \ell \neq j.$$

Further, since i's allocation is decremented by one, while that of j is incremented by one in iteration q, we must have (by the condition in line 8 and by the criteria selecting k and k' in lines 9 and 10)

$$U_j(y'_j+1) - U_j(y'_j) > U_i(y'_i = \hat{y}_i+1) - U_i(y'_i - 1 = \hat{y}_i). \quad (21)$$

We consider two cases based on whether the allocation to j is incremented or decremented or unchanged between iterations p and q.

Case 1: $\hat{y}_j \leq y'_j$**:** In this case, by the strict concavity of U_j, we have $U_j(\hat{y}_j + 1) - U_j(\hat{y}_j) \geq U_j(y'_j + 1) - U_j(y'_j) > U_i(\hat{y}_i + 1) - U_i(\hat{y}_i)$. (The second inequality is by (21).) This contradicts (20) (and the fact that y_i is incremented in the p^{th} iteration).

Case 2: $\hat{y}_j > y'_j$**:** This implies that y_j is decremented in one of the iterations $p, \ldots, q-1$. Let $p \leq q'' < q$ denote the iteration before q, when y_j was last decremented. Let $y_j = y''_j$ at the end of iteration q''. This implies that at the beginning of iteration q'', $y_j = y''_j + 1$ holds. Since flow i is not decremented in iterations $p+1, \ldots, q''$, $y_i = \hat{y}_i + 1$ holds at the beginning iteration q''. Since y_j, and not y_i, is decremented in iteration q'', by the criterion in line 10, the following holds at the beginning of iteration q'': $U_j(y''_j + 1) - U_j(y''_j) \leq U_i(\hat{y}_i + 1) - U_i(\hat{y}_i)$. Since $y''_j \leq y'_j$, by the concavity of U_j, we have $U_j(y'_j + 1) - U_j(y'_j) \leq U_j(y''_j + 1) - U_j(y''_j)$, which by the previous inequality implies $U_j(y'_j + 1) - U_j(y'_j) \leq U_i(\hat{y}_i + 1) - U_i(\hat{y}_i)$, which contradicts (21). ■

We are now ready for the optimality proof.

Theorem 1 *Algorithm $\mathcal{A}_{\text{SBS}}$ optimally solves* SBSRA *and has a worst-case time complexity of $O(N^2)$.*

Proof: We will first prove that Y is optimal for the case $\eta = 0$. In this case, $\sum_{i=1}^{N} y_i^* = S = \sum_{i=1}^{N} \lfloor y_i^* \rfloor$. This is only possible when $y_i^* = \lfloor y_i^* \rfloor$ for all $1 \leq i \leq N$, that is, when Y^* is integral. This in turn implies that the optimal solution to SBSRA-frac is integral and hence is also the optimal solution to SBSRA. Since the assignment $Y := Y^*$ in line 2 is unaltered until the algorithm terminates at line 4, Y is optimal.

For the case $\eta > 0$, we will prove correctness assuming that the **while** loop at line 5 terminates. (Termination is proved later.) Note that in each of the η iterations of the **for** loop in line 12, exactly one flow is allocated one slot, and so, in all, $\gamma + r - 1$ slots will have been allocated among the flows

before the **for** statement is executed for the r^{th} time, where $1 \leq r \leq \eta+1$. We next show that before the execution of the **for** statement for the r^{th} time, where $1 \leq r \leq \eta+1$, y is an optimal allocation of the initial $\gamma + r - 1$ slots.

Suppose not. Then there must exist some other allocation $y' = \langle y'_i \rangle \neq y$ of these slots such that $U(y') > U(y)$. Since $y' \neq y$ and $\sum_{k=1}^{N} y_k = \sum_{k=1}^{N} y'_k = \gamma + r - 1$, there must exist i and j such that $y'_i > y_i$ and $y'_j < y_j$. Let $i = \text{argmax}_p(y'_p - y_p)$, $y'_i = y_i + \alpha$, $j = \text{argmin}_p(y_p - y'_p)$, and $y'_j = y_j - \beta$. By the previous argument, $\alpha, \beta \in \mathbb{Z}_+$.

We consider the following three cases.

Case 1: $\alpha \geq 2$**:** From the strict concavity of U_i and U_j, we have $U_i(y'_i) - U_i(y'_i - 1) < U_i(y_i + 1) - U_i(y_i)$ (because $y'_i \geq y_i + 2$) and $U_j(y_j) - U_j(y_j - 1) \leq U_j(y'_j + 1) - U_j(y'_j)$ (because $y'_j \leq y_j - 1$). (19) in Claim 2 above implies $U_i(y_i + 1) - U_i(y_i) \leq U_j(y_j) - U_j(y_j - 1)$. By these three inequalities, we have $U_i(y'_i) + U_j(y'_j) < U_i(y'_i - 1) + U_j(y'_j + 1)$, This contradicts that y' is an optimal allocation of the initial $\gamma + r - 1$ slots.

Case 2: $\beta \geq 2$**:** Proof as in Case 1.

Case 3: $\alpha = 1$ **and** $\beta = 1$**:** By the condition of this case, for each i for which $y'_i = y_i + 1$ holds, there must be a distinct j, such that $y'_j = y_j - 1$ (because $\sum_{k=1}^{N} y_k = \sum_{k=1}^{N} y'_k = \gamma$). By Claim 2 (Ineq. (19)), we have $U_i(y_i + 1) - U_i(y_i) \leq U_j(y_j) - U_j(y_j - 1)$, that is $U_i(y'_i) + U_j(y'_j) \leq U_i(y_i) + U_j(y_j)$. Summing over all distinct pairs of flows as (i, j), we have $\sum_{\{(i,j)|y'_i = y_i+1, y'_j = y_j-1\}} U_i(y'_i) + U_j(y'_j) \leq U_i(y_i) + U_j(y_j)$. The utilities of the remaining flows remain unchanged, and hence, we have $U(y') \leq U(y)$, which is a contradiction to the assumption that $U(y') > U(y)$.

Thus, Y is an optimal allocation of the initial $\gamma + r - 1$ slots before the r^{th} execution, and an optimal allocation of $\gamma + \eta = S$ slots when the **for** loop terminates.

Algorithm complexity: We now turn to determining the complexity of the algorithm. Line 1 takes $O(N^2)$ time, while line 2 performs N assignments. Since η is at most $N - 1$, the **for** loop in lines 12–14 is executed $O(N)$ times. A binary MIN-HEAP may be used to determine the minimum in line 13. Construction of this heap at the beginning of the loop would require $O(N)$ time, while subsequent maintenance and $\min$ operations would take $O(\log(N))$ time. Thus, the complexity of the for loop is $O(N \log N)$.

To determine the algorithm's complexity, we are left with determining the number of iterations of the **while** loop and the complexity of each of its iterations. First note that termination is guaranteed. For this, note that the assignment $prev_i = \infty$ when $y_i = 0$ in line 7 ensures that $y_i \geq 0$ holds for each i at the end of the **while** loop. By Claim 3, this limits the total number of decrements, and hence, the iterations of the while loop, to γ. We now determine a more accurate bound.

Note that $\eta \leq N - 1$ slots are allocated to the flows in the **for** loop of lines 12–14. Let η_i denote the allocation to flow i in this loop. Now, suppose line 11 of the **while** loop is executed more than η times. Then, after iteration $\eta+1$, the total number of increments and decrements that line is each equal to $\eta+1$. By Claim 3, this implies that there exists a flow i, whose allocation is decremented by more than η_i, *i.e.*, $y_i \leq \lfloor y_i^* \rfloor - \eta_i - 1$ holds, after $\eta+1$ iterations. By Claim 3 again, y_i cannot be incremented in the while loop, and hence, $y_i \leq \lfloor y_i^* \rfloor - 1$ would hold when the **for** loop terminates. Therefore, since termination of the **while** loop is guaranteed, and hence, $\mathcal{A}_{\text{SBS}}$ determines an optimal solution, by Lemma 2, there does not exist a flow j with $y_j > \lfloor y_j^* \rfloor + 1$, at the end of either the while or the for loops. Note that in each iteration of the while loop (that does not terminate the loop), there is exactly one increment. At the beginning of the while loop, $y_i = \lfloor y_i^* \rfloor$ holds for all $1 \leq i \leq N$. By Claim 3, once y_i is incremented, it is never decremented, and vice versa. Thus, the number of iterations of the while loop may exceed η by at most $N - 1$. The total number of iterations is thus at most $\eta + N - 1 = O(N)$.

The $\max$ and $\min$ operations in lines 9 and 10 may be performed using a binary MAX-HEAP and a MIN-HEAP, respectively, to store the values of $next_i$ and $prev_i$, respectively, for all i. Besides the construction of the heaps at the beginning of the loop requiring $O(N)$ time, all other updates can be done in $O(\log N)$ time. Thus the total complexity of the while loop is $O(N \log N)$. Therefore, the worst-case time complexity of Algorithm $\mathcal{A}_{\text{SBS}}$ is $O(N^2)$(in line 1) $+ O(N \log N) = O(N^2)$. ■