

# Real-Time Video Multicast in WiMax Networks

Supratim Deb, Sharad Jaiswal, Kanthi Nagaraj  
 Bell Labs Research, India  
 {supratim,jsharad,kanthicn}@alcatel-lucent.com

**Abstract**—IEEE 802.16e WiMAX is a promising new technology for broadband access networks. Amongst the class of applications that can be supported is *real time video services* (such as IPTV, broadcast of live events etc.). These applications are bandwidth hungry and have stringent delay constraints. Thus, scalable support for such applications is a challenging problem.

To address this challenge, we consider a combination of approaches using multicast, layer encoded video and adaptive modulation of transmissions. Using these, we develop algorithms to ensure efficient, fair and timely delivery of video in WiMAX networks. The corresponding resource allocation problem is challenging because scheduling decisions (within a WiMAX base station) are performed in real-time across two dimensions, time and frequency. Moreover, combining layered video with appropriate modulation calls for novel MAC algorithms.

We model the multicast resource allocation problem in WiMAX and demonstrate this problem to be NP-hard. We present a fast greedy algorithm that is (i) provably within a constant approximation of the optimal solution (based on a metric that reflects video quality as perceived by the user), and (ii) performs within 87-95% of the optimal as demonstrated by realistic simulations. We also demonstrate that our algorithm offers a 25% improvement over a naive algorithm. Moreover, in terms of the average rate received by each user, our algorithm out-performs the naive algorithm by more than 50%.

## I. INTRODUCTION

Recently, the new IEEE 802.16e WiMAX standard has attracted a lot of attention. This standard describes a wireless broadband technology that can support mobile users. The physical layer used in 802.16e is Scalable OFDMA, which is resistant to multipath and has high spectral efficiency. Also, deployed WiMAX networks will typically have allocated large chunks of spectrum (10 – 20 MHz). Therefore, WiMAX systems are expected to offer high bandwidths (75Mbps given a 20Mhz spectrum) and cover large areas (few kms).

One potential deployment scenario for WiMAX is as access networks. Considering that WiMAX can support broadband speeds, a class of applications that can be supported is *real time video services*. This would include IPTV services, Internet streaming video (such as YouTube) and live telecast of sports and entertainment events.

Such applications are bandwidth hungry and have stringent delay constraints. Thus, a WiMAX network would require an efficient resource allocation system to meet these requirements, especially to scale with the number of users. Since there could be extensive shared interest across users for video content (especially, say, for IPTV channels and event broadcasts), *multicast* is a natural and efficient mechanism for real time video dissemination.

A WiMAX network (similar to cellular networks) would be divided into cells and further sub-divided into sectors. Within

a sector, the MAC layer of a WiMAX base station directs users to transmit on (on the uplink) and listen to (on the downlink) a particular time slot and frequency channel. Since a central entity schedules users, implementing multicast is easier in a WiMAX network: a base station simply needs to inform all users belonging to a multicast group to listen into the same time slot and frequency channel. Thus, WiMAX has natural support for multicast.

Scheduling (resource allocation) in a WiMAX base station MAC is done once every *scheduling frame*. A scheduling frame is made up of time slots on one axis, and frequency channels on the other. A time slot and frequency channel combination is one allocable resource, and is referred to as a *tile* (we will elaborate on this in a later section). Moreover, the WiMAX MAC also has the flexibility to specify both the modulation and coding that can be applied to a particular tile.

In a wireless network, the channel conditions can vary considerably across users. This poses interesting challenges for resource allocation to the base station MAC. A user with a good channel can receive data at a higher modulation (and thus, higher rate) compared to a user with a poor channel. Also, sending data at a higher modulation consumes less resources (tiles). Now, consider a multicast group with two users, *A* with a very good channel condition and *B* with a very poor channel. Clearly, if the real-time video is transmitted at a low modulation both *A* and *B* can access the video, but this will deprive *A* from potentially receiving better quality video. On the other hand, if the video is transmitted at high modulation, *A* can receive high quality video because of its good channel and this also uses fewer resources, but *B* will be starved from receiving any video at all. Thus the question is: how to provide good quality video to users experiencing good channels while providing a minimal quality to all other users, given a limited number of resources.

Our approach in this work is to utilize the notion of *layer encoded video*. The video stream is split into multiple layers and each successive layer enhances the quality of the video. The WiMAX MAC layer will then ensure that all users get access to the base layer (apart from ensuring a minimum rate, this also ensures that the play-back delay at the user meets the requirement of real-time video services.). The MAC also has to judiciously allocate additional resources for users that can receive the enhancement layers. This ensures that users with good channel conditions get superior video quality. Finally, the scheduling has to also ensure a level of *fairness* in the rates received by users across different multicast groups.

Thus, given this approach of using layered encoded video for video dissemination, the WiMAX MAC is faced with the following problem within any scheduling frame: *for any*

time slot within the scheduling frame, which layer of which multicast group should be transmitted at what modulation-coding level?

We illustrate the challenges in making this decision. Consider a base station and one multicast group with three users  $A$ ,  $B$ , and  $C$ . Suppose the base-station can simultaneously transmit at 10 different “frequencies”, and at one of the three rates: 8 kbps, 16 kbps, and 32 kbps<sup>1</sup>. Say, users  $A$ ,  $B$  and  $C$  can receive data at no more than 8, 16 and 32 kbps respectively, at any of the frequencies, because of their channel conditions. Also, suppose the video stream is split into multiple layers, each with a rate 32 kbps.

One solution to this scenario would be to transmit 2 video-layers, each using up 4 frequencies and 8 kbps/frequency (to ensure  $4 \times 8 = 32$  kbps required for the layer), and transmit 1 video-layer using 2 frequencies and 16 kbps/frequency. There are many other possibilities. Indeed, with  $K$  layers and  $m$  modulations there could be as many as  $m^K$  possibilities (and  $m^{KG}$  possibilities, if there are  $G$  multicast groups). Which among these is the best, can be decided based on an appropriate objective function, and we study this in detail in this paper. However, the point to note is that, even with one multicast group, there could be an exponential number of feasible solutions. Since, the MAC resource allocation has to be done in real-time (typically within a 5-10 ms scheduling frame), we need low-complexity and efficient MAC algorithms to perform the resource allocation.

In this paper, we model and present efficient algorithms for this problem. Specifically,

- 1) We model the resource allocation problem for supporting multicast for real time video services in WiMAX networks.
- 2) We prove the above problem to be NP-hard, and present a low-complexity, constant-factor approximation algorithm.
- 3) Through extensive simulations we demonstrate that our algorithm performs within 87-95% of the optimal (based on an utility function that reflects video quality as perceived by the user). Also, our approach significantly outperforms a naive heuristic.

This paper is organized as follows. In Section II we discuss related work. We give a brief background on certain features of WiMAX networks (that will be of use in this paper) and to the layered encoding approach in Section III. We then give a description of the system model, and a formulation of the problem we study in Section IV. We describe the algorithms for resource allocation in Section V. We present the simulation results in Section VI and conclude in Section VII.

## II. RELATED WORK

We now place our work in the context of other related work.

**Layer encoded video in wireline networks:** The notion of layered video has been studied extensively in the context

<sup>1</sup>How this simplistic model mildly represents WiMAX networks will be clear from our discussion later in the paper. In short, we are using “frequency” as a substitute for the resources available to a BS while making a scheduling decision. In WiMAX, these are a combination of time slots and frequency sub-channels.

of wireline networks [6], [15], [23]–[26], [28]. However, having to deal with adaptive modulation based on user channel conditions in a broadcast environment makes the problem substantially different in the case of WiMAX networks.

**Multicast in cellular data networks:** Resource allocation for multicast has recently been studied in the context of EVDO networks in [10]. However, resource allocation in WiMAX is very different from CDMA as decisions are made a) every scheduling frame b) across two dimensions, time-slots and sub-channels (we elaborate more on this later). Thus, the techniques and algorithms required in WiMAX systems are different. Also, the work in [10] is for a generic multicast service and does not account for the requirements of real-time video delivery. Another work that deals with scheduling for real-time video with broadcast/multicast in EVDO networks is [14]. However, they do not ensure any “fair” allocation of resources unlike our work or the work in [10].

**Scheduling in OFDMA:** The underlying physical layer technology used in WiMAX is OFDMA. Most of the considerable body of work on resource allocation in OFDMA/OFDM is in the context of unicast [7], [8], [11], [12], [16], [20], [21], [29]. Moreover, these works do not account for the specifications in WiMAX standard. In fact, they take an idealized dependence between received power and rate, based on the Shannon formula. As we shall see later, the discrete set of rate-power combinations allowed in WiMAX calls for modified algorithms at the MAC.

**Layer encoded video in wireless networks:** The work closest to ours is [9]. Here the authors have considered layered video for transmitting real-time unicast traffic in CDMA networks. Our work differs in three ways: first, we consider multicast, which makes the problem more complex; second, our work is targeted towards WiMAX networks with a very different resource allocation mechanism; and third, the key to providing variable video quality in our work is through the judicious use of layered video coupled with adaptive modulation and coding, which is not considered in [9].

Recently, a white paper about Qualcomm Inc’s Mediaflo [1] also proposed the notion of using two layers for video distribution over 3G wireless networks. However, the details of this mechanism are not publicly available. Although our work is intended for WiMAX networks, our work can be also be viewed as i) providing a solid algorithmic foundation for supporting layer encoded video with adaptive modulation and coding, and ii) doing a detailed performance evaluation to study the system performance.

## III. BACKGROUND: WiMAX AND LAYERED VIDEO

In the following we discuss features of WiMAX essential for the exposition of the ideas in this paper. For detailed background information on WiMAX networks, refer [2] and the IEEE 802.16e standards [3].

**WiMAX frames:** A scheduling decision in the WiMAX base-station MAC is done once per *frame*. A WiMAX frame consists of time slots in one dimension and frequency sub-

channels in the other<sup>2</sup>. A scheduling decision assigns a sub-channel and time slot combination to a WiMAX user. Once the decision has been made for the entire frame, it is broadcast by the WiMAX base station to all the users.

**Support for Real-time services:** The WiMAX MAC allows a user connection to select amongst five service classes: UGS (Unsolicited Grant Service), rtPS (Real Time Polling Service), nrtPS (non-Real Time Polling Service), ErtPS(Extended Real-Time Polling Service) and BE (Best Effort). Of these, rtPS is designed to support real-time video. However, the standard does not specify how a base station would allocate resources to support such requests (especially when there are many competing requests and limited resources). While our work provides a general framework to deploy real-time services in WiMAX, it can also be used to fill this gap.

**Distributed sub-carrier permutation:** Sub-channels in WiMAX consist of sub-carriers. Sub-channelization can be done in two modes 1) diversity permutation 2) contiguous permutation. In the diversity permutation mode, sub-carriers that form a sub-channel are chosen randomly from the entire frequency spectrum. In the contiguous permutation mode, a sub-channel is made up of adjacent sub-carriers. The diversity permutation mode has been recommended for use with mobile users or for users with low signal-to-interference-plus-noise ratio (SINR). The diversity permutation mode is also useful to average out inter cell interference since sub-carriers are selected randomly [17].

**Modulation-coding schemes and cross-layer adaption in WiMAX:** WiMAX allows three choices of modulation, namely, QPSK, 16-QAM, and 64-QAM along with three choices of FEC schemes for a total of six distinct permissible modulation-coding combinations. When a user's channel conditions are good (e.g., when its close to the base station) then it can potentially receive at high rates by opting for higher modulation (such as 64-QAM). Conversely, it must switch to a lower modulation when faced with poor channel conditions.<sup>3</sup> WiMAX MAC allows for changing modulation and coding per time slot and also across subchannels. For this to work, it requires the WiMAX PHY to collect information on the channel condition of every user based on feedback from the user (once at the beginning of every frame). The PHY then passes this information up to the WiMAX MAC.

**Layer encoded video:** When different users see different channel quality, a key challenge is to provide high quality transmissions to as many users while providing a minimal quality to all the users. Our approach in this work is to use the notion of layered encoding. In layer encoded video, the video stream is split into a base-layer video stream and multiple enhancement layer streams [26]. The base layer in a video frame is absolutely essential for reception of the video frame. The enhancement layers simply add to the quality of the video.

<sup>2</sup>Note that each sub-channel consists of 48 frequency sub-carriers; however, a sub-channel is the minimum amount of resource that can be allocated in the *frequency* dimension.

<sup>3</sup>This is because most applications have a target Bit Error Rate (BER) they can tolerate. The BER of a user depends on : transmitted power ( $P$ ) and channel gain of the user ( $h$ ). If  $Q$  is the minimum received power required to support the target BER for a particular modulation-coding combination, then we require  $P.h(t) \geq Q$ .

For a subscriber, the quality of the video improves with the number of layers it subscribes to. Suppose, the base layer of the video has a rate of 32 kbps and each enhancement layer has a rate of 64 kbps, then the video rate for a user subscribing to the first three enhancement layers would be  $32 + 3 \times 64 = 224$  kbps, and the video rate for a user subscribing to a single enhancement layer would be 96 kbps. Finally, an enhancement layer is useful only if all the other "lower" enhancement layers are subscribed to by the user.

## IV. SYSTEM MODEL AND PROBLEM FORMULATION

### A. Model and Notations

As mentioned earlier, scheduling in the WiMAX MAC occurs once in every scheduling frame (henceforth, simply referred to as a *frame* for simplicity). Each frame consists of time-slots and sub-channels. A time-slot and sub-channel combination (referred to as a *tile* in the rest of the paper) is the minimum allocable resource unit in the WiMAX MAC. Let  $R$  be the total number of tiles in a frame. As discussed before, the WiMAX MAC can choose one among multiple modulation-coding (MC) combinations. Let there be  $m$  MCs denoted by  $M_i, i = 1, 2, \dots, m$ , and let the rate of the MC  $M_i$  be  $r_{M_i}$ . Also, assume there are  $G$  multicast groups, with  $N_g$  users in the  $g^{\text{th}}$  multicast group.

We consider diversity mode sub-channelization. In this mode, the mobile-stations report the average CINR (carrier-to-interference-ratio) to the base-station [17]. The base-station MAC can use the CINR values of the different users to decide the modulation-coding combination to be used.<sup>4</sup> Note that the diversity sub-carrier permutation mode essentially implies that, for any user, channel attenuations are sub-channel independent.

We now describe our use of layer encoded video. Let the base layer video rate be  $\lambda_b$ . Also, let the rate of each enhancement layer be fixed and equal to  $\lambda$ , and let there be a maximum of  $K$  enhancement layers. In principle, for flexibility, enhancement layer could have different rates. However, by increasing  $K$ , the granularity of the enhancement layers can be made small enough to offset this problem. As a design decision, we also require that the layers, when fed into the MAC, have a constant rate. We believe this decision will have very little impact on the system performance. This is for the following reasons: (i) video traffic could already be encoded as constant bit rate (CBR) video (ii) or, if the encoding is variable bit rate (VBR), then existing smoothing techniques already ensure that the variability in the video rate is very small [13]. Thus, a small amount of buffering at the base station (to ensure a constant rate into the MAC) would result in a minimal additional play-back delay (order of milliseconds) at the user.

<sup>4</sup>This is not to be confused with the term Adaptive Modulation and Coding (AMC) defined in the WiMAX standards for use in the contiguous sub-carrier permutation mode. AMC refers to the capability of using different modulation-coding scheme for different parts of the *frequency* band. This does not apply to the diversity sub-carrier mode considered in this paper. Our use of the term adaptive modulation simply refers to the WiMAX MAC's ability to change modulation across tiles based on PHY layer feedback.

We also make an assumption that the transmit power across different sub-channels is fixed and cannot be changed by the MAC. While WiMAX standards allows for adaptive power allocation, we make this assumption for two reasons. First, power regulations vary across different regions, and this decision makes our design universally applicable. Second, most first generation WiMAX equipment does not support adaptive power.

For the convenience of the reader, we tabulate the notations used in the rest of the paper in Table I. Some notations will be clear as we formulate the problem through the section.

TABLE I  
NOTATIONS USED IN THE PAPER

$R$	Number of tiles in a frame
$m$	Number of modulation-coding combinations
$M_i$	$i^{\text{th}}$ modulation-coding combination
$r_{M_i}$	Rate of $M_i$ in information-bits/tiles
$G$	Number of multicast groups
$N_g$	Number of users in the $g^{\text{th}}$ multicast group
$n_{(g,i)}$	Number of users in the $g^{\text{th}}$ multicast group who can successfully receive transmissions using $M_i$
$\lambda_b$	Video base-layer rate in bits/frame
$\lambda$	Video enhancement-layer rate in bits/frame
$K$	Maximum number of enhancement layers
$s_i$	Number of tiles required in a scheduling-frame to transmit an enhancement layer using $M_i$
$m_{(g,k)}$	Index of modulation used for transmitting $k^{\text{th}}$ layer of $g^{\text{th}}$ multicast group

Given the system model and our assumption, we now formally describe our objectives and constraints and then present the problem formulation.

### B. Fairness based utility criterion

We now consider the criterion to decide the rates for different users. Our goal will be to ensure that rate allocation is ‘‘fair’’ across users. In this work, we use the notion of *proportional fairness* to allocate rates. This notion of fairness, first used by Kelly *et al.* [19], has also been used in cellular data networks, such as EVDO [5]. Intuitively, a proportional fair allocation ensures that the aggregate of proportional changes, between a proportional fair allocation and any other rate allocation, is always negative. Proportional fair allocation is also equivalent to maximizing the sum of the logarithm of the rates. Based on this observation, our goal in this work is to maximize the following utility function (where  $r_u$  is the rate allocated to user  $u$ ).

$$\text{Util}(\{u : r_u\}) = \sum_{g=1}^G \sum_{u=1}^{N_g} \log(r_u) \quad (1)$$

Throughout this work, we assume the log rate functions to be positive. This could be achieved by (i) expressing  $r_u$  in appropriate units (kbps or bits/frame etc.) (ii) modifying the utility functions as  $\log(1 + r_u)$ . None of these make any difference to the motivation behind considering a logarithmic utility functions and our results.

**Remark 1.** *Our goal is to ensure fair allocation within every scheduling frame. In other words, our framework finds a short-term fair allocation. On the other hand, the scheduling algorithm for EVDO ensures fair, long-term allocation of rates.*

*Since we are primarily concerned with real-time applications, the long term average rate is not meaningful. What is more important is the instantaneous rate, because real-time packets are valuable only if transmitted within a tolerable delay. Thus, for delay sensitive applications, ensuring short-run fairness is more important.*

### C. System constraints

We now consider the constraints imposed by our model.

**Packing constraint:** Let  $R_g$  be the number of tiles allocated to multicast group  $g$ . The packing constraint ensures that the total tiles allocated in a scheduling-frame cannot exceed  $R$ .

$$\sum_{g=1}^G R_g \leq R \quad (2)$$

**Real-time rate constraints:** As discussed earlier, in real-time transmissions, packets are not useful if not served in time. Although buffering at the end user can, to some extent, make up for the transmission delays, 1) it is not applicable for live transmissions, 2) the buffer sizes may not be sufficiently large (depending on the device). Thus, we enforce the constraint that if any  $k^{\text{th}}$  layer of a multicast group is to be transmitted to a user, then the number of tiles allocated in the frame is sufficient to support the rate of that layer. Thus, if modulation  $M_j$  is used to transmit an enhancement layer, then the number of tiles/scheduling-frame required by the layer is  $s_j = \lceil \lambda / r_{M_j} \rceil$  where  $r_{M_j}$  is in bits/tile. The same constraint obviously holds for the base layer.

Let  $z_{(g,k)}$  be a 0-1 variable that equals 1 only if the  $k^{\text{th}}$  layer of group  $g$  is transmitted. If the  $g^{\text{th}}$  multicast group is allocated  $R_g$  tiles, then the real-time constraint of transmitting the enhancement layers of the  $g^{\text{th}}$  multicast group can be expressed as

$$\sum_{k=1}^K z_{(g,k)} s_{m_{(g,k)}} \leq R_g. \quad (3)$$

**Minimum rate constraint:** We wish to ensure a minimum quality of transmission for each group. Hence, we require the transmission of the base-layer, or  $z_{(g,0)} = 1$ .

### D. Problem Statement

First note that, the received rate for a user receiving  $l$  enhancement layers is  $\lambda_b + l\lambda$ . Let  $n_{(g,i)}$  be the number of users in group  $g$  who can receive modulations  $M_i$ . Clearly,  $n_{(g,i)} - n_{(g,i+1)}$  is precisely the number of users who can receive only till modulation  $M_i$ , and each of these users can receive the enhancement layers that use  $M_i$  or a lower modulation (let there be  $l_i$  such enhancement layers). Then, by grouping the users for every group according to the highest modulations they can receive, we have

$$\sum_{g=1}^G \sum_{u=1}^{N_g} \log(r_u) = \sum_{g=1}^G \sum_{i=1}^m (n_{(g,i)} - n_{(g,i+1)}) \log(\lambda_b + l_i \lambda).$$

We now state the **WiMAX multicast resource allocation (WiMRA)** optimization problem.

**Given:**  $R, G, N_g, \lambda_b, \lambda, s_i, M_i, n_{(g,i)}$  ( $i = 1, 2, \dots, m$  and  $g = 1, 2, \dots, G$ ).

**To find:** For each  $(k, g)$  tuple, find if enhancement layer  $k$  of group  $g$  is to be transmitted (denoted by 0 – 1 variables  $z_{(g,k)}$ ) and if transmitted, the modulation  $M_{m_{(g,k)}}$ , to be used such that the following is optimized.

$$\begin{aligned} & \text{Maximize} \quad \sum_{g=1}^G \sum_{i=1}^m (n_{(g,i)} - n_{(g,i+1)}) \log(\lambda_b + \lambda l_i) \\ & \text{subject to (2) and (3) and } z_{g,0} = 1 \quad \forall g. \end{aligned}$$

### E. Hardness Results

We state the hardness of the WiMRA problem.

**Theorem 1.** *WiMRA is NP-hard even for  $G = 1$ .*

*Proof:* See Appendix A.

## V. APPROXIMATION ALGORITHMS FOR WiMRA

In this section, we present approximation algorithms for solving the WiMRA problem. We start by describing two key properties that should be satisfied by an optimal solution.

### A. Two key properties of an optimal solution

The first property states that higher layers should use a higher modulation.

**Lemma V.1.** *In the optimal solution of the WiMRA problem, if the  $j^{\text{th}}$  layer of a multicast group uses modulation  $M_i$  and the  $(j+1)^{\text{th}}$  layer uses modulation  $M_{i'}$ , then it must be that  $i' \geq i$ .*

*Proof:* Suppose not, ie, suppose  $i' < i$ . We will now arrive at a contradiction.

Suppose the utility of the optimal solution is  $C_o$ , and let  $C_m$  be the utility of another solution that uses modulation  $M_{i'}$  for the  $j^{\text{th}}$  layer, modulation  $M_i$  for the  $(j+1)^{\text{th}}$  layer, and for every other layer it uses the same modulation as the optimal solution. Now note that, in the modified solution, the number of users who get  $j$  layers increases by  $|N_{M_{i'}} - N_{M_i}|$  since  $i' < i$ . Also, the number of users who get exactly  $j+1$  layers in the modified solution is  $N_{M_{i'}}$  because  $N_{M_{i'}} \subseteq N_{M_i}$  (since  $i' < i$ ). Furthermore, the number of users who get all the other layers remains same as in the optimal solution. Thus it follows that  $C_m = C_o + |N_{M_{i'}} - N_{M_i}| \log[\lambda_b + j\lambda] > C_o$ . This violates the optimality of  $C_o$  and hence we have a contradiction. It follows that  $i' \geq i$ . ■

The next property is about the modulation that the base layer should use.

**Lemma V.2.** *In the optimal solution, the base layer of multicast group  $g$  has to be transmitted using modulation  $M_b$ , where*

$$b = \max_i \{i : n_{(g,i)} = N_g\}.$$

*Proof:* The proof follows from the fact 1) that every user has to access the base-layer 2) the number of tiles required to fit the base layer will diminish with increasing modulation. ■

In the rest of this section, we assume that the base layer is transmitted using modulation  $M_1$ . Our algorithms can be easily modified if this is not the case. More precisely, if the base layer is transmitted using  $M_b$  for  $b \geq 2$ , then, the choice of modulation for the enhancement layers would be from the restricted set  $\{M_b, M_{b+1}, \dots, M_m\}$ .

We continue with the description of our algorithm. The base layers of the multicast groups should be assigned modulation according to Lemma V.2. Let  $R' < R$  be the tiles remaining for the enhancement layer of the multicast groups. Suppose we split the  $R'$  tiles so that group  $g$  is allocated  $R_g$  tiles, and let  $C_g(R_g)$  be the resulting contribution to the total utility due to group  $g$ . Then the optimization problem can be posed as

$$\max_{\sum_g R_g \leq R'} C_g(R_g). \quad (4)$$

■ If  $C_g(R_g)$  is known, then the above is a variant of the knapsack problem. However, computing  $C_g(R_g)$  is the same problem as WiMRA when  $G = 1$  (which is NP-hard by Theorem 1). Therefore, as a first step we present algorithms for the WiMRA problem for the  $G = 1$  case, and in the subsequent subsection, we extend the solution to arbitrary  $G$ .

### B. Algorithms for the $G = 1$ case

We show how to use, Lemma V.1 to restate the WiMRA problem for  $G = 1$  in a simpler fashion. By Lemma V.1, between two successive layers, the higher one cannot be assigned a lower modulation. Also, recall that the base layer modulation can be decided using Lemma V.2, and we assume this modulation to be  $M_1$  wlog. Let the first  $x_1$  layers be assigned modulation  $M_1$ , let the next  $x_2$  layers be assigned modulation  $M_2$  and so forth (where,  $x_1, x_2, \dots$  are non-negative integers and can be zero). Also, let  $n_i$  be the number of users who can access modulation  $M_i$ . Then, we observe that the number of users who get exactly  $x_1$  layers equals  $n_1 - n_2$ . Similarly,  $n_2 - n_3$  users get exactly  $x_1 + x_2$  layers, and so on. The WiMRA problem can be restated as follows:

$$\begin{aligned} & \max_{\mathbf{x}} C(\mathbf{x}) = \sum_{i=1}^m (n_i - n_{i+1}) \log(\lambda_b + \lambda \sum_{j \leq i} x_j) \quad (5) \\ & \text{subject to, } \sum_{i=1}^m s_i x_i \leq R', \sum_{i=1}^m x_i \leq K. \end{aligned}$$

Here,  $\mathbf{x} = (x_1, x_2, x_3, \dots, x_n)$ , and  $R' = R - \lceil \lambda_b / r_{M_1} \rceil$  is the set of tiles left for the enhancement layers (after the base layer is catered to using  $M_1$ ), and  $n_{m+1} = 0$ .

Given this new formulation, and in light of the hardness result, the natural question is: what is the best possible approximation guarantee that can be achieved with polynomial complexity for the WiMRA problem?

We have developed an algorithm (that we omit due to lack of space) for the WiMRA problem with  $G = 1$  that has a  $(1 - \epsilon)$  approximation algorithm, with a complexity roughly  $O(mK^2 N_g / \epsilon)$ . Note that, once we extract the values of  $n_i$  via a pre-processing step, the input size of WiMRA is independent of  $N_g$  and hence this algorithm is technically not a polynomial time algorithm for WiMRA. Also, since the resource allocation should be performed in real-time, the

dependence on the number of users could be prohibitive in many scenarios. However, this algorithm is useful i) when the number of users is small, and ii) as a benchmark for performance evaluation of our low-complexity algorithms for WiMRA to be presented later in the section.

1) *A poly(number of users) time near optimal algorithm for  $G = 1$ :* We will rewrite the formulation in (5) in a slightly different manner first. Let us use the notation  $y_i = \sum_{j \leq i} x_j$  in (5). The constraint  $\sum_{i=1}^m s_i x_i$  can be rewritten in terms of  $y_i$ 's as  $\sum_{i=1}^m (s_i - s_{i+1}) y_i \leq R$  where  $s_{m+1} = 0$ . Note that,  $s_i - s_{i+1} > 0$  for all  $i$ . Letting  $w_i = s_i - s_{i+1}$ , we can rewrite problem (5) as follows.

$$\max_{\mathbf{y}} C(\mathbf{y}) = \sum_{i=1}^m (n_i - n_{i+1}) \log(\lambda_b + \lambda y_i) \quad (6)$$

subject to

$$\sum_{i=1}^m w_i y_i \leq R, \quad y_i \leq y_{i+1} \leq K \quad (7)$$

The key idea behind the algorithm is to quantize the utility function. More precisely, define the function  $f_\delta(y)$  over integers  $y$  as follows:

$$f_\delta(y) = \lfloor \frac{\log(\lambda_b + \lambda y)}{\delta} \rfloor \quad (8)$$

We shall find a polynomial algorithm when the logarithmic terms in utility function in (6) are substituted by  $f_\delta(y)$ 's. It is not hard to argue that this only results in small loss in the optimal utility. This fact is captured by the following lemma.

**Lemma V.3.** *Let  $\tilde{C}(\mathbf{y}) = \sum_{i=1}^m (n_i - n_{i+1}) \delta f_\delta(y_i)$ . Suppose  $\tilde{\mathbf{y}}$  maximize  $\tilde{C}(\mathbf{y})$  subject to the constraints described by (7), and let  $\mathbf{y}^*$  maximize  $C(\mathbf{y})$ . Then,*

$$C(\tilde{\mathbf{y}}) \geq C(\mathbf{y}^*) - N\delta.$$

*Proof:* See Appendix B ■

We now show how to solve the problem of maximizing  $\tilde{C}(\mathbf{y}) = \sum_{i=1}^m (n_i - n_{i+1}) \delta f_\delta(y_i)$  subject to the constraints described by (7). To this end, define  $S(l, j, k')$  as follows:

$$S(l, j, k') = \min \left\{ \sum_{i=1}^l w_i y_i : \sum_{i=1}^l (n_i - n_{i+1}) f_\delta(y_i) \geq j, \quad y_i \leq y_{i+1} \leq k' \right\}.$$

Note that, the  $\tilde{\mathbf{y}}^*$  maximizing  $\tilde{C}(\mathbf{y})$  can be expressed as

$$\tilde{C}(\tilde{\mathbf{y}}^*) = \delta \max_j \{ j : S(m, j, K) \leq R' \} \quad (9)$$

We can now write down the following recursion for solving  $S$ .

$$S(l, j, k') = \min_{y \leq k'} [w_l y + S(l-1, j - (n_l - n_{l-1}) f_\delta(y), y)] \quad (10)$$

The recursion given by (10) can be solved by building a dynamic programming table. The range of values of  $j$  is clearly  $j \leq N \log(\lambda_b + K\lambda)/\delta$ . Since,  $k'$  takes  $K$  distinct values, and there are  $m$  distinct modulations, the complexity of solving the recursion is  $O(mK^2N \log(\lambda_b + K\lambda)/\delta)$ . By choosing  $\delta$  small enough we arrive at an  $(1 - \epsilon)$  algorithm.

Specifically, if we choose  $\delta = \epsilon \log \lambda_b$ , we can use the fact that the optimal utility is at least  $N \log(\lambda_b)$  to arrive at the following result.

**Theorem 2.** *The problem of solving WiMRA for  $G = 1$  can be solved with an approximation guarantee of  $(1 - \epsilon)$  using the recursion (10) by setting  $\delta = \epsilon \log(\lambda_b)$ . Furthermore, the computation complexity of solving the recursion is  $O(mK^2N \log(\lambda_b + K\lambda)/(\epsilon \log \lambda_b))$ .*

The proof of the above follows from the preceding discussion. Clearly, the complexity of the algorithm could be prohibitive, as  $K$  and  $N$  both could be quite large in practice.

2) *A polynomial time greedy algorithm for  $G = 1$ :* We now describe an efficient algorithm for the WiMRA problem with a complexity  $O(mK)$ . This algorithm selects the modulations for successive layers in a greedy manner, as described soon. One motivation for applying a greedy algorithm is the concavity of the utility function in (5). Indeed, greedy algorithms perform well for many optimization problems with a concave or sub-modular utility function. There exist greedy algorithms for maximizing a sub-modular function subject to a single resource constraint [22]. However, this cannot be applied to our optimization problem which has two constraints.

We denote by  $\mathbf{e}_i$  the vector with 0 in all positions, and 1 in the  $i^{\text{th}}$  position. The greedy algorithm starts with  $\mathbf{x} = (0, 0, \dots, 0)$ , and it has three phases (refer to the Algorithm *GreedyOneGroup* for the explanation below):

*Phase-I:* We first remove from consideration all the infeasible modulations, i.e., the ones that cannot support a complete enhancement layer within  $R'$  tiles. Clearly, only if  $s_j \leq R'$ , then  $M_j$  is a feasible modulation. Let  $l_{\min}$  be the lowest modulation that is feasible.

*Phase-II:* In this phase (step 3-6), we increment the values of  $x_i$ 's. In each step we select and increment an  $x_i$  by 1 in a manner so that we get the maximum increase in utility with the least increase in LHS of the constraints  $\sum_i s_i x_i \leq R'$  and  $\sum_i x_i \leq K$ . More specifically, in each greedy step, we change  $\mathbf{x}$  to  $\mathbf{x} + \mathbf{e}_j$  so that  $(C(\mathbf{x} + \mathbf{e}_j) - C(\mathbf{x})) / (s_j + R'/K)$  is maximum among all  $j \geq l_{\min}$ .<sup>5</sup> Refer the *while* loop from Step-3 to Step-6 of *GreedyOneGroup*. The *while* loop exits when one of the constraints in (5) is violated.

*Phase-III:* Note that the last greedy step violates either  $\sum_i s_i x_i \leq R'$  or  $\sum_i x_i \leq K$ . Hence, we *undo* the last greedy step (Step-7 of *GreedyOneGroup*). However, the performance guarantee of the greedy steps can be only be provided without this *undo* operation (this will be clear from our analysis later in the section). However, it is not hard to show that the contribution of the last greedy step to the utility can only be  $C(\mathbf{e}_{l_{\min}})$ . Hence, we perform Step-8 to Step-10.

We now show that the greedy algorithm can provide an  $O(1)$  approximation guarantee. We first prove the following key property of the utility function.

<sup>5</sup>If we adapt the standard greedy approach to our problem, then we will choose  $\mathbf{e}_j$  to maximize  $(C(\mathbf{x} + \mathbf{e}_j) - C(\mathbf{x})) / s_j$ , but this fails to give any performance guarantee for our problem.

**Algorithm 1** *GreedyOneGroup*: Greedy Algo for  $G = 1$ 

- 1: Initialize  $\mathbf{x} = (0, 0, \dots, 0)$ .
- 2: Find  $l_{\min} = \min_j \{j : s_j \leq R'\}$
- 3: **while**  $(\sum_{i=1}^m s_i x_i \leq R'$  and  $\sum_{i=1}^m x_i \leq K)$  **do**
- 4: Find the new modulation to be added as follows:

$$\mathbf{u} = \arg \max_{\{e_j : j \geq l_{\min}\}} \left\{ \frac{C(\mathbf{x} + \mathbf{e}_j) - C(\mathbf{x})}{s_j + \frac{R'}{K}} \right\}$$

- 5:  $\mathbf{x} \leftarrow \mathbf{x} + \mathbf{u}$
- 6: **end while**
- 7:  $\mathbf{x} \leftarrow \mathbf{x} - \mathbf{u}$  {Delete the last added element}
- 8: **if**  $C(\mathbf{x}) \leq C(\mathbf{e}_{l_{\min}})$  **then**
- 9:  $\mathbf{x} \leftarrow \mathbf{e}_{l_{\min}}$
- 10: **end if**

**Lemma V.4.** For two vectors  $\tilde{\mathbf{x}}$  and  $\mathbf{x}$ ,

$$C(\tilde{\mathbf{x}}) - C(\mathbf{x}) \leq \sum_{i=1}^m (\tilde{x}_i - x_i)^+ [C(\mathbf{x} + \mathbf{e}_i) - C(\mathbf{x})],$$

where we use the notation  $x^+ = \max\{0, x\}$ .

*Proof:* See Appendix C. ■

We can now prove the following approximation guarantee of the greedy algorithm.

**Theorem 3.** The greedy algorithm for  $G = 1$  has complexity  $O(mK)$ , and it guarantees a solution that is within a factor  $(1 - e^{-1/2})/2$  of the optimal.

*Proof:* Define  $z_t$  as the utility differential between the optimal solution and the greedy approach after  $t$  iterations of the greedy algorithm. In other words, if  $\mathbf{x}^*$  is the optimal value of  $\mathbf{x}$  maximizing the utility, and  $\mathbf{x}(t)$  is the output of the greedy approach after  $t$  iterations,

$$z_t = C(\mathbf{x}^*) - C(\mathbf{x}(t)).$$

Let the element added in Step-4 of the greedy algorithm in the  $t^{\text{th}}$  iteration be  $\mathbf{e}_{j(t)}$ , i.e.,  $\mathbf{x}(t) = \mathbf{x}(t-1) + \mathbf{e}_{j(t)}$ . We have,

$$\begin{aligned} z_{t-1} &\leq \sum_i (x_i^* - x_i)^+ [C(\mathbf{x}(t-1) + \mathbf{e}_i) - C(\mathbf{x}(t-1))] \\ &\leq \sum_i (x_i^* - x_i)^+ [C(\mathbf{x}(t-1) + \mathbf{e}_{j(t)}) - C(\mathbf{x}(t-1))] \frac{s_i + \frac{R'}{K}}{s_{j(t)} + \frac{R'}{K}} \\ &\leq \frac{[C(\mathbf{x}(t)) - C(\mathbf{x}(t-1))]}{s_{j(t)} + \frac{R'}{K}} \sum_{i=1}^m [x_i^* (s_i + \frac{R'}{K})] \leq [z_{t-1} - z_t] \frac{2R'}{s_{j(t)} + \frac{R'}{K}}, \end{aligned}$$

where the first step follows from Lemma V.4 and the second step follows from Step-4 of the greedy algorithm. Suppose the *while loop* in Step-3 of the greedy algorithm is executed  $l$  times. It follows from the above that,

$$\begin{aligned} z_l &\leq z_{l-1} \left(1 - \frac{s_{j(l)} + \frac{R'}{K}}{2R'}\right) \leq z_0 \prod_{t=1}^l \left(1 - \frac{s_{j(t)} + \frac{R'}{K}}{2R'}\right) \\ &\leq z_0 \left(1 - \frac{1}{2lR'} \sum_{t=1}^l (s_{j(t)} + \frac{R'}{K})\right)^l, \end{aligned}$$

where the last step follows from the AM-GM inequality. Now note that when the *while loop* exits either  $\sum_{t=1}^l s_{j(t)} > R'$

or  $\sum_{t=1}^l R'/K > R'$  and thus  $\sum_{t=1}^l (s_{j(t)} + R'/K) > R'$ . It follows that

$$z_l \leq z_0 \left(1 - \frac{1}{2l}\right)^l \leq z_0 e^{-1/2}.$$

Since  $z_0 = C(\mathbf{x}^*)$ , we have  $C(\mathbf{x}(l)) \geq (1 - e^{-1/2})C(\mathbf{x}^*)$ . Now note that when the while loop exits, one or both of the constraints  $\sum_i s_i x_i \leq R'$ ,  $\sum_i x_i \leq K$  will be violated for the first time. We get around this by deleting the last element in Step-7. Thus the utility of the algorithm is  $C(\mathbf{x}_{l-1})$ . Note that,  $C(\mathbf{x}_l) - C(\mathbf{x}_{l-1}) = C(\mathbf{x}_{l-1} + \mathbf{e}_l) - C(\mathbf{x}_{l-1}) \leq C(\mathbf{e}_l) \leq C(\mathbf{e}_{l_{\min}})$ . Thus,

$$\max\{C(\mathbf{x}_{l-1}), C(\mathbf{e}_{l_{\min}})\} \geq \frac{C(\mathbf{x}_l)}{2} \geq (1 - e^{-1/2}) \frac{C(\mathbf{x}^*)}{2}$$

and the result follows, since the greedy algorithm has utility  $\max\{C(\mathbf{x}_{l-1}), C(\mathbf{e}_{l_{\min}})\}$  (see Step-8 of the greedy algorithm for  $G = 1$ ). ■

**Remark 2.** The greedy algorithm requires the values of  $n_i - n_{i+1}$ , which could be extracted through a pre-processing stage. These need to be recomputed only when new users join or leave, and existing users change their locations significantly. Also, this can be driven from the user side, i.e., only when a user's channel condition and as a consequence the set of modulations it can access, change substantially, it can send an update to the base station. Hence, the incremental complexity of updating  $n_i$  would be small.

### C. Algorithms for $G > 1$

Let  $R'$  be the tiles remaining after allocating tiles for base layer of every multicast group (using Lemma V.2). We now describe, how best to distribute  $R'$  tiles among the enhancement layers of the  $G$  multicast groups. We first describe a knapsack based algorithm that has a  $G^3$  dependence on the running time. Next, we present a simple greedy algorithms that has a  $G \log G$  dependence on the running time.

1) A knapsack based algorithm for  $G > 1$ : Define  $T_l(S)$  as follows.

$$T_l(S) = \max_{\sum_{g=1}^l R_g \leq S} \sum_{g=1}^l C_g(R_g)$$

Our goal is to find  $T_G(R')$ . It is easy to note that we can write the following recursion for  $T_l$ 's.

$$T_l(S) = \max_{\tilde{R}} [C_l(\tilde{R}) + T_{l-1}(S - \tilde{R})] \quad (11)$$

If the  $C_g(R_g)$ 's can be computed in polynomial time, then the above recursion can be used to devise an FPTAS for computation of  $T_G(R')$  using a standard knapsack approach [27]. However,  $C_g(R_g)$ 's can be only solved approximately in polynomial time using the algorithms in Section V-B. This just introduces an additional approximation factor in the performance of the knapsack algorithm for computing  $T_G(R')$ .

In the following, we simply outline the basic steps for computing  $T_G(R)$  for completeness. The approach is standard and is a simple extension of knapsack algorithm in [27]. The algorithm has the two following phases and takes  $\delta > 0$  as a parameter.:

- 1) **Phase-1:** Quantize  $C_g(R_g)$ 's for all  $g$  in steps of  $(1+\delta)$ . In other words, find

$$x_g(s) = \min_{R_g} \{R_g : C_g(R_g) \geq (1+\delta)^s\}$$

for all  $s = 1, 2, \dots, \log(N \log(\lambda_b + K\lambda)) / \log(1+\delta)$ . This can be done using a binary search. The binary search requires  $\log R$  computations of  $C_g$ 's which can be done approximately using the algorithms in Section V-B. Define the quantized versions of  $C_g$ 's as

$$C_g^Q(R_g) = (1+\delta)^s \text{ for } x_g(s) \leq R_g < x_g(s+1)$$

- 2) **Phase-2:** Quantize the function

$$T_l(S) = \max_{\sum_{g=1}^l R_g \leq S} C_g^Q(R_g) .$$

Essentially, we compute

$$z_l(s) = \min_R \{R : T_l(R) \geq (1+\delta)^s\}$$

using binary search where each iteration of the binary search involves solving the recursion

$$T_l(S) = \max_s [(1+\delta)^s + \tilde{T}_{l-1}(S - x_l(s))] .$$

In the above  $\tilde{T}_{l-1}$  is the quantized version of  $T_{l-1}$ . Thus, we can find  $T_G(R)$  by building a dynamic programming table.

If the complexity of computing  $C_g(R_g)$ 's is  $c$ , then the complexity of the first phase is  $O(Gc \log R \log(N \log(\lambda_b + K\lambda)) / \log(1+\delta))$ , and the complexity of the second phase is  $O(G \log R \log^2(GN \log(\lambda_b + K\lambda)) / \log^2(1+\delta))$ .

If the approximation error of computing  $T_{l-1}$  is  $\alpha$ , the approximation error for computing  $T_l$  is  $\alpha(1+\delta)$  since we quantize  $T_l$  in steps of  $(1+\delta)$ . Thus  $T_G(R)$  can be computed in polynomial time with an approximation factor of  $\alpha(1+\delta)^G$ . By setting  $(1+\delta)^G = 1+\epsilon$ , we arrive at the following result:

**Theorem 4.** *If the problem WiMRA for  $G = 1$  can be solved within an approximation factor of  $\alpha < 1$  and with a computational complexity of  $c$ , then the problem can be solved for a generic  $G$  within an approximation factor of  $\alpha/(1+\epsilon)$  and with computational complexity*

$$O\left(\frac{1}{\log(1+\epsilon)} Gc \log R \log(b_{\max}) + \frac{1}{\log^2(1+\epsilon)} G^3 \log R \log^2(Gb_{\max})\right)$$

where  $b_{\max} = \max\{g : N_g \log(\lambda_b + K\lambda)\}$ .

In practice, the value of  $G$  might be large and thus the  $G^3$  dependence could be prohibitive. In the following, we present a simple greedy algorithm for  $G > 1$ .

- 2) *A greedy algorithm for  $G > 1$ :* We now present a greedy algorithm for solving

$$\max_{\sum_{g=1}^G R_g \leq R'} \sum_{g=1}^G C_g(R_g) .$$

The greedy algorithm has two phases as described below (see Algorithm *GreedyManyGroups*).

*Phase-I:* In this phase, we quantize the functions  $C_g(R_g)$  in steps of  $(1+\epsilon)$  for any fixed  $\epsilon > 0$ . In other words, for each  $C_g(\cdot)$ , we find  $x_g(s)$  defined as

$$x_g(s) = \min_{R_g} \{R_g : C_g(R_g) \geq C_g(0)(1+\epsilon)^s\} \quad s = 1, 2, \dots, \quad (12)$$

where  $C_g(0) = N_g \log \lambda_b$ . This step (Step-2 of the algorithm) ensures that there are only few (polynomial in number) potential candidates for  $R_g$ 's. The quantized versions of  $C_g$ ,  $C_g^Q$ , is defined as

$$C_g^Q(R_g) = C_g(0)(1+\epsilon)^s \text{ for } x_g(s) \leq R_g < x_g(s+1)$$

The idea behind the greedy algorithm is to give tiles to the group that provides best value for tiles, *i.e.*, the group that provides maximum increase in the utility per unit of tile. To this end, for each point where  $C_g^Q(R_g)$  changes, we compute the following slopes:

$$\Delta_g(s, r) = \frac{C_g(0)((1+\epsilon)^r - (1+\epsilon)^s)}{x_g(r) - x_g(s)} . \quad (13)$$

Next, for each  $x_g(s)$  we obtain the current-best-tiles as that  $x_g(r)$ ,  $r > s$  for which  $\Delta_g(s, r)$  is maximum. To this end, we compute the following quantities for each  $s$  (Step-4):

$$\alpha_g(s) = \arg \max_{r>s} \Delta_g(s, r), \quad \beta_g(s) = \max_{r>s} \Delta_g(s, r) \quad (14)$$

Basically,  $\beta_g(s)$  is the best increase of utility per additional tile when the  $g^{\text{th}}$  group already has  $x_g(s)$  amount of tile.

*Phase-II:* In the second phase, we keep increasing the values of  $R_g$  in a greedy manner to provide the maximum increase in the utility function per additional number of tiles. This is described by *while* loop in Step-9 of the algorithm. Essentially, we maintain a list of candidate current-best-tiles for each multicast group based on the tiles they have got so far (the candidate current-value-tile are obtained from the  $\alpha_g(s)$ 's). At every iteration of the *while* loop, we pick the group whose candidate current-best-tiles provides the maximum increase in utility per unit of additional tile required. When the *while* loop is exited, we do away with the last iteration to ensure that the total amount of tile utilized remains less than  $R$ . Finally, in Step-17 to Step-19, we make sure that, if any group by itself can do better than greedy, then that group gets all the resources.

The complexity of the algorithm can be shown to scale as  $G \log G$  since complexity of sorting is  $\log G$ . The proof of the performance guarantee of the algorithm follows along the lines of the proof of Theorem 3 with certain differences. We skip the details and state the following main result.

**Theorem 5.** *If the problem WiMRA for  $G = 1$  can be solved within an approximation factor of  $\alpha < 1$  and with a computational complexity of  $c$ , then the greedy algorithm solves for generic  $G$  within an approximation factor of  $0.63\alpha/(1+\epsilon)^2$  and with computational complexity*

$$O\left(G \frac{\log b_{\max}}{\log(1+\epsilon)} (c \log R + \frac{\log b_{\max}}{\log(1+\epsilon)} + \log G)\right) ,$$

where  $b_{\max} = \max\{g : N_g \log(\lambda_b + K\lambda)\}$ .

*Proof:* See Appendix D. ■

---

**Algorithm 2 GreedyManyGroups: Greedy Algo for  $G > 1$** 


---

- 1: Assign the modulation for every base layer using Lemma V.2. Let  $R'$  be the number of remaining tiles.
  - 2: Obtain  $x_g(s)$  (as in (12)) to quantize  $C_g(\cdot)$ 's in steps of  $(1 + \epsilon)$ . This can be done using a binary search with  $\log R$  computations of  $C_g$ 's (use Algorithm *GreedyOneGroup* to compute  $C_g$ 's).
  - 3: Compute the slopes  $\Delta_g(s, r)$  as defined in (13).
  - 4: Compute  $\alpha_g(s), \beta_g(s)$  as defined in (14) for all  $g$  and all  $s$ .
  - 5: Initialize  $\text{cand}_g := \alpha_g(0)$  as current best tile  $\forall g$ .
  - 6: Compute  $\text{slope}_g = \beta_g(0)$
  - 7: Sort  $\{g : \text{slope}_g\}$ 's and create the sorted list *SortedSlopes*.
  - 8: Initialize  $R_g = 0$  for all  $g$ .
  - 9: **while**  $\sum_g R_g < R'$  **do**
  - 10:   Obtain  $j = \arg \max_g \{g : \text{slope}_g\}$  from *SortedSlopes*.
  - 11:    $R_j^{\text{old}} \leftarrow R_j, R_j \leftarrow x_j(\text{cand}_j)$
  - 12:    $\text{slope}_j \leftarrow \beta_j(\text{cand}_j), \text{cand}_j \leftarrow \alpha_j(\text{cand}_j)$
  - 13:   Re-sort *SortedSlopes* with new  $\text{slope}_j$ .
  - 14: **end while**
  - 15:  $R_j \leftarrow R_j^{\text{old}}$  {Do away with the last update in the *while* loop}
  - 16: For each  $g$ , compute  $R_g^{\text{max}} = \max_s \{x_g(s) : x_g(s) \leq R\}$ ,  $g' = \arg \max_g \{C_g^Q(R_g^{\text{max}})\}$ . {This finds the best contribution to the utility that could be possibly made by any single multicast group.}
  - 17: **if**  $C_{g'}^Q(R_g^{\text{max}}) > \sum_g C_g^Q(R_g)$  **then**
  - 18:    $R_g \leftarrow 0$  for all  $g, R_{g'} \leftarrow R_g^{\text{max}}$
  - 19: **end if**{If there is any group that alone can do better than the greedy, then give all the tiles to that group}
- 

## VI. EXPERIMENTAL VALIDATION

We now evaluate our proposed algorithms through detailed simulations, with realistic models. Our goals are to 1) demonstrate the benefits of combining layered video with cross-layer adaption of modulation, 2) demonstrate that our proposed greedy algorithm performs close to the optimal for most realistic scenarios, and 3) study the impact of varying certain design parameters (e.g., number of layers, frequency of channel feedback) on the system performance.

### A. Setup

We now describe the overall setup, PHY and MAC layer parameters and details of the input video format used in our simulations. We also provide a short description of the algorithms that are evaluated.

**Overall Setup.** We consider one base station, serving both mobile and stationary users, within a cell of radius 3 kms. The users are initially placed randomly across the cell. Each user joins one multicast group, that is selected randomly. In our experiments, we vary the number of multicast groups while keeping the total number of users constant at 100.

**PHY and MAC layer.** The PHY and MAC layer simulation parameters are primarily based on the specifications in [4]. The COST-231 Hata propagation model is used to estimate the propagation losses. For mobile users, short term Rayleigh fading, which depends on the speed of the user, is simulated using Jake's model. Time correlation for shadowing is simulated as per Gudmundson model. Thus, the channel is time varying, but is assumed to be constant for the duration of a scheduling frame. Further, we do not consider the channel frequency selective, because of the reasons listed in section III.

TABLE II  
SIMULATION PARAMETERS

Base station radius	3 Km
Carrier Frequency	3.5 GHz
Number of subcarriers	1024
Number of data subcarriers	720
Number of OFDM symbols per frame	48
Number of subcarriers per subchannel	48
Physical frame duration	5ms
Ratio of tiles divided as uplink:downlink	2:3
Number of Modulations	6
Log-Normal Shadowing SD ( $\sigma_s$ )	8 dB
Propagation Model	COST 231 Hata
Small-scale fading	Jake's model

We consider a 10 MHz spectrum in 3.5 GHz range for our simulations. The values of all the PHY and MAC layer parameters are listed in the following Table ??.

**Video Input.** We assume layer encoded video, with a base layer of 32 kbps. The total rate of all enhancement layers equals 512 kbps, and is divided equally across the number of enhancement layers used.

**Performance Metric.** We use the following metrics to compare the different algorithms: 1) the utility function  $\sum_{g=1}^G \sum_{u=1}^{N_g} \log(r_u)$ , described in (1), to reflect the aggregate user perceived video-quality 2) average data rate received by all users, over the simulation run.

**Algorithms evaluated.** We compare solutions computed from three different algorithms, 1) *Optimal Solution*: We have developed an algorithm with a  $G^3/\epsilon^2$  dependence for an approximation guarantee of  $(1 + \epsilon)$ . While this algorithm is too slow to be used in practice, by setting  $\epsilon = 0.01$  we can get a close approximation to the optimal solution. 2) *Greedy Heuristic*: Solution obtained by the *GreedyManyGroups* algorithm described in Section V. 3) *Naive Heuristic*: We use a simple heuristic which is agnostic to user channel conditions and the number of users in a multicast group. The heuristic makes a decision as follows a) available tiles within a frame are divided equally among the multicast groups, b) each multicast group sends the base layer and one enhancement layer, using a fixed modulation. The rate is decided by the number of available tiles.

### B. Results and Discussion

We now present the results from our simulations. All our results are averaged over 2000 scheduling frames each of 5 msec duration. We assume that 30% of users are mobile, traveling at a speed of 60 km/hr<sup>6</sup>.

In Figure 1(a), we plot the average number of users that can support a particular modulation-coding mechanism, in our simulations. The x-axis corresponds to the following modulation-coding levels: QPSK 1/2, QPSK 3/4, QAM 1/2, QAM 3/4, 64-QAM 2/3, 64-QAM 3/4. As explained earlier, supporting 64-QAM 3/4 would require the best channel conditions (and

<sup>6</sup>However, we assume that their positions do not change i.e. although the small scale fading varies with time, the path loss of the user remains constant during the course of simulation. This does not affect our results since the objective is to study the effect of small scale fading on our algorithms. This assumption is similar to the assumption in [18].

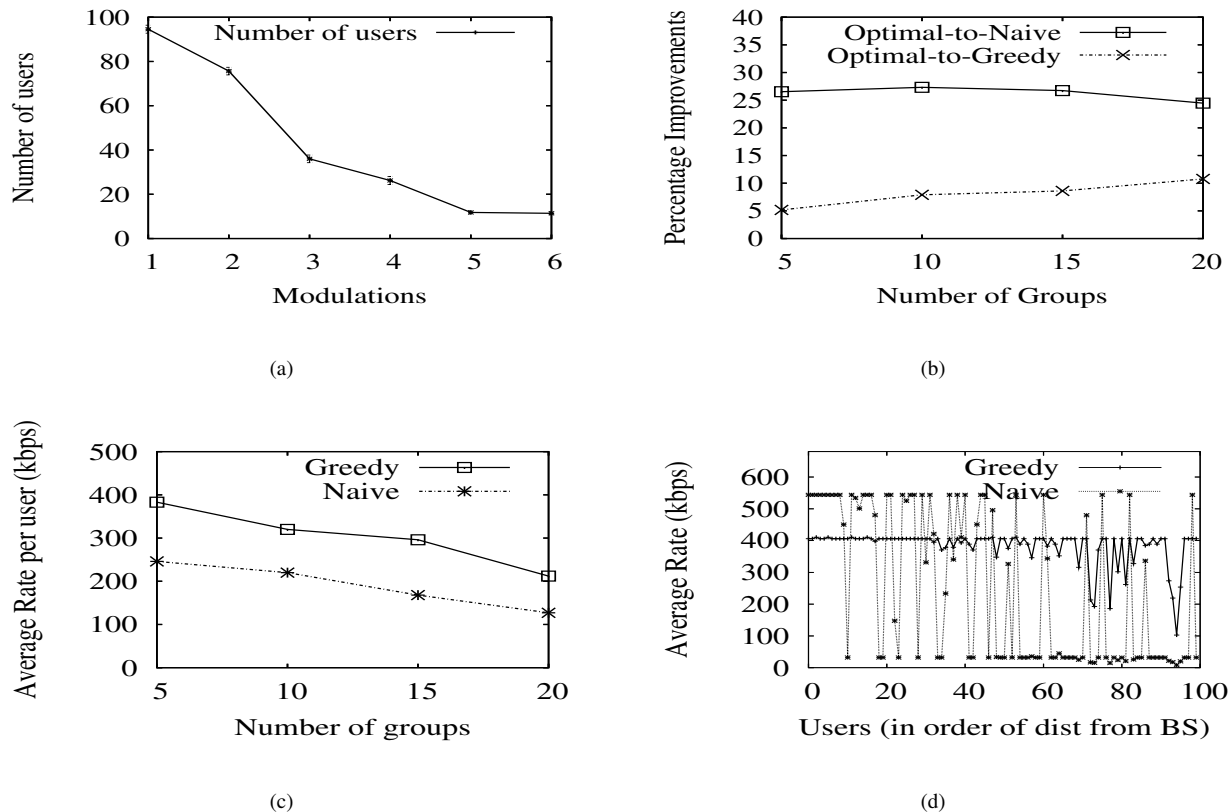


Fig. 1. a) Average number of users that can support a particular modulation-coding mechanism b) Comparing optimal with greedy and naive for user perceived video quality c) Comparing optimal with greedy and naive for average user rate. d) Average rate achieved by each user (users ordered on their distance from base station).

will provide the highest rates), relative to the other modulation-coding schemes. As can be observed, there is considerable diversity in the channel conditions of the users.

Figure 1(b) plots the perceived video quality (y-axis plots percentage improvement optimal would give over (i) greedy (ii) naive) while varying the number of multicast groups. We observe that the greedy algorithm always performs within 87–95% of the optimal solution and is around 25% better than the naive heuristic.

In Figure 1(c) we change the metric to average rate received per user (in kbps). As can be seen, with this metric the greedy algorithm outperforms the naive heuristic by more than 50%<sup>7</sup>.

The above two figures clearly establish that combining cross-layer adaption with layered video exploits the available resources much better, and reaps significant gains. We have observed similar results by varying the percentage of mobile users.

We also evaluate the fairness of the rates allocated by our greedy algorithm. Figure 1(d) plots each user’s average rate across 2000 frames, with 5 groups in the system. On the x-axis, the users are ordered based on their distance from base station. We can see that the greedy algorithm achieves an equitable distribution of rates across users, as opposed to the

naive algorithm where users close to the base station get high video rates, while users away from base station get lower rates.

We now study the impact on performance by varying certain design parameters.

*Number of layers:* The complexity of the greedy algorithm depends on the number of layers used for video transmission. Hence, we are interested in quantifying the benefits of using more layers. Our results show that, while increasing number of enhancement layers from 0 to 5 increases the utility metric by 25%, an increase from 5 to 10 layers gives an incremental benefit of just 1%. Hence, even by using a small number of layers we are ensured good performance.

*Feedback Frequency:* We assume that the base station knows the channel quality of each user. IEEE 802.16e supports a mechanism for user feedback on channel conditions, on a per frame basis. However, when the system scales to hundreds of users, it may not be possible for each user to report its observed SINR frequently. Hence, we study the impact of a mechanism in which an user gives feedback on channel conditions once every  $X$  frames. For these experiments, we assume that 90% of users are mobile, with a speed of 60 kmph. We found that even with very infrequent updates (once in 400 frames, or every 2 sec) the overall system utility reduced by around 3%. This indicates that the frequency of updates will not be a bottleneck for the system to scale to hundreds of users.

<sup>7</sup>The average rate received by an user dips with the increase in the number of groups. This is because, as the number of groups increases the available resources available to each group (and hence the number of enhancement layers it can send) decreases. This brings down the average user rate.

## VII. CONCLUDING REMARKS

In this work we have presented a technique for supporting real-time multicast services in WiMAX networks. Our main contributions are to (i) develop novel MAC algorithms that combine adaptive modulation and coding with layered video to optimize user perceived video quality, and (ii) to evaluate the system performance and the impact of different systems parameters through extensive simulations.

Our work assumes that the MAC can detect real-time video applications that require appropriate multicast scheduling support. In practice, such applications may be implemented over IP multicast, and a software layer is required that maps applications implemented over IP multicast to layer-2 WiMAX multicast. Clearly, our next goal would be to explore the feasibility of implementing this scheduling mechanism in the MAC of a real WiMAX base station.

We have also assumed that we are given the number of resources (within a scheduling frame) that can be used for supporting real-time services. In a real system, such applications would co-exist with other (delay tolerant) applications. A suitable allocation of resources across these two class of applications (with different requirements) is another area of future work.

Finally, while our algorithms do not directly apply to the *fractional frequency reuse* frequency planning mode in WiMAX, they can be modified to account for FFR. Detailed evaluation of the system under FFR is left as a future work.

## REFERENCES

- [1] <http://www.mediaflo.org/>.
- [2] [http://www.wimaxforum.org/technology/WiMAX\\_IMT\\_2000/](http://www.wimaxforum.org/technology/WiMAX_IMT_2000/).
- [3] IEEE 802.16e standard, <http://standards.ieee.org/getieee802/802.16.html>.
- [4] *Mobile WiMAX Part I: A Technical Overview and Performance Evaluation*, [http://www.wimaxforum.org/technology/WiMAX\\_IMT\\_2000](http://www.wimaxforum.org/technology/WiMAX_IMT_2000).
- [5] M. Andrews. A survey of scheduling theory in wireless data networks. In *IMA summer workshop on wireless communications*, 2005.
- [6] R. Cohen and H. Radha. Streaming fine-grained scalable video over packet based networks. In *GLOBECOM*, 2000.
- [7] M. Ergen, S. Coleri, and P. Varaiya. Qos aware adaptive resource allocation techniques for fair scheduling in OFDMA based broadband wireless access systems. *IEEE Trans. Broadcasting*, 49(4), 2003.
- [8] C. Wong et al. Multiuser OFDM with adaptive subcarrier, bit, and power allocation. *IEEE JSAC*, 17(10), 1999.
- [9] G. Su et al. Distortion management of real-time mpeg-4 video over downlink multicode cdma networks. In *IEEE ICC*, 2004.
- [10] H. Won et al. Multicast scheduling in cellular data networks. In *INFOCOM*, 2007.
- [11] J. Gross et al. Subcarrier allocation for variable bit rate video streams in wireless ofdm systems. In *VTC*, 2003.
- [12] J. Huang et al. Downlink scheduling and resource allocation for OFDM systems. In *CISS*, 2006.
- [13] J. Kangasharju et al. Distributing layered encoded video through caches. *IEEE Trans. Computers*, 51(6), 2002.
- [14] K. Kang et al. Dynamic scheduling for scalable media transmission over CDMA2000 1xEV-DO broadcast and multicast networks. In *IFIP Networking*, 2005.
- [15] M. Luby et al. Wave and equation based rate control using multicast round trip time. In *ACM SIGCOMM*, 2002.
- [16] R. Agrawal et al. Optimal scheduling for OFDMA systems. In *Asilomar*, 2006.
- [17] T. Kwon et al. Design and implementation of a simulator based on a cross-layer protocol between mac and phy layers in a wibro compatible ieee 802.16e OFDMA system. *IEEE Communications Magazine*, 43(12), 2005.
- [18] V. Mhatre et al. Impact of network load on forward link inter-cell interference in cellular data networks. In *IEEE Transactions on Wireless Communications, Volume 5*, 2006.

- [19] F. Kelly, A. Maulloo, and D. Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal Operational Research Society*, 49, 1998.
- [20] A. Khattab and K. Elsayed. Opportunistic scheduling of delay sensitive traffic in OFDMA-based wireless networks. In *WoWMOM*, 2006.
- [21] H. Kim and Y. Han. A proportional fair scheduling for multicarrier transmission systems. *IEEE Comm. Letters*, 9(3), 2005.
- [22] A. Krause and C. Guestrin. A note on the budgeted maximization of submodular functions. In *Technical Report CMU-CALD-05-103*, 2005.
- [23] A. Legout and E. Biersack. PLM: Fast convergence of cumulative layered multicast transmission schemes. In *ACM SIGMETRICS*, 2000.
- [24] X. Li, S. Paul, and M. Ammar. Layered video multicast with retransmission. In *INFOCOM*, 1998.
- [25] S. Mccanne, V. Jacobson, and M. Vetterli. Receiver driven layered multicast. In *ACM SIGCOMM*, 1996.
- [26] H. Radha, M. Schar, and Y. Chen. The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP. *IEEE Trans. Multimedia*, 3(1), 2001.
- [27] V. Vazirani. *Approximation Algorithms*. Springer, 2001.
- [28] B. Vickers, C. Albuquerque, and T. Suda. Adaptive multicast of multi-layered video: Rate-based and credit-based approaches. In *INFOCOM*, 1998.
- [29] G. Zhang. Subcarrier and bit allocation for real-time services in multiuser OFDM systems. In *IEEE ICC*, 2004.

## APPENDIX

### A. Proof of Theorem 1

*Proof:* We will reduce a certain knapsack problem to a version of WiMRA with  $G = 1$ . The maximum size unbounded knapsack problem goes as follows:

**(MSUK)** Given  $n$  items of sizes  $s_1, s_2, \dots, s_n$ , and a knapsack of size of  $B$ , find a multi-set of items (*i.e.*, each item can be picked multiple times) with maximum total size that can be fit into the knapsack. In other words, find non-negative integers  $x_i, 1 \leq i \leq n$  such that  $\sum_i s_i x_i \leq B$  and  $\sum_i s_i x_i$  is maximized.

The preceding problem is known to be NP-hard.

WLOG we will assume  $s_1 \leq s_2 \leq s_3 \dots \leq s_n$ . We will now reduce the preceding to our problem. In the setting of WiMRA, let there be  $B + s_1 \lambda_b / \lambda$  tiles (*i.e.*,  $R = B + s_1 \lambda_b / \lambda$ ),  $n$  modulations  $\{M_1, M_2, \dots, M_n\}$  and let the modulations be ordered so that  $r_{M_1} \geq r_{M_2} \geq \dots \geq r_{M_n}$ . We will choose the values of  $\lambda_b$  and  $\lambda$  later in the proof. Set the rate of the  $i^{\text{th}}$  modulation as  $r_{M_i} = \lambda / s_i$  for all  $1 \leq i \leq n$ . This implies that any enhancement layer that uses modulation  $M_i$  will occupy  $\lambda / r_{M_i} = s_i$  tiles. Let there be at least one user in the group who can receive data only at modulation  $M_1$ . This implies that the base layer has to be transmitted at modulation  $M_1$  and this will occupy  $\lambda_b / r_{M_1}$  tiles. Now, note that the problem WiMRA for  $G = 1$  is equivalent to choosing at most  $K$  modulations from the set  $\{M_1, M_2, \dots, M_n\}$ . Suppose, a solution chooses  $M_i$  for  $x_i$  distinct enhancement layers. Then, clearly, the constraint (2) is equivalent to  $s_1 \lambda_b / \lambda + \sum_{i=1}^n s_i x_i \leq B + s_1 \lambda_b / \lambda$  because the base layer occupies  $\lambda_b / R(M_1) = s_1 \lambda_b / \lambda$  tiles, and there are  $x_i$  enhancement layers that use modulation  $M_i$  and each enhancement layer that uses  $M_i$  occupies  $\lambda / r_{M_i} = s_i$  tiles. Thus  $\sum_{i=1}^n s_i x_i \leq B$ , and hence  $x_i$ 's are a feasible solution to the MSUK problem.

We shall now relate the objective functions of the two problems. Let  $N_{M_i}$  the set of users who can receive transmissions with modulation  $M_i$ . We set  $N_{M_i} \subseteq N_{M_{i-1}}$  for  $2 \leq i \leq n$

and let  $|N_{M_i}| = s_i$  for  $1 \leq i \leq n$ . Now, we note that in the optimal solution WiMRA, if the  $j^{\text{th}}$  layer uses modulation  $M_i$  and the  $(j+1)^{\text{th}}$  layer uses modulation  $M_{i'}$ , then it must be that  $i' \geq i$  (see Lemma V.1 for a proof of this).

With this fact, we will now write down the objective function of WiMRA. Note that the  $|N_{M_1} - N_{M_2}| = s_1 - s_2$  users get only till the  $x_1^{\text{th}}$  enhancement layer,  $|N_{M_2} - N_{M_3}| = s_1 - s_2$  users get till  $(x_1 + x_2)^{\text{th}}$  enhancement layers, and so on so forth. Thus,

$$\begin{aligned} & \text{Util}(\{i : x_i\}) \\ &= (s_1 - s_2) \log[\lambda_b + x_1 \lambda] + (s_2 - s_3) \log[\lambda_b + (x_1 + x_2) \lambda] \\ &+ (s_3 - s_4) \log[\lambda_b + (x_1 + x_2 + x_3) \lambda] \dots s_n \log[\lambda_b + \sum_{j \leq i} x_j \lambda] \\ &= (s_1 - s_n) \log(\lambda_b) + \sum_i (s_i - s_{i+1}) \log[1 + \sum_{j \leq i} x_j \frac{\lambda}{\lambda_b}] \end{aligned}$$

Thus our problem is equivalent to maximizing

$$C = \sum_i (s_i - s_{i+1}) \ln[1 + \sum_{j \leq i} x_j \frac{\lambda}{\lambda_b}] .$$

We have, by using the inequality  $x - x^2/2 \leq \ln(1+x) \leq x$  ( $x \leq 1$ ), for large enough  $\lambda_b$ ,

$$C \leq \frac{\lambda}{\lambda_b} s_i x_i$$

and

$$C \geq \frac{\lambda}{\lambda_b} s_i x_i - (\frac{n s_n \lambda}{\lambda_b})^2 .$$

Thus, by choosing  $n^2 s_n^2 \lambda / \lambda_b \leq 1/2$ , we have

$$\sum_i s_i x_i - \frac{1}{2} \leq \frac{\lambda_b}{\lambda} C \leq \sum_i s_i x_i .$$

Now suppose  $\tilde{x}_i$  maximizes  $(\lambda_b/\lambda)C$  and  $x_i^*$  maximizes  $\sum_i s_i x_i$  subject to  $\sum_i s_i x_i \leq B$ . Also,  $\tilde{x}_i$  is feasible solution to the MSUK problem. Thus,

$$\sum_i s_i x_i^* \leq \frac{\lambda_b}{\lambda} C(\{i : x_i^*\}) + \frac{1}{2} \leq \frac{\lambda_b}{\lambda} C(\{i : \tilde{x}_i\}) + \frac{1}{2} \leq \sum_i s_i \tilde{x}_i + \frac{1}{2} .$$

Since  $\sum_i s_i x_i^* \geq \sum_i s_i \tilde{x}_i$  and  $s_i, x_i^*, \tilde{x}_i$ 's are all integers  $\sum_i s_i x_i^* = \sum_i s_i \tilde{x}_i$ . Hence we have shown that MSUK can be solved using a version of WiMRA. Hence the result. ■

### B. Proof of Lemma V.3

*Proof:* Note that

$$\begin{aligned} \delta f_\delta(y) &\leq \log(\lambda + y\lambda) \leq \delta f_\delta(y) + \delta . \\ \Rightarrow \tilde{C}(y) &\leq C(y) \leq \tilde{C}(y) + N\delta \end{aligned}$$

Thus,

$$\begin{aligned} C(\mathbf{y}^*) &\leq \tilde{C}(\mathbf{y}^*) + \sum_{i=1}^m (n_i - n_{i+1})\delta \\ &\leq \tilde{C}(\mathbf{y}^*) + N\delta \\ &\leq \tilde{C}(\tilde{\mathbf{y}}) + N\delta \\ &\quad (\because \tilde{\mathbf{y}} \text{ maximizes } \tilde{C}(y)) \\ &\leq C(\tilde{\mathbf{y}}) + N\delta . \end{aligned}$$

Hence the result.

### C. Proof of Lemma V.4

*Proof:* In the following we use the notation  $\mathbf{x} \geq \mathbf{x}'$  to mean that  $x_i \geq x'_i$  for all  $1 \leq i \leq m$ .

Note that, it follows from the concavity of log function that

$$C(\mathbf{x} + \mathbf{e}_i) - C(\mathbf{x}) \geq C(\mathbf{x}' + \mathbf{e}_i) - C(\mathbf{x}') \quad (15)$$

for all  $i$  whenever  $\mathbf{x} \leq \mathbf{x}'$ .

Now, define  $\mathbf{x}'$  as  $x'_i = \max\{\tilde{x}_i, x_i\}$ ,  $1 \leq i \leq m$ . Clearly,  $C(\tilde{\mathbf{x}}) \leq C(\mathbf{x}')$ . Also,  $\mathbf{x} \leq \mathbf{x}'$ . Clearly, starting with  $\mathbf{x}$ , we can keep adding +1 appropriately to the different components of the  $m$ -dimensional vector till we have  $\mathbf{x}'$ . Suppose, we can add elements  $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_l$  successively to  $\mathbf{x}$  and get  $\mathbf{x}'$ , where each  $\mathbf{u}_i$  is from the set  $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m\}$ . Let  $\mathbf{U}_l = \sum_{j=1}^l \mathbf{u}_j$  and  $\mathbf{U}_0$  be the all-zero vector. Then we can write the telescopic sum

$$\begin{aligned} C(\mathbf{x}') - C(\mathbf{x}) &= \sum_{j=1}^l (C(\mathbf{x} + \mathbf{U}_j) - C(\mathbf{x} + \mathbf{U}_{j-1})) \\ &= \sum_{j=1}^l (C(\mathbf{x} + \mathbf{U}_{j-1} + \mathbf{u}_j) - C(\mathbf{x} + \mathbf{U}_{j-1})) \\ &\leq \sum_{j=1}^l (C(\mathbf{x} + \mathbf{u}_j) - C(\mathbf{x})) \quad (\text{from (15)}) \end{aligned}$$

Now note that exactly  $(x'_1 - x_1)$  of the  $\mathbf{u}_i$ 's are  $\mathbf{e}_1$ , exactly  $(x'_2 - x_2)$  of the  $\mathbf{u}_i$ 's are  $\mathbf{e}_2$ , exactly  $(x'_3 - x_3)$  of the  $\mathbf{u}_i$ 's are  $\mathbf{e}_3$ , forth. It thus follows that,

$$\begin{aligned} C(\mathbf{x}') - C(\mathbf{x}) &\leq \sum_{j=1}^l (C(\mathbf{x} + \mathbf{u}_j) - C(\mathbf{x})) \\ &\leq \sum_{i=1}^m (x'_i - x_i) (C(\mathbf{x} + \mathbf{e}_i) - C(\mathbf{x})) \end{aligned}$$

The result follows since  $C(\tilde{\mathbf{x}}) \leq C(\mathbf{x}')$ . ■

### D. Proof of Theorem 5

*Proof:* Let  $\{g : R_g^*\}$  denote the optimal solution to the problem of maximizing  $C(\{g : R_g\})$ . If  $C_g^Q(\cdot)$  is the quantized utility function, it is not hard to argue that,

$$\frac{C_g(R_g^*)}{1 + \epsilon} \leq C_g^Q(R_g^*) \leq C(R_g^*) .$$

In the greedy algorithm, let  $R_g^{(l)}$  be the amount of resource given to the  $g^{\text{th}}$  multicast group after  $l$  iterations of the *while* loop. Let  $z_l = \sum_g (C_g^Q(R_g) - C_g^Q(R_g^{(l)}))$  and let  $R^{(l)} = \sum_g R_g^{(l)}$ . Let  $j$  be the multicast group which gets additional

resource in the  $l^{\text{th}}$  iteration of the *while* loop. Note that,

$$\begin{aligned}
z_l &= \sum_g (C_g^Q(R_g^*) - C_g^Q(R_g^{(l)})) \\
&\leq \sum_{g: C_g^Q(R_g^*) > C_g^Q(R_g^{(l)})} \frac{C_g^Q(R_g^*) - C_g^Q(R_g^{(l)})}{R_g^* - R_g^{(l)}} (R_g^* - R_g^{(l)}) \\
&\leq \sum_{g: C_g^Q(R_g^*) > C_g^Q(R_g^{(l)})} \frac{C_j^Q(R_j^{(l+1)}) - C_j^Q(R_j^{(l)})}{R_j^{(l+1)} - R_j^{(l)}} (R_g^* - R_g^{(l)}) \\
&\leq (C_j^Q(R_j^{(l+1)}) - C_j^Q(R_j^{(l)})) \frac{\sum_g R_g^*}{R_j^{(l+1)} - R_j^{(l)}} \\
&\leq (z_l - z_{l+1}) \frac{R}{R^{(l+1)} - R^{(l)}}
\end{aligned}$$

It follows that

$$z_{l+1} \leq z_l \left(1 - \frac{R^{(l+1)} - R^{(l)}}{R}\right) \leq z_0 \prod_{t \leq l} \left(1 - \frac{R^{(t+1)} - R^{(t)}}{R}\right)$$

from which it follows that

$$\begin{aligned}
z_{l+1} &\leq z_0 \left(1 - \frac{1}{Rl} \sum_{t=0}^l (R^{(t+1)} - R^{(t)})\right)^l \\
&= z_0 \left(1 - \frac{1}{Rl} R^{(l+1)}\right)^l \leq z_0 e^{-\frac{R^{(l+1)}}{R}}
\end{aligned}$$

Thus, when the *while* loop is exited with  $R^{(l)} \geq R$ , the approximation factor of the greedy algorithm is  $1 - e^{-1} = 0.63$ . However, the last iteration of the *while* loop just about makes the total resource exceed  $R$ , and so we undo the last iteration of the *while* loop (line-14). This may reduce the utility by at most  $((1 + \epsilon)^{r+1} - (1 + \epsilon)^r) N_g \log \lambda_b = \epsilon (1 + \epsilon)^r N_g \log \lambda_b \leq \epsilon \sum_g C_g^Q(\tilde{R}_g)$  where  $\tilde{R}_g$  is the resource allocated to the  $g^{\text{th}}$  group at the end of greedy algorithm. The last inequality clearly follows from Step 18 of the greedy algorithm where we ensure that, if any single group can contribute to the utility more than the allocations in the *while* loop, then all the tiles are given to that single group. The results follows from this after some additional simplifications. ■