Inter-Session Network Coding Schemes for 1-to-2 Downlink Access-Point Networks with Sequential Hard Deadline Constraints

Xiaohang Li, Chih-Chun Wang, and Xiaojun Lin
Center for Wireless Systems and Applications, School of ECE, Purdue University, West Lafayette, IN 47907
Email: {li179, chihw}@purdue.edu, linx@ecn.purdue.edu

***Abstract*—Next generation wireless networks will carry traffic from a wide range of applications, and many of them may require packets to be delivered before their respective deadlines. In this paper, we investigate using inter-session network coding to send packets wirelessly for two *deadline-constrained* unicast sessions. Specifically, each unicast session aims to transmit a file, whose packets have hard sequential deadline constraints. We first characterize the corresponding deadline-constrained capacity region under heterogeneous channel conditions and heterogeneous deadline constraints. We show that this deadline-constrained capacity region can be achieved *asymptotically* by modifying the existing *generation-based (G-B) schemes*. However, despite its asymptotic optimality, the G-B scheme has very poor performance for small and medium file sizes. To address these problems, we develop a new immediately-decodable network coding (IDNC) scheme that empirically demonstrates much better performance for short file sizes, and we prove analytically its asymptotically optimality when used to send large files. Our analysis uses a novel version of drift analysis, which could also be of independent interest to other IDNC schemes.**

I. Introduction

The advance of broadband wireless technologies has enabled a number of innovative wireless services. Many of these applications (such as video streaming) may require packets to be delivered before their respective deadlines. In this paper, we consider the scenario of sending two unicast deadline-constrained sessions over an unreliable wireless channel. Each unicast session downloads a *different* file, from a common base-station (BS). Each packet of the file has a delivery deadline, which is sequentially placed along the time axis. If a packet is not delivered before its deadline, it is considered useless. The underline channel is unreliable and may randomly cause packet erasure (packet drop). We are interested in using inter-session network coding (NC) to improve the deadline-constrained throughput in this simplified abstraction. As we will see, the asymmetry due to heterogeneous channel conditions and heterogeneous deadlines imposes new challenges in this deadline-constrained 2-unicast setting.

It is well-known that without deadline constraints, NC can increase the throughput [1] while still admitting efficient implementation [2], [3]. Such results have been obtained for both the *single-multicast* setting, where all destinations are requesting the same file, [4], or the *multiple-unicast* setting, where each destination requests its own file [5]. However, if not properly designed, NC could introduce "decoding delay." For example, in the *generation-based (G-B) schemes* [3], each user must accumulate a sufficient number of coded packets from a generation before it can decode any information packet, which incurs long delay. How to design an NC scheme subject to the deadline constraints becomes a challenging problem as one needs to carefully control the decoding delay while maximizing the throughput.

Existing studies on inter-session NC transmission schemes either do not account for the lossy wireless channel, or do not consider the delay/hard-deadline requirement. For example, the studies in [6]–[8] mostly focus on lossless channels. For noisy channels, [5] proposes a practical network coding scheme for multiple unicast-sessions while [9] characterize the corresponding information-theoretic capacity region. Recently, [10] characterizes the capacity region of 2-session unicast for an access-point network with broadcast packet erasure channels. These studies focus on throughput without considering delay. In contrast, our paper focuses on the delay aspect when coding over two unicast sessions. Readers are referred to [4], [11]–[14] and the references therein for the delay analysis in the simpler setting of a single multicast/broadcast session.

In this work, we first demonstrate that, by modifying the generation-based (G-B) scheme, it asymptotically achieves the optimal deadline-constrained capacity for large file sizes. However, this asymptotically optimality requires both the generation size and the file size to approach infinity. As a result, the G-B scheme has poor deadline-constrained throughput for practical settings of small to medium file sizes. To combat the delay inefficiency, recent practical NC protocols have focused more on the "immediately decodable" NC (IDNC) schemes [4], [5], [15]. In this work, we develop a new IDNC scheme that remedies the drawback of the G-B schemes for short-to-medium file sizes. Prior studies of similar IDNC schemes either do not consider deadline-constraints at all [16], or only consider the single multicast case [4].

This paper makes two main contributions. First, we develop new IDNC schemes that demonstrate numerically much better performance for small file sizes. The design of the proposed schemes are highly non-trivial and has to carefully avoid the potential pitfalls of existing IDNC schemes under the new setting of 2-unicast with heterogeneous channel conditions and deadline constraints (see Section V-A). Our second contribution is to prove rigorously that the proposed IDNC schemes are optimal in the *asymptotic regime of large files.* Our analysis uses a novel version of drift analysis, which could also be of independent interest to other IDNC schemes. Finally, we

This work has been partially supported by the NSF grants CNS-0721484, CNS-0721477, CNS-0643145, CCF-0845968, CNS-0905331, ECCS-1407603, CCF-1422997, CNS-1457137, and ARO grant W911NF-14-1-0368, as well as two grants from Purdue Research Foundation. Part of this work has appeared in Allerton Conference 2011 as an invited paper.

believe that our study of the 2-user case uncovers interesting and non-trivial insights that could serve as a precursor to the full study for the case with an arbitrary number of users.

This paper is organized as follows. Section II introduces the system model. Section III discusses the capacity region with deadline constraints. Section IV studies the G-B scheme for sequential deadline constraints and reveals its poor performance at small file sizes. Section V proposes a new IDNC scheme, and demonstrates empirically its superior performance. Section VI provides the throughput analysis of IDNC schemes for asymptotically large file sizes. Section VII presents the simulation results for the proposed IDNC schemes. Section VIII concludes the paper.

II. The Setting

Consider the scenario that the base station (BS) sends two files to 2 users, respectively. That is, user j is interested in file j for $j = 1, 2$. Sometimes we also use "session j" to refer to (the transmission of) the data packets for user j. Each file j contains N_j packets, denoted by $\{X_{j,n}\}_{n=1}^{N_j}$.

Assume slotted transmission. We define the time when the BS is ready to transmit as the time origin, and assume that all packets are available at the BS at time 0. Each packet $X_{j,n}$ ($j = 1, 2$) has a deadline $\tau_{j,n}$ such that after time slot $\tau_{j,n}$ the packet $X_{j,n}$ is no longer useful for user j. We assume

$$\tau_{j,n} = \lambda_j \cdot n, \quad \forall n \in \{1, \ldots, N_j\},$$

where λ_j is the (sequential) deadline increment for session j. We allow the deadlines to differ across the two sessions, i.e., we may have $\lambda_1 \neq \lambda_2$. Without loss of generality, we assume that $T = \lambda_1 N_1 = \lambda_2 N_2$, that is, the total duration T for each file transmission is the same.[1]

We consider random and unreliable single-hop wireless broadcast channels. Both users can overhear the transmission with certain probability. For $j = 1, 2$, we use $C_j(t) = 1$ to denote the event that user j can receive a packet successfully at time t; and $C_j(t) = 0$, otherwise. We assume that $C_j(t)$ is independently and identically distributed (i.i.d.) across time, and $C_1(t)$ and $C_2(t)$ are independent. The success probabilities for channels 1 and 2 are denoted by p_1 and p_2, respectively, where p_1 may be different from p_2. We assume that both p_1 and p_2 are known to the BS, at the end of each time slot, the BS has perfect feedback from both users regarding whether the transmitted packet has been successfully received.

Our goal is to design a coding/scheduling policy that maximizes the number of successful (unexpired) packet receptions. More specifically, let $D_j(n) = 1$ if user j can successfully decode/recover $X_{j,n}$ before its deadline $\tau_{j,n}$; and $D_j(n) = 0$, otherwise. We define the total number of unexpired successes of user j by $N_j^{\text{success}} \triangleq \sum_{n=1}^{N_j} D_j(n)$, and call $\frac{\mathsf{E}\{N_j^{\text{success}}\}}{N_j}$ the *timely delivery ratio* of user j. Our goal is to maximize the minimum of the timely delivery ratios, i.e., maximizing $\min\left(\frac{\mathsf{E}\{N_1^{\text{success}}\}}{N_1}, \frac{\mathsf{E}\{N_2^{\text{success}}\}}{N_2}\right)$. Note that if the file size is large, the actual realization of $\frac{N_j^{\text{success}}}{N_j}$ will be fairly close to its expected value and we thus focus only on reporting the mean value. For small file sizes, we will also compare the mean and the actual distribution of $\frac{N_j^{\text{success}}}{N_j}$ by numerical simulations.

Remark: Our notions of delivery ratios subject to deadline constraints are similar to other studies in the literature for deadline constrained wireless control algorithms [17]. We note that, for practical purposes, the above model may be somewhat simplistic. For example, for video streaming applications, this model does not capture the different priority among different types of video frames and/or the potential dependency across video packets [18]–[20]. Nonetheless, this simple model allows us to capture the key tradeoff between delay and throughout when network coding is involved. As the readers will see, even under this simple model, one needs to carefully control the coded and uncoded transmissions to fully utilize the potential gain due to network coding.

III. The Deadline-Constrained Capacity Region

Recall that $T = \lambda_1 N_1 = \lambda_2 N_2$ and consider an interval $(0, T]$. Suppose that during this interval, on average $r_j T$ packets of flow-j can be delivered before their deadlines, where $r_j \triangleq \frac{\mathsf{E}\{N_j^{\text{success}}\}}{\lambda_j N_j}$. We refer to r_j as the *time-wise throughput* for flow-j, which differs from the *timely delivery ratio* only by a factor $1/\lambda_j$. Obviously, $r_j \leq \frac{1}{\lambda_j}$ since the best scenario is to deliver all N_j packets before $T = \lambda_j N_j$. In [10], it is shown that even when not considering the sequential deadline constraints, the best possible time-wise throughput (r_1, r_2) must satisfy:[2]

$$\frac{r_1}{p_1} + \frac{r_2}{1 - (1 - p_1)(1 - p_2)} \leq 1 \tag{1}$$

$$\frac{r_2}{p_2} + \frac{r_1}{1 - (1 - p_1)(1 - p_2)} \leq 1. \tag{2}$$

Since the capacity without deadlines is always an upper bound on the capacity with deadlines, the above analysis proves the following outer bound on the deadline-constrained capacity.

Proposition 1. *For any scheme in a deadline constrained system, the time-wise throughput vector (r_1, r_2) must satisfy:*

$$\mathcal{R} = \Big\{(r_1, r_2) : 0 \leq r_1 \leq \frac{1}{\lambda_1}, 0 \leq r_2 \leq \frac{1}{\lambda_2}, \text{ and } (r_1, r_2) \text{ satisfies (1) and (2) simultaneously}\Big\}. \tag{3}$$

We note that there are two cases depending on the values of p_1, p_2, λ_1, and λ_2. See Fig. 1, where the red and blue lines represent the constraints (1) and (2), respectively. The dashed rectangle highlights the special point $(\frac{1}{\lambda_1}, \frac{1}{\lambda_2})$. In Fig. 1(a)

[1] If the duration of one file is longer than the other, then after the completion of the shorter file, we can then treat the remaining packets of the longer session as a single, separate, unicast session, which can be easily analyzed since there is no other session to be coded together.

[2] The intuition behind these two inequalities is as follows. Consider (1) first. Since we would like to send $r_1 T$ packets to user 1, transmitting those packets (either in an uncoded or in a coded way) would require $\frac{r_1 T}{p_1}$ number of time slots on average. Note that even though sometimes we may use NC to serve two destinations simultaneously, roughly speaking before doing so some version of each session-2 packet needs to be received by at least one of the users before it can be mixed with a session-1 packet [10]. As a result at least $\frac{r_2 T}{1-(1-p_1)(1-p_2)}$ number of time slots should be dedicated to sending session-2 packets (not mixed with any session-1 transmission). Since the total time budget is T, the above heuristics imply (1). By swapping the roles of sessions 1 and 2, we also have (2).

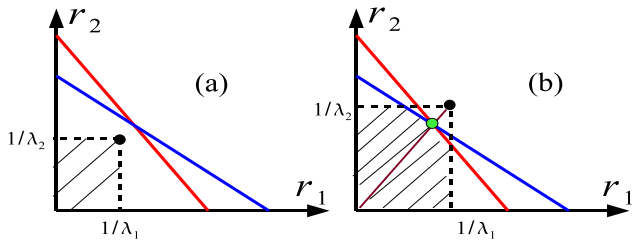


Fig. 1. Illustration of the deadline-constrained capacity region. (a) The over-provisioned case; and (b) The under-provisioned case.

$(\frac{1}{\lambda_1}, \frac{1}{\lambda_2})$ satisfies the constraints (1) and (2). We refer to this case as the "over-provisioned case", because the time-wise throughput is limited by $(\frac{1}{\lambda_1}, \frac{1}{\lambda_2})$ instead of the deadline-free capacity described by (1) and (2). In contrast, in Fig. 1(b) the point $(\frac{1}{\lambda_1}, \frac{1}{\lambda_2})$ violates (1) and (2), which we call the "under-provisioned case" since the deadline-constrained throughput region is limited by the deadline-free capacity. Clearly, in this case there is no hope to achieve $(\frac{1}{\lambda_1}, \frac{1}{\lambda_2})$.

We now draw the connection between the above capacity outer-bound $\mathcal{R}$ to our design objective. In the literature, a common design goal is to come up with an algorithm that achieves any point in $\mathcal{R}$. However, since in Section II we are only interested in maximizing the minimum of the timely delivery ratios, it is sufficient to focus on only one point. Specifically, define

$$\gamma \triangleq \min\left(\frac{1}{\frac{1/\lambda_1}{p_1}+\frac{1/\lambda_2}{p_1+p_2-p_1p_2}}, \frac{1}{\frac{1/\lambda_1}{p_1+p_2-p_1p_2}+\frac{1/\lambda_2}{p_2}}\right), \quad (4)$$

where $\gamma < 1$ in the under-provisioned case and $\gamma > 1$ in the over-provisioned. We also define $\gamma = 1$ to be the critically-provisioned case. By simple arithmetics and Proposition 1, one can easily prove that

$$\min\left(\frac{\mathsf{E}\{N_1^{\text{success}}\}}{N_1}, \frac{\mathsf{E}\{N_2^{\text{success}}\}}{N_2}\right) \le \min(1,\gamma). \quad (5)$$

Thus, in order to maximize the minimum of timely delivery ratios, it is sufficient to achieve the time-wise throughput vector $\min(1,\gamma)(\frac{1}{\lambda_1}, \frac{1}{\lambda_2})$ (which is shown as the green point in Fig. 1(b)). Hence, our goal in this work is to design new NC schemes that approaches the outer bound $\min(1,\gamma)$ in (5).

IV. Modifying Existing Generation-Based Schemes

The generation-based (G-B) scheme is widely used [3], [10] in existing throughout-oriented analysis, which divides the whole file into several generations and transmits each generation sequentially. In each time slot the BS only encodes packets that belong to the current generation. After receiving enough coded packets, the receiver can decode the entire generation. The BS then moves on to the next generation. In the following, we will show that, although the 2-unicast G-B scheme proposed in [10] may be modified to asymptotically achieve the outer bound in (5), for a deadline-constrained setting such a scheme has serious drawbacks and may yield poor performance for small and median file sizes.

We start with the scheme of [10]. For sessions 1 and 2 we choose the corresponding generation sizes to be M_1 and M_2, respectively, and we enforce that $\lambda_1 M_1 = \lambda_2 M_2$. In this way, both sessions will have the same number of generations. In the G-B scheme of [10], the BS first sends all session-1 and session-2 packets of the current generation in an uncoded manner until each packet is received by at least one user. If a session-j ($j = 1, 2$) packet is received by user j, then the BS does not need to consider such a packet any more. For the rest of the packets that have not been received by the desired users, the BS sends coded packets by randomly mixing them together, until both users can decode the entire generation. Then, the BS moves on to the next generation. [10] shows that this scheme can support the largest possible throughput vector when $M_1, M_2 \to \infty$.

However, this G-B scheme from [10] cannot be used directly in the deadline-constrained scenario due to the following two reasons. Firstly, due to the deadline constraints, roughly speaking the transmission of each generation of M_1 session-1 packets and M_2 session-2 packets must be carried out in approximately $\lambda_1 M_1 = \lambda_2 M_2$ time slots. However, if $(\frac{1}{\lambda_1}, \frac{1}{\lambda_2})$ falls into the under-provisioned case as in Fig. 1(b), by Proposition 1 it is simply impossible for every packet to meet its deadline constraint. One potential remedy for this problem is the following. According to the outer bound in (5), at most $\min(1,\gamma)$ fraction of packets can be supported. Therefore, one could deliberately discard $(1-\min(\gamma,1))$ fraction of the packets, i.e., those that can never be supported. The remaining packets now have a better chance to be decoded. Secondly, there is little time to perform coding for the first generation packets since the first generation packets will likely expire before the receiver collects a sufficient number of coded packets. One potential remedy for this issue is to discard the first generation of both sessions 1 and 2, and start encoding generation-2 packets from the very beginning. In this way, more time is allowed for all the subsequent encoding/decoding.

With the above two observations, the G-B scheme may be modified as follows for the deadline-constrained setting:

---

§ G-B-Scheme

1: Discard the first generation of both sessions 1 and 2. Set $\mathsf{GenID} \leftarrow 2$
2: For $j = 1, 2$ choose arbitrarily $M_j(1-\min(\gamma,1))$ user-j packets from the GenID-th generation and *drop those packets*, i.e, remove them from any future consideration.
3: **for** time $t = (\mathsf{GenID}-2)\cdot\lambda_1 M_1 + 1$ to time $t = (\mathsf{GenID}-1) * \lambda_1 M_1$ **do**
4: **if** there is still a user-j packet of the GenID-th generation that is not heard by any user **then**
5: The BS transmits one of such packets (that is not heard by any user) uncodedly.
6: **else**
7: The BS generates a single coded packets by randomly mixing (see [2]) all user-1 packets in the GenID-th generation that have been heard only by user 2, and all user-2 packets that have been heard only by user 1. The BS sends the RLNC-generated packet.
8: **end if**
9: **end for**

10: In the end of time $t = (\mathsf{GenID} - 1) * \lambda_1 M_1$, user 1 (resp. user 2) can decode its desired packets if it has received enough inter-session coded packets of the GenID-th generation.
11: GenID$\leftarrow$ GenID+1 and go back to Line 2.

One can easily prove that this G-B scheme can asymptotically achieve the upper bound in (5), under the following two conditions: (i) the generation size approaches infinity, and (ii) the file sizes N_1 and N_2 approach infinity [3]. The former is to ensure, by the law of large numbers, that a sufficient number of coded packets can be received in the time allotted to each generation so that they can be decoded [3]. The latter is to ensure that the normalized throughput penalty of dropping the first generation is negligible. Detailed discussion of the above algorithm can be found in our technical report [21].

Weakness of the G-B scheme: However, we want to make it clear that it is not our intention to advocate the above G-B scheme. Although it seems to attain the optimal throughput in the above asymptotic limits, it clearly will perform poorly if the file sizes are not very large. Specifically, for a fixed small-to-medium file size, the network code designer faces the following dilemma. If we choose large generation sizes, the effect of losing the first generation will be significant. If we choose small generation sizes, there will be a high chance that the users will not be able to receive coded packets for decoding within the time allotted to each generation.

These insights are verified by simulation, see Figs. 2 and 3. Here, we use "G-B M_1-M_2" to denote the above G-B scheme with generation sizes M_1 and M_2 for sessions 1 and 2, respectively. In Figs. 2 and 3, we plot the Cumulative Distribution Function (CDF) of the quantity below:

$$\frac{1}{\min(1,\gamma)} \min\left(\frac{\mathsf{E}\{N_1^{\text{success}}\}}{N_1}, \frac{\mathsf{E}\{N_2^{\text{success}}\}}{N_2}\right). \qquad (6)$$

which measures how close the timely delivery ratio is to the upper bound $\min(1,\gamma)$. Since (6) is upper bounded by 1, we draw a vertical line at 100% in Figs. 2 and 3. The curve for each scheme is collected from 1000 simulation instances. The only difference between Figs. 2 and 3 is the file sizes. When the file sizes are large (Fig. 2, $N_1 = N_2 = 40000$), using larger generation sizes gets closer to the upper bound. However, when the file sizes are small (Fig. 3 , $N_1 = N_2 = 400$), the G-B schemes suffer poor performance for both small and large generation sizes (4 and 40, respectively) due to the reasons that we explained earlier. The curve "IDNC" corresponds to the new scheme that we will later propose, which achieves superior performance for both large and small file sizes.

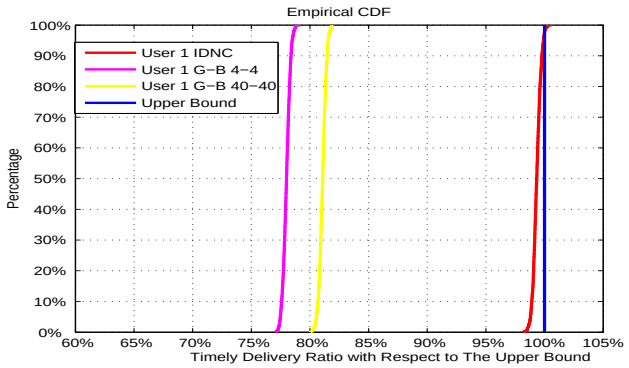


Fig. 2. CDF of user 1 for IDNC and G-B when $\lambda_1 = \lambda_2 = 3$, $N_1 = N_2 = 40000$, $p_1 = 0.4$, $p_2 = 0.4$, in 1000 instances of Monte Carlo simulation. The horizontal axis is the percentage with respect to the upper bound in (5).

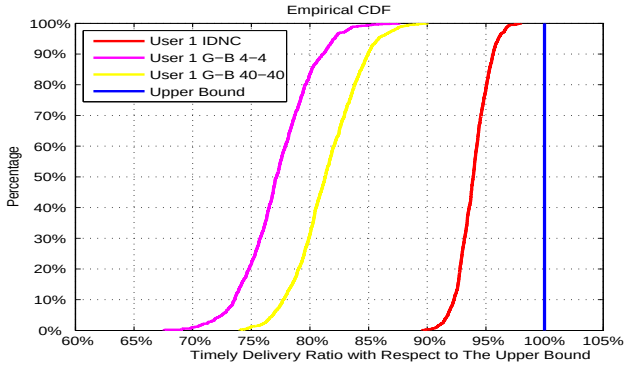


Fig. 3. CDF of user 1 for IDNC and G-B under a similar setting as in Fig. 2, except that now the file sizes are small $N_1 = N_2 = 400$.

V. The IDNC Scheme

A. The Basics of IDNC Schemes and Two Issues When Applying IDNC to A Deadline-Constrained System

To overcome the delay inefficiency of G-B schemes, recent practical protocols have focused more on the "immediately decodable" NC (IDNC) schemes [5], [15]. An IDNC scheme for two unicast sessions has the following structure. Suppose that users 1 and 2 are interested in packets X and Y, respectively. Initially, the BS sends X and Y uncodedly until each packet is received by at least one user. Suppose that due to random channel realization, user 1 has overheard Y but not X and user 2 has overheard X but not Y. We call the (unexpired) packet X (resp. Y) a coding opportunity of user 1 (resp. user 2). The BS can now combine the two coding opportunities and send $[X+Y]$, which serves two receivers simultaneously. Note the desired packet X (resp. Y) can be *immediately decoded* by user 1 (resp. user 2) upon receiving $[X+Y]$. With this feature, the IDNC schemes generally demonstrate much faster startup phase [22] and are more suitable for time-sensitive applications.

However, designing an IDNC scheme for the deadline-constrained setting is non-trivial. [4] designs an IDNC scheme that is throughput-optimal for the deadline-constrained *single-multicast* setting. However, when performing coding over 2 heterogeneous unicast sessions, we need to take into account the following new issues.

Issue 1: Not all packets can be delivered when the system is under-provisioned (Fig. 1(b)). As we discussed in Section IV, the best that one can hope is to approach the outer bound (5). Similar to the modified G-B scheme in Section IV, we propose to proactively drop on average $(1-\min(\gamma,1))$ fraction of the packets in the IDNC scheme.

Issue 2: The "leading user problem" that arises from the heterogeneity of the channels and deadlines. We note that Issue 2 is orthogonal to Issue 1: unlike issue 1 that happens

only when the system is under-provisioned, Issue 2 may happen for both over- and under-provisioned systems. For better illustration of Issue 2, consider an over-provisioned scenario (Fig. 1(a)) for which Issue 1 does not exist. Recall that in a typical IDNC scheme, each packet is sent repeatedly in an uncoded fashion until it is received by at least one user. Therefore, after sending the session-1 packets uncodedly, the N_1 session-1 packets are partitioned into two groups depending on their reception status: Those received by user 1 already, and those received by user 2 only but not by user 1. (The latter is what we defined as the coding opportunity of user 1.) By simple probability computation, the average number of coding opportunities of user 1 is $\frac{N_1 \cdot p_2(1-p_1)}{1-(1-p_1)(1-p_2)}$. By symmetry, the average number of coding opportunities of user 2 is $\frac{N_2 \cdot p_1(1-p_2)}{1-(1-p_1)(1-p_2)}$.

Note that a coding opportunity of user 1 will later be combined with that of user 2 to generate a coded packet that is the linear sum of the two. Whenever the linear sum is received by user j, $j = 1, 2$, we "consume" one coding opportunity of user j. As a result, it takes on average $\frac{N_1 \cdot p_2(1-p_1)}{(1-(1-p_1)(1-p_2))p_1}$ number of coded transmissions to "use up" the coding opportunities of user 1; and it takes $\frac{N_2 \cdot p_1(1-p_2)}{(1-(1-p_1)(1-p_2))p_2}$ number of coded transmissions to "use up" the coding opportunities of user 2. However, due to the heterogeneity of the channels and deadlines, these two numbers may not match. This mismatch creates a new problem. Specifically suppose

$$\frac{N_1 \cdot p_2(1-p_1)}{(1-(1-p_1)(1-p_2))p_1} > \frac{N_2 \cdot p_1(1-p_2)}{(1-(1-p_1)(1-p_2))p_2}. \quad (7)$$

The above inequality implies that after consuming all user-2 coding opportunities, we may still have some user-1 coding opportunities that have not been used up. In all the existing IDNC solutions, after fully consuming all the coding opportunities of user-2, the scheduler will declare the remaining "not-paired" coding opportunities of user-1 as "leftovers." Since those leftover coding opportunities have no packet to be combined with, they need to be transmitted *uncodedly*. This observation indicates that, even without deadline constraints, at least two strategies are needed. Strategy #1 is to "wait until a coding opportunity can be combined with another one," which generally leads to higher throughput. However, the second Strategy #2, i.e., to "transmit a coding opportunity uncodedly" is also necessary when there is a mismatch between the numbers of coding opportunities. Without Strategy #2, those leftover coding opportunities will wait forever in the queue, which also hurts the throughput.

If there are no deadline constraints, one may defer Strategy #2 until the very end, when it becomes obvious which coding opportunities are "leftover." While such a design is optimal for throughput, it leads to catastrophic consequences under deadline constraints. Specifically, under deadline constraints, it is critical to make transmission decisions as early as possible. If we were to defer the decision of "which coding opportunities to apply Strategy #2" to the very end, these coding opportunities would have already expired.

To recover from this sub-optimality, when (7) holds, an optimal IDNC scheme should proactively prevent flow-1 from generating too many coding opportunities rather than wait until the end of transmission to deal with the leftover packets. Towards this end, our IDNC scheme declares some flow-1 packets as not-intended-for-coding from the very beginning. Namely, even if these flow-1 packets (which will be labeled as uncoded-Tx-only in our proposed IDNC scheme below) are overheard by d_2, we do not consider them as coded opportunities for flow-1 any more. Rather, our scheme will continue transmitting them in an uncoded fashion. Such a mechanism will make sure that all coding-opportunities of flows 1 and 2 can be combined with each other and there is no leftover in the end of the transmission. An equivalent view of our design choice is that we pre-declare a carefully-computed fraction of coding opportunities of flow-1 as uncoded-Tx-only and apply Strategy #2 to them, so that in the end there is no leftover packet in the system.

For future reference, we say "user 1 is a leading user" if (7) is satisfied, since there are more coding opportunities of user 1 than that could be combined with the coding opportunities of user 2. Otherwise, we say user 2 is the leading user. Noticing $\lambda_1 N_1 = \lambda_2 N_2 = T$, we can easily show that the following definitions are equivalent:

$$\text{user 1 is leading} \iff \frac{\lambda_1 p_1^2(1-p_2)}{\lambda_2 p_2^2(1-p_1)} < 1.$$

B. The Proposed IDNC Scheme

We now design a new IDNC scheme that outperforms the G-B schemes, which contains the following 3 major ingredients.
(a) To resolve Issue 1, we proactively drop $(1 - \min(\gamma, 1))$ fraction of packets from both sessions.
(b) To resolve Issue 2, we proactively declare $1 - \min(\frac{\lambda_1 p_1^2(1-p_2)}{\lambda_2 p_2^2(1-p_1)}, 1)$ fraction of user-1 packets as "ineligible" for coded transmission. These packets, even if they become coding opportunities later, will only be transmitted in an uncoded manner.
(c) Finally, since coded transmissions are more efficient, we always prioritize coded transmissions over uncoded transmissions after fully implementing mechanisms (a) and (b). This is in contrast with the IDNC scheme [10] that prioritizes coded transmission without implementing (a) and (b).

We now describe the details of the proposed IDNC scheme. To begin with, we introduce some definitions. In our new IDNC scheme, the BS keeps two registers n_1 and n_2. The register n_j is to keep track of the next uncoded packet to be sent for session j. Since both n_1 and n_2 evolve over time, we sometimes use $n_j(t)$ to denote the value of n_j at the end of time t. The BS also keeps two lists of packets: L_{10} and L_{01}. List L_{01} contains all unexpired coding opportunities of user 1 (those heard by user 2 but not yet by user 1). Symmetrically, list L_{10} contains all unexpired coding opportunities of user 2. Each packet is also associated with a status, which can take one of the following four values "not-processed", "dropped", "uncoded-Tx-only" and "coding-eligible". The BS uses two arrays status1$[i]$, $i = 1, \cdots, N_1$, and status2$[i]$, $i = 1, \cdots, N_2$ to keep track of the status of the session-1 and session-2 packets, respectively. Initially, all packets have the status not-processed, meaning that the packet status has not been decided. In addition, the BS keeps 4 floating-point

registers, denoted by x_1, x_2, y_1, and y_2, which will be used to decide how many packets to drop when the system is under-provisioned. We also assume that at the end of each time slot, both users send an ACK or NACK message back to the BS.

We now provide the pseudo code of the proposed IDNC scheme. The corresponding intuition will be elaborated right after the pseudo code.

1: $n_1 \leftarrow 1$, $n_2 \leftarrow 1$, $L_{10} \leftarrow \emptyset$, $L_{01} \leftarrow \emptyset$, status1$[i]$ $\leftarrow$not-processed, status2$[i]$ $\leftarrow$not-processed, for all i; $x_1, y_1, x_2, y_2 \leftarrow 0$.
2: **for** $t = 1$ to $\lambda_1 N_1$ **do**
3: In the beginning of time t, run the sub-routine SCHEDULE-PACKET-TRANSMISSION.
4: In the end of time t, run the sub-routine UPDATE-PACKET-STATUS.
5: **end for**

The two sub-routines are described separately as follows.

§ SCHEDULE-PACKET-TRANSMISSION

1: **if** $n_2 \leq N_2$ & $n_1 \leq N_1$ **then**
2: **while** status1$[n_1]$ =not-processed **do**
3: Set $x_1 \leftarrow x_1 + \min(\gamma, 1)$.
4: **if** $\lfloor x_1 \rfloor > y_1$ where $\lfloor \cdot \rfloor$ is the floor function **then**
5: Set $y_1 \leftarrow \lfloor x_1 \rfloor$.
6: With probability $\min\left(\frac{\lambda_1 p_1^2(1-p_2)}{\lambda_2 p_2^2(1-p_1)}, 1\right)$, set status1$[n_1]$ $\leftarrow$ coding-eligible.
7: Otherwise, set status1$[n_1]$ $\leftarrow$ uncoded-Tx-only.
8: **else**
9: Set status1$[n_1]$ $\leftarrow$dropped.
10: Set $n_1 \leftarrow n_1 + 1$.
11: **end if**
12: **end while**
13: Repeat the steps from Line 2 to Line 12 with the roles of users 1 and 2 swapped, i.e, we focus on user 2 now.
14: **if** both L_{10} and L_{01} are non-empty **then**
15: Choose the oldest packet X_{1,j_1^*} from L_{01} and the oldest packet X_{2,j_2^*} from L_{10}. Broadcast the sum $[X_{1,j_1^*} + X_{2,j_2^*}]$.
16: **else**
17: **if** $n_1\lambda_1 \leq n_2\lambda_2$ **then**
18: Send uncoded packet X_{1,n_1} directly.
19: **else if** $n_1\lambda_1 > n_2\lambda_2$ **then**
20: Send uncoded packet X_{2,n_2} directly.
21: **end if**
22: **end if**
23: **else**
24: Choose the oldest unexpired packets in the system (including those in $L_{01} \cup L_{10}$ and those haven't been sent) and send that packet uncodedly.
25: **end if**

§ UPDATE-PACKET-STATUS

1: **if** an uncoded packet X_{1,n_1} was sent in the current time slot **then**
2: **if** X_{1,n_1} is received by d_1 **then**
3: Set $n_1 \leftarrow n_1 + 1$.
4: **else if** X_{1,n_1} was received only by d_2 and status1$[n_1]$ =coding-eligible **then**
5: Add X_{1,n_1} to L_{01} and set $n_1 \leftarrow n_1 + 1$.
6: **end if**
7: **else if** an uncoded packet X_{2,n_2} was sent in the current time slot **then**
8: Repeat the steps from Line 2 to Line 6 with the roles of users 1 and 2 swapped.
9: **else**
10: Suppose the coded packet being sent is $[X_{1,j_1^*} + X_{2,j_2^*}]$, the sum of X_{1,j_1^*} and X_{2,j_2^*}.
11: **if** $[X_{1,j_1^*} + X_{2,j_2^*}]$ was received by d_1 **then**
12: Remove X_{1,j_1^*} from L_{01}.
13: **end if**
14: **if** $[X_{1,j_1^*} + X_{2,j_2^*}]$ was received by d_2 **then**
15: Remove X_{2,j_2^*} from L_{10}.
16: **end if**
17: **end if**
18: Remove all expired packets from the system.

We next provide more detailed explanation of the pseudo-code. Readers can refer to the high-level idea explained at the beginning of the subsection. Let us first focus on the sub-routine SCHEDULE-PACKET-TRANSMISSION. Line 1 checks whether we have reached the terminal phase of the transmission. When either $n_1 > N_1$ or $n_2 > N_2$ holds (Line 24), we simply choose the oldest available packet to transmit. When we are in the main loop (Lines 2-22), i.e., when both $n_1 \leq N_1$ and $n_2 \leq N_2$ hold, we first assign the packet status for both X_{1,n_1} and X_{2,n_2} (Lines 2 to 12), if their status is still "not-processed". There are two considerations when we assign the status to these packets. First, in order to resolve Issue 1 in the under-provisioned case, we would like to set $(1 - \min(\gamma, 1))$ fraction of packets to "dropped". This step corresponds to Lines 3-4 and Lines 9-10. Suppose $\gamma < 1$ (i.e., the under-provisioned case). We can see that x_1 increases at the rate of γ, and $y_1 = \lfloor x_1 \rfloor$. Each time x_1 exceeds the next integer value (i.e., $\lfloor x_1 \rfloor > y_1$), a packet is set to either "coding-eligible" or "uncoded-Tx-only". Otherwise, if x_1 does not exceed the next integer value, a packet is dropped (Lines 9-10). Hence, it is easy to see that we drop roughly $((1/\gamma) - 1)$ number of packets every $1/\gamma$ packets. Thus, the long term average packet drop rate is $\frac{1/\gamma - 1}{1/\gamma} = 1 - \gamma$. On the other hand, if $\gamma \geq 1$ (i.e., the critical or over-provisioned case), Lines 3-4 ensure that no packets are dropped.

To resolve Issue 2, we proactively label some packets as "uncoded-Tx-only". Specifically, If user 1 is the leading user, then $\frac{\lambda_1 p_1^2(1-p_2)}{\lambda_2 p_2^2(1-p_1)} < 1$, as discussed in (7). Lines 6 to 7 ensure that $1 - \min(\frac{\lambda_1 p_1^2(1-p_2)}{\lambda_2 p_2^2(1-p_1)}, 1)$ fraction user-1 packets have their status set to uncoded-Tx-only. Note that in this case, all user-2 packets have their status set to coding-eligiblesince now $\frac{\lambda_2 p_2^2(1-p_1)}{\lambda_1 p_1^2(1-p_2)} > 1$ and $\min(\frac{\lambda_2 p_2^2(1-p_1)}{\lambda_1 p_1^2(1-p_2)}, 1) = 1$ when executing Line 13. The case when user 2 is the leading user is symmetric.

Once we finish setting the packet status, we give priority to coded transmissions first (Lines 14 and 15). If we cannot find a pair of complementary coding opportunities, then we

alternate between sending uncoded packets for users 1 and 2, by comparing the values of $n_1\lambda_1$ and $n_2\lambda_2$ (Lines 17 to 21). Namely, we choose the next uncoded packet depending on which is the closest to expire. Lines 17 to 21 also lead to the following simple lemma that bounds the difference between $\lambda_1 n_1(t)$ and $\lambda_2 n_2(t)$, the proof of which can be found in our technical report [21].

Lemma 1. *For any time slot t, we have* $|\lambda_1 n_1(t) - \lambda_2 n_2(t)| \leq \max(\lambda_1, \lambda_2) \cdot \lceil \frac{1}{\gamma} \rceil$.

The sub-routine UPDATE-PACKET-STATUS is more straightforward. If an uncoded packet X_{1,n_1} was sent and received by d_1 (see Lines 2-3), then there is no need to retransmit this packet. We simply shift our focus to the next packet ($n_1 \leftarrow n_1 + 1$). If X_{1,n_1} is received by d_2 but not by d_1, then this packet may become a new coding opportunity. However, as explained in Issue 2, if user 1 is the leading user, then sometimes we need to forgo a coding opportunity and continue sending it in an uncoded manner. This is decided by the packet status specifically. If the packet status was set to uncoded-Tx-only, then we do not put the overheard packet X_{1,n_1} in the list L_{01}. That is, X_{1,n_1} will not participate in any future coding transmissions and will still be transmitted uncodedly next time. Only when the packet status is coding-eligible (see Line 4) will the overheard X_{1,n_1} be put into the list L_{01}. On the other hand, if user 1 is the leading user, then our scheme will label all flow-2 packets as coding-eligible. Later, our proof will show that this design achieves the right balance, i.e., with more and more coding opportunities accumulated in the long run, the coding opportunities of flow-1 and flow-2 will eventually balance themselves. Finally, if a coded packet was sent, Lines 11-18 simply perform packet updates to remove the packets that have either expired or have already been decoded by the target user.

The proposed IDNC scheme achieves much better performance than G-B schemes for both large and small file sizes. In Figs. 2 and 3, we compare the performance of IDNC schemes with different G-B schemes. When the file size is large ($N_1 = N_2 = 40000$, Fig. 2), the IDNC scheme is very close to the upper bound in (5). When the file size is small ($N_1 = N_2 = 400$, Fig. 3), the IDNC scheme is still much closer to the upper bound in (5) than the G-B schemes.

C. The G-B scheme versus the proposed IDNC scheme

We conclude this section by summarizing the benefits of the proposed IDNC scheme over the G-B scheme both in terms of the asymptotic limits and the practical performance. First, note that in the G-B scheme, if we fix the generation size, even when we let the file size grows to infinity, the performance of the G-B scheme will still be strictly bounded away from the optimal. In other words, in order for the G-B scheme to achieve the optimal in the asymptotic limit of infinite file sizes, the generation size must be scaled in a careful way depending on the file sizes. In contrast, the design of IDNC scheme does not require such a dependency on the file sizes. Specifically, we will show in the next section that, as (N_1, N_2) increases, the IDNC scheme will automatically attain the optimal performance. In this sense, the IDNC scheme is *universal* and easier to use in practice.

Second, as shown in Fig. 3 our proposed IDNC scheme has much better performance for small or medium file sizes. This advantage over the G-B scheme is due to their different ways of treating the first group of packets. In the G-B scheme, since it mixes one generation of packets as a group, all of them need to be decoded at the same time. Thus, their delivery time is effectively changed to the *last* delivery time among this group/generation. However, the earliest deadline of the first generation is $t = \lambda_1 \cdot 1$ (i.e., the deadline of the first packet), which is very short, and so are the deadlines of many other packets in the first generation. Thus, it is unlikely that any G-B scheme can meet their deadline constraints. Due to this reason, the G-B scheme discards the first generation altogether as if all packets have already expired. However, this leads to significant performance degradation for small and medium file sizes. In contrast, the IDNC scheme mixes packets one by one. Therefore, IDNC has the luxury of waiting until $t = \lambda_1 \cdot 1$ to discard packet 1 (if it has not been delivered yet), and then waiting until $t = \lambda_1 \cdot 2$ to discard packet 2 (if it has not been delivered). The stark contrast between the G-B scheme (mixing a generation of packets) and the IDNC scheme (mixing packets one by one) is the reason why the IDNC scheme has much better performance for small and medium file sizes in practice.

VI. ASYMPTOTIC PERFORMANCE OF THE IDNC SCHEME

In addition to the numerically verified superior performance of the IDNC scheme for small and medium file sizes, in this section we will prove its throughput optimality for the case of *asymptotically large* file sizes, where the impact of randomness could be more analytically quantified. Motivated by the success of various existing asymptotic throughput analysis [9], [10], we believe that such an analysis is very useful since it provides a concrete understanding why the proposed IDNC scheme is operating in the right way.

We note that for G-B schemes (e.g., in Section IV), the coding operations for one generation is completely decoupled from the operations in the next generation. Thus, by taking large generation sizes, we can apply a "law of large numbers" argument within each generation. In contrast, for IDNC schemes all the packets are treated equally and mingled seamlessly through mechanisms (a) to (c) in Section V-B, which is the reason behind its superior performance. Due to this lack of separation, it is much more challenging when applying any "law of large numbers" argument. A main contribution of this paper is to provide a novel Lyapunov drift analysis to address this difficulty.

Lemma 2 below is critical to characterize the asymptotic performance of our IDNC scheme for large file sizes.

Lemma 2. *Consider our IDNC scheme with system parameter values* λ_1, λ_2, p_1, *and* p_2. *Then for any* $\epsilon > 0$, *there exists* $B > 0$ *such that for all fixed* t_1 *and* t_2 *satisfying* $(t_2 - t_1) \geq B$,

we have for $j = 1, 2$,

$$\mathsf{E}\Big\{n_j(t_2) - n_j(t_1)\Big|t_2 < \min(\lambda_1 n_1(t_1), \lambda_2 n_2(t_1))\Big\} \leq \frac{(t_2 - t_1)\max(\gamma, 1)(1+\epsilon)}{\lambda_j}. \quad (8)$$

The detailed proof for Lemma 2 can be found in Appendix A and [21]. The high-level interpretation of this lemma is provided as follows. Consider any two fixed time instances t_1 and t_2. The conditioned event $t_2 < \min(\lambda_1 n_1(t_1), \lambda_2 n_2(t_1))$ states that both $n_1(t_1)$ and $n_2(t_1)$ will not expire by the time t_2. Without loss of generality, focus on user 1 ($j = 1$). The term $n_1(t_2) - n_1(t_1)$ quantifies how many new session-1 packets have been "injected" to the system during the time interval $(t_1, t_2]$. For the right hand side of (8), if $\gamma \leq 1$ (i.e., the under- or critically-provisioned case), then $\max(\gamma, 1) = 1$ and the term $(t_2 - t_1)/\lambda_1$ corresponds to the number of session-1 packets that will expire in the interval $(t_1, t_2]$. Thus, Lemma 2 states that the expected growth of $n_1(t)$ is bounded from above by a value roughly proportional to how fast the packets of session 1 expire. On the other hand, if $\gamma > 1$ (i.e., the over-provisioned case), the expected growth of $n_1(t)$ could be faster, but still bounded from above.

Lemma 2 applies a Law of Large Numbers (LLN) argument in a novel way. Note that when conditioning on $t_2 < \min(\lambda_1 n_1(t_1), \lambda_2 n_2(t_1))$, none of these newly injected packets $X_{1,n_1(t_1)}$, $X_{1,n_1(t_1)+1}$, $\cdots$, $X_{1,n_1(t_2)-1}$ will expire during time $(t_1, t_2]$. Therefore, those packets will have a similar behavior as if in a system without deadline constraints. Thankfully, when there are no deadline constraints, we do not need to worry about the highly complicated relationship between scheduling and packet expiration, which greatly simplifies the analysis. Indeed, we can use the LLN (recall that $t_2 - t_1 \geq B$ is sufficiently large) to bound the numbers of uncoded and coded transmissions during $(t_1, t_2]$, which eventually leads to (8). For details, see Appendix A and [21].

Lemma 3. *For* $j = 1, 2$, $\mathsf{E}\{n_j(t)\} \leq \frac{\max(\gamma,1)t}{\lambda_j} + o(t)$.

Lemma 3 can be viewed as a stronger version of Lemma 2 that the expected growth of $n_j(t)$ is upper bounded by a value that relates to how fast the packets expire. Intuitively, if $n_j(t)$ advances too quickly, then the conditioned event in (8) will occur, and thus the growth of $n_j(t)$ will slow down due to Lemma 2. We provide the formal proof below.

Proof: For ease of exposition, we first assume that user 1 is the leading user. For any given $\epsilon > 0$, we use the value of B specified in Lemma 2. We will describe how to choose the value of ϵ in the later part of this proof. For a given $\epsilon > 0$, we define $q_j(t) \triangleq n_j(t) - \frac{\max(\gamma,1)t(1+2\epsilon)}{\lambda_j}$ and $q_j^+(t) \triangleq \max(q_j(t), 0)$ for $j = 1, 2$. We first note that $n_j(t)$, the index of the next to-be-sent uncoded packets must satisfy $n_j(t) \geq \frac{t}{\lambda_j}$. Next we show that $q_j(t)$ and $q_j^+(t)$ cannot be very large due to Lemma 2. Towards this end, consider a (t_1, t_2) pair satisfying $B_0 \triangleq t_2 - t_1 > B$ and any fixed constant K satisfying $K > \frac{\max(\gamma,1)(1+2\epsilon)}{\lambda_1}$. We first show that

$$\text{If } q_1^+(t_1) > K \cdot B_0, \text{ then } q_1^+(t_1 + B_0) = q_1(t_1 + B_0) \geq 0. \quad (9)$$

The reason is that

$$q_1^+(t_1) > KB_0 \geq 0 \Rightarrow q_1(t_1) = n_1(t_1) - \frac{\max(\gamma,1)t_1(1+2\epsilon)}{\lambda_1} > KB_0. \quad (10)$$

As a result,

$$q_1^+(t_1 + B_0) = n_1(t_1 + B_0) - \frac{\max(\gamma,1)(t_1 + B_0)(1+2\epsilon)}{\lambda_1} \geq n_1(t_1) - \frac{\max(\gamma,1)(t_1 + B_0)(1+2\epsilon)}{\lambda_1} \quad (11)$$

$$> KB_0 - \frac{\max(\gamma,1)B_0(1+2\epsilon)}{\lambda_1} > 0, \quad (12)$$

where (11) follows from $n_1(t)$ is non-decreasing with respect to t; and (12) follows from (10). Since $q_1(t_1 + B_0) > 0$, we thus have $q_1^+(t_1 + B_0) = q_1(t_1 + B_0)$. We now note that by the definition of $q_1(t)$, it is easy to see that the condition $q_1(t_1) > KB_0 + \max\left(1, \frac{\lambda_2}{\lambda_1}\right)$ implies that $\lambda_1 n_1(t_1) - \max(\lambda_1, \lambda_2) > t_2$. By Lemma 1, this further implies that $t_2 \leq \min(\lambda_1 n_1(t_1), \lambda_2 n_2(t_1))$. We then have

$$\mathsf{E}\Big\{q_1^+(t_1 + B_0) - q_1^+(t_1)\Big|q_1(t_1) > KB_0 + \max(1, \frac{\lambda_2}{\lambda_1})\Big\} = \mathsf{E}\Big\{q_1(t_1 + B_0) - q_1(t_1)\Big|q_1(t_1) > KB_0 + \max(1, \frac{\lambda_2}{\lambda_1})\Big\} \quad (13)$$

$$= \mathsf{E}\Big\{n_1(t_1 + B_0) - n_1(t_1)\Big|q_1(t_1) > KB_0 + \max(1, \frac{\lambda_2}{\lambda_1})\Big\} - \frac{B_0\max(\gamma,1)(1+2\epsilon)}{\lambda_1} \quad (14)$$

$$\leq \frac{B_0\max(\gamma,1)(1+\epsilon)}{\lambda_1} - \frac{B_0\max(\gamma,1)(1+2\epsilon)}{\lambda_1} = -\frac{B_0\max(\gamma,1)\epsilon}{\lambda_1} < 0, \quad (15)$$

where (13) follows from $q_1^+(t_1) > 0$ implies $q_1(t_1) > 0$ and (9); (14) follows from the definition of $q_1(t)$, and the first inequality of (15) follows from Lemma 2. Eq. (15) thus shows that $q_1(t)$ has a negative drift[3] whenever it is larger than $\approx \frac{2B_0}{\lambda_1}$. As a result, $q_1(t)$ is unlikely to take a large value. More precisely, we can show that for any $\epsilon_1, \epsilon_2 > 0$, there exists a $t_0 > 0$ such that

$$\forall t > t_0, \quad \mathsf{P}\left(q_1(t) < \epsilon_1 t\right) = \mathsf{P}\left(q_1^+(t) < \epsilon_1 t\right) > 1 - \epsilon_2. \quad (16)$$

The above has shown that $q_1(t)$ cannot be very large. We next show that $n_1(t)$ cannot be much larger than $\frac{\gamma t}{\lambda_1}$ either. Specifically, the following inequality holds for any $t > t_0$,

$$\mathsf{E}\{n_1(t)\} = \mathsf{E}\left\{\frac{\max(\gamma,1)t(1+2\epsilon)}{\lambda_1} + q_1(t)\right\} = \mathsf{E}\left\{\frac{\max(\gamma,1)t(1+2\epsilon)}{\lambda_1} + q_1(t)\Big|q_1(t) < \epsilon_1 t\right\}\mathsf{P}(q_1(t) < \epsilon_1 t) + \mathsf{E}\{n_1(t)|q_1(t) \geq \epsilon_1 t\}\mathsf{P}(q_1(t) \geq \epsilon_1 t)$$

$$\leq \left(\frac{\max(\gamma,1)t(1+2\epsilon)}{\lambda_1} + \epsilon_1 t\right) + t\epsilon_2, \quad (17)$$

[3]Compared to existing results that often use functions of the queue lengths as the Lyapunov function, this work focuses on proving the negative drift of a new quantity $q_1(t)$, i.e., the *packet index advancement* of the system.

where in the last step, we have used the fact that $n_1(t)$ is always upper bounded by t regardless of whether $q_1(t) \geq \epsilon_1 t$ or not. Note that we can choose arbitrarily small ϵ, ϵ_1, and ϵ_2 and (17) still holds for sufficiently large t. As a result, (17) shows that the expectation $\mathsf{E}\{n_1(t)\}$ is upper bounded by $\frac{\max(\gamma,1)t}{\lambda_1} + o(t)$. Similarly, we can prove $\mathsf{E}\{n_2(t)\} \leq \frac{\max(\gamma,1)t}{\lambda_2} + o(t)$. ∎

Thus far, we have assumed that the file sizes are infinite so that we can continue executing Lines 2 to 22 of SCHEDULE-PACKET-TRANSMISSION without worrying about the degenerate cases of executing Line 24. Under this infinite-file-size assumption, define $T_j^{\text{success}}(t)$ as the number of packets that have arrived/been decoded at destination j before expiration during time 1 to t. We then have the following lemma.

Lemma 4. *We have* $\mathsf{E}\{T_j^{success}(t)\} \geq \frac{\gamma t}{\lambda_j} - o(t)$.

In the following, we provide a detailed proof for the critically- and over-provisioned case ($\gamma \geq 1$) and some high-level discussion for the under-provisioned case $\gamma < 1$ as well.

Proof: We first consider the critically- and over-provisioned case ($\gamma \geq 1$). We define $T_j(t)$ as the number of time slots when the BS transmits an uncoded packet for session j up to time t (those time slots when Lines 18 or 20 of SCHEDULE-PACKET-TRANSMISSION are executed, with the assumption that user 1 is the leading user). Since user 2 is not the leading user, the BS transmits every session-2 packet uncodedly until it has been received by at least one user. We thus have

$$\mathsf{E}\{T_2(t)\} \leq \mathsf{E}\{n_2(t)\}\frac{1}{p_1+p_2-p_1p_2}, \tag{18}$$

where the inequality is because some uncoded packets are expired before they can be received by any user, and hence the expected transmission time for each packet is no larger than the case when there is no expiration. Next, we consider $T_1(t)$. Note that for session 1, some packets would be transmitted in an uncoded manner even after they have been received by user 2. (Recall Issue 2) $T_1(t)$ is thus comprised of two types of transmissions: The first type counts the number of time slots in which the BS transmits an uncoded packet of session 1 that has not been heard by any user. The second type counts the number of time slots in which the BS transmits a session-1 packet uncodedly even though that packet has been heard by user 2 already (due to its status being set to uncoded-Tx-only and in which case the BS continues to transmit this packet until user 1 receives it). The first part can be upper bounded by $\mathsf{E}\{n_1(t)\}\frac{1}{p_1+p_2-p_1p_2}$ in the same way as in (18). We use $\mathsf{UCO}(t)$ to denote the total number of the second type of uncoded-Tx-only transmission during the interval $[1,t]$. We then have

$$\mathsf{E}\Big\{\mathsf{UCO}(t)\Big\} \leq \mathsf{E}\{n_1(t)\}\Big(1-\frac{\lambda_1 p_1^2(1-p_2)}{\lambda_2 p_2^2(1-p_1)}\Big) \times \left(\frac{p_2(1-p_1)}{1-(1-p_1)(1-p_2)}\right)\frac{1}{p_1}. \tag{19}$$

The explanation of (19) is as follows. Per our proposed scheme, out of all $n_1(t)$ session-1 packets that have been transmitted during time interval $[1,t]$, a fraction $(1-\frac{\lambda_1 p_1^2(1-p_2)}{\lambda_2 p_2^2(1-p_1)})$ of them have their status set to uncoded-Tx-only. Out of those with status set to uncoded-Tx-only, a fraction of $\left(\frac{p_2(1-p_1)}{1-(1-p_1)(1-p_2)}\right)$ will be heard by d_2 first (strictly before it is heard by d_1). For those that have been heard by d_2 first, it takes, on average, additional $1/p_1$ time slots of transmission before it can be heard by the intended user d_1. The inequality is again to take into account that some packets may expire even before finishing its corresponding transmission. Combining the first and second parts, we obtain

$$\begin{aligned}\mathsf{E}\{T_1(t)\} &\leq \frac{\mathsf{E}\{n_1(t)\}}{p_1+p_2-p_1p_2} + \text{Eq.(19)} \\ &= \frac{\mathsf{E}\{n_1(t)\}}{p_1}\left(1-\frac{\lambda_1 p_1^2(1-p_2)}{\lambda_2 p_2(p_1+p_2-p_1p_2)}\right).\end{aligned} \tag{20}$$

Note that when we transmit an uncoded packet for session 1, the expected "reward" is p_1 since only user 1 can benefit from this transmission. When we transmit a coded packet, the expected reward for user 1 is p_1 and the expected reward for user 2 is p_2 since both destinations can benefit from the coded transmission. Note that by definition the total number of coded transmission in the $[1,t]$ interval is $t - T_1(t) - T_2(t)$. We can now lower bound the expected total rewards for user 1:

$$\begin{aligned}&\mathsf{E}\{T_1^{\text{success}}(t)\} = p_1\mathsf{E}\{T_1(t)\} + p_1\mathsf{E}\{t-T_1(t)-T_2(t)\} \\ &= p_1 t - p_1\mathsf{E}\{T_2(t)\} \\ &\geq p_1 t - p_1\frac{\gamma t}{\lambda_2}\frac{1}{p_1+p_2-p_1p_2} - o(t) \\ &= \frac{\gamma t}{\lambda_1} - o(t),\end{aligned} \tag{21}$$

where the inequality follows from $\mathsf{E}\{n_2(t)\} \leq \frac{\gamma t}{\lambda_2} + o(t)$ and (18); and (21) follows by plugging in the definition of γ and arithmetic simplification. Similarly, we can lower bound the expected total rewards for user 2

$$\begin{aligned}&\mathsf{E}\{T_2^{\text{success}}(t)\} = p_2\mathsf{E}\{T_2(t)\} + p_2\mathsf{E}\{t-T_1(t)-T_2(t)\} \\ &= p_2 t - p_2\mathsf{E}\{T_1(t)\} \\ &\geq p_2 t - p_2\frac{\gamma t}{p_1\lambda_1}\left(1-\frac{\lambda_1 p_1^2(1-p_2)}{\lambda_2 p_2(p_1+p_2-p_1p_2)}\right) - o(t) \\ &= \frac{\gamma t}{\lambda_2} - o(t),\end{aligned} \tag{22}$$

where (22) follows by plugging in the definition of γ and arithmetic simplification.

We next consider the under-provisioned case. Compared to the critically- and over-provisioned case, the main difference is that for the under-provisioned case, a new packet-dropping mechanism is used in Line 2 to Line 12 of SCHEDULE-PACKET-TRANSMISSION. Therefore, we need to carefully take that into account in our analysis. Use the same definition of $T_1(t)$ and $T_2(t)$ as in the previous proof, we can upper bound $\mathsf{E}\{T_2(t)\}$ as $\mathsf{E}\{T_2(t)\} \leq \mathsf{E}\{n_2(t)\}\frac{\gamma}{p_1+p_2-p_1p_2}$, where $\gamma < 1$ takes into account that out of $n_2(t)$ packets, $1-\gamma$ of them will be dropped and only γ portion of them will contribute to the uncoded transmission. Since user 1 is the leading user, $T_1(t)$ is still comprised of two parts. One part is when the BS transmits uncoded packets of session 1. The other part is when a session 1 packet has been received by user 2 first, and the

BS continues to transmit this packet until user 1 receives it. We can upper bound the first part and second part separately, and then upper bound $\mathsf{E}\{T_1(t)\}$. By the same argument as in the previous proof, expected total rewards for users 1 and 2 are lower bounded by

$$\mathsf{E}\{T_1^{\text{success}}(t)\} \geq \frac{t}{\lambda_1}\Big(\frac{1}{\frac{1/\lambda_1}{p_1}+\frac{1/\lambda_2}{p_1+p_2-p_1p_2}}\Big) - o(t), \quad (23)$$

and

$$\mathsf{E}\{T_2^{\text{success}}(t)\} \geq \frac{t}{\lambda_2}\Big(\frac{1}{\frac{1/\lambda_1}{p_1}+\frac{1/\lambda_2}{p_1+p_2-p_1p_2}}\Big) - o(t). \quad (24)$$

Since $\gamma = \frac{1}{\frac{1/\lambda_1}{p_1}+\frac{1/\lambda_2}{p_1+p_2-p_1p_2}}$ when user 1 is the leading user for the under-provisioned case, we have proven $\mathsf{E}\{T_j^{\text{success}}(t)\} \geq \frac{\gamma t}{\lambda_j} - o(t)$, for $j = 1, 2$. The details of the proof for the under-provisioned case can be found in [21]. ■

We are now ready to state the main result of the section.

Proposition 2. *Given system parameters p_1, p_2, λ_1, and λ_2, for any $\epsilon > 0$, there exists a sufficiently large N_1 (and $N_2 = \frac{\lambda_1 N_1}{\lambda_2}$) such that the proposed IDNC scheme achieves $\mathsf{E}\{N_1^{success}\}/N_1 \geq \min(\gamma, 1) - \epsilon$ and $\mathsf{E}\{N_2^{success}\}/N_2 \geq \min(\gamma, 1) - \epsilon$.*

The main idea of the proof is to use the previously proven Lemma 3 that $n_j(t) = \frac{\max(\gamma,1)t}{\lambda_j} + o(t)$ to prove that the majority of those delivered packets (all will have index $\leq n_j(t)$) are indeed within the first $\frac{\max(\gamma,1)t}{\lambda_j}$ packets. The above argument then shows that, for the infinite-file-size setting with $t \to \infty$, a close-to-one portion of the first $\frac{t}{\lambda_j}$ packets will be delivered/decoded before expiration. Thus, this result implies that even when focusing on the finite-file sizes N_1 and N_2, as long as N_1 and N_2 are sufficiently large, we can decode almost all N_1 and N_2 packets in $\lambda_1 N_1 = \lambda_2 N_2$ time slots. The detailed proof of Proposition 2 is provided in [21].

VII. Simulation Results for Finite File Sizes

We have rigorously analyzed the asymptotical optimality of proposed IDNC scheme for large file sizes $N_1 \to \infty$ and $N_2 \to \infty$, and empirically demonstrated its superior performance over G-B schemes for small file sizes. Next, we use simulation to verify its performance of the proposed IDNC scheme in a larger class of settings.

A. Performance for Large N_1 and N_2

We first assume that the successful delivery probabilities for user 1 and user 2 are $p_1 = 0.5$ and $p_2 = 0.6$, respectively. Then we consider the following 5 cases with (λ_1, λ_2) being (2,4), (3,4), (4,4), (5,4), and (6,4), respectively (we name them as Cases 1 to 5, respectively). By checking whether $(\frac{1}{\lambda_1}, \frac{1}{\lambda_2})$ satisfies (1) and (2), one can see that Case 1 is under-provisioned and Cases 2 to 5 are over-provisioned. For all cases we use $N_1 = 40000$. Recall that we require $\lambda_1 N_1 = \lambda_2 N_2$. We thus set N_2 to be 20000, 30000, 40000, 50000, and 60000 in the 5 cases.

We first show the capacity region without deadline constraints in Fig. 4, i.e., according to (1) and (2), as shown by the area beneath the two solid lines. We then use different markers to denote the time-wise throughput $(\frac{N_1^{\text{success}}}{\lambda_1 N_1}, \frac{N_2^{\text{success}}}{\lambda_2 N_2})$ from simulation for the 5 cases. The circles indicate the corresponding theoretical upper bound of both sessions, which are given by $\min(\gamma, 1)\big(\frac{1}{\lambda_1}, \frac{1}{\lambda_2}\big)$ in the discussion of (5).

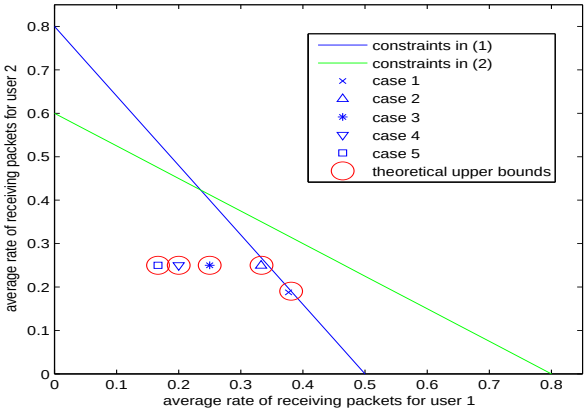


Fig. 4. Average rate of receiving packets for user 1 and user 2 when N_1 and N_2 are large.

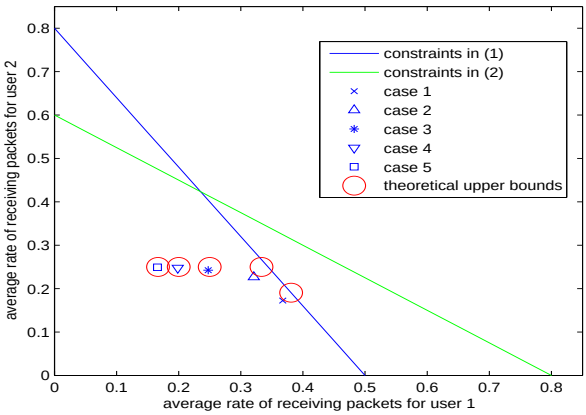


Fig. 5. Average throughput for users 1 and 2 when N_1 and N_2 are small.

More specifically, Case 1 is the under-provisioned scenarios and the throughput is limited by the two lines rather than by the maximum throughput $(\frac{1}{\lambda_1}, \frac{1}{\lambda_2})$. Cases 2 to 5 are the over-provisioned scenarios for which the throughput is decided by the maximum throughput $(\frac{1}{\lambda_1}, \frac{1}{\lambda_2})$. We observe that in all cases, the achievable throughput coincides with the theoretic upper bound, as predicted by Proposition 2.

B. Performance for Small N_1 and N_2

We are also interested in the performance of the IDNC scheme in the finite regime (when N_1 and N_2 are small). In Fig. 5 we plot the normalized throughput for both users when N_1 and N_2 are small. We use the same parameters as in Section VII-A except with smaller file sizes (N_1, N_2) being (400, 200), (400, 300), (400, 400), (400, 500), and (400, 600). We can observe that, although the numbers of packets for both session 1 and session 2 are small, the achievable throughput

TABLE I
COMPARISON FOR IDNC SCHEMES WITH & WITHOUT KNOWN CHANNEL WHEN N_1 AND N_2 ARE SMALL. THE ACHIEVABLE PERCENTAGE IS COMPUTED BY (6).

	The achievable percentage with known p	The achievable percentage with estimated $\hat{p}$
Case 1	0.8714	0.8763
Case 2	0.9280	0.9322
Case 3	0.9799	0.9807
Case 4	0.9905	0.9896
Case 5	0.9944	0.9940

are still very close to the theoretical upper bound (even when we are considering the under-provisioned case 1).

To investigate the detailed behavior of the proposed IDNC solution, we repeat the Monte Carlo experiment for 1000 times and plot the cumulative distribution function of the throughput in Fig. 2 and 3 of small and median file sizes, respectively. We can see that the throughput concentrate highly on its mean (with the cdf being a sharp step function) even for small file sizes. Therefore, our expectation-based analysis can be viewed as a good, much more tractable proxy/substitute than the traditional outage-probability-based analysis.

C. Comparison of the G-B scheme with the IDNC scheme

In Figs. 2 and 3, we compare the performance between the G-B scheme and the IDNC scheme. We can easily see that in Fig. 2 the performance of the IDNC scheme approaches the upper bound. It is still true even for small file size in Fig. 3. The IDNC scheme dynamically arranges the operations of coded transmission, uncoded transmission, and drops packets in an "online" fashion, while the G-B scheme blindly stick to the pre-fixed order for these operations. Also, the cdf of the proposed IDNC scheme concentrates highly on the mean to a greater degree than the G-B scheme, which again demonstrates the superiority of the IDNC scheme in a practical scenario. Finally, the IDNC scheme takes less complexity in the encoding process, and consumes relatively smaller buffer size for storing the coding opportunities.

D. Extensions For Unknown Channel Parameters

Although our proof of asymptotic optimality assumes that the BS knows the channel parameters p_1 and p_2, we believe that IDNC schemes can also achieve good performance without knowing (p_1, p_2) a priori. Specifically, since the users would send an ACK at the end of each time slot, the BS can use this feature to "estimate" $(\hat{p}_1, \hat{p}_2)$ *on-the-fly* and plug them into the IDNC subroutines as a substitute for the actual (p_1, p_2). In the following, we use simulation to study the performance of this "adaptive" IDNC scheme that estimates (p_1, p_2) on the fly. We consider the same 5 cases as in previous discussion. For small N_1 and N_2 as in Section VII-B $(N_1, N_2 \leq 600)$, we summarize our finding in Table I. We find that, even for small file size, the performance with channel estimation is very close to the performance with known channel parameters (The variations can be attributed to the randomness of the Monte-Carlo simulation). When N_1 and N_2 are large, the channel estimate is even more accurate and the adaptive channel parameter estimation incurs almost zero penalty in our numerical evaluation. We thus omit the detailed comparison table. The above observation suggests that our IDNC scheme is robust and it approaches the optimal throughput even when the channel parameters are unknown *a priori*. The intuition behind is that our IDNC can operate even if the initial channel estimates are not very accurate. Then, as time evolves, the more accurate the channel estimates become, the better the delay-constrained throughput can be, which is indeed the case once we accumulate enough ACKs.

VIII. CONCLUSION AND DISCUSSION

In this work, we have studied inter-session network coding for sending two unicast sessions over an unreliable wireless channel with heterogeneous channel conditions and heterogeneous deadline constraints. We developed both a generation-based (G-B) scheme and an immediately-decodable network coding (IDNC) scheme that maximize the normalized throughput subject to hard deadline constraints. Both the proposed G-B and the newly designed IDNC scheme are proven to be asymptotically optimal (when the file size is large). On the other hand, the IDNC scheme has significantly less complexity and buffer requirements, and achieves close-to-optimal throughput even for small file sizes, an attribute not found in the G-B solutions.

APPENDIX A
PROOF OF LEMMA 2 FOR THE OVER PROVISIONED CASE

We first present a detailed proof of Lemma 2 for the over-provisioned case, that is, $\gamma \geq 1$. The following discussion is conditioned on the event that in the end of time t_1, we have $\mathcal{A}_{t_1} \triangleq \{t_2 < \lambda_1 n_1(t_1), t_2 < \lambda_2 n_2(t_1)\}$. Define

$$\Delta n_1 = \left\lfloor \frac{(t_2 - t_1)}{\left(\frac{1/\lambda_1}{p_1} + \frac{1/\lambda_2}{p_1+p_2-p_1p_2}\right)\lambda_1} \right\rfloor + 1, \tag{25}$$

$$\Delta n_2 = \left\lfloor \frac{(t_2 - t_1)}{\left(\frac{1/\lambda_1}{p_1} + \frac{1/\lambda_2}{p_1+p_2-p_1p_2}\right)\lambda_2} \right\rfloor + 1. \tag{26}$$

Note that by our definition, $\Delta n_1 \lambda_1 \approx \Delta n_2 \lambda_2$.

From the beginning of time $t_1 + 1$, let us temporarily suspend the "expiration mechanism" and use our proposed scheme to transmit packets while allowing the supposedly-expired packets to remain in the system. We first examine how long it takes before the register $n_1(t)$ evolves from its current value $n_1(t_1)$ to a different value $n_1(t_1) + \Delta n_1$, and the register $n_2(t)$ evolves from its current value $n_2(t_1)$ to a different value $n_2(t_1) + \Delta n_2$. More specifically, we use t_3 to denote the (random) time slot that is the first time slot $t \geq t_1$ such that both $n_1(t)$ is at least $n_1(t_1) + \Delta n_1$ and $n_2(t)$ is at least $n_2(t_1) + \Delta n_2$.

We would take 3 major steps to reach the conclusion for Lemma 2, which are summarized in the following three corollaries, respectively. Under the assumption that the expiration mechanism is suspended from t_1 and onward, we first prove that the random variable t_3 is no less than the given constant t_2 with high probability in Corollary 1. Based on Corollary 1, we will then show that the "growth" of $n_j(t)$ from time

t_1+1 to t_2 is upper bounded by $\approx \frac{(t_2-t_1)\gamma}{\lambda_j}$ in Corollary 2: Finally, in Corollary 3 we will take into account the hard deadline constraints and show that even with the hard deadline constraints, we still have the result that the "growth"of $n_j(t)$ from time t_1+1 to t_2 is upper bounded by $\approx \frac{(t_2-t_1)\gamma}{\lambda_j}$.

Without loss of generality, we assume that user 1 is the leading user, that is, $\frac{\lambda_1 p_1(p_1-p_1p_2)}{\lambda_2 p_2(p_2-p_1p_2)} < 1$. So $\frac{1}{\gamma} = \frac{1/\lambda_1}{p_1} + \frac{1/\lambda_2}{p_1+p_2-p_1p_2} \leq 1$ by (4).

Corollary 1. *Without considering hard deadline constraints, for any $\epsilon > 0$, $\delta > 0$, if $t_2 - t_1$ is sufficiently large, then*

$$\mathsf{P}\left((t_3-t_1) > (t_2-t_1)(1-\epsilon)\middle|\,\mathcal{A}_{t_1}\right) > 1-\delta. \tag{27}$$

Proof: We define UT_1 (which stands for "Uncoded Transmission") as the number of time slots in $[t_1+1, t_3]$ when the proposed scheme schedules an *uncoded* packet transmission for Session 1. Note that by our definitions, all those uncoded transmissions must be used to transmit $X_{1,n}$ for some $n \geq n_1(t_1)$. Similarly, we also define UT_2 as the number of time slots in $[t_1+1, t_3]$ when the proposed scheme schedules an uncoded packet transmission for Session 2 packets $X_{2,n}$ with the indices being $n \geq n_2(t_1)$. Define

$$H_{1,n} = \Big|\{t > t_1 : \text{in the beginning of time } t, \text{ the scheme schedules an uncoded transmission of } X_{1,n}\}\Big|. \tag{28}$$

Since we stop an uncoded transmission if any one of the destinations successfully receives it, we have

$$\mathsf{E}\{H_{1,n}|\mathcal{A}_{t_1}\} = \frac{1}{1-(1-p_1)(1-p_2)} = \frac{1}{p_1+p_2-p_1p_2} \tag{29}$$

for all $n \geq n_1(t_1)$. As a result, the total number of time slots to transmit the uncoded session-1 packets is $\mathsf{UT}_1 \geq \sum_{i=n_1(t_1)+1}^{n_1(t_1)+\Delta n_1-1} H_{1,i}$, where the inequality is because uncoded session 1 packets with indices being $n_1(t_1)$ or being larger than $n_1(t_1)+\Delta n_1 - 1$ may also be transmitted during $[t_1+1, t_3]$. Similarly, $\mathsf{UT}_2 \geq \sum_{i=n_2(t_1)+1}^{n_2(t_1)+\Delta n_2-1} H_{2,i}$.

Since each $H_{1,i}$ and $H_{2,j}$ are of i.i.d. (conditional) geometric distribution with expectation (29), for any $\epsilon_1, \delta_1 > 0$, we can choose a sufficiently large B_1 such that if $\Delta n_1 > B_1$ and $\Delta n_2 > B_1 \frac{\lambda_1}{\lambda_2}$, then

$$\begin{aligned}&\mathsf{P}\left(\mathsf{UT}_1+\mathsf{UT}_2 > (1-\epsilon_1)\frac{\Delta n_1+\Delta n_2-2}{p_1+p_2-p_1p_2}\middle|\,\mathcal{A}_{t_1}\right)\\ &\geq \mathsf{P}\left(\sum_{i=1}^{\Delta n_1+\Delta n_2-2} H_i > (1-\epsilon_1)\frac{\Delta n_1+\Delta n_2-2}{p_1+p_2-p_1p_2}\right) > 1-\delta_1,\end{aligned} \tag{30}$$

where $\{H_i\}$ are i.i.d. geometric random variables with expectation $\frac{1}{p_2+p_2-p_1p_2}$ and (30) follows from the weak law of large numbers. Let $O_{1,n}$ denote a Bernoulli random variable that is 1 if, when repeatedly sending $X_{1,n}$ uncodedly, it was d_2 that received $X_{1,n}$ first; $O_{1,n} = 0$, if d_1 and d_2 received $X_{1,n}$ simultaneously or d_1 received it first. Symmetrically, we define the Bernoulli random variable $O_{2,n}$ such that $O_{2,n}$ is 1 if, when repeatedly sending $X_{2,n}$ uncodedly, it was d_1 that received $X_{2,n}$ first; $O_{2,n} = 0$, if d_1 and d_2 received $X_{2,n}$ simultaneously or d_2 received it first. When $X_{1,n}$ has been received by user 2 first and not by user 1, the BS would decide whether or not to keep transmitting this packet in the uncoded fashion until it's received by user 1, or not. We define $\mathsf{FC}_{1,n}$ (which stands for "Flip a Coin") as a Bernoulli random variable to indicate the decision result. $\mathsf{FC}_{1,n} = 1$ if the BS decides to keep transmitting this packet uncodedly until it's received by user 1; $\mathsf{FC}_{1,n} = 0$ if not. By our algorithm, $\mathsf{FC}_{1,n} = 1$ with probability $1 - \frac{\lambda_1 p_1(p_1-p_1p_2)}{\lambda_2 p_2(p_2-p_1p_2)}$, $\mathsf{FC}_{1,n} = 0$ with probability $\frac{\lambda_1 p_1(p_1-p_1p_2)}{\lambda_2 p_2(p_2-p_1p_2)}$. To distinguish from the uncoded transmission and for the ease of proof, we name the uncoded retransmission of coding opportunity of user 1 as "Single Transmission", as the single transmission is meant for user 1 only. We define $\mathsf{ST}_{1,n}$ as

$$\mathsf{ST}_{1,n} \triangleq \Big|\{t > t_1 : \text{in time } t, \text{ coding opportunity for user 1 } X_{1,n} \text{ is transmitted until user 1 receives it.}\}\Big|. \tag{31}$$

Note that $\mathsf{ST}_{1,n}$ is with expectation $\frac{p_2-p_1p_2}{p_1+p_2-p_1p_2}\Big(1 - \frac{\lambda_1 p_1(p_1-p_1p_2)}{\lambda_2 p_2(p_2-p_1p_2)}\Big)\frac{1}{p_1}$ for any $n \geq n_1(t_1)$ (recall that we have temporarily suspended "expiration"). By the weak law of large numbers, we also have for any $\delta_4 > 0$, $\epsilon_4 > 0$, there exists a B_4 such that if $\Delta n_1 > B_4$, we have

$$\begin{aligned}&\mathsf{P}\Big(\sum_{i=n_1(t_1)+1}^{n_1(t_1)+\Delta n_1-1} \mathsf{ST}_{1,i} \leq (\Delta n_1-1)\frac{p_2-p_1p_2}{p_1+p_2-p_1p_2}\\ &\quad\times\Big(1-\frac{\lambda_1 p_1(p_1-p_1p_2)}{\lambda_2 p_2(p_2-p_1p_2)}\Big)\frac{1}{p_1}(1-\epsilon_4)\Big|\mathcal{A}_{t_1}\Big) \leq \delta_4.\end{aligned} \tag{32}$$

We now define $\mathsf{CT}_{1,n}$ as follows:

$$\mathsf{CT}_{1,n} \triangleq \Big|\{t > t_1 : \text{in time } t, \text{ packet } X_{1,n} \text{ is mixed (coded) with some other } X_{2,n'} \text{ packets.}\}\Big|, \tag{33}$$

where $\mathsf{CT}_{1,n}$ stands for the coded transmission for packet $X_{1,n}$. Note that for any given n, the packets $X_{1,n}$ may be sent in a coded form for several (not necessarily adjacent) time slots and each time the companying $X_{2,n'}$ may be different, i.e., different n'. By similar analysis, and the fact that Δn_1 and Δn_2 can be converted to each other by (25) and (26), we can show that for any $\delta_6 > 0$, there exists a B_6 such that if $\Delta n_1 > B_6$, we have

$$\begin{aligned}&\mathsf{P}\Bigg(\sum_{i=n_1(t_1)+1}^{n_1(t_1)+\Delta n_1-1} \mathsf{CT}_{1,i} \leq (\Delta n_2-1)\Big(\frac{p_1-p_1p_2}{(p_1+p_2-p_1p_2)p_2}\Big)\\ &\quad\times(1-\epsilon_5)\Big|\mathcal{A}_{t_1}\Bigg) \leq \delta_6.\end{aligned} \tag{34}$$

We also have for any $\delta_7 > 0$, there exists a B_7 such that if $\Delta n_2 > B_7$, we have

$$\begin{aligned}&\mathsf{P}\Bigg(\sum_{i=n_2(t_1)+1}^{n_2(t_1)+\Delta n_2-1} \mathsf{CT}_{2,i} \leq (\Delta n_2-1)\Big(\frac{p_1-p_1p_2}{(p_1+p_2-p_1p_2)p_2}\Big)\\ &\quad(1-\epsilon_5)\Big|\mathcal{A}_{t_1}\Bigg) \leq \delta_7.\end{aligned} \tag{35}$$

Define TCT as the total number of coded transmission in time $[t_1+1, t_3]$. We then notice the following facts: (i) In the beginning of time t_3, the scheme must either transmit an uncoded packet $X_{1,n_1(t_1)+\Delta n_1-1}$, or transmit an uncoded packet $X_{2,n_2(t_1)+\Delta n_2-1}$ and it is received by one of the destinations (that is why $n_1(t)$ changes to $n_1(t_1)+\Delta n_1$, or $n_2(t)$ changes to $n_2(t_1)+\Delta n_2$). (ii) Therefore, at the end of time t_3-1, we must have $\min(|L_{10}|, |L_{01}|) = 0$. That is, there are no packets to be coded at the end of time t_3-1. (iii) Therefore, at the end of time t_3-1, either (a) there is no $\{X_{1,n} : n \in (n_1(t_1), n_1(t_1)+\Delta n_1 - 1]\}$ in L_{01}, or (b) there is no $\{X_{2,n} : n \in (n_2(t_1), n_2(t_1)+\Delta n_2 - 1]\}$ in L_{10}. From the above three facts, we have either all time instants of sending $X_{1,n}$ codedly are in $[t_1, t_3]$ or all time instants of sending $X_{2,n}$ codedly are in $[t_1, t_3]$. As a result, we have

$$\mathsf{TCT} \geq \min\Big(\sum_{i=1}^{n_1(t_1)+\Delta n_1-1} \mathsf{CT}_{1,i}, \sum_{i=1}^{n_2(t_1)+\Delta n_2-1} \mathsf{CT}_{2,i}\Big). \tag{36}$$

Both (34) and (35) can be made arbitrarily close to zero by choosing sufficiently large B_6 (Δn_1 is sufficiently large so that Δn_2 is large enough) and B_7. Jointly by setting $B_5 = \max(B_6\frac{\lambda_1}{\lambda_2}, B_7)$ and the union-bound arguments, we could prove that: for any $\epsilon_5, \delta_5 > 0$, we can choose a sufficiently large B_5 such that if $\Delta n_2 > B_5$, we have

$$\mathsf{P}\left(\mathsf{TCT} > (\Delta n_2 - 1)\left(\frac{p_1 - p_1p_2}{(p_1+p_2-p_1p_2)p_2}\right)(1-\epsilon_5)\middle|\mathcal{A}_{t_1}\right) > 1-\delta_5. \tag{37}$$

To summarize what we have proven thus far, we define

$$\text{Term-1} \triangleq \frac{\Delta n_1 + \Delta n_2 - 2}{p_1+p_2-p_1p_2}, \tag{38}$$

$$\text{Term-2} \triangleq (\Delta n_1 - 1)\left(\frac{p_2-p_1p_2}{p_1+p_2-p_1p_2}\cdot\Big(1-\frac{\lambda_1 p_1(p_1-p_1p_2)}{\lambda_2 p_2(p_2-p_1p_2)}\Big)\frac{1}{p_1}\right), \tag{39}$$

$$\text{Term-3} \triangleq \frac{(\Delta n_2 - 1)(p_1-p_1p_2)}{(p_1+p_2-p_1p_2)p_2}. \tag{40}$$

Our previous analysis (30), (32), and (37) proves that the following three inequalities hold with close-to-one probability: (i) $\mathsf{UT}_1+\mathsf{UT}_2 \geq (1-\epsilon_1)$Term-1, (ii) $\sum_{i=n_1(t_1)+1}^{n_1(t_1)+\Delta n_1-1}\mathsf{ST}_{1,i} \geq (1-\epsilon_4)$Term-2, and (iii) $\mathsf{TCT} \geq (1-\epsilon_5)$Term-3. Further, we can prove by simple arithmetic operations that

$$\text{Term-1} + \text{Term-2} + \text{Term-3} \geq (t_2-t_1) - o(t_2-t_1), \tag{41}$$

where the details could be found in our technical report [21].

Since for any time slot in $[t_1+1, t_3]$ we either send an uncoded or a coded transmission, we must have $t_3 - t_1 = \mathsf{UT}_1 + \mathsf{UT}_2 + \sum_{i=n_1(t_1)+1}^{n_1(t_1)+\Delta n_1-1}\mathsf{ST}_{1,i} + \mathsf{TCT}$. As as result, we have proven that for any $\epsilon_8, \delta_8 > 0$, there exists a $B_8 > 0$ such that if $t_2 - t_1 > B_8$ (so that Δn_1 and Δn_2 are sufficiently large), we have

$$\mathsf{P}\left((t_3-t_1) > (t_2-t_1)(1-\epsilon_8)\middle|\mathcal{A}_{t_1}\right) > 1-\delta_8. \tag{42}$$

Namely, with close to one probability, the random time t_3, at the end of which $n_1(t)$ is at least $n_1(t_1)+\Delta n_1$ and $n_2(t)$ is at least $n_2(t_1)+\Delta n_2$ for the first time, is no less than $t_1+(t_2-t_1)(1-\epsilon_8)$. The proof for Corollary 1 is complete. ■

Corollary 2. *Without considering hard deadline constraints, for any $\epsilon > 0$, there exists a sufficiently large B such that if $t_2 - t_1 > B$, then*

$$\mathsf{E}\Big\{n_j(t_2) - n_j(t_1)|\mathcal{A}_{t_1}\Big\} \leq \frac{(t_2-t_1)\gamma(1+\epsilon)}{\lambda_j}. \tag{43}$$

Proof: By Corollary 1, for any $\epsilon_8 > 0$, with close-to-one probability we have $t_3 \geq t_1+(t_2-t_1)(1-\epsilon_8)$. By the definition of the random stopping time t_3, with close-to-one probability, one of the following two statements holds at the end of time $t^* \triangleq t_1+(t_2-t_1)(1-\epsilon_8)$: (i) $n_1(t^*) < n_1(t_1)+\Delta n_1$, or (ii) $n_2(t^*) < n_2(t_1)+\Delta n_2$. Therefore,

$$\mathsf{P}\big(n_2(t^*) < n_2(t_1)+\Delta n_2 \quad \text{or} \quad n_1(t^*) < n_1(t_1)+\Delta n_1\big|\mathcal{A}_{t_1}\big) \geq 1-\delta_8. \tag{44}$$

By Lemma 1, both the distances $|\lambda_2 n_2(t_1) - \lambda_1 n_1(t_1)|$ and $|\lambda_2 n_2(t^*) - \lambda_1 n_1(t^*)|$ are upper bounded by $\max(\lambda_1, \lambda_2)$. We thus have

$$n_2(t^*) \leq n_2(t_1) + \Delta n_2 \tag{45}$$

$$\Rightarrow n_1(t^*) \leq n_1(t_1) + \Delta n_1 + 2\frac{\max(\lambda_1,\lambda_2)}{\lambda_1} \tag{46}$$

$$\Rightarrow n_1(t^*) \leq n_1(t_1) + \Delta n_1 + 2\frac{\lambda_2}{\lambda_1} + 2. \tag{47}$$

Combining (44) and (47) we have

$$\mathsf{P}\big(n_1(t^*) \leq n_1(t_1)+\Delta n_1 + 2\frac{\lambda_2}{\lambda_1} + 2\big|\mathcal{A}_{t_1}\big) > 1-\delta_8. \tag{48}$$

We then notice that for all $j \in \{1,2\}$, we must have $n_1(t_2) - n_1(t^*) \leq t_2 - t^*$. The reason is that for every time slot, the register $n_1(t)$ can increase at most by 1 in the over-provisioned scenario. Since the difference between t_2 and t^* is $(t_2-t_1)\epsilon_8$, (48) implies

$$\mathsf{P}\left(n_1(t_2) - n_1(t_1) \leq \Delta n_1 + 2\frac{\lambda_2}{\lambda_1} + 2 + (t_2-t_1)\epsilon_8\middle|\mathcal{A}_{t_1}\right) > 1-\delta_8. \tag{49}$$

Further, $n_1(t_2) - n_1(t_1) \leq t_2 - t_1$ since for each time slot the register $n_1(t)$ can increase by at most one. By (49), we can upper bound the expectation of $n_1(t_2) - n_1(t_1)$:

$$\mathsf{E}\Big\{n_1(t_2) - n_1(t_1)|\mathcal{A}_{t_1}\Big\} \leq \Big(\Delta n_1 + \frac{\lambda_2}{\lambda_1} + 2 + (t_2-t_1)\epsilon_8\Big) \times (1-\delta_8) + \delta_8(t_2-t_1). \tag{50}$$

By noticing that Δn_1 is linearly proportional to (t_2-t_1) while all other terms are sub-linear (with either a ϵ or a δ coefficient), (50) thus implies that for any $\epsilon > 0$, there exists a sufficiently large B such that if $t_2 - t_1 > B$, then

$$\mathsf{E}\Big\{n_1(t_2) - n_1(t_1)|\mathcal{A}_{t_1}\Big\} \leq \frac{(t_2-t_1)\gamma(1+\epsilon)}{\lambda_1}. \tag{51}$$

By similar argument, we have

$$\mathsf{E}\Big\{n_2(t_2) - n_2(t_1)|\mathcal{A}_{t_1}\Big\} \leq \frac{(t_2-t_1)\gamma(1+\epsilon)}{\lambda_2}. \tag{52}$$

■

Corollary 3. *After considering hard deadline constraints, for any $\epsilon > 0$, there exists a sufficiently large B such that if $t_2 - t_1 > B$, then*

$$\mathsf{E}\Big\{n_j(t_2) - n_j(t_1)|\mathcal{A}_{t_1}\Big\} \leq \frac{(t_2-t_1)\gamma(1+\epsilon)}{\lambda_j}. \qquad (53)$$

The proof of Corollary 3 is the most involved and the details can be found in [21] which uses the techniques of *shadow random variables*. The intuition of Corollary 3 is that any packets injected into the system will have index larger than $n_j(t_1)$. The event $\mathcal{A}_{t_1}$ being true further ensures that all those packets will not expire before t_2. As a result, when focusing on the regime between $[t_1, t_2]$, the difference between the systems with and without delay constraints is negligible, which allows us to use the results of Corollary 2 to prove Corollary 3.

Then the proof for the over-provisioned case of Lemma 2 is complete.

Due to space constraints, the proof for the under-provisioned case of lemma 2 can be found in our technical report [21].

REFERENCES

[1] S. Li, R. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inform. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.

[2] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inform. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[3] P. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. of Allerton Conference*, 2003.

[4] X. Li, C.-C. Wang, and X. Lin, "On the capacity of immediately-decodable coding schemes for wireless stored-video broadcast with hard deadline constraints," *IEEE Journal on Selected Areas in Communications, Issue on Trading Rate for Delay at the Application and Transport Layers*, vol. 29, no. 5, pp. 1094–1105, May 2011.

[5] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: Practical wireless network coding," in *Proc. of ACM SIGCOMM*, 2006.

[6] A. Eryilmaz and D. Lun, "Control for inter-session network coding," in *NetCod*, 2007.

[7] D. L. D. Traskov, N. Ratnakar, R. Koetter, and M. Médard, "Network coding for multiple unicasts: An approach based on linear optimization," in *Proc. of ISIT*, 2006.

[8] T. Ho, Y. Chang, and K. Han, "On constructive network coding for multiple unicasts," in *Allerton Conference*, 2006.

[9] W.-C. Kuo and C.-C. Wang, "On the capacity of 2-user 1-hop relay erasure networksthe union of feedback, scheduling, opportunistic routing, and network coding," in *ISIT*, 2011.

[10] L. Georgiadis and L. Tassiulas, "Broadcast erasure channel with feedback – capacity and algorithms," in *NetCod*, 2009.

[11] A. Eryilmaz, A. Ozdaglar, and M. Médard, "On delay performance gains from network coding," in *Proc. of CISS*, 2006.

[12] J.-K. Sundararajan, P. Sadeghi, and M. Médard, "A feedback-based adaptive broadcast coding scheme for reducing in-order delivery delay," in *NetCod*, 2009.

[13] W. Yeow, A. Hoang, and C. Tham, "Minimizing delay for multicast-streaming in wireless networks with network coding," in *Proc. of INFOCOM*, 2009.

[14] E. Drinea, C. Fragouli, and L. Keller, "Delay with network coding and feedback," in *Proc. of ISIT*, 2009.

[15] P. Chaporkar and A. Proutiere, "Adaptive network coding and scheduling for maximizing throughput in wireless networks," in *Proc. of ACM MobiCom*, 2007.

[16] D. Nguyen, T. Nguyen, and B. Bose, "Wireless broadcast using network coding," in *NetCod*, 2007.

[17] I. Hou and P. Kumar, "Admission control and scheduling for qos guarantees for variable-bit-rate applications on wireless channels," in *Proc. ACM Intl Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc). New Orleans, USA, May 2009, pp. 175184.*

[18] H. Seferoglu, L. Keller, B. Cici, A. Le, and A. Markopoulou, "Cooperative video streaming on smartphone," in *Proc. 49th Annual Allerton Conf. on Comm., Contr., and Computing. Monticello, IL, USA, Oct.2011.*

[19] L. Keller, A. Le, B. Cici, H. Seferoglu, C. Fragouli, and A. Markopoulou, "Microcast: cooperative video streaming on smartphones," in *Proc. 10th Intl Conf. on Mobile Sys., App., and Services (MobiSys). Low Wood Bay, UK, Jun. 2012.*

[20] "Contentaware instantly decodable network coding over wireless networks," in *Proc. Intl Conf. on comp. networking, and commun. (ICNC). Anaheim, USA, Feb. 2015.*

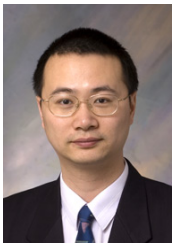
[21] X. Li, C.-C. Wang, and X. Lin, "Inter-session network coding schemes for two unicast sessions with sequential hard deadline constraints," *Technical Report, Purdue University, http://docs.lib.purdue.edu/ecetr/432/*, 2012.

[22] J. Barros, R. Costa, D. Munaretto, and J. Widmer, "Effective delay control in online network coding," in *Proc. of INFOCOM*, 2009.

Xiaohang Li received the B.E. degree in Electronic Engineering and Information Science from University of Science and Technology of China, Hefei, China, in 2007. He received Ph.D. degree in Electrical and Computer Engineering at Purdue University, West Lafayette, IN, in 2013. His research interests include network coding, wireless network scheduling, internet of things, etc.

Chih-Chun Wang is currently an Associate Professor of the School of Electrical and Computer Engineering of Purdue University. He received the B.E. degree in E.E. from National Taiwan University, Taipei, Taiwan in 1999, the M.S. degree in E.E., the Ph.D. degree in E.E. from Princeton University in 2002 and 2005, respectively. He worked in Comtrend Corporation, Taipei, Taiwan, as a design engineer in 2000 and spent the summer of 2004 with Flarion Technologies, New Jersey. In 2005, he held a post-doctoral researcher position in the Department of Electrical Engineering of Princeton University. He joined Purdue University as an Assistant Professor in 2006, and became an Associate Professor in 2012. He is currently a senior member of IEEE and has been an associate editor of IEEE Transactions on Information Theory since 2014 and the technical co-chair of the 2017 IEEE Information Theory Workshop. His current research interests are in the delay-constrained information theory and network coding. Other research interests of his fall in the general areas of networking, optimal control, information theory, detection theory, and coding theory.

Dr. Wang received the National Science Foundation Faculty Early Career Development (CAREER) Award in 2009.



Xiaojun Lin (S'02 / M'05 / SM'12) received his B.S. from Zhongshan University, Guangzhou, China, in 1994, and his M.S. and Ph.D. degrees from Purdue University, West Lafayette, Indiana, in 2000 and 2005, respectively. He is currently an Associate Professor of Electrical and Computer Engineering at Purdue University.

Dr. Lin's research interests are in the analysis, control and optimization of wireless and wireline communication networks. He received the IEEE INFOCOM 2008 best paper award and 2005 best paper of the year award from *Journal of Communications and Networks*. His paper was also one of two runner-up papers for the best-paper award at IEEE INFOCOM 2005. He received the NSF CAREER award in 2007. He is currently serving as an Associate Editor for IEEE/ACM Transactions on Networking and an Area Editor for (Elsevier) Computer Networks journal, and has served as a Guest Editor for (Elsevier) Ad Hoc Networks journal.