Dynamic Wireless Network Coding with Overhearing and Variable Channel Rates

Constantinos Fragiadakis[†*] Georgios S. Paschos[§] Leonidas Georgiadis[‡*] Leandros Tassiulas[†*]

[*]Informatics & Telematics Institute, CERTH, Greece,
[†]Dept. of Computer & Communication Eng., University of Thessaly, Greece,
[‡]School of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Greece,
[§]Mathematical and Algorithmic Sciences Lab, France Research Center, Huawei Technologies Co. Ltd.
kofragia@uth.gr, georgios.paschos@huawei.com, leonid@auth.gr, leandros@uth.gr

***Abstract*—We study a 1-hop broadcast channel with two receivers. The receivers have side information obtained by overhearing wireless channels. The relay takes control decisions by coding transmissions based on its knowledge of side information in the receivers. We consider two control mechanisms. In the ACK system, the relay has definite knowledge of side information announced via overhearing reports. In the NACK system, the relay has statistical knowledge of side information and receives feedback after every decoding failure.**

Our contribution is as follows. We provide the minimal evacuation times for the two systems and obtain analytical expressions of the throughput region for the ACK and the code-constrained region for the NACK system. When the transmission rates are the same ($r_1 = r_2$), or when the receiver with the highest transmission rate has perfect side information ($p_f = 1$), we show that the two regions are equal. We then provide simple joint XOR coding and scheduling policies that achieve those regions, and thus are throughput optimal. Subsequently, we evaluate the report overhead performance for both mechanisms and reflect on the involved tradeoff with throughput. Ultimately, we demonstrate by simulations that the proposed throughput optimal policies can be appropriately enhanced to have good delay properties, especially for protocols that utilize sequenced packet delivery.

***Index Terms*—wireless network coding, delay analysis, 1-hop broadcast channel, partial overhearing information, stochastic control.**

I. INTRODUCTION

Wireless interflow network coding mixes information from different flows and increases the performance of multiple unicast flows. This approach has been embraced by the research community in works like COPE [1], [2] and many followup works (e.g. [3]–[6]) but important considerations prevent this method from finding acceptance in the industry. One associated problem is *how the coding node can be efficiently informed of the content of the decoding buffers of the receivers*. This is the motivation behind our work, where we consider two approaches to solve this problem.

The first approach [1] is based on reporting overhearing to all neighbors via an ACKing mechanism. This scheme suffers from the large amount of messages that need to be circulated when the number of receivers and neighbors is high, but this problem can be partly mitigated by compressing ACK packets in batches. A concern about this approach is how timely those reports can be under very high transmission rates.

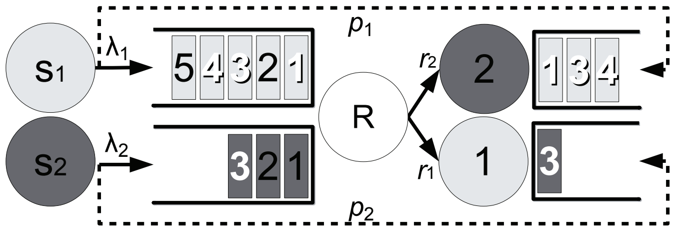

Fig. 1. The system under consideration; packets from two unicast flows arrive at the coding node *R* and are destined to two different receivers. Due to side overhearing channels, a copy of the arriving packet, destined to one receiver, also arrives at the other with a probability.

The second approach, also due to [1] (and studied further in [7], [8]), is based on statistical information and reports on decoding failures. This reduces the number of reports required at the expense of throughput performance, since the coding node makes decisions oblivious to the actual random overhearing events [9].

Consider the example of Fig. 1. Two non-symmetric unicast flows are defined, $f_1 : s_1 \to 1$ and $f_2 : s_2 \to 2$. Both flows use the relay node *R* as a forwarder, which employs interflow network coding by XORing packets from the two flows. The receivers 1 and 2 utilize the overhearing erasure channels to obtain packets destined to the other receiver. For example, receiver 1 receives packets destined to receiver 2, with probability p_2, whenever the source s_2 sends them to *R*. Each receiver i can accept transmissions at a certain transmission rate r_i or lower.

We focus on the downlink part which entails the complexity of the problem; node *R* must make coding and scheduling decisions in order to achieve some objectives, e.g. maximize throughput. In this context, we will call *ACK* the system where node *R* learns the content of the decoding buffers of $1, 2$ via explicit reports that follow each overhearing event and *NACK* the system where the decisions are made based on the probabilities of the overhearing channels and feedback reports following each unsuccessful attempt. Our contributions are summarized as follows:

1) We give an outer bound for the throughput region of the ACK system assuming general coding (including

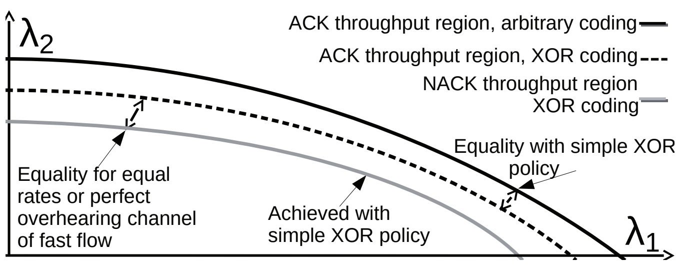

Fig. 2. Summary of our contributions regarding throughput regions and policies that achieve them. We characterize the throughput region with arbitrary coding for the ACK system and achieve it with a simple XOR policy. We characterize the throughput region with XOR coding for the NACK system and achieve it with a simple XOR policy. We finally show the cases when both regions are equal.

non-linear coding)-the equivalent information theoretic capacity region is shown in [10]. We show that this region can be achieved by simple XOR policies which operate without knowledge about arrivals.

2) For the NACK system, we give in closed-form the code-constrained throughput region by constraining the system to the use of XOR coding. We propose a simple evacuation coding policy which achieves it. Interestingly, for the case of equal rates $r_1 = r_2$, or when the receiver with the highest rate has perfect overhearing (i.e. $p_f = 1$, where $f = \arg\max_i r_i$) this region is identical to the throughput region of the ACK system. Fig. 2 gives an illustration of the above results.
3) We prove that the XOR policies proposed to achieve maximum throughput have the additional property of evacuating a snapshot of packets in minimum time.
4) We study performance tradeoffs between the two systems for a range of system parameters comparing the throughput efficiency and the report overhead.
5) We additionally demonstrate via simulations that appropriate enhancements of these policies have good delay properties. Here, we measure delay as the difference between the largest packet ID in the system and the last consecutive packet ID among departed packets. This measure is especially important for protocols that utilize sequenced packet delivery, like TCP.

In a nutshell, we provide optimal solutions that utilize simple XOR operations, require minimal information about system state, are oblivious to arrivals and can be embraced by resource limited wireless devices.

II. Discussion of Modeling assumptions

Our model studies the system with two flows. This is not a limiting assumption, given that in practical wireless networks, the vast majority of interflow coding opportunities involve a small number of nodes, with simple pairwise XOR coding providing most of the throughput gains [11]–[13] and without the decoding hassle of coding a large number of packets (which can be significant, see [14]).

Our model currently assumes a perfect uplink channel, but it can easily be expanded to include erasures in the uplink. To do so for the channel of flow 1, we assume a probability of erasure in the direct channel q_1 and a probability of erasure in the overhearing channel u_1. The system retransmits the uplink packet after an erasure in the channel. Since the uplink transmission is essentially a Bernoulli trial experiment, it is possible to replace such a model with a perfect uplink channel and a modified overhearing probability. It can be found through probability theory that the probability of overhearing after correct reception in the relay is given by $p_1 = \frac{1-u_1}{1-q_1 u_1}$. The same analysis is applicable for flow 2. Given this formula for calculating the equivalent overhearing probability, the erasure uplink channel can be captured while the subsequent analysis remains unchanged.

Finding the throughput region for a downlink erasure channel, even when the probability mass function of an erasure event is a simple i.i.d Bernoulli trial, is more complicated. Consider an optimal policy for the channel without erasures (as the ones we introduce in this work), with the addition of retransmission in the event of erasure at the channel of the intended receiver. Although we believe that this policy-plus-retransmissions is optimal for the policy we introduce in the ACK system, its usage is not obvious in the case of the NACK system. For another policy for the ACK system and a presentation of the complications of this problem, see [15]). Due to these difficulties, we leave the study of erasures in the downlink channel for the future and concentrate on the problem without erasures.

We note, however, that if we assume symmetric erasures, i.e. if both direct downlink channels are either on or off due to an erasure, then our model is still valid.

III. System model

We begin by providing the basic concepts and definitions common to both ACK and NACK systems. We then describe the additional details for each specific system.

A. Common Model

System. Consider a 1-hop broadcast network with one transmitting node *R* (a coding relay) and two receivers $1, 2$. Time is slotted, where slot t occupies the time interval $[t, t+1)$. Packets arrive at *R* at the beginning of each slot. For every slot, *R* makes a control decision that determines the transmissions of *R* during the slot.

Packets. Each packet is defined by a positive integer indicating its *packet ID* and it is additionally classified by its destination, namely receivers 1 or 2. We also define flow i as the set of packets destined at receiver i, $i \in 1, 2$.

Arrivals. The packets arrive according to a stochastic arrival process with rate λ_1 and λ_2 correspondingly, see Fig. 1. We assume i.i.d. packet arrivals within each slot.

Overhearing. Whenever a packet of flow 1 (2) arrives at the relay *R*, a copy of it arrives at receiver 2 (1) with a probability p_1 (p_2). This probability corresponds to random overhearing events which are independent from one another. A packet is called *good* when it is overheard, otherwise it is called *bad*. The relay is aware of p_1, (p_2).

Storage. *R* stores arriving packets in the input queues (hereinafter *queues*) while the receivers store packets useful for decoding in the decoding buffers (hereinafter *buffers*). For the latter, the stored packets are either overheard, or previously received coded packets which haven't left the system yet. An example is given in Fig. 1. The number of queues and their properties are determined by the specific system under consideration (ACK or NACK).

Rates. Receiver i is associated with a transmission rate r_i, $i \in 1, 2$ which is the maximum number of packets transmitted by *R* such that i will correctly receive all transmissions. Reception corresponds to PHY layer operations, e.g. demodulation.

XOR Coding and Decoding. The relay *R* can perform XOR coding, i.e. bitwise modulo-2 additions of two packets x_1, x_2 denoted by $x_1 \oplus x_2$. The decoding is straight-forward if both the XORed packet and one of the native packets involved in the combination are known to the receiver; e.g, applying a XOR addition on x_2 and $x_1 \oplus x_2$ provides x_1.

Controls. For every slot t, *R* chooses a control $c(t)$ which corresponds to a packet encoding and a transmission rate is determined. If the encoding is directed to receiver i only (*singleton* control), then exactly r_i packets are transmitted at t. If the encoding is directed to both receivers (*pair* control), then $\min\{r_1, r_2\}$ packets are transmitted so that both receivers can decode the message. Whenever there are not enough packets to send with the chosen rate, *dummy* non-informative packets are used to fill in this number. Control specifics are further determined by the ACK and NACK systems.

Departures. We assume whenever a packet is obtained in native form (i.e uuencoded) by the intended receiver, the packet and all coded functions of it depart the system. The packet can be obtained either by the direct transmission of the packet from *R* using a singleton control, or after a XOR decoding.

Finally, we assume the uplink and downlink broadcast channel is erasure-free and we discuss this assumption in section II.

B. ACK System

In the context of the ACK system, whenever there is an overhearing event, an acknowledgment (ACK) is transmitted to *R*. Therefore, *R* is aware of whether a packet is $good/bad$. Notice that knowing the probabilities of overhearing p_1 and p_2 is redundant in this system. See Figure 3 for an example.

Queues. We use four queues to classify packets upon arrival, named $Q_1^g, Q_1^b, Q_2^g, Q_2^b$, based on whether they are *good/bad* (g/b) and their flow (1/2).

State. The state of the system at time slot t is $S_{det}(t) = (k_1, k_2, n_1, n_2)$, where k_1 (k_2) is the number of packets of flow 1 (2) in all queues, and n_1 (n_2) is the number of packets in Q_1^g (Q_2^g) that reside in the queues of the relay. Choosing this particular state representation is important for our analysis.

Controls. The control set is defined as

$$\mathcal{C}_{det} \triangleq \{g_1, b_1, g_2, b_2, g_1 \oplus g_2\}$$

where, for example, control g_1 denotes the transmission of r_1 packets from queue Q_1^g. The control $\{g_1 \oplus g_2\}$ is directed to both receivers (sent at rate $\min\{r_1, r_2\}$) and the controls $\{g_i, b_i\}$ are directed to receiver i (sent at rate r_i), $i = 1, 2$. Note, that we omit controls that apply XORs on bad packets. Although this is a constrained control set, it can be shown [8] that optimal performance can be achieved using this set.

Policies. A *policy* is a rule that maps the system state at the beginning of slot t to a control $c(t) \in \mathcal{C}_{det}$.

C. NACK System

In the context of the NACK system, the controller does not have definite knowledge about whether a packet is bad or $good$ and hence it uses the knowledge of probabilities p_1, p_2 to make decisions. Additionally, we assume a feedback mechanism which corresponds to transmissions of NACK packets from the receivers to *R* whenever a receiver fails to decode a given transmission. We assume that at the end of each slot, all feedback corresponding to the transmissions of the slot will be gathered. Feedback is then used to determine whether a packet is $good$ or bad.

Decoding Failure and NACK Feedback. Singleton packets are always decoded by the receivers. Therefore, a decoding failure always corresponds to a XOR transmission. Suppose that receiver 1 cannot decode a XOR packet, and sends a NACK message. *R* can then infer that receiver 1 did not decode, and moreover it has not previously overheard the packet destined to receiver 2, i.e. the pairing packet is a bad packet. Due to the feedback rules described, if one NACK is received, a bad packet leaves the system and a $good$ packet of the flow whose receiver sent the NACK stays in the system. If two NACK messages are received following a transmission, then both packets stay in the system and characterized as bad.

Queues. We use six queues to classify packets. Upon arrival, the packets of flow i are pushed in a queue named Q_i^u also called *unknown* queues since the *good/bad* state of the packets in those queues is not known. When the state of a packet becomes known through feedback, it is popped from Q_i^u and pushed to queues Q_i^g, or Q_i^b, whose semantics are the same as in the ACK system. See Fig. 3 for an example.

State. The system state is $S_{sto}(t) = (k_1, k_2, n_1, n_2, m_1, m_2)$, where k_i is the total number of packets of flow i in the queues, n_i the number of packets in Q_i^g and m_i the number of packets in Q_i^b.

Controls. The control set is defined as

$$\mathcal{C}_{sto} \triangleq \mathcal{C}_{det} \bigcup \{u_1, u_2, g_1 \oplus u_2, u_1 \oplus g_2, u_1 \oplus u_2\}$$

The set is again constrained to exclude XOR controls involving packets that are known to be bad. This happens without loss of optimality, a proof is omitted for brevity. Controls $\{g_i, u_i\}$ are directed to receiver i (as before), while the rest of the controls are directed to both receivers. It is important to notice that, given their initial classification as $unknown$ packets, the only way for packets to be enqueued in the Q_i^b queues is after a $u_1 \oplus u_2$ transmission were both $unknown$ packets are bad, in which case *both* packets are moved to their corresponding queues $Q_i^b, i \in 1, 2$. Also notice that a future transmission b_1 (or b_2) with rate $min(r_1, r_2)$ will make both bad packets depart, due to the fact that we store the previous XOR combination at the buffers of the receivers. Beside the above special transmission, all other controls manage to cause the departure of the bad packets they code.

Policies. A *policy* is a rule that maps a system state at the beginning of slot t to a control $c(t) \in \mathcal{C}_{sto}$.

IV. Stability considerations

Consider the set of queues at the coding node, denoted Q. Denote the sum of backlogs of queues in Q under policy σ at

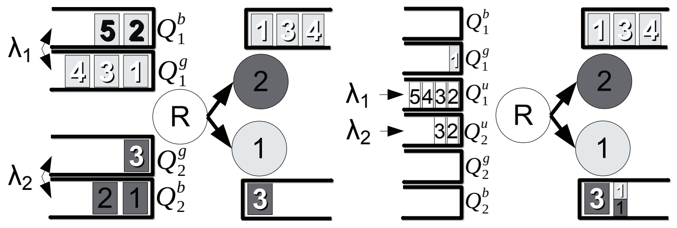
Fig. 3. (left)The example of Fig 1 under the ACK system. (right) The same example under the NACK system and after the control $u_1 \oplus u_2$ is used with $r = 2$. Packets are classified as *good* (overheard), *bad* (not overheard) or *unknown* (no knowledge of overhearing event and only in the NACK system). For the NACK system, all packets are initially classified as unknown and some of the gradually move to *good* or *bad* queues, according to the received NACKs.

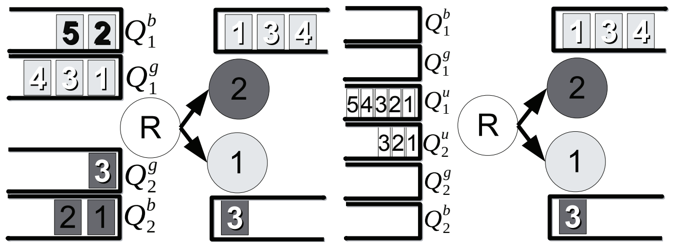
Fig. 4. System instances of the ACK and NACK cases under the same initial arrivals. (left) An instance of the ACK system. The *good*/*bad* state of all packets that arrived is known. (right) An instance of the NACK system. The *good*/*bad* state of the initial arrivals is not known, therefore all packets are enqueued to Q_i^u queues.

the end of time slot t as $X_i^\sigma(t)$. As in [16], we say that the system is stable if:

$$\lim_{q\to\infty} \limsup_{t\to\infty} \Pr\left(X_i^\sigma(t) > q\right) = 0.$$

Note that the definition of stability does not include the buffers. Due to the definition of departures, though, stability of queues implies stability for the buffers.

Consider the set of all vectors $\boldsymbol{\lambda} = (\lambda_1, \lambda_2)$ for which the system is stable under policy σ; the closure of this set denoted by Λ^σ is called the *stability region* of the policy σ. The region $\Lambda \triangleq \cup_\sigma \Lambda^\sigma$ characterizes the system and is called the *throughput region*. In case we constrain the allowable set of codes (e.g. to XOR only) we will refer to the corresponding region as the *code-constrained throughput region*, see [17].

We expect the code-constrained region of the NACK system to be a subset of the throughput region of the ACK (see Figure 2) due to the partial overhearing information available at the relay and the restriction to XORing.

V. Evacuation Time and Stability

In order to study the throughput regions and gain insight into the form of the policies that can achieve throughput optimality, we consider a special operation of the system, which is based on evacuating system *instances*. We plan to use prior work [16] which connects the evacuation times of such instances to the system throughput region. In particular, we will derive analytic expressions for the regions of interest by means of calculating the evacuation time of an arbitrary instance in each system.

For the ACK case, we define as a system instance a realization of the random tuple (K_1, K_2, M_1, M_2), where K_i is the total number of packets at the input destined to receiver i and M_i is a binomial RV with parameters (K_i, p_i) indicating how many packets out of K_i are good. For the NACK case, we define a system instance as a realization of the random tuple $S = (K_1, K_2, 0, 0, 0, 0)$ where $K_i, i \in \{1, 2\}$ is the total number of packets at the input destined to receiver i . For both systems, the packets of the instance enter the system at $t = 1$ and no further arrivals are introduced for $t > 1$. Both the ACK and NACK instances correspond to an equivalent “initial condition” where the packets are at the input queues with knowledge about the overhearing that corresponds to the capabilities of each system [16]. Figure 4 shows an example.

An admissible *evacuation policy* π is any rule for deriving a finite sequence of eligible control actions at the end of which all packets of any possible instance have departed from the system. An example of a sequence of eligible control actions that evacuate the ACK instance of Figure 4 with $r_1 = r_2 = 1$, could be $\{g_1 \oplus g_2, b_1, b_1, b_2, b_2, g_1, g_1\}$ corresponding to packet transmissions $\{x_1 \oplus y_3, x_2, x_5, y_1, y_2, x_3, x_4\}$, where x_i is a packet of flow 1 and y_i a packet of flow 2. This sequence uses 7 slots, which turns out to be the minimum evacuation time in this example, in the sense that any other sequence of controls cannot evacuate this instance in less than 7 slots. Correspondingly, for the NACK system, an example could be $\{u_1 \oplus u_2, g_1 \oplus u_2, g_1 \oplus u_2, u_1, u_1, u_1, u_1, g_1\}$ corresponding to packets $\{x_1 \oplus y_1, x_1 \oplus y_2, x_1 \oplus y_3, x_2, x_3, x_4, x_5, x_1\}$ and 8 slots. An example of an evacuation policy could be the rule: ”send all packets singleton” or ”apply coding whenever there are two packets that belong to different flows, otherwise send the packets singleton”. The former is admissible, since any instance will eventually be evacuated. The latter is not admissible since a possible instance is that which contains two packets from different flows that are both *bad*. The latter policy will never evacuate those packets since it will always XOR code them and the receivers will fail to decode the combination.

A. Infimum of Average Evacuation Time

We will now present a formal definition for the *infimum of average evacuation time*. Consider the ACK system. Let (k_1, k_2, N_1, N_2) be a system instance and $\pi \in \Pi$ be an admissible evacuation policy where Π is the set of all admissible evacuation policies. We denote with $T^\pi(k_1, k_2, N_1, N_2)$ the *evacuation time* of policy π, which is the number of slots required to evacuate the instance under policy π. T^π is random because a policy could randomly choose control sequences in general. Therefore, we define $\overline{T}^\pi(k_1, k_2) \triangleq \mathbb{E}[T^\pi(K_1, K_2, M_1, M_2)|K_1 = k_1, K_2 = k_2]$ to be the average evacuation time of an admissible policy π over all possible instances for a system with k_1, k_2 packets at the inputs and over all possible control sequences generated by π. Finally, we define $\overline{T}^\star(k_1, k_2) \triangleq \inf_{\pi\in\Pi}\{\overline{T}^\pi(k_1, k_2)\}$ the infimum of average evacuation time over all admissible policies for a system with k_1, k_2 packets at the inputs.

Now consider the NACK system. Let $(k_1, k_2, 0, 0, 0, 0)$ be a system instance and $\pi \in \Pi$ be an admissible evacuation policy where Π is the set of all admissible evacuation policies. We denote with $T^\pi(k_1, k_2, 0, 0, 0, 0)$ the *evacuation time* of policy π and $\overline{T}^\pi(k_1, k_2) \triangleq \mathbb{E}[T^\pi(K_1, K_2, 0, 0, 0, 0)|K_1 = k_1, K_2 = k_2]$ the average evacuation time of π as before. The infimum is defined as above. The use of infimum is necessary to allow the optimal evacuation time to be achieved in the limit by a sequence of policies, and hence to avoid constraining the consideration to sets of policies with minimum evacuation times.

B. Epoch-Based Policies

Let $\pi \in \Pi$ be an admissible evacuation policy for a system instance where Π is the set of all admissible evacuation policies. We will describe a way to use π to form a policy $\sigma(\pi)$ admissible in the system with arrivals. We call such policies *epoch-based*.

The *initial system instance* is defined as the system instance created by the arrivals in the system at $t = 0$; denote the instance that occurs after arrivals as I^0. The *first epoch* or *epoch 0* is defined as the time needed to evacuate the initial system instance under policy π, which is $T^\pi(I^0)$. If no arrivals have occurred, then $T^\pi(0) = 1$ by convention. All further arrivals during the evacuation are blocked from entering the system and temporarily stored; denote this set of packets as I^1. The *epoch j* is recurrently defined as the time needed to evacuate the system instance that is created by the packets that arrived during epoch $j-1$, i.e $T^\pi(I^{j-1})$.

An $epoch - based$ policy for the system described in III, denoted as $\sigma(\pi)$ is defined as a policy that evacuates the system in epochs, as described previously, using policy π [16].

C. Throughput Region and Optimal Policies

Finally, we present the theorems of [16] that form the basis of our analysis, namely characterizing the efficiency of the studied systems, and proving that optimal evacuation policies can be used to define optimal epoch-based policies for our model.

LEMMA 1 [SUBADDITIVITY AND LINEAR GROWTH]: *The function $\overline{T}^\star(k_1, k_2)$ is subadditive, is upper bounded by a linear function and the following limit exists*

$$\hat{T}(\lambda_1, \lambda_2) = \lim_{t\to\infty} \frac{\overline{T}^\star(\lceil t\lambda_1 \rceil, \lceil t\lambda_2 \rceil)}{t}.$$

Proof: In [16], Lemma 1 is shown under a general class of policies, provided that these policies have certain features and under some assumptions on system operation. Most of them hold trivially in our system. Assumption 5) in [16] can be verified by considering a simple policy that evacuates all packets in the system in a random order using native transmissions. This policy evacuates the system in exactly $\left\lceil \frac{k_1}{r_1} \right\rceil + \left\lceil \frac{k_2}{r_2} \right\rceil < \frac{k_1}{r_1} + \frac{k_2}{r_2} + 2$ slots, thus 5) is satisfied. The same policy can be used to show 6). ■

The following are consequence of lemma 1:

PROPOSITION 2 [THROUGHPUT REGION VIA EVACUATION TIMES FROM [16]]: *The throughput region is the set of rates $\lambda_1, \lambda_2 \geq 0$ satisfying $\hat{T}(\lambda_1, \lambda_2) \leq 1$.*

PROPOSITION 3 [THROUGHPUT OPTIMAL POLICIES VIA EVACUATION POLICIES FROM [16]]: *Suppose that for an evacuation policy π we have*

$$\limsup_{t\to\infty} \frac{\overline{T}^\pi(\lceil t\lambda_1 \rceil, \lceil t\lambda_2 \rceil)}{t} = \hat{T}(\lambda_1, \lambda_2), \quad \forall\ \lambda_1, \lambda_2 \qquad (1)$$

then, the epoch-based policy $\sigma(\pi)$ is throughput optimal.

In the next two sections we use propositions 2 and 3 to characterize the performance of the ACK and NACK systems. Moreover they motivate us to derive epoch-based policies which achieve maximum performance.

VI. ANALYSIS OF THE ACK SYSTEM

Consider the class Π_{det} of evacuation policies described in Algorithm 1

Algorithm 1 Class Π_{det} of optimal evacuation policies set for the ACK system

```
Input: An instance of an ACK system
Output: Control sequence {c(1), c(2), ...} that evacuates the system.
  $t = 1$
  while $\mathcal{C}_{\text{det}}(t) = \emptyset$ do
    if $\{g_1 \oplus g_2\} \in \mathcal{C}_{\text{det}}(t)$ then   $c(t) = g_1 \oplus g_2$
    else   $c(t) =$ any singleton control
      (Each policy in the class defines a different order of singleton controls.)
    $t = t + 1$
```

In Appendix A we consider the ACK system and a broad class of policies that can perform *arbitrary coding*. We then show that the policies in the restricted class Π_{det} satisfies (1), where:

$$\hat{T}(\lambda_1, \lambda_2) = \frac{\lambda_1}{r_1} + \frac{\lambda_2}{r_2} - \frac{\min\{p_1\lambda_1, p_1\lambda_2\}}{\max\{r_1, r_2\}}, \quad \forall\ \lambda_1, \lambda_2$$

and therefore combining with proposition 3 we have:

THEOREM 4 [THROUGHPUT REGION]: *Consider an ACK system with overhearing probabilities p_1, p_2, transmission rates r_1, r_2 and the ability to perform* arbitrary *coding operations. The system's throughput region is the area defined by $\lambda_1, \lambda_2 \geq 0$ satisfying:*

$$\frac{\lambda_1}{r_1} + \frac{\lambda_2}{r_2} - \frac{\min\{p_1\lambda_1, p_1\lambda_2\}}{\max\{r_1, r_2\}} \leq 1. \qquad (2)$$

Here, we have relaxed the demand that admissible policies only use pair or singleton encoding. By "arbitrary" we mean policies with transmissions that can be of any encoding, linear or non-linear. Therefore, this result provides the achievable throughput for a very general class of systems which assume that the overhearing events are known to the controller via reports.

Moreover, using proposition 3 we conclude that any epoch-based policy $\sigma(\pi), \pi \in \Pi_{\text{det}}$ is throughput optimal.

VII. ANALYSIS OF THE NACK SYSTEM

In this section, we study a set of evacuation policies Π for the NACK system, constrained to the use of XORs (i.e. general coding is not considered). Let $(f, s) = (1, 2)$ if $r_1 \geq r_2$ and $(f, s) = (2, 1)$ otherwise, where f=fast and s=slow. Consider the class of evacuation policies described in Algorithm 2

In Appendix B we prove that these policies satisfy (1) of proposition 2, where:

Algorithm 2 Class Π_{sto} of optimal evacuation policies for the NACK system

```
Input: An instance of a NACK system
Output: Control sequence $\{c(1), c(2), \dots\}$ that evacuates the system.
  if $1 - p_f > \frac{r_s}{rf}$ then Choose only singleton controls that evacuate the system in any order and exit.
  $t = 1$
  while $\mathcal{C}_{\text{sto}}(t) = \emptyset$ do
    if $g_1 \oplus g_2 \in \mathcal{C}_{\text{sto}}(t)$ then   $c(t) = g_1 \oplus g_2$
    else if $u_1 \oplus g_2 \in \mathcal{C}_{sto}(t)$ then   $c(t) = u_1 \oplus g_2$
    else if $u_2 \oplus g_1 \in \mathcal{C}_{sto}(t)$ then   $c(t) = g_1 \oplus u_2$
    else if $u_1 \oplus u_2 \in \mathcal{C}_{sto}(t)$ then   $c(t) = u_1 \oplus u_2$
    else  $c(t) =$ any singleton control
      (Each policy in the class defines a different order of singleton controls. During this step, controls $b_1$ and $b_2$ are used in the way explained in subsection III-C )
    $t = t + 1$
```

$$\hat{T}(\lambda_1, \lambda_2) = \frac{\lambda_1}{r_1} + \frac{\lambda_2}{r_2} - \frac{\min\{\lambda_1 p_1, \lambda_2 p_2\}}{p_f}\left[\frac{1}{r_f} - \frac{1-p_f}{r_s}\right]^+, \quad \forall\ \lambda_1, \lambda_2$$

and therefore, by proposition 3:

THEOREM 5 [CODE-CONSTRAINED REGION]: *Consider a NACK system with overhearing probabilities p_1, p_2, transmission rates r_1, r_2 and the ability to perform XOR coding operations. The system's throughput region is the area defined by $\lambda_1, \lambda_2 \geq 0$ satisfying:*

$$\frac{\lambda_1}{r_1} + \frac{\lambda_2}{r_2} - \frac{\min\{\lambda_1 p_1, \lambda_2 p_2\}}{p_f}\left[\frac{1}{r_f} - \frac{1-p_f}{r_s}\right]^+ \leq 1, \tag{3}$$

where $r_s \triangleq \min\{r_1, r_2\}$, $r_f \triangleq \max\{r_1, r_2\}$ and $[x]^+ \triangleq \max(x, 0)$. Note, that the throughput region for a system with no network coding is simply given by $\lambda_1, \lambda_2 \geq 0$ satisfying $\frac{\lambda_1}{r_1} + \frac{\lambda_2}{r_2} \leq 1$. Therefore we conclude that whenever the term in the brackets $[.]^+$ is negative, network coding is not beneficial, and the maximum throughput is achieved without coding.

Using proposition 3 we conclude that any epoch-based policy $\sigma(\pi), \pi \in \Pi_{\text{sto}}$ achieves the code-constraint region.

A very important consideration is the following: if $r_1 = r_2$, or if $p_f = 1$ the terms cancel out and (3) equals (2), therefore the code-constrained region of the NACK system and the throughput region of the ACK system are equal. That is, *if the maximum rates that each receiver can successfully receive packets are equal, or if the overhearing of the faster rate is perfect, the code-constrained throughput region of the NACK system is equal to the throughput region under arbitrary coding*. This is an important result, since despite the lack of overhearing reports, simple XOR policies in the set Π_{sto} with statistical knowledge and NACKs can achieve the maximum system efficiency. In Fig. 5 we plot the regions for two different settings.

VIII. OVERHEAD COMPARISON

In this section, we study the throughput-overhead tradeoff between the ACK and NACK systems.

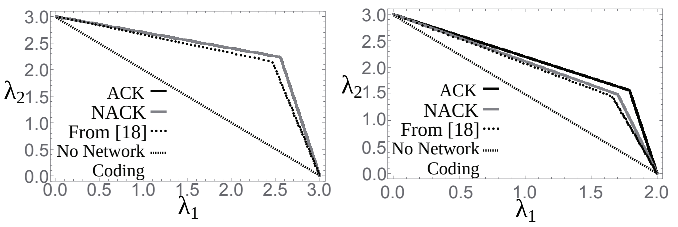

Fig. 5. Throughput regions of no network coding and ACK system and code-constrained regions of the NACK system with or without storing XORs (from [18]). Parameters: $p_1 = 0.7, p_2 = 0.8, r_2 = 3$ and $r_1 = 3$ (left), $r_1 = 2$ (right).

For the ACK system, assuming $\mathcal{N}_1, \mathcal{N}_2$ represent the set of neighboring nodes of sources $1, 2$, the average rate of overhearing reports $W^{det}(\lambda_1, \lambda_2)$ is calculated as

$$W^{det}(\lambda_1, \lambda_2) = \lambda_1 \sum_{i \in \mathcal{N}_1 - R} p_{1,i} + \lambda_2 \sum_{i \in \mathcal{N}_2 - R} p_{2,i},$$

where $1 - p_{1,i}$ is defined to be the erasure probability of the link from source 1 to neighbor i and $1 - p_{2,i}$ likewise. For the NACK system, note that the number of NACK messages is upper bounded by the number of *bad* packets. We say upper bounded because it is possible that *bad* packets are sent as singleton, in which case they are decoded. *good* packets are transmitted without feedback messages. Thus, using policy π^*, the corresponding rate is

$$W^{sto}(\lambda_1, \lambda_2) = q_1^{\text{XOR}}(1 - p_1)\lambda_1 + q_2^{\text{XOR}}(1 - p_2)\lambda_2,$$

where q_i^{XOR} is the fraction of *bad* packets of flow i that are transmitted using XOR controls. If $1 - p_f \leq \frac{\min(r_1, r_2)}{\max(r_1, r_2)}$, no coding is performed and thus $q_1^{\text{XOR}} = q_2^{\text{XOR}} = 0$. Else, we can find an upper bound, when (λ_1, λ_2) lies on the boundary of the throughput region.

$$\overline{q}_1^{\text{XOR}} = \begin{cases} 1 & \text{if } \lambda_1 p_1 \leq \lambda_2 p_2 \\ \frac{\lambda_2 p_2}{\lambda_1 p_1} & \text{otherwise.} \end{cases}$$

and similarly for $\overline{q}_2^{\text{XOR}}$ by exchanging 1 and 2. There are four reasons identified why the NACK system is more efficient in terms of the number of report messages (i.e. overhead).

(i) In the NACK system NACKs can be used, while in the ACK system, ACKs are necessitated. This makes significant difference if the overhearing probabilities p_1, p_2 are close to 1, as is the case where we expect higher throughput benefits.
(ii) If $\lambda_1 p_1 \neq \lambda_2 p_2$, some of the *bad* packets are transmitted natively, thus $q_i^{\text{XOR}} < 1$ for some i.
(iii) When the source nodes have multiple neighbors, reports from neighbors that are not useful to the coding node (relay) are avoided in the NACK system.
(iv) If (λ_1, λ_2) is in the interior of the region without coding, then the overhead for the NACK is very small.

Let $\mathcal{N}_1 = \{R, 2\}$ and $\mathcal{N}_2 = \{R, 1\}$, in which case the best performance of the ACK system is obtained with respect to (iii) above. Also, we calculate the worst case performance for the NACK system as regards (iii) and (iv).

In Fig. 6, 7, we present performance plots for *through-*

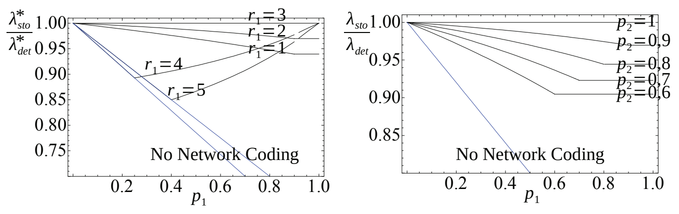

Fig. 6. **Throughput efficiency** varying p_1. Parametric plots vs r_1 (left) and vs p_2 (right). Default parameters: $\lambda_1 = \lambda_2$, $p_2 = 0.9$, $r_1 = 2$, $r_2 = 3$, $\alpha = 1$. Blue lines refer to no network coding comparative performance.

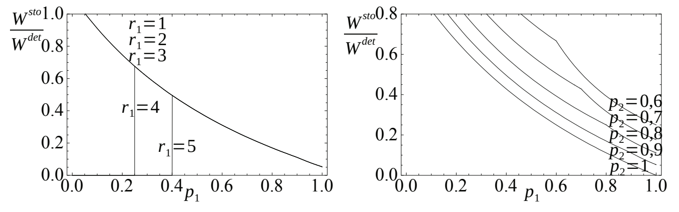

Fig. 7. **Worst-case report overhead** varying p_1. Parametric plots vs r_1 (left) and vs p_2 (right). Default parameters: $\lambda_1 = \lambda_2$, $p_2 = 0.9$, $r_1 = 2$, $r_2 = 3$.

put efficiency $\frac{\lambda^\star_{\text{sto}}}{\lambda^\star_{\text{det}}}$ defined as the ratio of the maximum sum throughput $\lambda^\star_1(\alpha) + \lambda^\star_2(\alpha)$ for the two systems and *worst-case report overhead* $\frac{W^{\text{sto}}}{W^{\text{det}}}$ defined as the ratio of average messaging rate calculated on the boundary of the NACK system region. In Fig. 6. we observe that the throughput drops significantly when p_f is small, i.e. when the fast flow has weak overhearing channel (see the case for $r_1 = 5$ in the left). In all other cases, the NACK system sacrifices only a small fraction of the throughput (stays always above 95%). In Fig. 7, the corresponding gain in overhead is shown. In the left plot, where $p_2 = 0.9$, we see that the NACK system requires at most 50% of the messages used in the ACK system (if $r_1 = 5$), while, independently of rate, this figure becomes as small as 5% if the probabilities are both high (e.g. for $p_1 = p_2 = 0.9$), which is the most practical case. In the right plot we verify that the NACK system is not a good option when both probabilities have middle values.

IX. Delay Properties

In the previous sections we studied simple evacuation mechanisms that provide maximum throughput if they are mapped to the real system with arrivals via the epoch framework. Here, we focus on the delay aspects of the system. In particular we observe that although epoch based policies are throughput optimal, they may suffer from high delay and jitter. Therefore we propose heuristics which mimic the evacuation behavior but also attempt to strike a good balance between throughput (long term efficiency) and delay (short term efficiency).

We measure delay as follows. Consider flow i and name the packets with their index of arrival. Denote by $M_i(t)$ the number of packets that have arrived up to time t. Note, that $M_i(t)$ is also the ID of the most recently arrived packet. Also, denote by $D_i(t)$ the greatest packet ID such that all packets with IDs $1, \dots, D_i(t)$ have departed the system at time t. We define the delay of flow i at slot t as the difference $M_i(t) - D_i(t)$. Observe that the delay reduces to the number of packets in the system when the packets depart the system in order of arrival. This delay metric is most relevant to protocols that emphasize on sequenced packet delivery; a prominent example of such protocols is the widely used TCP protocol. A small delay means that the system does not lack behind in delivering the packets in the order they are transmitted. Ideally we would like to minimize the expected long term delay, or in formal terms:

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} (M_i(t) - D_i(t)) = 0, \quad \forall i. \tag{4}$$

It is known that delay minimization in networks with dynamic control is a difficult problem. Explicit solutions only exist for symmetric networks [19]. In this paper we focus on heuristic solutions whose *good* performance is verified by simulations.

We plan to use our epoch-based policies proposed for throughput optimality in sections VI and VII. There are two important considerations regarding the delay performance of these policies. Below we discuss these considerations which will lead us to delay-driven enhancements.

A. Delay due to Missed Coding Opportunities

Missed coding opportunities occur when *good* packets that can be paired are sent singleton instead. This can happen when *good* packets that are blocked from entering the system because the epoch at the time of arrival is not yet over. Then these packets fail to pair with *good* packets *in the system* from the opposite flow that have no pair within the system and are sent singleton instead. This has no impact on throughput (as it was proven optimal) but it has impact on packet delay. This motivates us to unblock packets and allow them in the system at the moment of their arrival.

Yet the problem may still persist due to the way we solve ties. When there are both *good* packets without a pair in the system and *bad* packets, we choose randomly between them to send singleton (see Algorithm 1, line 6). Suppose that *good* packets are randomly chosen for singleton transmission and their queue is emptied. If at the next slot *good* packets of the opposite flow arrive at the system (because we unblocked arrivals, see the previous paragraph) then the arriving *good* packets would fail to pair because the opposite *good* queue is empty. This problem also impacts delay and motivates us to choose *bad* singleton controls over *good* singleton controls when both exist, i.e we could first send the *bad* packets and in the next slot pair the *good* packets in the system with the arriving *good* packets.

To address this problem, *we allow packets to enter the system at the moment of their arrival* and *we refrain from taking singleton good controls until it is the only remaining choice*. This policy is not epoch-based but we expect throughput optimality to be maintained since the only difference in controls is that pair transmissions between *good* packets will occur where otherwise both packets would have been transmitted singleton. The same arguments can also be used for the NACK system; *unknown* packets that would otherwise be transmitted singleton will now be transmitted paired with other *unknown* or *good* packets.

B. Delay due to Dummy Packets

Recall that when there are not enough packets for a given control, we send dummy packets. This is done even when there exist other packets in the queues to be transmitted, which has impact on delay. A simple scenario to demonstrate this problem is this: consider an ACK system with $r_1 = 2$, $r_2 = 3$ and all queues have exactly one packet. The throughput optimal epoch-based policy proposed in section VI will choose $g_1 \oplus g_2$ and then a dummy packet will be used to patch the transmission. This will evacuate the *good* queues. But during this slot, a *bad* packet from flow 1 could also be transmitted instead of a dummy packet, since the rate of control $g_1 \oplus g_2$ is the same as b_1. This would improve the delay of the *bad* packet, thus dummy packets exacerbate the delay of real packets. Similar arguments can be constructed for the NACK system. Since in a lightly loaded system this phenomenon will occur frequently, we need to enhance our policies.

To address this problem we *give preference to controls that have enough packets to transmit prior to controls that require dummy packets.* When all possible controls involve the use of dummy packets, we *choose one of the two rates and attempt to transmit as many packets as possible by mixing several controls together*. The choice of rate is such that we maximize the expected amount of packets we can evacuate. One example would be to transmit both the XOR combination of the *good* packets and the *bad* packet of the slow flow in the example of the previous paragraph. This way we end up transmitting the maximum number of packets at each slot. Similar policies that maximize the number of departed packets in every slot have been proposed before for delay minimization in symmetric systems [20].

C. Enhanced Policies and Simulation Results

Algorithm 3 Class Σ_{det} for the system with arrivals

```
Input: An ACK system state at slot $t$
Output: Control decision $c(t)$
  if $g_1 \oplus g_2 \in \mathcal{C}_{\text{det}}(t)$ then $c(t) = g_1 \oplus g_2$
  else if $b_1 \in \mathcal{C}_{det}(t)$ or $b_2 \in \mathcal{C}_{det}(t)$ then $c(t) = b_1$ or $c(t) = b_2$
      (whichever of the two controls is available, but if both are available, choose the control that evacuates the packets with the smallest ID)
  else if $g_1 \in \mathcal{C}_{det}(t)$ or $g_2 \in \mathcal{C}_{det}(t)$ then $c(t) = g_1$ or $g_2$
      (whichever of the two controls is available; due to order of chosen controls, at most one of those controls is available)
  else Set control $c(t)$ to transmit as many packets as possible with a chosen rate.
```

In Algorithms 3 and 4 we propose the delay-aware policies for the system with arrivals. These policies are based on the epoch-based policies proposed in previous section where the delay enhancements explained in the previous paragraphs are also added. For the sake of brevity, when we write control $c(t) \in \mathcal{C}_{det}(t)$ (or $\mathcal{C}_{sto}(t)$), we mean that the packets that are available for transmission under this control are more or equal to the rate used for this control.

In Figure 8, we show the throughput region of our implementation with the above heuristic for the system with arrivals as it occurs in our simulations. Notice that there is no difference between the throughput region of epoch-based policies and their delay-aware counterparts, which is a indication that

Algorithm 4 Class Σ_{sto} for the system with arrivals

```
Input: A NACK system state at slot $t$
Output: Control decision $c(t)$
  $slow = \arg\min_i(r_i, i \in \{1,2\})$
  if $g_1 \oplus g_2 \in \mathcal{C}_{\text{sto}}(t)$ then $c(t) = g_1 \oplus g_2$
  else if $b_{slow} \in \mathcal{C}_{sto}(t))$ then $c(t) = b_{slow}$
      (During this step, control $b_{slow}$ is used in the way explained in subsection III-C)
  else if $u_1 \oplus g_2 \in \mathcal{C}_{sto}(t)$ or $u_2 \oplus g_1 \in \mathcal{C}_{sto}(t)$ then $u_1 \oplus u_2 \in \mathcal{C}_{sto}(t)$
      (whichever of the two controls is available; due to order of chosen controls, at most one of those controls is available)
  else if $u_1 \oplus u_2 \in \mathcal{C}_{sto}(t)$ then $c(t) = u_1 \oplus u_2$
  else if $u_1 \in \mathcal{C}_{sto}(t)$ or $u_2 \in \mathcal{C}_{sto}(t)$ then $c(t) = u_1$ or $u_2$
      (whichever of the two controls is available; due to order of chosen controls, at most one of those controls is available)
  else if $g_1 \in \mathcal{C}_{sto}(t)$ or $g_2 \in \mathcal{C}_{sto}(t)$ then $c(t) = g_1$ or $g_2$
      (whichever of the two controls is available; due to order of chosen controls, at most one of those controls is available)
  else Set control $c(t)$ to transmit as many packets as possible with a chosen rate.
```

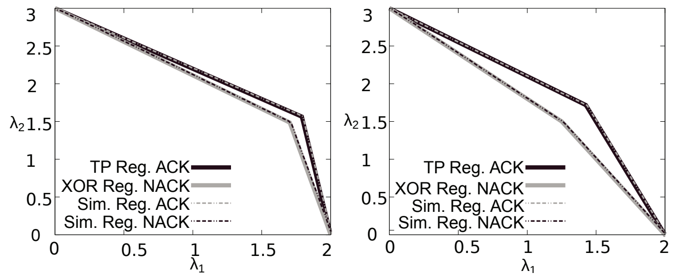

Fig. 8. Throughput region achieved by our algorithm implementation in our simulations. In both cases, $(r_1, r_2) = (2, 3)$. (left) Case for $(p_1, p_2) = (.7, .8)$. (right) Case for $(p_1, p_2) = (.6, .5)$.

our delay modifications do not hurt throughput optimality. We conducted our simulations for 10^6 time slots per experiment. To find the throughput region, we used iterative convergence, where for a number of 30 points, equally spaced in the x-axis (λ_1) we computed the corresponding y coordinate (λ_2) such that the system is marginally stable. Stability is determined if delay surpasses a certain threshold, for our case more than 10^4 packets for some t and i. The tolerance used is 10^{-3}.

In Figure 9 we present the delay performance of our policies as compared to *Random Linear Network Coding* -a state of the art algorithm for network coding- a *backpressure-type* policy proposed in [21] and COPE.

1) Our simulation of RLNC

This policy considers the packets at the input in different generations of size g. In each generation, g packets from each receiver are coded together forming N_g equations with randomly drawn coefficients. In some cases, some receivers do not participate if they do not have any packets. We assume an idealized version of the policy where the coefficients are pseudo-random and they result in linearly independent coded packets. These equations are transmitted until all receivers have decoded all N_g packets. Side information packets are also linearly independent equations that can be used to accelerate the decoding. However, the required transmissions are calculated based on the receiver with the smallest number of side information packets. When the first generation is decoded, the transmissions stop and we proceed to the second generation. When all the packets of the generation are decoded we move to the next generation. We expect this policy to be inefficient compared to the optimal because it requires all the receivers to decode all packets of a generation for those packets to depart the system which exacerbates delay. Moreover, the slowest possible rate is chosen for transmissions so that all receivers can decode. This puts RLNC in a significant disadvantage if the service rates are different.

Another important consideration with RLNC is that the controller is completely oblivious to overhearing events. Consider the scenario that a whole generation of packets are *bad* in both flows. Due to lack of knowledge on overhearing events, it will try to code together *bad* packets, resulting in their delayed departure since their linear combinations will be transmitted at the slowest rate. Our policies have feedback on overhearing events and therefore can use this knowledge to send *bad* packets of the faster flow at the faster rate.

2) Work at [21]

Here, we consider a *backpressure*-based mechanism for both our feedback systems. The backpressure is applied to a network of *virtual-queues* as explained in [21]. A notable disadvantage of this approach is that the receivers do not store XORed packets at their buffers; a received coded packet is either immediately decoded or discarded. The lack of storing XORed packets in receivers results in the loss of some coding opportunities in the NACK system.

3) COPE [1]

Here we consider the application of the COPE algorithm in our model. COPE proposes a statistical mechanism whereby packets are XORed in FIFO order of their arrival when the probability that they will be decoded is greater than a threshold G. The default is $G = 0.8$ which we use in the simulations. The COPE algorithm can work in two modes of operation. First, it delays packets until there are enough packets to code with. Second, it never delays packets i.e it transmits when there is a transmission opportunity. Since the first choice is suboptimal for packet delay, we only consider the second mode of operation.

On the ACK system, the COPE algorithm, denoted COPE-ACK, has full knowledge of when a coded packet can be decoded or not. Therefore the decision to code two packets is straightforward, since the probability of correct decoding is either 1 or 0. In this case COPE-ACK is similar to our ACK policy (Algorithm 3) given that COPE-ACK never delays packets. Consequently, we do not simulate COPE-ACK.

On the NACK system however, the COPE algorithm, denoted COPE-NACK, p_1, p_2 and NACK replies inform the relay on the probabilities of correct decoding. After using the statistics, COPE-NACK then XOR codes packets depending on G. In this way, our work is an improvement of COPE-NACK in providing a different threshold for the XOR coding algorithm to start coding packets (i.e $1 - p_f \leq \frac{\min(r_1,r_2)}{\max(r_1,r_2)}$) and prove that this threshold is optimal. Additionally, with our buffer mechanism where the receivers store non-decoded packets, we quickly recover from decoding errors within a slot of their occurrence.

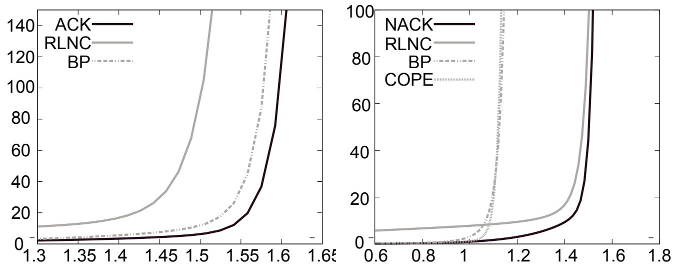

Fig. 9. Average delay performance for both the ACK and NACK systems as compared to Random Linear Network Coding, COPE and the back-pressure policy described in [21], with load factor $\boldsymbol{\lambda} = (\lambda, \lambda)$. X-axis denotes λ, Y-axis denotes mean delay. (left) ACK case. (right) NACK case. For all cases, the overhearing probabilities are $(p_1, p_2) = (.7, .8)$ and rate $(r_1, r_2) = (2, 3)$

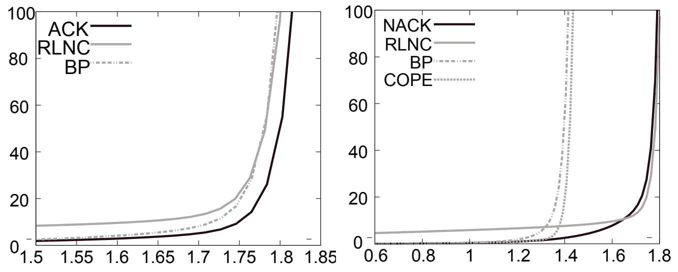

Fig. 10. Average delay performance for both the ACK and NACK systems as compared to Random Linear Network Coding, COPE and the back-pressure policy described in [21], with load factor $\boldsymbol{\lambda} = (\lambda, .5\lambda)$. X-axis denotes λ, Y-axis denotes mean delay. (left) ACK case. (right) NACK case. For all cases, the overhearing probabilities are $(p_1, p_2) = (.7, .8)$ and rate $(r_1, r_2) = (2, 3)$

In Fig 9 and 10 we show that our proposed delay-aware policies outperform RLNC, COPE-NACK and backpressure-based policies in all studied examples.

In Figures 11 we provide some experiments with arrival rates within the stability region but close to the boundary, along with system backlog. We chose our simulations for a small time interval (1000 slots) to show how $M_i(t) - D_i(t)$ evolves since for long simulations they become indistinguishable.

X. Related work

Most wireless network coding schemes use the ACK system [2]–[6], [22]. These schemes suffer from the problem of keeping the coding nodes informed with the packet indices that have been overheard by its nexthops. NCRAWL [23] addresses that by using reception reports only when a node can't decode a packet. This reduces the amount of reports, but increases the likelihood that the coding node will make a mistake in predicting which packets are overheard by the nexthops, something that inevitably lowers throughput. We use this model of reporting in the 2-user NACK model. A different approach is introduced by I^{2}NC [24] which combines interflow with intraflow coding to reduce the complexity of acknowledgment messages at the expense of immediate decodability.

Stability in networks with interflow network coding without overhearing is studied in [3] and [11], [25]–[27]. Also, in

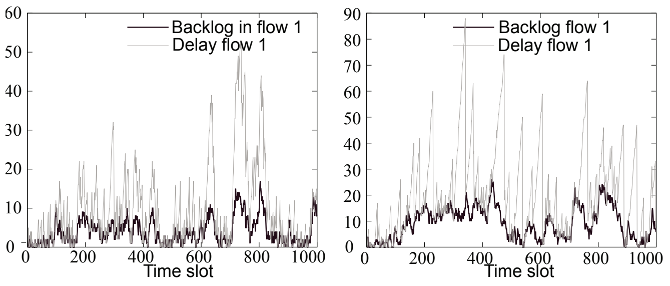

Fig. 11. Overview of delay for a system with service vector $(r_1, r_2) = (2, 3)$ and overhearing probabilities $(p_1, p_2) = (.7, .8)$, with load vector close to the boundary of the throughput region (97% of capacity). We also provide the backlog of the system at each slot. The difference of delay minus the backlog is indicative of the instantaneous number of out-of-order packets. (left) ACK system with arrival vector $(\lambda_1, \lambda_2) = (1.72, 1.63)$ (right) NACK system with arrival vector $(\lambda_1, \lambda_2) = (1.72, 1.46)$

[18], [28] the studies are extended to capture overhearing with reports, which corresponds to the 2-user ACK system. Note that in these works, the code-constrained stability region is provided, i.e. the stability region under the assumption that XOR coding is used. The 1-hop model is also studied in [10] where the information theoretic capacity is given in the case of overhearing events provided as side information- a model equivalent to the 2-user ACK system we study here. With the exception of [18], all these works do not consider the 2-user NACK system. In [18], the 2-user NACK system with feedback is studied under the assumption that receivers are not allowed to store coded packets and the code-constrained throughput region is provided in parametric form. The obtained throughput region is strictly smaller than that of the 2-user ACK system. In this work, we extend [18] by allowing the storage of coded packets. We show, that if $r_1 = r_2$ or $p_f = 1$, then the 2-user NACK system can achieve the same throughput as the 2-user ACK one by the use of a simple XOR-based scheduling policy and feedback reports. Thus, the number of reports can be reduced significantly in this case without throughput losses.

Studies of the broadcast channel with erasures relate to our work, see [29]. In these studies, the problem is different since the side information for decoding is obtained from past erased transmissions; however, the techniques used are similar. In [30], the authors show that the capacity can be achieved by XOR coding for the case of 2-4 receivers. In [15] the authors study the 2 user ACK system on a channel with erasures and provide an optimal policy based on Deficit Max-Weight Algorithm [31] applied on a virtual network queue. A different but related research topic is that of index coding; subsets of information bits are known to subsets of the receivers and we seek the transmission policy that minimizes the time to complete reception by all receivers, [17], [32]. Our work differs from index coding in the fact that the source has partial knowledge of what information each receiver has. Also, for the 2-user ACK system, we extend the index coding problem to variable transmission rates r_1, r_2. The relation between index coding and wireless network coding with overhearing is further explored in [33].

XI. Conclusion

In the problem of reporting overhearing events in wireless network coding, we study the ACK and NACK system. We derive analytical expressions for the throughput region of the first and the code-constrained region of the second and we show that the two are equal when $r_1 = r_2$ or $p_f = 1$. When $r_1 \neq r_2$, we analyze the throughput-overhead tradeoff and conclude that the NACK system is a very efficient approach when the overhearing probabilities are sufficiently high. Alongside with the theoretical results, we propose simple and efficient evacuation policies which can be used in practice to achieve optimal throughput for the case of two receivers or performance for more than two receivers. We extend these policies by proposing delay enhancements and evaluate those by simulations.

Acknowledgments

The work of G. Paschos was supported by the WiNC project of the Action: Supporting Postdoctoral Researchers, funded by national and Community funds (European Social Fund).

The work of C. Fragiadakis is supported by scholarship from the National Scholarship Foundation of Greece from funds by the "Education and Lifelong Learning" Program of the European Social Fund .

This research has been co-financed by the European Union (European Social Fund ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: Thales. Investing in knowledge society through the European Social Fund.

References

[1] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft, "XORs in The Air: Practical Wireless Network Coding," in *ACM SIGCOMM*, 2006.

[2] S. Katti, H. Rahul, W. Hu, D. Katabi, M.Medard, and J.Crowcroft, "Xors in the air: Practical wireless network coding," *IEEE/ACM Transactions on Networking*, vol. 16, pp. 497–510, Jun. 2008.

[3] P. Chaporkar and A. Proutiere, "Adaptive Network Coding and Scheduling for Maximizing Throughput in Wireless Networks," in *ACM MOBICOM*, 2007.

[4] S. Rayanchu, S. Sen, J. Wu, S. Banerjee, and S. Sengupta, "Loss-Aware Network Coding for Unicast Wireless Sessions: Design, Implementation, and Performance Evaluation," in *ACM SIGMETRICS*, 2008.

[5] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading Structure for Randomness in Wireless Opportunistic Routing," in *ACM SIGCOMM*, 2007.

[6] B. Scheuermann, W. Hu, and J. Crowcroft, "Near-Optimal Co-ordinated Coding in Wireless Multihop Networks ," in *ACM CONEXT*, 2007.

[7] I. Broustis, G. S. Paschos, D. Syrivelis, L. Georgiadis, and L. Tassiulas, "NCRAWL: Network Coding for Rate Adaptive Wireless Links," arXiv:1104.0645.

[8] G. S. Paschos, C. Fragiadakis, L. Georgiadis, and L. Tassiulas, "Wireless Network Coding with Partial Overhearing Information," in *Proceedings of IEEE INFOCOM*, Apr. 2013.

[9] A. O. F. Atya, I. Broustis, S. Singh, D. Syrivelis, S. V. Krishnamurthy, and T. L. Porta, "Wireless Network Coding: Deciding When to Flip the Switch," in *IEEE INFOCOM*, 2013.

[10] C.-C. Wang, "On the capacity of wireless 1-hop intersession network codinga broadcast packet erasure channel approach," *Information Theory, IEEE Transactions on*, vol. 58, no. 2, pp. 957–988, 2012.

[11] N. M. Jones, B. Shrader, and E. Modiano, "Optimal routing and scheduling for a simple network coding scheme," in *Proceedings of IEEE INFOCOM*, Apr. 2012, pp. 352–360.

[12] L. Jilin, J. Lui, and M. Dah, " How many packets can we encode? an analysis of practical wireless network coding," in *IEEE INFOCOM*, Apr. 2008, pp. 371–375.

[13] P. Mannersalo, G. S. Paschos, and L. Gkatzikis, "Performance of wireless network coding: motivating small encoding numbers," *arXiv:1010.0630v1*, 2010.

[14] J. Qureshi, C. H. Foh, and J. Cai, "Optimal solution for the index coding problem using network coding over gf(2)," in *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2012 9th Annual IEEE Communications Society Conference on*, June 2012, pp. 209–217.

[15] W.-C. Kuo and C.-C. Wang, "Robust and optimal opportunistic scheduling for downlink 2-flow inter-session network coding with varying channel quality," in *33rd Conference on Computer Communications (INFOCOM), Toronto, Canada.* IEEE, 2014.

[16] L. Georgiadis, G. S. Paschos, L. Tassiulas, and L. Libman, "Stability and Capacity through Evacuation Times," in *Information Theory Workshop, (ITW)*, Sep. 2012.

[17] M. J. Neely, A. S. Tehrani, and Z. Zhang, "Dynamic Index Coding for Wireless Broadcast Networks," in *INFOCOM*, Apr. 2012.

[18] G. S. Paschos, L. Georgiadis, and L. Tassiulas, "Scheduling with Pairwise XORing of packets under Statistical Overhearing Information and Feedback," *Queueing Syst. Theory Appl.*, vol. 72, no. 3-4, pp. 361–395, Dec. 2012.

[19] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan and Claypool Publishers, 2010.

[20] L. Tassiulas and A. Ephremides, "Dynamic scheduling for minimum delay in tandem and parallel constrained queueing models," *Ann Oper Res*, vol. 48, no. 4, pp. 333–355, Aug. 1994. [Online]. Available: http://dx.doi.org/10.1007/BF02024520

[21] G. S. Paschos, L. Georgiadis, and L. Tassiulas, "Optimal scheduling of pairwise XORs under statistical overhearing and feedback," in *RAWNET workshop: Workshop on Resource Allocation and Cooperation in Wireless Networks, WiOPT*, Apr. 2011.

[22] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "Xors in the air: practical wireless network coding," *IEEE/ACM Transactions on Networking (TON)*, vol. 16, no. 3, pp. 497–510, 2008.

[23] I. Broustis, G. S. Paschos, D. Syrivelis, L. Georgiadis, and L. Tassiulas, "NCRAWL: Network Coding for Rate Adaptive Wireless Links," *to appear in EURASIP Journal on Wireless Communications and Networking*, 2013.

[24] H. Seferoglu, A. Markopoulou, and K. K. Ramakrishnan, "I2nc: Intra- and inter-session network coding for unicast flows in wireless networks," in *Proceedings of IEEE INFOCOM*, Apr. 2011, pp. 1035–1043.

[25] T. C. Ho, Y.-H. Chang, and K. J. Han, "On Constructive Network Coding for Multiple Unicasts," in *44th Annual Allerton Conference on Communication, Control, and Computing*, 2006.

[26] A. Eryilmaz and D. S. Lun, "Control for Inter-session Network Coding," in *Proc. Workshop on Network Coding, Theory & Applications*, 2007.

[27] Y. E. Sagduyu, D. Guo, and R. Berry, "Throughput and stability of digital and analog network coding for wireless networks with single and multiple relays," in *Proceedings of the 4th Annual International Conference on Wireless Internet*, ser. WICON '08, 2008, pp. 68:1–68:9.

[28] A. Khreishah, C.-C. Wang, and N. B. Shroff, "Rate control with pairwise intersession network coding," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 816–829, Jun. 2010.

[29] L. Georgiadis and L. Tassiulas, "Broadcast Erasure Channel with Feedback – Capacity and Algorithms," in *Workshop on Network Coding, Theory and Applications (NetCod)*, Jun. 2009.

[30] S. Athanasiadou, M. Gatzianas, L. Georgiadis, and L. Tassiulas, "XOR-based coding algorithms for the 3-user broadcast erasure channel with feedback," in *RAWNET workshop: Workshop on Resource Allocation and Cooperation in Wireless Networks, WiOPT*, Apr. 2012.

[31] H. C. Zhao, C. H. Xia, Z. Liu, and D. Towsley, "A unified modeling framework for distributed resource allocation of general fork and join processing networks," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 38, no. 1. ACM, 2010, pp. 299–310.

[32] M. Chaudhry and A. Sprintson, "Efficient algorithms for index coding," in *INFOCOM*, Apr. 2008.

[33] C. Fragiadakis, G. S. Paschos, and L. Tassiulas, "On the wireless network coding with overhearing and index coding," in *submitted to IEEE CAMAD*, 2013.

[34] P. Billingsley, *Probability and Measure*. John Wiley & Sons, 1995.

Appendix A
ACK Analysis

This appendix covers the proof of theorem 4 and is structured as follows. To prove the theorem we need to derive a lower bound on evacuation time achievable by any policy. Since this is the best possible performance, it naturally corresponds to an outer bound for the throughput region. To obtain the best performance we assume arbitrary coding available. In the subsequent sections we evaluate the class of evacuation policies Π_{det} and then combine the two bounds to derive the result. For this appendix, we let $r_s = min\{r_1, r_2\}, N_{min} = min\{N_1, N_2\}, N_{max} = max\{N_1, N_2\}, T^{bdet}(k_1, k_2) \triangleq \frac{k_1}{r_1} + \frac{k_2}{r_2} - \frac{\mathbb{E}[\min\{M_1, M_2\}]}{\max\{r_1, r_2\}}$

A. Lower Bound under Arbitrary Coding

Theorem 6 [Lower bound with arbitrary coding]: *The ACK system satisfies under any $\pi \in \Pi$:*

$$\overline{T}^{\pi}(k_1, k_2) \geq T^{bdet}(k_1, k_2),$$

Proof: The proof can be found in [8], pp. 4-5 section IV and we omit it here due to lack of space. ■

B. Upper Bound of Π_{det}

Let (k_1, k_2, N_1, N_2) be a system instance for the ACK system. Let $\overline{r} \triangleq r_1$ if $N_1 \geq N_2$ and $\overline{r} \triangleq r_2$, otherwise. Since the policies in Π_{det} do not depend on the values of the bits in the packets, their evacuation times given a realization for N_1, N_2 are deterministic. By counting slots for the two steps of the policy, we have

$$T^{\pi}(k_1, k_2, N_1, N_2) \leq \left\lceil \frac{N_{min}}{r_s} \right\rceil + \left\lceil \frac{N_{max} - N_{min}}{\overline{r}} \right\rceil$$
$$+ \left\lceil \frac{k_1 - N_1}{r_1} \right\rceil + \left\lceil \frac{k_2 - N_2}{r_2} \right\rceil \leq \frac{k_1}{r_1} + \frac{k_2}{r_2} - \frac{N_{min}}{r_s} + 4 \quad (5)$$

for all $\pi \in \Pi_{det}$.

C. Optimality of Π_{det}

Proof of theorem 4: Consider $k_i = \lceil \lambda_i t \rceil$, $i = 1, 2$ packets to be evacuated and note that the number of good packets per flow are binomial random variables, denoted by $M_1(k_1, p_1), M_2(k_2, p_2)$ correspondingly. Since the status of arriving packets (*good*, *bad*) is an i.i.d. process, we have,

$$\lim_{t\to\infty} \mathbb{E}\left[\frac{M_i(\lceil \lambda_i t \rceil, p_i)}{t}\right] = p_i \lambda_i, \quad (6)$$

and also, by the strong law of large numbers,

$$\lim_{t\to\infty} M_i(\lceil \lambda_i t \rceil, p_i, \omega)/t = p_i \lambda_i \quad \text{w.p.1.} \quad (7)$$

Recall that $\overline{T}^{\star}(k_1, k_2)$ is the infimum of the average evacuation time over all policies, hence a lower bound of $\overline{T}^{\pi}$. By Theorem 6:

$$\mathbb{E}\big[T^{bdet}(k_1, k_2)\big] \leq \overline{T}^{*}(k_1, k_2) \leq \mathbb{E}[T^{\pi}(k_1, k_2, N_1, N_2)] \quad (8)$$

Where $N_i = M_i(k_i, p_i, \omega)$. We calculate the limit of the upper bound of $\overline{T}^{\pi}(k_1, k_2)$ using the RHS of (5)

$$\lim_{t\to\infty} \mathbb{E}\left[\frac{\frac{k_1 t}{r_1} + \frac{k_1 t}{r_1} - \frac{\min\{M_1(k_1 t, p_1, \omega), M_2(k_2 t, p_2, \omega)\}}{\max\{r_1, r_2\}} + 4}{t}\right]$$
$$= \frac{k_1}{r_1} + \frac{k_2}{r_2} - \lim_{t\to\infty} \mathbb{E}\left[\frac{\min\{M_1(k_1 t, p_1, \omega), M_2(k_2 t, p_2, \omega)\}}{t \max\{r_1, r_2\}}\right]$$
$$= \frac{k_1}{r_1} + \frac{k_2}{r_2} - \frac{\min\{p_1 k_1, p_2 k_2\}}{\max\{r_1, r_2\}}, \quad \text{w.p.1,}$$

where in the last step, we exchange the order of limit expectation and $\min$ function due to uniform integrability which follows from convergence in expectation (6) and almost everywhere convergence (7) of the involved sequences, see [34] Th. 16.14. We can repeat the limit derivation for the case of T^{bdet} and derive the same limit, hence from (8) we conclude

$$\hat{T}(\lambda_1, \lambda_2) = \frac{\lambda_1}{r_1} + \frac{\lambda_2}{r_2} - \frac{\min\{p_1\lambda_1, p_2\lambda_2\}}{\max\{r_1, r_2\}}$$

■

Appendix B
NACK Analysis

This appendix provides the proof of theorem 5 by means of two intermediate results, namely theorems 7 and 8. First we focus on deriving a lower bound on how fast the NACK system is evacuated in the asymptotic sense, assuming the use of XOR-based policies. Then we also show that the asymptotic rate is matched by our proposed policy.

For this appendix, we let $r_f = max\{r_1, r_2\}, r_s = min\{r_1, r_2\}, (.)^+ \triangleq \max\{., 0\}, B^{req} \triangleq \frac{\lambda_1}{r_1} + \frac{\lambda_2}{r_2} - \frac{\min(\lambda_1 p_1, \lambda_2 p_2)}{p_f}\left[\frac{1}{r_f} - \frac{1-p_f}{r_s}\right]^+$.

A. Lower Bound on the Growth Rate for the NACK System

Theorem 7 [Lower bound on the growth rate]: *For the NACK system, constrained to the use of XOR coding, it holds*

$$\liminf_{t\to\infty} \frac{\overline{T}^{\pi}\left(\lceil t\lambda_1\rceil, \lceil t\lambda_2\rceil\right)}{t} \geq B^{req}, \textit{ for all } \pi \in \Pi. \quad (9)$$

Proof: We assume that the packets are served from the queues in an FCFS manner, since all packets in a given queue are statistically equivalent and thus reordering them does not change the expected outcome.

Define $\mathcal{C}^x(t) \subset \mathcal{C}_{sto}(t)$ to be the set of controls that involve XOR of two packets such that the corresponding packets are available for transmission.

We partition the set of policies Π to three sets, the subset of policies using only single controls Π_{sin}, the subset of policies using always XOR controls if $\mathcal{C}^x(t) \neq \emptyset$, called Π_{xor} and the rest Π_{mix}. We immediately get

$$\overline{T}^{\pi}(k_1, k_2) \geq \frac{k_1}{r_1} + \frac{k_2}{r_2}, \text{ for all } \pi \in \Pi_{sin}. \quad (10)$$

Next we will find a bound for policies in Π_{xor} and ultimately we will show that the policies in Π_{mix} are outperformed (in asymptotic sense) by those in $\Pi_{sin} \cup \Pi_{xor}$.

Remember that $M_i(k_i, p_i), i \in \{1, 2\}$ is the random variable that denotes the number of good packets in flow i. Let $M_{\min} \triangleq \min\{M_1, M_2\}$ and recall $(f, s) = (1, 2)$ if $r_1 \geq r_2$ and $(f, s) = (2, 1)$ otherwise. Observe that the following hold under any policy in Π_{xor}:

1) While XOR controls are still available (i.e. $\mathcal{C}^x(t) \neq \emptyset$), a *good* packet departs only if coded with another *good* packet independently of the XOR control used.
2) At the end of the slot that the packets from one flow are all evacuated for the first time, it holds: exactly M_{min} *good* packets of both flows have departed.

We make the following helpful conventions:

1) In case of a $\{u_1 \oplus u_2\}$ control involving two *bad* packets followed by a single control of one of the two *bad* packets (the combination evacuates both packets), we assign one evacuated packet to each control.
2) Then, all XOR controls evacuate exactly one packet with the exception of the control $\{g_1 \oplus g_2\}$, which evacuates two packets.

Let $J(i) - 1, i = 0, 1$ be the number of packets in front of the $M_{min} + i$-th *good* packet in the unknown queue of the fast flow at time 0. Using the law of iterative expectations we get $\mathbb{E}[J(0)] = \mathbb{E}[M_{\min}]/p_f$ and $\mathbb{E}[J(1)] = (\mathbb{E}[M_{\min}] + 1)/p_f$.

All packets of the slow flow plus the *bad* packets of fast flow of at least up to $J(0)$ are evacuated in slots of r_s packets requiring one transmission each. Then the remaining $k_f - J(0)$ packets of the fast flow are evacuated in slots of r_f packets. Thus, for any $\pi \in \Pi_{XOR}$

$$\begin{aligned}\overline{T}^{\pi} &\geq \mathbb{E}\left[\left\lceil \frac{k_s + J(0) - M_{min}}{r_s}\right\rceil\right] + \mathbb{E}\left[\left\lceil \frac{k_f - J(0)}{r_f}\right\rceil\right] \\ &\geq \mathbb{E}\left[\frac{k_s + J(0) - M_{min}}{r_s}\right] + \mathbb{E}\left[\frac{k_f - J(0)}{r_f}\right] \\ &= \frac{k_s}{r_s} + \frac{(1-p_f)\mathbb{E}[M_{min}]}{p_f r_s} + \frac{k_f}{r_f} - \frac{\mathbb{E}[M_{min}]}{p_f r_f} \\ &= \frac{k_1}{r_1} + \frac{k_2}{r_2} - \frac{\mathbb{E}[M_{min}]}{p_f}\left[\frac{1}{r_f} - \frac{1-p_f}{r_s}\right]\end{aligned}$$

which combined with (10) yields

$$\overline{T}^{\pi}(k_1, k_2) \geq \frac{k_1}{r_1} + \frac{k_2}{r_2} - \frac{\mathbb{E}[M_{min}]}{p_f}\left[\frac{1}{r_f} - \frac{1-p_f}{r_s}\right]^+,$$

for all $\pi \in \Pi_{sin} \cup \Pi_{xor}$. Using $\lim_{t\to\infty} \frac{\mathbb{E}[M_{min}]}{t} = \min\{k_1 p_1, k_2 p_2\}$ found above, we conclude that

$$\liminf_{t\to\infty} \frac{\overline{T}^{\pi}\left(\lceil t\lambda_1\rceil, \lceil t\lambda_2\rceil\right)}{t} \geq B^{req}, \quad \pi \in \Pi_{sin} \cup \Pi_{xor},$$

Next, we consider set Π_{mix}.

Pick a policy $\pi \in \Pi_{mix}$. Let $L_s(k_s, k_f, \omega), L_f(k_s, k_f, \omega)$ be random variables denoting the number of packets that were evacuated with controls $\{g_s\}, \{u_s\}$ and $\{g_f\}, \{u_f\}$ respectively. We have $l_i \triangleq \mathbb{E}[L_i(k_s, k_f, \omega)]$ and $0 \leq l_i \leq k_i$, for $i \in \{s, f\}$.

Let $G_s(k_s, k_f, \omega), G_f(k_s, k_f, \omega)$ be the number of *good* packets that were evacuated with the above controls in the fast and slow flow respectively. Furthermore, let $H_i(k_s, k_f, \omega)$ be the number of *good* packets evacuated by controls $\{g_i\}$. By the law of large numbers we have w.p.1:

$$\lim_{t\to\infty} \frac{G_i(k_s t, k_f t, \omega)}{t} = \mathbb{E}[H_i] + p_i(\mathbb{E}[L_i] - \mathbb{E}[H_i]) \geq p_i l_i. \quad (11)$$

All k_s packets and the $k_f - L_f$ packets of the fast flow are evacuated with rate r_s. Therefore, the expected number of timeslots needed to evacuate these packets is:

$$\begin{aligned}\overline{T}_1 &\geq \mathbb{E}\left[\frac{k_s}{r_s} + \frac{k_f - L_f}{r_s} - \frac{\min(M_s - G_s, M_f - G_f)}{r_s}\right] \\ &= \frac{k_s}{r_s} + \frac{k_f - l_f}{r_s} - \frac{\mathbb{E}[\min(M_s - G_s, M_f - G_f)]}{r_s}, \quad (12)\end{aligned}$$

where we have subtracted the time corresponding to XORs between *good* packets. Also, the inequality is due to the assumption that no dummy packets were used. The rest L_f packets are evacuated with rate r_f thus:

$$\overline{T}_2 \geq \mathbb{E}\left[\left\lceil \frac{L_f}{r_f} \right\rceil\right] \geq \frac{l_f}{r_f}. \tag{13}$$

Therefore, using (12) and (13), we have:

$$\overline{T}^{\pi}(k_1,k_2) = \overline{T}_1 + \overline{T}_2 \geq \frac{k_s}{r_s} + \frac{k_f}{r_s} - l_f\left(\frac{1}{r_s} - \frac{1}{r_f}\right) - \frac{\mathbb{E}[\min(M_s - G_s, M_f - G_f)]}{r_s}$$

Define $B^{mix} \triangleq \liminf_{t\to\infty} \frac{\overline{T}^{\pi}(\lceil tk_1\rceil, \lceil tk_2\rceil)}{t}$, $\pi \in \Pi_{mix}$. Taking the limit in RHS above, using uniform integrability of the considered random sequences, we get w.p.1:

$$B^{mix} \overset{(11)}{\geq} \frac{k_s}{r_s} + \frac{k_f}{r_s} - l_f\left(\frac{1}{r_s} - \frac{1}{r_f}\right) - \frac{\min\left(p_s(k_s - l_s), p_f(k_f - l_f)\right)}{r_s}. \tag{14}$$

Consider the conditions $p_sk_s \geq p_fk_f$, $p_s(k_s - l_s) \geq p_f(k_f - l_f)$ and $\overline{p_f} > \frac{r_s}{r_f}$ Using $0 \leq l_i \leq k_i$, $i \in \{s,f\}$, and subtracting B^{req} from both sides of (14) we verify that $B^{mix} \geq B^{req}$ under all possible combinations of the above conditions. ■

B. Asymptotic Optimality of Π_{sto}

THEOREM 8 [ASYMPTOTIC OPTIMALITY OF POLICY π^*]: *For the NACK system operating under policy π^* we have*

$$\limsup_{t\to\infty} \frac{\overline{T}^{\pi^*}(\lceil t\lambda_1\rceil, \lceil t\lambda_2\rceil)}{t} \leq B^{req}. \tag{15}$$

Proof: We follow the steps of the proof of Theorem 7 closely. First, note that if $1 - p_f > \frac{r_s}{r_f}$ is true, then π^* chooses only single controls and we quickly get

$$\hat{T}^{\pi^*}(\lambda_1,\lambda_2) = \frac{\lambda_1}{r_1} + \frac{\lambda_2}{r_2}.$$

If the condition is false, then $\pi^* \in \Pi_{XOR}$. The difference from the proof of Theorem 7 is how packets between $J(0)$ and $J(1)$ are treated.

$$\begin{aligned}\overline{T}^{\pi^*}(k_1,k_2) &\leq \mathbb{E}\left[\left\lceil \frac{k_s + J(1) - M_{min}}{r_s} \right\rceil\right] + \mathbb{E}\left[\left\lceil \frac{k_f - J(0)}{r_f} \right\rceil\right] \\ &\leq \mathbb{E}\left[\frac{k_s + J(1) - M_{min}}{r_s}\right] + \mathbb{E}\left[\frac{k_f - J(0)}{r_f}\right] + 2 \\ &= \frac{k_s}{r_s} + \frac{(1-p_f)\mathbb{E}[M_{min}]}{p_f r_s} + \frac{1}{p_f r_s} + \frac{k_f}{r_f} - \frac{\mathbb{E}[M_{min}]}{p_f r_f} + 2 \\ &= \frac{k_1}{r_1} + \frac{k_2}{r_2} - \frac{\mathbb{E}[M_{min}]}{p_f}\left[\frac{1}{r_f} - \frac{1-p_f}{r_s}\right] + 2 + \frac{1}{p_f\min\{r_1,r_2\}}.\end{aligned}$$

Taking the $\limsup$ of the above and taking into account Theorem 7 completes the proof. ■

Combining theorems 7 and 8 proves theorem 5.

Constantinos Fragiadakis received his diploma in Computer and Telecommunications Engineering from University of Thessaly, Greece in 2013. He is currently a M.S student in the same University and works for the Center for Research and Technology Hellas (CERTH). His research interests are in the area of mathematical modeling, stochastic optimization and optimal control of networking systems. His M.S studies are funded under scholarship from the National Scholarship Foundation of Greece.

Georgios S. Paschos received the diploma in electrical and computer engineering from Aristotle University of Thessaloniki, Thessaloniki, Greece, in 2002, and the Ph.D. degree from the University of Patras, Patras, Greece, in 2006, both in electrical and computer engineering. From 2012 to 2014, he was a Postdoctoral Associate with the Laboratory for Information and Decision Systems (LIDS),Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. He joined Huawei Research France in 2014. His main interests are in the areas of wireless communications, networks, and stochastic modeling.

Leonidas Georgiadis (S76M78SM96) received the Diploma degree in electrical engineering from Aristotle University of Thessaloniki, Thessaloniki, Greece, in 1979, and the M.S. and Ph.D. degrees in electrical engineering from the University of Connecticut, Storrs, CT, in 1981 and 1986, respectively. Since 1995, he has been with the Telecommunications Department, Aristotle University, Thessaloniki, Greece. His interests are in the areas of wireless networks, high-speed networks, distributed systems, routing, scheduling, congestion control, modeling, and performance analysis.

Leandros Tassiulas (S89M91SM05F07) received the Diploma from the Aristotle University of Thessaloniki, Thessaloniki, Greece, in 1987, and the Ph.D. degree from the University of Maryland, College Park, MD, USA, in 1991, both in electrical engineering. He has been a Professor of telecommunication networks with the Department of Computer and Communication Engineering, University of Thessaly, Volos, Greece, since 2002 and Associate Director of the Informatics and Telematics Institute of the Center for Research and Technology Hellas (CERTH). His research interests are in the field of computer and communication networks with emphasis on fundamental mathematical models and algorithms, architectures and protocols of wireless systems, sensor networks, novel internet architectures, and network testbed experimentation.