A New Capacity-Approaching Scheme for General 1-to-K Broadcast Packet Erasure Channels with ACK/NACK

Chih-Hua Chang and Chih-Chun Wang
School of Electrical and Computer Engineering, Purdue University, U.S.A.

***Abstract*—The capacity region of 1-to-K broadcast packet erasure channels with ACK/NACK is well known for some scenarios, e.g., $K \leq 3$, etc. However, existing achievability schemes either require knowing the target rate $\vec{R}$ in advance, and/or have a complicated description of the achievable rate region that is difficult to prove whether it matches the capacity or not. This work proposes a new network coding scheme with the following features: (i) its achievable rate region is identical to the capacity region for *all* the scenarios in which the capacity is known; (ii) its achievable rate region is much more tractable and has been used to derive new capacity rate vectors; (iii) it employs *sequential encoding* that naturally handles dynamic packet arrivals; (iv) it automatically adapts to unknown packet arrival rates $\vec{R}$; (v) it is based on $\mathsf{GF}(q)$ with $q \geq K$. In addition to analytically characterizing the achievable rate region of the proposed scheme, numerical simulation has been used to verify its queue length and delay performance.**

***Index Terms*—Broadcast capacity, stability region, packet erasure channel, network coding.**

I. Introduction

Broadcast channels (BC) have been extensively studied as one of the earliest subjects in network information theory [1]. Although the capacity for the most general broadcast channel model is still unknown, the exact capacity region is known for cases like the degraded broadcast channel models [2], degraded broadcast channels with message side information [3], etc. Recently, BC with causal channel output feedback is also considered along the lines of degree-of-freedom analysis [4] and the ACKnowledgement-feedback for broadcast packet erasure channels (PEC) [5]–[10]. Although the latter is based on a packet-based setting, which is quite different from the symbol-based studies of most physical layer solutions, the PEC setting is widely used to model potential packet loss in modern wireless communication networks [5], [11]–[18]. In addition, the insights derived from the PEC setting could also benefit the symbol-based settings, see the recent results in [19]–[21].

Specifically, the 1-to-K broadcast PEC[1] is a memoryless and stationary broadcast channel for which a single packet is transmitted for each time slot and may be received by a random subset of K destinations. For each k two possible outcomes may happen at destination d_k: The destination may receive the transmitted packet successfully without error, or it may receive "nothing" due to packet erasure. After transmission, all K destinations inform the source whether the packet is received or not, i.e., sending ACK if successful reception and NACK if erasure. Due to the broadcast nature, even if a packet is intended for one destination, other destinations may overhear the packet, which can later be used as side information.

Network coding is an important component when characterizing the capacity of the broadcast PEC model with causal ACK/NACK. For example, consider a transmitter s with two packets X and Y that need to be delivered to receivers d_1 and d_2, respectively. If during previous transmissions d_1 overheard Y and d_2 overheard X, then s can send $[X+Y]$ that benefits both d_1 and d_2 simultaneously. This simple idea has been generalized for the 1-to-K broadcast PEC with causal ACK/NACK. The main challenge of designing a capacity-approaching network coding scheme for general 1-to-K broadcast PECs lies in how to encode packets based on the latest overheard *side information* at different destinations to ensure that each destination is able to decode its desired packets within the shortest amount of time.

The 1-to-K broadcast PEC with ACK/NACK has been studied in many scenarios. The capacity region of the memoryless 1-to-2 broadcast PEC was characterized in [6], which shows that ACK/NACK feedback strictly improves the capacity of 1-to-2 broadcast PEC. [24] extends the results of 1-to-2 broadcast PEC to characterize the *stability region* for stochastic sequential arrival sources. Optimal scheduling of 1-to-2 broadcast PECs with sequential packet arrivals and time-varying channels is proposed in [14], which shows that any sequential arrival rates within the Shannon capacity region can be stabilized by the proposed joint network coding and network scheduler. The broadcast PEC with ACK/NACK and arbitrary prior message side information at destinations is investigated in [25]. Most achievability results in the literature are based on linear network coding operations, though recent results by [26] also explore typicality-based designs.

This work introduces the new concept of *opportunistic*

This work was supported in part by NSF grants ECCS-1407603, CCF-1422997, and CCF-1618475. Part of the results were presented in the 2017 IEEE International Symposium on Information Theory (ISIT).

[1]In this work, we use the information-theoretic terminology of broadcast channels. That is, there are K destinations and each of them desires one of the K independent streams of packets. In the networking terminology, such a setting is named differently as the *K-unicast setting*. The broadcast setting herein (i.e., the K-unicast setting) is sharply different from the so-called 1-to-K multicast setting in the networking literature, where all K destinations desire the same common stream of packets. Some results of the 1-to-K multicast setting can be found in [22], [23].

packet evolution[2], which is used to study two separate but closely related subjects of the general 1-to-K broadcast PEC with ACK/NACK.

Subject 1: Deriving a new inner bound that sheds further insights on the capacity region. On this front, we derive a new inner bound that is provably tight for all the scenarios in which the capacity regions are known.[3] That is, in terms of analytically characterizing the capacity region, the new inner bound is as good as any existing inner bounds [5], [17], [18].

In terms of the proof/construction techniques, two main ingredients of the existing inner bounds [5], [17], [18] are the concepts of *user ordering* and *collapsed overhearing set matching*.[4] The new concept of opportunistic packet evolution completely subsumes these two concepts, and the resulting new inner bound admits a much simpler form, which allows for more efficient numerical evaluation of the achievable rates and has been used to analytically identify new capacity rate vectors, a difficult task with the existing inner bounds.

Subject 2: We depart from the traditional block code setting and design a new sequential coding scheme that has short delay and small memory usage, automatically adapts to dynamic packet arrivals, and attains the aforementioned new inner bound. Specifically, most achievability schemes, e.g., [5], [17], have the following drawbacks. *Drawback 1:* To transmit packets at a given rate vector $\vec{R}$, the schemes have to first solve a linear-programming (LP) problem with $\vec{R}$ as input and then use the optimal LP solution to adjust the design parameters for the target $\vec{R}$. However, in practice the packet arrival rates $\vec{R}$ are usually not known in advance.

Drawback 2: The network coding opportunity exists *after* some node d_k overhears some previous transmissions. Therefore, there is a strict causality relationship between when the coding opportunity is created (through overhearing) and when one can start to combine packets to capitalize the created coding opportunity. In order to maximize the achievable rates, the existing schemes [5], [17], [26] impose a strict *user/transmission order* that takes into account the causality constraints. Since a strict transmission order requires packets to "wait in the queues" before transmission, it incurs long queuing delay in practice.

Drawback 3: Intuitively, network coding transmits a linear sum of multiple packets for which the overhearing set of each constituent packet is "matched" with the other constituent packets. Since there are $2^K - 1$ non-empty subsets of all K destinations, there are $2^K - 1$ number of ways to "match" the overhearing sets. For each time instant, one of these $2^K - 1$ ways is used to generate the coded packet. Unfortunately, it turns out that strictly adhering to this simple rule yields strictly suboptimal performance. To remedy the deficiency, the concept of *collapsed overhearing set matching* (COSM) is introduced in [5], [17], [18] and has since been one of the central ingredients used to achieve the capacity region of the case of $K = 3$ and to design high-performance inner bounds for general K. Nonetheless, searching for the COSM involves comparing roughly $O\left(\left(\frac{K+1}{\ln(K+2)}\right)^{K+1}\right)$ coding choices for each time instant, a dramatic increase from the intuitive number $2^K - 1$. The complexity of COSM is exceedingly high even for very small K.

Drawback 4: The theoretic schemes in [5], [17] allow for asymptotically large memory, asymptotically large queuing/decoding delay, and asymptotically large complexity, which make it difficult to implement in practice. Note that there are several existing schemes that aim to simplify the code design and make network coding practical, see [28] and the references therein, but at the cost of sacrificing the provable optimality.

Our new network code design addresses the above four drawbacks simultaneously. Firstly, our scheme requires only the knowledge of the underlying channel statistics, and will automatically adapt to the unknown arrival rates $\vec{R}$. Secondly, by using the new concept of opportunistic packet evolution, our scheme does not impose any user/transmission order and no longer needs to search for COSM. Instead, for each time instant the new scheme chooses one out of $2^K - 1$ possible coding choices but can still attain rates that are equal or better than all the existing schemes. Thirdly, the proposed scheme is based on sequential coding, which has short delay and low memory usage.

Comparison to the closest existing work [18]: The authors in [18] design a sequential coding scheme that addresses Drawbacks 1, 2, and 4 successfully. In addition, only binary XOR operations are used in [18] and the scheme admits the highly desired feature of *instantaneous decodability*. For comparison, our results are based on $\mathsf{GF}(q)$ instead of binary XOR. Our scheme is not instantly decodable, though numerical results show that the total delay, queueing plus decoding delay, is still quite manageable for practical scenarios.

The defining difference between our scheme and [18] is the new notion of opportunistic packet evolution, which enables our scheme to successfully address Drawback 3. In the following, we highlight the benefits of opportunistic packet evolution.

- Provable optimality: Our new scheme is provably capacity achieving for all the scenarios for which the capacity region is known, also see footnote 3. For comparison, the *inter-level* scheme in [18] is provably optimal only when the following conditions hold simultaneously: (i) $K = 4$ and (ii) the channel is symmetric and spatially independent. Our new inner bound can also be used to identify new capacity vectors, which is difficult to do with the result in [18] due to its high complexity.
- Inner bound description: Our achievable rate region (ARR) has a very compact form that consists of 2 inequalities. For comparison, the scheme in [18] reacts to various

[2]Part of the results were presented in [27] with a simplified model. This work extends the concept of opportunistic packet evolution to a general setting and provides the detailed proofs.

[3] The capacity region of the 1-to-K broadcast PECs is known if any of the following three conditions hold: (i) $K \leq 3$, (ii) the channel is *symmetric*, or (iii) the channel is *spatially independent* and the rate vectors are *one-sided fair*.

[4]The concepts of user ordering and collapsed overhearing set matching are powerful tools in terms of finding the largest achievable rate region. But they also have significant drawbacks both analytically, i.e., being too complicated to be tractable, and in practice, i.e., incurring too much delay and complexity. Also see the detailed discussion of Drawbacks 2 and 3 in Subject 2 for their practical implications.

TABLE I
COMPARISON OF THE NUMBERS OF CODING CHOICES. COSM STANDS FOR "COLLAPSED OVERHEARING SET MATCHING" AND B_m REPRESENTS THE m-TH BELL NUMBER. FOR REFERENCE, $B_6 = 203$, $B_{11} = 678570$, AND $B_{16} \approx 10^{10}$.

No. users	Our scheme	Low-complexity intra-level scheme in [18]: Without COSM is thus strictly suboptimal	Inter-level scheme in [18]: With COSM
1	1	1	1
2	3	5	8
3	7	25	47
4	15	124	244
5	31	616	1227
K	$2^K - 1$	$\left(\sum_{m=0}^{K}\binom{K}{m}B_{m+1}\right) - 2^{K+1} + K + 1$	No clean expression available.

sub-cases differently, which makes the corresponding ARR more difficult to describe. For example, for $K = 5$ and asymmetric channels, our ARR is described as an LP problem with 31 variables. The ARR of the general *inter-level scheme* in [18] is an LP problem with 1227 variables.

- Backpressure algorithms: Both this work and [18] are based on *backpressure scheduling* and can automatically adapt to any unknown $\vec{R}$. However, our backpressure scheduler has a significantly smaller number of "competing actions" to consider during each backpressure computation, which greatly reduces the complexity and delay inherent in the backpressure solution. Taking $K = 5$ for example, our scheme compares the backpressures of only 31 different actions while [18] compares the backpressures of 1227 actions. Also see Table I.

The outline of this paper is as follows. Section II presents the model of the 1-to-K broadcast PEC with ACK/NACK. Section III describes the new achievable rate region for the 1-to-K broadcast PECs and Section IV proposes the corresponding new scheme. Section V converts the theoretical achievability scheme to a practical header-based scheme with the cost of a small amount of control overhead. The added overhead is not needed during the theoretical discussion but would highly facilitate practical implementation. Section VI presents the simulation results. Section VII concludes this work.

II. PROBLEM SETTING

This work considers two major settings, the block-coding and the sequential-coding settings. The goal of the block-coding setting is that given a block of $n \cdot R_k$ messages how to transmit the messages with asymptotically small error probability within n channel usage. The goal of the sequential-coding setting is that given i.i.d. Poisson random arrival processes with arrival rates R_k, how to transmit the messages *error free* while keeping the queue lengths stable. The largest limit of the former setting is known as the *Shannon capacity region* and the limit of the latter is termed the *optimal stability region*. In the following, we formally define these two settings and prove that the stability region is an inner bound of the capacity region. Since this work focuses exclusively on the inner bound, in the sequel we focus on characterizing a new stability region.

A. The 1-to-K Broadcast Packet Erasure Channel

For any positive integer K, we use $[K] \triangleq \{1, 2, \dots, K\}$ to denote the set of integers from 1 to K and use the notation $2^{[K]} \triangleq \{S : S \subseteq [K]\}$ to denote the collection of all subsets of $[K]$. We consider a 1-to-K broadcast PEC from source s to destinations d_k, $k \in [K]$. There are K independent packet streams, one stream for each d_k. All packets are drawn independently and uniformly randomly from a fixed finite field $\mathsf{GF}(q)$ satisfying $q \geq K$. The scenario of $\mathsf{GF}(q)$ with $q < K$ is beyond the scope of this work.

At any discrete time slot $t \in \{1, 2, \cdots\}$, source s sends a packet $Y(t) \in \mathsf{GF}(q)$ as an input of the broadcast PEC. The channel outputs a K-dimensional vector $(Z_1(t), \cdots, Z_K(t))$, where for each k we have $Z_k(t) \in \{Y(t), *\}$. Herein $Z_k(t) = Y(t)$ means that the transmitted packet $Y(t)$ is received successfully by d_k and $Z_k(t) = *$ means that the transmitted $Y(t)$ is "erased" at d_k. We define $S_{\text{rx}}(t) \triangleq \{k : Z_k(t) = Y(t)\}$ as the set of indices k for which $Z_k(t) = Y(t)$, i.e., $S_{\text{rx}}(t)$ is the *reception set* at time t. The distribution of the random 1-to-K broadcast PEC is thus determined uniquely by the distribution of the reception set $S_{\text{rx}}(t)$.

We assume that the random process of $\{S_{\text{rx}}(t) : t \in \{1, 2, \cdots\}\}$ is *stationary* and *memoryless* and does not depend on the input $\{Y(t) : t\}$. The probability that $Y(t)$ is received by all d_k for $k \in S$ is then denoted by

$$p_S \triangleq \sum_{\tilde{S} \in 2^{[K]} : \tilde{S} \supseteq S} \Pr(S_{\text{rx}}(t) = \tilde{S}). \tag{1}$$

The above definition of p_S can be further generalized as follows: For any disjoint subsets $S, T \in 2^{[K]}$.

$$p_{S\overline{T}} \triangleq \sum_{\tilde{S} \in 2^{[K]} : \tilde{S} \supseteq S, \tilde{S} \cap T = \emptyset} \Pr(S_{\text{rx}}(t) = \tilde{S}). \tag{2}$$

That is, $p_{S\overline{T}}$ represents the probability that $Y(t)$ is received by all d_k for $k \in S$ but by none of the d_k for $k \in T$. The new input argument T excludes some events that were previously counted when computing p_S. Note that we do not care whether $Y(t)$ is received or not by those destinations in $[K] \backslash (S \cup T)$ thus the summation in (2). Comparing (1) and (2), it is clear that $p_S = p_{S\overline{\emptyset}} \geq p_{S\overline{T}}$ for all $S \cap T = \emptyset$.

For simplicity, when either S or T is empty, we use $p_{\overline{T}}$ and p_S as shorthand for $p_{\emptyset\overline{T}}$ and $p_{S\overline{\emptyset}}$ respectively. For the special instance of $T = \emptyset$ and $S = \{k\}$ containing only one element k, we use $p_k \triangleq p_{\{k\}\overline{\emptyset}}$ as shorthand, which is simply the marginal success probability for destination d_k.

For convenience, we also define

$$p_{\cup S} \triangleq 1 - p_{\overline{S}}.$$

which is the probability that *at least* one of the destination $k \in S$ receives the packet $Y(t)$.

We close this subsection by introducing a commonly used definition.

Definition 1. *A 1-to-K broadcast PEC is* spatially independent *if the distribution of $S_{\text{rx}}(t)$ satisfies*

$$p_{S\overline{[K]\backslash S}} = \left(\prod_{i\in S} p_i\right)\left(\prod_{j\in[K]\backslash S}(1-p_j)\right), \ \forall S \in 2^{[K]}. \quad (3)$$

The results of this work do not assume the underlying 1-to-K broadcast PEC being spatially independent, unless explicitly stated otherwise.

B. The Block-Coding Setting

Given any rate vector $\vec{R} \triangleq (R_1, \ldots, R_K)$, we now define the traditional block-coding setting with block length n. We denote $\mathbf{X}_k = \{X_{k,i} \in \mathsf{GF}(q) : i = 1, 2, \cdots, nR_k\}$ as the nR_k packets of stream k. For any time $t \in [n]$, source s sends a coded symbol

$$Y(t) = f_t^{(\text{bl})}(\mathbf{X}_1, \ldots, \mathbf{X}_K, [S_{\text{rx}}]_1^{t-1}) \in \mathsf{GF}(q)$$

where the encoding function $f_t^{(\text{bl})}(\cdot)$ takes all information symbols $\{\mathbf{X}_k\}$ and the past reception sets $[S_{\text{rx}}]_1^{t-1} \triangleq \{S_{\text{rx}}(\tau) : \tau \in [t-1]\}$ as input and generates the coded symbol $Y(t) \in \mathsf{GF}(q)$. The knowledge of $[S_{\text{rx}}]_1^{t-1}$ models the causal ACK/NACK fed back from the destinations to the source. The superscript (bl) emphasizes that it is the block-coding setting. In the end of time n, each d_k decodes

$$\hat{\mathbf{X}}_k = g_k^{(\text{bl})}([Z_k]_1^n, [S_{\text{rx}}]_1^n), \quad (4)$$

based on all the received packets $[Z_k]_1^n \triangleq \{Z_k(t) : t \in [n]\}$ and all the reception sets $[S_{\text{rx}}]_1^n$. Here we assume $[S_{\text{rx}}]_1^n$, the network-wide ACK/NACK information is known to all the destinations.[5]

A network block code of length n is defined by the corresponding n encoding functions $f_t^{(\text{bl})}(\cdot)$, $t \in [n]$, and K decoding functions $g_k^{(\text{bl})}(\cdot)$, $k \in [K]$. The achievable rates of a 1-to-K broadcast PEC with ACK/NACK are then defined by

Definition 2. *Given any fixed* $\mathsf{GF}(q)$*, $(R_1, \cdots, R_K)$ is achievable if for any $\epsilon > 0$, there exists a network block code of length n such that*

$$\Pr\left(\bigcup_{k\in[K]} \{\hat{\mathbf{X}}_k \neq \mathbf{X}_k\}\right) \leq \epsilon.$$

The capacity region is the closure of all achievable rate vectors $(R_1, \ldots, R_K)$.

The exact capacity region for general channel parameters $\{p_{S\overline{[K]\backslash S}} : \forall S \in 2^{[K]}\}$ remains an open problem. We close this subsection by restating the best known existing capacity outer bound results [5], [17].

Proposition 1 ([5], [17]). *A K-permutation (or simply permutation) is a bijective function from the set $[K]$ to itself. That is, $\pi : [K] \mapsto [K]$. Given any permutation π, for all $j \in [K]$ we define $S_j^\pi \triangleq \{\pi(l) : l \in [j]\}$ as the set of the first j elements according to π. Any achievable rate vector $(R_1, \ldots, R_K)$ must satisfy the following $K!$ inequalities:*

$$\sum_{j=1}^K \frac{R_{\pi(j)}}{p_{\cup S_j^\pi}} \leq 1, \quad \forall \pi. \quad (5)$$

For example, if $K = 3$, the above outer bound becomes:

$$\begin{cases} \frac{R_1}{p_1} + \frac{R_2}{p_{\cup\{1,2\}}} + \frac{R_3}{p_{\cup\{1,2,3\}}} \leq 1 \\ \frac{R_2}{p_2} + \frac{R_1}{p_{\cup\{1,2\}}} + \frac{R_3}{p_{\cup\{1,2,3\}}} \leq 1 \\ \frac{R_1}{p_1} + \frac{R_3}{p_{\cup\{1,3\}}} + \frac{R_2}{p_{\cup\{1,2,3\}}} \leq 1 \\ \frac{R_3}{p_3} + \frac{R_1}{p_{\cup\{1,3\}}} + \frac{R_2}{p_{\cup\{1,2,3\}}} \leq 1 \\ \frac{R_2}{p_2} + \frac{R_3}{p_{\cup\{2,3\}}} + \frac{R_1}{p_{\cup\{1,2,3\}}} \leq 1 \\ \frac{R_3}{p_3} + \frac{R_2}{p_{\cup\{2,3\}}} + \frac{R_1}{p_{\cup\{1,2,3\}}} \leq 1 \end{cases} \quad (6)$$

In [5], [17] it is proven that in the scenario of $K = 3$, this outer bound is indeed the capacity region.

C. The Sequential-Coding Setting

In this setting, the packets arrive sequentially for each time slot. We denote the number of user-k packets arriving at time slot t by $A_k(t)$ and assume $A_k(t)$ is an i.i.d. Poisson random process with mean $\mathsf{E}\{A_k(t)\} = R_k$. At time t, we denote $M_k(t) \triangleq \sum_{\tau=1}^t A_k(\tau)$ as the cumulative number of packet arrivals until time t and $\mathbf{X}_k(t) = \{X_{k,i} : i = 1, 2, \ldots, M_k(t)\}$ as the set of all user-k packets that have already arrived at source s by time t. Obviously, $\mathbf{X}_k(t)\backslash\mathbf{X}_k(t-1)$ represents the content of the user-k packets arriving during time t. We use $\vec{A}(t) = (A_1(t), \ldots, A_K(t))$ to denote the arrival patterns of all K destinations at time t.

Unlike the block-coding setting which is not worried about memory usage, any sequential network code has to carefully manage its memory. Specifically, the content of the memory at the *end* of time t is denoted by $Q(t) \in (\mathsf{GF}(q))^*$, a variable-length string of symbols in $\mathsf{GF}(q)$. In the networking terminology, $Q(t)$ represents the content of the "queue" at time t.

At the beginning of each time slot t, source s transmits a coded symbol $Y(t)$ using the broadcast PEC. That is,

$$Y(t) = f_t^{(\text{sq})}\left(Q(t-1), \bigcup_{k\in[K]} \mathbf{X}_k(t)\backslash\mathbf{X}_k(t-1), [S_{\text{rx}}]_1^{t-1}\right) \quad (7)$$

where $f_t^{(\text{sq})}(\cdot)$ is the sequential encoding function that takes as input $Q(t-1)$, the content of the memory at the end of time

[5]In practice, each d_k naturally knows whether itself has received the packet $Y(t)$ or not, but may not know the reception status of other $d_{\tilde{k}}$, $\tilde{k} \neq k$. Therefore, in practice, source s may need to *broadcast* the network-wide information $[S_{\text{rx}}]_1^n$ to each individual d_k using additional $\frac{n\cdot K}{\log_2(q)\min_k p_k}$ packets. For sufficiently large q, the rate penalty of broadcasting $[S_{\text{rx}}]_1^n$ is negligible and our model thus directly assumes that $[S_{\text{rx}}]_1^n$ is known to all d_k.

$t-1$, the new arrivals of user-k packets at time t for all $k \in [K]$, and the past reception status $[S_{\text{rx}}]_1^{t-1}$ obtained through causal ACK/NACK feedback. The superscript (sq) emphasizes we consider the setting of sequential coding.

After the transmission over the broadcast PEC, new ACK/NACK will be fed back to source s and in the end of time t the source updates its memory

$$Q(t) = f_{\text{buff},t}^{(\text{sq})}\left(Q(t-1), \bigcup_{k\in[K]} \mathbf{X}_k(t)\backslash\mathbf{X}_k(t-1), [S_{\text{rx}}]_1^t\right). \tag{8}$$

Comparing to (7), the buffer management function $f_{\text{buff},t}^{(\text{sq})}$ has an additional input argument $S_{\text{rx}}(t)$, the reception status of time t.

The definition of decodability of a sequential encoder/decoder pair needs special attention, and we define the sequential decoder $g_{k,t}^{(\text{sq})}$ as follows.

$$\hat{\mathbf{X}}_k(t) = g_{k,t}^{(\text{sq})}\left([Z_k]_1^t, Q(t), [S_{\text{rx}}]_1^t, [\vec{A}]_1^t\right) \tag{9}$$

and we require the decoder function $g_{k,t}^{(\text{sq})}$ to satisfy

$$\Pr\left(\hat{\mathbf{X}}_k(t) \neq \mathbf{X}_k(t)\right) = 0, \quad \forall k, t. \tag{10}$$

The rationale behind (9) and (10) is as follows. Consider the end of time t. It is likely that $\mathbf{X}_k(t)$ cannot be fully decoded by d_k at the end of the current time t. One reason is that some of them may have just arrived at s and are still waiting for their turn to be transmitted, the so-called *queueing delay*. Another reason is that they may be transmitted in a coded form and thus some *decoding delay* is needed before actual decoding.

That said, if at the end of time t a genie conveys to destination d_k all the (undelivered) entries/content in the current source queue $Q(t)$ and the past reception status $[S_{\text{rx}}]_1^t$, then d_k must be able to perfectly decode all $M_k(t)$ packets in $\mathbf{X}_k(t)$. Otherwise, some packets in $\mathbf{X}_k(t)$ are *permanently lost* and can no longer be decoded in the future since only the content in $Q(t)$ may have any impact on future encoding. Following this rationale, d_k, if equipped by a genie with the knowledge of all the received symbols $[Z_k]_1^t$, the queue content $Q(t)$, the past network-wide reception status $[S_{\text{rx}}]_1^t$, and the past arrival patterns $[\vec{A}]_1^t$, must be able to decode all packets in $\mathbf{X}_k(t)$ error-freely, which is captured by (9) and (10).

A sequential network coding scheme is thus described by the $K+2$ infinite series of the encoding functions $f_t^{(\text{sq})}$, the buffer management functions $f_{\text{buff},t}^{(\text{sq})}$, and the decoding functions $g_{k,t}^{(\text{sq})}$ in (7), (8), (9), and (10). We then define the stability region as follows.

Definition 3. *A rate $(R_1, \ldots, R_K)$ is stable if there exists a sequential network coding scheme such that*

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=1}^{t} \mathsf{E}\left\{|Q(\tau)|\right\} < \infty \tag{11}$$

where $|Q(\tau)|$ is the length of the queue $Q(\tau)$.

Remark: The above definition is compatible with the traditional stability region definition for non-coded solutions. Specifically, a traditional scheme has a very simple buffer management function $f_{\text{buff},t}^{(\text{sq})}$ in (8), that stores the new packets $\mathbf{X}_k(t)\backslash\mathbf{X}_k(t-1)$ in the queue while removing those packets that have been successfully delivered (those being ACKed). The transmission function $f_t^{(\text{sq})}$ of the traditional non-coded scheme then picks one packet in the queue $Q(t-1)$ or the newly arrived packets $\mathbf{X}_k(t)\backslash\mathbf{X}_k(t-1)$ and transmits the chosen packet uncodedly. Which packet to choose is decided by a "network scheduler", which takes the past reception status $[S_{\text{rx}}]_1^{t-1}$ as input. The decodability conditions (9) and (10) hold naturally for the traditional uncoded scheme since this simple memory management scheme ensures that $Q(t)$ contains all the packets that have not been delivered. The goal of the network scheduler is then to stabilize the queue length $|Q(t)|$ as in (11) while using simple uncoded $f_t^{(\text{sq})}$ and $f_{\text{buff},t}^{(\text{sq})}$. Our new definitions can be viewed as a generalization of the traditional ones by allowing both the transmission $Y(t)$ and the packets stored in the memory $Q(t)$ to be coded.

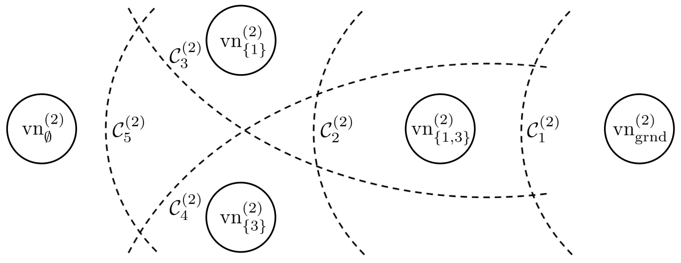
$\text{vn}_{\emptyset}^{(2)}$
$\text{vn}_{\{1\}}^{(2)}$
$\text{vn}_{\{3\}}^{(2)}$
$\text{vn}_{\{1,3\}}^{(2)}$
$\text{vn}_{\text{grnd}}^{(2)}$
$\mathcal{C}_1^{(2)}$
$\mathcal{C}_2^{(2)}$
$\mathcal{C}_3^{(2)}$
$\mathcal{C}_4^{(2)}$
$\mathcal{C}_5^{(2)}$

Fig. 1. The virtual nodes and proper-cuts for $K=3$ and $k=2$.

We close this subsection by establishing the relationship between the block-coding-based capacity region and the sequential-coding-based stability region.

Proposition 2. *If a rate vector $(R_1, \ldots, R_K)$ is stable, then it also belongs to the capacity region.*

The proof of Proposition 2 is provided in Appendix A.

Since the stability region is a subset of the capacity region, in the sequel we will describe a new stability region and the result naturally serves as a new inner bound of the capacity region. A summary of all the notations is provided in Table II for easier reference.

III. Main Results

A. A New Stability Region

To describe the new stability region, we first introduce the following definitions.

Definition 4. *For any fixed $k \in [K]$, we define 2^{K-1}* virtual nodes *and label each of them by a subset $S \subseteq [K]\backslash\{k\}$. It is thus convenient to simply denote each virtual node as $\text{vn}_S^{(k)}$ for all $S \in 2^{[K]\backslash\{k\}}$. We then define another virtual node, called the* ground node*, and denoted it by $\text{vn}_{\text{grnd}}^{(k)}$. Totally there are $2^{K-1}+1$ virtual nodes for a given k and these virtual nodes form a* virtual sub-network *of destination d_k. The K virtual sub-networks (totally $K(2^{K-1}+1)$ virtual nodes) jointly form the overall virtual network.*

TABLE II
SUMMARY OF NOTATIONS.

Symbol	Description
$Y(t)$	Transmitted packet at time t
$Z_k(t)$	Received signal $Z_k(t) \in \{Y(t), *\}$ of d_k at time t
$S_{\text{rx}}(t)$	Reception set $\{k \in [K] : Z(t) = Y(t)\}$ at time t
p_S	Probability that $Y(t)$ is received by all d_k for all $k \in S$
$p_{S\overline{T}}$	Probability that $Y(t)$ is received by all d_i for $i \in S$ but none of the d_j for $j \in T$
$p_{\cup S}$	Probability that at least one of d_k, $k \in S$, receives $Y(t)$
$\vec{p}$	Marginal success probability vector $(p_1, \ldots, p_K)$
$\vec{R}$	Rate vector $(R_1, \ldots, R_K)$
$\mathbf{X}_k$	The set of nR_k packets of stream k, i.e., $\{X_{k,i} \in \mathsf{GF}(q) : i = 1\ldots, nR_k\}$
$A_k(t)$	Number of packets for d_k arrived at time t
$\vec{A}(t)$	Arrival pattern $(A_1(t), \ldots, A_K(t))$ of all K destinations at time t
$M_k(t)$	Cumulative number of packet arrivals till time t, i.e., $\sum_{\tau=1}^{t} A_k(\tau)$
$X_{k,i}$	The i-th input packet for d_k
$\mathbf{X}_k(t)$	The set of packets for d_k already arrived by time t, i.e., $\{X_{k,i} : i = 1, \ldots, M_k(t)\}$
$Q(t)$	Content of the source memory at the end of time t
$\text{vn}_S^{(k)}$	Virtual node corresponding to set $S \in 2^{[K]\backslash\{k\}}$ in the k-th virtual sub-network
$\text{vn}_{\text{grnd}}^{(k)}$	Virtual ground node in the k-th virtual sub-network
$\mathcal{C}^{(k)}$	A (proper) cut of the k-th virtual sub-network
$Q_S^{(k)}$	Queue containing packets for d_k but overheard by d_i, $i \in S$
$\mathbf{U}^{(k)}(t)$ or $\mathbf{U}^{(k)}$	Matrix containing the coding vectors of queued packets (at the source) for d_k at the end of time t
$\mathbf{V}^{(k)}(t)$ or $\mathbf{V}^{(k)}$	Matrix containing the coding vectors of all the packets received by d_k at the end of time t
$W_{(k,i)}$	The coded packet corresponding to the original input packet $X_{k,i}$
$\mathsf{row}_{\mathsf{ptr}}(\mathbf{U}^{(k)})$	Row vector of $\mathbf{U}^{(k)}$ referred by the pointer ptr
$\vec{\mathbf{X}}_{\mathsf{arr}}(t)$ or $\vec{\mathbf{X}}_{\mathsf{arr}}$	Column vector that consists of all packets arrived by time t according to their arrival order
β_k	Encoding coefficient for the packet from flow-k queue
$\mathbf{y}(t)$	Global coding vector (kernel) corresponding to the coded packet $Y(t)$
H	Packet header
PL	Packet payload

Definition 5. *For any fixed k, consider the k-th virtual sub-network. We say a subset of the $2^{K-1}+1$ virtual nodes $\{\text{vn}_S^{(k)} : \forall S\} \cup \{\text{vn}_{\text{grnd}}^{(k)}\}$, denoted by $\mathcal{C}^{(k)}$, is a* cut *for d_k if $\text{vn}_{\emptyset}^{(k)} \notin \mathcal{C}^{(k)}$ and $\text{vn}_{\text{grnd}}^{(k)} \in \mathcal{C}^{(k)}$.*

Definition 6. *For any fixed k, we say a cut $\mathcal{C}^{(k)}$, is a* proper-cut *if it also satisfies: If $\text{vn}_{S_1}^{(k)} \in \mathcal{C}^{(k)}$ for some S_1, then for all $S_2 \supseteq S_1$ we must have $\text{vn}_{S_2}^{(k)} \in \mathcal{C}^{(k)}$.*

An illustration of the above three definitions is provided in Fig. 1, for which we assume $K = 3$ and plot the k-th virtual sub-network for $k = 2$. The corresponding virtual nodes are $\left\{\text{vn}_S^{(2)} : \forall S \subseteq [3]\backslash\{2\}\right\} = \left\{\text{vn}_{\emptyset}^{(2)}, \text{vn}_{\{1\}}^{(2)}, \text{vn}_{\{3\}}^{(2)}, \text{vn}_{\{1,3\}}^{(2)}\right\}$ plus the ground node $\text{vn}_{\text{grnd}}^{(2)}$. There are exactly 5 proper-cuts:

$$\mathcal{C}_1^{(2)} = \left\{\text{vn}_{\text{grnd}}^{(2)}\right\}, \tag{12}$$

$$\mathcal{C}_2^{(2)} = \left\{\text{vn}_{\text{grnd}}^{(2)}, \text{vn}_{\{1,3\}}^{(2)}\right\}, \tag{13}$$

$$\mathcal{C}_3^{(2)} = \left\{\text{vn}_{\text{grnd}}^{(2)}, \text{vn}_{\{1\}}^{(2)}, \text{vn}_{\{1,3\}}^{(2)}\right\}, \tag{14}$$

$$\mathcal{C}_4^{(2)} = \left\{\text{vn}_{\text{grnd}}^{(2)}, \text{vn}_{\{3\}}^{(2)}, \text{vn}_{\{1,3\}}^{(2)}\right\}, \tag{15}$$

$$\mathcal{C}_5^{(2)} = \left\{\text{vn}_{\text{grnd}}^{(2)}, \text{vn}_{\{1\}}^{(2)}, \text{vn}_{\{3\}}^{(2)}, \text{vn}_{\{1,3\}}^{(2)}\right\}. \tag{16}$$

A new stability region can then be described as follows.

Proposition 3. *A rate vector $\vec{R} \triangleq (R_1, \ldots, R_K)$ is* stable *if there exists $2^K - 1$ non-negative variables $\left\{x_T \geq 0 : \forall T \in 2^{[K]}\backslash\{\emptyset\}\right\}$ satisfying the following two groups of conditions.*

[Condition 1:] The time-sharing condition:

$$\sum_{T:T\in 2^{[K]}\backslash\{\emptyset\}} x_T < 1; \tag{17}$$

and [Condition 2:] The min-cut condition: For all k and for all $\mathcal{C}^{(k)}$ being a proper cut,

$$\sum_{\substack{S:S\in 2^{[K]\backslash\{k\}},\\ \text{vn}_S^{(k)}\notin\mathcal{C}^{(k)}}} x_{S\cup\{k\}} \cdot \left(p_k + \sum_{\substack{S_X:\\ S_X\in\mathcal{F}(k,S,\mathcal{C}^{(k)})}} p_{S_X\overline{[K]\backslash(S_X\cup S)}} \right) \geq R_k, \tag{18}$$

where the set $\mathcal{F}(k, S, \mathcal{C}^{(k)})$ is defined by

$$\mathcal{F}(k,S,\mathcal{C}^{(k)}) \triangleq \Big\{ S_X \in 2^{[K]\backslash(S\cup\{k\})} : \exists \tilde{S} \subseteq (S_X \cup S) \\ \text{s.t. } \text{vn}_{\tilde{S}}^{(k)} \in \mathcal{C}^{(k)} \Big\}. \tag{19}$$

Condition 1 contains a single time-sharing inequality. To illustrate Condition 2, we expand it for the case of $K = 3$ in the following. When $K = 3$, any fixed k has 5 distinct proper-cuts $\mathcal{C}^{(k)}$. There are thus $3 \cdot 5 = 15$ inequalities in Condition 2. Specifically, using the $\mathcal{C}_1^{(2)}$ to $\mathcal{C}_5^{(2)}$ defined in

(12) to (16), Condition 2 becomes

$$\mathcal{C}_1^{(2)} : \left(x_{\{2\}} + x_{\{1,2\}} + x_{\{2,3\}} + x_{\{1,2,3\}}\right) \cdot p_2 \geq R_2, \quad (20)$$

$$\mathcal{C}_2^{(2)} : x_{\{2\}} \left(p_2 + p_{\{1,3\}\overline{2}}\right) + x_{\{1,2\}} \left(p_2 + p_{3\overline{2}}\right) + x_{\{2,3\}} \left(p_2 + p_{1\overline{2}}\right) \geq R_2, \quad (21)$$

$$\mathcal{C}_3^{(2)} : x_{\{2\}} \left(p_2 + p_{1\overline{\{2,3\}}} + p_{\{1,3\}\overline{2}}\right) + x_{\{2,3\}} \left(p_2 + p_{1\overline{2}}\right) \geq R_2, \quad (22)$$

$$\mathcal{C}_4^{(2)} : x_{\{2\}} \left(p_2 + p_{3\overline{\{1,2\}}} + p_{\{1,3\}\overline{2}}\right) + x_{\{1,2\}} \left(p_2 + p_{3\overline{2}}\right) \geq R_2, \quad (23)$$

$$\mathcal{C}_5^{(2)} : x_{\{2\}} \cdot \left(p_2 + p_{1\overline{\{2,3\}}} + p_{3\overline{\{1,2\}}} + p_{\{1,3\}\overline{2}}\right) \geq R_2. \quad (24)$$

For example, consider proper-cut $\mathcal{C}_3^{(2)}$ and the corresponding (18) in Condition 2. By the definition of $\mathcal{C}_3^{(2)}$ in (14), there are exactly two virtual nodes $\text{vn}_\emptyset^{(2)}$ and $\text{vn}_{\{3\}}^{(2)}$ not in $\mathcal{C}_3^{(2)}$, i.e., the summation in (18) is over $S = \emptyset, \{3\}$.

For the case $S = \emptyset$, we have $\mathcal{F}(2, \emptyset, \mathcal{C}_3^{(2)}) = \{\{1\}, \{1,3\}\}$. The reason is that S_X in (19) is now chosen from $2^{[K]\backslash S\cup\{k\}} = \{\emptyset, \{1\}, \{3\}, \{1,3\}\}$. If $S_X = \{1\}$ or $\{1,3\}$, there exists $\tilde{S} = \{1\}$ satisfying $\tilde{S} \subseteq S_X \cup S$ and $\text{vn}_{\tilde{S}}^{(k)} \in C_3^{(2)}$. Furthermore, no such $\tilde{S}$ exists if $S_X = \emptyset$ or $S_X = \{3\}$. As a result, $\mathcal{F}(2, \emptyset, \mathcal{C}_3^{(2)}) = \{\{1\}, \{1,3\}\}$, which then contributes to the term $x_{\{2\}} \left(p_2 + p_{1\overline{\{2,3\}}} + p_{\{1,3\}\overline{2}}\right)$ in (22).

For the case $S = \{3\}$, we have $\mathcal{F}(2, \{3\}, \mathcal{C}_3^{(2)}) = \{\{1\}\}$. The reason is that S_X in (19) is now chosen from $2^{[K]\backslash S\cup\{k\}} = \{\emptyset, \{1\}\}$. If $S_X = \{1\}$, there exists $\tilde{S} = \{1\}$ satisfying $\tilde{S} \subseteq S_X \cup S$ and $\text{vn}_{\tilde{S}}^{(k)} \in \mathcal{C}_3^{(2)}$. Furthermore, no such $\tilde{S}$ exists if $S_X = \emptyset$. As a result, $\mathcal{F}(2, \{3\}, \mathcal{C}_3^{(2)}) = \{\{1\}\}$, which contributes to the term $x_{\{2,3\}} \left(p_2 + p_{1\overline{2}}\right)$ in (22). The other four inequalities in (20) to (24) can be derived similarly.

Proposition 3 is a direct result of Proposition 5 that will be formally described in Section IV.

Corollary 1. *For all the scenarios in which the capacity region is known, i.e., either (i) $K \leq 3$, or (ii) the symmetric channels, or (iii) the channel is spatially independent and the rate vector is one-sided fair (see [5] for detailed definitions), the above stability region matches the capacity region.*

The proof of Corollary 1 is relegated to Appendix B-A.

It is worth noting that the stable rate region in Proposition 3 meets the existing outer bound in Proposition 1 for many other scenarios not specified in Corollary 1. For example, we can examine the solution space of the LP problem in Proposition 3 by a brute-force but computer-aided search, which leads to

Example 1. *For $K = 4$ and spatially independent broadcast PEC with marginal success probability $\vec{p} = (p_1, p_2, p_3, p_4) = (\frac{1}{3}, \frac{2}{5}, \frac{1}{2}, \frac{4}{7})$, the rate vector $\vec{R} = (R_1, R_2, R_3, R_4) = (\frac{96}{1193}, \frac{672}{5965}, \frac{288}{1193}, \frac{1952}{8351})$ does not belong to any of the scenarios in Corollary 1. However, it is in the boundary of the outer bound (5) and also lies within the stable rate region in Proposition 3. Therefore, it is an optimal capacity vector for the given $\vec{p}$.*

The proof of Example 1 is relegated to Appendix B-B.

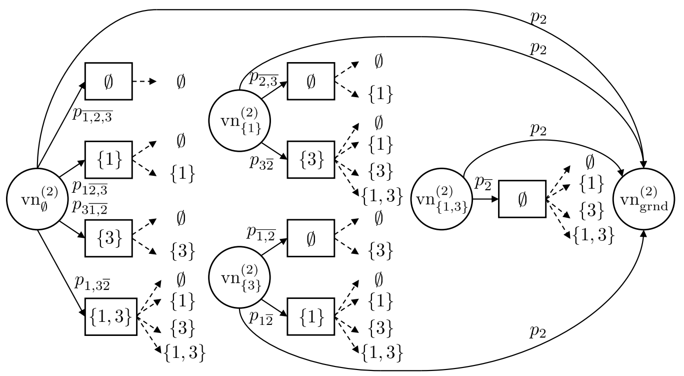

Fig. 2. The virtual sub-network for $K = 3$ and $k = 2$, where the virtual nodes are represented by circles and the auxiliary nodes by squares. The dotted edge marked by a set S is actually connected to $\text{vn}_S^{(2)}$ with ∞ capacity. For example, there are 4 outgoing edges of the auxiliary node (square) labeled by $\{1,3\}$. They are connected to $\text{vn}_\emptyset^{(2)}$, $\text{vn}_{\{1\}}^{(2)}$, $\text{vn}_{\{3\}}^{(2)}$, and $\text{vn}_{\{1,3\}}^{(2)}$, respectively, with each edge having infinite capacity. Note that the edge from auxiliary node $\{1,3\}$ to $\text{vn}_\emptyset^{(2)}$ forms part of a *self loop* in the virtual network.

Example 1 is not unique in any sense. In fact, for $K = 4$ and the *spatially independent* channels, we have run 10^5 numerical trials with different $\vec{p}$ and we have not found any $\vec{R}$ that is in the outer bound but not in Proposition 3. However, if $K = 4$ and the underlying channel is not spatially independent, we have found some rate vectors $\vec{R}$ that are in the outer bound but not in the new stability region. One such example is provided in Appendix B-C.

IV. The New Achievability Scheme

In Section IV-A, we will provide a high-level max-flow/min-cut-based interpretation of the LP problem in Proposition 3. Later in Section IV-B we will describe a sequential network coding scheme when assuming unlimited computing power. Finally, in Section V we will revise the scheme to take into account several practical considerations.

A. The Connection to the Virtual Network

We first introduce a virtual-network-based interpretation for the capacity inner bound in Proposition 3, which will shed some useful but very high-level intuition of the proposed design. It is worth noting that this virtual network representation is not needed when describing the proposed scheme. Therefore, some readers may choose to skip this subsection and directly start from the detailed scheme description in Section IV-B.

In Section III-A, we have described the $2^{K-1} + 1$ virtual nodes in the k-th virtual sub-network. We now describe its edges. See Fig. 2 for illustration.

For each $\text{vn}_S^{(k)}$ that is indexed by a subset $S \subseteq [K]\backslash\{k\}$, we add (i) an edge of capacity p_k that ends in $\text{vn}_{\text{grnd}}^{(k)}$; (ii) $2^{K-(|S|+1)}$ auxiliary nodes, each denoted by a subset $S_X \in 2^{[K]\backslash(S\cup\{k\})}$. For example suppose $K = 3$ and we focus on $\text{vn}_{\{1\}}^{(2)}$ with $S = \{1\}$. Since $[K]\backslash(S \cup \{k\}) = \{3\}$, we add two auxiliary nodes denoted by $S_X = \emptyset$ and $S_X = \{3\}$, respectively. To distinguish an auxiliary node from a regular virtual node, we represent the auxiliary nodes by squares in Fig. 2.

(iii) For each auxiliary nodes indexed by S_X, we add an edge connecting $\text{vn}_S^{(k)}$ and its auxiliary node S_X with capacity $p_{S_X\overline{[K]\backslash(S\cup S_X)}}$; (iv) Finally, for each auxiliary node S_X of $\text{vn}_S^{(k)}$, we add an edge of infinite capacity to virtual node $\text{vn}_{\tilde{S}}^{(k)}$ for all $\tilde{S} \subseteq (S\cup S_X)$. Take the virtual network in Fig. 2 for example, for which $K=3$ and $k=2$. Suppose we focus on $\text{vn}_{\{1\}}^{(2)}$ with $S=\{1\}$ and the corresponding auxiliary node indexed by $S_X=\emptyset$. Then we add an edge connecting $\text{vn}_{\{1\}}^{(2)}$ and its auxiliary node $S_X=\emptyset$ with capacity $p_{\overline{\{2,3\}}}$. Furthermore, we add two infinite-capacity edges from $S_X=\emptyset$ to $\text{vn}_{\emptyset}^{(2)}$ and $\text{vn}_{\{1\}}^{(2)}$, respectively. The description of the k-th virtual sub-network is complete.

In the following we describe how our new network coding scheme and its operations are mapped to the "packet movement" within the virtual network. For any subset $T\in 2^{[K]}$, it is well known that a single coded packet can deliver[6] $|T|$ different packets, one for each destination d_k, $k\in T$ if the following conditions hold: If for all $k\in T$ there exists a flow-k packet X_{k,i_k} that has previously been overheard by all other destinations $d_{\tilde{k}}$ for all $\tilde{k}\in T\backslash\{k\}$, then we can send a linear sum $\sum_{k\in T} X_{k,i_k}$ that benefits all d_k, $k\in T$, simultaneously. Roughly speaking, the value of x_T in (17) corresponds to the frequency of sending a linear sum of $|T|$ packets, one packet from each flow $k\in T$. Since each time slot can only choose one specific T, the frequencies $\{x_T\}$ must satisfy the time-sharing condition in (17).

The intuition behind Proposition 3 is that each virtual node $\text{vn}_S^{(k)}$ represents a queue that stores the flow-k packets that have been overheard by all $d_{\tilde{k}}$, $\tilde{k}\in S$. Since combining flows of $k\in T$ can simultaneously benefit all flows $k\in T$, a larger x_T value should help move more packets out of $\text{vn}_S^{(k)}$ where $S=T\backslash\{k\}$. That is why we scale the virtual network edge capacity of the k-th virtual sub-network, see (18), by $x_{S\cup\{k\}}$. A close look at (18) shows that for any fixed T, x_T will appear in (18) for all $k\in T$, which reflects the fact that the coded transmission can simultaneously benefit all d_k with $k\in T$.

When a coded packet combined from the node (queue) $\text{vn}_S^{(k)}$, $k\in T$, is received by destinations $\{d_i : i\neq k, i\in S_{\text{rx}}\}$, the coded packet can potentially move to the nodes (queues) $\text{vn}_{\tilde{S}}^{(k)}$, $\tilde{S}\subseteq S\cup S_{\text{rx}}$. Therefore we use the auxiliary node S_X to indicate the *additional overhearing set* $S_X = S_{\text{rx}}\backslash T = S_{\text{rx}}\backslash(S\cup\{k\})$ such that $S\cup S_{\text{rx}} = S\cup S_X$. The probability that the additional overhearing set S_X is observed, computed by $\Pr(S_X\subseteq S_{\text{rx}}\subseteq S_X\cup S) = p_{S_X\overline{[K]\backslash(S\cup S_X)}}$ in (2), is thus the capacity from the virtual node $\text{vn}_S^{(k)}$ to its auxiliary node S_X. Since the flows out of the auxiliary nodes is already constrained by the flows into the auxiliary nodes, we place no capacity limit on the out links of the auxiliary nodes, thus the infinite capacity.

Before any further discussion, we provide the following lemmas.

[6]Obviously this statement assumes that the coded packet is received successfully for all d_k, $k\in T$.

Lemma 1. *For any cut $\mathcal{C}^{(k)}$, the left hand side of* (18) *represents the corresponding cut value in the k-th virtual sub-network, i.e., the summation of the capacity (scaled by the node scheduling frequency x_T) of the edges crossing the cut $\mathcal{C}^{(k)}$.*

Lemma 2. *The following two statements are equivalent. [Statement 1:] The minimum cut value of all cuts (not necessarily proper-cut) is no less than R_k; [Statement 2:] The minimum cut value of all proper cuts is no less than R_k.*

The proofs of Lemmas 1 and 2 are relegated to Appendix C.

By Lemma 1, (18) asserts that the minimum cut value of all proper cuts must be no less than R_k, the desired rate of the k-th session. Then by Lemma 2 it further implies that the minimum cut value of all cuts is no less than R_k. Finally, by the max-flow/min-cut theorem, we can find a max-flow that achieves R_k in the virtual sub-network. As a result, if we can establish the relationship between (i) the achievable rate region of a sequential network coding scheme and (ii) the max-flow values of the virtual network, then Proposition 3 can be used to characterize the achievable rate region of the given network coding scheme. This is the basic road map of our approach.

Remark: The concept of using virtual networks to represent network coding operations is not new. Nonetheless, our virtual network representation is significantly different from existing ones, e.g., [18]. Firstly, the existing virtual network representation [18] uses 1-to-1 edges, i.e., each edge is from a single $\text{vn}_S^{(k)}$ to another $\text{vn}_{\tilde{S}}^{(k)}$. In contrast, by letting $\text{vn}_S^{(k)}$ first connect to an auxiliary node labeled by S_X and then adding edges of infinite capacity from S_X to many virtual nodes $\text{vn}_{\tilde{S}}^{(k)}$ for all $\tilde{S}\subseteq(S\cup S_X)$, we essentially create a 1-to-many *hyper-edge* from $\text{vn}_S^{(k)}$ to $\{\text{vn}_{\tilde{S}}^{(k)} : \tilde{S}\subseteq(S\cup S_X)\}$ for each S_X, see Fig. 2; Secondly, the virtual network in [18] is acyclic as the packets only move to queues of *higher levels*. Our construction contains many cycles and self-loops as illustrated in Fig. 2. This new cyclic, hyper-graph-based virtual network is a direct result of the new concept of opportunistic packet evolution.

B. A New Sequential Network Coding Scheme

We now present a new sequential coding scheme, the operation of which requires only the statistics $p_{S\overline{[K]\backslash S}}$ of the underlying PEC.

Our sequential network coding scheme has five components:

1) Maintain the queues and the global coding kernels.
2) Handle random sequential packet arrivals at time t.
3) Select a set $T^*\in 2^{[K]}$ of flows to be "added" together.
4) Generate the coded packet $Y(t)$.
5) Update the queues and the global coding kernels based on the reception status $S_{\text{rx}}(t)$ fed back from the destinations to the source.

Component 1: The queues and coding information at source. The discussion of this component can be further divided into two sub-components.

Component 1.1: Physical Memory Usage. Source s maintains $K\cdot 2^{K-1}$ queues, denoted by $Q_S^{(k)}$ for all $k\in[K]$ and $S\in 2^{[K]\backslash\{k\}}$. Each element of the queue $Q_S^{(k)}$ belongs to $\mathsf{GF}(q)$, which represents the coded or uncoded entries (payload) for destination d_k that have been overheard[7] by destinations $\{d_i : i \in S\}$. Initially at time 0, all queues $Q_S^{(k)}$ are empty.

Component 1.2: Computation Task. Source s maintains $2K$ matrices: $\mathbf{U}^{(k)}$ and $\mathbf{V}^{(k)}$ for all $k \in [K]$. Unlike the queues described previously, these $2K$ matrices can be deterministically computed from the packet arrival processes $[A_k]_1^t$ and the past channel reception status $[S_{\text{rx}}]_1^{t-1}$. Therefore, they can be computed *on-the-fly*, see (7), and no actual memory usage is needed. However, for the ease of exposition we describe these matrices as if they are being stored in memory.

Specifically, each $\mathbf{U}^{(k)}$ matrix stores the *global encoding kernels* [29] of the coded packets in the queues $\{Q_S^{(k)}\}$. The number of rows of $\mathbf{U}^{(k)}$ is thus equal to $\sum_S |Q_S^{(k)}|$. Each $\mathbf{V}^{(k)}$ matrix stores the global encoding kernels of all the packets that have been successfully delivered to d_k. The number of rows of $\mathbf{V}^{(k)}$ is equal to $\sum_{\tau=1}^{t} 1_{\{k \in S_{\text{rx}}(\tau)\}}$.

The detailed update rules of $\mathbf{U}^{(k)}$ and $\mathbf{V}^{(k)}$ and $Q_S^{(k)}$ will be discussed in Components 2 and 5. Initially at time 0, these $2K$ matrices are 0×0 matrices.

Component 2: Sequential packet arrivals. Whenever the i-th flow-k information packet arrives at s, denoted by $X_{k,i}$, we store it in the queue $Q_\emptyset^{(k)}$ by creating a new entry $W_{\mathsf{ptr}} = X_{k,i} \in \mathsf{GF}(q)$ where ptr is the pointer of the new entry W_{ptr}. The queue update is now complete. We note that the pointer ptr can be of any arbitrary format. One convenient choice is to let $\mathsf{ptr} = (k,i)$ being a vector (k,i). Note that even though when a packet arrives, the newly queued element $W_{(k,i)}$ is identical to the newly arrived $X_{k,i}$, the other queued element $W_{(k',i')}$ may not be equal to $X_{k',i'}$. *The pointer* $\mathsf{ptr} = (k,i)$ *(or* $\mathsf{ptr} = (k',i')$*) only serves as a unique pointer/index to the queued element and does not necessarily dictate the content of the queued element.* For example, we may have $W_{(2,3)} = X_{2,3} + X_{1,4}$, which means that the queue entry being referred by the pointer $(k,i) = (2,3)$ is a linear sum of $X_{2,3}$, the third packet of user d_2, and $X_{1,4}$, the fourth packet of user d_1.

We now describe how to update the matrices $\mathbf{U}^{(k)}$ and $\mathbf{V}^{(k)}$. We first add an all-zero column vector to the right of all $2K$ matrices $\mathbf{U}^{(k)}$ and $\mathbf{V}^{(k)}$. Namely, all $2K$ matrices are getting wider by one column. The number of rows of the added column vector should be clear from the context. Intuitively, the new column corresponds to the newly arrived packet $X_{k,i}$.

Let $\mathbf{e}_{(k,i)} = (0, \ldots, 0, 1)$ denote a row vector with value 1 in the last position, the position corresponding to the new column (k,i), and 0s otherwise. We then add $\mathbf{e}_{(k,i)}$ to the bottom of $\mathbf{U}^{(k)}$. Namely, $\mathbf{U}^{(k)}$ is now getting both taller and wider since one new packet $X_{k,i}$ now appears in queue $Q_\emptyset^{(k)}$ of user-k. All the other $\mathbf{U}^{(k')}$ with $k' \neq k$ just get wider (due to the added column) but the heights remain the same.

Note that for ease of exposition, we use the same pointer $\mathsf{ptr} = (k,i)$ to refer to either an entry in $Q_S^{(k)}$ or the corresponding row in $\mathbf{U}^{(k)}$. For example, the actual content of the W_{ptr} is equal to the vector product $W_{\mathsf{ptr}} = \mathsf{row}_{\mathsf{ptr}}(\mathbf{U}^{(k)}) * \vec{\mathbf{X}}_{\mathsf{arr}}$ where $\mathsf{row}_{\mathsf{ptr}}(\mathbf{U}^{(k)})$ is the row vector of $\mathbf{U}^{(k)}$ referred to by ptr and $\vec{\mathbf{X}}_{\mathsf{arr}}$ is a column vector that consists of all the packets that have arrived according to its arrival order (namely, according to the order when the columns were created.) Note that $\vec{\mathbf{X}}_{\mathsf{arr}}$ contains the packets of all K users, not just user d_k. The reason is that our scheme performs intersession coding over all K users' packets.

Component 3: Selecting a set $T^* \in 2^{[K]}$ of flows to be added together. For each time t, we select a new T^* as follows. For all $k \in [K]$ and all $\check{S} \in 2^{[K]\backslash\{k\}}$, we compute

$$q(k, \check{S}) \triangleq \min_{\tilde{S} \in 2^{\check{S}}} \left| Q_{\tilde{S}}^{(k)} \right|, \tag{25}$$

where $\left| Q_{\tilde{S}}^{(k)} \right|$ is the number of entries in $Q_{\tilde{S}}^{(k)}$. The minimum operation in (25) given k and $\check{S}$ is to compute the minimum queue length associated to the hyper edge that reaches all $\tilde{S} \in 2^{\check{S}}$ in the flow-k virtual sub-network. Also see the discussion in Section IV-A. Since a hyper edge originated from $\text{vn}_S^{(k)}$ through auxiliary node S_X can reach all virtual nodes $\text{vn}_{\tilde{S}}^{(k)}$ satisfying $\tilde{S} \subseteq \check{S} = S \cup S_X$, by setting $\check{S} = S \cup S_X$, the expression $q(k, S \cup S_X)$ computes the minimum[8] of the queue lengths of all the destinations of $\text{vn}_S^{(k)}$ when the additional overhearing set is S_X. Note that the additional overhearing is a random set. Since the backpressure is defined as the difference between the queue length of a given node and the (expected) queue lengths of all its next-hop neighbors, we compute the backpressure by

$$\text{bp}\left(Q_S^{(k)}\right) \triangleq \left|Q_S^{(k)}\right| - \sum_{S_X \in 2^{[K]\backslash(S\cup\{k\})}} p_{S_X \overline{[K]\backslash(S \cup S_X)}} q(k, S \cup S_X). \tag{26}$$

The target set T^* is then selected as follows.

$$T^* = \underset{\text{non-empty } T \in 2^{[K]}}{\arg\max} \sum_{k \in T} \text{bp}\left(Q_{T\backslash\{k\}}^{(k)}\right). \tag{27}$$

That is, if we choose to transmit the sum of the packets in queues $\{Q_{T\backslash\{k\}}^{(k)} : k \in T\}$ for some T, then such coded transmission can benefit all d_k with $k \in T$. Therefore, their individual backpressures are also summed together. Eq. (27) chooses the T^* with the largest sum.

For example, suppose $K = 3$ and the channel is spatially independent with marginal success probability $(p_1, p_2, p_3) = (0.8, 0.4, 0.5)$. At time t, suppose the lengths of $K \cdot 2^{K-1} = 12$ queues are listed as in the second column of Table III. We can calculate the corresponding $q(k,S)$ values by (25) and the $\text{bp}(Q_S^{(k)})$ by (26). For example

$$\begin{aligned} q(2, \{1,3\}) &= \min_{\tilde{S} \subseteq \{1,3\}} \left| Q_{\tilde{S}}^{(2)} \right| \\ &= \min\left\{ \left|Q_\emptyset^{(2)}\right|, \left|Q_{\{1\}}^{(2)}\right|, \left|Q_{\{3\}}^{(2)}\right|, \left|Q_{\{1,3\}}^{(2)}\right| \right\} = 1 \end{aligned}$$

[7]A more precise statement would be "*non-interfering* to destinations $\{d_i : i \in S\}$" rather than "overheard by $\{d_i : i \in S\}$". The subtle difference will be explained in Appendix E. For ease of exposition, we use the term "overheard" in this subsection.

[8]The reason we choose the minimum is mostly based on the corresponding Lyapunov drift analysis in Appendix F. Intuitively, since we can opportunistically decide to which virtual node of the hyper edge should we move the packet, we should move the packet to the one with the smallest queue length (i.e., the destination node currently experiences the largest backpressure), and thus the minimum operation.

TABLE III
A $K=3$ EXAMPLE OF COMPUTING $q(k,S)$ AND THE BACKPRESSURE TERM $\text{bp}(Q_S^{(k)})$ FROM $\left|Q_S^{(k)}\right|$.

(k,S)	$Q_S^{(k)}$	$q(k,S)$	$\text{bp}(Q_S^{(k)})$
$(1,\emptyset)$	1	1	0.84
$(1,\{2\})$	2	1	1.90
$(1,\{3\})$	1	1	0.88
$(1,\{2,3\})$	0	0	0
$(2,\emptyset)$	5	5	3.68
$(2,\{1\})$	3	3	1.80
$(2,\{3\})$	1	1	0.40
$(2,\{1,3\})$	2	1	1.40
$(3,\emptyset)$	3	3	1.90
$(3,\{1\})$	2	2	1.00
$(3,\{2\})$	4	3	2.90
$(3,\{1,2\})$	2	2	1.00

and

$$\text{bp}\left(Q_{\{3\}}^{(2)}\right)=\left|Q_{\{3\}}^{(2)}\right|-p_{\overline{\{1,2\}}}q(2,\{3\})-p_{1\overline{2}}q(2,\{1,3\})=0.4.$$

With the $\text{bp}(Q_S^{(k)})$ values, we calculate $\sum_{k\in T}\text{bp}(Q_{T\backslash\{k\}}^{(k)})$ for all $T=\{1\},\{2\},\{3\},\{1,2\},\{1,3\},\{2,3\},\{1,2,3\}$ and they are $0.84, 3.68, 1.90, 3.70, 1.88, 3.30, 2.40$ respectively. Finally we choose the target set $T^*=\{1,2\}$, which has the maximum backpressure sum 3.70.

Component 4: Generating the coded packet. After the set T^* is decided, for each $k\in T^*$, we extract the head-of-line element $W_{(k,j_k)}$ from queue $Q_{T^*\backslash\{k\}}^{(k)}$. If queue $Q_{T^*\backslash\{k\}}^{(k)}$ is empty, we simply set the head-of-line element to be $W_{(k,0)}\triangleq 0$, the zero/null packet. The to-be-transmitted coded packet $Y(t)$ is computed by

$$Y(t)=\sum_{k\in T^*}\beta_k\cdot W_{(k,j_k)} \tag{28}$$

That is, the transmitting packet is the linear sum of the head-of-line packets multiplied by β_k. Define

$$\mathbf{y}(t)=\sum_{k\in T^*}\beta_k\cdot\mathsf{row}_{(k,j_k)}(\mathbf{U}^{(k)}). \tag{29}$$

Recall that $\mathsf{row}_{(k,j_k)}(\mathbf{U}^{(k)})$ is the global coding kernel of the stored packet $W_{(k,j_k)}$. $\mathbf{y}(t)$ is thus the global coding kernel for the coded transmission $Y(t)$. Note that only $Y(t)$ is transmitted and we do not transmit $\mathbf{y}(t)$.

For ease of exposition, one may assume β_k is chosen uniformly randomly from a sufficiently large $\mathsf{GF}(q)$ and skip to Component 5 directly. This simple random coding scheme will work if $q=\infty$. On the other hand, the following paragraphs present a deterministic construction that holds for any finite $q\geq K$.

For all $k\in T^*$, construct a matrix $\mathbf{U}_{j_k}^{\prime(k)}$ by removing $\mathsf{row}_{(k,j_k)}(\mathbf{U}^{(k)})$ corresponding to $W_{(k,j_k)}$ from $\mathbf{U}^{(k)}$. The coefficients $\{\beta_k:k\in T^*\}$ can be deterministically chosen by following two steps. Step 1: Check whether $\mathsf{row}_{(k,j_k)}(\mathbf{U}^{(k)})$ is linearly dependent of the rows of $((\mathbf{U}_{j_k}^{\prime(k)})^\mathsf{T},(\mathbf{V}^{(k)})^\mathsf{T})^\mathsf{T}$, where $((\mathbf{U}_{j_k}^{\prime(k)})^\mathsf{T},(\mathbf{V}^{(k)})^\mathsf{T})^\mathsf{T}$ is the vertical concatenation of matrices $\mathbf{U}_{j_k}^{\prime(k)}$ and $\mathbf{V}^{(k)}$. If so, then set $\beta_k=0$. Namely, this is a degenerate case and the corresponding $W_{(k,j_k)}$ will no longer participate in the coding computation. Perform this dependency check for all $k\in T^*$ and denote by T' those non-degenerate k for which the β_k values are still undecided.

Step 2: Find arbitrarily one set of $\{\beta_k:k\in T'\}$ values satisfying the following property: Define $\mathbf{y}(t)\triangleq\sum_{k\in T'}\beta_k\cdot\mathsf{row}_{(k,j_k)}(\mathbf{U}^{(k)})$. For all $k\in T'$, vector $\mathbf{y}(t)$ has to be linearly independent of the rows of $((\mathbf{U}_{j_k}^{\prime(k)})^\mathsf{T},(\mathbf{V}^{(k)})^\mathsf{T})^\mathsf{T}$. Lemma 5 in Appendix D guarantees the existence of such coefficients $\{\beta_k:k\in T'\}$. The description of how to generate the coded packet $Y(t)$ is complete.

Component 5: Packet movement among the queues — The opportunistic packet evolution component. We describe the packet movement for one specific k_0 value while the same mechanism has to be applied to all $k\in[K]$.

If $k_0\notin T^*$, then no entry in $\{Q_S^{(k_0)}:\forall S\}$ will move. If $k_0\in T^*$, then we consider one and only one queue: $Q_{T^*\backslash\{k_0\}}^{(k_0)}$. Recall that $W_{(k_0,j_{k_0})}$ is the head-of-line packet of $Q_S^{(k_0)}$ selected in Component 4. In the degenerate case $Q_{T^*\backslash\{k_0\}}^{(k_0)}=\emptyset$ or equivalently $W_{(k_0,j_{k_0})}=0$, then no further action will be needed. Component 5 is complete. We go back to Component 2 for time $t+1$.

In a non-degenerate case, that is $Q_{T^*\backslash\{k_0\}}^{(k_0)}\neq\emptyset$, we first remove the entry $W_{(k_0,j_{k_0})}$ of $Q_{T^*\backslash\{k_0\}}^{(k_0)}$ as well as the corresponding $\mathsf{row}_{(k_0,j_{k_0})}(\mathbf{U}^{(k_0)})$. If d_{k_0} has received the coded transmission $Y(t)$, then insert the global coding kernel $\mathbf{y}(t)$ of $Y(t)$ as a new row to the matrix $\mathbf{V}^{(k_0)}$. If d_{k_0} did not receive $Y(t)$, then we insert row vector $\mathbf{y}(t)$ as a new row to matrix $\mathbf{U}^{(k_0)}$ and insert the coded content $Y(t)$ to a new destination queue $Q_{\tilde{S}^*}^{(k_0)}$. Effectively, the location of the old entry $W_{(k_0,j_{k_0})}$ has been moved from the old queue $Q_{T^*\backslash\{k_0\}}^{(k_0)}$ to the new queue $Q_{\tilde{S}^*}^{(k_0)}$ while the content is updated from the old $W_{(k_0,j_{k_0})}$ to the new $Y(t)$.

The new destination queue $Q_{\tilde{S}^*}^{(k_0)}$ is decided as follows. Define $\check{S}=(T^*\backslash\{k_0\})\cup S_{\text{rx}}(t)$ where the reception set $S_{\text{rx}}(t)$ is obtained from ACK/NACK. Define $\tilde{S}^*$ as the $\tilde{S}$ that minimizes the expression $q(k_0,\check{S})$ in (25). Then we choose $Q_{\tilde{S}^*}^{(k_0)}$ as the destination queue.

Component 5 is the most distinct feature of our scheme. Unlike existing solutions that choose the destination queue as a deterministic function of the reception set $S_{\text{rx}}(t)$ (usually in an acyclic fashion), Component 5 decides the movement *opportunistically* with the help of $S_{\text{rx}}(t)$ and the neighbors' queue lengths, possibly in a cyclic way. That is, it is possible that $\tilde{S}^*=T^*\backslash\{k_0\}$ or even $\tilde{S}^*\subsetneq T^*\backslash\{k_0\}$. Since the newly stored entry $Y(t)$ in $Q_{\tilde{S}^*}^{(k_0)}$ differs from the older entry $W_{(k_0,j_{k_0})}$, the stored packet *evolves* over time. Jointly these two features give the name of *opportunistic packet evolution* of our design.

We use the following example to illustrate the operations in Component 5. Consider $K=3$ and suppose that at time t, the target set is $T^*=\{1,2,3\}$. Also suppose after transmission, the reception set is $S_{\text{rx}}(t)=\{3\}$. We now discuss how the packets move within the queues.

Consider the packets for d_2, i.e., $k_0=2$. The head-of-line packet $W_{(2,j_2)}$ is first removed from queue $Q_{\{1,3\}}^{(2)}$ and the corresponding row is removed from $\mathbf{U}^{(2)}$. Then, suppose the flow-2 queue lengths are $|Q^{(2)}_{\{\emptyset\}}|=5$, $|Q^{(2)}_{\{1\}}|=3$, $|Q^{(2)}_{\{3\}}|=1$, and $|Q^{(2)}_{\{1,3\}}|=2$. Since $\check{S}=(T^*\backslash\{k_0\})\cup S_{\text{rx}}(t)=\{1,3\}$ we compute

$$q(2,\{1,3\})=\min\left\{|Q^{(2)}_{\{\emptyset\}}|,|Q^{(2)}_{\{1\}}|,|Q^{(2)}_{\{3\}}|,|Q^{(2)}_{\{1,3\}}|\right\}=1$$

and the minimizer $\tilde{S}^*=\{3\}$. The new packet $Y(t)$ is then injected to queue $Q^{(k_0)}_{\tilde{S}^*}=Q^{(2)}_{\{3\}}$ and $\mathbf{y}(t)$ is inserted back to $\mathbf{U}^{(2)}$.

For the other destination d_3, i.e., $k_0=3$. the head-of-line entry $W_{(3,j_3)}$ is first removed from $Q^{(3)}_{\{1,2\}}$ and the corresponding row is removed from $\mathbf{U}^{(3)}$. Since d_3 received the packet (because $S_{\text{rx}}(t)=\{3\}$), we inject $\mathbf{y}(t)$ to matrix $\mathbf{V}^{(3)}$. The packet movement for destination d_1 is similar to that of d_2.

The proposed scheme operates by repeatedly executing Components 2 to 5 for each time slot. We can then prove the following results.

Proposition 4. *The above 5-component sequential coding scheme satisfies the decodability condition in* (9) *and* (10).

Proposition 5. *The proposed 5-component sequential coding scheme can stabilize any given rate vector $\vec{R}=(R_1,\cdots,R_K)$ if $\vec{R}$ satisfies Proposition 3.*

Proposition 6. *The proposed 5-component sequential coding scheme can stabilize $\vec{R}=(R_1,\cdots,R_K)$ only if $\vec{R}$ satisfies Proposition 3, provided we replace the strict inequality in* (17) *by $\leq$.*

The proof of Proposition 4 is provided in Appendix E. The proofs of Propositions 5 and 6 are relegated to Appendix F.

Remark: The proposed scheme allows cyclic packet movement and thus packets sometimes move from $Q^{(k)}_{S_1}$ to $Q^{(k)}_{S_2}$ with $S_1\supsetneq S_2$. This is in sharp contrast with the schemes in [5], [18], which always move packets to queues of larger overhearing sets (thus being acyclic). The drawback of the strict acyclic packet movement is that when deciding to send a coding operation that combines all flows $k\in T$, the queue lengths of the $|T|$ participating virtual queues $\{Q^{(k)}_{T\backslash\{k\}}:k\in T\}$ may be highly unbalanced since many packets may have already moved out of $Q^{(k_0)}_{T\backslash\{k_0\}}$ for a specific k_0. To mitigate the unbalanced queue lengths, the schemes in [5], [18] searches for *collapsed overhearing set matching* (COSM) when the encoding set being T. That is, when a flow-k_0 queue $Q^{(k_0)}_{T\backslash\{k_0\}}$ is empty, COSM chooses the packet from a queue $Q^{(k_0)}_{S}$ with strictly larger overhearing set $S\supsetneq T\backslash\{k_0\}$. This substitution effectively moves some packets from $Q^{(k_0)}_{S}$ to $Q^{(k_0)}_{T\backslash\{k_0\}}$ to create more overhearing set matching opportunities.

The performance improvement of COSM is due to its exhaustive examination of all possible coding combinations by temporarily suppressing some of the overhearing sets. However, searching for COSM is of exceedingly high complexity. To mitigate the complexity, the proposed scheme uses backpressure to balance virtual queue lengths preemptively so that the packets are evenly distributed among all virtual queues. As a result, when choosing a coding decision T, we only need to combine packets from $\{Q^{(k)}_{T\backslash\{k\}}:k\in T\}$ without resorting to expensive search of COSM. Note that the proposed backpressure scheme continuously rebalances the queue lengths for each time slot based on the feedback $S_{\text{rx}}(t)$, and the rebalancing sometimes moves packets to queues of smaller overhearing subsets in order to keep those queues from depletion. One contribution of this work is to prove that such a counterintuitive packet movement, once performed optimally, attains the same performance as the more complicated COSM-based schemes.

V. Practical Sequential Coding Scheme with Overhead

In Section II, we assume *network-wide* channel output feedback $S_{\text{rx}}(t)$ and traffic pattern $\vec{A}(t)$ are causally available to all the destinations and we assume that source s is of unlimited computing power/memory so that the global encoding kernel matrices $\mathbf{U}^{(k)}$ and $\mathbf{V}^{(k)}$ can either be computed on-the-fly or be completely stored with no penalty. In this section, we modify the previous scheme to handle the lack of the knowledge of $S_{\text{rx}}(t)$ and $\vec{A}(t)$ at d_k and the limited computing power and memory at s in practice. The modification incurs additional overhead that is absent in the previous theoretic setting and we examine the overhead by simulation in Section VI.

A. A Header-based Implementation

A standard approach in practical network coding protocols is to append a *header* to each of the packets in the queue $Q^{(k)}_S$, which contains the *global encoding kernel*. The global coding vector $\mathbf{y}(t)$ (the header) is transmitted together with the payload $Y(t)$. Destination d_k can then decode the original message packets $X_{k,i}$, $i=1,2,\cdots$ using the global coding vector $\mathbf{y}(t)$.

However, recall that the global encoding kernel of any entry $W_{(k,i)}$ is $\mathsf{row}_{(k,i)}(\mathbf{U}^{(k)})$, the row vector of the matrix $\mathbf{U}^{(k)}$, and per our construction its dimension grows every time there is a new packet arrival. The overhead of sending the header thus increases over time. Furthermore, to implement the above naive scheme, matrices $\mathbf{U}^{(k)}$ and $\mathbf{V}^{(k)}$ need to be stored in s and again the increasing size of $\mathbf{U}^{(k)}$ and $\mathbf{V}^{(k)}$ requires memory usage that grows linearly with respect to time. In the sequel, we focus on how one can "prune" the matrices in practice.

Our modification contains two parts. First, we let each queuing entry $W_{(k,i)}=(\mathsf{H},\mathsf{PL})$ contain two parts, a header H and payload PL. The header H consists of the global encoding kernel $\mathsf{row}_{(k,i)}(\mathbf{U}^{(k)})$. However, since the dimension of $\mathsf{row}_{(k,i)}(\mathbf{U}^{(k)})$ grows with time, we take advantage of the observation that the vector $\mathsf{row}_{(k,i)}(\mathbf{U}^{(k)})$ is sparse and let H store the lossless compressed version of $\mathsf{row}_{(k,i)}(\mathbf{U}^{(k)})$. Specifically, we let H contain a set of tuples $(k,i,\beta_{k,i})$, where $\beta_{k,i}$ is the global encoding coefficient multiplied to the i-th packet of destination k. Namely, we have

$$\mathsf{PL}=\sum_{(k,i,\beta_{k,i})\in\mathsf{H}}\beta_{k,i}X_{k,i}. \tag{30}$$

Since the header only records the coefficients of all participating packets, the header size is small for sparse coding vectors. Since in the original scheme, the $\mathbf{U}^{(k)}$ matrices store all the global coding vectors for all the entries of $Q_S^{(k)}$ for d_k, this lossless compression effectively replaces/subsumes the $\mathbf{U}^{(k)}$ matrices and we do not need to store $\mathbf{U}^{(k)}$ anymore.

Even though we do not need to store the $\mathbf{U}^{(k)}$ matrix, we still need to store and constantly update the $\mathbf{V}^{(k)}$ matrix for all $k \in [K]$. Define $\mathcal{HS}$ as the collection of the headers of all queueing entries stored in the $K \cdot 2^{K-1}$ virtual queues $\{Q_S^{(k)} : k \in [K], S \in 2^{[K]\setminus\{k\}}\}$. The pseudo code in Algorithm 1 describes the corresponding operations of updating and pruning the $\mathbf{V}^{(k)}$ matrix.

Algorithm 1 Pruning the $\mathbf{V}^{(k)}$ Matrix at time t

```
1: input $\{\mathbf{V}^{(k)} : k \in [K]\}$ and $\mathcal{HS}$.
2: Recall that each header $\mathsf{H} \in \mathcal{HS}$ contains a variable number of tuples $(k, i, \beta_{k,i})$. Temporarily ignore the last coordinate $\beta_{k,i}$ in the tuple and define $\mathcal{J}_U \triangleq \{(k,i) : \forall (k,i) \in \mathsf{H}, \mathsf{H} \in \mathcal{HS}\}$ as the collection of all $(k,i)$ that appear in the stored headers.
3: Define $\mathcal{N}_0$ as the collection of $(k,i)$ simultaneously satisfying (i) packet $X_{k,i}$ has already arrived, and (ii) $(k,i) \notin \mathcal{J}_U$. Namely, $\mathcal{N}_0$ contains those $(k,i)$ that no longer appears in any of the headers.
4: for $(k_0, i_0) \in \mathcal{N}_0$ do
5:   for $k \in [K]$ do
6:     if the $(k_0, i_0)$ column has not been pruned from $\mathbf{V}^{(k)}$ before then
7:       Swap the $(k_0, i_0)$ column of $\mathbf{V}^{(k)}$ with its leftmost column. Namely, shift the $(k_0, i_0)$ column to the left as the first column of $\mathbf{V}^{(k)}$.
8:       if the first column of the new $\mathbf{V}^{(k)}$ is not a zero column then
9:         Apply Gaussian elimination to $\mathbf{V}^{(k)}$ so that the first column becomes $\mathbf{e}_1 = (1, 0, \ldots, 0)^T$.
10:        Remove the first row of the resulting $\mathbf{V}^{(k)}$.
11:      end if
12:      Remove the first column of $\mathbf{V}^{(k)}$.
13:    end if
14:    Apply Gaussian elimination to $\mathbf{V}^{(k)}$ so that the resulting $\mathbf{V}^{(k)}$ is of the row-echelon form. Remove all the zero rows of $\mathbf{V}^{(k)}$.
15:  end for
16: end for
17: return the final matrices $\{\mathbf{V}^{(k)} : k \in [K]\}$.
```

After pruning matrix $\mathbf{V}^{(k)}$, recall that when choosing the coding coefficients $\{\beta_k\}$ in Component 4, we have to find β_k such that the resulting $\mathbf{y}(t)$ in (29) is independent of the row of $((\mathbf{U}'^{(k)}_{j_k})^\mathsf{T}, (\mathbf{V}^{(k)})^\mathsf{T})^\mathsf{T}$ for all k. Let $\tilde{\mathbf{V}}^{(k)}$ denote the output matrix after applying the above pruning procedure to the original $\mathbf{V}^{(k)}$ matrix. Recall that $\mathbf{U}^{(k)}$, $\mathbf{U}'^{(k)}$, and $\mathbf{V}^{(k)}$ in Component 4 have the same set of columns while $\tilde{\mathbf{V}}^{(k)}$ only keeps a subset of the columns of $\mathbf{V}^{(k)}$. If we keep the columns of the $\mathbf{U}^{(k)}$ and $\mathbf{U}'^{(k)}$ matrices that also appear in $\tilde{\mathbf{V}}^{(k)}$ and denote the resulting matrices and global coding kernel as $\tilde{\mathbf{U}}^{(k)}$, $\tilde{\mathbf{U}}'^{(k)}$, and $\tilde{\mathbf{y}}(t)$, respectively, then we have the following lemma.

Lemma 3. $\mathbf{y}(t) \in \langle \mathbf{U}'^{(k)} \rangle \oplus \langle \mathbf{V}^{(k)} \rangle$ *if and only if* $\tilde{\mathbf{y}}(t) \in \langle \tilde{\mathbf{U}}'^{(k)} \rangle \oplus \langle \tilde{\mathbf{V}}^{(k)} \rangle$.

The proof of Lemma 3 is relegated to Appendix G.

That is, instead of checking whether the coding vector belongs to the row space of the three original matrices $\mathbf{U}^{(k)}$, $\mathbf{U}'^{(k)}$, and $\mathbf{V}^{(k)}$ of which the size grows linearly with respect to time, now we only need to generate the coding vector $\tilde{\mathbf{y}}(t)$ according to the pruned versions $\tilde{\mathbf{U}}^{(k)}$, $\tilde{\mathbf{U}}'^{(k)}$, and $\tilde{\mathbf{V}}^{(k)}$. (Matrices $\tilde{\mathbf{U}}^{(k)}$ and $\tilde{\mathbf{U}}'^{(k)}$ are derived implicitly from the header of the entries in $Q_S^{(k)}$.)

B. Other Issues for Practical Implementation

The benefits of the modified scheme in Section V-A are two-fold. Firstly, the source node uses packet headers to reduce its encoding complexity and memory usage. Secondly, the destination d_k does not need the knowledge of the network-wide reception status $S_{\text{rx}}(t)$ and only needs the coding vectors in the headers during decoding.

In this subsection, we discuss some other practical issues that may arise during implementation. They are beyond the scope of this theoretical study and we thus only outline some possible solutions. The designs discussed in this subsection will not be used during the simulation in Section VI.

1) Unknown Channel Statistic: All our results assume that the channel statistics $p_{S\overline{[K]\setminus S}}$ is stationary and known to the source. When $p_{S\overline{[K]\setminus S}}$ is unknown, we can estimate $p_{S\overline{[K]\setminus S}}$ on-the-fly via the history of the ACK/NACK feedback. The estimation $\hat{p}_{S\overline{[K]\setminus S}}$ can then be used as a substitute of the true $p_{S\overline{[K]\setminus S}}$ during the backpressure computation. If the channel statistics are indeed stationary, then the empirical estimation will eventually converge to the true distribution.

2) Variable Header Length: The header-based scheme in Section V-A allows for variable header lengths, which may grow unboundedly as time progresses, even though in simulation it grows quite slowly. To address this potential unboundedness, during implementation we may set an upper limit such that whenever a queueing entry has header length exceeding that limit, the source node would stop the normal coding operations and transmit that entry uncodedly until it is received by its target destination. In this way, the header length can be kept strictly bounded away from infinity.

3) Delay: As will be verified by simulation in Section VI, the proposed scheme has reasonable delay and queue length performance when operating away from the capacity. However, when we are getting closer and closer to the capacity, inevitably the queue lengths will grow to infinity just like any queueing system that operates closely to its theoretic limit. Furthermore, our scheme also exhibits the so-called *decoding delay*, which again grows unboundedly when operating arbitrarily close to capacity.

One solution that mitigates this fundamental phenomenon is *admission control*. Periodically, we could stop admitting new packets and focus on sending all packets in the current queues and emptying the queues completely. By periodically suspending the incoming packets and emptying the queues, we can ensure that the decoding and queueing delay remain bounded all the time. Like all other admission control schemes, the cost of this scheme is that the achievable throughput decreases since no new packets are admitted during suspension. There are many variants how the admission control can be better implemented, e.g., statistically or deterministically, which is beyond the scope of this work.

4) Deadline: It is possible that each individual packet $X_{k,i}$ has some deadline, i.e., it needs to be delivered by a certain amount of time. One solution to improve the deadline performance is that when a packet has been in the queues for too long and is about to expire, we can give it higher scheduling priority and temporarily overwrite the throughput-optimal backpressure scheduler described in Section IV-B. In this way, we can control the deadline at the cost of reduced throughput.

VI. Simulation Results

In this section, we assume that the arrival of the flow-k packets is i.i.d. Poisson with rate R_k. Assume $K = 4$ and spatially independent PEC with marginal success probability $(p_1, p_2, p_3, p_4) = (\frac{1}{3}, \frac{2}{5}, \frac{1}{2}, \frac{4}{7})$. Example 1 shows that $\vec{R}^* \triangleq (R_1^*, R_2^*, R_3^*, R_4^*) = (\frac{96}{1193}, \frac{672}{5965}, \frac{288}{1193}, \frac{1952}{8351})$ is at the boundary of the true capacity region. We then set the actual arrival rate to be $\vec{R}_\alpha = (R_1, R_2, R_3, R_4) = \alpha\vec{R}^*$ where $0 < \alpha \leq 1.03$ measures how closely we are operating at the capacity. Each simulation trial lasts for 10^5 time slots. For reference, we use the same (p_1, p_2, p_3, p_4) value and compute the largest α that can possibly be achieved by a traditional non-network-coded scheme. That is, the largest α satisfying

$$\frac{\alpha R_1^*}{p_1} + \frac{\alpha R_2^*}{p_2} + \frac{\alpha R_3^*}{p_3} + \frac{\alpha R_4^*}{p_4} \leq 1 \tag{31}$$

is $\alpha^*_{\text{time.sharing}} = \frac{1193}{1688} \simeq 0.71$.

To measure the sum of queue lengths $\sum_{k,S} |Q_S^{(k)}|$, we first fix the α value being considered. Then for each trial we compute the time average of $\sum_{k,S} |Q_S^{(k)}|$ over the 10^5 time slots. We repeat 100 trials for any given α value and Fig. 3 reports the average queue length for different α values with the corresponding 95% confidence intervals. For any $\alpha < 1$, the sum of queue lengths remains very small even if we continue the simulation trial beyond 10^5 time slots. If $1 < \alpha$, then the queue lengths grow linearly with respect to the number of time slots. Note that the sum of queue lengths corresponds to how many packets the source s needs to store in the memory during sequential encoding, which is around 54.9 packets when $\alpha = 0.95$. It is worth noting that the queue lengths of any uncoded scheme will become unbounded whenever $\alpha > \alpha^*_{\text{time.sharing}} = 0.71$.

The control overhead is measured by the length of the header of each transmission. We noticed that in our header-based scheme in Section V-A, the header length grows whenever we mix multiple packets together. Recall that the virtual packet movement is originated from the queue $Q_S^{(k)}$ and ends in the destination queue $Q_{\tilde{S}^*}^{(k)}$. To further control the growth rate of the header length, we notice that if $\tilde{S}^* \subseteq S$, then instead of injecting a new packet $Y(t)$ with coding vector $\mathbf{y}(t)$ into

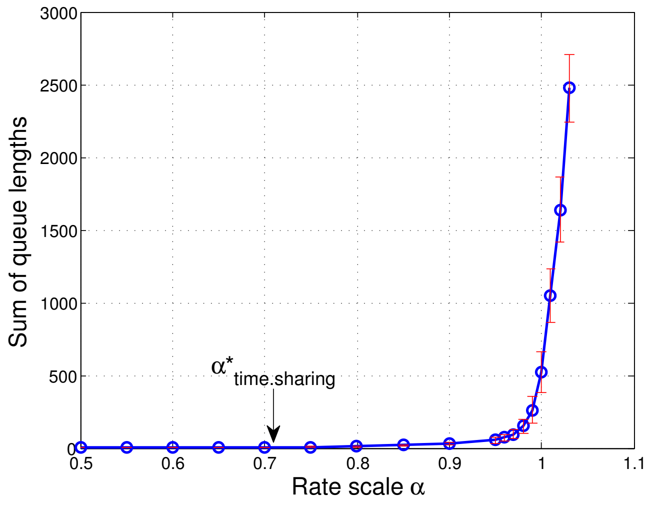

Fig. 3. The sum of queue lengths $\sum_{k,S} |Q_S^{(k)}|$ for different α.

the destination queue $Q_{\tilde{S}^*}^{(k)}$, we can simply inject the original packet $W_{(k,i)}$ packet with coding vector $\mathsf{row}_{(k,i)}(\mathbf{U}^{(k)})$ into the same destination queue $Q_{\tilde{S}^*}^{(k)}$. The reason is that the overhearing set of the original queue being S is larger than the new overhearing set $\tilde{S}^*$ and thus we can reuse the same packet $W_{(k,i)}$ instead of the new packet $Y(t)$. Since the original coding vector $\mathsf{row}(k,i)(\mathbf{U}^{(k)})$ generally has a shorter expression than the new coding vector $\mathbf{y}(t)$, this modification reduces the growth rate of the header length.

Using this modified scheme, we run 5 trials with $\alpha = 0.95$ and the empirical pmf of header length (number of triples), computed over 10^5 time slots and 5 trials, is shown in Fig. 4. The average header length per transmission is about 2.7, which reflects the average number of combined packets in a coded packet. In fact the maximum header length in 10^5 time slots is about 135 (unit: tuples) and 95% packets has header length no larger than 10 (unit: tuples). That is, if each $(k, i, \beta_{k,i})$ tuple in the header takes 4 bytes, then 95% packets use no more than 40 bytes for headers, or 4% of a regular 1000 bytes packet, and the average size of header is $2.7 \cdot 4 = 10.8$ bytes, which is 1.1% of the payload length.

Finally, we compare the per-packet total delay, which is measured by the time span between a packet first arriving at source s until it being fully decoded at destination d_k (queueing plus decoding delay). We run 5 trials with $\alpha = 0.95$ and Fig. 5 shows the empirical pmf of the total delay over 10^5 time slots and 5 trials. The average per-packet total delay is 513.6 time slots. Specifically the average per-packet total delay are 882.8, 774.3, 406.6, and 370.2 for each destination d_1, d_2, d_3, and d_4, respectively. Due to the spatial independent PEC in the simulation, the delay is roughly inversely proportional to marginal success probability (p_1, p_2, p_3, p_4).

Figs. 4 and 5 also plot the empirical pmfs of the header length and the total delay when operated at $\alpha = 0.9$. Comparing the case of $\alpha = 0.9$ and the previous, more demanding case of $\alpha = 0.95$, one can see that the average header length decreases slightly from 2.7 to 2.5; but the average total decoding delay decreases 5 fold from 513.6 to 108.7. The

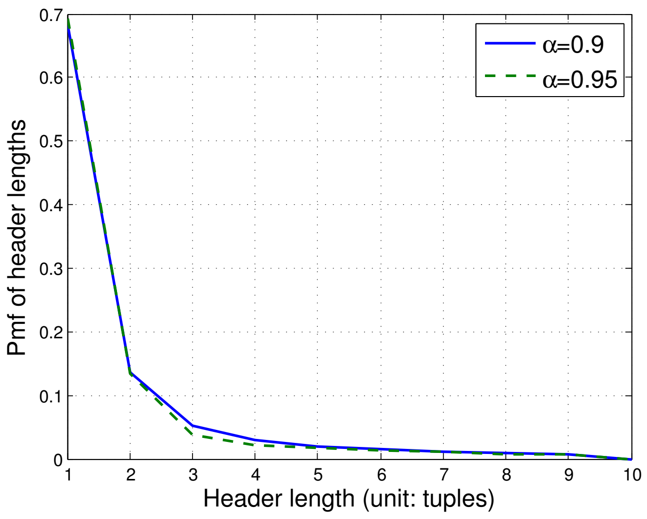

Fig. 4. Pmf of the header length: When $\alpha = 0.9$, the largest header length in the simulation of 10^5 time slots is 116 and the average is 2.5. When $\alpha = 0.95$, the largest and the average header lengths become 135 and 2.7.

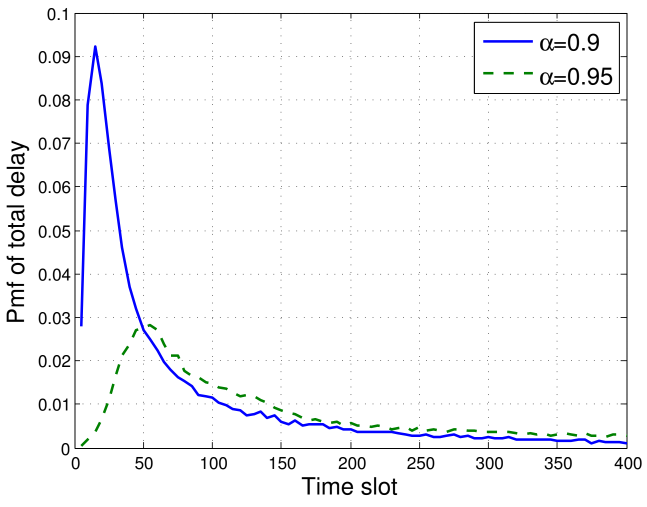

Fig. 5. Pmf of the total delay: When $\alpha = 0.9$, the largest total delay in the simulation of 10^5 time slots is 3332 and the average is 108.7. 94% of the packets have total delay ≤ 400 time slots. When $\alpha = 0.95$, the largest and the average delay become 13382 and 513.6. 67% of the packets have total delay ≤ 400 time slots.

substantial delay reduction is as expected since the closer we are operating to the capacity, the higher number of packets will be queued (longer queueing delay) and their mixture will lead to much longer decoding delay at the receiver.

VII. Conclusion

We have presented a new network coding scheme for 1-to-K broadcast packet erasure channels with causal ACK/NACK. The achievable rate region matches the capacity region for all the scenarios in which the capacity is known. The proposed scheme has many desirable features, including sequential encoding and being adaptive to unknown arrival rates.

Appendix A
Proof of Proposition 2

Assumptions and notations: Consider any arbitrarily given rate vector $\vec{R} = (R_1, \cdots, R_K)$. Without loss of generality, also assume $\min_{k\in[K]} R_k > 0$ and $\min_{k\in[K]} p_k > 0$. Otherwise, we can simply remove the degenerate user and treat the K-session transmission problem as an equivalent $(K-1)$-session transmission problem. For notational simplicity, we denote $R_{\max} \triangleq \max_{k\in[K]} R_k$ and $p_{\min} \triangleq \min_{k\in[K]} p_k$. Also note that since we assume the arrival process being Poisson with arrival rate R_k, the variance of the arrival process is also R_k. We use $V_k \triangleq R_k$ to denote the variance of the i.i.d. arrival process and $V_{\max} \triangleq R_{\max}$ to denote the maximum of the variances of the K arrival processes.[9]

High-level description of the proof: For any stable rate $\vec{R}$ and the corresponding sequential network coding scheme, see Definition 3, and for any $\epsilon > 0$ and $0 < \delta < R_{\min}$, we will use this sequential coding scheme to design a block coding scheme with block length n and rate vector

$$\vec{R}^{\delta} \triangleq \vec{R} - \delta \cdot \mathbf{1} = (R_1 - \delta, \cdots, R_K - \delta)$$

such that the error rate of the block coding scheme is less than or equal to ϵ. This shows that for any stable $\vec{R}$ of the sequential-coding setting, the corresponding $\vec{R}^{\delta}$ is achievable in the block-coding setting. Since the capacity region is the closure of all achievable rates, this construction completes the proof of Proposition 2.

Detailed proof: We first prove the following lemma.

Lemma 4. *For any stable rate $\vec{R} = (R_1, \ldots, R_K)$ and the corresponding sequential network coding scheme, see Definition 3, there exist a constant $D < \infty$ and an infinite deterministic integer sequence $t_1 < t_2 < t_3 < \cdots$ such that*

$$\mathsf{E}\{|Q(t_i)|\} \leq D, \quad \forall i \in [1, \infty), \tag{32}$$

where $|Q(t)|$ is the amount of memory usage at time t.

Proof. Since $\vec{R}$ is stable, we can define a finite number U_Q by

$$U_Q \triangleq \limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=1}^{t} \mathsf{E}\{|Q(\tau)|\} < \infty. \tag{33}$$

For any finite but fixed $D > U_Q$, we will prove by contradiction that we can find a subsequence $t_1 < t_2 < \cdots$ satisfying Lemma 4. Suppose not. Then there exists $T \in \mathbb{N}$ such that $\mathsf{E}\{|Q(t)|\} > D$ for all $t \geq T$. Together with (33) we then have $U_Q \geq D$, which contradicts the initial choice of $D > U_Q$. The proof is thus complete. □

Using the D value and the sequence $\{t_i\}$ that satisfy Lemma 4, we then define three positive integers n_2, n_1, and

[9]By using the variance notation $V_{\max}$, our proof also holds for any i.i.d. arrival processes with finite second moment $V_{\max} < \infty$.

n by

$$n_2 \triangleq \left\lceil \max\left\{ \frac{8K}{p_{\min}} \ln\left(\frac{3K}{\epsilon}\right), \frac{6KD}{p_{\min}\cdot\epsilon} \right\} \right\rceil \tag{34}$$

$$n_1 \triangleq t_{i_0} > \max\left\{ 2n_2\left(\frac{R_{\max}}{\delta}-1\right), \frac{12KV_{\max}}{\epsilon\delta^2} \right\} \tag{35}$$

$$n \triangleq n_1 + n_2. \tag{36}$$

That is, we first find the n_2 value by (34). Then we choose n_1 value to be one of the entries of $\{t_i\}$ that is strictly larger than the right-hand side of (35). Finally, we define $n = n_1 + n_2$, which will be used as the block coding length in our construction. In the following we describe how to construct a new block coding scheme of length n from the given sequential coding scheme.

Given any $(nR_1^\delta, \dots, nR_K^\delta)$ packets that need to be transmitted via a block coding scheme, we first use these packets to "simulate" K i.i.d. Poisson packet arrival processes with arrival rates $(R_1, \cdots, R_K)$ for n_1 time slots. More specifically, in the beginning of transmission all nR_k^δ packets are marked as "not-yet-arrived". Then for each time slot t, we randomly generate the $A_k(t)$ value according to a Poisson distribution with parameter R_k, and we change the labels of $A_k(t)$ packets in the original nR_k^δ packets from "not-yet-arrived" to "arrived".

Recall that $M_k(n_1)$ denote the cumulative number of packet arrivals during the first n_1 time slots, which is a random variable with average $\mathsf{E}\{M_k(n_1)\} = n_1 R_k$. If the cumulative number of packet arrivals $M_k(n_1) < nR_k^\delta$, then it means that in the end of time n_1 some packets are still labeled as "not-yet-arrived". In this case we declare encoding/decoding failure and the block coding scheme terminates. If $M_k(n_1) \geq nR_k^\delta$, then the labels of all packets have eventually been changed to "arrived". For the last $M_k(n_1) - nR_k^\delta$ packets, i.e., after all nR_k^δ packets have been labeled "arrived", we can simply insert new all-zero packets (the dummy packets) into the system.

Since the original nR_k^δ packets now "arrive" at source s as a Poisson random process with rate R_k, we can apply the sequential coding scheme during the first n_1 time slots. At the end of time n_1, the total number of packets in the memory of the source is $|Q(n_1)|$. Then for the remaining n_2 time slots we "broadcast" the remaining packets in the queue $Q(n_1)$ to all K destinations. Namely, each destination d_k, $k \in [K]$, would like to know all the content of $Q(n_1)$ by the end of time $n = n_1 + n_2$ (during time $[n_1+1, n_1+n_2]$). To achieve this task, we transmit the $|Q(n_1)|$ packets one-by-one to d_1 first, and then transmit all $|Q(n_1)|$ packets one-by-one to d_2, so on and so forth until all d_k receive all the $|Q(n_1)|$ packets.[10]

If source s fails to deliver all $|Q(n_1)|$ packets to all K destinations during time $[n_1+1, n_1+n_2]$, we declare encoding/decoding failure and the block coding scheme terminates. Note that since the sequential coding scheme is decodable, see (9) and (10) in Section II, after delivering all packets in $Q(n_1)$ to all K destinations, each d_k can then decode its desired nR_k^δ packets. The description of the block coding scheme is thus complete.

Analyzing the above block code, the decoding error can only follow from the following three events: **Error Type 1:** When simulating the Poisson arrival processes in the first n_1 time slots, the actual number of random arrivals of user-k is smaller than nR_k^δ. That is, some of the to-be-transmitted packets never "arrive" at the source and thus cannot be delivered by our new scheme. **Error Type 2:** $|Q(n_1)| > \frac{3D}{\epsilon}$. Namely, at the end of time n_1, there are too many packets in the queue at the source. **Error Type 3:** $|Q(n_1)| \leq \frac{3D}{\epsilon}$ but there exists a destination d_k such that not all $|Q(n_1)|$ packets are successfully delivered to d_k during the n_2 time slots. Applying the union bound to the error probability, we have

$$\begin{aligned}
&\Pr\left(\bigcup_{k\in[K]} \{\hat{\mathbf{X}}_k \neq \mathbf{X}_k\}\right) \\
&\leq \sum_{k\in[K]} \Pr\left(M_k(n_1) < nR_k^\delta\right) + \Pr\left(|Q(n_1)| > \frac{3D}{\epsilon}\right) + \\
&\sum_{k\in[K]} \Pr\left(|Q(n_1)| \leq \frac{3D}{\epsilon}, \text{not all } |Q(n_1)| \text{ delivered to } d_k\right).
\end{aligned} \tag{37}$$

We now upper bound each of the three probability terms individually. We upper bound the first term by the Chebyshev inequality. Specifically, $M_k(n_1)$ is a Poisson random variable with mean $n_1 R_k$ and variance $n_1 V_k$. Comparing the mean of $M_k(n_1)$ versus the original number of packets nR_k^δ, we have

$$\begin{aligned}
\mathsf{E}\{M_k(n_1)\} - nR_k^\delta &= n_1 R_k - nR_k^\delta \\
&= n_1 R_k - (n_1+n_2)(R_k - \delta) \\
&= n_1\delta + n_2\delta - n_2 R_k \\
&\geq n_1\delta + n_2\delta - n_2 R_{\max} \\
&> n_1\delta + n_2\delta - \left(\frac{n_1}{2} + n_2\right)\delta = \frac{n_1\delta}{2} > 0.
\end{aligned} \tag{38}$$

where the inequality in (38) follows from the following argument. By (35) we have $n_1 > 2n_2(\frac{R_{\max}}{\delta} - 1)$, which implies $(\frac{n_1}{2} + n_2)\delta > n_2 R_{\max}$ and thus (38). Then each summand of the first summation term in (37) is upper bounded by

$$\Pr\left(M_k(n_1) < nR_k^\delta\right)$$
$$\leq \Pr\left(\left|\frac{M_k(n_1)}{n_1} - R_k\right| > \frac{\delta}{2}\right) \tag{39}$$
$$\leq \frac{V_k/n_1}{\delta^2/4} \tag{40}$$
$$\leq \frac{4V_{\max}}{n_1\delta^2} \leq \frac{\epsilon}{3K}. \tag{41}$$

where (39) follows from (38); (40) follows from the Chebyshev inequality; and (41) follows from the definition of n_1 in (35). The summation in the first term is thus upper bounded by $\frac{\epsilon}{3}$.

[10]Herein we use *uncoded broadcast* during time $[n_1+1, n_1+n_2]$. We can also use *network-coded-broadcast*. Both will serve the same purpose in our proof and we choose the former for its simplicity.

The second term in (37) is bounded by Markov's inequality. That is,

$$\begin{aligned}\Pr\left(|Q(n_1)| > \frac{3D}{\epsilon}\right) &\\ \leq \frac{\epsilon}{3D}\mathsf{E}\{|Q(n_1)|\} &\\ \leq \frac{\epsilon D}{3D} = \frac{\epsilon}{3}. & \qquad (42)\end{aligned}$$

where (42) follows from the definition of choosing $n_1 = t_{i_0}$ in (35), which ensures $\mathsf{E}\{|Q(n_1)|\} \leq D$.

The third term in (37) is bounded by the Chernoff bound. By (34) we have

$$\frac{3D}{\epsilon} \leq \frac{n_2 p_{\min}}{2K} \leq \frac{n_2 p_k}{2K}. \qquad (43)$$

For simplicity, we divide n_2 time slots into K sub-intervals. Each sub-interval has length n_2/K and is responsible for delivering $|Q(n_1)|$ packets to user d_k for some $k \in [K]$. If we denote Γ_k as the number of d_k's received packets during the $\frac{n_2}{K}$ time slots, then Γ_k is a Binomial random variable with parameter $(\frac{n_2}{K}, p_k)$ and the event "not all the $|Q(n_1)|$ packets are delivered successfully to d_k by the end of time n" is equivalent to $\Gamma_k < |Q(n_1)|$, i.e.,

$$\begin{aligned}&\Pr\Bigg(|Q(n_1)| \leq \frac{3D}{\epsilon},\\ &\qquad \text{not all } |Q(n_1)| \text{ packets are delivered to } d_k\Bigg)\\ &\leq \Pr\left(\Gamma_k < |Q(n_1)|,\ |Q(n_1)| \leq \frac{3D}{\epsilon}\right)\\ &\leq \Pr\left(\Gamma_k \leq \left(1-\frac{1}{2}\right)\frac{n_2 p_k}{K}\right) \qquad (44)\\ &\leq \exp\left(-\frac{1}{4}\cdot\frac{n_2 p_k}{2K}\right) \qquad (45)\\ &\leq \exp\left(-\frac{n_2 p_{\min}}{8K}\right) \leq \frac{\epsilon}{3K}. \qquad (46)\end{aligned}$$

where (44) follows from (43), (45) follows from the Chernoff bound, and (46) follows from the definition of n_2 in (34).

Hence (37), (41), (42), and (46) yield

$$\Pr\left(\bigcup_{k\in[K]}\{\hat{\mathbf{X}}_k \neq \mathbf{X}_k\}\right) \leq \epsilon. \qquad (47)$$

This shows that the above block coding scheme achieves rate $\vec{R}^{\delta}$. The proof is thus complete.

Appendix B
On The Optimality of Proposition 3

In this appendix, we prove that Proposition 3 is optimal in many scenarios while being suboptimal in some other scenarios.

A. Proof of Corollary 1

Comparing Propositions 1 and 3, "the stability region matching the capacity region" is equivalent to the following statement:

> Given any rate $\vec{R}$ satisfying the outer bound (5) in Proposition 1, there exists non-negative variables $\{x_T : T \in 2^{[K]}\backslash\{\emptyset\}\}$ satisfying Conditions 1 and 2 in Proposition 3, provided we change the strict inequality in (17) to $\leq$.

In the sequel, we prove that the above statement holds for the following cases.

Case 1: $K = 3$. Consider any fixed rate (R_1, R_2, R_3) that is within the outer bound described in Proposition 1. For any three distinct indices $i, j, k \in \{1, 2, 3\}$ (no two being equal), define a function

$$\tau(i,j,k) \triangleq \frac{R_i}{p_{\cup\{i,j,k\}}} + \frac{R_j}{p_{\cup\{j,k\}}} + \frac{R_k}{p_k}, \qquad (48)$$

Since (R_1, R_2, R_3) satisfies Proposition 1, we have $\max_{i,j,k}\tau(i,j,k) \leq 1$.

We now construct the variables $\{x_T : T \in 2^{[3]}\backslash\{\emptyset\}\}$ by

$$\begin{aligned}x_{\{2\}} =& \frac{R_2}{p_{\cup\{1,2,3\}}}, \qquad (49)\\ x_{\{1,2\}} =& -\frac{R_1+R_2}{p_{\cup\{1,2,3\}}} - \frac{R_3}{p_3} + \max\{\tau(1,2,3), \tau(2,1,3)\}, \qquad (50)\\ x_{\{1,2,3\}} =& \frac{R_1+R_2+R_3}{p_{\cup\{1,2,3\}}} + \frac{R_1}{p_1} + \frac{R_2}{p_2} + \frac{R_3}{p_3} + \max_{\forall i,j,k}\tau(i,j,k)\\ &- \max\{\tau(1,2,3), \tau(2,1,3)\}\\ &- \max\{\tau(1,3,2), \tau(3,1,2)\}\\ &- \max\{\tau(3,2,1), \tau(2,3,1)\}. \qquad (51)\end{aligned}$$

Variables $x_{\{1\}}$ (resp. $x_{\{3\}}$) is defined in a symmetric way as $x_{\{2\}}$ by swapping the user indices 2 and 1 (resp. 2 and 3) in (49). $x_{\{1,3\}}$ (resp. $x_{\{2,3\}}$) is defined in a symmetric way as $x_{\{1,2\}}$ by swapping the user indices 2 and 3 (resp. 1 and 3) in (50).

We now prove that the x_T variables in the above construction are non-negative. It is clear from (49) that $x_{\{2\}} \geq 0$. By symmetry we also have $x_{\{1\}} \geq 0$ and $x_{\{3\}} \geq 0$.

We also have

$$\begin{aligned}x_{\{1,2\}} \geq& -\frac{R_1+R_2}{p_{\cup\{1,2,3\}}} - \frac{R_3}{p_3} + \tau(2,1,3) \qquad (52)\\ =& R_1\left(\frac{1}{p_{\cup\{1,3\}}} - \frac{1}{p_{\cup\{1,2,3\}}}\right) \geq 0 \qquad (53)\end{aligned}$$

where (52) follows from (50) and the equality of (53) follows from (48). Eq. (53) is non-negative since by definition $p_{\cup\{1,3\}} \leq p_{\cup\{1,2,3\}}$. By symmetry, $x_{\{1,3\}}$ and $x_{\{2,3\}}$ are also non-negative.

To show that $x_{\{1,2,3\}} \geq 0$, without loss of generality, we assume $\max_{i,j,k}\tau(i,j,k) = \tau(1,2,3)$ and discuss the following cases.

(i) Consider $\tau(1,3,2) \geq \tau(3,1,2)$ and $\tau(3,2,1) \geq \tau(2,3,1)$. Since $\tau(1,2,3)$ is assumed to be the largest, the inequality $\tau(1,2,3) \geq \tau(1,3,2)$ also yields

$$R_3\left(\frac{1}{p_3} - \frac{1}{p_{\cup\{2,3\}}}\right) \geq R_2\left(\frac{1}{p_2} - \frac{1}{p_{\cup\{2,3\}}}\right).$$

Therefore (51) becomes

$$
\begin{aligned}
x_{\{1,2,3\}} &= \frac{R_3}{p_3} - \frac{R_3}{p_{\cup\{2,3\}}} - \frac{R_2}{p_{\cup\{1,2\}}} + \frac{R_2}{p_{\cup\{1,2,3\}}} \\
&\geq R_2\left(\frac{1}{p_2} - \frac{1}{p_{\cup\{2,3\}}} - \frac{1}{p_{\cup\{1,2\}}} + \frac{1}{p_{\cup\{1,2,3\}}}\right) \geq 0,
\end{aligned}
$$

where the last nonnegative inequality follows from [5, Lemma 5].

(ii) Consider $\tau(1,3,2) \geq \tau(3,1,2)$ and $\tau(3,2,1) < \tau(2,3,1)$. Eq. (51) becomes

$$
x_{\{1,2,3\}} = R_3\left(\frac{1}{p_3} - \frac{1}{p_{\cup\{2,3\}}} - \frac{1}{p_{\cup\{1,3\}}} + \frac{1}{p_{\cup\{1,2,3\}}}\right) \geq 0.
$$

(iii) Consider $\tau(1,3,2) < \tau(3,1,2)$ and $\tau(3,2,1) \geq \tau(2,3,1)$. Since $\tau(1,2,3)$ is the largest, $\tau(1,2,3) \geq \tau(3,1,2)$ and it implies

$$
\frac{R_1}{p_{\cup\{1,2,3\}}} + \frac{R_2}{p_{\cup\{2,3\}}} + \frac{R_3}{p_3} \geq \frac{R_3}{p_{\cup\{1,2,3\}}} + \frac{R_1}{p_{\cup\{1,2\}}} + \frac{R_2}{p_2}.
$$

Therefore (51) becomes

$$
\begin{aligned}
x_{\{1,2,3\}} =& \frac{R_1 + R_2 - R_3}{p_{\cup\{1,2,3\}}} + \frac{R_3}{p_3} - \frac{R_1}{p_{\cup\{1,2\}}} - \frac{R_2}{p_{\cup\{1,2\}}} \\
\geq& R_2\left(\frac{1}{p_2} - \frac{1}{p_{\cup\{2,3\}}} - \frac{1}{p_{\cup\{1,2\}}} + \frac{1}{p_{\cup\{1,2,3\}}}\right) \geq 0.
\end{aligned}
$$

(iv) Consider $\tau(1,3,2) < \tau(3,1,2)$ and $\tau(3,2,1) < \tau(2,3,1)$, which implies

$$
\begin{aligned}
R_1\left(\frac{1}{p_{\cup\{1,2\}}} - \frac{1}{p_{\cup\{1,2,3\}}}\right) &> R_3\left(\frac{1}{p_{\cup\{2,3\}}} - \frac{1}{p_{\cup\{1,2,3\}}}\right), \\
R_3\left(\frac{1}{p_{\cup\{1,3\}}} - \frac{1}{p_{\cup\{1,2,3\}}}\right) &> R_2\left(\frac{1}{p_{\cup\{1,2\}}} - \frac{1}{p_{\cup\{1,2,3\}}}\right).
\end{aligned}
$$

The product of the above two inequalities implies

$$
R_1\left(\frac{1}{p_{\cup\{1,3\}}} - \frac{1}{p_{\cup\{1,2,3\}}}\right) > R_2\left(\frac{1}{p_{\cup\{2,3\}}} - \frac{1}{p_{\cup\{1,2,3\}}}\right).
$$

However since $\tau(1,2,3) \geq \tau(2,1,3)$, we also have

$$
R_2\left(\frac{1}{p_{\cup\{2,3\}}} - \frac{1}{p_{\cup\{1,2,3\}}}\right) \geq R_1\left(\frac{1}{p_{\cup\{1,3\}}} - \frac{1}{p_{\cup\{1,2,3\}}}\right).
$$

This contradiction shows that this case is impossible.

Thus far we have proven the non-negativity of the $\{x_T\}$ variables. We now prove that those $\{x_T\}$ satisfy (17) and (18). By the definitions in (49) to (51) one can directly verify that

$$
\sum_{\forall T \in 2^{[3]} \backslash \{\emptyset\}} x_T = \max_{i,j,k} \tau(i,j,k) \tag{54}
$$

which is no larger than one, see the discussion right after (48). The time-sharing condition (17) thus holds.

To verify (18), we consider only the case of $k = 2$. The cases of $k = 1, 3$ follow by symmetry. There are five possible proper cuts in Condition 2, namely $\mathcal{C}_i^{(2)}$, $i \in \{1,2,3,4,5\}$ in (12) to (16). In the following, we examine (18) for each of these five cuts separately, also see (20) to (24).

For $\mathcal{C}_1^{(2)} = \{\mathrm{vn}_{\mathrm{grnd}}^{(2)}\}$, inequality (20) becomes

$$
\begin{aligned}
&\left(x_{\{2\}} + x_{\{1,2\}} + x_{\{2,3\}} + x_{\{1,2,3\}}\right) \cdot p_2 \\
&= \left(\frac{R_2}{p_2} + \max_{\forall i,j,k} \tau(i,j,k) - \max\{\tau(1,3,2), \tau(3,1,2)\}\right) \cdot p_2 \\
&\geq R_2.
\end{aligned} \tag{55}
$$

For $\mathcal{C}_2^{(2)} = \{\mathrm{vn}_{\mathrm{grnd}}^{(2)}, \mathrm{vn}_{\{1,3\}}^{(2)}\}$, inequality (21) becomes

$$
\begin{aligned}
&x_{\{2\}}\left(p_2 + p_{\{1,3\}\overline{2}}\right) + x_{\{1,2\}}\left(p_2 + p_{3\overline{2}}\right) + x_{\{2,3\}}\left(p_2 + p_{1\overline{2}}\right) \\
&= x_{\{2\}} \cdot \left(p_{\cup\{1,2\}} + p_{\cup\{2,3\}} - p_{\cup\{1,2,3\}}\right) \\
&\quad + x_{\{1,2\}} \cdot p_{\cup\{2,3\}} + x_{\{2,3\}} \cdot p_{\cup\{1,2\}} \\
&= \left(x_{\{2\}} + x_{\{1,2\}}\right) \cdot p_{\cup\{2,3\}} + \left(x_{\{2\}} + x_{\{2,3\}}\right) \cdot p_{\cup\{1,2\}} - R_2 \\
&\geq 2R_2 - R_2 = R_2.
\end{aligned} \tag{56}
$$

For $\mathcal{C}_3^{(2)} = \{\mathrm{vn}_{\mathrm{grnd}}^{(2)}, \mathrm{vn}_{\{1\}}^{(2)}, \mathrm{vn}_{\{1,3\}}^{(2)}\}$, inequality (22) becomes

$$
\begin{aligned}
&x_{\{2\}}\left(p_2 + p_{1\overline{\{2,3\}}} + p_{\{1,3\}\overline{2}}\right) + x_{\{2,3\}}\left(p_2 + p_{1\overline{2}}\right) \\
&= \left(x_{\{2\}} + x_{\{2,3\}}\right) \cdot p_{\cup\{1,2\}} \\
&\geq \left(\tau(3,2,1) - \frac{R_3}{p_{\cup\{1,2,3\}}} - \frac{R_1}{p_1}\right) \cdot p_{\cup\{1,2\}} = R_2.
\end{aligned} \tag{57}
$$

The case of $\mathcal{C}_4^{(2)} = \{\mathrm{vn}_{\mathrm{grnd}}^{(2)}, \mathrm{vn}_{\{3\}}^{(2)}, \mathrm{vn}_{\{1,3\}}^{(2)}\}$ is symmetric to the case of $\mathcal{C}_3^{(2)} = \{\mathrm{vn}_{\mathrm{grnd}}^{(2)}, \mathrm{vn}_{\{1\}}^{(2)}, \mathrm{vn}_{\{1,3\}}^{(2)}\}$. For $\mathcal{C}_5^{(2)} = \{\mathrm{vn}_{\mathrm{grnd}}^{(2)}, \mathrm{vn}_{\{1\}}^{(2)}, \mathrm{vn}_{\{3\}}^{(2)}, \mathrm{vn}_{\{1,3\}}^{(2)}\}$, inequality (24) becomes

$$
x_{\{2\}} \cdot \left(p_2 + p_{1\overline{\{2,3\}}} + p_{3\overline{\{1,2\}}} + p_{\{1,3\}\overline{2}}\right) = R_2, \tag{58}
$$

where (58) follows from (49). Our construction of $\{x_T\}$ thus satisfies (18). The proof of Case 1 is complete.

Case 2: The channel is spatially independent and the rate-vector is one-sided fair. Without loss of generality, we assume the marginal success probability p_k are listed from the smallest to the largest. That is, $0 < p_1 \leq p_2 \leq \cdots \leq p_K$. A rate vector $\vec{R}$ is one-sided fair if $R_i \cdot (1 - p_i) \geq R_j \cdot (1 - p_j)$ for all $i < j$. Detailed discussion on one-sided fairness can be found in [5].

Consider any spatially independent channel and any one-sided fair rate vector $(R_1, \cdots, R_K)$ satisfying the outer bound (5). For any k and $S \in 2^{[K] \backslash \{k\}}$, define

$$
u_S^{(k)} \triangleq R_k \cdot \left(\sum_{\forall S_1 : ([K] \backslash S) \subseteq S_1 \subseteq [K]} \frac{(-1)^{|S_1| - (K - |S|)}}{p_{\cup S_1}}\right). \tag{59}
$$

We now construct the desired variables $\{x_T\}$ by choosing $x_T = u_{T \backslash k^*(T)}^{(k^*(T))}$, where $k^*(T) = \min\{i : i \in T\}$.

In the following we prove that the above $\{x_T\}$ are non-negative and satisfy Condition 1 in (17) and Condition 2 (18) simultaneously. Specifically, [5] proves that $u_S^{(k)}$ is non-negative for all $k \in [K]$ and $S \in 2^{[K] \backslash \{k\}}$. Since $u_{T \backslash k^*(T)}^{(k^*(T))} \geq 0$, we thus have $x_T \geq 0$ for all T.

To prove Condition 1 in (17), we have

$$
\sum_{T \in 2^{[K]} \backslash \{\emptyset\}} u_{T \backslash k^*(T)}^{(k^*(T))} = \sum_{k \in [K]} \left(\sum_{S \subseteq \{k+1, \cdots, K\}} u_S^{(k)}\right) \tag{60}
$$

$$
= \sum_{k \in [K]} \frac{R_k}{p_{\cup[k]}} \leq 1, \tag{61}
$$

where the (61) follows the Möbius inversion [30] of partially ordered sets with two functions $u_S^{(k)}$ in (59) and $v_T^{(k)} \triangleq \frac{R_k}{p_{\cup T}}$.

To prove Condition 2 in (18), we use the result of Lemma 9 in Appendix F and show that there exist nonnegative variables $\vec{y}$ defined in (85) such that (86), (87), and (88) are satisfied. Given the variables $\{x_T\}$ above, let the $\vec{y}$ variables for any fixed $k \in [K]$ be

$$y_{k;S_I \to d_k} = u_{S_I}^{(k)} \cdot p_k, \quad \forall S_I \in 2^{[K]\backslash\{k\}}, \tag{62}$$

$$y_{k;S_I \to S_O}^{[S_X]} = \begin{cases} u_{S_I}^{(k)} \cdot p_{S_X \overline{[K]\backslash(S_I \cup S_X)}}, & S_O = S_I \cup S_X, \\ 0, & \text{otherwise}, \end{cases}$$
$$\forall S_I \in 2^{[K]\backslash\{k\}}, S_X \in 2^{[K]\backslash(S_I \cup \{k\})}. \tag{63}$$

Since $u_S^{(k)}$ are nonnegative, we have $\{y_{k;S_I \to d_k}\}$ and $\{y_{k;S_I \to S_O}^{[S_X]}\}$ are also nonnegative. Recall that we define $x_T = u_{T\backslash k^*(T)}^{(k^*(T))}$. Since $\vec{R}$ is one-sided fair, by [5, Lemma 5] we have $u_S^{(k)} \leq x_{S\cup\{k\}}$ for all k and S. This directly implies that $\vec{y}$ satisfies (87) and (88). To show (86) holds, for a given $k \in [K]$ and $S \in 2^{[K]\backslash\{k\}}$ the right-hand side of (86) is

$$u_S^{(k)} \cdot \left(p_k + \sum_{S_X \in 2^{[K]\backslash(S\cup\{k\})}} p_{S_X \overline{[K]\backslash(S \cup S_X)}} \right) = u_S^{(k)} \cdot p_{\overline{S}}. \tag{64}$$

We then consider the left-hand side of (86). If $S = \emptyset$, we have $u_\emptyset^{(k)} = \frac{R_k}{p_{\cup[K]}} = \frac{R_k}{1-p_{\overline{[K]}}}$ and $p_{\overline{\emptyset}} = 1$. The left-hand side of (86) then becomes

$$R_k + u_\emptyset^{(k)} \cdot p_{\overline{[K]}} = \frac{R_k}{1-p_{\overline{[K]}}} = u_\emptyset^{(k)} \cdot p_{\overline{\emptyset}}. \tag{65}$$

On the other hand, if $S \in 2^{[K]\backslash\{k\}}\backslash\{\emptyset\}$, the left-hand side of (86) is given by

$$\sum_{S_I \in 2^{[S]}} u_{S_I}^{(k)} \cdot p_{(S\backslash S_I)\overline{[K]\backslash S}} = u_S^{(k)} \cdot p_{\overline{S}}. \tag{66}$$

Therefore, (86) always holds with equality. The proof of Case 2 is complete.

Case 3: Symmetric channels. A 1-to-K broadcast PEC is said to be symmetric if the success probability $p_{S_1\overline{[K]\backslash S_1}} = p_{S_2\overline{[K]\backslash S_2}}$ for any $S_1, S_2 \in 2^{[K]}$ satisfying $|S_1| = |S_2|$. Consider any symmetric channel and any rate vector $\vec{R} = (R_1, \ldots, R_K)$ satisfying the outer bound (5). Without loss of generality, we also assume $R_1 \geq R_2 \geq \cdots \geq R_K$, which is always possible after relabeling the destinations. We then choose non-negative $x_T = u_{T\backslash k^*(T)}^{(k^*(T))}$, where $k^*(T) = \min\{i : i \in T\}$. Since the choice of $\{x_T\}$ is the same as that in Case 2, $\{x_T\}$ satisfying Condition 1 as in (61).

To show that such $\{x_T\}$ also satisfies Condition 2, we choose the same non-negative $\vec{y}$ as in (62) and (63) and we will show that $\vec{y}$ also satisfies (86), (87), and (88) in Lemma 9. The reason is as follows. Due to the symmetric channel and the sorted R_k in descending order, it is obvious that $x_T = u_{T\backslash k^*(T)}^{(k^*(T))} = \max_{k\in T}\{u_{T\backslash\{k\}}^{(k)}\}$. Then by the same reasons as discussed in Case 2, inequalities (87) and (88) is satisfied. Then following the same reason of (64), (65), and (66), $\vec{y}$ also satisfies (86). The proof of Case 3 is complete.

B. Proof of Example 1

Suppose the channel is spatially independent with marginal success probability being $\vec{p} = (p_1, p_2, p_3, p_4) = (\frac{1}{3}, \frac{2}{5}, \frac{1}{2}, \frac{4}{7})$ and rate vector being $\vec{R} = (R_1, R_2, R_3, R_4) = (\frac{96}{1193}, \frac{672}{5\cdot 1193}, \frac{288}{1193}, \frac{1952}{7\cdot 1193})$. One can verify that all $4! = 24$ inequalities of the outer bound in Proposition 1 are satisfied. Specifically, the following four inequalities are satisfied with equality

$$\begin{cases} \frac{R_1}{p_{\cup\{1,2,3,4\}}} + \frac{R_4}{p_{\cup\{2,3,4\}}} + \frac{R_2}{p_{\cup\{2,3\}}} + \frac{R_3}{p_3} = 1, \\ \frac{R_4}{p_{\cup\{1,2,3,4\}}} + \frac{R_1}{p_{\cup\{1,2,3\}}} + \frac{R_2}{p_{\cup\{2,3\}}} + \frac{R_3}{p_3} = 1, \\ \frac{R_4}{p_{\cup\{1,2,3,4\}}} + \frac{R_2}{p_{\cup\{1,2,3\}}} + \frac{R_1}{p_{\cup\{1,3\}}} + \frac{R_3}{p_3} = 1, \\ \frac{R_4}{p_{\cup\{1,2,3,4\}}} + \frac{R_2}{p_{\cup\{1,2,3\}}} + \frac{R_3}{p_{\cup\{1,3\}}} + \frac{R_1}{p_1} = 1, \end{cases} \tag{67}$$

and the remaining 20 inequalities are satisfied with strict inequality.

We now turn our attention to Proposition 3, the stability region. Obviously, this channel model is asymmetric. The rate vector is also not one-sided fair since $R_2 \cdot (1-p_2) = \frac{2016}{25\cdot 1193} < \frac{144}{1193} = R_3 \cdot (1-p_3)$, also see the discussion in the end of Appendix B-A. We choose the x_T variables as follows.

$$x_{\{1\}} = \frac{105}{1193}, \; x_{\{2\}} = \frac{147}{1193}, \; x_{\{3\}} = \frac{315}{1193}, \; x_{\{4\}} = \frac{305}{1193},$$
$$x_{\{1,2\}} = \frac{441}{61\cdot 1193}, \; x_{\{1,3\}} = \frac{945}{61\cdot 1193}, \; x_{\{1,4\}} = \frac{15}{1193},$$
$$x_{\{2,3\}} = \frac{21}{1193}, \; x_{\{2,4\}} = \frac{21}{1193}, \; x_{\{3,4\}} = \frac{45}{61\cdot 1193},$$
$$x_{\{1,2,3\}} = \frac{10101}{11\cdot 61\cdot 1193}, x_{\{1,2,4\}} = \frac{1023}{61\cdot 1193}, x_{\{2,3,4\}} = \frac{51}{1193},$$
$$x_{\{1,3,4\}} = \frac{15345}{7\cdot 61\cdot 1193}, \; x_{\{1,2,3,4\}} = \frac{364101}{7\cdot 11\cdot 61\cdot 1193}.$$

Clearly these x_T are non-negative and their total sum is $\frac{5603521}{5603521} = 1$. The cut condition in (18) can be verified by simple computer programs. The proof of Example 1 is thus complete.

C. An Example of Unachievable Rates for $K = 4$

Consider the case of $K = 4$, rate vector $\vec{R} = (0.173284, 0.146994, 0.174068, 0.127000)$. We introduce the following equivalent notation for the PEC parameters $\{p_{S\overline{[K]\backslash S}}\}$. That is,

$$p_{b_1b_2b_3b_4} = p_{\{i:b_i=1\}\overline{\{j:b_j=0\}}}. \tag{68}$$

That is, each b_i indicates whether the i-th user has received the packet. For example, $p_{1011} = p_{\{1,3,4\}\overline{2}}$ is the probability that d_1, d_3, and d_4 receive the packet but not d_2. Using this new notation, we set the channel parameters to be

$$p_{0000} = .1085, \; p_{0001} = .0132, \; p_{0010} = .1072, \; p_{0011} = .1040,$$
$$p_{0100} = .0846, \; p_{0101} = .0950, \; p_{0110} = .0563, \; p_{0111} = .0059,$$
$$p_{1000} = .1043, \; p_{1001} = .1099, \; p_{1010} = .0065, \; p_{1011} = .0344,$$
$$p_{1100} = .0023, \; p_{1101} = .0693, \; p_{1110} = .0033, \; p_{1111} = .0953,$$

such that $\sum_{b_1,b_2,b_3,b_4\in\{0,1\}} p_{b_1b_2b_3b_4} = 1$.

A simple computer program can be used to verify that these choices of $\vec{R}$ and $p_{b_1b_2b_3b_4}$ satisfy all 4! inequalities of the outer bound in Proposition 1. If we define $\vec{R}_\alpha = \alpha \cdot \vec{R}$, we can use an LP solver to find the largest possible α such that $\vec{R}_\alpha$ satisfies the inner bound in Proposition 3. In this example, the largest $\alpha = 99.86\%$. This shows that the inner bound in Proposition 3 does not always meet the outer bound in Proposition 1.

Appendix C
Proof of Lemmas 1 and 2

We first prove Lemma 1. Assume any arbitrary but fixed virtual node scheduling frequency $\{x_T : T \in 2^{[K]}\backslash\{\emptyset\}\}$. The cut value of a node-cut $\mathcal{C}^{(k)}$ (not necessarily a proper cut) is the summation of the capacity of edges of the following two types. Type 1: Edges from a virtual node $\mathrm{vn}_S^{(k)} \notin \mathcal{C}^{(k)}$ to an auxiliary node $S_X \in 2^{[K]\backslash(S\cup\{k\})}$ that has an outgoing infinite-capacity edge ending in a $\mathrm{vn}_{\tilde{S}}^{(k)} \in \mathcal{C}^{(k)}$; Type 2: Edges from a virtual node $\mathrm{vn}_S^{(k)} \notin \mathcal{C}^{(k)}$ to the sink node $\mathrm{vn}_{\text{grnd}}^{(k)}$.

For any given $\mathrm{vn}_S^{(k)} \notin \mathcal{C}^{(k)}$, the total edge capacity of Type-1 edges are:

$$x_{S\cup\{k\}}\left(\sum_{\substack{\forall S_X:\\ S_X\in\mathcal{F}(k,S,\mathcal{C}^{(k)})}} p_{S_X\overline{[K]\backslash(S_X\cup S)}}\right) \tag{69}$$

where $p_{S_X\overline{[K]\backslash(S_X\cup S)}}$ is the edge capacity entering auxiliary node S_X; and $\mathcal{F}(k,S,\mathcal{C}^{(k)})$ defined in (19) consists of all auxiliary nodes $\{S_X \in 2^{[K]\backslash(S\cup\{k\})}\}$ that have an infinite-capacity edge ending in some $\mathrm{vn}_{\tilde{S}}^{(k)} \in \mathcal{C}^{(k)}$.

For any given $\mathrm{vn}_S^{(k)} \notin \mathcal{C}^{(k)}$, there is only one Type-2 edge and its capacity is

$$x_{S\cup\{k\}}p_k. \tag{70}$$

By adding (69) and (70) together and then by summing over all $\mathrm{vn}_S^{(k)} \notin \mathcal{C}^{(k)}$, the left-hand side of (18) computes the cut value of $\mathcal{C}^{(k)}$. The proof of Lemma 1 is complete.

We now prove Lemma 2. Since any proper-cut is also a cut, [Statement 1] implies [Statement 2]. We now prove that [Statement 2] implies [Statement 1].

Suppose [Statement 2] holds. For any virtual node $\mathrm{vn}_S^{(k)}$, define $\mathcal{V}(\mathrm{vn}_S^{(k)}) \triangleq \{\mathrm{vn}_{S'}^{(k)} : S' \supseteq S\}$. By definition, we always have $\mathrm{vn}_S^{(k)} \in \mathcal{V}(\mathrm{vn}_S^{(k)})$. Then for any arbitrary cut $\partial^{(k)}$, define $\mathcal{C}^{(k)} \triangleq \left(\bigcup_{\mathrm{vn}_S^{(k)}\in\partial^{(k)}} \mathcal{V}(\mathrm{vn}_S^{(k)})\right)\cup \mathrm{vn}_{\text{grnd}}^{(k)}$. It is clear that $\partial^{(k)} \subseteq \mathcal{C}^{(k)}$ and it is also true by Definition 6 that the above $\mathcal{C}^{(k)}$ is a proper cut.

From the definition of $\mathcal{F}$ in (19), it is clear that $\mathcal{F}(k,S,\partial^{(k)}) \subseteq \mathcal{F}(k,S,\mathcal{C}^{(k)})$ since $\partial^{(k)} \subseteq \mathcal{C}^{(k)}$. Furthermore, by the construction of $\mathcal{C}^{(k)}$ any $\mathrm{vn}_{\tilde{S}}^{(k)} \in \mathcal{C}^{(k)}$ corresponds to a $\mathrm{vn}_S^{(k)} \in \partial^{(k)}$ satisfying $S \subseteq \tilde{S}$. Therefore (19) implies that we actually have $\mathcal{F}(k,S,\partial^{(k)}) = \mathcal{F}(k,S,\mathcal{C}^{(k)})$. If we use $\mathrm{cv}(\partial^{(k)})$ and $\mathrm{cv}(\mathcal{C}^{(k)})$ to denote the cut values of $\partial^{(k)}$ and $\mathcal{C}^{(k)}$, respectively, we then have

$$\mathrm{cv}\left(\partial^{(k)}\right) = \sum_{\substack{S:S\in 2^{[K]\backslash\{k\}},\\ \mathrm{vn}_S^{(k)}\notin\partial^{(k)}}} x_{S\cup\{k\}}\cdot\left(p_k + \sum_{\substack{\forall S_X:\\ S_X\in\mathcal{F}(k,S,\partial^{(k)})}} p_{S_X\overline{[K]\backslash(S_X\cup S)}}\right)$$

$$\geq \sum_{\substack{S:S\in 2^{[K]\backslash\{k\}},\\ \mathrm{vn}_S^{(k)}\notin\mathcal{C}^{(k)}}} x_{S\cup\{k\}}\cdot\left(p_k + \sum_{\substack{\forall S_X:\\ S_X\in\mathcal{F}(k,S,\partial^{(k)})}} p_{S_X\overline{[K]\backslash(S_X\cup S)}}\right) \tag{71}$$

$$= \sum_{\substack{S:S\in 2^{[K]\backslash\{k\}},\\ \mathrm{vn}_S^{(k)}\notin\mathcal{C}^{(k)}}} x_{S\cup\{k\}}\cdot\left(p_k + \sum_{\substack{\forall S_X:\\ S_X\in\mathcal{F}(k,S,\mathcal{C}^{(k)})}} p_{S_X\overline{[K]\backslash(S_X\cup S)}}\right) = \mathrm{cv}\left(\mathcal{C}^{(k)}\right) \tag{72}$$

$$\geq R_k \tag{73}$$

where (71) follows from changing the summation over $\mathrm{vn}_S^{(k)} \notin \partial^{(k)}$ to a smaller set $\mathrm{vn}_S^{(k)} \notin \mathcal{C}^{(k)}$; (72) follows from the fact that $\mathcal{F}(k,S,\partial^{(k)}) = \mathcal{F}(k,S,\mathcal{C}^{(k)})$; and (73) follows from [Statement 2]. Since $\mathrm{cv}(\partial^{(k)}) \geq R_k$ is now proven for any arbitrary $\partial^{(k)}$, [Statement 1] holds. The proof is thus complete.

Appendix D
A Simple Schwartz-Zippel Lemma

Lemma 5. *Consider any integer M and any finite field $\mathsf{GF}(q)$ satisfying $q \geq M \geq 1$. For any M vectors $\mathbf{b}_1,\ldots,\mathbf{b}_M$ and any M matrices $\mathbf{B}^{(1)},\ldots,\mathbf{B}^{(M)}$, if $\mathbf{b}_m$ is linearly independent of the rows of $\mathbf{B}^{(m)}$ for all $1 \leq m \leq M$, then there exists $\{\beta_m \in \mathsf{GF}(q) : m \in [M]\}$ such that the sum $\sum_{m=1}^M \beta_m \cdot \mathbf{b}_m$ is linearly independent of the rows of $\mathbf{B}^{(m)}$ for all $m = 1$ to M.*

Proof. Lemma 5 holds trivially if $M = 1$. In the following proof we assume $M \geq 2$. For any matrix $\mathbf{B}$ and any row vector $\mathbf{v}$, we use $\mathbf{v} \in \langle\mathbf{B}\rangle$ to denote that $\mathbf{v}$ belongs to the row space of $\mathbf{B}$. For any $m \in [M]$, let

$$\mathcal{D}_m \triangleq \left\{(\beta_1,\cdots,\beta_M) \in (\mathsf{GF}(q))^M : \left(\sum_{i=1}^M \beta_i\mathbf{b}_i\right) \in \langle\mathbf{B}^{(m)}\rangle\right\}.$$

Namely, $\mathcal{D}_m$ contains the $(\beta_1,\cdots,\beta_M)$ coefficients such that the resulting vector $\left(\sum_{i=1}^M \beta_i\mathbf{b}_i\right)$ belongs to the row space of $\mathbf{B}^{(m)}$. Clearly, any coefficient vector $(\beta_1,\cdots,\beta_M)$ inside $(\mathsf{GF}(q))^M\backslash\bigcup_{m=1}^M \mathcal{D}_m$ satisfies Lemma 5. As a result, we only need to prove that the set $(\mathsf{GF}(q))^M\backslash\bigcup_{m=1}^M \mathcal{D}_m$ is non-empty, or, equivalently, $\left|\bigcup_{m=1}^M \mathcal{D}_m\right| < q^M$.

Fix any m value and fix any $(\beta_1,\ldots,\beta_{m-1},\beta_{m+1},\ldots,\beta_M)$ values, we claim that there exists at most one $\beta_m \in \mathsf{GF}(q)$ such that the combined vector $(\beta_1,\ldots,\beta_M)$ is in $\mathcal{D}_m$. We prove this claim by contradiction. Suppose there are two distinct coefficients β_m and β'_m such that

$$\mathbf{w} \triangleq \left(\beta_m \cdot \mathbf{b}_m + \sum_{k\in[M],k\neq m} \beta_k \cdot \mathbf{b}_k\right) \in \langle \mathbf{B}^{(m)} \rangle$$

$$\text{and } \mathbf{w}' \triangleq \left(\beta'_m \cdot \mathbf{b}_m + \sum_{k\in[M],k\neq m} \beta_k \cdot \mathbf{b}_k\right) \in \langle \mathbf{B}^{(m)} \rangle.$$

Then the difference $\mathbf{w}-\mathbf{w}' = (\beta_m - \beta'_m)\cdot \mathbf{b}_m$ must also be in $\langle \mathbf{B}^{(m)} \rangle$. However recall that Lemma 5 assumes $\mathbf{b}_m$ is linearly independent of the rows of $\mathbf{B}^{(m)}$. We thus have $\beta_m - \beta'_m = 0$, which contradicts to the assumption that β_m and β'_m are distinct.

Since for any $M-1$ coefficients $(\beta_1,\ldots,\beta_{m-1},\beta_{m+1},\ldots,\beta_M)$, there is at most one β_m satisfying $(\beta_1,\ldots,\beta_M)\in\mathcal{D}_m$, we must have $|\mathcal{D}_m| \leq q^{M-1}$. Also note that the zero vector $(\beta_1,\ldots,\beta_M) = (0,0,\ldots,0) \in \mathcal{D}_m$ for all m. By a simple union bound argument, $\left|\bigcup_{m=1}^{M} \mathcal{D}_m\right| \leq M\cdot q^{M-1} - (M-1) < q^M$, where the last inequality holds strictly as long as $q \geq M \geq 2$. The proof is complete.

□

Appendix E
Proof of Proposition 4

We now analyze the 5-component sequential coding scheme described in Section IV. Since some quantities like the $2K$ matrices $\mathbf{U}^{(k)}$ and $\mathbf{V}^{(k)}$ evolve over time, we append (t) to indicate the quantities as a function of time t when necessary.

Recall that $\mathbf{X}_k(t)$ denotes the set of the flow-k packets that have already arrived at source s by the end of time t and the total number is denoted by $M_k(t) = |\mathbf{X}_k(t)|$. At each time t, the linearly coded packets $Y(t)$ and queuing entries $W_{(k,j_k)}(t) \in Q_S^{(k)}(t)$ for all $S \in 2^{[K]\backslash\{k\}}$ are linear combination of all the arrived packets $(\mathbf{X}_1(t),\cdots \mathbf{X}_K(t))$. Therefore the corresponding global encoding kernel will be an $M(t)$-dimensional row vector in $(\mathsf{GF}(q))^{M(t)}$, where $M(t) \triangleq \sum_{k=1}^{K} M_k(t)$ is the total number of arrived packets.

For simplicity, we use $\mathbf{u}_{(k,j)}(t) = \mathsf{row}_{(k,j)}(\mathbf{U}^{(k)}(t))$ to denote the row corresponding to queued entry $W_{(k,j)}(t)$. Therefore, we have $W_{(k,j)}(t) = \mathbf{u}_{(k,j)}(t)\cdot \vec{\mathbf{X}}_{\mathsf{arr}}$ where $\vec{\mathbf{X}}_{\mathsf{arr}}$ is a column vector that consists of all the packets that have arrived according to its arrival order. For ease of notation, we sometimes abuse the notation slightly by writing $W_{(k,j)}(t) = \mathbf{u}_{(k,j)}(t)\cdot \mathbf{X}(t)$ where $\mathbf{X}(t) \triangleq \{\mathbf{X}_1(t),\ldots,\mathbf{X}_K(t)\}$ is the collection of all the arrived packets. In this notation, we implicitly assume that the set $\mathbf{X}(t)$ is arranged accordingly to the arrival order, the same way as the vector $\mathbf{u}_{(k,j)}(t)$ is arranged.

For any k and $1 \leq i \leq M_k(t)$, we use $\mathbf{e}_{(k,i)}$ to denote an $M(t)$-dimensional row vector with the coordinate/column corresponding to the packet $X_{k,i}$ being one and all the other coordinates being zero. Define the matrix $\mathbf{I}^{(k)}(t)$ as the vertical concatenation of rows $\mathbf{e}_{(k,1)},\ldots,\mathbf{e}_{(k,M_k(t))}$.

For example, suppose five packets have arrived by time t. We sort them from the oldest to the youngest (according to the order of arrivals) and they are $X_{1,1}, X_{2,1}, X_{1,2}, X_{3,1}, X_{2,2}$. Namely, the first packet of the user-2 stream is the second oldest among the five arrived packets and the second packet of the user-2 stream is the youngest of all. Then

$$\mathbf{I}^{(2)}(t) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

One can see that $\mathbf{I}^{(2)}(t)$ is *not an identity matrix* even though it describes the coding vectors of all original uncoded user-2 packets. The need for defining such $\mathbf{I}^{(k)}(t)$ matrix is due to the fact that the arrivals of packets of different users are multiplexed, therefore consecutive columns generally does not represent consecutive packets of the same user. Instead, the columns are based on the arrival order.

Also we define $\mathbf{W}_k(t)$ as a column vector that vertically concatenates all the queue entries $W_{(k,i)}$ in flow-k queues; and define $\mathbf{Z}_k(t)$ as a column vector that vertically concatenates all the packets that have been received by destination d_k. Then we have the following lemmas.

Lemma 6. *In the end of time t, for all $k \in [K]$ we have $\mathbf{W}_k(t) = \mathbf{U}^{(k)}(t)\mathbf{X}(t)$, $\mathbf{Z}_k(t) = \mathbf{V}^{(k)}(t)\mathbf{X}(t)$, and $\mathbf{X}_k(t) = \mathbf{I}^{(k)}(t)\mathbf{X}(t)$.*

Proof. The result is straightforward following the construction of $\mathbf{U}^{(k)}$ and $\mathbf{V}^{(k)}$ in Section IV-B and the definitions in this appendix. □

Recall that $\langle \mathbf{A} \rangle$ denotes the row space of matrix $\mathbf{A}$ and we use $\langle \mathbf{A} \rangle \oplus \langle \mathbf{B} \rangle$ to denote the *sum space* of $\langle \mathbf{A} \rangle$ and $\langle \mathbf{B} \rangle$. That is,

$$\langle \mathbf{A} \rangle \oplus \langle \mathbf{B} \rangle \triangleq \langle (\mathbf{A}^\mathsf{T}, \mathbf{B}^\mathsf{T})^\mathsf{T} \rangle. \tag{74}$$

Then we have

Definition 7 (Non-interfering vector)**.** *In the end of time t, any arbitrarily given row vector $\mathbf{u}$ is "non-interfering" from the perspective of destination d_i if $\mathbf{u} \in \langle \mathbf{V}^{(i)}(t) \rangle \oplus \langle \mathbf{I}^{(i)}(t) \rangle$.*

Lemma 7. *In the end of time t, for any fixed $k \in [K]$ and fixed $S \in 2^{[K]\backslash\{k\}}$, consider any arbitrarily given entry $W_{(k,j)}(t)$ in queue $Q_S^{(k)}$. The corresponding vector $\mathbf{u}_{(k,j)}(t)$ is* non-interfering *from the perspective of d_i for all $i \in S\cup\{k\}$.*

Proof. We prove this lemma by mathematical induction on time index t. First consider in the end of the time 0 (before any transmission), since all the queues are empty, Lemma 7 holds in the end of time 0.

Suppose Lemma 7 is satisfied in the end of time $t-1$. We first argue that any vector $\mathbf{u}$ that is non-interfering for user d_k at time $t-1$ is also non-interfering for user d_k at time t, once we pad the end of $\mathbf{u}$ by zeros that take into account the newly added columns during time t. The reason is that $\langle \mathbf{V}^{(k)}(t) \rangle$ represents the row space spanned by the global encoding kernels of all the packets received by d_k at time t. As t becomes larger, the row space becomes larger as well. Again, if we abuse the notation slightly by implicitly assuming zeros are padded to take into account the newly added columns, then this simple observation implies $\langle \mathbf{V}^{(k)}(t-1) \rangle \subseteq \langle \mathbf{V}^{(k)}(t) \rangle$. Similarly, $\langle \mathbf{I}^{(k)}(t-1) \rangle \subseteq \langle \mathbf{I}^{(k)}(t) \rangle$

since $\mathbf{I}^{(k)}(t)$ represents the row space spanned by the user-k packets that have already arrived at the source s. Since both $\mathbf{V}^{(k)}(t)$ and $\mathbf{I}^{(k)}(t)$ monotonically increase over time, any $\mathbf{u}$ that is non-interfering at time $t-1$ is also non-interfering at time t. As a result, we only need to consider those queued entries that are either newly injected to the queues during time t, or those that have been modified during time t.

According to Section IV-B, Component 2 will insert new entries in the queues and Component 5 will modify the queued entries according to the packet reception set $S_{\text{rx}}(t)$. All other components do not change the queued entry. In the sequel, we discuss Components 2 and 5, respectively.

In Component 2, if there is a flow-k input packet $X_{k,i}$ arriving at source, then we create a new entry $W_{(k,i)}(t) = X_{k,i}$ in $Q^{(k)}_{\emptyset}$. In the end of time t we add a new row $\mathbf{e}_{(k,i)}$ to matrix $\mathbf{U}^{(k)}$. Since $\mathbf{e}_{(k,i)} \in \langle \mathbf{I}^{(k)}(t)\rangle$, the newly added entry $W_{(k,i)}(t)$ in $Q^{(k)}_{\emptyset}$ is non-interfering from the perspective of d_k. Lemma 7 holds for the newly inserted entry.

After transmission $Y(t)$, Component 5 updates the queues according to the reception set $S_{\text{rx}}(t)$. We discuss the following cases in the end of time t.

Case 1: $k \in S_{\text{rx}}(t)$, that is, d_k receives $Y(t)$. According to Component 5, the entry $W_{(k,j_k)}(t-1)$ is then removed from the queue without creating any new entry. Since Lemma 7 focuses only on the entries still in the queues, Lemma 7 holds naturally.

Case 2: $k \notin S_{\text{rx}}(t)$. According to *Component 5*, the entry $W_{(k,j_k)}(t-1)$ is removed from $Q^{(k)}_{T^*\backslash\{k\}}(t)$ and a new entry $W_{(k,j_k)}(t) = Y(t)$ is injected to a queue $Q^{(k)}_{\tilde{S}^*}(t)$ satisfying $\tilde{S}^* \subseteq (T^*\backslash\{k\}) \cup S_{\text{rx}}(t)$, with the corresponding new global encoding kernel $\mathbf{u}_{(k,j_k)}(t) = \mathbf{y}(t)$.

By induction, for any fixed $k' \in T^*$, $\mathbf{u}_{(k',j_{k'})}(t) \in \langle \mathbf{V}^{(i)}(t-1)\rangle \oplus \langle \mathbf{I}^{(i)}(t-1)\rangle$ for all $i \in (T^*\backslash\{k'\}) \cup \{k'\} = T^*$. Therefore, the corresponding global encoding kernel $\mathbf{y}(t) = \sum_{k'\in T^*} \beta_{k'} \mathbf{u}_{(k',j_{k'})}(t) \in \langle \mathbf{V}^{(i)}(t-1)\rangle \oplus \langle \mathbf{I}^{(i)}(t-1)\rangle$ is non-interfering for all $i \in T^*$. Since the new packet $W_{(k,j_k)}(t) = Y(t)$ will be injected to $Q^{(k)}_{\tilde{S}^*}$ with the global encoding kernel $\mathbf{u}_{(k,j_k)}(t) = \mathbf{y}(t)$, what remains to prove is that $\mathbf{y}(t)$ is non-interfering for all $i \in (\tilde{S}^* \cup \{k\})\backslash T^*$.

To that end, we observe that since $\tilde{S}^* \subseteq (T^*\backslash\{k\}) \cup S_{\text{rx}}(t)$, we have $((\tilde{S}^* \cup \{k\})\backslash T^*) \subseteq S_{\text{rx}}(t)$. On the other hand, for all $i \in S_{\text{rx}}(t)$, user d_i receives $Y(t)$. Therefore, $\mathbf{y}(t) \in \langle \mathbf{V}^{(i)}(t)\rangle$ for all $i \in S_{\text{rx}}(t)$. The newly added vector $\mathbf{y}(t)$ is non-interfering from the perspective of all d_i, $i \in (\tilde{S}^* \cup \{k\}\backslash T^*) \subseteq S_{\text{rx}}(t)$. Discussion of Cases 1 and 2 show that Lemma 7 holds after Component 5 as well. The proof is thus complete by induction on the time index t. $\square$

We now prove the following lemma.

Lemma 8. *In the end of time t we have $\langle \mathbf{V}^{(k)}(t)\rangle \oplus \langle \mathbf{U}^{(k)}(t)\rangle = \langle \mathbf{V}^{(k)}(t)\rangle \oplus \langle \mathbf{I}^{(k)}(t)\rangle$ for all $k \in [K]$.*

Proof. We prove this lemma by mathematical induction on time index t. First consider in the end of time 0 (before any transmission). Since there has not been any packet arrival at source s, we have $\langle \mathbf{I}^{(k)}(0)\rangle = \{\mathbf{0}\}$, i.e., the row space of the arrived messages contains only the all-zero vector. Similarly, since all queues are empty at time 0, we have $\langle \mathbf{U}^{(k)}(0)\rangle = \{\mathbf{0}\}$. Finally, since no packet has ever been delivered to the destinations, we have $\langle \mathbf{V}^{(k)}(0)\rangle = \{\mathbf{0}\}$. As a result, Lemma 8 holds at time 0.

Suppose, Lemma 8 is satisfied in the end of time $t-1$. We will prove that Lemma 8 still holds when we execute Components 2 to 5 sequentially in time t.

In the beginning of time t, in Component 2, we add new row vector $\mathbf{u}_{(k,i)} = \mathbf{e}_{(k,i)}$ for each arrival input packet $X_{k,i}$, $i \in \{M_k(t-1)+1, \ldots, M_k(t)\}$. For simplicity, we use $\mathbf{E}^{(k)}(t)$ to denote the matrix with the rows $\{\mathbf{e}_{(k,i)} : i = M_k(t-1) + 1, \ldots, M_k(t)\}$. It is then clear that $\langle \mathbf{U}^{(k)}(t)\rangle = \langle \mathbf{U}^{(k)}(t-1)\rangle \oplus \langle \mathbf{E}^{(k)}(t)\rangle$ and $\langle \mathbf{I}^{(k)}(t)\rangle = \langle \mathbf{I}^{(k)}(t-1)\rangle \oplus \langle \mathbf{E}^{(k)}(t)\rangle$. Therefore, (8) holds after executing Component 2.

Component 3 selects the coding set T^* and does not change the matrices $\mathbf{U}^{(k)}(t)$, $\mathbf{V}^{(k)}(t)$, and $\mathbf{I}^{(k)}(t)$. Similarly, Component 4 chooses the mixing coefficients β_k, $k \in T^*$, and does not change the matrices $\mathbf{U}^{(k)}(t)$, $\mathbf{V}^{(k)}(t)$, and $\mathbf{I}^{(k)}(t)$. As a result, (8) holds after executing Components 3 and 4 as well.

We now consider Component 5. For ease of notation, we use the subscript "old" to denote the $\mathbf{U}^{(k)}(t)$ and $\mathbf{V}^{(k)}(t)$ matrices right before we execute Component 5 and use the subscript "new" to denote the matrices right after we execute Component 5. Recall that for all $k_0 \in T^*$, we choose the head-of-line packet $W_{(k_0,j_{k_0})}(t)$ and construct $\mathbf{U}^{(k_0)}(t)$ by removing the row $\mathbf{u}_{(k_0,j_{k_0})}(t)$ corresponding to $W_{(k_0,j_{k_0})}(t)$ from $\mathbf{U}^{(k_0)}(t)$. We now consider the following cases.

Case 1: Consider those destinations d_{k_0} such that $k_0 \notin T^*$. According to Component 5, there is no change to any of the flow-k_0 queues. Therefore, Lemma 8 holds for such k_0 after executing Component 5.

Case 2: Consider those destinations d_{k_0} such that $k_0 \in T^*$ and $k_0 \in S_{\text{rx}}(t)$. If $Q^{(k_0)}_{T^*\backslash\{k_0\}}$ is empty, then we use a null packet $W_{(k_0,0)}$ in Component 4 such that $\mathbf{U}^{(k_0)}_{\text{new}} = \mathbf{U}^{(k_0)}_{\text{old}}$ and $\mathbf{V}^{(k_0)}_{\text{new}} = \mathbf{V}^{(k_0)}_{\text{old}}$ and Lemma 8 thus holds. If $Q^{(k_0)}_{T^*\backslash\{k_0\}}$ is non-empty, according to Component 5, the entry $W_{(k_0,j_{k_0})}(t)$ is removed from $Q^{(k_0)}_{T^*\backslash\{k_0\}}(t)$ without creating new entry. Therefore the new row space $\langle \mathbf{V}^{(k_0)}_{\text{new}}\rangle$ becomes $\langle \mathbf{V}^{(k_0)}_{\text{old}}\rangle \oplus \langle \mathbf{y}(t)\rangle$ and the new row space $\langle \mathbf{U}^{(k_0)}_{\text{new}}\rangle = \langle \mathbf{U}'^{(k_0)}_{\text{old}}\rangle$. In the following we will prove

$$\begin{aligned}
&\langle \mathbf{V}^{(k_0)}_{\text{new}}\rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{new}}\rangle \\
=&\langle \mathbf{V}^{(k_0)}_{\text{old}}\rangle \oplus \langle \mathbf{y}(t)\rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}}\rangle \\
=&\langle \mathbf{V}^{(k_0)}_{\text{old}}\rangle \oplus \langle \mathbf{y}(t)\rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}}\rangle \quad (75)\\
=&\langle \mathbf{V}^{(k_0)}_{\text{old}}\rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}}\rangle. \quad (76)
\end{aligned}$$

Eq. (76) holds due to the following reason. Lemma 7 shows that for all $k \in T^*$ $\mathbf{u}_{(k,j_k)}(t)$ is non-interfering from the perspective d_i, $i \in (T^*\backslash\{k\}) \cup \{k\} = T^*$. Therefore by the induction condition, we have for all $k \in T^*$ $\mathbf{u}_{(k,j_k)}(t) \in \langle \mathbf{V}^{(k_0)}(t)\rangle \oplus \langle \mathbf{I}^{(k_0)}(t)\rangle = \langle \mathbf{V}^{(k_0)}_{\text{old}}\rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}}\rangle$ and hence $\mathbf{y}(t) = \sum_{k\in T^*} \beta_k \mathbf{u}_{(k,j_k)}(t) \in \langle \mathbf{V}^{(k_0)}_{\text{old}}\rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}}\rangle$.

We then show that equality (75) always holds. Since $\mathbf{U}'^{(k_0)}_{\text{old}}$ is obtain by removing $\mathbf{u}_{(k_0,j_{k_0})}(t)$, if $\mathbf{u}_{(k_0,j_{k_0})}(t) \in \langle \mathbf{V}^{(k_0)}_{\text{old}}\rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}}\rangle$, then $\langle \mathbf{V}^{(k_0)}_{\text{old}}\rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}}\rangle = \langle \mathbf{V}^{(k_0)}_{\text{old}}\rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}}\rangle$ and

(75) holds naturally. If $\mathbf{u}_{(k_0,j_{k_0})}(t) \notin \langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}} \rangle$, we then notice that

$$\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}} \rangle \supseteq \langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}} \rangle \tag{77}$$

and

$$\mathsf{rank}(\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}} \rangle) \geq \mathsf{rank}(\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}} \rangle) - 1 \tag{78}$$

where both inequalities are due to that $\mathbf{U}'^{(k_0)}_{\text{old}}$ is obtained from $\mathbf{U}^{(k_0)}_{\text{old}}$ by removing only 1 row. Nonetheless, by the way we select $\{\beta_k\}$ in Component 4 such that $\mathbf{y}(t) \notin \langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}} \rangle$. Therefore, we also have

$$\begin{aligned}&\mathsf{rank}(\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{y}(t) \rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}} \rangle) \\ &= \mathsf{rank}(\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}} \rangle) + 1. \end{aligned} \tag{79}$$

Together, we have

$$\begin{aligned}
&\mathsf{rank}(\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}} \rangle) && (80)\\
&=\mathsf{rank}(\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{y}(t) \rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}} \rangle) && (81)\\
&\geq\mathsf{rank}(\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{y}(t) \rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}} \rangle) && (82)\\
&\geq \left(\mathsf{rank}(\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}} \rangle) - 1\right) + 1 && (83)
\end{aligned}$$

where (81) is a repeat of (76); (82) follows from (77); and (83) follows from (78) and (79). This implies that the two subspaces $\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{y}(t) \rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}} \rangle$ and $\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{y}(t) \rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}} \rangle$ have the same rank. Since (77) also implies

$$\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{y}(t) \rangle \oplus \langle \mathbf{U}^{(k_0)}_{\text{old}} \rangle \supseteq \langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{y}(t) \rangle \oplus \langle \mathbf{U}'^{(k_0)}_{\text{old}} \rangle$$

the two spaces must be equal. We have thus proven (75). Since (75) holds after executing Component 5, by the induction hypothesis, we have proven Lemma 8 for such k_0 after executing Component 5.

Case 3: Consider the destinations d_{k_0} such that $k_0 \in T^*$ and $k_0 \notin S_{\text{rx}}(t)$. That is d_{k_0} does not receive the desired packet $Y(t)$. According to Component 5, the entry $W_{(k_0,j_{k_0})}(t)$ is then removed from $Q^{(k_0)}_{T^*\backslash\{k_0\}}(t)$ and an entry $W_{(k_0,j_{k_0})}(t) = Y(t)$ is created in $Q^{(k_0)}_{\tilde{S}^*}(t)$ corresponding to new row $\mathbf{u}_{(k_0,j_{k_0})}(t) = \mathbf{y}(t)$. Therefore the new row space $\langle \mathbf{V}^{(k_0)}_{\text{new}} \rangle$ equals to $\langle \mathbf{V}^{(k_0)}_{\text{old}} \rangle$ and the new row space $\langle \mathbf{U}^{(k_0)}_{\text{new}} \rangle = \langle \mathbf{U}'^{(k_0)}_{\text{old}} \rangle \oplus \langle \mathbf{y}(t) \rangle$. Comparing Cases 2 and 3, we notice that the only difference is that in Case 2, the new vector $\mathbf{y}(t)$ is later classified as part of the $\mathbf{V}^{(k_0)}_{\text{new}}$ matrix but in Case 3, the vector $\mathbf{y}(t)$ is later classified as part of the $\mathbf{U}^{(k_0)}_{\text{new}}$ matrix. Since the statement in Lemma 8 does not distinguish whether the changes actually happen to $\mathbf{V}^{(k_0)}_{\text{new}}$ or $\mathbf{U}^{(k_0)}_{\text{new}}$, by the same arguments as in Case 2, Lemma 8 holds for Case 3 as well.

Since Lemma 8 holds after sequentially executing Components 2 to 5, the proof is completed by induction on the time index t. □

Note that Lemma 8 directly implies the decodability defined in (9) and (10). Specifically, in the end of time slot t for any $k \in [K]$, $\langle \mathbf{V}^{(k)}(t) \rangle \oplus \langle \mathbf{U}^{(k)}(t) \rangle = \langle \mathbf{V}^{(k)}(t) \rangle \oplus \langle \mathbf{I}^{(k)}(t) \rangle$ implies that there exists a matrix $\mathbf{G}^{(k)}(t)$ such that

$$\begin{bmatrix}\mathbf{V}^{(k)}(t)\\ \mathbf{I}^{(k)}(t)\end{bmatrix} = \mathbf{G}^{(k)}(t)\begin{bmatrix}\mathbf{V}^{(k)}(t)\\ \mathbf{U}^{(k)}(t)\end{bmatrix}.$$

Therefore by Lemma 6, we have

$$\begin{bmatrix}\mathbf{Z}_k(t)\\ \mathbf{X}_k(t)\end{bmatrix} = \mathbf{G}^{(k)}(t)\begin{bmatrix}\mathbf{Z}_k(t)\\ \mathbf{W}_k(t)\end{bmatrix}. \tag{84}$$

Note that per our definition the received linearly coded packets are $[Z_k]_1^t = \mathbf{Z}_k(t)$ and the packets in the queues are $Q(t) = \{\mathbf{W}_1(t), \mathbf{W}_2(t), \cdots, \mathbf{W}_K(t)\}$. Therefore, (84) implies that there exists a decoder $g^{(\text{sq})}_{k,t}$ such that $\mathbf{X}_k(t) = g^{(\text{sq})}_{k,t}(\mathbf{Z}_k(t), \mathbf{W}_k(t), [S_{\text{rx}}]_1^t, [\vec{A}]_1^t)$, see (9) and (10). The proof of Proposition 4 is complete.

Appendix F
Proof of Propositions 5 and 6

Lemma 9. *Given any arbitrary integer $k \in [K]$ and rate value R_k, the statement "$\vec{x} \triangleq \{x_T \geq 0 : \forall T \in 2^{[K]}\backslash\{\emptyset\}\}$ satisfies Condition 2 in Proposition 3 for the given k" is equivalent to the existence of non-negative variables*

$$\vec{y} \triangleq \Big\{ y_{k;S_I\to d_k} \geq 0, y^{[S_X]}_{k;S_I\to S_O} \geq 0 : S_I \in 2^{[K]\backslash\{k\}}, S_X \in 2^{[K]\backslash(S_I\cup\{k\})}, S_O \in 2^{S_I\cup S_X} \Big\} \tag{85}$$

satisfying the following three groups of equations: Group 1:

$$R_k \cdot 1_{\{S=\emptyset\}} + \sum_{\forall S_X, S_I} y^{[S_X]}_{k;S_I\to S} \leq y_{k;S\to d_k} + \sum_{\forall S_X, S_O} y^{[S_X]}_{k;S\to S_O}, \quad \forall S \in 2^{[K]\backslash\{k\}} \tag{86}$$

Group 2:

$$y_{k;S_I\to d_k} \leq x_{S_I\cup\{k\}} \cdot p_k, \quad \forall S_I \in 2^{[K]\backslash\{k\}} \tag{87}$$

and Group 3:

$$\sum_{\forall S_O \in 2^{S_I\cup S_X}} y^{[S_X]}_{k;S_I\to S_O} \leq x_{S_I\cup\{k\}} \cdot p_{S_X\overline{[K]\backslash(S_I\cup S_X)}}, \quad \forall S_I \in 2^{[K]\backslash\{k\}}, S_X \in 2^{[K]\backslash(S_I\cup\{k\})} \tag{88}$$

Proof. From Lemmas 1 and 2, it is clear that Condition 2 in Proposition 3 is equivalent to "the minimum-cut-value of the k-th virtual sub-network in Section IV is no less than R_k". In the k-th virtual sub-network, if we use variables $y_{k;S_I\to d_k}$ to denote the flow value over the edge from $\text{vn}^{(k)}_{S_I}$ to $\text{vn}^{(k)}_{\text{grnd}}$ and use $y^{[S_X]}_{k;S_I\to S_O}$ to denote the flow value over the edge from $\text{vn}^{(k)}_{S_I}$ though auxiliary node S_X and then to $\text{vn}^{(k)}_{S_O}$, then Group 1 equality (86) is simply the flow-conservation law if we replace the $\leq$ in (86) by $=$. Groups 2 and 3 inequalities in (87) and (88) impose that the flow values are no larger than the assigned edge capacities. Therefore this lemma becomes a directly result of applying the max-flow min-cut theorem to the k-th virtual sub-network, if we replace the $\leq$ in (86) by $=$. Finally, it is well known, see [30], that replacing $=$ in the flow condition (86) by $\leq$ does not alter the max-flow characterization. The proof of this lemma is thus complete. □

Throughout this section of appendix, we perform exclusively queue-length-based stability analysis [31], which focuses on the length of the queue $|Q^{(k)}_S|$ instead of the content of the queue $Q^{(k)}_S$. To simplify the notation, in this appendix we slightly abuse the notation and directly use $Q_S^{(k)}$ to represent the queue length so that we do not need to add the length operator $|\cdot|$ for all the queues.

Definition 8 (Offered packet movement)**.** *The offered packet movement (including real and dummy packets) at time t is a set of Bernoulli random variables*

$$\left\{Y_{k;S_I\to d_k}(t) : k\in[K], S_I\in 2^{[K]\backslash\{k\}}\right\} \tag{89}$$

$$\text{and } \left\{Y^{[S_X]}_{k;S_I\to S_O}(t) : k\in[K], S_I\in 2^{[K]\backslash\{k\}},\right. \\ \left. S_X\in 2^{[K]\backslash(S_I\cup\{k\})}, S_O\in 2^{S_I\cup S_X}\right\} \tag{90}$$

such that the random variable $Y_{k;S_I\to d_k}(t)=1$ if and only if there is a packet removed from the virtual queue $Q_{S_I}^{(k)}$ to destination d_k (see Component 5) and $Y^{[S_X]}_{k;S_I\to S_O}(t)=1$ if and only if there is a packet movement from $Q_{S_I}^{(k)}$ to $Q_{S_O}^{(k)}$ in the scenario $k\notin S_{\text{rx}}(t)$ and $S_X=S_{\text{rx}}(t)\backslash S_I$.

The offered packet movement random variable $\vec{Y}(t)$ is defined from the edge's perspective. We now define the queue-length displacement random variables $\Delta Q_S^{(k)}(t)$, which is defined based on the queue's perspective.

$$\Delta Q_S^{(k)}(t) \triangleq A_k(t)\cdot 1_{\{S=\emptyset\}} + \sum_{\forall S_X,S_I} Y^{[S_X]}_{k;S_I\to S}(t) \\ - Y_{k;S\to d_k}(t) - \sum_{\forall S_X,S_O} Y^{[S_X]}_{k;S\to S_O}(t). \tag{91}$$

It is clear from the above definition, we have

$$Q_S^{(k)}(t+1) = \left(Q_S^{(k)}(t)+\Delta Q_S^{(k)}(t)\right)^+ \tag{92}$$

since $\Delta Q_S^{(k)}(t)$ captures the packet arrival process $A_k(t)$ and the packet removal/movement random variables $\vec{Y}(t)$ at time t.

A. Proof of Proposition 5

We prove this proposition by the Lyapunov drift analysis. Define a Lyapunov function as follows.

$$L_q(\vec{Q},t) \triangleq \frac{1}{2}\sum_{k\in[K],S\in 2^{[K]\backslash\{k\}}} \left(Q_S^{(k)}(t)\right)^2. \tag{93}$$

We will prove that there exists a negative drift when the summation of all queue lengths is sufficiently large. That is,

$$\mathsf{E}\left\{ L_q(\vec{Q},t+1) - L_q(\vec{Q},t) \,\middle|\, \vec{Q}(t)\right\} \\ \leq \mathsf{const} - \epsilon\cdot\left(\sum_{k\in[K],S\in 2^{[K]\backslash\{k\}}} Q_S^{(k)}(t)\right), \tag{94}$$

for some fixed constant const. The negative drift then immediately implies that the queue lengths are stable.

To prove the negative drift, we first notice that by Lemma 9 if there exists variables $\vec{x}$ satisfying (18) and strict (17), then there exist $\vec{y}$, joint with $\vec{x}$, satisfying (86), (87), (88), and strict (17). We now argue that, without loss of generality, we can further assume that $\vec{x}$ and $\vec{y}$ satisfy (17), (87), and (88) with exact equality, and satisfy (86) with strict inequality for all $k\in[K]$ and $S\in 2^{[K]\backslash\{k\}}$.

To that end, we observe that since (17) holds with strict inequality, we can increase all x_T for all T by the same amount $\delta>0$ such that the new x_T satisfies (17) with equality. The increase of x_T for all T ensures that (87) is now satisfied with strict inequality since we assume $p_k>0$ for all k. Inequality (88) still holds since x_T only appears in the right-hand side of (88). Since (87) is now strict inequality, we can increase $y_{k;S_I\to d_k}$ value by a strictly positive value until (87) becomes equality for all $k\in[K]$, $S_I\in 2^{[K]\backslash\{k\}}$. The increase of $y_{k;S_I\to d_k}$ ensures that (86) becomes strict inequality for all $k\in[K]$, $S_I\in 2^{[K]\backslash\{k\}}$, since $y_{k;S_I\to d_k}$ only appears on the right-hand side of (86). Finally, if any of (88) is a strictly inequality, then we can increase the corresponding $y^{[S_X]}_{k;S_I\to S_I}$ value until (88) becomes equality. The increase of $y^{[S_X]}_{k;S_I\to S_I}$ does not alter (86) since $y^{[S_X]}_{k;S_I\to S_I}$ appears on both sides of (86). The final $\vec{x}$ and $\vec{y}$ satisfy (17), (87), and (88) with equality and satisfy (86) with strict inequality for all k,S. This important observation implies that for some $\epsilon>0$,

$$R_k\cdot 1_{\{S=\emptyset\}} + \sum_{\forall S_X,S_I} y^{[S_X]}_{k;S_I\to S} - y_{k;S\to d_k} - \sum_{\forall S_X,S_O} y^{[S_X]}_{k;S\to S_O} \\ \leq -\epsilon,\quad \forall k\in[K], S\in 2^{[K]\backslash\{k\}}. \tag{95}$$

We are now ready to prove the negative drift. Squaring (92) and taking the conditional expectations yields

$$\begin{aligned} &\mathsf{E}\left\{\left(Q_S^{(k)}(t+1)\right)^2\middle|\,\vec{Q}(t)\right\} \\ \leq&\mathsf{E}\left\{\left(Q_S^{(k)}(t)+\Delta Q_S^{(k)}(t)\right)^2\middle|\,\vec{Q}(t)\right\} \\ =&\mathsf{E}\left\{\left(Q_S^{(k)}(t)\right)^2\middle|\,\vec{Q}(t)\right\} + 2\cdot\mathsf{E}\left\{Q_S^{(k)}(t)\cdot\Delta Q_S^{(k)}(t)\middle|\,\vec{Q}(t)\right\} \\ &+\mathsf{E}\left\{\left(\Delta Q_S^{(k)}(t)\right)^2\middle|\,\vec{Q}(t)\right\}. \end{aligned} \tag{96}$$

We then notice that for any given k, $Y_{k;S_I\to d_k}(t)$ and $Y^{[S_X]}_{k;S_I\to S_O}(t)$ denote the packet movement within the virtual queue at time t. Therefore, at most one of them can be of value 1 at a given time t. Then by the definition of $\Delta Q_S^{(k)}$ in (91), we have for all $k\in[K]$, $S\in 2^{[K]\backslash\{k\}}$

$$A_k(t)\cdot 1_{\{S=\emptyset\}} - 1 \leq \Delta Q_S^{(k)}(t) \leq A_k(t)\cdot 1_{\{S=\emptyset\}} + 1. \tag{97}$$

We thus have

$$\begin{aligned} &\mathsf{E}\left\{\left(\Delta Q_S^{(k)}(t)\right)^2\middle|\,\vec{Q}(t)\right\} \leq \mathsf{E}\left\{(A_k(t)+1)^2\middle|\,\vec{Q}(t)\right\} \\ =&\mathsf{E}\left\{(A_k(t))^2\middle|\,\vec{Q}(t)\right\} + 2\cdot\mathsf{E}\left\{A_k(t)\middle|\,\vec{Q}(t)\right\} + 1 \\ =&(R_k+R_k^2) + 2R_k + 1, \end{aligned} \tag{98}$$

where we use the fact that a Poisson random variable $A_k(t)$ with arrival rate R_k has $\mathsf{E}\{A_k^2(t)\} = R_k + R_k^2$. Then by (96) and (98), the Lyapunov drift function at time t becomes

$$\begin{aligned}&\mathsf{E}\left\{ L_q(\vec{Q},t+1) - L_q(\vec{Q},t)\middle| \vec{Q}(t)\right\}\\ &\leq \sum_{k\in[K],S\in 2^{[K]\backslash\{k\}}} \left(\frac{1}{2}\mathsf{E}\left\{\left(\Delta Q_S^{(k)}(t)\right)^2\middle|\vec{Q}(t)\right\}\right.\\ &\quad \left.+\ \mathsf{E}\left\{Q_S^{(k)}(t)\cdot\Delta Q_S^{(k)}(t)\middle|\vec{Q}(t)\right\}\right)\\ &=\mathsf{const} + \sum_{k\in[K],S\in 2^{[K]\backslash\{k\}}} Q_S^{(k)}(t)\cdot\mathsf{E}\left\{\Delta Q_S^{(k)}(t)\middle|\vec{Q}(t)\right\}\end{aligned}$$

where the constant being $\mathsf{const} \triangleq \sum_{k\in[K]} 2^{K-2}(R_k^2+3R_k+1)$. The remaining proof of the negative drift becomes proving

$$\sum_{k\in[K],S\in 2^{[K]\backslash\{k\}}} Q_S^{(k)}(t)\cdot\mathsf{E}\left\{\Delta Q_S^{(k)}(t)\middle|\vec{Q}(t)\right\} \tag{99}$$

$$\leq -\epsilon\left(\sum_{k\in[K],S\in 2^{[K]\backslash\{k\}}} Q_S^{(k)}(t)\right). \tag{100}$$

To prove (100), we first show that our scheme of choosing the coding set $T^*(t)$ in Component 3 and the destination queue $Q_{\tilde{S}^*}^{(k_0)}$ in Component 5 minimizes the summation in (99) among all the schemes which move packets among the nodes in the virtual network, conditioning on knowing the queue lengths $\vec{Q}(t)$.

Suppose that at time t, a competing scheme chooses the coding set $T'(t)$ and we denote the resulting offered packet movement by $\vec{Y}'(t)$ and the resulting queue-length displacement by $\Delta Q_S'^{(k)}(t)$ for all k,S. We then have

$$\begin{aligned}&\sum_{\forall k,S} Q_S^{(k)}(t)\cdot\mathsf{E}\left\{\Delta Q_S'^{(k)}(t)\middle|\vec{Q}(t)\right\}\\ &=\sum_{k\in[K]} R_k\cdot Q_\emptyset^{(k)}(t) - \sum_{\forall k,S} Q_S^{(k)}(t)\mathsf{E}\left\{Y'_{k;S\to d_k}(t)\middle|\vec{Q}(t)\right\}\\ &-\left(\sum_{\forall k,S_I,S_X,S_O}\left(Q_{S_I}^{(k)}(t)-Q_{S_O}^{(k)}(t)\right)\mathsf{E}\left\{Y'^{[S_X]}_{k;S_I\to S_O}(t)\middle|\vec{Q}(t)\right\}\right)\end{aligned} \tag{101}$$

where (101) follows from (91). Recall that $T'(t)$ is a deterministic function of $\vec{Q}(t)$ and denote the packet movement destination by $Q_{\tilde{S}'}^{(k)}(t)$, where $\tilde{S}'$ is a function[11] of S_X. Due to the memorylessness of the underlying PECs, we have the conditional expectation being

$$\mathsf{E}\left\{Y'_{k;S_I\to d_k}(t)\middle|\vec{Q}(t)\right\} = 1_{\{T'(t)=S_I\cup\{k\}\}}\cdot p_k \tag{102}$$

$$\begin{aligned}&\mathsf{E}\left\{Y'^{[S_X]}_{k;S_I\to S_O}(t)\middle|\vec{Q}(t)\right\} = 1_{\{T'(t)=S_I\cup\{k\}\}}\cdot\\ &\quad\Pr\left(k\notin S_{\text{rx}}(t), S_X = S_{\text{rx}}(t)\backslash S_I\right)\cdot 1_{\{S_O=\tilde{S}'\}}\end{aligned} \tag{103}$$

where (102) holds since the offered movement $Y'_{k;S_I\to d_k}(t)=1$ if and only if $T'(t)=S_I\cup\{k\}$, $k\in S_{\text{rx}}(t)$, and by the memorylessness of the channel; and (103) holds since $Y'^{[S_X]}_{k;S_I\to S_O}(t)=1$ if and only if $T'(t)=S_I\cup\{k\}$, $k\notin S_{\text{rx}}(t)$, $S_X=S_{\text{rx}}(t)\backslash S_I$, $S_O=\tilde{S}'$, and by the memorylessness of the channel.

[11]A more precise notation should be $\tilde{S}'(S_X)$ instead. However for notational simplicity we simply use $\tilde{S}'$.

Observe that $\Pr(k\notin S_{\text{rx}}(t), S_X = S_{\text{rx}}(t)\backslash S_I) = p_{S_X\overline{[K]\backslash(S_I\cup S_X)}}$, eq. (101) then becomes

$$\begin{aligned}&\sum_{k\in[K]} R_k\cdot Q_\emptyset^{(k)}(t) - \sum_{\forall S_I\in 2^{[K]\backslash\{k\}}} 1_{\{T'(t)=S_I\cup\{k\}\}}\cdot\\ &\left(Q_{S_I}^{(k)}(t)\cdot p_k + \sum_{\forall S_X} p_{S_X\overline{[K]\backslash(S_I\cup S_X)}}\left(Q_{S_I}^{(k)}(t)-Q_{\tilde{S}'}^{(k)}(t)\right)\right)\\ &=\sum_{k\in[K]} R_k\cdot Q_\emptyset^{(k)}(t) - \sum_{k\in T'(t)}\left(Q_{T'(t)\backslash\{k\}}^{(k)}(t)\right.\\ &\quad\left.-\sum_{\forall S_X\in 2^{[K]\backslash T'(t)}} p_{S_X\overline{[K]\backslash(S_X\cup T'(t)\backslash\{k\})}}\cdot Q_{\tilde{S}'}^{(k)}(t)\right),\end{aligned} \tag{104}$$

where the last equality uses the observation that $p_k + \sum_{\forall S_X\in 2^{[K]\backslash T'(t)}} p_{S_X\overline{[K]\backslash(S_X\cup T'(t)\backslash\{k\})}} = 1$.

Therefore from (104) we obtain

$$\sum_{\forall k,S} Q_S^{(k)}(t)\cdot\mathsf{E}\left\{\Delta Q_S'^{(k)}(t)\middle|\vec{Q}(t)\right\}$$

$$\geq \sum_{k\in[K]} R_k\cdot Q_\emptyset^{(k)}(t) - \sum_{k\in T'(t)}\text{bp}\left(Q_{T'(t)\backslash\{k\}}^{(k)}(t)\right) \tag{105}$$

$$\geq \sum_{k\in[K]} R_k\cdot Q_\emptyset^{(k)}(t) - \sum_{k\in T^*(t)}\text{bp}\left(Q_{T^*(t)\backslash\{k\}}^{(k)}(t)\right) \tag{106}$$

$$=\sum_{\forall k,S} Q_S^{(k)}(t)\cdot\mathsf{E}\left\{\Delta Q_S^{(k)}(t)\middle|\vec{Q}(t)\right\} \tag{107}$$

where (105) follows from (101), (104), and the definition of the backpressure expression in (26). Specifically, for the same event $\{k\notin S_{\text{rx}}(t), S_X = S_{\text{rx}}(t)\backslash T'(t)\}$, the backpressure expression $\text{bp}(Q_{T'(t)\backslash\{k\}}^{(k)}(t))$ involves the term $q(k,(T'(t)\backslash\{k\})\cup S_X)$ in (25), which chooses $\tilde{S}^*$ that minimizes $Q_{\tilde{S}}^{(k)}$. Comparing the backpressure expression to (104), replacing $\tilde{S}'$ by $\tilde{S}^*$ decreases the overall value and we thus have (105). Eq. (106) follows from (27) since our coding set choice $T^*(t)$ maximizes the sum of the backpressure; (107) holds by the same reasons as (101), (104), and (26) with the following simple substitution $\Delta Q_S'^{(k)}(t)=\Delta Q_S^{(k)}(t)$, $T'(t)=T^*(t)$, and $\tilde{S}'=\tilde{S}^*$. The above arguments show that at any time t our scheme attains the minimum $\sum_{\forall k,S} Q_S^{(k)}(t)\cdot\mathsf{E}\{\Delta Q_S^{(k)}(t)|\vec{Q}(t)\}$ among all possible scheme designs that move packets among the nodes in the virtual network.

Given $\vec{x}$ and $\vec{y}$ satisfying (87), (88) with equality and (95), we consider a competing scheme that chooses the coding set $T'(t)$ randomly according to the probability distribution $\{x_T\}$ (recalling that $\sum_T x_T = 1$), and chooses the new destination queue $Q_{\tilde{S}'}^{(k)}$, under event $\{k\notin S_{\text{rx}}(t), S_X = S_{\text{rx}}(t)\backslash T'(t)\}$, randomly with the conditional distribution

$$\Pr\left(\tilde{S}'\middle| S_X, T'(t)\right) = \frac{y^{[S_X]}_{k;T'(t)\backslash\{k\}\to\tilde{S}'}}{x_{T'(t)}\cdot p_{S_X\overline{[K]\backslash(S_X\cup T'(t)\backslash\{k\})}}} \tag{108}$$

which satisfies $\sum_{\forall\tilde{S}'}\Pr\left(\tilde{S}'\middle| S_X, T'(t)\right) = 1$. If we use $\Delta Q_S'^{(k)}(t)$ to denote the drift under this competing scheme, we will have

$$\sum_{k\in[K],S\in 2^{[K]\backslash\{k\}}} Q_S^{(k)}(t)\cdot \mathsf{E}\left\{ \Delta Q_S'^{(k)}(t) \middle| \vec{Q}(t)\right\}$$
$$\leq -\epsilon\left(\sum_{k\in[K],S\in 2^{[K]\backslash\{k\}}} Q_S^{(k)}(t)\right). \tag{109}$$

The reason is that this scheme randomly chooses the coding set $T'(t)$ and destination queue $\tilde{S}'$ independently such that

$$\mathsf{E}\left\{Y'_{k;S\to d_k}(t)\middle|\vec{Q}(t)\right\} = \Pr\left(T'(t)=S\cup\{k\}\right)\cdot p_k$$
$$= y_{k;S\to d_k} \tag{110}$$
$$\mathsf{E}\left\{Y'^{[S_X]}_{k;S_I\to S_O}(t)\middle|\vec{Q}(t)\right\} = \Pr\left(T'(t)=S_I\cup\{k\}\right)\cdot$$
$$p_{S_X\overline{[K]\backslash(S_I\cup S_X)}}\cdot\Pr\left(S_O|S_X,T'(t)\right) = y^{[S_X]}_{k;S_I\to S_O}. \tag{111}$$

where (110) and (111) hold due to $\Pr(T'(t)=S_I\cup\{k\}) = x_{S_I\cup\{k\}}$, the memorylessness of the underlying PECs, and (108). Consequently, we have

$$\mathsf{E}\left\{\Delta Q_S'^{(k)}(t)\middle|\vec{Q}(t)\right\} = R_k\cdot 1_{\{S=\emptyset\}} + \sum_{\forall S_X,S_I} y^{[S_X]}_{k;S_I\to S}$$
$$- y_{k;S\to d_k} - \sum_{\forall S_X,S_O} y^{[S_X]}_{k;S\to S_O} \leq -\epsilon, \tag{112}$$

where the equality holds by substituting (110) and (111) into (91) and by (95).

Finally, since we have already proven that our scheme minimizes (99), by (99) and (109) we thus have (100). The proof of the negative drift and the corresponding Lyapunov analysis is thus complete.

B. *Proof of Proposition 6*

Consider any rate $\vec{R}=(R_1,\ldots,R_K)$ that can be stabilized by the proposed 5-component sequential coding scheme. By Lemma 4, there exists a constant $D<\infty$ and a subsequence $\{t_m\}$ such that for all positive integers $m=1,2,\cdots$

$$\sum_{k\in[K],S\in 2^{[K]\backslash\{k\}}} \mathsf{E}\left\{Q_S^{(k)}(t_m)\right\}\leq D. \tag{113}$$

Since each queue length is always nonnegative, we immediately have: For all $k\in[K]$ and $S\in 2^{[K]\backslash\{k\}}$,

$$\mathsf{E}\left\{Q_S^{(k)}(t_m)\right\}\leq D. \tag{114}$$

Let $T^*(t)$ denote the coding set chosen in Component 3. We now construct[12] the following nonnegative variables $\vec{x}$ and $\vec{y}$ based on our 5-component sequential coding scheme:

$$x_T \triangleq \lim_{m\to\infty}\frac{1}{t_m}\sum_{\tau=0}^{t_m-1} 1_{\{T^*(\tau)=T\}}, \tag{115}$$
$$y_{k;S_I\to d_k} \triangleq \lim_{m\to\infty}\frac{1}{t_m}\sum_{\tau=0}^{t_m-1}\mathsf{E}\left\{Y_{k;S_I\to d_k}(\tau)\right\}, \tag{116}$$
$$y^{[S_X]}_{k;S_I\to S_O} \triangleq \lim_{m\to\infty}\frac{1}{t_m}\sum_{\tau=0}^{t_m-1}\mathsf{E}\left\{Y^{[S_X]}_{k;S_I\to S_O}(\tau)\right\}. \tag{117}$$

[12]The limits in (115) to (117) may not converge. If so, we simply use a convergent subsequence of the original sequence $\{t_m\}$ and the rest of the proof holds verbatim.

We then prove that above $\vec{x}$ variables satisfy (17) with equality and (18) in Proposition 3. By Lemma 9, we equivalently show that the above $\vec{x}$ and $\vec{y}$ satisfy (17) with equality and jointly satisfy (86) to (88). Firstly, we have $\sum_{\forall T\in 2^{[K]}\backslash\{\emptyset\}} x_T = \lim_{m\to\infty}\frac{t_m}{t_m}=1$, which satisfies (17) with equality. To prove (87), we notice that by Component 5

$$Y_{k;S_I\to d_k}(\tau) = 1_{\{T^*(\tau)=S_I\cup\{k\}\}\}}\cdot 1_{\{k\in S_{\text{rx}}(\tau)\}}. \tag{118}$$

Since the channel is memoryless, by taking the expectation of (118) and by comparing (115) and (116), inequality (87) thus holds with equality. By the packet movement rule in Component 5, we also have

$$\sum_{S_O\in 2^{S_I\cup S_X}} Y^{[S_X]}_{k;S_I\to S_O}(\tau) = 1_{\{T^*(\tau)=S_I\cup\{k\}\}}\cdot 1_{\{S_{\text{rx}}(\tau)=S_X\}}. \tag{119}$$

Again by the memorylessness of the channel and by comparing (115) and (117), inequality (88) holds with equality.

To prove (86), we notice that by (92), we can bound $\Delta Q_S^{(k)}$ by the queue length difference

$$\Delta Q_S^{(k)}(t)\leq Q_S^{(k)}(t+1)-Q_S^{(k)}(t).$$

Therefore we have

$$\lim_{m\to\infty}\frac{1}{t_m}\sum_{\tau=0}^{t_m-1}\mathsf{E}\left\{\Delta Q_S^{(k)}(\tau)\right\}$$
$$\leq\lim_{m\to\infty}\frac{1}{t_m}\left(\mathsf{E}\left\{Q_S^{(k)}(t_m)\right\}-\mathsf{E}\left\{Q_S^{(k)}(0)\right\}\right)$$
$$\leq\lim_{m\to\infty}\frac{D}{t_m}=0 \tag{120}$$

where (120) is by (114). By comparing the definition of $\Delta Q_S^{(k)}$ in (91) with the definitions of $\vec{y}$ in (116) and (117), the computed $\vec{y}$ variables must satisfy (86). The proof of Proposition 6 is complete.

Appendix G
Proof of Lemma 3

We prove Lemma 3 by showing that Lemma 3 holds for each iteration of $(k_0,i_0)\in\mathcal{N}_0$ in Algorithm 1. Since changing the order of columns for all the matrices $\mathbf{U}^{(k)}$ and $\mathbf{V}^{(k)}$ is equivalent to changing the labels of arrival packets, Lemma 3 holds after moving the column of index (k_0,i_0) to the left in Line 7 of Algorithm 1.

Recall that $\mathcal{N}_0$ is the collection of the old (k,i) that no longer appears in any of the headers in the queue; and $\mathbf{y}(t)$ is the linear combination of row vectors in $\mathbf{U}^{(k)}$, $k\in[K]$, where $\mathbf{U}^{(k)}$ is the matrix derived from the columns that still in the $\mathbf{V}^{(k)}$ being considered. Since after the column swap in Line 7 the first column is $(k_0,i_0)\in\mathcal{N}_0$, we can represent $\mathbf{y}(t) = \begin{bmatrix}0 & \tilde{\mathbf{y}}(t)\end{bmatrix}$. If the first column (k_0,i_0) is also a zero column in $\mathbf{V}^{(k)}$ then it is clear that Lemma 3 holds after removing (k_0,i_0) in both $\mathbf{U}^{(k)}$ and $\mathbf{V}^{(k)}$. On the other hand, if the first column is not a zero column, we apply Gaussian elimination to $\mathbf{V}^{(k)}$, which does not alter the row space $\langle\mathbf{V}^{(k)}\rangle$. We can then write the resulting $\tilde{\mathbf{V}}^{(k)}$ and the corresponding $\tilde{\mathbf{U}}'^{(k)}$ by

$$\mathbf{U}'^{(k)}=\begin{bmatrix}\mathbf{0} & \tilde{\mathbf{U}}'^{(k)}\end{bmatrix} \text{ and } \mathbf{V}^{(k)}=\begin{bmatrix}1 & \mathbf{v}\\ \mathbf{0} & \tilde{\mathbf{V}}^{(k)}\end{bmatrix}.$$

Therefore $\mathbf{y}(t) \in \langle \mathbf{U}'^{(k)} \rangle \oplus \langle \mathbf{V}^{(k)} \rangle$ if and only if there exists a coding vector $\mathbf{c} \triangleq [\mathbf{c}_1, \gamma, \mathbf{c}_2]$ such that

$$\mathbf{y}(t) = \begin{bmatrix} 0 & \tilde{\mathbf{y}}(t) \end{bmatrix} = \mathbf{c} \begin{bmatrix} \mathbf{U}'^{(k)} \\ \mathbf{V}^{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{c}_1 & \gamma & \mathbf{c}_2 \end{bmatrix} \begin{bmatrix} \mathbf{0} & \tilde{\mathbf{U}}'^{(k)} \\ 1 & \mathbf{v} \\ \mathbf{0} & \tilde{\mathbf{V}}^{(k)} \end{bmatrix}$$
$$= \begin{bmatrix} \gamma & \mathbf{c}_1 \tilde{\mathbf{U}}'^{(k)} + \gamma \mathbf{v} + \mathbf{c}_2 \tilde{\mathbf{V}}^{(k)} \end{bmatrix}.$$

Examining the above equality then leads to $\gamma = 0$ and

$$\tilde{\mathbf{y}}(t) = \begin{bmatrix} \mathbf{c}_1 & \mathbf{c}_2 \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{U}}'^{(k)} \\ \tilde{\mathbf{V}}^{(k)} \end{bmatrix} \in \langle \tilde{\mathbf{U}}'^{(k)} \rangle \oplus \langle \tilde{\mathbf{V}}^{(k)} \rangle. \qquad (121)$$

Therefore Lemma 3 also holds after removing the first row and first column of $\mathbf{V}^{(k)}$.

Let $\mathbf{V}_{\text{ref}}^{(k)}$ be the row-echelon form of $\mathbf{V}^{(k)}$ after removing all the zero rows. Since $\langle \mathbf{V}_{\text{ref}}^{(k)} \rangle = \langle \mathbf{V}^{(k)} \rangle$, Lemma 3 holds after Line 14 of Algorithm 1. The proof of Lemma 3 is thus complete.

References

[1] T. M. Cover, "Comments on broadcast channels," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2524–2530, Oct. 1998.

[2] P. Bergmans, "Random coding theorem for broadcast channels with degraded components," *IEEE Trans. Inf. Theory*, vol. 19, no. 2, pp. 197–207, Mar. 1973.

[3] Y. Wu, "Broadcasting when receivers know some messages a priori," in *Proc. IEEE Int. Symp. Inform. Theory*, Jun. 2007, pp. 1141–1145.

[4] M. A. Maddah-Ali and D. Tse, "Completely stale transmitter channel state information is still very useful," *IEEE Trans. Inf. Theory*, vol. 58, no. 7, pp. 4418–4431, Jul. 2012.

[5] C.-C. Wang, "On the capacity of 1-to-K broadcast packet erasure channels with channel output feedback," *IEEE Trans. Inf. Theory*, vol. 58, no. 2, pp. 931–956, Feb. 2012.

[6] L. Georgiadis and L. Tassiulas, "Broadcast erasure channel with feedback - capacity and algorithms," in *Proc. Workshop on Network Coding, Theory, and Applications (Netcod)*, Jun. 2009, pp. 54–61.

[7] C.-C. Wang and J. Han, "The capacity region of two-receiver multiple-input broadcast packet erasure channels with channel output feedback," *IEEE Trans. Inf. Theory*, vol. 60, no. 9, pp. 5597–5626, Sept 2014.

[8] J. Han and C.-C. Wang, "General capacity region for the fully connected three-node packet erasure network," *IEEE Trans. Inf. Theory*, vol. 62, no. 10, pp. 5503–5523, Oct 2016.

[9] R. Y. Chang, S.-J. Lin, and W.-H. Chung, "Symbol and bit mapping optimization for physical-layer network coding with pulse amplitude modulation," *IEEE Trans. Wireless Commun.*, vol. 12, no. 8, pp. 3956–3967, August 2013.

[10] C.-H. Chang, R. Y. Chang, and Y.-C. Huang, "A comparative analysis of secrecy rates of wireless two-way relay systems," in *Proc. IEEE GLOBECOM*, Dec 2015, pp. 1–6.

[11] I.-H. Hou, V. Borkar, and P. R. Kumar, "A theory of qos for wireless," in *Proc. IEEE INFOCOM*, April 2009, pp. 486–494.

[12] I.-H. Hou and P. R. Kumar, "Utility maximization for delay constrained qos in wireless," in *Proc. IEEE INFOCOM*, March 2010, pp. 1–9.

[13] I.-H. Hou, "Scheduling heterogeneous real-time traffic over fading wireless channels," *IEEE/ACM Trans. Netw.*, vol. 22, no. 5, pp. 1631–1644, Oct 2014.

[14] W.-C. Kuo and C.-C. Wang, "Robust and optimal opportunistic scheduling for downlink two-flow network coding with varying channel quality and rate adaptation," *IEEE/ACM Trans. Netw.*, vol. 25, no. 1, pp. 465–479, Feb 2017.

[15] X. Li, C.-C. Wang, and X. Lin, "Inter-session network coding schemes for 1-to-2 downlink access-point networks with sequential hard deadline constraints," *IEEE/ACM Trans. Netw.*, vol. 25, no. 1, pp. 624–638, Feb 2017.

[16] L. Deng, C.-C. Wang, M. Chen, and S. Zhao, "Timely wireless flows with general traffic patterns: Capacity region and scheduling algorithms," *IEEE/ACM Trans. Netw.*, vol. 25, no. 6, pp. 3473–3486, Dec 2017.

[17] M. Gatzianas, L. Georgiadis, and L. Tassiulas, "Multiuser broadcast erasure channel with feedback–capacity and algorithms," *IEEE Trans. Inf. Theory*, vol. 59, no. 9, pp. 5779–5804, Sep. 2013.

[18] S. Athanasiadou, M. Gatzianas, L. Georgiadis, and L. Tassiulas, "Stable XOR-based policies for the broadcast erasure channel with feedback," *IEEE/ACM Trans. Netw.*, vol. 24, no. 1, pp. 476–491, Feb. 2016.

[19] W. Nam, S.-Y. Chung, and Y. H. Lee, "Capacity of the gaussian two-way relay channel to within $\frac{1}{2}$ bit," *IEEE Trans. Inf. Theory*, vol. 56, no. 11, pp. 5488–5494, Nov 2010.

[20] S. Zhang, S. C. Liew, and P. P. Lam, "Hot topic: Physical-layer network coding," in *Proc. Mobile Computing and Networking (MobiCom)*, ser. MobiCom '06. New York, NY, USA: ACM, 2006, pp. 358–365.

[21] B. Nazer and M. Gastpar, "Compute-and-forward: Harnessing interference through structured codes," *IEEE Trans. Inf. Theory*, vol. 57, no. 10, pp. 6463–6486, Oct 2011.

[22] A. Eryilmaz, A. Ozdaglar, M. Medard, and E. Ahmed, "On the delay and throughput gains of coding in unreliable networks," *IEEE Trans. Inf. Theory*, vol. 54, no. 12, pp. 5511–5524, Dec 2008.

[23] L. Yang, Y. E. Sagduyu, J. Zhang, and J. H. Li, "Deadline-aware scheduling with adaptive network coding for real-time traffic," *IEEE/ACM Transactions on Networking*, vol. 23, no. 5, pp. 1430–1443, Oct 2015.

[24] Y. E. Sagduyu, L. Georgiadis, L. Tassiulas, and A. Ephremides, "Capacity and stable throughput regions for the broadcast erasure channel with feedback: An unusual union," *IEEE Trans. Inf. Theory*, vol. 59, no. 5, pp. 2841–2862, May 2013.

[25] A. Papadopoulos and L. Georgiadis, "Broadcast erasure channel with feedback and message side information, and related index coding result," *IEEE Trans. Inf. Theory*, vol. 63, no. 5, pp. 3161–3180, May 2017.

[26] Z. Li, C. He, and S. Yang, "On the capacity of the two-user erasure broadcast channel with mixed csit," in *Proc. IEEE Information Theory Workshop (ITW)*, Nov 2017, pp. 499–503.

[27] C.-H. Chang and C.-C. Wang, "A new capacity-approaching protocol for general 1-to-k broadcast packet erasure channels with ACK/NACK," in *Proc. IEEE Int. Symp. Inform. Theory*, June 2017, pp. 201–205.

[28] C.-C. Wang, D. Koutsonikolas, Y. C. Hu, and N. Shroff, "FEC-based AP downlink transmission schemes for multiple flows: Combining the reliability and throughput enhancement of intra- and inter-flow coding," *Perform. Eval.*, vol. 68, no. 11, pp. 1118–1135, Nov. 2011.

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

[30] A. Bondy and U. Murty, *Graph Theory*, ser. Graduate Texts in Mathematics. Springer London, 2011.

[31] M. J. Neely and R. Urgaonkar, "Opportunism, backpressure, and stochastic optimization with the wireless broadcast advantage," in *Proc. Asilomar Conference on Signals, Systems and Computers*, Oct 2008, pp. 2152–2158.