On the Throughput Capacity of Opportunistic Multicasting with Erasure Codes

Ulaş C. Kozat
DoCoMo USA Labs
3240 Hillview Avenue, Palo Alto, CA
kozat@docomolabs-usa.com

Abstract—In this paper, we concentrate on opportunistic scheduling for multicast information. We pose the problem as a multicast throughput optimization problem. As a solution we present how one can jointly utilize fixed-rate and rateless erasure coding along with simple rate adaptation techniques in order to achieve the optimal multicast throughput per user.

We first investigate the performance of the proposed system under i.i.d. channel conditions. Our analysis shows a linear gain for the multicast capacity over i.i.d. Rayleigh fading channels with respect to the number of users. Since the established results require coding over large number of blocks and hence induce large decoding delays, we extend our analysis to the cases where we code over shorter block lengths and thus quantify the delay-capacity tradeoffs under a simple setting. We further look into non-i.i.d. channel conditions and show achievable gains by modifying a scheduling heuristic whose fairness is well-established for opportunistic scheduling of unicast flows.

Our overall evaluations demonstrate that under both i.i.d. and non-i.i.d. channel conditions, opportunistic multicasting with erasure coding can significantly improve the performance over the traditional techniques used in today’s communication systems.

I. INTRODUCTION

Opportunistic scheduling (also known as multiuser diversity) in wireless networks has been a popular subject in the recent years. In fact, its certain elements have already become part of the 3G cellular systems [1]–[4]. The main underlying idea of this scheme is to take advantage of the variations in the wireless channel conditions to maximize the system and/or user throughput capacity.

In the traditional setting, a wireless system consists of a base station and multiple users that share the same channel to receive their individual unicast flows from the base station. The base station runs a centralized scheduler that allocates the shared channel to a unique user at each channel use and transmits only the flows destined to that user. Once the channel qualities of each receiver is known at the transmitter and assuming they are slowly changing over the time, the central scheduler at the base station can target the users that have more favorable channel conditions to optimize a given utility function.

More recently, researchers have started looking into multicast scenarios and extending opportunistic scheduling ideas to the cases where users of the channel are not necessarily interested in different, but the same information [1], [5]. In this paper we too focus on a similar multicast scenario and investigate the means of optimizing the per user multicast throughput capacity. To this end we will make use of two key techniques: (i) opportunistic multicast scheduling and (ii) fixed-rate or rateless erasure coding.

In contrast to the opportunistic unicast scenarios, opportunistic multicast does not generally target one user at a time, but instead it searches for the best subset of users to schedule at each channel use. Scheduling is implicitly accomplished in a cross-layered coordination with the physical layer by using a proper transmission rate that will guarantee successful decoding only at the targeted subset of users in the next channel use. For instance, scheduling one user at a time (e.g., by setting the transmission rate that matches to the best user’s channel conditions) and scheduling all users at the same time (e.g., by setting the transmission rate that matches to the worst user’s channel conditions) become special cases of the opportunistic multicasting. As it will be clear from our analysis in the later sections these extremes are quite suboptimal in terms of achievable throughput capacity over a wide range of channel conditions as captured by their signal to noise ratios.

One problem with the opportunistic multicasting is that the subscribed users of the multicast channel would not always be able to decode the transmitted information. However, in multicasting the goal is to deliver the whole content to everyone in the multicast group. One way to overcome this problem is to treat the opportunistic multicasting as an erasure channel above the scheduling layer, where the loss probabilities in the range of [0.5, 0.9] are typical rather than exceptions due to the throughput optimization. Instead of sending the original multicast message blocks, then the base station schedules and transmits encoding blocks output by an erasure code. Depending on the system architecture, channel assumptions, and delay considerations, we have different erasure coding options at our disposal such as fixed rate erasure codes (e.g., Reed-Solomon block codes, burst-erasure correcting codes, Tornado codes, etc. [6]) and rateless codes (e.g., LT codes, Raptor Codes, online codes [7]–[9]). Hence, by coding appropriately we can indeed achieve the throughput capacity of opportunistic scheduling.

Using these two pillars of our proposed multicast system, we first investigate in this paper the achievable throughput capacity under independent and identically distributed (i.i.d.) channel conditions across the users. We establish capacity gains over different SNR ranges and number of users in the system. Particularly, we prove that the capacity gains
 Opportunistic multicasting appears in a limited fashion in [5]. Authors propose a queueing-based approach, where $N$ users are divided into distinct subsets of size $N/2$ and a separate service queue is defined for each subset. In the proposed solution, when a multicast packet is to be sent, the scheduler finds the subset $S$ that can support the highest transmission rate (i.e., the subset which has the $N/2$ highest SNR values) and sends the packet at that rate. Scheduler also copies the packet to the service queue that corresponds to the complementary subset of $S$, i.e., $SC = \{1, \ldots, N\}/S$ before discarding it. When the nodes in $SC$ have the highest $N/2$ SNR values, the scheduler sets the transmission rate according to $SC$. Thus, the same multicast packet is eventually scheduled in two different time-slots and received by all the receivers. Authors analyze how the throughput and delay scale within this set-up. They also propose hybrid-ARQ based modifications to their scheme to improve the delay at the expense of some capacity loss. Our assumptions differ from [5] in many critical points. First of all, we do not limit ourselves to the scheduling decisions for the best half of the users and rescheduling of the same packet later on. In fact, our analysis shows that one generally obtains much better throughput by scheduling to more than half of the users at each time slot. Secondly, our strategy relies on sending erasure encoded blocks rather than original message blocks and hence defines a radically different system. This also requires a separate analysis of delay considerations. Thirdly, we analyze both i.i.d. and non-i.i.d. channel conditions.

In another closely related work [10], authors propose to use application layer forward error correction with multicast scheduling. Their system model relies on setting a target threshold $T$ such that the transmitter sends the packets when at least $T$ nodes are able to receive the packet successfully. Authors analyze the capacity optimizing thresholding strategy over i.i.d and non-i.i.d. cases as well as single and multiple antenna systems. Our model is different in the sense that transmitter is work-conserving, i.e., it always transmits. However, the transmission rate itself can be low or high depending on the current channel realization. We believe that our system model is more appropriate for contention-free cellular networks.

Other related works include the ones proposed for multi-hop wireless networks where the problem is generally treated from the coverage (i.e., reaching to more number of nodes) vs. data rate point of view [11]. In those settings, the goal is not to maximize the broadcast/multicast capacity from a single transmitter point of view but from the network point of view, which makes the problem setting quite different than ours.

Having covered the most relevant works to ours, we are now ready to proceed into the next section for the details of our system model, assumptions, and problem formulation.
Here, $s(t)$ denotes the multicast message sent by the transmitter with average power constraint $E[|s(t)|^2] \leq P$. The channel fading coefficient $h_i$ of user $i$ is assumed to be circularly symmetric complex Gaussian random variable. Similarly, the additive noise $n_i$ is also assumed to be circularly symmetric complex white Gaussian noise with zero mean and unit variance. $h_i$ remains the same during a time slot, and it is assumed to be known both at the receiver and the transmitter. However, $h_i$ varies in an i.i.d. fashion from slot to slot. For two different users $i$ and $j$, we will always assume that $h_i$ and $h_j$ are independent processes, but we will consider both the identical (see Section IV) and non-identical (see Section VI) distributions across users. $n_i$ on the other hand is always assumed to be i.i.d. across users.

Using this model, we can express the capacity per channel use in slot $k$ as:

$$C_i[k] = \log_2 (1 + SNR_i[k])$$

where $SNR_i[k] = \|h_i[k]\|^2 P$ and it represents the average signal to noise ratio for user $i$ during $k$th time slot. Note that whenever the transmitter sends at a rate $R[k] > C_i[k]$, node $i$ cannot recover any of the transmitted information during slot $k$. In contrast, if the transmitter sets its rate to any value $R[k] \leq C_i[k]$, $i$ can recover all the information sent during slot $k$.

In traditional systems the transmission rate is set with respect to the worst case channel profile, and hence, the lowest available channel rate is used to deliver multicast/broadcast information. In our model we allow the transmitter to perform rate adaptation for multicast transmissions as follows. At the beginning of each time slot, transmitter sets its transmission rate $R[k]$ and it keeps sending at the same rate for the duration of the time slot. Accordingly, the throughput capacity of user $i$ until the end of $n$th time slot can be expressed as:

$$\Psi_i[n] = \frac{1}{n} \sum_{j=0}^{n} I_i[j] \cdot R[j] \quad (1)$$

Here $I_i[j]$ is an indicator function that equals to one if user $i$ can successfully receive in the $j$th slot and it is zero otherwise. Eq. 1 implies the following tradeoff: We can force scheduling of user $i$ at more and more slots at the risk of reducing the transmission rate $R$ due to the fluctuations in user $i$’s channel conditions.

A. Optimization Problem

The system model summarized so far leads us to the following optimization problem:

$$\lim_{n \to \infty} \left( \max_{R[0], \ldots, R[n]} \min_{i=1, \ldots, N} \Psi_i[n] \right) \quad (2)$$

In other words, we want to maximize the minimum throughput capacity across all multicast users. The problem as stated in (2) is overly complicated to solve and may not lead to an intuitive solution. Instead we will simplify the problem for i.i.d. and non-i.i.d. cases by constraining the rules of picking the transmission rate at each time-slot.

1) I.I.D. Case: When the channel conditions are i.i.d. across the users, the transmitter is constrained to choose a transmission rate to serve the best $L$ users in each slot. $L$ can be any value from 1 to $N$, but once it is fixed, it remains the same until the number of users in the system changes or until the multicast session ends. To be more specific, let the ordered capacities of users at the beginning of the next slot $(n+1)$ be $C_{(1)} \geq C_{(2)} \geq \ldots \geq C_{(L)} \geq \ldots \geq C_{(N)}$. The transmitter sets the channel rate as $R[n+1] = C_{(L)}$. Notice that $R[n]$ becomes an i.i.d. random process. Furthermore $I_i[n]$ also becomes an i.i.d. random process, which is also independent from $R[n]$. This is due to the fact that any receiver is equally likely to be among the $L$ best users given the i.i.d. channel statistics. Therefore, $I_i[n] = 1$ with probability $p = L/N$ and $I_i[n] = 0$ with probability $(1-p) = (1-L/N)$.

With this rate selection rule we can revisit (1) and observe that:

$$\Psi_i(L) \triangleq \lim_{n \to \infty} \Psi_i[n] = E[I_i[j] \cdot C_{(L)}]$$

$$= E[I_i[j]] \cdot E[C_{(L)}] = \frac{L}{N} E[C_{(L)}] \quad (3)$$

almost surely according to the strong law of large numbers. Since equation (3) does not depend on a particular user and the sole parameter to change is $L$, transmitter selects $L^*$ that optimizes the long-term throughput capacity:

$$L^* = \arg \max \left\{ \frac{L}{N} E[C_{(L)}] \right\} \quad (4)$$

2) Non-I.I.D. Case: When the channel conditions vary independently but non-identically across the users, we need to take into account the fairness issue. This situation typically arises from the geographical spread of users. Users who are closer to the base station has typically lower signal attenuation and hence better average signal power than the ones further apart. In the literature, this fairness issue is well-tackled for unicast opportunistic scheduling [2], [3], [13]. One popular unicast scheduling algorithm which is part of 3G systems is the Proportional-Fair Sharing (PFS) [4]. In PFS, the transmitter normalizes the achievable capacity of each user in the next slot by the throughput observed by that user. Hence, each user competes against its own mean rather than the other users. It turns out that this scheduling strategy is optimum in the sense of maximizing certain utility functions, e.g., sum of logarithm of each users long-term throughput [2]. Due to its lack of short-term fairness and stability problems, other normalization techniques that take into account inter-scheduling times, queue lengths, delay jitters, etc. [3], [13]. We reuse a modified

\footnote{The slot index is omitted for notation simplicity.}
version of PFS metric that replaces average throughput with the mean capacity of each user as summarized below.

In order to set the rate of multicast channel in the next slot \((n+1)\) the transmitter first computes \(C_i[n+1] = C_i[n+1]/E[C_i]\) for each receiver \(i\). Then it sorts the normalized capacities as \(C_{(1)} \geq C_{(2)} \geq \ldots \geq C_{(L)} \geq \ldots \geq C_{(N)}\). The transmitter picks the \(L\) highest values in the sorted list and constructs the receiver index set \(J_L[n+1] = \{j_1, \ldots, j_L\}\) by finding unique \(j^\prime\)’s in the set \(\{1, \ldots, N\}\) such that \(C_j[n+1]/E[C_j] = \tilde{C}_{(m)}\) is satisfied for \(m = 1, \ldots, L\). The multicast channel rate in time slot \((n+1)\) is then computed as:

\[
R[n+1] = \min_{j \in J_L[n+1]} C_j[n+1]
\]

One crucial point here is that it is always the case that \(R[n+1] \leq C(L)\) and it is typical to have \(R[n+1] < C(L)\). In other words, we typically schedule more than \(L\) users in a given time slot.

Now, when we revisit (1), we can see that the rate selection process depends only on the current channel conditions, which is i.i.d. from one slot to another for a given user. Therefore, a user’s chances of being scheduled in a given time-slot is independent of the past and future schedules. However, depending on the computed \(R\) value, this time some users might have better or worse chance of being scheduled in a given slot. This renders \(R[j]\) and \(I_t[j]\) correlated. With these observations and again using strong law of large numbers, we can express \(\Psi_t\) almost surely as:

\[
\Psi_t = E[I_t \cdot \min_{j \in J_L} C_j]
\]

And our optimization problem then becomes finding the best \(L^*\) such that:

\[
L^* = \arg \max_L \left\{ \min_{i \in \{1, \ldots, N\}} E \left[ I_i \cdot \min_{j \in J_L} C_j \right] \right\}
\]

\[
(5)
\]

**B. Utilizing Erasure Codes**

One crucial component in multicast scheduling is to make sure that all of the multicast users receive each and every block of the multicast content under normal operations. In our system however, the scheduler can decide to serve a proper subset of \(N\) users in the system. Suppose as a result of the scheduling decision \(K\) users can successfully recover the blocks transmitted in the current time slot. From the remaining \((N-K)\) users point of view, there occurs a burst of block erasures, where the burst length is of the same size as the slot length. Therefore, our opportunistic multicasting strategy creates an artificial erasure channel. Depending on the implementation strategy and target decoding delays, one can readily apply existing erasure coding techniques. Here, we underline two of such schemes, namely fixed rate Maximum Distance Separable (MDS) codes such as Reed-Solomon codes and rateless codes such as Raptor codes and point out their differences.

An \((l, m)\) block code with rate \(r = m/l\), produces \(l\) encoding blocks from \(m\) original message blocks. MDS codes are efficient in the sense that the decoder can recover the original \(m\) blocks from any of the \(m\) encoding blocks out of the generated \(l\) blocks [6]. As opposed to fixed rate codes, a rateless code can generate as many encoding blocks as needed in a probabilistic fashion with very low probability of repetitions. The downside is however recovering the original \(m\) blocks becomes a probabilistic event. In [14], authors provide a tight performance expression on failure probability for Raptor Codes that is valid for block lengths \(m > 200:\)

\[
P_f(l, m) = \begin{cases} 1, & \text{if } l < m \\ 0.85 \times 0.567^{l-m}, & \text{if } l \geq m \end{cases}
\]

Above expression states that for \(m > 200\) and \(l \geq m\) the performance is a function of the coding overhead \((l - m)\) and independent of the content size in number of blocks. When \(l = m\), unlike MDS codes we are not guaranteed to recover the \(m\) original blocks. On the contrary we fail with a high probability of 0.85. But the good news is that with as low as 50 extra blocks we can achieve \(P_f < 10^{-12}\), which as an overhead goes to zero as \(m \rightarrow \infty\).

Fixed rate erasure codes require a careful assessment of the wireless channel and scheduling decisions. If any of these conditions change, they cannot easily accommodate the changes unless they are already designed with a pessimistic view. Therefore, a tight control is required for optimal point of operation and it is best if erasure coding is applied at the base station next to or above the packet scheduler over small number of blocks.

Rateless codes on the other hand is quite flexible and can blindly accommodate the throughput fluctuations. If different receivers observe different short-term or long-term throughput, each receiver achieves its own throughput and does not create a bottleneck for other users. They can be best applied in the application layer at the remote server side where the whole content is available or it can be applied at the base-station after enough buffering if server does not support application layer FEC. Rateless codes can be also used as an inefficient fixed rate code by enforcing a rate limit.

We will use the following two lemmas in the next sections, for which we present the proofs in the appendix. It suffices here to say that these lemmas state that over large number of blocks, the throughput results obtained as solutions of the optimization problems we defined in the previous section are achievable by using erasure codes mentioned in this section.

**Lemma 1:** Under the i.i.d. conditions given in Section III-A.1, the optimum rate \(\Psi^*_t = \tilde{E}[C_{(L^*)}]\), where \(L^*\) as given in (4) can be achieved with high probability by using an MDS code of rate

\[
r = \frac{\Psi^*_t}{\tilde{E}[C_{(L^*)}]} = \frac{L^*}{N}.
\]

Raptor codes can achieve \(\Psi^*_t - \epsilon\) with high probability for infinitesimal but finite \(\epsilon > 0\).

**Lemma 2:** Under the non-i.i.d. conditions given in Section III-A.2, the rate \(\Psi^* = \min_i E[I_i \cdot \min_{j \in J_L} C_j]\) can be achieved for all receivers with high probability by using an
MDS code of rate
\[ r = \frac{\Psi}{E[\min_{j \in J_L} C_j]} \].
Raptor codes can achieve \( \frac{\Psi_i}{E[I_i \cdot \min_{j \in J_L} C_j]} - \epsilon \) for each receiver \( i \) with high probability for infinitesimal but finite \( \epsilon > 0 \).

IV. THROUGHPUT ANALYSIS FOR I.I.D. RAYLEIGH CHANNELS

In this section, we analyze the long-term per user throughput under i.i.d. Rayleigh channel conditions. We first obtain a closed form integral equation to compute the throughput capacity when we target the best \( L \) users in each time slot and set the channel rate accordingly. Since \( 1 \leq L \leq N \), we will then numerically solve the optimization problem by computing the throughput for each \( L \). We will also validate our finding with Monte Carlo simulation of the system. We further establish a simple and provable scalability rule.

Let us denote the channel SNR of \( i \)th user as \( X_i \) and suppose \( X_i \) is distributed according to the probability distribution function \( f_X \). We also denote cumulative distribution function as \( F_X = P(X_i \leq x) \) and define its complementary function as \( F^c_X = 1 - F_X \). When we consider an ordered sequence of receiver SNRs from highest to lowest at a given time slot, we will refer to the \( L \)th largest value as \( X_{(L)} \), i.e., \( X_1 \geq X_2 \geq \ldots \geq X_{(L)} \geq \ldots \geq X_{(N)} \) in that slot. We further define \( f_{X_{(L)}} \), \( F_{X_{(L)}} \), and \( F^c_{X_{(L)}} \) as the probability density function, cumulative distribution function, and complementary cumulative distribution function of \( X_{(L)} \), respectively.

Remember that we are trying to maximize the following expression \( \Psi(L) = \frac{L}{(\ln 2)N} \int_0^\infty \ln(1+x) f_{X_{(L)}}(x) dx \).

By definition \( F^c_{X_{(L)}}(x) = P(X_{(L)} > x) \). Let us also define the set \( \Omega(x) = \{ i : X_i > x \} \) and let \( |\Omega| \) denote its cardinality. Since the event \( \{ X_{(L)} > x \} \) is equivalent to \( \{ |\Omega(x)| \geq L \} \), we can compute \( F^c_{X_{(L)}}(x) \) as:
\[ F^c_{X_{(L)}}(x) = \sum_{j=L}^N \binom{N}{j} (1 - F_X(x))^j \cdot (F_X(x))^{(N-j)} \]

3I.I.D. channel conditions across different time slots as well as among different receivers.
4We dropped the user index \( i \) since in this case each user will have the same throughput.

So far we have not actually specified the distribution of \( X_i \)'s. Under our circularly symmetric complex Gaussian fading assumption, SNR's are exponentially distributed, i.e., \( F_X(x) = 1 - e^{-\lambda x} \) with mean \( 1/\lambda \). Thus, we can rewrite the throughput expression as:
\[ \Psi(L) = \frac{L}{N \ln 2} \sum_{j=L}^N \binom{N}{j} \int_0^\infty e^{-\lambda x} \cdot (1 - e^{-\lambda x})^{(N-j)} \frac{dx}{1+x} \]

At this point it is straightforward to compute the long term multicast throughput for each possible \( L \) value. Using Lemma-1, we can further establish that each of these multicast throughput points are indeed achievable via erasure coding.
with high probability.

We choose our comparison baseline with respect to a more traditional system where the transmission rate is set with respect to the worst user. This baseline operation coincides with $L = N$. Hence, we normalize achievable throughput at each $L$ by $\overline{Ψ}(N)$. In Fig. 1 we plot this normalized throughput at 10dB average SNR for i.i.d. Rayleigh channel using both the expression (6) and Monte Carlo simulations. The marks on the curves show the results of the numerical calculations and the lines show the simulation results. The horizontal axis indicate the $L$ parameter. Each simulation point is obtained by averaging over 100000 time slots. As it is visible from these curves, our numerical results and simulation results do well agree with each other.

One of the critical observations here is that the peak values occur at $L = 7, 32, 64$ when there are $N = 10, 50, 100$ multicast users in the system, respectively. This signals a linear gain relation between the normalized peak capacities and the multicast group size (as depicted by the linear arrow on the plot). We formalize this statement in the following theorem.

**Theorem 1:** Under i.i.d. Rayleigh channel conditions, $\overline{Ψ}(L^*/\overline{Ψ}(N))$ scales as $Θ(N)$.

**Proof:** We can sandwich $\overline{Ψ}(L^*)$ between the optimum throughput capacity achievable per user and $\overline{Ψ}(N/2)$. In [5], authors prove for a median user scheduler, i.e., $L = N/2$, that the total throughput capacity scales as $Θ(N)$. They also show that the optimum achievable system capacity scales as $Θ(N^*)$ (Theorem 3). Thus, $N\overline{Ψ}(L^*)$ scales as $Θ(N)$. In [5], authors also show that $N\overline{Ψ}(N) = Θ(1)$ (Theorem 1). Hence, $(N\overline{Ψ}(L^*))/N\overline{Ψ}(N) = \overline{Ψ}(L^*/\overline{Ψ}(N))$ scales as $Θ(N)$. 

Even for a relatively small multicast group size (e.g., 10 users), we have observed almost twice as much multicast throughput than the baseline method. Serving to the best few users on the other hand turned out to be inferior to the baseline as well due to the fact that receivers are quite rarely scheduled and the increased transmission rate cannot recover this scheduling penalty.

Another important observation is that considering the peak values are not very sensitive to the slight changes around the peak value for a given average SNR, one does not need to tune very precisely to the optimum point and it suffices to identify the average SNR and then a common $L$ can be used over a wide range of multicast group size. For instance, in 10dB case and for $N \geq 10$, we can target around 60% of the receivers at each channel use.

We also examine the throughput capacity when we fix the multicast group size and change the average channel SNR. In Fig. 2, we show the results for a 10-user system with 5dB, 10dB, and 20dB average SNRs. As it might be expected, the capacity gains reduce as channel quality gets better. The peak values also move towards right and optimum strategy tries to schedule more users in each slot as average channel qualities get better. This figure suggest that even for as high as 20dB SNR, we observe a 34% throughput improvement. Nevertheless, the system design should be judicious enough to consider an operational switch to the traditional uncoded packet scheduling with respect to the worst case channel profile when there are not enough number of users in the system and/or SNR quality is high enough. As it will be more clear in the next section, the main penalty we pay for the shown multicast capacity gains becomes the decoding delay.

![Fig. 3. Normalized capacity result for 10-user system under i.i.d. Rayleigh fading. Throughputs are obtained by over 1000 frames of length 1000-slot.](image)

**V. CODING OVER SHORTER BLOCKS: THROUGHPUT-Delay TRADEOFFS**

The throughput gains with the proposed system comes at the expense of decoding delays. We are required to generate encoding block over a very large number of message blocks, strictly speaking infinitely many. In this section, we will discuss the various ways of reducing this decoding delay.

Typical wireless designs consider systematic erasure codes, where first uncoded portion is delivered followed by the parity blocks, as an obvious way of reducing decoding delays. The expectation is that wireless channel (or network) occasionally causes erasures. If no erasures occur, receivers get the original information from systematic blocks without any decoding delay. When erasures occur, most popular erasure codes such as Reed-Solomon and Raptor codes must receive at least $m$ blocks (i.e., the size of the original message) to be able to recover the missing block. When erasures occur in bursts and burst sizes are known a head of time, more optimal encoding strategies have also been proposed [6]. Unfortunately, in our system erasures are not exceptions but they are enforced by the scheduling decisions. As visible from our figures erasures of 20% to 50% are the typical margins we need to consider. The burst of erasures last many slots in a random fashion\(^4\) with large variance at these erasure probabilities, which prevents us to use even the delay-optimized burst erasure codes effectively.

In this paper, rather than focusing on the design of delay-efficient erasure code designs suitable for our scheduler channel, we insist relying on the aforementioned erasure codes.

\(^4\)In fact they are geometrically distributed in our system model.
while focusing on the short-term throughput performances. We divide the coding intervals into frames of \( n \) slots. By this way, we automatically upper-bound the decoding delays by \( \Theta(n) \). Per user throughput expression is already given by (1) and for i.i.d. case it can be written as:

\[
\Psi_i[n] = \frac{1}{n} \sum_{j=0}^{n} I_i[j] C_{L_i}[j]
\]

Notice that the throughput expression is a random variable and it can fluctuate significantly for small \( n \) values. Since unlike before we chopped the original message into segments, applied the erasure encoding for each segment separately, and fit them into scheduler frames, even though some users can get higher throughput than the average, i.e., receive more blocks due to more scheduling or participating in higher rate slots, their goodput (i.e., original message blocks they receive) becomes the same as the worst case user. Therefore, the achievable capacity becomes \( \min_i(\Psi_i[n]) \) in each frame. The long term average consequently can be computed by averaging \( \min_i(\Psi_i[n]) \) over many frames.

We simulated different scenarios to assess the performance losses due to this framing with respect to the long-term capacity results obtained in the previous section. We show the results for a 10-user system at 10dB SNR in figures 3 and 4 for 1000-slot and 100-slot frames, respectively. We plot the average capacity over the frames as well as the standard deviation from this average. We also included 99% throughput curves, i.e. 99% of the time all the users achieve this throughput, on the same figures. Under the same system settings, our results in the previous section indicate a normalized capacity gain close to 2 at the peak value when \( L = 7 \). For a 1000-slot framing, the delay is upper-bounded by 1000 slots at the expense of 10% throughput loss at this peak value to sustain the throughput 99% of the time. To reduce the delay upper-bound to 100 slots, the penalty we pay is more severe corresponding to 27% throughput loss at the 99-th percentile. The good news is that we can still deliver 45% higher throughput than the baseline approach.

Since the capacity losses at the 99-th percentile is lower-bounded by the losses with respect to the standard deviation, we can actually quantify the throughput losses by computing the first and second order statistics of \( \Psi_i[n] \). Using the independence of \( I_i[j] \)'s and \( C_{L_i}[j] \)'s, we obtain:

\[
\mu = E[\Psi_i[n]] = \frac{L}{N} E[C_{L_i}]
\]

\[
\sigma = \sqrt{Var[\Psi_i[n]]} = \sqrt{\frac{1}{n} \left( \frac{L}{N} E[C^2_{L_i}] - \mu^2 \right)} = \Theta \left( \frac{1}{\sqrt{n}} \right)
\]

Hence, while the coding delays get larger as \( \Theta(n) \), the performance losses reduce as \( \Theta \left( \frac{1}{\sqrt{n}} \right) \).

VI. IMPACT OF NON-I.I.D. CHANNEL STATISTICS

In this section, we present the throughput results obtained by solving for the optimization problem given in (5). Here, we slightly loosen up our i.i.d. channel assumption to accommodate the situations where users can see different average signal strengths. In the results to follow, we define different network scenarios for a given multicast group size by randomly picking SNR values in \([5dB,20dB]\) interval for each user per scenario.

Remember that unlike the i.i.d. case, we determine \( L \) target users to be scheduled by looking at the normalized capacities. Users who are not in the top \( L \) spot in terms of the normalized capacities can still be able to receive if they happen to have an achievable channel rate higher than the set value at the transmitter. This mostly likely happens to users with much better SNR values. This results in different long-term
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2008 proceedings.

Using a Raptor code we can operate at any $L$ each scenario with high probability. On the other hand, by an MDS code we can indeed achieve the worst case user’s throughput as well as the max-min capacity of the system in each scenario with high probability. According to the Lemma 2, by using any $L$ we target, the ratio becomes one. As we target less and less users, the ratio gets larger.

In Fig. 6, we plot the average number of users who can actually receive at the selected channel rate irrespective of whether these users were in the targeted subset of users or not. The points in the $y$-axis are normalized with respect to the $L$, e.g., when we target 10 users ($L=10$) actually on the average 12 users can decode the information. Obviously, when we target everyone, the ratio becomes one. As we target less and less users, the ratio gets larger.

We also show the simulation results for 100 user case in Fig. 7 to show the similar tendencies hold for more users. The scenarios however gets tighter around the average since typicality of any given scenario increases as the number of users increases. The max-min capacity occurs at $L^* = 65$, which is again very close to the optimum point for the i.i.d. case. This shows again a tendency of linear relation between the normalized capacity gains at the peak points and the multicast group size. However, a rigorous proof of this observation is not addressed in this paper.

VII. Conclusion and Future Work

In this paper we have investigated the possible multicast capacity improvements in a cellular network setting by jointly utilizing opportunistic multicasting along with fixed rate and rateless block erasure coding. We posed two different max-min optimization problems, one for the case where all users observe the same i.i.d. channel processes and the other for the case where distinct users observe independent but non-identical channel processes. We have shown that in both cases per user multicast throughput has been significantly improved by focusing each transmission onto a proper subset of the multicast users in the system rather than trying to serve all the users at each channel use.

As one of the main results of the paper we have shown that when the channel statistics are i.i.d. Rayleigh, the system throughput normalized by the achievable throughput when all the users are scheduled to receive scales as $\Theta(N)$, i.e., linearly with the number of multicast receivers in the system. We have also obtained exact throughput expressions for the specified scheduling rule for any multicast group size and average channel quality. We validated our numerical results with brute-force system simulations.

In the paper we have further looked into the delay-throughput tradeoffs in a limited fashion by focusing only on the coding over smaller number of blocks. We observed that although it is possible to upper-bound the decoding delay by $\Theta(n)$, we pay a penalty of losses in the capacity gain that is lower bounded by $\Theta(1/\sqrt{n})$. 

![Fig. 6. The average number of users scheduled as a function of the size of the targeted subset. Each point in the figure is normalized with respect to the size of the targeted subset.](image)

![Fig. 7. Normalized capacity result for 100-user system under independent but non-identical Rayleigh fading. Users pick random average SNR values in [5,20]dB interval. Markers show the performance averaged over the worst and best users across 350 different SNR scenarios, dashed and dotted lines indicate the upper and lower values observed around the average curves, respectively.](image)
As for the non-i.i.d. case, we presented a generalization of the approaches in unicasting to the multicast problem. We pose the question as a max-min optimization problem and evaluated the performance via simulations. We observed that none of the users starved for throughput and over the baseline approach it was possible to increase the throughput capacity of each user. We also observed linear normalized capacity gain improvements as a function of the multicast group size. Our findings indicate that a rateless code-based erasure coding over large blocks can be quite instrumental in delivering different throughput for distinct users.

Although our results demonstrate that it is quite compelling to apply opportunistic multicasting along with applying fixed rate or rateless erasure coding at the scheduler or at the application layer, there are a number of critical points that are not addressed in this paper and left out as our immediate future work.

We implicitly assumed that the transmitter can transmit at any channel rate achievable by the users. Although this is not a very limiting assumption given that many advanced communication systems support a large number of channel rates, it is also desirable to evaluate opportunistic multicasting over a more limited number of channel rates.

Some of our findings can be generalized to channel models other than the Rayleigh fading models. However, it is important to evaluate our proposed system over more realistic channel models that at least incorporates temporal and spectral correlations into the channel fading.

Another major assumption in our model is that we assumed perfect estimates for the achievable channel capacities at each receiver in the next channel use. Although we anticipate that opportunistic multicasting should be more robust than opportunistic unicasting, it needs to be validated.

**APPENDIX**

**Proof:** [Lemma 2] The proof for MDS part of the lemma is the same as in Lemma 1, but this time we select $m = (\overline{\Psi} - \epsilon) \times n_e \times \tau$ and $l = (E[\min_{j \in \mathcal{J}_r} C_{ij}] - \epsilon) \times n_e \times \tau$. For Raptor codes, to show the achievability, we will use a large multicast message of size $m$ and compute the finish time by an arbitrary receiver $i$. We will assume that when a receiver completes the download, it leaves the system and a dummy receiver with the same channel statistics replaces it. Therefore, the scheduler transmits $l = (E[\min_{j \in \mathcal{J}_r} C_{ij}] - \epsilon) \times n_e \times \tau$ bits for large enough $n \geq n_e$, for any $\epsilon > 0$, $n_e$ can be made large enough such that user $i$ can also successfully receive $(E[I_i \cdot \min_{j \in \mathcal{J}_r} C_{ij}] - \epsilon) \times n_i \times \tau$ bits in the same period. User $i$ can recover the original multicast message in $n_i$ slots (where $n \geq n_i \geq n_e$) if $E[I_i \cdot \min_{j \in \mathcal{J}_r} C_{ij}] - \epsilon \times (n_i - 1) \times \tau < m \times (1 + \epsilon) \leq (E[I_i \cdot \min_{j \in \mathcal{J}_r} C_{ij}] - \epsilon) \times n_i \times \tau$. Then, the multicast throughput of user $i$ can be computed as $m/(n_i \times \tau) \geq (E[I_i \cdot \min_{j \in \mathcal{J}_r} C_{ij}] - \epsilon')$, where $\epsilon'$ can be made arbitrarily small by picking large enough $n_e$. The message length $m$ can be trivially set as $m = (\overline{\Psi} - \epsilon) \times n_e \times \tau$.

**REFERENCES**


