Adaptive Modulation for Finite Horizon Multicasting of Erasure-coded Data

Gek Hong Sim
University Carlos III, Madrid
Institute IMDEA Networks, Madrid
allyson.sim@imdea.org

Balaji Rengarajan
Institute IMDEA Networks, Madrid
balaji.rengarajan@imdea.org

Joerg Widmer
Institute IMDEA Networks, Madrid
joerg.widmer@imdea.org

***Abstract*—We design an adaptive modulation scheme to support opportunistic multicast scheduling in wireless networks. Whereas prior work optimizes capacity, we investigate the finite horizon problem where (once or repeatedly) a fixed number of packets has to be transmitted to a set of wireless receivers in the shortest amount of time – a common problem, e.g., for software updates or video multicast.**

In the finite horizon problem, the optimum coding and modulation schemes critically depend on the recent reception history of the receivers and require a fine balance between maximizing overall throughput and equalizing individual receiver throughput. We formulate a dynamic programming algorithm that optimally solves this scheduling problem. We then develop two low complexity heuristics that perform very close to the optimal solution and are suitable for practical online scheduling in base stations. We further analyze the performance of our algorithms by means of simulation in a wide range of wireless scenarios. They substantially outperform existing solutions based on throughput maximization or favoring the user with the worst channel, and we obtain a 35% performance improvement over the former and a 100% improvement over the latter in a scenario with Rayleigh fading.

I. Introduction

In recent years, multicasting data to mobile users (e.g., for the purpose of video streaming, video conferencing, IPTV, distribution of news and alerts, or application and operating system updates) has gained in popularity and importance. As an example, the most recent mobile network architecture LTE includes the evolved Multimedia Broadcast Multicast Service (eMBMS) specifically for the purpose of distributing data and mobile TV content in a cellular network. Since the amount of such traffic in cellular networks is increasing rapidly and wireless resources are usually scarce and costly, improving the efficiency of wireless multicast is of high practical relevance.

The most common method for wireless multicasting is broadcasting. The base station (BS) transmits at some fixed low rate or the rate supported by the worst receiver to ensure that all receivers are able to receive the multicast transmission. It exploits the wireless broadcast gain whereby a single transmission simultaneously serves all receivers. Opportunistic multicast scheduling (OMS) improves over plain broadcast by exploiting multiuser diversity [1]. Having the BS transmit at a rate higher than the broadcast rate to the subset of receivers that can receive at this rate improves overall throughput and minimizes broadcast delay [2]. The intuition is that in an environment with variable channels, receivers that are not served in the current slot since their channel conditions are bad will be served in later slots when their conditions improve, and thus over time all receivers will eventually receive all the data. Hence, there is a tradeoff between multicast gain and multiuser diversity gain. As an extreme case, the BS may even unicast data to the receiver that supports the highest rate, serving receivers one-by-one and entirely foregoing the broadcast gain for the largest possible multiuser diversity gain. Selecting the transmission rate and thus the subset of receivers to multicast to is a complex problem that has been the focus of a range of OMS algorithms [3]. To simplify the scheduling problem and improve performance when multicasting data, such algorithms often use erasure codes that ensure that with high probability, each packet received by a receiver is useful (unless the receiver has decoded all of the packets that exist in the system) [4], i.e., the identity of the received packets is unimportant. Fixed rate LDPC [5] or rateless LT or Raptor codes [6] are examples for such erasure codes that work well in practice and have good performance.

Most of the existing literature consider OMS algorithms for the infinite-horizon multicast problem, where the sender has an infinite number of packets to send and the goal of the optimization is to maximize throughput capacity. This setting is also a good approximation for the case of multicasting very large files. In practice, however, multicasting data with a size on the order of hundreds to several thousands of packets is much more common, particularly for mobile networks. Mobile apps and operating system updates often have a size of several tens of MB, corresponding to thousands of packets of size 1kB. When streaming video, it is common to apply erasure coding to blocks consisting of one or several groups of pictures (GOP) [7], where a GOP usually consists of a few hundreds of packets, depending on the video rate. Since erasure coded blocks can only be decoded after they have been fully received, coding over larger blocks of video data would unnecessarily increase the playout delay of the video.

In this paper we therefore consider the finite horizon multicast problem, where a fixed amount of erasure coded data has to be delivered to a set of wireless receivers. Whenever the BS transmits a packet, the optimization algorithm has to select a suitable modulation and coding scheme (MCS). The MCS

978-1-4673-5494-3/13/$31.00

determines the amount of data transmitted per time slot and thus the data rate. At the same time, the MCS influences the packet loss probability, where more robust MCSs that transport less data are more likely to be decodable at a receiver. (We assume that the transmit power is fixed.) Our main objective is to minimize the completion time, i.e., the time needed for all receivers to successfully receive the data.

The finite horizon multicast problem is inherently more complex than the infinite horizon counterpart. When multicasting an infinite amount of data among a homogeneous group of receivers (i.e., with the same average channel conditions and receive rates), the optimum tradeoff between multiuser diversity and multicast gain only depends on the number of receivers and their current channel conditions. In expectation, differences in the amount of data received by the different receivers will even out over time and therefore do not have to be taken into account. In contrast, the optimum decision in the finite horizon case also depends on the amount of data received thus far by each receiver (or, more accurately, on the amount of data each receiver still needs to obtain in order to decode the full block of data and thus complete). Intuitively, in case a receiver is lagging behind but many other receivers are also still far from completing, the lagging receiver may catch up by itself and jointly maximizing throughput for all these receivers may be the optimum decision. If, however, all other receivers are close to completion, optimizing the MCS (and hence the transmit rate) for the lagging receiver only may be the optimum choice to minimize overall completion time, given that all other receivers are likely to complete before the lagging receiver in any case.

The main contributions of our paper are as follows:

- We formalize the finite horizon OMS problem and propose a dynamic programming (*Dyn-Prog*) based solution, that optimally adapts the MCS to minimize the *completion time*, the time at which all receivers successfully receive the required amount of data.
- The high complexity of *Dyn-Prog* renders this approach unsuitable for many practical scenarios. We therefore propose a simple state-based heuristic that selects the MCS that maximizes the instantaneous throughput for the receiver with the minimum number of packets which thus had the lowest throughput so far (called *Max-min*).
- We further design a slightly more complex adaptive algorithm that selects the MCS that results in an expected system state (given by numbers of packets received by the different receivers) that has the lowest expected completion time. Due to the complexity of accurately calculating completion time, we use a weighted Euclidean distance metric (called weighted completion time or *Weighted-CT*). It measures the distance between the different possible future states and the final state where all receivers completed, with weights based on average throughput estimates of the receivers.
- We compare the performance of our two low-complexity heuristics to the optimal *Dyn-Prog* solution as well as to existing approaches that greedily maximize the throughput for all receivers and a broadcasting scheme that always transmits to all receivers. We analyze scenarios with homogeneous and heterogeneous receiver sets under a basic multi-state channel model as well as Rayleigh fading. Under Rayleigh fading, the *Max-min* algorithm provides a performance gain of 20% over the throughput maximization scheme and a gain of 40% over the broadcasting scheme. At a very slight increase in complexity, the *Weighted-CT* heuristics performs very close to the optimal *Dyn-Prog* strategy in almost all scenarios, achieving performance gains of 30% and 100% over the throughput maximization and broadcast scheme, respectively. For both our heuristics, the gains that can be obtained under the basic multi-state channel model are even larger.

Our paper is organized as follows. In Section II, we review existing work on opportunistic multicast scheduling. Section III provides an overview of our system model including the channel model and the MCS dependent packet loss model. In Section IV, we detail the optimal scheduling algorithm based on a dynamic programming formulation. To address the problem of state space explosion and high complexity of the dynamic programming solution, we propose two low-complexity heuristics in Section V. We also discuss some basic scenarios to provide an intuition into how the heuristics trade off receiver throughput depending on the current state of the system. Simulation results that compare the relative performance of the different algorithms are presented in Section VI and Section VII concludes the paper.

II. Related Work

The idea of OMS was pioneered by Gopala and Gamal [1] who studied the tradeoff between multiuser diversity and multicast gain. They studied the performance of three different scheduling mechanisms that adapt the transmit rate to the user with the best channel, the worst channel, and the median channel, respectively. In their follow-up paper [8], they analyzed the performance achieved by serving a fixed fraction of users. This restriction is relaxed in [4], where the authors show that dynamic selection ratios that select more than 50% of the users can achieve higher throughput. Furthermore, a throughput maximizing scheme for erasure-coded multicast (F-OMS) is presented, where the user selection ratio depends only on the set of multicast users in the system.

The authors of [9] propose algorithms with a static selection ratio (fixed for all transmissions) and a dynamic selection ratio (adapted to the instantaneous channels at each transmission) that maximize overall throughput. In [10], the authors extend their work of [9] from homogeneous to heterogeneous scenarios, composed of different groups of homogeneous users. A similar optimization algorithm for multicast throughput maximization in homogeneous to heterogeneous networks is proposed in [11]. While all of these works target the infinite horizon case, in [2] the authors do consider scenarios with a finite number of multicast packets. Using extreme value theory, they derive the optimal selection ratio for each transmission that minimizes completion time, i.e., the time period during which each user is selected often enough to receive the whole block of data. In contrast to our work, their optimization algorithm does not consider the state of the receivers in terms of number of packets received.

All of the above papers use a simple outage based channel model, where packet errors are deterministic. Receivers with channel conditions better than a certain threshold are guaranteed to receive the packet and all others are guaranteed to lose the packet. In real wireless systems, packet errors are much more random and depend on noise and interference. In our model, we explicitly take the relationship between the channel conditions, the chosen MCS, and the probability of error into account. Furthermore, all of the above papers – except [2] – focus on the infinite-horizon scenario and thus the performance of the proposed algorithms is sub-optimal in the finite-horizon problem we consider here.

The problem of minimizing the overall delay for all users to receive a certain number of packets is studied in [12] through a dynamic programming approach. This work does not consider erasure coding over a larger block of data but repeatedly multicasts a single packet until each receiver has obtained it. The BS then multicasts the next packet in the same manner, and so on. The goal of the optimization algorithm is therefore to minimize the number of transmission required to multicast a single packet to all receivers, and the state of the system is the number of receivers that did not yet receive the packet. The algorithm adapts its decision to the changes in the set of users that still need to receive the packet and maximizes the throughput for those users. The approach is mainly suitable for a single homogeneous group of users, since its complexity increases exponentially with the number of user groups in heterogeneous scenarios. The method of multicasting a single packet repeatedly is less efficient than multicasting blocks of erasure coded packets as is done above.

The most basic scheme against which we compare our proposed algorithms is the *Broadcast* algorithm (called LCG user rate in [3]), where the transmission rate is limited by the receiver that currently has the worst channel. This scheme ensures successful transmission to all receivers at all times but may sacrifice a lot of throughput when channels are highly variable. We further compare against a scheme called *Greedy* that optimizes the selection ratio at each transmission opportunistically based on the current channel states of all receivers so as to maximize total throughput. This mechanism has a performance that is indicative of the different selection ratio based throughput maximization algorithms above.

For reliable broadcasting in multicast applications, we use erasure coding as in [4], [10], [2], [11]. The main difference that distinguishes our work is the optimization of completion time in finite horizon multicasting. Further, we do not assume deterministic packet errors, i.e., channel outage, but model the packet error rate based on the channel conditions and the chosen MCS. We evaluate the proposed algorithm under realistic channel assumptions that include the effects of small-scale Rayleigh fading.

III. System Model

We model the system as a time-slotted broadcasting system with a single Base Station (BS) and N mobile users scattered within the coverage radius of the cell. Each user must receive a block of data of B bits, called the block size, and we assume that, thanks to erasure coding, each packet transmitted by the BS and received by a mobile user is useful if the user has received less than B bits In case multiple blocks of data are to be transmitted in succession, the BS will start transmitting the next data block only after all the receivers fully received the current block.

A time slot is of fixed duration. Thus, the BS broadcasts the same number of symbols, which, depending on the MCS corresponds to a variable number of bits. We assume that the BS can select one of M MCSs, indexed by $m = 1, \dots, M$. The number of bits per slot that can be transmitted using MCS m is denoted by R_m.

Perfect Channel State Information (CSI) and knowledge of the number of bits a user has successfully received is assumed to be available at the base station prior to the transmission in each time slot. The users see independent channel instances $h_i[k]$ at each time slot k, and the discrete-time channel model for the received signal at user i is given by:

$$s_i^{\text{rx}}[k] = h_i[k]s^{\text{tx}}[k] + n_i[k], \tag{1}$$

where $s_i^{\text{rx}}[k]$ is the signal received by user i at time slot k, $s^{\text{tx}}[k]$ is the signal broadcast from the BS at time slot k, and $n_i[k]$ is the additive white Gaussian noise term with power spectral density N_0.

For the analysis, we assume a discrete set of C channel realizations $\mathcal{H}_i$ for user i. For the dynamic programming, clearly we need a discrete set of channels. However, the heuristic that we delelop works with continuous channels as well. The probability of user i seeing channel coefficient $h_i \in \mathcal{H}_i$ in a slot is assumed to be $\alpha_i(h_i)$. The vector of channels perceived by all users, also referred to as the channel combination, is denoted by $\mathbf{h} = \{h_i, i = 1 \dots, N\}$ where $h_i \in \mathcal{H}_i$. Note that, channel realization h_i also corresponds to channel SNR γ_{h_i}. With some abuse of notation, we denote by $\mathcal{H}$ the set of all possible channel combinations, and by $\alpha(\mathbf{h}) = \prod_{i=1}^{N} \alpha_i(h_i)$, the probability of a channel combination $\mathbf{h} \in \mathcal{H}$. Note that the total number of channel combinations is C^N.

Different from the prior work discussed in Section II, we do not assume deterministic channel outage but use the probability of error (PER) for a given channel and MCS from [13] in order to model the erasure probability. Therefore, for a channel instance $h_i \in \mathcal{H}_i$, the PER for user i under MCS m is represented by $p_i^m(h_i)$, and the probability of success (PSR) is given by $q_i^m(h_i) = 1 - p_i^m(h_i)$.

Note that, to reduce the state space, we can use a normalized block size by replacing the block size that is measured in units of the greatest common divisor of the MCS rates instead of in bits. As an example, for a block size of $B = 1800000$ bits, $M = 2$ and MCSs with rates of 6Mbps and 9Mbps, the normalized block size is $B = 6000$ and the normalized transmitted data per slot $R_1 = 2$ and $R_{2=}3$, respectively (in units of 3kb if we assume a slot duration of 1ms).

In this paper, we use the following terms:
(1) A *strategy* g specifies the MCS $g(\mathbf{h})$ for each channel combination $\mathbf{h} \in \mathcal{H}$. Hence, the total number of strategies is $S = M^{C^N}$. We denote by $\mathcal{G}$, the set of all possible strategies.
(2) The *state* consists of the vector of the number of bits received by each user i denoted by $\mathbf{x} = \{x_i, i = 1, \ldots, N\}$. The state space $\mathcal{X}$ consists of all states where the number of bits received by all users is positive and less than or equal to B. The *initial state* where none of the users have any information is $\mathbf{x}^0$ and the end state where all the users have received B bits is denoted by $\mathbf{x}^B$.
(3) A *policy* μ maps any given state $\mathbf{x} \in \mathcal{X}$ to the strategy $g^\mu_\mathbf{x}$ to be used in that state.
(4) The *expected completion time* $D_\mu(\mathbf{x})$ is the mean time required to get from state $\mathbf{x}$ to the end state $\mathbf{x}^B$ under policy μ.

IV. Optimization Problem

In this section, we consider the case of memoryless channels and formulate the problem as a stochastic shortest path problem [14] with cost per stage equal to 1 (time needed per slot is fixed, $\tau = 1$) and no terminal cost. We assume that the probability of successfully receiving a packet is non-zero for every combination of modulation scheme and channel condition, though it might be extremely low for some combinations.

A. Dynamic programming based solution (Dyn-Prog)

Let $\mathcal{E} = \{\mathbf{e} \,|\, |\mathbf{e}| = N, e_i \in \{0, 1\}\}$ be the set of all vectors of size N whose components take values 0 or 1. The transition probability from state $\mathbf{x} \in \mathcal{X}$ to state $\mathbf{y} \in \mathcal{X}$ when MCS m is used under channel combination $\mathbf{h}$ is given by:

$$\rho^m_\mathbf{h}(\mathbf{x}, \mathbf{y}) = \sum_{\substack{\min(\mathbf{x}+R_m\mathbf{e},B)=\mathbf{y} \\ \mathbf{e}\in\mathcal{E}}} \left(\prod_{i=1}^{N} p^m_i(h_i)^{e_i} q^m_i(h_i)^{1-e_i} \right), \tag{2}$$

where the above minimization is defined element-wise. Note that in the case of every user experiencing an erasure, the state remains unchanged.

The state space is finite, and there clearly exists a finite integer K such that there is a positive probability of terminating after K steps irrespective of the policy. Thus, the optimal policy μ^* satisfies Bellman's equations at every state $\mathbf{x}$:

$$D_{\mu^*}(\mathbf{x}) = \min_{g\in\mathcal{G}} \left(1 + \sum_{\mathbf{h}\in\mathcal{H}} \alpha(\mathbf{h}) \sum_{\mathbf{y}\in\mathcal{X}} \rho^{g(\mathbf{h})}_\mathbf{h}(\mathbf{x}, \mathbf{y}) D_{\mu^*}(\mathbf{y}) \right), \tag{3}$$

and the optimal strategy at state $\mathbf{x}$

$$g^{\mu^*}(\mathbf{x}) = \operatorname*{argmin}_{g\in\mathcal{G}} \left(\sum_{\mathbf{h}\in\mathcal{H}} \alpha(\mathbf{h}) \sum_{\mathbf{y}\in\mathcal{X}} \rho^{g(\mathbf{h})}_\mathbf{h}(\mathbf{x}, \mathbf{y}) D_{\mu^*}(\mathbf{y}) \right). \tag{4}$$

Since the state space is finite, there are several options to solve for the optimal policy as well as the minimum expected completion time. We choose a simple value iteration approach. Starting from the end state $\mathbf{x}^B$, we use Bellmann's equation Eq. 3 to determine the completion times of the states that only depend on the end state (for which the completion time is known to be 0). We then proceed in the same manner to determine the expected completion times of states that only depend on states for which the completion time is already known, until the completion times for all states are computed. This process also yields the optimum policies from Eq. 4.

B. A simple two user example

Consider a scenario with two users ($N = 2$), with identically distributed channels. Let $\mathcal{H}_1 = \mathcal{H}_2 = \{L, H\}$, and $\mathcal{H} = \{HH, HL, LH, LL\}$, where L and H denote channels with low and high channel quality, respectively. The base station can choose one of three modulation schemes in each slot. The probability of packet error when MCS m is used is denoted by $p^m(L)$ and $p^m(H)$ for both users under the low and high channel, respectively. A strategy is defined by specifying the modulation scheme to be used for each vector channel in $\mathcal{H}$.

In this example, Bellman's equation at state $\{x_1, x_2\}$ is:

$$\begin{aligned} &D_{\mu^*}(\{x_1, x_2\}) = \\ &1 + \min_{g\in\mathcal{G}} \sum_{\mathbf{h}\in\mathcal{H}} \alpha(\mathbf{h}) \Big(p^{g(\mathbf{h})}(h_1) p^{g(\mathbf{h})}(h_2) D_{\mu^*}(\{x_1, x_2\}) \\ &\quad + p^{g(\mathbf{h})}(h_1) q^{g(\mathbf{h})}(h_2) D_{\mu^*}(\{x_1, \min(x_2 + R_{g(\mathbf{h})}, B)\}) \\ &\quad + q^{g(\mathbf{h})}(h_1) p^{g(\mathbf{h})}(h_2) D_{\mu^*}(\{\min(x_1 + R_{g(\mathbf{h})}, B), x_2)\}) \\ &\quad + q^{g(\mathbf{h})}(h_1) q^{g(\mathbf{h})}(h_2) D_{\mu^*}(\{\min(x_1 + R_{g(\mathbf{h})}, B), \\ &\qquad\qquad \min(x_2 + R_{g(\mathbf{h})}, B)\}) \Big) \end{aligned}$$

We evaluate the optimal policy in a scenario where the H and L channels for the users are $\mathcal{H}_1 = \mathcal{H}_2 = \{5dB, 28dB\}$. The probability of L and H are $\alpha_1(L) = \alpha_2(L) = 0.75$ and $\alpha_1(H) = \alpha_2(H) = 0.25$. We choose such a highly variable channel, since it makes it easier to demonstrate the decision tradeoffs that the algorithm makes in the different regions of the state space. The probability of each channel combination in $\mathcal{H}$ can be easily obtained by multiplying the respective channel probabilities. For simplicity, we only use $M = 3$ MCSs with normalized rates of $R_{m=1} = 1$, $R_{m=2} = 4$ and $R_{m=3} = 9$ and the PER for each MCSs and channel instant is listed in Table I.

TABLE I
PER for different MCS and SNR value pair

$p^m_i(h_i)$	$m = 1$	$m = 2$	$m = 3$
$\gamma_H = 28dB$	0	0	0.08
$\gamma_L = 5dB$	0.23	0.97	1

In Fig. 1, each drift vector (arrow) reflects optimal policy at that state. It shows the expected future state given the optimum MCSs chosen for the different channel instances, and hence the length of a vectors is indicative of the throughput obtained by the corresponding policy.[1] We set the normalized block size $B = 20$. At the initial state $\mathbf{x}^0 = \{0, 0\}$ the optimal policy is $g(\mathbf{h}) = \{1, 3, 3, 3\}$, i.e., MCS $m = 1$ is used for

[1]For better readability we plot only every other policy vector and double their lengths.

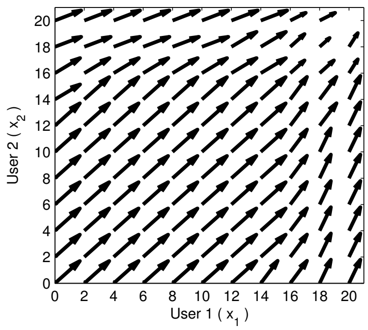

Fig. 1. Optimal policy with the dynamic programming algorithm (*Dyn-Prog*)

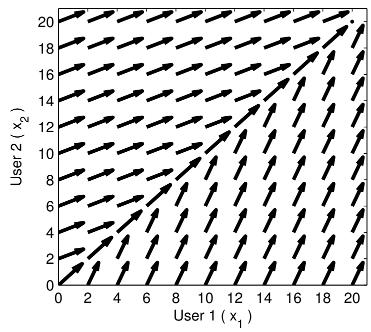

Fig. 2. Max-min Algorithm

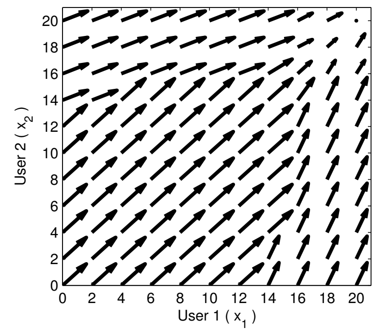

Fig. 3. Weighted completion time (Weighted-CT) algorithm

channel combination LL and MCS $m = 3$ is used for channel combinations LH,HL, and HH. This particular policy is a greedy policy which gives the maximum throughput to both users. This policy is also used in almost all states up to $\mathbf{x} = \{14, 14\}$. Closer to the borders of the state space, the policy changes from greedy to more and more favoring the user that is lagging behind. This accounts for the fact that the leading user is likely to finish before the trailing user even if MCS decisions are optimized for the trailing user and aims to prevent the loss of multiuser diversity caused by a user finishing early. For $\{14 < x_1 \leq 16, x_2 < 14\}$ and $\{x_1 < 14, 14 < x_2 \leq 16\}$, the predominant policies are $g(\mathbf{h}) = \{1, 3, 2, 3\}$ and $g(\mathbf{h}) = \{1, 2, 3, 3\}$, respectively, where a more conservative coding modulation scheme is chosen when the trailing user has a bad channel. This policy sacrifices throughput to prevent the trailing user from falling further behind. Even closer to the borders the policies are $g(\mathbf{h}) = \{1, 3, 1, 3\}$ and $g(\mathbf{h}) = \{1, 1, 3, 3\}$, even further trading off overall throughput for a higher packet reception probability for the trailing user. When both users received a similar number of bits and are close to the end state $\mathbf{x}^B$, the algorithm also chooses a more conservative MCS indicated by a shorter arrow length to avoid overshooting (i.e., unnecessarily delivering more than B bits to both users).

While solving the stochastic shortest path problem minimizes average completion time and provides the optimal policy, the size of the state space and the computational complexity increase exponentially with N, the number of users. Therefore, despite providing the optimal solution, the above approach is not, in general, practical for actual network implementation.

V. State-Aware Heuristics

Due to the high complexity of the *Dyn-Prog* algorithm introduced in the previous section, we propose two low-complexity heuristics that mimic the characteristic of *Dyn-Prog* algorithm.

A. *Maximize minimum throughput heuristic (*Max-min*)*

This heuristic is based only on the current state (the number of bits that each user has successfully received), and the current channel conditions. At each slot, the user with the least number of received bits is identified as the worst user who is most likely to require the highest number of slots to receive all data. Note that this is indeed the case when the users are homogeneous and perceive identical channel distributions. In the case of heterogeneous users, those users that perceive channel conditions that are worse (on average) are highly likely to also be the trailing users and thus most likely to finish last. The algorithm uses in each slot the MCS that maximizes the throughput for the trailing user. If both users have the same number of bits, the algorithm greedily maximizes sum throughput for both.

Fig. IV-B depicts the average drift resulting from such a policy in a scenario with the same parameter setting as explained in Section IV-B for homogeneous users. There are two predominant strategies that are used for all states off the diagonal. As *Max-min* sacrifices overall throughput in favor of the trailing user as soon as a user falls behind, the resulting strategies are $g(\mathbf{h}) = \{1, 3, 1, 3\}$ and $g(\mathbf{h}) = \{1, 1, 3, 3\}$. On the diagonal, *Max-min*'s sum throughput maximization leads to the same strategy as in the *Dyn-Prog* solution, except for the last state before finishing. Since in contrast to the *Dyn-Prog*, *Max-min* does not explicitly take expected completion time into account, it does not switch to more conservative symmetric strategies of $g(\mathbf{h}) = \{1, 2, 2, 2\}$ and $g(\mathbf{h}) = \{1, 1, 1, 1\}$, respectively, that deliver the required number of bits to finish with a lower packet loss probability compared to using the highest MCS $m = 3$ (which would result in delivering more bits than necessary to the receivers).

Overall, we note that compared to the optimal *Dyn-Prog* the *Max-min* algorithm is more conservative and ensures that the progression of state is with high probability along the diagonal where both users have the same number of bits.

B. *Weighted completion time heuristic (*Weighted-CT*)*

In many cases, favoring the trailing user is overly conservative. In particular, when the number of pending bits is large for all users and the relative lag is small, the probability that the currently trailing user finishes last is small. We now present a heuristic that more closely models the decisions taken by the *Dyn-Prog* algorithm to achieve a better tradeoff between instantaneous sum throughput and balancing the number of bits pending for different users.

At slot k, with state $\mathbf{x}$ and channel $\mathbf{h}$, we evaluate the average drift and determine the expected next state, $\mathbf{y}^m$, conditioned on using modulation scheme m as:

$$y_i^m = x_i + q_i^m(h_i)R_m,\; i = 1, \ldots, N \tag{5}$$

We then estimate the additional time required, on average, for all users to receive B bits relative to other states. Since computing the average remaining time under the optimal policy is computationally intensive, we use a weighted Euclidean distance measure in order to characterize the difference in completion times from different states. The metric $\tau_{\mathbf{y}}$associated with state $\mathbf{y}$ is:

$$\tau_{\mathbf{y}} = \sqrt{\sum_{i=1}^{N}\left(\frac{(B-y_i)}{w_i}\right)^2} \tag{6}$$

Here, the weights reflect the average channel conditions perceived by each user, with a user that perceives poor channels on average associated with a lower weight. The modulation scheme chosen at slot k is $m^* = \arg\min_m \tau_{\mathbf{y}^m}$.

1) Choice of weights: We choose weights w_i that are proportional to the average throughput achieved by the user under a hypothetical policy that chooses the MCS uniformly at random, i.e.,

$$w_i = \sum_{\mathbf{h}\in\mathcal{H}}\sum_{m=1}^{M}\alpha(\mathbf{h})q_i^m(h_i)R_m.$$

As the actual choice of MCS depends on the state as well as the channels of the other users and cannot be determined in advance (sort of using the optimum decisions given by the *Dyn-Prog*), this hypothetical policy is a very simple method to capture the relative throughput differences among the users.

In practical scenarios, the channel distribution of individual users may not be known in advance. Further, the channel statistics of a mobile user may change over time, albeit on time scales that are slow with respect to average completion times. In such settings, we use exponentially weighted averaging in order to track the user weights. The estimated weight of user i at slot k, $\hat{w}_i[k]$, when the perceived channel is $\mathbf{h}$ is given by:

$$\hat{w}_i[k] = (1-\beta)\hat{w}_i[k-1] + \beta\sum_{m=1}^{M}q_i^m(h_i)R_m, \tag{7}$$

where β is a constant that is chosen to be sufficiently small.

Fig. IV-B shows the drifts for the proposed *Weighted-CT* algorithm. Our choice of weights indeed captures well the relative desirability of the different states. While the set of strategies used is not as rich as in the *Dyn-Prog* approach, in particular at the transition between the greedy throughput maximization strategy and the more conservative border strategies, the strategies in the majority of state space are almost the same. In particular, this holds for states around the diagonal which are much more likely to occur in practice than states far off the diagonal where the number of bits for the two users differs a lot.

VI. RESULTS

In this section, we evaluate the performance of our proposed algorithms in homogeneous and heterogeneous user scenarios and compare them to the existing *Broadcast* and *Greedy* schemes discussed in Section II. For simple scenarios ($N = 2$ and $C = 2$), we also compare our results to the optimal *Dyn-Prog* solution. We start with simple scenarios to provide an intuition for the algorithms that helps to better understand the more complex scenarios. We then study the impact of block size B as well as number of users N and finally, we analyze performance under multipath Rayleigh fading channels with the ITU Pedestrian B path loss model [15].

In all of the simulations, we consider three modulation types QPSK (4-QAM), 16-QAM, and 64-QAM, with channel coding and data rates as presented in [13]. The corresponding PER with respect to the instantaneous channel quality for each MCS is obtained from the Vienna LTE link level simulator v1.7 [13]. For the performance metric we use system throughput which we compute as follows:

$$\eta = \frac{BF}{Ds} \tag{8}$$

where η is the system throughput in *Mbps*, B is the normalized block size as explained in Section III and $F = 5kb$ is the size of each unit of B. D is the completion time in terms of time slots and s is the duration of a slot, which is equivalent to a subframe time in LTE (i.e., 1ms).

A. *Comparison to the optimal* Dyn-Prog *solution*

In this section, we first analyze the performance of the different algorithms in a simple $N = 2$ scenario with $C = 2$ channel instances and $M = 13$ modulation schemes (CQI=3 to CQI=15 in [13]), which allows to obtain the optimum *Dyn-Prog* solution. The system model that we use here is the one explained in Section IV-B. We analyze both homogeneous and heterogeneous user scenarios. In these scenarios, the stationary channel probabilities for the H and L channel are $\alpha(H) = 0.25$ and $\alpha(L) = 0.75$, respectively. The normalized block size is $B = 200$.

1) Homogeneous network: In the homogeneous scenario, both users have the same channel statistics but independent channel instances. We present the results of increasing the channel variability δ, where δ is the SNR difference between the H and L channel of each user. Here, the lowest δ is $\delta =$ 0.71dB (γ_H = 9dB and γ_L = 8.29dB) and the highest δ is δ = 20.04dB (γ_H = 21dB and γ_L =0.96dB). Note that as δ increases, the SNR of the L channel γ_L decreases and γ_H increases. The sum throughput T_{sum} for each H and L channel pair for different δ is fixed and computed as follows:

$$T_{sum} = \sum_{i=1}^{N}\max(q_i^m(H) * R_m(H) + q_i^m(L) * R_m(L)) \tag{9}$$

where $q_i^m(H)$ and $R_m(H)$ are the PSR and data rate of modulation m for the H channel (respectively for the L channel). $R_m(H)$ is the modulation that maximizes the throughput of the H channel.

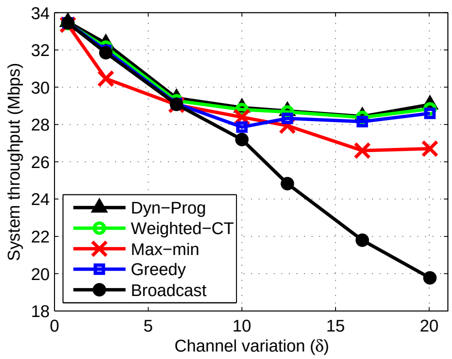

Fig. 4. Homogeneous network with increasing channel variability δ for N = 2 and B = 200.

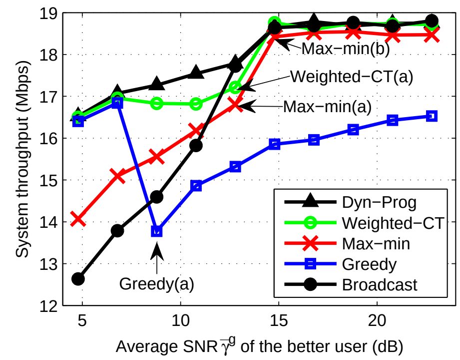

Fig. 5. Heterogeneous network with increasing heterogeneity for N = 2 and B = 200.

Fig. 4 depicts the system throughput of each scheme as δ increases. *Broadcast* performs the worst because its broadcast rate is limited by the user whose instantaneous channel γ is the lowest. As δ increases further, *Broadcast*'s system throughput decreases significantly due to the reduction in γ_L. In general, *Max-min* performs worse than the other schemes except for *Broadcast*. Its performance also decreases with increasing δ. Since *Max-min* transmits at high rate if the trailing user has a H channel and at a low rate otherwise, it maximizes the throughput of the trailing user regardless of whether the user is trailing a lot or trailing just by a single packet. It therefore outperforms *Broadcast* because it opportunistically transmits at high rate but has lower performance than the other schemes. The performance of *Dyn-Prog*, *Weighted-CT* and *Greedy* are very similar. As the average channel for both users is equal, *Weighted-CT* weighs both users equally (see Eq. 7). *Weighted-CT* makes its decision based on the MCS which maximizes the system throughput while taking into account the users' state. It's performance closely matches that of *Dyn-Prog*. *Greedy* only maximizes instantaneous throughput and performs close to but not as good as *Weighted-CT* because it ignores state information. Note that the performance of *Greedy*, *Weighted-CT* and *Dyn-Prog* first decreases but slightly improves later (i.e., at $\delta = 20.04$dB). This is because of the MCS choice for the given H and L channel. For instance, when $\delta = 16.43$dB, the throughput optimization choices are $R_m = \{17, 17, 17$ and $4\}$ for the channel combination $\mathcal{H}$ as given in Section IV-B, but the throughput optimization choices at $\delta = 20.04$dB are $R_m = \{19, 19, 19$ and $3\}$. It can easily be computed that the normalized throughput at $\delta = 20.04$dB is slightly higher than that at $\delta = 16.43$dB.

2) Heterogeneous network: In the heterogeneous channel model, one of the users has a better average channel (*good* (g) user) than the other user (*bad* (b) user). In Fig. 5, we evaluate the system throughput when increasing the average SNR $\bar{\gamma}$ of the *good* user $\bar{\gamma}^g$ and fixing the average SNR of the *bad* user $\bar{\gamma}^b$. On the left extreme, users are homogeneous. The $\bar{\gamma}$ of H and L channel of the users are $\gamma_H^g = \gamma_H^b = 11$dB and $\gamma_L^g = \gamma_L^b = -1.43$dB, respectively. On the other extreme, at $\bar{\gamma}^g = 22$dB, the H and L of the *good* user are $\gamma_H^g = 29dB$ and $\gamma_L^g = 16.57$dB, respectively while the H and L channels for the *bad* user remain unchanged (i.e., $\gamma_H^b = 11$dB and $\gamma_L^b = -1.43$dB). The δ between each pair of H and L channel is always 12.43dB.

As the difference of $\bar{\gamma}$ between *bad* and *good* user increases, the *Broadcast* and *Max-min* schemes which always favor the trailing user have an advantage over the *Greedy* scheme. *Broadcast* is the best strategy when it is optimum to only serve the user with the worst channel (i.e., when the $\bar{\gamma}$ between user is very different). The *bad* user is more likely to be trailing as $\bar{\gamma}^g$ increases. Between the points marked *Max-min(a)* and *Max-min(b)*, there is a significant improvement on the system throughput. At point *Max-min(a)*, the *good* user may still trail with some probability, whereas but at point *Max-min(b)* the probability that the *good* user trails is near zero. Therefore, as $\bar{\gamma}^g$ increases, it becomes increasingly likely that *bad* user will trail and finish last, and hence *Max-min* is close to optimal. At point marked *Max-min(a)*, the performance difference between *Max-min* and *Broadcast* is due to two main factors: (1) *Max-min* transmits at a higher rate when the trailing user has an H channel but the leading user does not. (2) *Max-min* transmits at greedy rate (i.e., rate that maximizes instantaneous throughput) when both users have the same state. Both factors lead to reduction in system throughput. The performance of the *Greedy* scheme degrades substantially at the point marked *Greedy(a)* since from this point on it ignores the *bad* user, serving first the good user entirely and then the bad user. As $\bar{\gamma}$ increases further, the time required to serve the good user decreases with the use of higher transmission rates, thus increasing overall system throughput. *Weighted-CT* manipulates w (see Eq. 7) to weigh user priority. Therefore, when the difference between $\bar{\gamma}^g$ and $\bar{\gamma}^b$ is very large, it only serves the *bad* user. On the other hand, if the difference is not too large (see point marked *Weighted-CT(a)*) and $\bar{\gamma}_H^b > \bar{\gamma}_L^g$, *Weighted-CT* prioritizes the *bad* user and this causes the *good* user to have low chance to successfully receive the transmitted data. Therefore, *Weighted-CT* does not perform as good as *Broadcast* at $\bar{\gamma}^g = 12.785$dB.

B. Larger scenarios

In this section, we present simulation results for two different settings: the impact of increasing the number of users N and the impact of increasing the block size B.

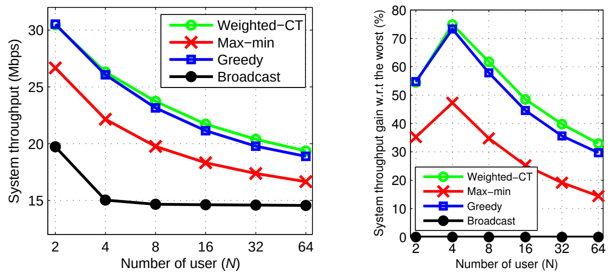

Fig. 6. Impact of increasing N in homogeneous network, B = 1000.

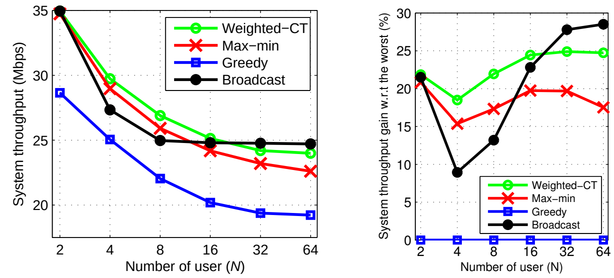

Fig. 7. Impact of increasing N in heterogeneous network, B = 1000.

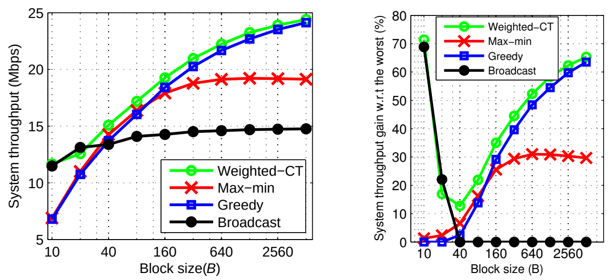

Fig. 8. Impact of increasing B in homogeneous network, N = 10.

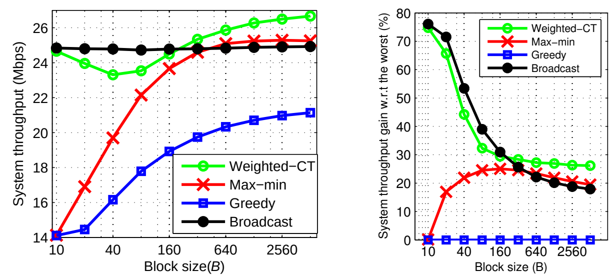

Fig. 9. Impact of increasing B in heterogeneous network, N = 10.

1) Impact of increasing N: For the scenario where N increases, we fix $B = 1000$ for both the homogeneous and the heterogeneous scenario. N increases exponentially from 2 to 64. Fig. 6 and Fig. 7 show that system throughput for all the schemes decrease as N increases since completion time is determined by the slowest user and as N increases, the probability that some users sees a high number of bad channels and lags far behind increases as well.

Homogeneous scenario. For the homogeneous scenario, the corresponding channel SNR of the H and L channels are $\gamma_H = 15$dB and $\gamma_L = 5$dB, respectively. In this scenario (see Fig. 6), *Broadcast* performs the worst as it transmit at the lowest rate more than 90% of the time. As N increases, a strategy that serves all users becomes better and better. The probability of having many users with an L channel is higher and therefore the performance gain of the other schemes over *Broadcast* decreases.

Heterogeneous scenario. In the heterogeneous scenario, the SNR of the H and L channels for the *good* users are $\gamma_H^g = 29$dB and $\gamma_L^g = 19$dB, respectively. The channel SNR of H and L for *bad* user are $\gamma_H^b = 15$dB and $\gamma_L^b = 5$dB, respectively. When $N > 2$, users are divided into two equal sized groups: one group has the channel statistics of the *good* user and the other group has the channel statistics of the *bad* user. As explained in Section VI-A2, since the difference in average SNR between the *good* and *bad* user groups is large, it is mainly important to serve the *bad* user group. *Greedy*, performs badly regardless of N because it spends too much time making the wrong decision to favor the *good* user group and ignoring the *bad* user group (see Fig. 7). It is interesting to note that *Broadcast* performs worse than *Weighted-CT* when $N < 16$ but outperforms it for $N \geq 16$. This is due to the fact that for small N there is some probability that a significant fraction of the bad users have an H channel and it would be better to use a higher MCS than the one chosen by *Broadcast*. As N increases further, the probability of having an H channel for sufficiently many users in the bad user group is very small. Beyond $N = 8$ the performance of *Broadcast* remains constant because it is limited by the L channel of the *bad* user group. In addition, *Broadcast* performs better than the other schemes because always serving all user of the bad group and transmitting at the broadcast rate is the optimal choice. *Weighted-CT* does not perform as well as *Broadcast* because *Weighted-CT* may sometimes transmit at higher rate when sufficiently many of the *bad* users have an H channel.

2) Impact of increasing B: Here, $N = 10$ and we investigate system throughput as B increases exponentially from 10 to 5120. As can be seen from Fig. 8 and Fig. 9, system throughput increases as B increases. When increasing B the fraction of time with many users in the system is larger and there are more opportunities to realize an opportunistic gain.

Homogeneous scenario. In Fig. 8, for very low B, *Broadcast* and *Weighted-CT* perform well because it is important ensure that all the users are served at each time slot. As *Greedy* and *Max-min* may transmit at a higher rate and thus perform worse, since already lagging by a small amount may increase overall completion time. As B increases, the total number of slots required for completion increases as well. Since all users have the same average channel and differences among the users are more and more likely to average out over time, it is best to maximize the throughput achievable at each time slot. *Greedy* exploits this advantage and it performs close to *Weighted-CT*. The performance of *Max-min* is initially similar to *Greedy* but later performs worse. A higher B increases completion time so that the channels seen by the users are more likely to average out. The average channel seen by the trailing user over the whole duration of the simulation becomes almost flat for $B > 640$, as can be seen from the *Max-min* curve. As mentioned in Section VI-B1, for $N = 10$, is is likely that at least one of the users sees the worst channel, therefore even with increasing B, the system throughput of *Broadcast* does not change as much as that of the other schemes.

Heterogeneous scenario. In this scenario, as explained in Section VI-B1, the difference between the average SNR of the *good* and *bad* user group is large. *Greedy*'s performance increases for increasing B for the reasons discussed in the previous section, but it still performs worst of all the schemes since it first serves the good users and then the bad ones instead of simultaneously serving both user groups. For heterogeneous user groups, *Weighted-CT* does not perform well when B is small due to the small number of time slots. There is not enough time for users to see average channel statistics and always serving all users is best. This impacts the performance of *Weighted-CT* because its decisions critically depends on the accurate estimation of each user's weight w. However, its performance improves and it outperforms *Broadcast* as B increases due to the improved estimation of w. *Max-min* is slightly better than *Broadcast* at higher B because it obtains an opportunistic gain when the trailing user has an H channel.

C. Multipath Rayleigh Fading networks

In the following, we use the flat Rayleigh fading model with path loss (from ITU-R Pedestrian B [15]) and evaluate the performance in both homogeneous and heterogeneous scenarios for different block sizes B and number of users N. In the homogeneous scenario, we place the users equidistant from the base station while in the heterogeneous scenario, the users are randomly distributed within the cell coverage area. The multipath Rayleigh channel distributions of the users are independent and identically distributed (i.i.d.). The transmit power is set such that the edge user is still able to receive the packet with some probability of success at the lowest MCS. Note that, since we use the pedestrian channel model, the rate of change in channel SNR is rather low. Therefore, when B and N are small, the probability of having users with the same average channel statistics even in a homogeneous scenario is small. For this reason, we expect to see different performance here as compared to that in Section VI-A and Section VI-B. In addition, since the channel model is the slowly varying pedestrian channel, we use $B = 2000$ instead of 1000 as in Section VI-B. This is to ensure that the completion time is significantly larger than the coherence time of the channel.

1) Impact of increasing N: Here, we observe the impact of increasing N exponentially from 2 to 64. In Fig. 10 and Fig. 11, we see that as N increases, the average system throughput decreases. This is consistent with the results we obtained in the simple scenario in Section VI-B1.

Homogeneous scenario. In Fig. 10, *Broadcast* performs close to *Max-min* at $N = 2$ because the probability that the trailing user is the worst user is high for the required short completion time with only two users. With the slow channel variation of the Rayleigh channel, at $N = 2$, we observe from the channel distribution that the user that has a better channel than the other will usually remain better for at least 50 slots. In the scenario with uncorrelated channels depicted in Fig. 6, the trailing user is very random at each slot and therefore we cannot conclude that the trailing user is the worst user, which explains the performance difference between *Broadcast* and *Max-min*. At higher N, the likelihood for the trailing user to be the one with the worst channel decreases and is essentially random. As a result, we observe a similar performance in both Fig. 10 and Fig. 6.

Heterogeneous scenario. We observe from the system throughput gain in Fig. 11 that the performance of *Greedy* is worse than the rest of the schemes when N is smaller than 32 because the probability of multiple users simultaneously receiving a packet successfully is small. However, as N increases, *Greedy* achieves an opportunistic gain by serving multiple (but not all) users with a higher rate while *Broadcast* transmits at a lower rate to all users. *Weighted-CT*, *Max-min* and *Broadcast* perform well for small N because it is important to serve the users who are further away from the BS. As N increases, a scheme can achieve opportunistic gain by serving more users than conservatively serving all users with low rate. The state aware *Max-min* performs better than both *Greedy* and *Broadcast* at high N because it has the opportunity to transmit at a higher rate, giving to the majority of the users. *Weighted-CT* that jointly maximizes the opportunistic gain for the users and takes into account the average completion time of each user always performs better than the other schemes when the number of transmission slots required is large enough (i.e., four times as many at $N = 64$ compared to $N = 2$).

2) Impact of increasing B: Here, we fix N to 10 and observe the impact of increasing B exponentially from 10 to 5120.

Homogeneous scenario. As explained in Section VI-C, the average channel between users is different when the number of transmission slots is small. In this case, shown in Fig. 12, the average number of transmission slots needed for the worst scheme (in this case, the *Greedy* scheme) to complete its transmission at $B \leq 100$ is only 90 slots. Therefore, we can conclude that with the slow rate of variation of the pedestrian channel, all users see different average channel statistics. This explains the bad performance of *Greedy* for $B \leq 100$. As explained in Section VI-B2, with limited transmission slots, the weight estimate with *Weighted-CT* does not match the channel statistics and thus *Weighted-CT* does not performs well for $B \leq 320$. At larger B, the average channel statistics are indeed observed, therefore performance is similar to Fig. 8.

Heterogeneous scenario. In Fig. 13, users are randomly distributed within the cell area and thus they experience heterogeneous channel conditions. We can easily compare the performance of this scenario to that in Fig. 9 and the reasons for the performance differences between schemes are the same as the ones in Section VI-B2. *Weighted-CT* does not underperform *Broadcast* as much as in Fig. 9 because the user with a poor average channel is also likely to have the worst instantaneous channel and vice versa.

VII. Conclusions

In this paper we investigated the finite horizon opportunistic multicast scheduling problem, where a wireless base station

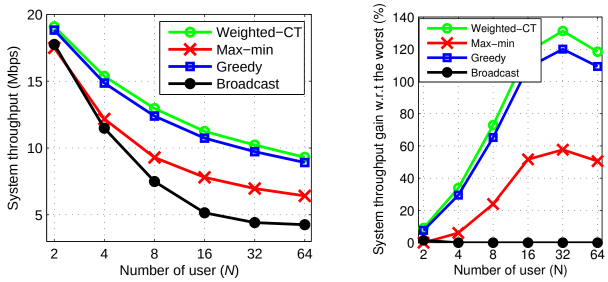

Fig. 10. Impact of increasing N for homogeneous multipath Rayleigh fading scenario, B = 2000.

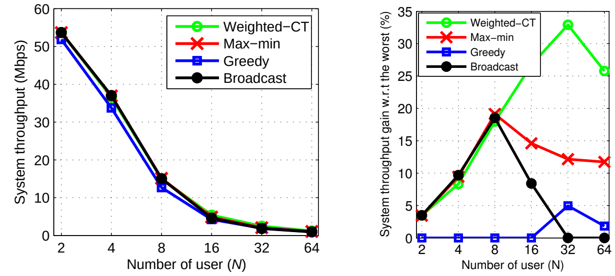

Fig. 11. Impact of increasing N for heterogeneous multipath Rayleigh fading scenario, B = 2000.

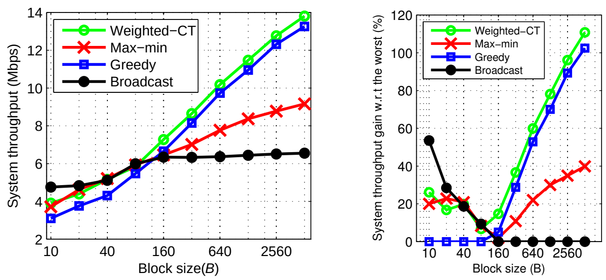

Fig. 12. Impact of increasing B for homogeneous multipath Rayleigh fading scenario, N = 10.

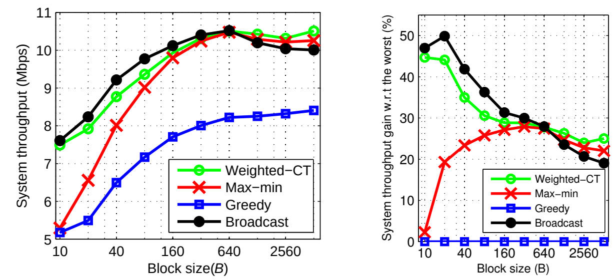

Fig. 13. Impact of increasing B for heterogeneous multipath Rayleigh fading scenario, N = 10.

transmits a fixed amount of erasure coded data to a set of receivers. We designed an algorithm based on dynamic programming that, to the best of our knowledge, is the first to explicitly take into account the system state in terms of received amount of data at each receiver for the selection of the optimum modulation and coding scheme. In addition to the well known tradeoff between broadcast gain and multi-user diversity gain that is inherent to opportunistic multicast scheduling, the finite horizon nature of our problem introduces an interesting further tradeoff, namely that of equalizing the finish times of the users versus the total system throughput. While it has been shown that serving a subset of users with good channels improves throughput, it also increases the likelihood that those users finish before the other users. The more users leave the system early, the lower the multiuser diversity gain that can be obtained through opportunistic scheduling from then on. This tradeoff is state dependent. Intuitively, throughput maximization is a reasonable strategy as long as users are far from finishing, whereas the closer users are to finishing, the more important it becomes to allow lagging users to catch up rather than optimizing throughput for all. Based on these insights, we designed two simple and practical heuristics that perform close to the more complex proposed dynamic programming solution and that outperform existing approaches that do not consider state. We performed an extensive range of simulations for homogeneous and heterogeneous user scenarios and showed that our heuristics outperform existing schemes by *Greedy* and *Broadcast* as much as 35% and 100%, respectively in realistic Rayleigh fading scenarios.

Acknowledgement

This article has been partially supported by the Madrid Community through the MEDIANET project (S2009-TIC1468).

References

[1] P. K. Gopala and H. E. Gamal, "Opportunistic multicasting," in *Asilomar Conference on Signals, Systems and Computers*, Nov. 2004.

[2] T.-P. Low, M.-O. Pun, Y.-W. P. Hong, and C.-C. J. Kuo, "Optimized opportunistic multicast scheduling (oms) over wireless cellular networks," *IEEE Transactions on Wireless Communications*, vol. 9, no. 2, pp. 791 –801, Feb. 2010.

[3] A. Richard, A. Dadlani, and K. Kim, "Multicast scheduling and resource allocation algorithms for OFDMA-based systems: A survey," *IEEE Communications Surveys Tutorials*, pp. 1–15, 2012.

[4] U. Kozat, "On the throughput capacity of opportunistic multicasting with erasure codes," in *IEEE Infocom*, Apr. 2008.

[5] A. Shokrollahi, "An Introduction to Low-Density Parity-Check Codes," ser. Lecture Notes in Computer Science. Springer, 2002.

[6] ——, "Raptor codes," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2551–2567, 2006.

[7] D. L. Gall, "MPEG: a video compression standard for multimedia applications," *Communications of the ACM - Special issue on digital multimedia systems*, vol. 34, no. 4, pp. 46–58, Apr. 1991.

[8] P. K. Gopala and H. E. Gamal, "On the throughput-delay tradeoff in cellular multicast," in *International Conference on Wireless Networks, Communications and Mobile Computing*, Jun. 2005.

[9] T.-P. Low, M.-O. Pun, and C.-C. J. Kuo, "Optimized opportunistic multicast scheduling over cellular networks," in *IEEE Globecom*, Dec. 2008.

[10] T.-P. Low, M.-O. Pun, Y.-W. P. Hong, and C.-C. J. Kuo, "Optimized opportunistic multicast scheduling (OMS) over heterogeneous cellular networks," in *IEEE ICASSP*, Apr. 2009.

[11] W. Huang and K. L. Yeung, "On maximizing the throughput of opportunistic multicast in wireless cellular networks with erasure codes," in *IEEE ICC*, Jun. 2011.

[12] ——, "Optimal opportunistic multicast for minimizing broadcast latency in wireless networks," in *IEEE ICC*, May 2010.

[13] C. Mehlfuhrer, M. Wrulich, J. C. Ikuno, D. Bosanska, and M. Rupp, "Simulating the long term evolution physical layer," in *Proc. of the 17th European Signal Processing Conference (EUSIPCO 2009)*, Glasgow, Scotland, aug 2009.

[14] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Vol. I*, 3rd ed. Athena Scientific, 2005.

[15] IMT-2000, *ITU-R M.1225: Guidelines for Evaluation of Radio Transmission Technologies for IMT-2000*, 1997.