Collision Helps!
An Analytical Study of ZigZag Decoding

Ali ParandehGheibi, Jay Kumar Sundararajan, Muriel Médard

1 Introduction

The nature of the wireless network is intrinsically different from the wired network because of the shared medium among several transmitters. Such a restriction requires a form of scheduling algorithm to coordinate access to the medium, usually in a distributed manner. The conventional approach to the Medium Access Control (MAC) problem is contention-based protocols in which multiple transmitters simultaneously attempt to access the wireless medium and operate under some rules that provide enough opportunities for the others to transmit. Examples of such protocols in packet radio networks include ALOHA, MACAW, CSMA/CA, etc.

However, in many of contention-based protocols it is possible that two or more transmitters transmit their packet simultaneously, resulting in a *collision*. The collided packets are considered lost in the conventional approaches, but Gollakota and Katabi [2] show how to recover multiple collided packets in a 802.11 system using ZigZag decoding when there are enough transmissions involving those packets. In fact, they suggest that each collision can be treated as a linearly independent equation of the packets involved. Therefore, the packets are recoverable only if the system of equations is full rank. ZigZag decoding provides a fundamentally new approach to handle collisions in a wireless setting without using any central scheduler, or knowledge about the network topology such as number of neighbors, etc. In this project, we wish to understand the effects of this new approach to interference management, in terms of the achievable throughput and delay for the multiple access communication.

We provide an abstraction of the multiple-access channel when ZigZag decoding is used at the receiver. We use this abstract model to analyze the delay and throughput performance of the system in various scenarios.

First, we analyze the scenario when each user has one packet to send. We characterize upper and lower bounds on the expected time to deliver all of the packets. We observe that the mean delivery time of the system with ZigZag decoding is strictly smaller than for a system with a centralized scheduler. Moreover, we provide a connection to matching theory to characterize the decoding process and the exact expected delivery time.

Second, we analyze the throughput of the system in a scenario where packets arrive at each sender according to a Bernoulli process. We characterize the stability region[1] of the system, and propose acknowledgement mechanisms to stabilize the queues at the senders. The stability region of the system with ZigZag decoding is *strictly larger* than that of the system with centralized scheduling.

The rest of this report is organized as follows. In Section 2, we present an abstract model of a system with ZigZag decoding. Section 3 is dedicated to mean delivery time characterization of the system and its comparison to centralized scheduling. In Section 4, we characterize the stability region of the multiple-access channel with ZigZag decoding. Finally, concluding remarks and extensions are discussed in Section 5.

2 System Model

We consider an n-user multiple-access erasure channel (c.f. Figure 1) where ZigZag decoding scheme is implemented at the receiver. Time is assumed to be slotted, and every slot can accommodate one packet transmission. The assumptions on the erasures are explained below. As mentioned in [2], with ZigZag decoding, every collision can be thought of as the reception of a linear equation in the colliding packets. Moreover, ZigZag decoding makes use of the fact that the lack of exact synchronization between successive transmissions means that two different collisions will convey two packets worth of information, even if the set of colliding packets is exactly the same. We model these facts in the form of the following assumption in our system. A successful reception is assumed to be an innovative linear combination of the packets involved in the transmission if and only if not all the packets involved are already decoded.

Nevertheless, because of the fading nature of the wireless channel, not all of packet transmissions result in a successful reception. We consider the following types of erasures to capture reception failures.

1. *Link Erasure*: Each of the individual links from sender i to the receiver

[1] The closure of the set of arrival rates for which there exist a service policy such that the expected length of the queues are uniformly bounded from above as time goes to infinity.

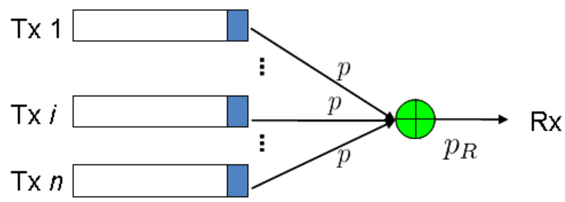

Figure 1: Multiple-access channel with n senders

may get erased independently across links and over time with probability p. If a packet is erased in this manner, then the linear equation at the receiver does not involve the packet sent from sender i. This type of erasure is to model the effect of deep fades at the transmitters, or back-off mechanisms implemented at the senders.

2. *Receiver Erasure*: At each time slot the receiver can receive the potentially collided transmission with probability $1 - p_R$. This model could capture the effect of deep fades at the receiver, or mediocre SNR for one of the transmitters, i.e., the interference is neither weak enough to be treated as noise, nor is it strong enough to perform ZigZag decoding successfully.

In the following sections we provide delay and throughput characterization of the system when ZigZag decoding is implemented at the receiver.

3 Delivery Time Characterization

Definition 1. Consider a multiple-access channel with n senders each having a single packet to transmit (cf. Fig 1). Given a MAC protocol, define the *delivery time*, T_D, as the time to transmit all packets successfully to the receiver.

The goal of this section is to provide upper and lower bounds for the expectation of the delivery time for ZigZag decoding, and to compare it with contention-based protocols and central scheduling mechanism.

For simplicity of the notations, we also assume that receiver side erasures do not take place, i.e., $p_R = 0$. The following results will generalize to the case with $p_R > 0$ as well, if the expected delay bounds are scaled down by a factor of $1 - p_R$. The reason is that with the receiver side erasures, a previously successful reception is now successful only with probability $(1 - p_R)$.

3.1 Centralized scheduling

We assume that the receiver can send acknowledgments upon receiving a packet. With centralized scheduling, we assume the following policy. The channel is initially reserved for sender 1, up to the point when its packet is acknowledged. At this point, the channel is reserved for channel 2, and so on. In this setting, the calculation of the expected delivery time is straightforward. For each sender, the delivery is complete in the first slot when the channel from that sender to the receiver is not under erasure. The delivery time for each sender is thus a geometric random variable, with mean $\frac{1}{1-p}$. This implies that the total expected delivery time under centralized scheduling policy is given by:

$$\mathbb{E}[T_D] = \frac{n}{1-p}.$$

It is important to note that the performance of centralized scheduling is an upper bound on the performance of other distributed backoff based approaches because it ensures that there is no collision. In distributed backoff based approaches, there is always some probability of a collision. We will now derive some upper and lower bounds on the delivery time for ZigZag decoding.

3.2 Bounding the delivery time for ZigZag decoding

Since there are n packets to be delivered, the receiver needs n linearly independent equations (also called degrees of freedom) involving these n packets. We can therefore divide the delivery time into n portions, where the i^{th} portion corresponds to the additional time required to receive the i^{th} degree of freedom, starting from the time of the previous (i.e. $(i-1)^{s}t$) innovative reception.

We define the following notation, for $i = 1, 2, \ldots n$:

$$\begin{aligned} T_i &= \text{Time of reception of the } i^{th} \text{ degree of freedom} \\ X_i &= T_i - T_{i-1} \quad (T_0 \text{ is assumed to be } 0). \\ D_i &= \text{Number of packets that have been decoded after the reception} \\ &\quad\ \text{at time } T_{i-1} \quad (D_0 \text{ is assumed to be } 0). \end{aligned}$$

Note that T_D is then given by:

$$T_D = T_n = \sum_{i=1}^{n} X_i \tag{1}$$

Due to the assumption outlined in Section 2, every transmission is innovative if and only if it involves at least one of the packets that have not yet been decoded. Now, it is easily seen that the number of decoded (and hence undecoded) packets does not change between two successive T_i's. Thus, after T_{i-1} and before T_i, any transmission will be innovative if and only if one of the $(n - D_i)$ undecoded senders is connected. This happens with a probability of $1-p^{n-D_i}$. This leads to our main observation: conditioned on the number of decoded packets, the time till the next successful innovative reception is a geometric random variable, with the probability of success given by

$$1 - p^{(\#\text{ of undecoded pkts})}$$

In other words,

$$(X_i|D_i) \sim \text{Geom}\left(\frac{1}{1-p^{n-Di}}\right)$$

Using this observation, we have:

$$\mathbb{E}[X_i] = \mathbb{E}\big[\mathbb{E}[X_i|D_i]\big] = \mathbb{E}\left[\frac{1}{1-p^{n-D_i}}\right] \tag{2}$$

Now, it is easy to see that $0 \leq D_i \leq (i-1)$. This is because, the number of decoded packets at T_{i-1} cannot exceed the number of received degrees of freedom, which is $(i-1)$. These bounds give the following bounds on $\mathbb{E}[X_i]$:

$$\frac{1}{1-p^n} \leq \mathbb{E}[X_i] \leq \frac{1}{1-p^{n-i+1}}$$

Adding the terms of the above inequality for $i = 1, 2, \dots n$, and substituting from Equation 1, we get:

$$\frac{n}{1-p^n} \leq \mathbb{E}[T_D] \leq \sum_{i=1}^{n} \frac{1}{1-p^{n-i+1}}$$

Intuitively, the lower bound corresponds to the case where D_i remains 0 till the very end. Only at slot T_n, it suddenly jumps to n. This means all the senders remain useful till the end, and hence the delivery happens faster. This situation could happen when the value of p is small, thereby allowing almost all users to collide most of the time. On the other hand, the upper bound corresponds to the case where $D_i = i - 1$. In other words, every time the rank increases by 1, a new packet is in fact decoded, and the corresponding sender is therefore useless for the remainder of the time, from

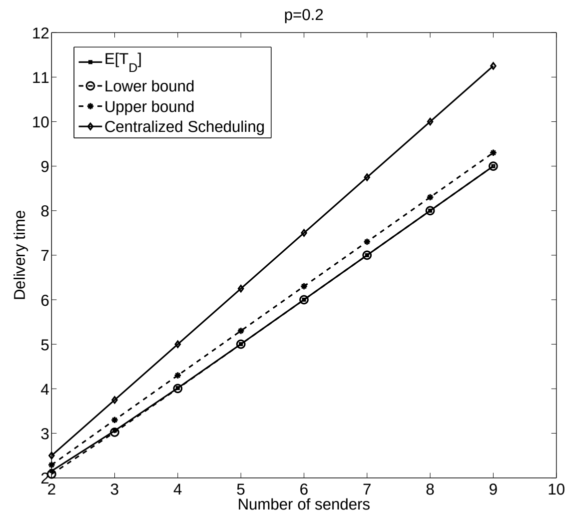

Figure 2: The delivery time for $p = 0.2$

an innovation point of view. This happens if p is so high that usually only one sender transmits, thereby causing immediate decoding.

The simulation plots in Figures 2, 3 and 4 show the actual delivery time, along with the upper and lower bounds. The delivery time of the centralized scheduler is also shown for comparison. There are three plots, showing the delivery time as a function of the number of senders n, for three different values of p – 0.2, 0.5 and 0.9. It can be seen that the actual value of the delivery time approaches the lower bound as n increases. This limit is reached earlier if p is smaller. In particular, even with a 20% loss rate, a value of $n = 3$ brings the performance very close to the lower bound.

3.3 Characterizing the decoding process

It is clear from Equation 2 that the expected delivery time depends on the evolution of the number of decoded packets D_i. In this subsection, we will provide a graph-theoretic characterization of D_i, which we hope will eventually help understand the exact characterization of the expected delivery delay.

To study the evolution of the decoding process, we will use a bipartite graph $G(t)$ to represent the receptions of the receiver up to slot t. One class of vertices in the bipartite graph has n vertices corresponding to the

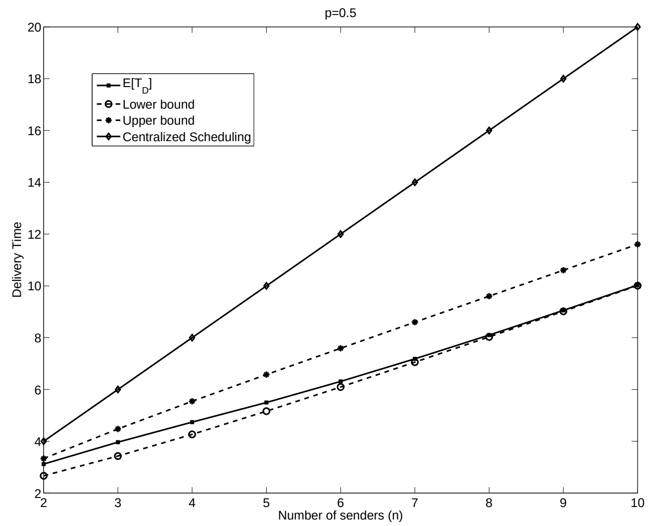

Figure 3: The delivery time for $p = 0.5$

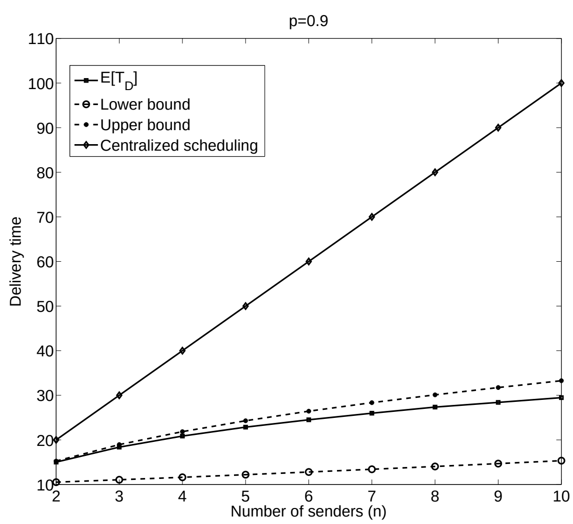

Figure 4: The delivery time for $p = 0.9$

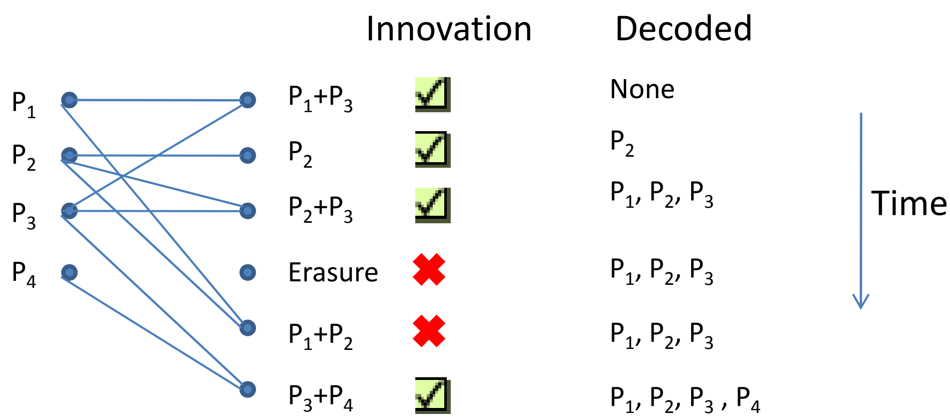

Figure 5: A bipartite graph representation of the receptions

n senders. The other class has t vertices corresponding to the time slots. An edge in the graph connects a sender vertex to a time slot vertex if the sender is connected to the receiver (i.e., does not experience an erasure) in the corresponding slot. Thus, in every slot, a new vertex is added to the time slot class, and is connected to all the senders who transmitted without erasure in that slot. Refer to Figure 5 for an example.

We will also use the following matrix representation $M(t)$ of this graph. The n rows of the matrix correspond to the senders. The t columns correspond to time slots. The entries of the matrix are indeterminate variables. The matrix has an indeterminate variable x_{ij} in row i and column j if the vertex for sender i is connected to the vertex for slot j. If there is no edge, then the matrix has a 0 entry. This representation is known as the Edmond's matrix corresponding to the bipartite graph. All operations on this matrix are performed viewing its entries as elements of the field of rational functions involving the x_{ij} variables, defined over $GF(2)$ as the base field.

Remark 1. *The motivation for using such a matrix is the assumption stated in Section 2, that a reception is innovative if and only if it involves at least one undecoded packet. This assumption means that once we know which packets are mixed in the collision, we do not have to worry about the exact coefficients used. It is as if the packets are being mixed with coefficients randomly chosen from a very large finite field, and hence the probability of linear dependence with what is already known is negligible, as long as at least one undecoded packet is involved. In the limit, this is essentially like picking each coefficient to be an independent indeterminate variable, and* *checking for linear dependence over the field of rational functions involving these variables, with $GF(2)$ as the base field. The following discussion assumes this limiting definition. In other words,* **the delivery is assumed to be complete if and only if $M(t)$ has a rank of n over the field of rational functions**.

We will use the following result known as Edmond's theorem (see Theorem 7.3 in [6]):

Theorem 1. *Let A be the $m \times m$ matrix obtained from a bipartite graph $G(U, V, E)$ with color classes U and V and edge set E as follows:*

$$A_{ij} = \begin{cases} x_{ij}, & (u_i, v_j) \in E \\ 0, & (u_i, v_j) \notin E \end{cases}$$

Define the multivariate polynomial $Q(x_{11}, x_{12}, \ldots x_{mm})$ as the determinant of A. Then, G has a perfect matching if and only if Q is not identically 0.

In our context, this result implies the following corollary, which gives a graph-theoretic characterization of the delivery time:

Corollary 1. *The rank of $M(t)$ is equal to the size of the maximum matching of $G(t)$.*

Proof. Suppose $G(t)$ has a maximum matching T of size r. Consider the submatrix of $M(t)$ with rows corresponding to the those sender vertices and columns corresponding to those time slot vertices, that are matched by T. This $r \times r$ submatrix has non-zero determinant due to Edmond's theorem. Hence, the rank of $M(t)$ is at least r, the size of the maximum matching of $G(t)$.

For the other direction, suppose $G(t)$ has a rank of r. Then, it has a collection of r linearly independent rows. Form a new matrix by retaining only these rows. This matrix also has a rank r, and therefore has r linearly independent columns. Clearly, the $r \times r$ submatrix of $G(t)$ obtained using only these columns also has a rank r. Using Edmond's theorem, there is a perfect matching in the subgraph of the bipartite graph induced by the corresponding sender and time slot vertices. This means, $G(t)$ has a matching of size at least r. Thus, the size of the maximum matching is lower bounded by the rank of $M(t)$. Thus, we have completed the proof. □

Now, we are ready to prove the main result of this section, which characterizes the set of decoded packets.

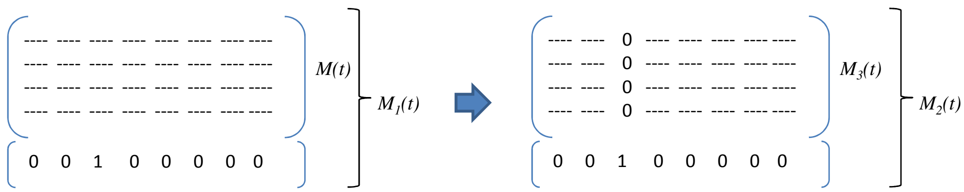

Figure 6: The matrix operations for the proof of Theorem 2

Theorem 2. *A sender's packet has been decoded if and only if the corresponding sender vertex in $G(t)$ is a part of every maximum matching.*

Proof. Consider an arbitrary sender k. Define the indicator vector $\mathbf{e}^{(\mathbf{k})}$ by $e_i = 0$ for all $i \neq k$ and $e_k = 1$. (This is the 1 in the field of rational functions involving the indeterminate variables x_{ij}, over $GF(2)$.)

We now construct the matrix $M_1(t)$ by appending $\mathbf{e}^{(\mathbf{k})}$ to $M(t)$ as a new row. Next, we construct the matrix $M_2(t)$ by performing row reduction on $M_1(t)$, in order to reduce all entries of column k to 0, except in the newly added row. (Note, all operations are performed in the field of rational functions involving the indeterminate variables.)

Since row operations do not change the rank, we have $rank(M_1(t)) = rank(M_2(t))$. Let $M_3(t)$ be the matrix obtained by removing the last (new) row from $M_2(t)$. Since the last row is the only row with a non-zero entry in column k, we have $rank(M_3(t)) = rank(M_2(t)) - 1$. The operations are shown in Figure 6.

It is easily seen that the sender k's packet has been decoded if and only if $\mathbf{e}^{(\mathbf{k})}$ is in the row space of $M(t)$, i.e., if and only if $rank(M(t)) = rank(M_1(t))$, which in turn is equal to $rank(M_2(t))$ and hence equal to $rank(M_3(t)) + 1$, from the above discussion. Now, $M_3(t)$ is the Edmond's matrix for the bipartite graph obtained by removing sender k's vertex from $G(t)$. From Theorem 1, the rank corresponds to the maximum matching size.

This implies that, sender k's packet has been decoded if and only if removing the corresponding vertex from $G(t)$ reduces the size of the maximum matching by 1. Now, this can happen if and only if sender k's vertex is part of every maximum matching. Thus, the proof is complete. □

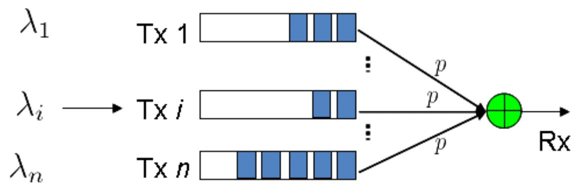

Figure 7: Multiple-access channel with n senders with Bernoulli arrivals

4 Stability Region Characterization

In this section, we consider a scenario when packets arrive at sender i according to a Bernoulli processes with rate λ_i (cf. Figure 7). We assume that the arrival processes at different senders are independent, and error-free feedback is available at each time slot. For simplicity of the notations, we also assume that receiver side erasures do not take place, i.e., $p_R = 0$. The following results generalize to the case with $p_R > 0$ by scaling the region down by a factor $1 - p_R$.

A *centralized scheduling policy* involves choosing at most one of the senders for transmission (service) so that any collision is avoided. If the packet is delivered successfully at the receiver, an acknowledgment is fed back to the sender and that packet is dropped from the sender's queue. The centralized scheduler requires coordination among the senders as well as information about the queue-length or the arrival rates. However, it does not have access to channel state before it is realized. Therefore, probability of packet loss is independently at least p at every time slot, and it is also independent of the implemented centralized scheduling policy. Thus, we have the following *necessary* conditions for the stability region:

$$\begin{aligned} \sum_{i=1}^{n} \lambda_i &< 1 - p, \\ \lambda_i &\geq 0, \quad i = 1, \ldots, n. \end{aligned} \tag{3}$$

In fact, it can be shown that the above conditions are also sufficient. The queues can be stabilized by a centralized scheduling policy that selects the sender with the longest queue for transmission [3]. In summary the stability region for centralized scheduling policies is a simplex given by (3). An example of such region for a two-user system is illustrated in Figure 8(a).

Note that the centralized scheduling policy may allocate the media to a sender whose channel gets erased during the transmission, and hence, wastes time slots even if there are other senders that are not suffering from

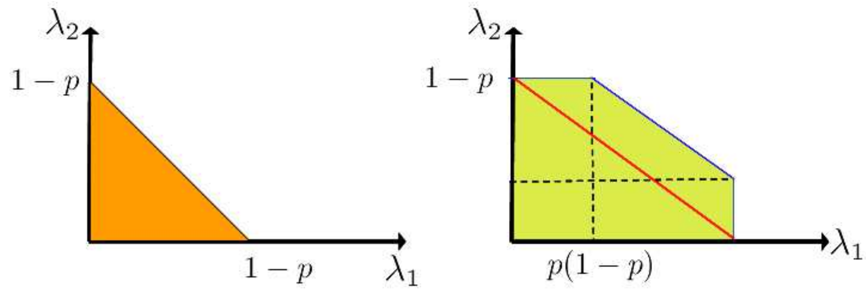

Figure 8: Stability region of a two-user multiple-access channel with (a) centralized scheduling (b) ZigZag decoding.

an erasure. However, if the realization of the channel state in the next time slot is known, such wastes can be avoided by choosing the transmitter from those that are connected to the receiver. Tassiulas and Ephremides [4] show that if information about channel state realization is available a priori, the following set of arrival rates are admissible:

$$\begin{aligned} \sum_{i \in S} \lambda_i &< 1 - p^{|S|}, \quad \text{for all } S \subseteq \{1, \ldots, n\}, \\ \lambda_i &\geq 0, \quad i = 1, \ldots, n, \end{aligned} \tag{4}$$

where $|S|$ denotes cardinality of set S. The region described in (4) can be achieved by serving the sender with longest queue-length among those that are connected to the receiver. Moreover, Tassiulas and Ephremides [4] show that it is not possible to stabilize the queues for any point outside the region described in (4). This can be seen as a consequence of *Cut-Set bound* (cf. [5]) applied to the multiple-access channel. The stability region for a two-user system is illustrated in Figure 8(b).

In the following, we show how to use ZigZag decoding scheme to achieve the dominant face of the stability region given in (4) without prior knowledge about channel state realizations.

Definition 2. The *priority-based* policy for a multiple-access channel is as follows. Fix a priority order of the senders with 1 being the highest priority.

- Transmission mechanism: Each sender transmits the head-of-line packet of its queue at every time slot

- Acknowledgement mechanism: Upon every reception, the receiver acknowledges the packet from the sender with highest priority among those packets that are involved in the reception. Consequently, each acknowledged packet is dropped from the corresponding sender's queue.

In the following, we show the priority-based policy can achieve vertices of the stability region given by (4). First, let us provide a simple characterization of the vertices of the dominant face of the region.

Lemma 1. *There exists a one-to-one correspondence between permutations of $\{1, \ldots, n\}$ and vertices of the dominant face of the region described in (4). In particular, for any permutation π, the corresponding vertex is given by*

$$\lambda_{\pi_i} = (1-p)p^{i-1}, \quad i = 1, \ldots, n.$$

Proof. See [7]. □

Theorem 3. *Any vertex on the dominant face of the region given by (4) can be achieved without prior knowledge about channel state realization by employing ZigZag decoding at the receiver.*

Proof. Fix a vertex, V, on the dominant face of the stability region. By Lemma 1, it corresponds to a permutation π of the senders. Without loss of generality, assume $\pi = (1, 2, \ldots, n)$. The rate-tuple corresponding to V is given by

$$\lambda_i = (1-p)p^{i-1}, \quad i = 1, \ldots, n. \tag{5}$$

Next, we show the priority-based policy defined in Definition 2 can achieve the vertex V. Let μ_i be the probability of acknowledging a packet from sender i at each time slot. Sender i is acknowledged if and only if the packet sent from i is not erased and all of the packets from senders with higher priority are erased. By independence of the erasures across links we obtain

$$\mu_i = p^{i-1}(1-p).$$

Note that an acknowledgement to sender i is equivalent to serving the queue at sender i by one. Hence, by independence of the erasures across time, μ_i is also the service rate of the queue at sender i. Therefore, for the arrival rates arbitrarily close to that of vertex V (see (5)), the sender side queues are stable. It remains to show that such policy results in successful decoding of the packets at the receiver. Note that every successful reception at the receiver is innovative. Suppose such reception involves k packets from

k of the senders. In order to decode these k packets, k linearly independent combination of such packets is required. A sufficient way to construct k linearly independent equations is to have $(k-1)$ equations that are linearly independent, but they do not involve the packet from the sender with the highest priority. This means that the packet from the sender with highest priority is not required for decoding to happen, and it can be dropped from sender's queue. Finally, the packets sent to the receiver will be eventually decoded because all of the senders's queues become empty infinitely often, i.e., no more degree of freedom is required. □

Corollary 2. *The dominant face of the stability region described in (4) is achievable without prior knowledge about channel state realization by employing ZigZag decoding at the receiver.*

Proof. Every point on the dominant face of the stability region can be written as a convex combination of the vertices of the dominant face. Moreover, each vertex can be achieved by a priority-based policy given in Definition 2, corresponding to that vertex. Therefore, every point on the dominant face can be achieved by time sharing between such policies. Note that the difference between he policies achieving different vertices is in the acknowledgement mechanism which takes place at the receiver, and no coordination among the transmitters is necessary. □

The priority-based requires knowledge of the arrival rates at the receiver to tune the acknowledgement mechanism. However, if the queue-length information at the receiver is available, we can mimic the policy by Tassiulas and Ephremides [4] by acknowledging the sender with longest queue. Achievability of the stability region in (4) is then a direct consequence of the results in [4].

5 Conclusions and Extensions

In this project, we have studied the impact of allowing ZigZag decoding on the throughput and the delay in the context of a collection of senders transmitting data over a multiple-access channel to a single receiver. We have focused on two situations – the completion time for each sender to deliver a single packet to the receiver, and the rate region in the case of streaming arrivals. Our conclusion is that ZigZag decoding achieves significant improvements in both the completion time as well as the rate region. Modulo the implementation constraints, ZigZag decoding is thus a promising new way to handling interference in wireless networks.

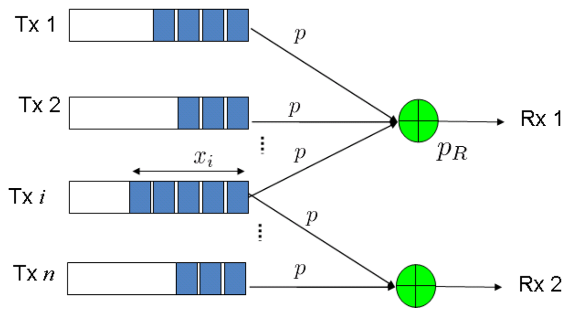

Figure 9: Generalized wireless network model including multiple-access and broadcast channels with erasures

Several generalizations of this work are possible. First, as mentioned before, the above models can be readily extended to the case where there are both link erasures and receiver erasure.

So far, we have relied on the channel to perform the "linear combinations". The next step would be to allow the senders themselves to send linear combinations of the packets that are in the transmission queue. In this case, the interesting question would be about the coding mechanism at the sender and the acknowledgement mechanism at the receiver so that delivery time is minimized. Coding across packets before transmission is particularly interesting if the model is further generalized to multiple receivers (cf. Figure 9), in which a sender can broadcast the packets to the receivers. The goal would be to characterize the delivery time at each receiver in terms of the amount of contention and load per receiver. Another interesting question is the rate region in the case of streaming arrivals, when we have multiple receivers.

References

[1] R. Koetter and M. Médard, *An Algebraic Approach to Network Coding*, IEEE/ACM Transactions on Networking, vol.11, no.5, pp. 782-795, Oct. 2003.

[2] S. Gollakota and D. Katabi, *ZigZag Decoding: Combating Hidden Terminals in Wireless Networks*, ACM SIGCOMM, 2008.

[3] L. Georgiadis, M. J. Neely and L. Tassiulas, *Resource Allocation and Cross-Layer Control in Wireless Networks*, Foundations and Trends in Networking, Vol. 1, no. 1, pp. 1-144, 2006.

[4] L. Tassiulas and A. Ephremides, *Dynamic server allocation to parallel queues with randomly varying connectivity*, IEEE Transactions on Information Theory, Vol. 39, No. 2, pp. 466-478, 1993

[5] T. Cover and J. Thomas, Elements of Information Theory, Wiley & Sons, New York, 1991. Second edition, 2006.

[6] R. Motwani and P. Raghavan, Randomized Algorithms, Cambridge University Press, Ninth edition, 2007.

[7] R. E. Bixby, W. H. Cunningham, and D. M. Topkis: Partial order of a polymatroid extreme point, Math. Oper. Res., 10 (1985), 367-378.