Sending Perishable Information: Coding Improves Delay-Constrained Throughput Even for Single Unicast

Chih-Chun Wang, *Senior Member, IEEE,* and Minghua Chen, *Senior Member, IEEE*

***Abstract*—This work considers network communications under a hard *timeliness constraint*, where a source node streams perishable information to a destination node over a directed acyclic graph subject to a hard delay constraint. Transmission along any edge incurs unit delay, and it is required that every information bit generated at the source at the *beginning* of time t to be received and recovered by the destination at the *end* of time $t + \mathsf{D} - 1$ where $\mathsf{D} > 0$ is the maximum allowed end-to-end delay. We study the corresponding *delay-constrained unicast capacity* problem.**

This work presents the first example showing that network coding (NC) can achieve strictly higher delay-constrained throughput than routing even for the single unicast setting and the NC gain can be arbitrarily close to 2 in some instances. This is in sharp contrast to the delay-unconstrained ($\mathsf{D} = \infty$) single-unicast case where the classic min-cut/max-flow theorem implies that coding cannot improve throughput over routing. Motivated by the above findings, a series of investigation on the delay-constrained capacity problem is also made, including (i) an equivalent multiple-unicast representation based on a time-expanded graph approach, (ii) a new delay-constrained capacity upper bound and its connections to the existing routing-based results [Ying, *et al.* 2011], (iii) an example showing that the penalty of using random linear NC can be unbounded, and (iv) a counter example of the tree-packing Edmonds' theorem in the new delay-constrained setting. Built upon the time-expanded graph approach, we also discuss how our results can be readily extended to cyclic networks. Overall, our results suggest that delay-constrained communication is fundamentally different from the well-understood delay-unconstrained one and call for investigation participation.

***Index Terms*—Network coding; max-flow/min-cut; delay-constrained communications; single unicast; network information theory.**

I. Introduction

Consider a network modeled as a directed acyclic graph G, for which each edge has a capacity constraint and incurs a unit transmission delay. A link with long delay is modeled as multiple edges in tandem, each with unit delay. Except when specified otherwise, we consider exclusively a delay-constrained single-unicast scenario where a single source node, denoted as s, streams perishable information to a single destination node, denoted as d, over the graph G. Every information bit generated at s at the *beginning* of time t has to be received and recovered by d by the *end* of time $t + \mathsf{D} - 1$. Namely, the maximum allowed end-to-end communication delay of any packet is $(t + \mathsf{D} - 1) - t + 1 = \mathsf{D}$, where the value of D is specified by the delay requirement of the applications. For easier reference, we use "at time t" to refer to "at the *beginning* of time t". Since the end of time $t+\mathsf{D}-1$ is equivalent to the beginning of time $t + \mathsf{D}$, our setting simply means that each packet generated at time t needs to be decoded at time $t + \mathsf{D}$.

In this paper, we formally define and study the *delay-constrained unicast capacity* problem, which characterizes the maximum rate at which s can stream perishable information to d subject to the delay constraint D.

The problem is important for delay-sensitive multimedia communication systems, and for delivering real-time control messages for cyber-physical systems. For example, in video conferencing, the system designer may want to maximize video throughput (thus video quality) subject to a video-delivery delay constraint of 250ms, to ensure interactive conferencing experience [3]. Similarly, in cyber-physical systems, time-critical applications impose latency constraints within which data or control messages must reach their targeting entities [2]. In general, an optimal delay-constrained communication scheme needs to decide the optimal routes of the information flow *in space* in order to fully utilize all the link capacity resources, while simultaneously tracking the delay of individual packets *in time* to ensure the packets can arrive at d and the information can be recovered before expiration. The design problem becomes even more involved when we move away from the traditional *routing paradigm* (also known as the store-and-forward paradigm) and allow for *network coding (NC)* [1], [31] at intermediate nodes that intelligently mix the information content in the packets before forwarding them. Such a 3-way coupling among space, time, and NC choices creates a unique challenge and our understanding of delay-constrained network capacity is still nascent. In the following, we briefly illustrate some unique phenomena of the delay-constrained setting.

When D is sufficiently large (*e.g.*, larger than the end-to-end delay of the longest path between s and d), any communication

Manuscript received October 20, 2015; revised October 1, 2016.

This work was supported in part by NSF grants CCF-0845968, CNS-0905331, ECCS-1407603, CCF-1422997, and CCF-1618475; and by National Basic Research Program of China (Project No. 2012CB315904 and 2013CB336700), and two grants from the University Grants Committee of the Hong Kong Special Administrative Region, China (Area of Excellence Project No. AoE/E-02/08 and General Research Fund Projects No. 411011 and 14209115).

Part of this work was presented in ISIT 2014.

C.-C. Wang is with the School of Electrical and Computer Engineering, Purdue University, Indiana, IN, 47906 USA e-mail: chihw@purdue.edu.

M. Chen is with the Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong e-mail: minghua@ie.cuhk.edu.hk.

scheme can always meet the delay constraint. In this case, the delay-unconstrained (since $\mathsf{D} = \infty$) single-unicast capacity can be characterized by the classic max-flow/min-cut theorem, and an optimal routing solution can be obtained in polynomial time using the Edmonds-Karp algorithm [7] (an improvement of the Ford-Fulkerson algorithm [11]), the push-&-relabel algorithm [13], simple linear programming (LP) solvers [42], [44], and a network-coding-based sounding approach [36]. See [12] for further references. Since optimal routing already achieves the capacity, *i.e.*, the min-cut value, NC cannot improve throughput over optimal routing when there is only one unicast flow in the network.

The story changes completely when D is small (*i.e.*, when the delay constraint is active). For example, the delay-constrained unicast routing capacity has to be computed by the concept of *soft edge-cuts* [24], [45], which is different from the standard graph-theoretic notion of edge cuts. Also, as will be illustrated in Section II-C, there are some simple network instances for which optimal routing can achieve strictly higher delay-constrained throughput than random linear network coding (RLNC), a sharp contrast to the delay-unconstrained case in which both RLNC and optimal routing can achieve the single-unicast capacity [18].

Overall, we observe that the landscape of delay-constrained unicast is fundamentally different from the well-understood delay-unconstrained one. In this paper, we study the delay-constrained unicast capacity problem and make the following contributions:

Firstly, the single-unicast delay-constrained network capacity problem is officially formulated, which allows for rigorous future investigation. One immediate result of the delay-constrained single-unicast formulation is an equivalent *delay-unconstrained* multiple-unicast time-expanded network representation.

Secondly, this work shows for the first time in the literature that for delay-constrained traffic, NC can achieve strictly higher throughput than optimal routing *even for single unicast* and the NC gain[1] can be arbitrarily close to 2.

Such a result is interesting in the following sense. Most of the Internet traffic is unicast. One of the fundamental results in NC is that routing achieves the single-unicast capacity when there is no delay constraint. This implies that to capitalize the NC benefits for delay-insensitive unicast traffic, one has to perform NC over multiple coexisting flows, the so-called inter-flow NC. It is known that designing the optimal inter-flow NC scheme is a notoriously hard problem, see [4], [16], [20], [22], [26]–[28], [32], [37]–[41] and the references therein. What exacerbates the problem is that even if we can design an optimal inter-flow NC scheme in a theoretic setting, in practice inter-flow NC requires additional coordinations among participating flows, including the tasks of hand-shaking, synchronization, joint buffer management, *etc*. Our result suggests that one may use NC to improve the performance of delay-sensitive traffic over optimal routing without any coordination among coexisting network flows.

Thirdly, based on a time-expanded graph approach, we propose a new upper bound on the delay-constrained unicast NC capacity, which provides deeper understanding to the overall delay-constrained network communication problem and sheds further insights to the existing routing-based delay-constrained results in [45] through the concept of *integrality gap*.

Fourthly, various aspects of the delay-constrained capacity are investigated, including the potentially unbounded penalty[2] of RLNC over routing; and a revisit of the Edmonds' tree-packing results [6] for the delay-constrained setting.

This paper is organized as follows. Section II compares this work to the existing work on network-coding-based delay minimization, and summarizes the existing results on routing (store-and-forward) based solutions. Section III formulates the delay-constrained capacity problem. Section IV states the main results of this work, including (i) An equivalent time-expanded multiple unicast formulation; (ii) the NC gain over routing; and (iii) A new delay-constrained capacity upper bound. Section V contains various other results on delay-constrained capacity, including (iv) The unboundedness of the penalty of using RLNC; and (v) An example showing that Edmonds' tree packing theorem no longer holds for the delay-constrained setting. Some remarks on generalizing the above results from acyclic to cyclic networks are also provided in Section V. Section VI concludes this work. In this paper, we formally present our discovery in the main body and leave the majority of the proofs in the appendices so that the logic flow of the statements is uninterrupted.

II. Comparison To The Existing Work

A. Existing results on delay-related network coding

1) Delay minimization of network coding: Decoding delay of NC is a very well studied problem, see [5], [9], [10] and the references therein. Most of the existing works along this line focus on how to minimize the decoding delay when using NC to attain the best possible throughput for delay-insensitive traffic. Namely, attaining the absolute optimal throughput is of the highest priority, and minimizing the delay is to ensure that the extra (decoding) delay incurred by NC is not excessive. Many of the existing results also focus on 1-hop networks with *random packet erasure.*

In contrast with the existing throughput-centric delay-minimization studies [5], [10], this work is delay-centric. Namely, we consider a hard delay constraint such that any packets that experience delay longer than D time slots are deemed useless and discarded. With the highest priority being the delay constraint, we focus on maximizing the throughput over any given *error-free multi-hop network.* This approach is a significant departure from the existing works on minimizing NC decoding delay.

In a way similar to this work, [9] studies the delay-capacity tradeoff. On the other hand, [9] focuses exclusively on the single-multicast scenario, assumes there is no link

[1] Intuitively, the NC gain of a given network instance is defined as the ratio between the optimal NC capacity and the optimal routing rate. A formal definition is given later in (12).

[2] Intuitively, the RLNC penalty of a network instance is defined as the ratio between the optimal routing rate and the optimal RLNC rate. A formal definition is given later in (27).

propagation delay, and studies the *decoding delay* related to the (large) underlying finite field size. For comparison, this work studies the single-unicast setting and focuses exclusively on the propagation delay of each link.

2) Delay-aware network code designs: Another line of network coding studies on delay is the delay-aware network code design, e.g., [8], [14], [25], [29], [35] and the references therein. In general, the goal along this line of work is to develop optimal network codes that maximize the throughput (achieving the min-cut value) while assuming that packets that arrive with long delay is as useful as packets with short delay. The difference is that the optimal code design now has to take into account the delay incurred in each edge/link in a way related to the traditional convolutional codes. The ring of rational power series is often used as the algebraic foundation of the corresponding network code design and study.

Among them, [14] studies the delay-constrained decodability problem, an identical setting as of this work. Specifically, [14] answers the question that for any given linear network code (with all the local encoding kernels explicitly specified), what is the necessary and sufficient algebraic condition that determines whether a given network code can sustain a delay-constrained throughput R. Therefore, for any given linear network code, one can use the results in [14] to verify whether it can achieve the targeted delay-constrained rate. For completeness, we will restate this important algebraic result [14] in Appendix I.

For comparison, instead of passively verifying the delay-constrained performance of any given linear network code, our work studies the best possible linear/non-linear network codes for a given delay constraint and compares it to that of the existing routing-based solutions, see Section II-B. We are particularly interested in exploring the graph-theoretic and information-theoretic connections, which is different from the algebraic approach in [14].

3) Time-stamped communications with infinite backlog: There also exist some works on infinite-backlog but timely delivery [32], [33]. In those settings, all the packets are available before the transmission starts and each packet has a fixed expiration time with uniform spacing. In terms of practical applications, our setting focuses more on the video conferencing scenarios where each message packet, once generated, needs to be delivered within a delay constraint. For comparison, the results in [32], [33] are more related to on-demand movie playback where all packets have already been stored in the data center and the design goal is to ensure smooth sequential packet delivery (thus the expiration time with uniform spacing). The main difference is that the expiration time of each packet is now predetermined and does not depend on when the packet is injected into the network.

For example, under the settings of [32], [33] it is a viable strategy to *buffer the packets* in certain ways to ensure smooth playback. In contrast, buffering packets is a very poor strategy in our delay-constrained setting since buffering packets will significantly increase the end-to-end delay and thus decrease the delay-constrained throughput.

B. Existing and some new results when only store-&-forward is allowed

We model the network as a finite directed acyclic graph $G = (V, E)$, where V is the node set and E is the edge set. We use $\mathrm{In}(v)$ and $\mathrm{Out}(v)$ to denote the collections of the incoming and outgoing edges of v, respectively. For any $e = (u, v) \in E$, we define $\mathsf{tail}(e) \triangleq u$ and $\mathsf{head}(e) \triangleq v$. Each e has a capacity constraint c_e and incurs unit delay. Links with long delay are thus modeled as a path of multiple edges.

Consider the store-and-forward paradigm. With delay constraint D, any packet that traverses from s to d through a path longer than D hops is deemed useless. Without loss of generality, we assume $\mathsf{D} \leq |E|$. Otherwise, the problem collapses back to the classic delay-unconstrained problem since all paths have length $\leq |E|$. We also assume $\mathrm{In}(s) = \emptyset$ and $\mathrm{Out}(d) = \emptyset$. For any integer k, we define $[1, k] \triangleq \{1, 2, \cdots, k\}$ and define $[1, \infty)$ as the set of positive integers.

Let $\mathcal{P}_{\mathsf{D}}$ denote the collection of all s-to-d paths of length "$\leq \mathsf{D}$ hops." Obviously $\mathcal{P}_{\mathsf{D}}$ is finite. When only store-and-forward is allowed, the largest delay-constrained routing capacity, denoted by R^*_{route}, can be computed by the following LP problem:[3]

$$R^*_{\text{route}} \triangleq \max_{\{x_P \geq 0 : P \in \mathcal{P}_{\mathsf{D}}\}} \sum_{P \in \mathcal{P}_{\mathsf{D}}} x_P \tag{1}$$

$$\text{subject to} \quad \forall e \in E, \sum_{P : P \ni e, P \in \mathcal{P}_{\mathsf{D}}} x_P \leq c_e, \tag{2}$$

which consists of $|E|$ inequalities and $|\mathcal{P}_{\mathsf{D}}|$ non-negative variables $\{x_P\}$ indexed by $P \in \mathcal{P}_{\mathsf{D}}$, where each x_P represents the communication rate assigned to a delay-respecting path P. The objective (1) is the sum of the throughput sent over the $|\mathcal{P}_{\mathsf{D}}|$ paths, and (2) imposes that the sum rate of all paths using an edge e must not exceed c_e. However, since $|\mathcal{P}_{\mathsf{D}}|$ grows exponentially with respect to $|G| \triangleq |V| + |E|$, the above LP characterization is not easily computable for large networks. To address the above concern of complexity, [45] proposes an equivalent alternative LP that can compute the optimal value of (1)–(2) in polynomial time of $|G|$. Since the results in [45] were stated without detailed proofs, we provide the details of [45] and the corresponding proofs in Appendix A.

By converting (1)–(2) to its dual problem, R^*_{route} can also be computed by the following cut-based LP problem.

$$\min_{\{y_e \geq 0 : e \in E\}} \sum_{e \in E} y_e c_e \tag{3}$$

$$\text{subject to} \quad \forall P \in \mathcal{P}_{\mathsf{D}}, \sum_{e : e \in P} y_e \geq 1. \tag{4}$$

One way to interpret this dual problem is to treat the variable y_e as the "cut strength" of a given edge e such that the cumulative cut strength experienced by each path is no less than 1.

Note that if we replace $\mathcal{P}_{\mathsf{D}}$ by $\mathcal{P}_\infty$, the latter of which contains *all* paths regardless of their lengths, then one can prove, by the standard proof techniques used in the max-flow/min-cut theorem, that for any given network instance, one of the

[3] It is sometimes called the hop-count-constrained max-flow problem.

minimizing $\{y_e^*\}$ of (3)–(4) satisfies $y_e^* \in \{0,1\}, \forall e \in E$. As a result, we can impose the integrality constraint $y_e \in \{0,1\}$ to the LP problem (3)–(4) without changing the objective value. Solving (3)–(4) with the integrality constraint is equivalent to finding the minimum edge cut (those e with $y_e^* = 1$). The equivalence between (1)–(2) and (3)–(4) with integrality constraints is the well-known max-flow/min-cut theorem.

However, with $\mathcal{P}_{\mathsf{D}}$ for some finite D, the minimizing $\{y_e^*\}$ can sometimes be fractional. Therefore, the delay-constrained R^*_{route} is now characterized by some kind of *soft* min-cut, a unique feature of the delay-constrained setting. Examples of y_e^* being fractional can be found in the end of Section IV-B and in [24].

Also observe that there are $|\mathcal{P}_{\mathsf{D}}|$ inequalities in (4), which grows exponentially with respect to $|G|$. In Appendix A, we have also derived a new polynomial-time version of the dual problem (3)–(4).

C. A simple comparison to RLNC

To illustrate the change of landscape when focusing on a delay-constrained setting, we provide a simple example for which the RLNC scheme [18] is no longer throughput optimal[4] when there is a hard delay constraint.

Consider the network in Fig. 1(a). The min-cut value between s and d is 2, which implies the existence of (at least) one pair of edge-disjoint paths (EDPs). There are actually two possible pairs of EDPs, see Figs. 1(b) and 1(c), respectively. Assume each edge incurs a unit delay. If there is no delay constraint, we can sustain throughput 2 by *routing* the packets either through Fig. 1(b) or through Fig. 1(c). However, with delay constraint $\mathsf{D} = 3$, only the two paths in Fig. 1(c) can be used to transmit information at rate 2. For comparison, one path in Fig. 1(b) has 4 hops, and the information transmitted along that path will expire before arriving at d.

We now apply RLNC to Fig. 1(a) while assuming a sufficiently large finite field $\mathsf{GF}(q)$ is used, say $\mathsf{GF}(3)$. At each time t, we send two information packets $X_t \in \mathsf{GF}(q)$ and $Y_t \in \mathsf{GF}(q)$ along the edges (s, v_2) and (s, v_1), respectively. Due to the unit-delay incurred in edge (s, v_1), at time t, we can send Y_{t-1} along (v_1, v_2). Node v_2 can now perform RLNC. We can assume, without loss of generality, that at time t node v_2 sends $M^{(t)}_{v_2 d} = X_{t-1} + Y_{t-2}$ and $M^{(t)}_{v_2 v_3} = X_{t-1} + 2Y_{t-2}$ along (v_2, d) and (v_2, v_3), respectively. Following similar derivation, by the *end* of time t, destination d should have received $M^{(t)}_{v_3 d} = X_{t-2} + 2Y_{t-3}$ and $M^{(t)}_{v_2 d} = X_{t-1} + Y_{t-2}$.

Since s starts to send Y_t and X_t at time t for all $t \geq 1$, we set $X_t = Y_t = 0$ for all $t \leq 0$. From the above derivation, by the end of time 3, d has received $M^{(3)}_{v_3 d} = X_1 + 2Y_0 = X_1$ and $M^{(3)}_{v_2 d} = X_2 + Y_1$. Recall that $\mathsf{D} = 3$. Therefore, d is interested in decoding both X_1 and Y_1, which were sent by s at the "beginning" of time 1 (3 slots earlier). One can verify that *knowing $M^{(3)}_{v_3 d} = X_1$ and $M^{(3)}_{v_2 d} = X_2 + Y_1$ is not sufficient for d to decode the desired X_1 and Y_1 since the value of Y_1 in $M^{(3)}_{v_2 d}$ is now "corrupted" by the future packet X_2 that has not been decoded yet.* Actually, even when time progresses, d is not able to decode both X_t and Y_t by the end of time slot $(t + \mathsf{D} - 1) = t + 2$ for any $t > 1$. One can prove that the RLNC throughput of this example is 1, which is strictly less than the routing capacity 2.

The above example shows that even the most basic result (e.g., RLNC being throughput optimal) in the delay-unconstrained setting may not hold once we consider delay constraints. This work aims to re-examine a couple of most fundamental single-unicast capacity problems while taking into account hard delay constraints.

III. Problem Formulation

We follow the network model introduced in Section II-B. Namely, the network is modeled as a directed acyclic graph $G = (V, E)$ with each edge $e \in E$ having capacity c_e and incurring unit delay. The amount of data is measured in *packets*, where each packet is assumed to be chosen from $[0, q-1]$, or equivalently each packet has $\log_2(q)$ bits. We assume that q is a sufficiently large fixed number and in the linear network coding literature q is sometimes assumed to be a power of a prime. We can now define the delay-constrained capacity of transmitting from a single source s to a destination d.

Definition 1: Given any network $G = (V, E)$ and any scalar $R > 0$, define the *message symbol set* $\mathcal{X}$ and the *edge symbol set* $\mathcal{M}_e$ by

$$\mathcal{X} \triangleq \left\{1, \cdots, \left\lceil 2^{\log_2(q) R} \right\rceil \right\}$$
$$\mathcal{M}_e \triangleq \left\{1, \cdots, \left\lfloor 2^{\log_2(q) c_e} \right\rfloor \right\}.$$

We say the delay-constrained rate R is feasible,[5] if for any given time span T, we have (i) T message symbols $X_t \in \mathcal{X}, \forall t \in [1, T]$; (ii) $(T + \mathsf{D} - 1) \cdot |E|$ edge-encoding functions: For all $t \in [1, T + \mathsf{D} - 1]$ and $e \in E$, if $\mathsf{tail}(e) = s$ then[6]

$$M^{(t)}_e = f_{e,t}(\{X_\tau : \tau \in [1, t]\}) \in \mathcal{M}_e, \tag{5}$$

and if $\mathsf{tail}(e) \neq s$ then

$$M^{(t)}_e = f_{e,t}(\{M^{(\tau)}_{\tilde{e}} : \tilde{e} \in \mathrm{In}(\mathsf{tail}(e)), \tau \in [1, t-1]\}) \in \mathcal{M}_e; \tag{6}$$

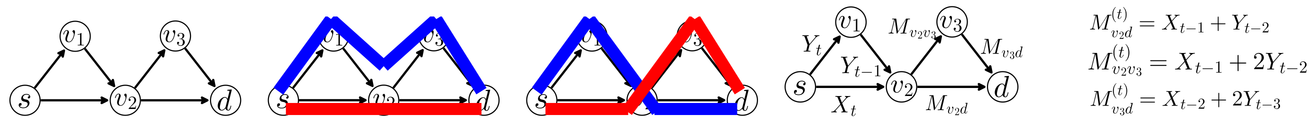
and (iii) T decoding functions: $\forall t \in [1, T]$,

$$\hat{X}_t = f_{\mathsf{DEC},t}(\{M^{(\tau)}_e : e \in \mathrm{In}(d), \tau \in [1, t + \mathsf{D} - 1]\}) \in \mathcal{X}; \tag{7}$$

[4]Since RLNC is a randomized solution, a more rigorous question to ask is how large is the probability that RLNC achieves the optimal rate. We use the statement "RLNC is suboptimal" in the sense that Prob(RLNC achieves optimal rate) is quite small when a large finite field is used.

[5]The network model considered herein is noiseless. For noisy networks, the traditional (ϵ, n) terminology may be needed to formally define the delay-constrained capacity.

[6]Here we use the convention that $X_t = 0$ for $t \in [T + 1, T + \mathsf{D} - 1]$.



(a) The network topology (b) The first pair of EDPs (c) The second pair of EDPs (d) The corresponding RLNC solution

Fig. 1. A simple example with $\mathsf{D} = 3$ that demonstrates the key differences of the delay-constrained setting.

such that $X_t = \hat{X}_t$ for all $t \in [1, T]$ for all possible[7] realization of $X_t = x_t \in \mathcal{X}$.

Definition 2: The delay-constrained capacity R^*_{NC} is the supremum of all feasible R.

It is worth emphasizing that by the above definitions R^*_{NC} is a function of the packet size q. Since routing can be considered as a special example of network coding, one can easily prove the following inequality

$$R^*_{\text{route}} \leq \lim_{q \to \infty} R^*_{\text{NC}}. \tag{8}$$

where R^*_{route} is defined in (1). On the other hand, it is possible to have $R^*_{\text{NC}} < R^*_{\text{route}}$ for some fixed small q. The reason is as follows. The definition of R^*_{route} in (1)–(2) is the maximum routing capacity in a *splittable flow setting* since the values $\{x_P : P \in \mathcal{P}_D\}$ are allowed to be fractional. On the other hand, for finite q, the definition of R^*_{NC} can be viewed as the maximum NC capacity in an *unsplittable setting* where the size of q controls the granularity of information splitting. Since splitting the flows can increase the achievable throughput, we may have $R^*_{\text{NC}} < R^*_{\text{route}}$ when q is small. On the other hand, when $q \to \infty$, the R^*_{NC} now corresponds to a splittable setting and we thus have (8).

IV. Main Results

In this section, we state our main results: (i) An equivalent time-expanded multiple unicast formulation; (ii) The NC gain over routing; and (iii) A new delay-constrained capacity upper bound.

A. An equivalent multiple-unicast formulation

For any finite network $G = (V, E)$, the time-expanded graph over a time horizon $t = [1, \tau]$ can be defined as follows.

Definition 3 ([30, Section IV]): For any fixed integer $\tau \geq 1$ and any fixed network $G = (V, E)$, the time-expanded graph $\overline{G}^{[\tau]} = (\overline{V}^{[\tau]}, \overline{E}^{[\tau]})$ contains $\tau \cdot |V|$ nodes, each node being labeled by a pair $[v, t]$ for all $v \in V$ and $t \in [1, \tau]$. The edge set $\overline{E}^{[\tau]}$ can be described/constructed as follows.

1) For each $e = (u, v) \in E$ with the corresponding capacity being c_e and for each $t \in [1, \tau - 1]$, there exists an edge $\overline{e}^{[t]} \in \overline{E}^{[\tau]}$ that connects $[u, t]$ and $[v, t+1]$. The capacity of the edge $\overline{e}^{[t]}$ is set to c_e.
2) For each node $u \in V$ and $t \in [1, \tau - 1]$, there exists an edge $\overline{e}^{[t]} \in \overline{E}^{[\tau]}$ that connects $[u, t]$ and $[u, t + 1]$. The capacity of the edge $\overline{e}^{[t]}$ is set to
$$t \cdot \left(\sum_{\tilde{e} \in \text{In}(u)} c_{\tilde{e}} \right).$$
if $u \neq s$; and is set to $t \cdot \log_q(\lceil q^R \rceil)$ if $u = s$. The capacity of $\overline{e}^{[t]}$ is to ensure that node $[u, t]$ can pass all the "external/incoming" information it has accumulated in the past (i.e., during time $[1, t]$), directly to $[u, t + 1]$.

With the above definition, we present an equivalent formulation of the delay-constrained capacity.

Proposition 1: A delay-constrained rate R is feasible if and only if for any T, the time-expanded graph $\overline{G}^{[T+\mathsf{D}]}$ can sustain simultaneously T multiple unicast flows, where each unicast flow is indexed by $t \in [1, T]$ and is from node $[s, t]$ to node $[d, t + \mathsf{D}]$ and has *zero-error rate* R.

Proof: The only if direction "$\Rightarrow$": If delay-constrained rate R is feasible in G, then for any given T we can design a feasible multiple-unicast scheme in $\overline{G}^{[T+\mathsf{D}]}$ in the following way. For any t value, we let the $[s, t]$-to-$[s, t + 1]$ edge carry all the symbols $\{X_\tau : \tau \in [1, t]\}$, and let the $[u, t]$-to-$[u, t + 1]$ edge, $u \neq s$, carry all the symbols $\{M^{(\tau)}_{\tilde{e}} : \tilde{e} \in \text{In}(u), \tau \in [1, t - 1]\}$. In this way, the $[u, t]$-to-$[v, t + 1]$ edge can compute and transmit $M^{(t)}_{(u,v)}$ described in (5) and (6), and the destination $[d, t + \mathsf{D}]$ can compute $\hat{X}_t$ described in (7). The multiple-unicast zero-error rate R is thus feasible in $\overline{G}^{[T+\mathsf{D}]}$.

The if direction "$\Leftarrow$": Suppose for any given T we can design a multiple-unicast scheme in $\overline{G}^{[T+\mathsf{D}]}$ with zero-error rate R. For all $(s, u) \in E$, we then choose the $M^{(t)}_{(s,u)}$ message in G to be the message in $\overline{G}^{[T+\mathsf{D}]}$ that is sent from $[s, t]$-to-$[u, t+1]$. Since for any given t, the values of $\{X_\tau : \tau \in [1, t]\}$ determine uniquely the message along the $[s, t]$-to-$[u, t + 1]$, the chosen $M^{(t)}_{(s,u)}$ must be of the form of (5).

For any t and any $(v, u_2) \in E$ satisfying $v \neq s$, we choose the $M^{(t)}_{(v,u_2)}$ message in G to be the message in $\overline{G}^{[T+\mathsf{D}]}$ that is sent from $[v, t]$-to-$[u_2, t + 1]$. Since the edge set

$$\{([u_1, \tau], [v, \tau + 1]) : \forall \tau \in [1, t - 1], (u_1, v) \in E\}$$

is a cut in $\overline{G}^{[T+\mathsf{D}]}$ separating $\{X_t : t \in [1, T]\}$, the chosen $M^{(t)}_{(v,u_2)}$ must be of the form of (6).

[7] The feasibility definition in this work allows the set of encoders and decoders (5)–(7) to be chosen differently when different T values are considered. In a broad sense, the role of T is similar to the traditional definition of block-length n in information theory. One alternative way of defining feasibility is to fix a set of infinitely many encoders and decoders, such that under this fixed encoder/decoder choices, any T message symbols can be extracted perfectly within a delay constraint D. That is, the same (infinite) set of encoders and decoders have to support T sequential messages for arbitrary T. We call the latter setting an *infinite-horizon setting*. It is obvious that any feasible R under an infinite-horizon setting is is also feasible under Definition 1. An interesting open question is whether these two definitions are indeed equivalent.

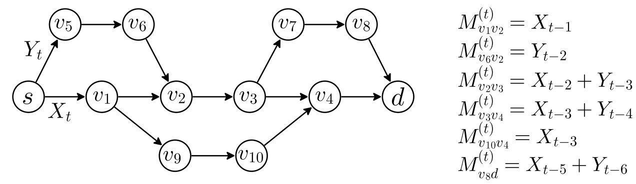


Fig. 2. A simple network with NC gain $\triangleq \frac{R^*_{\text{NC}}}{R^*_{\text{route}}} = \frac{4}{3}$, where the edge capacity $c_e = 1$ for all edges e.

For any t, we choose the decoder function $f_{\text{DEC},t}$ in G to be the decoder function in $\overline{G}^{[T+\mathsf{D}]}$ used by destination $[d, t+D]$. Since the edge set

$$\{([u,\tau],[d,\tau+1]) : \forall \tau \in [1, t+\mathsf{D}-1], (u,d) \in E\}$$

is a cut in $\overline{G}^{[T+\mathsf{D}]}$ separating $\{X_t : t \in [1,T]\}$ from destination $[d, t+\mathsf{D}]$, the chosen decoder $f_{\text{DEC},t}$ must be of the form of (7). Since the chosen encoders/decoders can support T coexisting unicast traffic in $\overline{G}^{[T+\mathsf{D}]}$ with rate R, the network coding solution in G can support delay-constrained rate R. The proof is thus complete. ■

The above connection between the delay-constrained capacity with the multiple-unicast capacity in a time-expanded network will later be used to derive a new upper bound on delay-constrained capacity. Also, being equivalent to the multiple-unicast capacity is the first hint that NC may strictly outperform routing (the store-and-forward policy) since it is known that NC can strictly enhance the multiple-unicast capacity.[8]

B. A simple network example with NC gain $= \frac{4}{3}$

Without a hard delay constraint ($\mathsf{D} = \infty$), one fundamental result of NC is

$$R^*_{\text{route}} \geq R^*_{\text{NC}} \text{ for all } q\text{; and } R^*_{\text{route}} = \lim_{q\to\infty} R^*_{\text{NC}} \tag{9}$$

for any single-unicast flow from s to d. The following example shows that (9) no longer holds for delay-constrained traffic (*i.e.*, when D is finite and small).

Consider the network in Fig. 2 with all edges having $c_e = 1$ and the delay constraint $\mathsf{D} = 6$. For ease of exposition,[9] we assume $q = 3$ and all the operations are based on finite field $\mathsf{GF}(3)$. We will show that such a network has $R^*_{\text{NC}} = 2 > R^*_{\text{route}} = 1.5$.

We first prove that $R^*_{\text{NC}} = 2$ by explicit NC construction. In Fig. 2, two packets X_t and Y_t are sent by s at time t and we assume $X_t = Y_t = 0$ for all $t \leq 0$. Since each edge incurs unit delay, we have $M^{(t)}_{v_1v_2} = X_{t-1}$ and $M^{(t)}_{v_6v_2} = Y_{t-2}$ along edges (v_1, v_2) and (v_6, v_2), respectively. We then let v_2 mix the two incoming packets and send $M^{(t)}_{v_2v_3} = X_{t-2} + Y_{t-3}$ along (v_2, v_3). By accounting for the delay incurred along the paths, we have $M^{(t)}_{v_3v_4} = X_{t-3} + Y_{t-4}$, $M^{(t)}_{v_{10}v_4} = X_{t-3}$, and $M^{(t)}_{v_8d} = X_{t-5} + Y_{t-6}$. Fig. 2 contains the summary of our NC choices thus far except for the $M^{(t)}_{v_4d}$ message. The remaining question to be answered is what is the right NC choice at node v_4?

If we perform RLNC at v_4, then v_4 will simply mix the two incoming packets together and send

$$\text{RLNC: } M^{(t)}_{v_4d} = M^{(t-1)}_{v_3v_4} + M^{(t-1)}_{v_{10}v_4} = 2X_{t-4} + Y_{t-5}. \tag{10}$$

Recall that $\mathsf{D} = 6$. Therefore, in the end of time 6, node d would like to decode both X_1 and Y_1. One can verify that in the end of time 6, node d receives $M^{(6)}_{v_8d} = X_1 + Y_0 = X_1$ and $M^{(6)}_{v_4d} = 2X_2 + Y_1$ since we assume $X_t = Y_t = 0$ for all $t \leq 0$. Similar to the example discussed in Section II-C, d cannot decode Y_1 since Y_1 is corrupted by X_2, which has not been decoded yet.

The same dilemma happens again when $t = 7$. In the end of time 7, node d would like to decode both X_2 and Y_2 and has already received $M^{(7)}_{v_8d} = X_2 + Y_1$ and $M^{(7)}_{v_4d} = 2X_3 + Y_2$. Then d can use $M^{(6)}_{v_4d} = 2X_2 + Y_1$ and $M^{(7)}_{v_8d} = X_2 + Y_1$ to decode both X_2 and Y_1. However, Y_2 still cannot be decided since $M^{(7)}_{v_4d} = 2X_3 + Y_2$ is corrupted by X_3, which has not been decoded yet.

On the other hand, we can perform the following optimal NC instead of the simple addition in (10). That is, instead of "adding" the two incoming packets, we now "subtract" $M^{(t-1)}_{v_{10}v_4}$ from $M^{(t-1)}_{v_3v_4}$:

$$\text{Optimal: } M^{(t)}_{v_4d} = M^{(t-1)}_{v_3v_4} - M^{(t-1)}_{v_{10}v_4} = Y_{t-5}. \tag{11}$$

Destination d can now decode X_1 and Y_1 from $M^{(6)}_{v_8d} = X_1 + Y_0 = X_1$ and $M^{(6)}_{v_4d} = Y_1$ within the delay constraint. An astute reader may notice that in the end of time 7, d has received $M^{(7)}_{v_8d} = X_2 + Y_1$ and $M^{(7)}_{v_4d} = Y_2$, where X_2 in $M^{(7)}_{v_8d}$ is "corrupted" by Y_1. *Nonetheless, d can remove Y_1 in the end of time 7 since d has decoded Y_1 in the end of time 6.* The above argument can be used to prove that d can decode X_t and Y_t (injected in the beginning of time t) by the end of time $t+5$, $\forall t \geq 1$. The $\mathsf{D} = 6$ constraint is met. Since min-cut$(s,d) = 2$ in Fig. 2, we have the delay-constrained NC capacity being $R^*_{\text{NC}} = 2$ packets per slot.

We then apply (3)–(4) to Fig. 2 and derive $R^*_{\text{route}} = 1.5$. The corresponding minimizing y^*_e are: $y^*_{sv_1} = y^*_{v_2v_3} = y^*_{v_4d} = 0.5$ and all other $y^*_e = 0$. The optimal routing solution in (1)–(2) will assign rate 0.5 to three different paths: $sv_5v_6v_2v_3v_4d$, $sv_1v_2v_3v_7v_8d$, and $sv_1v_9v_{10}v_4d$. This example shows that NC strictly outperforms optimal routing even for the single-unicast setting!

C. How large can the NC gain be?

In the previous example, the NC throughput gain over routing is $\frac{R^*_{\text{NC}}}{R^*_{\text{route}}} = \frac{2}{1.5}$. An interesting open question is *what is the largest NC gain in a single-unicast delay-constrained setting?* Specifically, we are interested in quantifying

$$\sup_{G \in \mathcal{G}_{\text{s-u}}, \mathsf{D} \in [1,\infty)} \mathsf{gain}_{\text{s-u}}(G, \mathsf{D}) \tag{12}$$

[8] In the multiple-unicast setting, NC is known to be beneficial for a general network. However, since the time-expanded graph is a special class of network, whether NC is still beneficial for multiple-unicasting in a time-expanded graph was not clear until the new results in Section IV-B.

[9] Our construction holds for any q for which there exists a ring of order q.

where $\mathcal{G}_{\text{s-u}}$ contains all possible network instances[10] with single-unicast (s-u) traffic, and $\mathsf{gain}_{\text{s-u}}(G,\mathsf{D}) \triangleq \frac{R^*_{\text{NC}}}{R^*_{\text{route}}}$ is the single-unicast NC gain over routing in G with delay constraint D.

For comparison, one can easily prove that *the delay-constrained NC gain can be unbounded for the single-multicast (s-m) networks and for the multiple-unicast (m-u) networks,* denoted by $\mathcal{G}_{\text{s-m}}$ and $\mathcal{G}_{\text{m-u}}$, respectively. Namely,

$$\sup_{G\in\mathcal{G}_{\text{s-m}},\forall\mathsf{D}} \mathsf{gain}_{\text{s-m}}(G,\mathsf{D}) \geq \sup_{G\in\mathcal{G}_{\text{s-m}}} \mathsf{gain}_{\text{s-m}}(G,2)=\infty \quad (13)$$

$$\sup_{G\in\mathcal{G}_{\text{m-u}},\forall\mathsf{D}} \mathsf{gain}_{\text{m-u}}(G,\mathsf{D}) \geq \sup_{G\in\mathcal{G}_{\text{m-u}}} \mathsf{gain}_{\text{m-u}}(G,3)=\infty \quad (14)$$

where the equality in (13) follows from the *combination network* construction in [34] and the equality in (14) follows from the *extended butterfly* construction in [17]. For the multiple unicast scenarios, we define $\mathsf{gain}_{\text{m-u}}(G,\mathsf{D}) \triangleq \frac{R^*_{\text{NC}}}{R^*_{\text{route}}}$ where we slightly abuse the notation and use R^*_{NC} and R^*_{route} to describe the *perfectly fair* (or equivalently *equal-rate*) capacity of all the coexisting flows, also see the feasible throughput definition in [15] and the concept of symmetric capacity in [23].

Nonetheless, the proofs of (13) and (14) cannot be applied to the single-unicast setting since they rely heavily on the fact that there are multiple destinations so that different destinations can either capitalize the diversity gain (for single multicast) or smartly cancel the interference of the other coexisting flows (for multiple unicast). These types of gains do not exist when there is only one destination[11] in the network.

Our best understanding of (12) is summarized as follows.

Proposition 2: For any $0<\epsilon<1$, there exists a network $G\in\mathcal{G}_{\text{s-u}}$ and delay constraint D satisfying

$$\frac{R^*_{\text{NC}}}{R^*_{\text{route}}} \geq 2-\epsilon.$$

Proposition 2 shows that for delay-constrained unicast traffic, NC gain over routing can be arbitrarily close to 2. A proof of Proposition 2 is provided in Appendix B.

In a broad sense, R^*_{route} characterizes the maximum number of EDPs with length $\leq\mathsf{D}$ hops, along which we can "squeeze through" R^*_{route} packets before expiration. Therefore, at least heuristically, any additional packets sent over the network (other than the original R^*_{route} packets) are either dependent or experiencing too long delay. Proposition 2 implies a rather counter-intuitive result:

> *With carefully-designed NC, those additional "useless" packets (either dependent or experiencing too long delay) can help us double the number of independent packets that can be decoded by d within the delay constraint.*

In terms of quantifying the largest possible NC gain in (12), Proposition 2 proves that

$$2 \leq \sup_{G\in\mathcal{G}_{\text{s-u}},\mathsf{D}\in[1,\infty)} \mathsf{gain}_{\text{s-u}}(G,\mathsf{D}). \quad (15)$$

In a recent work [19, Theorem 1], it has been shown that the NC gain over routing satisfies that for any fixed D

$$\sup_{G\in\mathcal{G}_{\text{s-u}}} \mathsf{gain}_{\text{s-u}}(G,\mathsf{D}) \leq 8\log_e(\mathsf{D}+1). \quad (16)$$

The gap between the lower bound (15) and the upper bound (16) is still substantial. Two observations can be made regarding the above upper and lower bound pair. Firstly, (16) goes to infinity when $\mathsf{D}\to\infty$. Therefore (16) may not be suitable to directly upper bound (12).

On the other hand, (16) suggests implicitly that when searching for network instances of large $\mathsf{gain}_{\text{s-u}}(G,\mathsf{D})$, it is critical to consider large delay requirement D and even larger network diameter[12] $\mathsf{dia}(G)$. This suggestion is consistent with our findings, for which our construction of (G,D) with $\mathsf{gain}_{\text{s-u}}(G,\mathsf{D})\to 2$ indeed has large D and large $\mathsf{dia}(G)$. Since the traditional ways of constructing network instances with large single-multicast and large multiple-unicast gains [17], [34] all have very short $\mathsf{dia}(G)$, it is less likely that those construction methods can lead to network instances with large $\mathsf{gain}_{\text{s-u}}(G,\mathsf{D})$.

D. Upper bounding the NC capacity

The delay-constrained R^*_{route} naturally serves as a lower bound on R^*_{NC}, assuming the underlying alphabet size q is sufficiently large. See (8). We now present a new upper bound on R^*_{NC}.

Proposition 3: The following integer programming (IP) problem computes an upper bound UB_{NC} on R^*_{NC} for any alphabet size q:

$$\min_{\{y_e:e\in E\}} \quad \sum_{e\in E} y_e c_e \quad (17)$$

$$\text{subject to} \quad \forall P\in\mathcal{P}_{\mathsf{D}},\ \sum_{e:e\in P} y_e \geq 1. \quad (18)$$

$$\forall e\in E,\ y_e\in\{0,1\}. \quad (19)$$

Proof: Consider any delay-constrained rate R that is feasible. By Proposition 1, for any T value, the time expanded graph $\overline{G}^{[T+\mathsf{D}]}$ can sustain T simultaneous unicast flows from $[s,t]$ to $[d,t+\mathsf{D}]$ for all $t\in[1,T]$ with individual rate R.

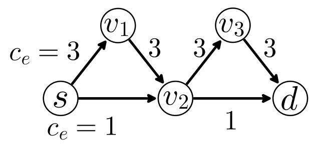
For any given IP solution $\{y_e\}$ satisfying (18)–(19), we construct an edge set in the time expanded graph $\overline{G}^{[T+\mathsf{D}]}$ by

$$\overline{E}_{\text{cut}} \triangleq \{([u,\tau],[v,\tau+1]) : \forall(u,v) \text{ s.t. } y_{(u,v)}=1, \quad \forall\tau\in[1,T+\mathsf{D}-1]\}. \quad (20)$$

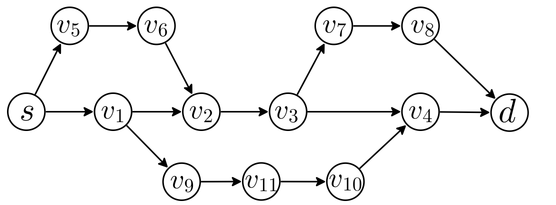
[10] More precisely, a *network instance* should contain a 5-tuple $(G,\{c_e\},s,d,q)$ where G is the network topology, $\{c_e\}$ is the edge capacity, s and d are the source/destination nodes, and q is the alphabet size under consideration. For notational simplicity, we use "$G\in\mathcal{G}_{\text{s-u}}$" as shorthand for the more precise expression "$(G,\{c_e\},s,d,q)\in\mathcal{G}_{\text{s-u}}$."

[11] In the single-unicast setting, one needs to consider a different type of interference. That is, optimal NC needs to remove the corruption caused by *future, not-yet decoded packets* within the same flow. See the detailed discussion of the suboptimal RLNC choice (10) versus the optimal choice (11). Such a new notion of interference is strongly coupled with the time-axis and calls for the development of new analysis tools.

[12] One can prove that for any fixed G, if $\mathsf{D}\geq\mathsf{dia}(G)$, then we always have $\mathsf{gain}_{\text{s-u}}(G,\mathsf{D})=1$. The reason is that $\mathsf{dia}(G)$ is an upper bound of all paths from s to d. Therefore if $\mathsf{D}\geq\mathsf{dia}(G)$ then any path will meet the delay constraint, which is as if $\mathsf{D}=\infty$, and the single-unicast network coding gain for $\mathsf{D}=\infty$ is known to be 1. This observation shows that a combination of large D but small $\mathsf{dia}(G)$ will not yield large $\mathsf{gain}_{\text{s-u}}(G,\mathsf{D})$.



(a) One network example with $\mathsf{D} = 3$.



(b) Another network example with $\mathsf{D} = 6$. All edges have $c_e = 1$.

Fig. 3. Illustration of the upper bounds in Propositions 3 and 4.

By interpreting (18) in the time expanded graph $\overline{G}^{[T+\mathsf{D}]}$, one can verify that the above choice of $\overline{E}_{\text{cut}}$ is an edge cut separating $[s,t]$ from $\{[d,\tau+\mathsf{D}] : \forall\tau \in [1,T]\}$ for all $t \in [1,T]$. By the *generalized network-sharing bound* in [21], we thus have

$$T \cdot R \leq \sum_{\overline{e} \in \overline{E}_{\text{cut}}} c_{\overline{e}} \tag{21}$$

$$= (T + \mathsf{D} - 1) \sum_e y_e c_e \tag{22}$$

where $\overline{e}$ represents an edge in the time expanded graph $\overline{G}^{[T+\mathsf{D}]}$ and $c_{\overline{e}}$ is the corresponding edge capacity; (21) follows from that the sum rate of the T coexisting flows is no larger than the generalized cut set value [21]; and (22) follows from the definition of $\overline{E}_{\text{cut}}$ in (20).

Ineq. (22) implies $R \leq \frac{T+\mathsf{D}-1}{T}\left(\sum_e y_e c_e\right)$ for all T. By letting $T \to \infty$ and by finding an IP solution $\{y_e\}$ that minimizes (17), the proof is complete. ■

Comparing Proposition 3 with (3)–(4), we see that *adding the integer condition (19) to the minimization problem turns R^*_{route}, a lower bound on R^*_{NC}, to an upper bound* UB_{NC} *on R^*_{NC}*. Proposition 3 also implies that for any network instance, if the minimizing y^*_e of (3)–(4) is integral, then the lower and upper bounds match and we have fully characterized the delay-constrained unicast NC capacity: $R^*_{\text{route}} = R^*_{\text{NC}} = \mathrm{UB}_{\text{NC}}$. On the other hand, for any network instance in which $R^*_{\text{NC}} > R^*_{\text{route}}$, e.g., Fig. 2, the y^*_e of (3)–(4) must be fractional, which is consistent with our observation in the end of Section IV-B.

To illustrate Proposition 3, we apply the upper bound to three network instances: Fig. 3(a), Fig. 2, and Fig. 3(b), respectively. We first focus on the IP problem generated by applying Proposition 3 to Fig. 3(a). Simple counting arguments can be used to show that the following assignment of y^*_e is an optimal solution of the corresponding IP problem: $y^*_e = 1$ if $e \in \{(s,v_2),(v_2,d)\}$ and $y^*_e = 0$ for all other e. As a result, we have $R^*_{\text{NC}} \leq 1 + 1 = 2$ for Fig. 3(a). The upper bound in Proposition 3 thus shows that the high-capacity edges $c_{(s,v_1)} = c_{(v_1,v_2)} = c_{(v_2,v_3)} = c_{(v_3,d)} = 3$ do not increase the delay-constrained capacity regardless how the network code is designed. Also note that the edge set $\{(s,v_2),(v_2,d)\}$ does not separate s from d since s can still reach d through $sv_1v_2v_3d$. On the other hand, the edge set $\{(s,v_2),(v_2,d)\}$ "cuts" all paths of length $\leq \mathsf{D} = 3$.

We now apply Proposition 3 to Fig. 2 and generate the corresponding IP problem. Some simple observation can be used to show that for this particular IP problem, the minimizing $\{y^*_e\}$ is not unique. One minimizing solution is $y^*_e = 1$ if $e \in \{(v_4,d),(v_8,d)\}$ and $y^*_e = 0$ for all other e. As a result, we have $R^*_{\text{NC}} \leq 1+1 = 2$ for Fig. 2. Proposition 3 is tight for Fig. 2 since Section IV-B proves that NC can indeed achieve the delay-constrained capacity 2 packets per time slot.

Unfortunately, Proposition 3 can be loose[13] in some scenarios. For example, consider the network in Fig. 3(b). Comparing Figs. 2 and 3(b), the only difference is that the path $v_1v_9v_{10}v_4$ in Fig. 2 is now lengthened to $v_1v_9v_{11}v_{10}v_4$. We apply Proposition 3 to Fig. 3(b) and generate the corresponding IP problem. One minimizing solution of the IP problem is $y^*_e = 1$ if $e \in \{(v_4,d),(v_8,d)\}$ and $y^*_e = 0$ for all other e. As a result, we have $R^*_{\text{NC}} \leq 1 + 1 = 2$ for Fig. 3(b). However, as will be shown later, the network in Fig. 3(b) has $R^*_{\text{NC}} = R^*_{\text{route}} = 1.5$. Namely, regardless how the network code is designed, the best achievable R^*_{NC} is no larger than R^*_{route}. The upper bound in Proposition 3 fails to provide a tight bound and is loose for the network in Fig. 3(b).

In the following, we provide an improved upper bound by directly describing the upper bound in the time-expanded network.

Proposition 4: For any fixed integer $L > 0$, we define a binary mapping $h : E \times [0, L-1] \mapsto \{0,1\}$ where E is the collection of all edges in G. Once the mapping h is fixed, we define the following edge set

$$\overline{E}_h \triangleq \{([u,t],[v,t+1]) : \forall e = (u,v) \in E, \\ \forall t \in [1, L+\mathsf{D}-1] \text{ s.t. } h(e, \bmod(t,L)) = 1\} \tag{23}$$

in the time expanded graph $\overline{G}^{[L+\mathsf{D}]}$.

If for any $t \in [1,L]$ the edge set $\overline{E}_h$ is always an edge cut separating $[s,t]$ from $[d,t+\mathsf{D}]$, then we have a new upper bound

$$R^*_{\text{NC}} \leq \sum_e \left(\frac{\sum_{l=0}^{L-1} h(e,l)}{L}\right) c_e. \tag{24}$$

Remark: An upper bound similar to Proposition 4 is discovered independently in [19, Theorem 3].

The following corollary shows that Proposition 4 is a strict generalization of Proposition 3.

Corollary 1: The upper bound in Proposition 4 with $L = 1$ is equivalent to the upper bound in Proposition 3.

The proofs of Proposition 4 and Corollary 1 are relegated to Appendix G.

We now use Proposition 4 to prove that $R^*_{\text{NC}} = R^*_{\text{route}} = 1.5$ for the network in Fig. 3(b). This shows that Proposition 4 can be strictly tighter than Proposition 3. It also shows that when the path $v_1v_9v_{10}v_4$ in Fig. 2 is getting longer, the side information carried through that path is getting older and thus

[13] An example in [19] shows that the gap between the upper bound in Proposition 3 and the lower bound R^*_{route} can approach infinity.

can no longer be used to remove the interference along the (v_4, d) edge. The R^*_{NC} capacity will thus reduce from 2 to 1.5 and the NC gain in Fig. 2 disappears.

To apply Proposition 4 to Fig. 3(b), we choose $L = 2$ and set

$$h(e,0) = \begin{cases} 1 & \text{if } e \in \{(s,v_1),(v_4,d)\} \\ 0 & \text{for all other } e \end{cases} \tag{25}$$

$$h(e,1) = \begin{cases} 1 & \text{if } e = (v_2,v_3) \\ 0 & \text{for all other } e \end{cases} \tag{26}$$

In Appendix H, we have proven that for the network in Fig. 3(b) with the binary mapping $h(e,t)$ given in (25)–(26), the edge set $\overline{E}_h$ is always an edge cut in the time expanded graph $\overline{G}^{[L+\mathsf{D}]}$ that separates $[s,t]$ from $[d,t+\mathsf{D}]$ for all $t \in [1, L]$. As a result, by Proposition 4 we have

$$R^*_{\text{NC}} \leq \sum_{e \in \{(s,v_1),(v_4,d),(v_2,v_3)\}} \frac{1}{2} \cdot c_e = 1.5.$$

V. Various Other Results On Delay-Constrained Capacity

This section contains various other results on delay-constrained capacity, including the unboundedness of the penalty of using RLNC and an example showing that Edmonds' tree packing theorem no longer holds for the delay-constrained setting. In the end of this section, we also discuss how our results can be applied to cyclic networks verbatim.

A. The penalty of using RLNC can be unbounded

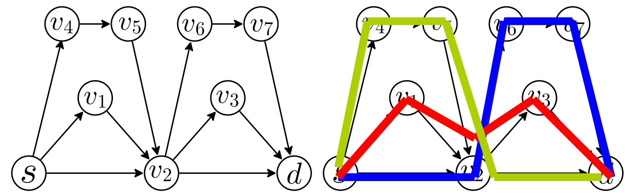
In Section II-C, we have shown that even when considering only a single unicast flow, the best RLNC scheme may have strictly worse performance than R^*_{route} when the traffic is delay-constrained. Similar to our investigation in Section IV-C where we studied the largest possible NC gain $\frac{R^*_{\text{NC}}}{R^*_{\text{route}}}$, in this section we study the largest possible penalty of using RLNC when compared to the routing solution. That is, we are interested in quantifying the following *RLNC penalty*:

$$\text{Largest RLNC Penalty} \triangleq \sup_{G \in \mathcal{G}_{\text{s-u}}, \mathsf{D} \in [1,\infty)} \frac{R^*_{\text{route}}}{R^*_{\text{RLNC}}}. \tag{27}$$

By the example in Section 5, we know that the largest RLNC penalty is no less than 2.

To quantify the RLNC penalty in (27), we first formally define the class of RLNC being considered. Specifically, we focus exclusively on the unit-edge-capacity network, i.e., $c_e = 1$ for all $e \in E$. All the vectors in our discussion, unless specified otherwise, are column vectors.

For any given integer rate R, the RLNC scheme is defined as follows. There are R message symbols in each time slot, and they are defined as a vector $\vec{X}(t) = (X_1(t), X_2(t), \cdots, X_R(t))^{\text{T}} \in (\mathsf{GF}(q))^R$ where $\mathsf{GF}(q)$ is the finite field of order q and we assume q is a power of a prime. The source s is associated with $|\text{Out}(s)|$ precoding vectors $\vec{\beta}_{s,e}, \forall e \in \text{Out}(s)$ and each is of dimension R. Each intermediate node $v \in V \backslash \{s, d\}$ is associated with $|\text{In}(v)| \cdot |\text{Out}(v)|$ local coding coefficients $\beta_{e_1,e_2} \in \mathsf{GF}(q), \forall e_1 \in \text{In}(v), e_2 \in \text{Out}(v)$. At every time slot t, source s sends the coded symbols



(a) The topology (b) The optimal routes

Fig. 4. A straightforward generalization of Fig. 1(a) with $\mathsf{D} = 4$. This network has $R^*_{\text{NC}} = R^*_{\text{route}} = 3$ and $R^*_{\text{RLNC}} = 2$.

$$M_e^{(t)} = \vec{\beta}^{\text{T}}_{s,e} \vec{X}(t) \tag{28}$$

along all its outgoing edges $e \in \text{Out}(s)$; and any intermediate node $v \in V \backslash \{s,d\}$ sends the coded symbols

$$M_e^{(t)} = \sum_{e_1 \in \text{In}(v)} \beta_{e_1,e} M_{e_1}^{(t-1)} \tag{29}$$

along all its outgoing edges $e \in \text{Out}(v)$.

During the design phase, the RLNC scheme chooses the precoding vectors $\vec{\beta}_{s,e}, \forall e \in \text{Out}(s)$ and the local coding coefficients $\beta_{e_1,e_2}, \forall e_1 \in \text{In}(v), e_2 \in \text{Out}(v), v \in V \backslash \{s,d\}$ independently and uniformly randomly from $(\mathsf{GF}(q))^R$ and from $\mathsf{GF}(q)$, respectively. We assume that after the design phase, both $\vec{\beta}_{s,e}$ and β_{e_1,e_2} are made known to all network nodes, including s and d. With that knowledge, the network can perform RLNC based on (28) and (29). We assume that the choices of $\vec{\beta}_{s,e}$ and β_{e_1,e_2} are fixed after the initial design phase and they do not change over time t.

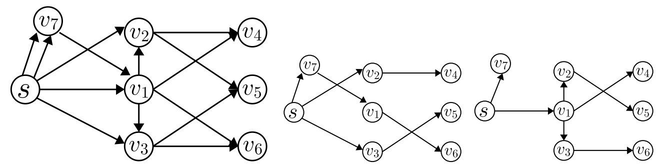
Definition 4: We say RLNC can support an integer delay-constrained rate R if the probability that with a random design phase, destination d can decode $\vec{X}(t)$ by the end of time slot $t + \mathsf{D} - 1$, $\forall t \in [1, \infty)$, approaches 1 when the underlying finite field size q is sufficiently large. The largest supportable rate of RLNC is denoted by R^*_{RLNC}.

We now have the following result.

Proposition 5: The penalty of using RLNC can be arbitrarily large. That is,

$$\sup_{G \in \mathcal{G}_{\text{s-u}}, \mathsf{D} \in [1,\infty)} \frac{R^*_{\text{route}}}{R^*_{\text{RLNC}}} = \infty.$$

The detailed proof of Proposition 5 is relegated to Appendix I, which will be based on explicit network construction. The very first idea of constructing a network with large penalty (27) is to generalize Fig. 1(a) in a way similar to Fig. 4(a). Specifically, there are 3 paths in Fig. 4(b) with length exactly equal to the delay requirement $\mathsf{D} = 4$. Therefore, the optimal $R^*_{\text{route}} = 3$. On the other hand, since all 3 paths share a common node v_2, in a similar way as discussed in Section II-C, the packets along the $sv_4v_5v_2d$ and the $sv_1v_2v_3d$ paths will be contaminated by some sort of *future packets* due to the random packet mixing at node v_2. Note that the packets along the $sv_2v_6v_7d$ are contaminated by the *past packets*, which can be automatically canceled by the packets decoded in the previous time slots.



(a) A network instance with $\mathsf{D} = 2$ and all $c_e = 1$. (b) One possible tree packing solution.

Fig. 5. An illustration that Edmonds' tree packing theorem does not hold for delay-constrained traffic.

As a result, one might expect that Fig. 4(a) has $R^*_{\text{RLNC}} = 1$ since only the packets entering d through the (v_7, d) edge will be free from the corruption of the future packets. In Appendix J, we prove that Fig. 4(a) actually has $R^*_{\text{RLNC}} = 2$. As a result the RLNC penalty of Fig. 4(a) is $\frac{3}{2}$, which is even smaller than the simple example in Fig. 1(a) where the RLNC penalty is 2. The reason $R^*_{\text{RLNC}} = 2$ in Fig. 4(a) is because destination d can carefully remove some of the "future interference" by intelligently combining the packets received from the three edges (v_2, d), (v_3, d), and (v_7, d). The main challenge of proving Proposition 5 is to construct a network and prove that even with an optimal decoder at d that can remove the "future interferences" by intelligently combining different received packets, the resulting R^*_{RLNC} is still arbitrarily away from the routing rate R^*_{route}. A simple generalization of Figs. 1(a) like the one described in 4(a) will not work.

In Appendix I, we show that for any $m \geq 2$, one can design a network for which $R^*_{\text{route}} = m$ and $R^*_{\text{RLNC}} = 1$ even with the use of an optimal decoder at destination d.

B. The 1-to-all scenario

In this section, we consider the 1-to-all broadcast scenario. Namely, source s would like to send the same information to all destinations $V \backslash s$ at rate R. The tree-packing theorem in [6], [43] proves that the largest possible information-theoretic capacity R^* can be achieved by pure routing without the need of NC. Namely, $R^*_{\text{NC}} = R^*_{\text{route}}$ for the 1-to-all scenario.

In this section, we will prove by an example that such an elegant result does not hold when the information is delay-constrained.

Proposition 6: Denote all the 1-to-all network scenarios by $\mathcal{G}_{\text{1-all}}$. We have

$$\sup_{G \in \mathcal{G}_{\text{1-all}}, \forall \mathsf{D}} \frac{R^*_{\text{NC}}}{R^*_{\text{route}}} = \infty.$$

Proof: We first describe an example network with $\frac{R^*_{\text{NC}}}{R^*_{\text{route}}} = \frac{2}{1.5}$. Since our construction is based on simple modification of the $\binom{3}{2}$ *combination network* construction in [34], by the same analysis on the most general $\binom{n}{m}$ combination networks in [34] one can complete the proof.

Consider the network in Fig. 5(a), for which the edge capacity $c_e = 1, \forall e \in E$ and there are two parallel edges connecting s and v_7. One can check that the min-cut values satisfy min-cut$(s, v_i) = 2$ for all $v_i \in V \backslash s$. Edmonds' tree packing results show that such a network can support 2 edge-disjoint spanning trees, see Fig. 5(b). Therefore when there is no delay constraint ($\mathsf{D} = \infty$) we have $R^*_{\text{NC}} = R^*_{\text{route}} = 2$.

We now impose a delay requirement $\mathsf{D} = 2$. It is clear that by sending $X(t)$, $Y(t)$, and $X(t) + Y(t)$ at time t along (s, v_1), (s, v_2), and (s, v_3), respectively, all six nodes v_1 to v_6 can decode both $X(t)$ and $Y(t)$ by the end of time $t+1$, provided v_1 to v_3 simply forwarding what they have received in the previous time slot to all its downstream neighbors. Source s can send both $X(t)$ and $Y(t)$ directly to v_7. Since all 7 nodes v_1 to v_7 can be satisfied simultaneously, we have $R^*_{\text{NC}} = 2$.

To prove that $R^*_{\text{route}} = 1.5$, we consider how to use routing to satisfy nodes v_4 to v_6 and temporarily ignore the needs of v_1 to v_3 and v_7. We observe that when satisfying nodes v_4 to v_6, the routing paths must not use any of the five edges $\{(s, v_7)_{\times 2}, (v_7, v_1), (v_1, v_2), (v_1, v_3)\}$. Otherwise, the paths will have length ≥ 3, which exceeds the delay requirement $\mathsf{D} = 2$. However, if we remove edges $\{(s, v_7)_{\times 2}, (v_7, v_1), (v_1, v_2), (v_1, v_3)\}$, then the resulting network is a $\binom{3}{2}$ combination network defined in [34]. The results in [34] show that the largest possible routing rate that satisfies v_4 to v_6 simultaneously is 1.5. By revising the scheme in [34] slightly, we can also satisfy nodes v_1 to v_3 and v_7 with rate 1.5. As a result, Fig. 5(a) has $R^*_{\text{route}} = 1.5$.

To prove that $\frac{R^*_{\text{NC}}}{R^*_{\text{route}}}$ can be unbounded, we start from any arbitrarily given $\binom{n}{m}$ combination networks described in [34] and modify it by introducing new edges among the n layer-1 nodes $(v_1, v_i)_{\times m-1}$ for all $i \in [2, n]$. The notation $(v_1, v_i)_{\times m-1}$ denotes $(m-1)$ parallel edges connecting v_1 and v_i. See the $\{(v_1, v_2), (v_1, v_3)\}$ edges in Fig. 5(a) for illustration. We then introduce an auxiliary node v_{2n+1} that add two edges $(s, v_{2n+1})_{\times m}$ and $(v_{2n+1}, v_1)_{\times m-1}$. See the $\{(s, v_7)_{\times 2}, (v_7, v_1)\}$ edges in Fig. 5(a) for illustration. Using the same argument, a routing solution cannot use those new edges due to the delay constraint. Therefore, the 1-to-all network coding gain for the modified network is the same as the network coding gain for the original $\binom{n}{m}$ combination

There is an error in the last paragraph in the proof of Proposition 6. Please see the official published paper for the corrected version. https://doi.org/10.1109/TIT.2016.2619717

C. Remarks on Cyclic Networks

All our results can be applied to cyclic networks verbatim. More specifically, Proposition 1 proves the equivalence between a delay-constrained unicast problem and a multiple-unicast problem in the corresponding time-expanded graph. Since the causality in time automatically converts any network (regardless being cyclic or not) into an acyclic time-expanded graph, Proposition 1 holds naturally for cyclic networks as well. Propositions 3 and 4 are proven based on the equivalent time-expanded graph (Proposition 1). As a result, they hold for cyclic networks as well. Propositions 2, 5, and 6 are proven by explicit construction of special acyclic network instances. Since acyclic networks are a special case of cyclic ones, Propositions 2, 5, and 6 hold for cyclic networks as well.

VI. Conclusion

This work studies the following problem: Given a hard delay constraint, how much perishable information one can send from s to d. We have provided a new formulation that converts the delay-constrained unicast capacity problem into a delay-unconstrained multiple unicsat problem. We have then proven that NC can strictly outperform optimal routing even for the single-unicast setting and the gain can be arbitrarily close to 2. Based on a time-expanded graph approach, two new delay-constrained capacity upper bounds are provided, which show that the difference between the NC capacity R^*_{NC} and the routing capacity R^*_{route} can be associated to the integrality gap of the LP-based minimum cut problem. We have also proven that the penalty of using RLNC versus pure routing can be unbounded and the classic Edmonds' tree packing results no longer hold for the delay-constrained traffic. Overall, our results suggest that delay-constrained communication is fundamentally different from the well-understood delay-unconstrained one and call for investigation participation.

There are several interesting directions for further investigation. In the current setting, information bits incur a fixed amount of delay when traversing a link. It is interesting to extend the study to consider a general setting where the delay of traversing a link is also a function of the traffic volume, which better models practical scenarios involving fast-timescale system/flow dynamics.

Appendix A
A Low-Complexity Computation of The Delay-Constrained Routing Capacity

The existing polynomial-complexity computation of the delay-constrained routing capacity in [45] is restated as follows.

Proposition 7 (Section IV.A [45]): We can compute R^*_{route} in (1)–(2) by the following flow-based LP problem with $|E| \cdot \mathsf{D}$ non-negative variables $x_e^{(h)}$ for all $e \in E$ and $h \in [1, \mathsf{D}]$:

$$\max_{x_e^{(h)} \geq 0} \quad \sum_{e \in \text{In}(d)} \sum_{h=1}^{\mathsf{D}} x_e^{(h)} \tag{30}$$

$$\text{subject to} \quad \forall v \in V \backslash \{s, d\}, \forall h \in [1, \mathsf{D}],$$

$$\sum_{e \in \text{In}(v)} x_e^{(h-1)} = \sum_{\tilde{e} \in \text{Out}(v)} x_{\tilde{e}}^{(h)} \tag{31}$$

$$\forall e \in E, \quad \sum_{h=1}^{\mathsf{D}} x_e^{(h)} \leq c_e. \tag{32}$$

Here each variable $x_e^{(h)}$ represents the sum of all rates assigned to paths $\{P \in \mathcal{P}_{\mathsf{D}} : \text{the } h\text{-th hop of } P \text{ is } e\}$. The objective in (30) is the aggregate rate of flows that arrive at d within D hops. The constraints in (31) say that the aggregate incoming flows to node v with hop count $h-1$ must be equal to the aggregate outgoing flows from node v with hop count h; these are essentially the flow balance equations with flow travelled-distance (in hops) taken into account. The constraints in (32) are link capacity constraints. Note that in (31) we use the convention that $x_e^{(0)} = 0$ for all $e \in E$.

Since the proof of Proposition 7 was omitted in [45], we sketch the proof in the following for completeness.

Proof: We first prove that any solution in the LP problem (1)–(2) leads to a valid solution for the LP problem (30)–(32). This is done by setting

$$x_e^{(h)} = \sum_{P \in \mathcal{P}_{\mathsf{D}}:\ \text{the } h\text{-th hop of } P \text{ is } e} x_P.$$

With the above construction of $x_e^{(h)}$, one can see that (31) holds naturally and (1) equals to (30). Also, (2) on x_P implies (32). The forward direction is thus proven.

We now prove that any solution in (30)–(32) leads to a valid solution in (1)–(2). We prove this by an iterative construction. Initially, we define $\underline{\mathcal{P}} = \mathcal{P}_{\mathsf{D}}$ and $\underline{x}_e^{(h)} = x_e^{(h)}$ for all e and h. For any $P \in \underline{\mathcal{P}}$, we choose $x_P = \min_{h \in [1,|P|]} \underline{x}_{e_{P,h}}^{(h)}$ where $e_{P,h}$ is the h-th hop of P. After choosing x_P, we decrease the value of $\underline{x}_{e_{P,h}}^{(h)}$ by x_P for all $h \in [1, |P|]$ and remove P from $\underline{\mathcal{P}}$. After decreasing the $\underline{x}_{e_{P,h}}^{(h)}$ values and reducing $\underline{\mathcal{P}}$, we repeat the construction for another $\tilde{P} \in \underline{\mathcal{P}}$ until $\underline{\mathcal{P}} = \emptyset$.

We now state and prove three claims regarding the above construction.

Claim 1: Throughout the process, all the $\underline{x}_e^{(h)}$ are non-negative and they satisfy (31). The non-negativity holds since $x_P = \min_{h \in [1,|P|]} \underline{x}_{e_{P,h}}^{(h)}$. Equality (31) holds since we subtract x_P from $\underline{x}_{e_{P,h}}^{(h)}$ for all $h \in [1, |P|]$. Note that $\underline{x}_e^{(h)} \geq 0$ also implies that the resulting x_P is non-negative.

Claim 2: When the iterative construction finishes, the resulting $\{x_P : \forall P \in \mathcal{P}_{\mathsf{D}}\}$ satisfies (2). To prove this claim, we notice that since x_P is deducted from $\underline{x}_{e_{P,h}}^{(h)}$ during our construction, we always have

$$\forall e \in E, \quad \sum_{P: P \ni e, P \in \mathcal{P}_{\mathsf{D}} \backslash \underline{\mathcal{P}}} x_P + \sum_{h=1}^{\mathsf{D}} \underline{x}_e^{(h)} = \sum_{h=1}^{\mathsf{D}} x_e^{(h)} \tag{33}$$

in the end of each iteration, where $\{x_P\}$ are the latest values assigned to each path P, $\{x_e^{(h)}\}$ are the original LP variable values we begin with, and $\{\underline{x}_e^{(h)}\}$ are the latest *residual values* in our construction. Then by (32) and by the non-negativity of $\underline{x}_e^{(h)}$, the resulting x_P in the end (*i.e.*, when $\underline{\mathcal{P}} = \emptyset$) must satisfy (2).

Claim 3: The expression (1) computed from the final x_P equals (30) computed from $x_e^{(h)}$. To prove this claim, we notice that by the construction of $\mathcal{P}_{\mathsf{D}}$, the last edge of any $P \in \mathcal{P}_{\mathsf{D}}$ must belong to $\text{In}(d)$. Since (33) holds for any edge $e \in \text{In}(d)$, we only need to prove that $\underline{x}_e^{(h)} = 0$ for all $e \in \text{In}(d)$ and $h \in [1, \mathsf{D}]$ in the end of our construction.

We prove this by contradiction. Suppose not. Then we have $\underline{x}_{\hat{e}}^{(\hat{h})} > 0$ for some $\hat{e} \in \text{In}(d)$ and $\hat{h}$. Since $\{\underline{x}_e^{(h)}\}$ satisfies (31) for all $v \in V \backslash \{s, d\}$ and h, we can find $\hat{h}$ edges, denoted by $\hat{e}_1$ to $\hat{e}_{\hat{h}}$, satisfying simultaneously (i) $\hat{e}_{\hat{h}} = \hat{e}$; (ii) $\hat{e}_1 \hat{e}_2 \cdots \hat{e}_{\hat{h}}$ form a path of length $\hat{h}$, which is denoted by $\hat{P}$; and (iii) $\underline{x}_{\hat{e}_i}^{(i)} > 0$ for $i \in [1, \hat{h}]$. Since $\hat{e}_{\hat{h}} = \hat{e} \in \text{In}(d)$, we have $\mathsf{head}(\hat{e}_{\hat{h}}) = d$. We now prove $\mathsf{tail}(\hat{e}_1) = s$ by contradiction. Suppose not. Then we focus on (31) with $h = 1$ and $v = \mathsf{tail}(\hat{e}_1)$. One can see that the left-hand side of (31) is zero since $\underline{x}_e^{(0)} = 0$ in our convention but the right-hand side is no less than $\underline{x}_{\hat{e}_1}^{(1)} > 0$.

This contradiction implies that $\mathsf{tail}(\hat{e}_1) = s$. As a result, $\hat{P}$ connects s and d using $\hat{h} \le \mathsf{D}$ hops. Therefore $\hat{P} \in \mathcal{P}_{\mathsf{D}}$.

On the other hand, all the paths in $\mathcal{P}_{\mathsf{D}}$ must have been considered in the iterative construction. Consequently, for any path $P \in \mathcal{P}_{\mathsf{D}}$, $\underline{x}^{(h)}_{e_{P,h}} = 0$ for at least one $h \in [1, |P|]$ since we subtract $x_P = \min_{h\in[1,|P|]} \underline{x}^{(h)}_{e_{P,h}}$ from all $\underline{x}^{(h)}_{e_{P,h}}$. Property (iii) of $\hat{P} \in \mathcal{P}_{\mathsf{D}}$ thus contradicts the fact that we have exhaustively considered all $P \in \mathcal{P}_{\mathsf{D}}$. Claim 3 is thus proven.

Jointly, Claims 1 to 3 complete the proof. ■

Since (30)–(32) is a polynomial-time computable version of the maximum flow problem (1)–(2), in this work we have derived the following polynomial-time computable version of the minimum cut problem (3)–(4).

Corollary 2: We can compute R^*_{route} by the following LP problem with $|E|$ non-negative variables $y_e \ge 0$, $\forall e \in E$, and $(|V| - 2) \cdot \mathsf{D}$ real-valued variables $y^{(h)}_v$, $\forall v \in V\backslash\{s, d\}, h \in [1, \mathsf{D}]$ such that

$$\min_{y_e\ge 0, y^{(h)}_v} \sum_{e\in E} y_e c_e \tag{34}$$

$$\text{s.t.} \quad y_e + y^{(h+1)}_{\mathsf{head}(e)} - y^{(h)}_{\mathsf{tail}(e)} \ge 0, \ \forall e \in E, \forall h \in [1, \mathsf{D}], \tag{35}$$

where in (35) we use the convention $y^{(\mathsf{D}+1)}_v = 0$ for all $v \in V$ and $y^{(h)}_s = y^{(h)}_d = 0$ for all $h \in [1, \mathsf{D}]$.

Proof: One can easily verify that Corollary 2 is the dual of Proposition 7. ■

Appendix B
The High-Level Description of The Proof of Proposition 2

The proof of Proposition 2 contains three major ingredients. Firstly, we will generalize the example in Section IV-B and describe a family of network instances $\{(G_m, \mathsf{D}_m) : \forall m \in \mathbb{N}\}$ indexed by m such that the corresponding NC gain, denoted by gain_m, is strictly larger than 1 for all $m \in \mathbb{N}$. Secondly, we will provide an *iterative genie-aided construction* such that with the help of a genie we can use a network instance of (G, D) with NC gain to construct another network instance (G', D') with NC gain' such that $\mathsf{gain}' > \mathsf{gain}$. Furthermore, when the iteration continues indefinitely the final gain' approaches 2 asymptotically. The family of network instances $\{(G_m, \mathsf{D}_m) : \forall m \in \mathbb{N}\}$, introduced as the first component, is essential in this iterative construction. Finally, we will describe how the iterative construction can be implemented without the help of a genie.

The family of network instances is described in Appendix C. The iterative genie-aided construction is described in Appendix D. Appendix E describes how to perform iterative construction without the help of a genie.

Appendix C
A Family Of Network Instances With $\mathsf{gain} > 1$

In this section, we generalize the result in Section IV-B and describe a set of network instances $\{(G_m, \mathsf{D}_m) : \forall m \in \mathbb{N}\}$ such that the corresponding NC gain, denoted by gain_m, is strictly larger than 1.

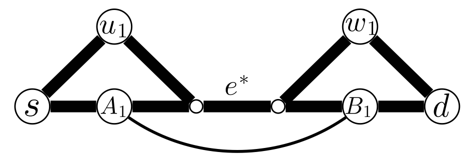


Fig. 6. A high-level description of the network instance that was previously described in Fig. 2.

A. A high-level network description

To that end, we first represent the network in Fig. 2 by an equivalent but more high-level description in Fig. 6. The high-level description contains two components. The primary component is the two triangles (one on the left and one on the right) plus the straight line from s to d. We use thick lines in Fig. 6 to represent the primary component. The two triangles symbolize the two possible *detours* from the straight line. One is to use node u_1 and the other is to use node w_1. Comparing Figs. 2 and 6, we can view node u_1 as a relabeling of node v_5 and node w_1 as a relabeling of node v_7. Note that each thick line in Fig. 6 does not correspond to an edge in Fig. 2. Instead, a thick line in Fig. 6 corresponds to a path in Fig. 2. For example, since we view u_1 as a relabeling of node v_5, the thick line connecting u_1 and the tail of e^* corresponds to the 2-edge path $v_5v_6v_2$ in Fig. 2 and the thick line connecting s and u_1 corresponds to the 1-edge path sv_5 in Fig. 2. Therefore, the left detour from s along u_1 to the tail of e^* has length 3 even though it is represented by two thick line segments.

We choose the delay requirement $\mathsf{D} = 6$ in a way that if we detour for exactly once, e.g., the longer path using u_1 but not using w_1, then the total length is exactly D. Equivalently, it means that the choice $\mathsf{D} = 6$ implies that if we detour for two times, using both u_1 and w_1, then the resulting path will have length $> \mathsf{D}$. For example, the path $sv_5v_6v_2v_3v_7v_8d$ in Fig. 2 corresponds to using both the detour through u_1 and the detour through w_1. Such path has 7 hops, which is $> \mathsf{D} = 6$.

The secondary component of the high-level description in Fig. 6 is a *parallel pipe* connecting (A_1, B_1) that does not use any edges in the primary component. We use a thin curve to represent this parallel pipe. We require that the delay incurred by the (A_1, B_1) parallel pipe is identical to the delay when traversing from A_1 to B_1 using the edges in the primary component. Comparing Figs. 2 and 6, we can view node A_1 as a relabeling of node v_1 and node B_1 as a relabeling of node v_4 in Fig. 2. The parallel pipe in Fig. 2 thus corresponds to the path $v_1v_9v_{10}v_4$ in Fig. 2. The delay incurred by the parallel pipe is 3, which is equal to the delay when traversing through the path $v_1v_2v_3v_4$ in the primary component.

All edges in our high-level description in Fig. 6 are of capacity $c_e = 1$.

We can now explain the NC solution using the high-level description of Fig. 6. Consider sending X_t through the primary component using the detour of w_1 only (i.e., path $sv_1v_2v_3v_7v_8d$ in Fig. 2). For easier reference, we term the above operation "sending X_t through the '*primary*' path su_0w_1d" to emphasize that we only use the edges in the primary component and the notation "u_0" indicates that we do not use the detour corresponding to u_1. Similarly, we can send Y_t through the primary path su_1w_0d, which emphasizes that we use the detour corresponding to u_1 and do not use the detour corresponding to w_1. Recall that the D value equals to the length of path su_0w_1d and equals to the length of path su_1w_0d. Therefore, the intuition is that by sending X_t and Y_t along paths su_0w_1d and su_1w_0d, respectively, our goal is for d to receive both X_t and Y_t before the deadline $t+\mathsf{D}$ through the two primary paths, respectively. For easier reference, we say that X_t (resp. Y_t) is the *desired packet* along path su_0w_1d (resp. su_1w_0d).

However, the two primary paths su_0w_1d and su_1w_0d share a bottleneck edge e^* in Fig. 6, or equivalently edge v_2v_3 in Fig. 2. If we perform NC (i.e., simple packet addition) at e^* without using the (A_1, B_1) parallel pipe, then by the deadline $t+\mathsf{D}$, d will receive packet $[X_t + Y_{t-\delta}]$ through path su_0w_1d for some $\delta > 0$ that represents the difference of the distances between the detour through u_1 and the direct path in the left triangle. Namely, the desired X_t packet will be corrupted by some old packet $Y_{t-\delta}$ since Y_t needs to traverse a longer path (detour through u_1) before entering the bottleneck edge e^* while X_t traverses to e^* through a shorter direct path. Similarly, by the deadline $t+\mathsf{D}$, d will receive packet $[Y_t + X_{t+\delta}]$ through path su_1w_0d. Namely, the desired Y_t packet will be corrupted by some future packet $X_{t+\delta}$ since X_t traverses to e^* through a shorter direct path.

To prevent the desired Y_t (along the primary path su_1w_0d) from the corruption of the future packet $X_{t+\delta}$, we use the (A_1, B_1) parallel pipe to remove the aforementioned corruption. Recall that the (A_1, B_1) parallel pipe incurs the same amount of delay as the primary path from A_1 to B_1. As a result, at node B_1 we can subtract the corruption $X_{t+\delta}$ (known from the information received through the parallel pipe) from the linear sum $[Y_t + X_{t+\delta}]$, which is received through the downstream path from e^* to B_1 in the primary component. *After subtraction, d will receive pure, uncorrupted Y_t through su_1w_0d before the deadline $t+\mathsf{D}$.*

Note that there is no need to introduce additional parallel pipe for the task of subtracting the corruption $Y_{t-\delta}$ for the desired X_t along the su_0w_1d path. The reason is that d has already decoded the old packet $Y_{t-\delta}$ in the previous time slots. Therefore, d can simply use its own knowledge of $Y_{t-\delta}$ to remove the corruption $Y_{t-\delta}$. In summary, by the deadline $t+\mathsf{D}$, d can decode X_t from the su_0w_1d path (with the help of previously obtained knowledge of $Y_{t-\delta}$) and can receive pure uncorrupted Y_t from the su_1w_0d path (with the help of the (A_1, B_1) parallel pipe).

In contrast with the edge-by-edge analysis in Section IV-B, this high-level description emphasizes the network topology and the corresponding network information flow, which will be useful when describing more complicated examples in the subsequent discussion.

B. A family of network instances

From the high-level description, one can see that the key ideas of Fig. 6 are (i) Create 2 primary paths of length equal to D (paths su_0w_1d and su_1w_0d); (ii) Make sure that the two paths use the bottleneck edge e^* in the center; (iii) Make each path correspond to "a route that detours in only one location"; (iv) Choose the D value such that detouring in two locations will violate the delay constraint; and (v) Finally use the parallel pipe to provide side information so that we can remove the corruption of "future interference $X_{t+\delta}$" from the path su_1w_0d so that d can receive uncorrupted Y_t by the deadline $t+\mathsf{D}$. In a broad sense, the use of the (A_1, B_1) parallel pipe alleviates the bottleneck e^*, which is used by both primary paths.

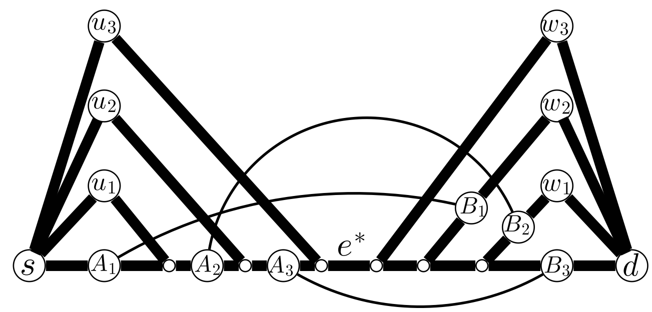


Fig. 7. An illustration of the class of network instances with $\mathsf{gain} > 1$. The illustration is based on $m = 4$.

The following construction demonstrates how we can create $m \geq 2$ primary paths of length equal to D and make sure all the m paths are using the same bottleneck edge e^* so that we can create more "congestion" in e^*, which, heuristically, should lead to a higher NC gain. Again our construction consists of the primary and the secondary components.

1) The primary component: For any fixed integer $m \geq 2$, the primary component of our construction has $2(m-1)$ triangles; $(m-1)$ of them are nested on the left-hand side of the network; $(m-1)$ of them are nested on the right-hand side of the network; and a straight line from s to d that connects the two groups of triangles. See Fig. 7 for the illustration of $m = 4$, in which the paths of the primary component are represented by thick lines.

We use the notation su_iw_jd, where $i, j \in [0, m-1]$, to denote the path that detours through u_i and w_j in the left and the right triangles, respectively. The notation u_0 (resp. w_0) means no detour in the left half (resp. the right half) of the network. We call the path $su_iw_{m-1-i}d$ the *primary path*. The sizes of the triangles are carefully chosen so that the primary paths $su_iw_{m-1-i}d, \forall i \in [0, m-1]$ have the same common length. Additionally, the sizes of the triangles are chosen in a way that *the larger the triangle, the longer the corresponding detour will be*. Take Fig. 7 for example. All the primary paths having the same length means

$$\begin{aligned}\mathsf{length}(su_0w_3d) &= \mathsf{length}(su_1w_2d)\\ &= \mathsf{length}(su_2w_1d) = \mathsf{length}(su_3w_0d).\end{aligned} \tag{36}$$

The assumption "the larger the triangle, the longer the detour" means

$$\begin{aligned}&\mathsf{length}(su_0w_0d) < \mathsf{length}(su_1w_0d)\\ &\qquad < \mathsf{length}(su_2w_0d) < \mathsf{length}(su_3w_0d)\\ \text{and}\quad &\mathsf{length}(su_0w_0d) < \mathsf{length}(su_0w_1d)\\ &\qquad < \mathsf{length}(su_0w_2d) < \mathsf{length}(su_0w_3d).\end{aligned} \tag{37}$$

Eqs. (36) and (37) together imply many other inequalities. For example, we also have

$$\begin{aligned}\mathsf{length}(su_1w_3d) &> \mathsf{length}(su_0w_3d)\\ &= \mathsf{length}(su_2w_1d) > \mathsf{length}(su_2w_0d).\end{aligned}$$

After fixing the network topology, we choose the delay requirement D_m value such that

$$\mathsf{D}_m = \mathsf{length}(su_iw_{m-1-i}d),\ \forall i = 0, \cdots, m-1, \quad (38)$$

is the length of all m primary paths.

2) The secondary component: The secondary component contains $(m-1)$ *parallel node-disjoint pipes* connecting (A_i, B_i) for $i = 1$ to $m-1$, which do not use any nodes/edges in the primary component except for the end nodes A_i and B_i. We use thin curves to represent these parallel node-disjoint pipes. We now describe the starting and ending nodes of these $m-1$ pipes.

Starting node A_i:

- If $i = 1$, then we choose A_1 to be a node satisfying simultaneously (i) A_1 is an interior node of the direct path su_0w_0d; (ii) A_1 is strictly upstream of the node where the direct path su_0w_0d merges with the detour path su_1w_0d, i.e., A_1 is on the left of the merge node. See Fig. 7 for illustration.
- If $i \in [2, m-1]$, then we choose A_i to be a node satisfying simultaneously (i) A_i is strictly downstream of the node where the direct path su_0w_0d merges with the detour path $su_{i-1}w_0d$, i.e., A_i is on the right of the merge node; and (ii) A_i is strictly upstream of the node where the direct path su_0w_0d merges with the detour path su_iw_0d. See Fig. 7 for illustration.

Ending node B_i:

- If $i \in [1, m-2]$, then we choose B_i to be a node satisfying simultaneously (i) B_i is an interior node of the detour path su_0w_{m-1-i}; and (ii) B_i is strictly downstream of the node where the direct path su_0w_0d diverges from the detour path $su_0w_{m-1-i}d$. See Fig. 7 for illustration.
- If $i = m-1$, then we choose B_{m-1} to be a node satisfying simultaneously (i) B_{m-1} is an interior node of the direct path su_0w_0d; (ii) B_{m-1} is strictly downstream of the node where the direct path su_0w_0d diverges from the detour path su_0w_1d. See Fig. 7 for illustration.

One can clearly see that the choices of A_i and B_i are not unique. Our construction holds for any arbitrary choices satisfying the above description.

After fixing the end nodes A_i and B_i, we impose that the delay incurred by the (A_i, B_i) parallel pipe is identical to the delay when traversing from A_i to B_i using only the edges in the primary component. We use (G_m, D_m) to describe the resulting network instance in the above construction. We assume that all edges in our construction (Fig. 7) are of capacity $c_e = 1$. Our construction is now complete.

Intuition: One way to interpret our construction is to view it (see Fig. 7) as an overlay of m paths of length D_m in an offset manner so that they form two groups of triangles (detours), one on the left and one on the right. The overlay is made so that all m paths share a common bottleneck edge e^*. Then introduce parallel pipes that can potentially be used to remove the undesired corruption in the bottleneck edge e^*. Also see our discussion in Appendix C-A.

Subsequently, we will quantify the NC gain for the network instance (G_m, D_m) for any given fixed integer $m \geq 2$.

C. Quantifying the NC gain of the proposed network family

Lemma 1: For any given $m \geq 2$, the delay-constrained NC capacity of (G_m, D_m) is $R^*_{\text{NC}} = m$ packets per time slot.

Proof: We prove this lemma by explicit network code construction. For all $i = 0$ to $m-1$, we send $X_t^{[i]}$ through the *primary path* $su_iw_{m-1-i}d$. With (38), we are hoping that d can receive the desired $X_t^{[m]}$ through path $su_iw_{m-1-i}d$ before the deadline $t + \mathsf{D}_m$. See Fig. 7 for illustration.

Define the "merging nodes" as the nodes for which the su_jw_0d path merges with the $su_{j+1}w_0d$ paths for all $j = 0$ to $m-2$. If we perform NC (i.e., simple packet addition) on all the merging nodes while neglecting all parallel pipes in the secondary component, then by the deadline $t + \mathsf{D}_m$ the packet received by d through path $su_iw_{m-1-i}d$ will have the following form:

$$\left(\sum_{j=0}^{i-1} X^{[j]}_{t+|\delta_{j,i}|}\right) + X_t^{[i]} + \left(\sum_{j=i+1}^{m-1} X^{[j]}_{t-|\delta_{j,i}|}\right) \quad (39)$$

where

$$|\delta_{j,i}| \triangleq |\mathsf{length}(su_jw_0d) - \mathsf{length}(su_iw_0d)|$$

is the absolute value of the difference between the distance from s to e^* using detour u_j and the distance from s to e^* using detour u_i. By our assumption in (37), for those $j < i$ the packet $X_t^{[j]}$ sent through the primary path $su_jw_{m-1-j}d$ will arrive at e^* earlier than $X_t^{[i]}$ does. This is why when focusing on $X_t^{[i]}$ the corruption for those $j < i$ is of the form $X^{[j]}_{t+|\delta_{j,i}|}$. Namely, those j values correspond to corruption from the "future packets." Similarly, for those $j > i$ the packet $X_t^{[j]}$ sent through the primary path $su_jw_{m-1-j}d$ will arrive at e^* later than $X_t^{[i]}$ does. The corruption is thus of the form $X^{[j]}_{t-|\delta_{j,i}|}$, which corresponds to the corruption from the "old packets."

We now discuss how to use the $\{(A_l, B_l) : l \in [1, m-1]\}$ parallel pipes. Again, we consider a fixed i value and the packet received by d through path $su_iw_{m-1-i}d$. Without loss of generality, we assume $i \geq 1$ and the scenario of $i = 0$ is a degenerate case. Recall that by our construction A_i is located in paths $su_jw_{m-1-j}d$ for all $j \in [0, i-1]$ but not in the path of interest $su_iw_{m-1-i}d$. Namely, the location of A_i allows node A_i to observe the interference term $\left(\sum_{j=0}^{i-1} X^{[j]}_{t+|\delta_{j,i}|}\right)$ before the interference corrupts the desired $X_t^{[i]}$ term.

Also recall that by our construction B_i is located in the path su_iw_{m-1-i} and the (A_i, B_i) parallel pipe in the secondary component incurs the same amount of delay as the path from A_i to B_i in the primary component. As a result, node A_i can transmit the interference term $\left(\sum_{j=0}^{i-1} X^{[j]}_{t+|\delta_{j,i}|}\right)$ to node B_i through the parallel pipe and node B_i can use this information to subtract the corruption from the linear sum in (39). After the subtraction performed by B_i, d will receive

$$X_t^{[i]} + \left(\sum_{j=i+1}^{m-1} X_{t-|\delta_{j,i}|}^{[j]} \right).$$

through the primary path $su_iw_{m-1-i}d$ by the deadline $t+\mathsf{D}_m$. Since the remaining corruption term $\left(\sum_{j=i+1}^{m-1} X_{t-|\delta_{j,i}|}^{[j]}\right)$ is resulted from the "old packets", d can use its existing knowledge about the previously decoded packets to subtract the corruption and decode the desired $X_t^{[i]}$ along the primary path $su_iw_{m-1-i}d$. Since d can decode $X_t^{[i]}$ through $su_iw_{m-1-i}d$ for all $i \in [0, m-1]$, the delay-constrained NC throughput is m packets per time slot. Since the min-cut value from s to d is also m, the above scheme is throughput optimal and the delay-constrained NC capacity R^*_{NC} is indeed m packets per time slot. ■

We now quantify the routing-based capacity R^*_{route}.

Lemma 2: For any given $m \geq 2$, the delay-constrained routing capacity of (G_m, D_m) is $R^*_{\text{route}} = m-1+2^{-(m-1)}$ packets per time slot.

Proof: We prove this lemma by providing a primal and a dual solutions for (1)–(2) and for (3)–(4), respectively, and then show that the duality gap is zero.

For the primal variables x_P, we define $2m-1$ paths, denoted by $\{P_i, i \in [0, m-1]\}$ and $\{Q_j : j \in [1, m-1]\}$, respectively. They are

$$P_i \triangleq su_iw_{m-1-i}d \tag{40}$$

$$Q_j \triangleq su_{j-1}A_jB_jw_{m-1-j}d. \tag{41}$$

Namely, P_i is one of the *primary paths*. Path Q_j is a path that uses the (A_j, B_j) pipe in the secondary component and before using (A_j, B_j) the path Q_j uses the detour corresponding to u_{j-1}. Then we set the primal variables as

$$\begin{aligned} &x_{P_0} = 2^{-(m-1)}, \\ &\forall i \in \{1, \cdots, m-1\},\ x_{P_i} = 2^{-(m-i)}, \\ &\forall j \in \{1, \cdots, m-1\},\ x_{Q_j} = 1 - 2^{-(m-j)}, \\ &\text{and all other } x_P = 0. \end{aligned} \tag{42}$$

One can verify that the above x_P assignment is a feasible primal solution with the objective value being $\sum_{P \in \mathcal{P}_{\mathsf{D}_m}} x_P = m - 1 + 2^{-(m-1)}$.

For the dual variables y_e, we consider $2m-1$ edges: For all $i \in [1, m-1]$, we define e_{A_i} as the unique edge satisfying $\mathsf{head}(e_{A_i}) = A_i$ and e_{B_i} as the unique edge satisfying $\mathsf{tail}(e_{B_i}) = B_i$. Namely, e_{A_i} is the edge that is the direct upstream of node A_i and e_{B_i} is the edge that is the direct downstream of node B_i. Recall that e^* is the bottleneck edge in the center. Then we set the dual variables as

$$\begin{aligned} &y_{e^*} = 2^{-(m-1)}, \\ &\forall i \in \{1, \cdots, m-1\},\ y_{e_{A_i}} = 2^{-i},\ y_{e_{B_i}} = 1 - 2^{-i}, \\ &\text{and all other } y_e = 0. \end{aligned}$$

One can verify that the above y_e assignment is a feasible dual solution with the objective value being $\sum_e y_e c_e = m - 1 + 2^{-(m-1)}$. The proof is thus complete. ■

In sum, the NC $\mathsf{gain}_m = \frac{m}{m-1+2^{-(m-1)}}$ for the m-th network instance (G_m, D_m). One can easily verify that $\max_{m\geq 2} \mathsf{gain}_m = \frac{4}{3}$, which is attained by either $m = 2$ or $m = 3$.

Remark: From the surface, the above construction should yield high NC gain since all m primary paths $su_iw_{m-1-i}d$ use the same bottleneck edge e^*. However, detailed analysis in Lemmas 1 and 2 shows that even if we create a highly-congested bottleneck e^*, the NC gain does not increase any further[14] and is still upper bounded by $\frac{4}{3}$. The reason is that in the proposed optimal NC solution, the parallel pipes are used to carry the side information that will be used for subtracting the "corruption caused by the future packets". However, for a routing solution, we can directly use those parallel pipes to carry the uncoded information, see (41) and (42). As a result, simply increasing the congestion level at e^* does not lead to a network instance with a larger NC gain, and the simplest example with $m = 2$ sill admits the largest NC gain $\frac{4}{3}$ we have found thus far. In the next sections, we will demonstrate how to take advantage of the special structure of (G_m, D_m) for large m and use it to design a network instance with NC gain arbitrarily close to 2.

APPENDIX D
A GENIE-AIDED ITERATIVE CONSTRUCTION

In this section, we will present the main principles of our iterative construction, which will be based on the new concepts of *throughput/delay (T/D) spectrum* and *T/D spread.*

A. The Throughput/Delay (T/D) spectrum and spread

Given any network instance, the delay-constrained NC capacity can be written as a function $\mathcal{R}_{\text{NC}}(\mathsf{D})$ of the delay requirement D. Here we use the calligraphic $\mathcal{R}$ to emphasize that it is a function of D. Obviously $\mathcal{R}_{\text{NC}}(\mathsf{D})$ is non-decreasing with respect to D. Similarly, the delay-constrained routing capacity can be defined as $\mathcal{R}_{\text{route}}(\mathsf{D})$. We call this pair of functions $(\mathcal{R}_{\text{NC}}(\mathsf{D}), \mathcal{R}_{\text{route}}(\mathsf{D}))$ the throughput/delay (T/D) spectrum of the network. For example, by simple computation one can show that the network in Fig. 2 has

$$\mathcal{R}_{\text{NC}}(\mathsf{D}) = \begin{cases} 0 & \text{if } \mathsf{D} \leq 4 \\ 1 & \text{if } \mathsf{D} = 5 \\ 2 & \text{if } \mathsf{D} \geq 6 \end{cases}$$

$$\mathcal{R}_{\text{route}}(\mathsf{D}) = \begin{cases} 0 & \text{if } \mathsf{D} \leq 4 \\ 1 & \text{if } \mathsf{D} = 5 \\ 1.5 & \text{if } \mathsf{D} = 6 \\ 2 & \text{if } \mathsf{D} \geq 7 \end{cases}$$

Although the T/D spectrum $(\mathcal{R}_{\text{NC}}(\mathsf{D}), \mathcal{R}_{\text{route}}(\mathsf{D}))$ is well-defined, it may be difficult to compute for a general network topology. On the other hand, the T/D spectrum $(\mathcal{R}_{\text{NC}}(\mathsf{D}), \mathcal{R}_{\text{route}}(\mathsf{D}))$ can be used to compute a simpler concept called the *T/D spread.*

[14] $\mathsf{gain}_m = \frac{m}{m-1+2^{-(m-1)}}$ actually decreases monotonically with respect to m.

Definition 5: For any network with T/D spectrum $(\mathcal{R}_{\text{NC}}(\mathsf{D}), \mathcal{R}_{\text{route}}(\mathsf{D}))$, the T/D spread around a given D is a tuple $(\mathsf{gain}, \Delta_L, \Delta_H)_{\mathsf{D}}$, where $\mathsf{gain} \triangleq \frac{\mathcal{R}_{\text{NC}}(\mathsf{D})}{\mathcal{R}_{\text{route}}(\mathsf{D})}$ and

$$\Delta_L \triangleq \sup\{x \in \mathbb{N} : \mathcal{R}_{\text{NC}}(\mathsf{D} - x) > 0\}; \quad (43)$$

$$\text{and } \Delta_H \triangleq \inf\{x \in \mathbb{N} : \mathcal{R}_{\text{route}}(\mathsf{D} + x) > \mathcal{R}_{\text{route}}(\mathsf{D})\}. \quad (44)$$

Namely, whenever the delay requirement satisfies $\mathsf{D}' < \mathsf{D} - \Delta_L$, we will have $\mathcal{R}_{\text{NC}}(\mathsf{D}') = \mathcal{R}_{\text{route}}(\mathsf{D}') = 0$. Whenever the delay requirement satisfies $\mathsf{D} \leq \mathsf{D}' < \mathsf{D} + \Delta_H$, we have $\mathcal{R}_{\text{route}}(\mathsf{D}') = \mathcal{R}_{\text{route}}(\mathsf{D})$. Intuitively, Δ_L describes the lower spread around D before the throughput $\mathcal{R}_{\text{NC}}(\mathsf{D}')$ and $\mathcal{R}_{\text{route}}(\mathsf{D}')$ drop completely to zero, and Δ_H describes the upper spread before the routing-based throughput $\mathcal{R}_{\text{route}}(\mathsf{D}')$ increases.

For example, with $\mathsf{D} = 6$ the network in Fig. 2 has $(\mathsf{gain}, \Delta_L, \Delta_H)_{\mathsf{D}=6} = (\frac{4}{3}, 1, 1)$. We sometimes slightly abuse the notation and refer the pair (Δ_L, Δ_H) as the T/D spread. It should be clear from the context whether the term T/D spread is referring to a tuple or a pair.

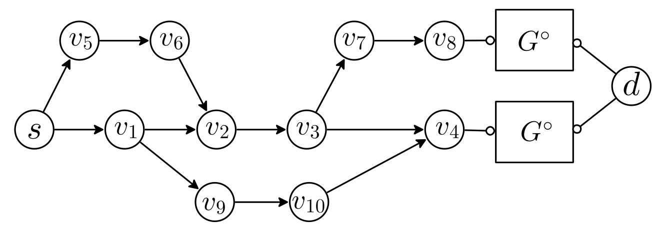
B. Illustration of a genie-aided construction

The example in Fig. 2 has T/D spread being $(\mathsf{gain}, \Delta_L, \Delta_H)_{\mathsf{D}=6} = (\frac{4}{3}, 1, 1)$. For easier reference, we denote the network in Fig. 2 by $\tilde{G}$. In this subsection, we assume that there is a genie that can convert the network $\tilde{G}$ to another finite network G° such that G° has T/D spread being $(\mathsf{gain}^\circ, \Delta_L^\circ, \Delta_H^\circ)_{\mathsf{D}^\circ} = (\frac{4}{3}, 0, \infty)$ around a new delay constraint D° that may be different from $\mathsf{D} = 6$. Namely, the resulting G° has the same NC gain as the original $\tilde{G}$ but has a different T/D spread. In general, such a conversion is impossible since one can prove that any network instance (G, D) with gain > 1 must have $\Delta_L > 0$ and $\Delta_H < \infty$. However, for sake of discussion, we assume such a genie exists and call the resulting network G° a *fictitious network* to distinguish it from an actual network instance.

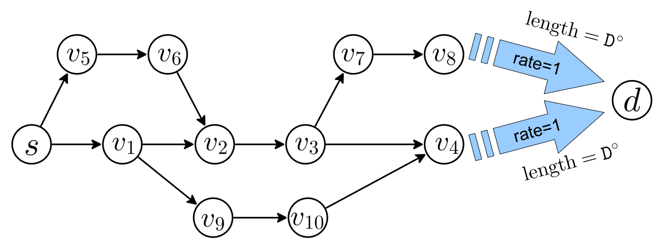
We now demonstrate how the fictitious[15] network G° can be used to construct a network instance of (G, D) with gain $= \frac{16}{11} > \frac{4}{3}$.

Without loss of generality we assume that $\mathcal{R}_{\text{NC}}(\mathsf{D}^\circ) = 1$ and $\mathcal{R}_{\text{route}}(\mathsf{D}^\circ) = \frac{3}{4}$ for the fictitious network G° since the corresponding gain is $\frac{4}{3}$. This can be achieved by scaling the capacity of each edge of G° proportionally until $\mathcal{R}_{\text{NC}}(\mathsf{D}^\circ) = 1$ and $\mathcal{R}_{\text{route}}(\mathsf{D}^\circ) = \frac{3}{4}$. Our construction will combine the original network in Fig. 2 with the new fictitious network G°. Specifically, we replace each of the two edges (v_4, d) and (v_8, d) of Fig. 2 by a copy of the G° network, respectively.[16] The description of the topology of the new network G is now complete and the resulting graph is illustrated in Fig. 8(a).

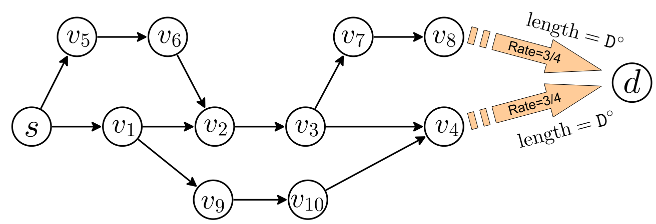
Recall that the NC gain of G° is $\frac{4}{3}$ at some fixed delay constraint D°. For the new network in Fig. 8(a), we set the new delay requirement to be $\mathsf{D} = 5 + \mathsf{D}^\circ$. The description of the network instance (G, D) is now complete. What remains to prove is that the new G in Fig. 8(a) has gain $= \frac{16}{11}$ when $\mathsf{D} = 5 + \mathsf{D}^\circ$.



(a) A new network instance G based on the fictitious network G°.



(b) The equivalent network from an NC's perspective.



(c) The equivalent network from a routing's perspective.

Fig. 8. Three closely related network instances with the same delay requirement $\mathsf{D} = 5 + \mathsf{D}^\circ$. The fictitious network G° has T/D spread $(\mathsf{gain}^\circ, \Delta_L^\circ, \Delta_H^\circ) = (\frac{4}{3}, 0, \infty)$.

We now argue that the delay-constrained NC capacity is still 2 packets per time slot. The reason is that since $\mathcal{R}_{\text{NC}}(\mathsf{D}^\circ) = 1$, we can always send 1 packet per time slot through the input terminal of G°, perform NC in the interior of G°, and extract the information packets from the output terminal of G° after D° slots. Therefore, from a NC's perspective, the network in Fig. 8(a) is no different than replacing the (v_4, d) and (v_8, d) edges by two paths of capacity 1 and length D°, respectively. See Fig. 8(b). Since we set the new delay requirement to be $\mathsf{D} = 5 + \mathsf{D}^\circ$, we can use the same analysis as in Section IV-B to prove that the NC capacity is 2 packets per time slot. The subtle difference herein is that the achievability scheme for Fig. 2 simply *forwards* the coded packets along the (v_4, d) and (v_8, d) edges. The achievability scheme for Fig. 8(a) needs to *perform the optimal NC solution* associated with G° when sending packets along the copies of G° that connects v_4 and v_8 to the destination d.

We now argue that the delay-constrained routing capacity of Fig. 8(a) is $\frac{11}{8}$ packets per time. To that end, the following Lemma 3 first proves that the network in Fig. 8(a) has the same delay-constrained routing capacity as the network in Fig. 8(c), which replaces the (v_4, d) and (v_8, d) edges by paths of capacity $\frac{3}{4}$ and length D°. Using Lemma 3 we can then compute the delay-constrained routing capacity of Fig. 8(a) by applying the LP computation (1)–(2) to Fig. 8(c). The end

[15]In Appendix E, we will provide a construction that directly uses $\tilde{G}$, without using the fictitious network G°.

[16]Our construction treats G° as a black box and does not depend on the actual topology of G°.

result shows that $R^*_{\text{route}} = \frac{11}{8}$ for both Figs. 8(a) and 8(c). The overall NC gain for the network in Fig. 8(a) is thus $\mathsf{gain} = \frac{2}{11/8} = \frac{16}{11}$.

Lemma 3: The networks in Figs. 8(a) and 8(c) have the same routing-based capacity R^*_{route}.

Proof: For easier reference, we call the network in Fig. 8(a) a G°*-compound network* and the network in Fig. 8(c) a *routing-equivalent network*.

It is easy to see that the delay-constrained routing capacity of the routing-equivalent network (Fig. 8(c)) is always no larger than that of the G°-compound network (Fig. 8(a)). The reason is that for whatever routing solution of the routing-equivalent network, we can always adapt it and find a routing solution for the G°-compound network since by our construction the constituent subnetwork G° is capable of supporting routing rate $\frac{3}{4}$ within a hard delay requirement D°.

We now argue that the delay-constrained routing capacity of the G°-compound network (Fig. 8(a)) is no larger than that of the routing-equivalent network (Fig. 8(c)). This direction is non-trivial since we do not know the underlying topology of the subnetwork G° and the proof needs to hold for any G° with T/D spread $(\mathsf{gain}^\circ, \Delta^\circ_L, \Delta^\circ_H) = (\frac{4}{3}, 0, \infty)$. Our proof consists of the following three observations.

Observation 1: Any path $\tilde{P}$ in the G°-compound network can be mapped to a path P in the routing-equivalent network by tracing the corresponding routes.

Observation 1 is self-explanatory. For example, $sv_1v_2v_3v_7v_8G^\circ d$ in Fig. 8(a) can be mapped to the path $sv_1v_2v_3v_7v_8d$ in Fig. 8(c).

We now make the second observation:

Observation 2: For any path $\tilde{P}$ in the G°-compound network, denote its image path in the routing-equivalent network by P. If $\mathsf{length}(P) > \mathsf{D}$*, then we must also have* $\mathsf{length}(\tilde{P}) > \mathsf{D}$.

The proof of Observation 2 is as follows. Consider any $\tilde{P}$ in the G°-compound network, of which the image path P in the routing-equivalent network satisfies $\mathsf{length}(P) > \mathsf{D}$. Since the only deadline-violating path (those with length $> \mathsf{D}$) in the routing-equivalent network (Fig. 8(c)) is $sv_5v_6v_2v_3v_7v_8d$, we must have $P = sv_5v_6v_2v_3v_7v_8d$ and $\tilde{P}$ must be of the form $\tilde{P} = sv_5v_6v_2v_3v_7v_8G^\circ d$. By the definition of the T/D spread in (43), G° cannot carry any positive routing rate from the input terminal of G° to the output terminal of G° in less than $\mathsf{D}^\circ - \Delta^\circ_L$ time slots. It means that any path connecting the input/output nodes of G° must have length no less than $\mathsf{D}^\circ - \Delta^\circ_L$ hops. Therefore, the path $\tilde{P} = sv_5v_6v_2v_3v_7v_8G^\circ d$ in Fig. 8(a) must have at least $6 + (\mathsf{D}^\circ - \Delta^\circ_L)$ hops. Since $\Delta^\circ_L = 0$, we have $\mathsf{length}(\tilde{P}) > \mathsf{D}$. The proof of Observation 2 is complete.

Observations 1 and 2 lead to the following results. If we use $\tilde{\mathcal{P}}_\mathsf{D}$ to denote the set of all deadline-respecting paths (those with length $\leq \mathsf{D}$) in the G°-compound network (Fig. 8(a)) and use $\mathcal{P}_\mathsf{D} = \{P_1, P_2, \cdots, P_{|\mathcal{P}_\mathsf{D}|}\}$ to denote the set of all deadline-respecting path in the routing-equivalent network (Fig. 8(c)), then we can partition $\tilde{\mathcal{P}}_\mathsf{D}$ into

$$\tilde{\mathcal{P}}_\mathsf{D} = \bigcup_{i=1}^{|\mathcal{P}_\mathsf{D}|} \tilde{\mathcal{P}}_i \tag{45}$$

where each disjoint subset $\tilde{\mathcal{P}}_i$ in the G°-compound network corresponds to a deadline-respecting path P_i in the routing-equivalent network. The reason is as follows. By Observation 1 every deadline-respecting path $\tilde{P} \in \tilde{\mathcal{P}}_\mathsf{D}$ can be mapped to a path in the routing-equivalent network. By Observation 2 the image path (after the mapping) must be deadline respecting as well. As a result, we can partition $\tilde{\mathcal{P}}_\mathsf{D}$ based on their image paths being P_1 to $P_{|\mathcal{P}_\mathsf{D}|}$. We thus have (45).

We now make the final observation.

Observation 3: Any deadline-respecting path in the G°-compound network must go through one and only one G° network. Furthermore, the corresponding sub-path inside G° must have length $< (\mathsf{D}^\circ + \Delta^\circ_H)$.

The proof of this observation is almost self-explanatory since (i) The collection of the two G° in Fig. 8(a) form a *cut*; (ii) The two G° cannot reach each other; and (iii) In this proof we assume the fictitious network G° has $\Delta^\circ_H = \infty$. However, the importance of this observation is significant. As will be proven in the next paragraph, Observation 3 implies that *from a routing perspective, each of the two G° subnetworks in the compound network is no different than a pipe of length* D° *that supports rate $\mathcal{R}_{route}(\mathsf{D}^\circ)$.*

The reason is as follows. Consider any arbitrary G° subnetwork in the compound network. We are interested in the deadline-respecting paths (those having length $\leq \mathsf{D} = 5 + \mathsf{D}^\circ$) that go through the G° of interest. Observation 3 implies that each of those deadline-respecting paths in the compound network uses a sub-path of length $\leq \mathsf{D}^\circ + \Delta^\circ_H - 1$ in G°. Therefore, the sum of all rates assigned to those deadline-respecting paths must be upper bounded by $\mathcal{R}_{\text{route}}(\mathsf{D}^\circ + \Delta^\circ_H - 1)$, the largest supportable rate when performing routing over (sub-)paths of length $\leq \mathsf{D}^\circ + \Delta^\circ_H - 1$ in G°. At the same time, by our T/D spread definition we have $\mathcal{R}_{\text{route}}(\mathsf{D}^\circ) = \mathcal{R}_{\text{route}}(\mathsf{D}^\circ + \Delta^\circ_H - 1)$. Therefore, the sum-rate of all deadline-respecting paths in the overall compound network that use G° must be upper bounded by $\mathcal{R}_{\text{route}}(\mathsf{D}^\circ)$. The G° subnetwork is thus no different than a pipe of length D° and capacity $\mathcal{R}_{\text{route}}(\mathsf{D}^\circ)$ from the routing's perspective.

With Observations 1 to 3, we are ready to complete the proof that the delay-constrained routing capacity of the G°-compound network (Fig. 8(a)) is no larger than the delay-constrained capacity of the routing-equivalent network (Fig. 8(c)).

Denote the two G° subnetworks in the G°-compound network (Fig. 8(a)) by G°_k, where $k = 1$ or 2 depending on which G° we choose. For any LP solution $\{x_{\tilde{P}} : \tilde{P} \in \tilde{\mathcal{P}}_\mathsf{D}\}$ of the

G°-compound network and for any $k \in \{1, 2\}$, we have

$$\sum_{i:P_i \text{ uses } P(G_k^\circ)} \left(\sum_{\tilde{P} \in \tilde{\mathcal{P}}_i} x_{\tilde{P}} \right) = \sum_{\tilde{P} \in \tilde{\mathcal{P}}_{\mathsf{D}} \text{ using } G_k^\circ} x_{\tilde{P}} \tag{46}$$

$$\leq \mathcal{R}_{\text{route}}(\mathsf{D}^\circ) = \frac{3}{4} = c_{P(G_k^\circ)} \tag{47}$$

where $P(G_k^\circ)$ denotes the special sub-path in the routing-equivalent network (Fig. 8(c)) that corresponds to the G_k° sub-network of interest, and $c_{P(G_k^\circ)}$ is the capacity of the path $P(G_k^\circ)$ in the routing-equivalent network, which equals $c_{P(G_k^\circ)} = \frac{1}{\mathsf{gain}^\circ} = \frac{3}{4}$. The equality in (46) follows from Observation 2 and (45). The inequality in (47) follows from our discussion of Observation 3, i.e., the sum of routing rates of all paths using G_k° is upper bounded by $\mathcal{R}_{\text{route}}(\mathsf{D}^\circ)$.

In addition to (46)–(47), the LP solution $\{x_{\tilde{P}} : \tilde{P} \in \tilde{\mathcal{P}}_{\mathsf{D}}\}$ of the compound network (Fig. 8(a)) also satisfies that for any e that is not inside any G_k°, we have

$$c_e = 1 \geq \sum_{\tilde{P} \in \tilde{\mathcal{P}}_{\mathsf{D}} \text{ using } e} x_{\tilde{P}} = \sum_{i:P_i \text{ uses } e} \left(\sum_{\tilde{P} \in \tilde{\mathcal{P}}_i} x_{\tilde{P}} \right) \tag{48}$$

where $c_e = 1$ follows from our compound-network construction; the inequality is the edge capacity constraint for any routing solution of the compound network; and the final equality follows from (45). Furthermore, the overall delay-constrained routing capacity of the compound network can be written as

$$\max_{x_{\tilde{P}}} \sum_{\tilde{P} \in \tilde{\mathcal{P}}_{\mathsf{D}}} x_{\tilde{P}} = \max_{x_{\tilde{P}}} \sum_{i=1}^{|\mathcal{P}_{\mathsf{D}}|} \left(\sum_{\tilde{P} \in \tilde{\mathcal{P}}_i} x_{\tilde{P}} \right) \tag{49}$$

where the equality follows from (45).

We now prove that any feasible LP solution $\{x_{\tilde{P}} : \tilde{P} \in \tilde{\mathcal{P}}_{\mathsf{D}}\}$ for the G°-compound network can be used to construct another feasible LP solution $\{x_{P_i}^{[\text{pure}]} : i = 1, \cdots, |\mathcal{P}_{\mathsf{D}}|\}$ for the routing-equivalent network, and the resulting $\{x_{P_i}^{[\text{pure}]}\}$ has the same objective value as the original $\{x_{\tilde{P}}\}$. To that end, we simply set each $x_{P_i}^{[\text{pure}]} \triangleq \sum_{\tilde{P} \in \tilde{\mathcal{P}}_i} x_{\tilde{P}}$ for all i. The resulting $\{x_{P_i}^{[\text{pure}]}\}$ is feasible since (46) and (48) for the original $\{x_{\tilde{P}}\}$ ensure that the new $\{x_{P_i}^{[\text{pure}]}\}$ will satisfy the edge-capacity constraints for the routing-equivalent network. The new construction $\{x_{P_i}^{[\text{pure}]}\}$ also has the same objective value as that of $\{x_{\tilde{P}}\}$ since the objective (49) for the compound network can be transcribed as the objective for the routing-equivalent network.

The above argument shows that any feasible routing solution $\{x_{\tilde{P}}\}$ of the G°-compound network can be used to construct another feasible routing solution $\{x_{P_i}^{[\text{pure}]}\}$ of the routing-equivalent network with the end-to-end throughput. As a result, we have proven that the delay-constrained routing capacity of the compound network (Fig. 8(a)) is no larger than that of the routing-equivalent network (Fig. 8(c)). ■

C. An iterative genie-aided construction of network instances with $\mathsf{gain} > 2 - \epsilon$

Thus far, we have demonstrated how to find a network instance with $\mathsf{gain} = \frac{16}{11}$ when assuming there is a genie

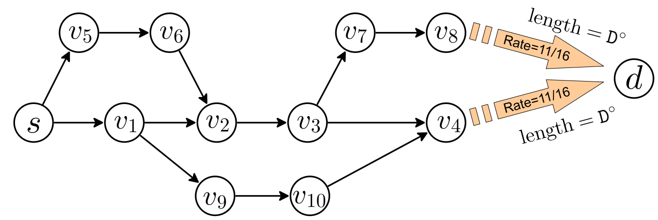


Fig. 9. The routing-equivalent network when the fictitious network G° has T/D spread $(\mathsf{gain}^\circ, \Delta_L^\circ, \Delta_H^\circ) = (\frac{16}{11}, 0, \infty)$.

who can convert Fig. 2 with $(\mathsf{gain}, \Delta_L, \delta_H) = (\frac{4}{3}, 1, 1)$ to a fictitious network G° with $(\mathsf{gain}^\circ, \Delta_L^\circ, \delta_H^\circ)_{\mathsf{D}^\circ} = (\frac{4}{3}, 0, \infty)$. The resulting graph is depicted in Fig. 8(a).

Suppose the same genie is very powerful and can convert any given network with T/D spread $(\mathsf{gain}, \Delta_L, \delta_H)$ to a fictitious network G° with $(\mathsf{gain}^\circ, \Delta_L^\circ, \delta_H^\circ)_{\mathsf{D}^\circ} = (\mathsf{gain}, 0, \infty)$, i.e., keeping the same gain but with the new $\Delta_L^\circ = 0$ and new $\Delta_H^\circ = \infty$. Then we can iteratively use the construction in Appendix D-B to further improve the NC gain.

That is, we use the genie to convert the network in Fig. 8(a) to another fictitious network G° with $(\mathsf{gain}^\circ, \Delta_L^\circ, \delta_H^\circ)_{\mathsf{D}^\circ} = (\frac{16}{11}, 0, \infty)$. Without loss of generality, we assume the NC capacity and the routing capacity of the new G° are $\mathcal{R}_{\text{NC}}(\mathsf{D}^\circ) = 1$ and $\mathcal{R}_{\text{route}}(\mathsf{D}^\circ) = \frac{11}{16}$, respectively, since $\mathsf{gain}^\circ = \frac{16}{11}$. Then we use the new G° and plug it into Fig. 8(a) again. By the same argument as used in Appendix D-B, the NC capacity of the new Fig. 8(a) (with the new G°) is still 2 packets per slot. Also, by the same argument as used in the proof of Lemma 3, the routing capacity of the new Fig. 8(a) is equal to the delay-constrained routing capacity of Fig. 9.

Using the LP formulation (1)–(2), one can prove that the routing capacity of Fig. 9 is $\frac{43}{32}$. The new NC gain over routing thus becomes $\mathsf{gain} = \frac{2}{43/32} = \frac{64}{43} > \frac{16}{11}$. We can then repeat the above process to continuously improve the NC gain.

By similar analysis, one can prove that if we start from any network with $\mathsf{gain}_{\text{old}} \leq 1.5$, then after converting it to a fictitious network G° with T/D spread $(\mathsf{gain}^\circ, \Delta_L^\circ, \delta_H^\circ)_{\mathsf{D}^\circ} = (\mathsf{gain}_{\text{old}}, 0, \infty)$ and using it to construct Fig. 8(a), the new network has NC gain being

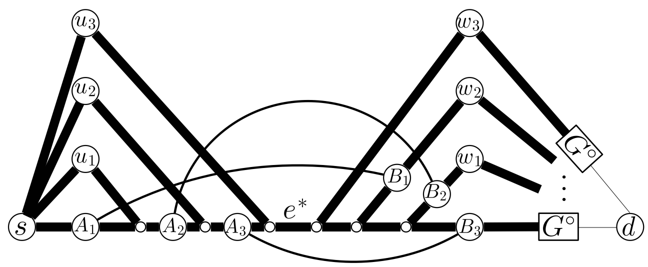
$$\mathsf{gain}_{\text{new}} = \frac{2 \cdot \mathsf{gain}_{\text{old}}}{0.5 + \mathsf{gain}_{\text{old}}}. \tag{50}$$

Since we assume that we have $\mathsf{gain}_{\text{old}} \leq 1.5$ to begin with, one can prove that $\mathsf{gain}_{\text{new}} \leq 1.5$ as well. Furthermore, by solving the fixed point equation of (50), one can prove that the above construction can generate network instances with NC gains arbitrarily close to 1.5 after a sufficiently large number of iterations. Although $\mathsf{gain} = 1.5$ is 12.5% improvement over the original gain $\frac{4}{3}$, the gain is still strictly bounded away from 2 when using the above iterative construction.

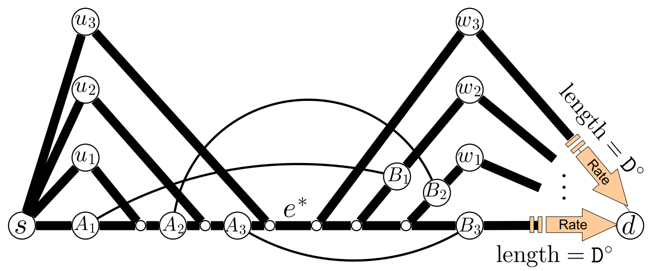
In the following, we demonstrate how to modify the above procedure and generate network instances with NC gains $> 2 - \epsilon$ for any arbitrarily given $\epsilon > 0$.

Step 1: For any given $\epsilon > 0$, we find an m value such that $2^{-(m-1)} < \epsilon$ and fix that m value throughout our construction.

Step 2: For any arbitrarily given network with NC gain satisfying $1 \leq \mathsf{gain}_{\text{old}} \leq 2 - 2^{-(m-1)}$, we use the genie to



(a) A new network instance G based on the fictitious network G°



(b) The corresponding routing-equivalent network. The capacity of each length-D° is $\mathsf{Rate} = \frac{1}{\mathsf{gain}_{\text{old}}}$.

Fig. 10. An illustration of the iterative genie-aided construction.

convert it to a fictitious network with $(\mathsf{gain}^\circ, \Delta_L^\circ, \delta_H^\circ)_{\mathsf{D}^\circ} = (\mathsf{gain}_{\text{old}}, 0, \infty)$. Without loss of generality, we assume the delay-constrained NC capacity of G° is $\mathcal{R}_{\text{NC}}(\mathsf{D}^\circ) = 1$ and the delay-constrained routing capacity of G° is $\mathcal{R}_{\text{route}}(\mathsf{D}^\circ) = \frac{1}{\mathsf{gain}_{\text{old}}}$.

Step 3: For the given m, we construct the network instance (G_m, D_m) that was previously described in Appendix C-B. For example, Fig. 7 is the high-level description of (G_m, D_m) when $m = 4$.

Step 4: For the constructed network G_m, there are m edges that enter the destination d. We replace each of the m edges by a copy of the previously created fictitious network G°. The description of the topology of the new network G is now complete. An example of the resulting graph for $m = 4$ is illustrated in Fig. 10(a).

Step 5: Recall that the delay requirement of the original G_m is denoted by D_m. For the new network in Fig. 10(a), we set the new delay requirement to be $\mathsf{D} = \mathsf{D}_m + \mathsf{D}^\circ - 1$. With the topology described in Step 4 and the delay requirement value D described in Step 5, the description of the new network instance (G, D) is now complete.

In the following Lemma 4, we will prove that the new (G, D) has the NC gain being

$$\mathsf{gain}_{\text{new}} = \frac{m \cdot \mathsf{gain}_{\text{old}}}{(m - 2 + 2^{-(m-1)}) + \mathsf{gain}_{\text{old}}}. \tag{51}$$

Since we assume that we have $1 \leq \mathsf{gain}_{\text{old}} \leq 2 - 2^{-(m-1)}$ to begin with, one can prove that $1 \leq \mathsf{gain}_{\text{old}} \leq \mathsf{gain}_{\text{new}} \leq 2 - 2^{-(m-1)}$ as well. We can then apply the genie to the new G and generate another fictitious network G° that can be used in Step 2 and the subsequent Steps 3 to 5. By iteratively repeating the above process, one can continuously improve the NC gain. By solving the fixed point equation of (51), one can prove that the above construction can generate network instances with NC gains arbitrarily close to $2 - 2^{-(m-1)}$. Since we choose the m value that satisfies $2^{-(m-1)} < \epsilon$, after a sufficiently large but finite number of iterations, the final resulting network instance will have gain $> 2 - \epsilon$ for the arbitrarily given $\epsilon > 0$.

Lemma 4: The new (G, D) generated in Steps 1–5 has NC gain equal to (51).

Proof: We first notice that the new (G, D) has delay-constrained NC capacity being m packets per second since we can reuse the NC strategy described in Lemma 1. The only change that needs to be made is the following. In the achievability scheme described in the proof of Lemma 1, we simply *forward* the coded packets along the edges entering d. In contrast, the achievability scheme of the new network (G, D) needs to *perform the optimal NC solution* associated with G° when sending packets along the copies of G° that enter the destination d.

We now prove that the new (G, D) has delay-constrained routing capacity being

$$R^*_{\text{route}} = \left(\frac{m - 2 + 2^{-(m-1)}}{\mathsf{gain}_{\text{old}}} + 1 \right) \text{ packets per slot.} \tag{52}$$

To that end, we first notice that all arguments in the proof of Lemma 3 can be applied verbatim to our new construction. Therefore, the routing capacity of the new Fig. 10(a) is equal to the delay-constrained routing capacity of Fig. 10(b), where Fig. 10(b) is the corresponding *routing-equivalent network* that replaces the G° subnetworks in Fig. 10(a) by paths of length D° and capacity $\frac{1}{\mathsf{gain}_{\text{old}}}$. To compute the routing capacity of Fig. 10(b), we follow the same approach as used in the proof of Lemma 2 by presenting a primal and a dual solution with zero duality gap.

For the primal variables x_P, we define $2m-1$ paths, denoted by $\{P_i, i \in [0, m-1]\}$ and $\{Q_j : j \in [1, m-1]\}$, in the same way as in (40) and (41). Then we set the primal variables to be

$$\begin{aligned}
& x_{P_0} = 1 - \frac{1}{\mathsf{gain}_{\text{old}}}(1 - 2^{-(m-1)}), \\
& \forall i \in \{1, \cdots, m-1\},\ x_{P_i} = \frac{1}{\mathsf{gain}_{\text{old}}} 2^{-(m-i)}, \\
& \forall j \in \{1, \cdots, m-1\},\ x_{Q_j} = \frac{1}{\mathsf{gain}_{\text{old}}}(1 - 2^{-(m-j)}), \\
& \text{and all other } x_P = 0.
\end{aligned}$$

One can verify that with the assumption of $\mathsf{gain}_{\text{old}} \leq 2 - 2^{-(m-1)}$, the above $\{x_P\}$ assignment is a feasible primal solution with the objective value being $\sum_P x_P = \frac{m-2+2^{-(m-1)}}{\mathsf{gain}_{\text{old}}} + 1$.

For the dual variables y_e, we consider $2m - 1$ edges: For all $i \in [1, m-1]$, we define e_{A_i} as the unique edge satisfying $\mathsf{head}(e_{A_i}) = A_i$ and e_{B_i} as the edge in $\text{In}(d)$ that is downstream of B_i. Namely, e_{A_i} is the edge that is the direct upstream of node A_i and has $c_{e_{A_i}} = 1$, and e_{B_i} is the edge that has capacity $c_e = \frac{1}{\mathsf{gain}_{\text{old}}}$ and is a downstream edge of node B_i. Recall that e^* is the bottleneck edge in the center.

Then we set the dual variables to be

$$y_{e^*} = 2^{-(m-1)},$$
$$\forall i \in \{1, \cdots, m-1\},\ y_{e_{A_i}} = 2^{-i},\ y_{e_{B_i}} = 1 - 2^{-i},$$
and all other $y_e = 0$.

One can verify that the above $\{y_e\}$ assignment is a feasible dual solution with the objective value being

$$\sum_e y_e c_e = \left(y_{e^*} + \sum_{i=1}^{m-1} y_{e_{A_i}} \right) \cdot 1 + \left(\sum_{i=1}^{m-1} y_{e_{B_i}} \right) \cdot \frac{1}{\mathsf{gain}_{\text{old}}}$$
$$= 1 + \frac{m - 2 + 2^{-(m-1)}}{\mathsf{gain}_{\text{old}}}.$$

The proof is thus complete. ■

In summary, we have provided a genie-aided iterative construction that leads to network instances with NC gain arbitrarily close to 2.

APPENDIX E
AN ITERATIVE CONSTRUCTION WITHOUT USING ANY GENIE

In this section, we first demonstrate our proposed construction on the simplest example, which is in parallel with our discussion in Appendix D-B. Then we describe our proposed construction for the most general setting, which is in parallel with our discussion in Appendix D-C.

A. An iterative construction without using any fictitious networks

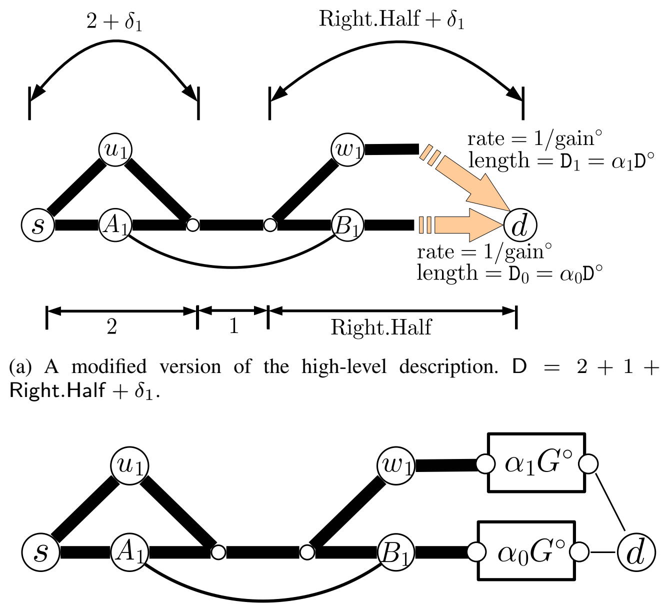
In this subsection, we will describe a compound network construction that uses any arbitrarily given network instance $(G^\circ, \mathsf{D}^\circ)$ with $(\mathsf{gain}^\circ, \Delta_L^\circ, \Delta_H^\circ)_{\mathsf{D}^\circ} = (\frac{4}{3}, 1, 1)$ to construct a network instance with $\mathsf{gain} = \frac{16}{11}$. Such a construction is in parallel to the discussion in Appendix D-B except that our construction is without the aid of a genie. Since our construction can be easily generalized to any $1 \le \mathsf{gain}^\circ \le 1.5$, $\Delta_L^\circ > 0$, and $\Delta_H^\circ < \infty$, we will simply use the tuple $(\mathsf{gain}^\circ, \Delta_L^\circ, \Delta_H^\circ)_{\mathsf{D}^\circ}$ when describing our scheme, instead of the actual numbers $(\frac{4}{3}, 1, 1)$. Without loss of generality, we also assume that for this G° we have $\mathcal{R}_{\text{NC}}(\mathsf{D}^\circ) = 1$ and $\mathcal{R}_{\text{route}}(\mathsf{D}^\circ) = \frac{1}{\mathsf{gain}^\circ}$.

Our construction needs the following definition.

Definition 6: For any given graph G and any positive integer α, the α-elongated version of G, denoted by $\alpha \cdot G$ or simply αG for brevity, is obtained by replacing every edge in G by a path of length α. The capacity of the new path in αG is set to be the same as the capacity c_e of the edge e in G it replaces.

Remark: It is self-explanatory that if the original graph G has the T/D spread being $(\mathsf{gain}, \Delta_L, \Delta_H)_{\mathsf{D}}$ at some D, then the new graph αG has the T/D spread being $(\mathsf{gain}, \alpha\Delta_L, \alpha\Delta_H)_{\alpha\mathsf{D}}$ at a new delay point $\alpha\mathsf{D}$.

Given any $(G^\circ, \mathsf{D}^\circ)$ with $(\mathsf{gain}^\circ, \Delta_L^\circ, \Delta_H^\circ)_{\mathsf{D}^\circ}$, we start from the high-level description in Fig. 6. For any given four strictly positive integer values α_1, α_0, δ_1, and $\mathsf{Right.Half}$, we can modify Fig. 6 and construct the network in Fig. 11(a). Specifically, the length of the detour using u_1 (the left triangle)



(a) A modified version of the high-level description. $\mathsf{D} = 2 + 1 + \mathsf{Right.Half} + \delta_1$.

(b) The corresponding compound network.

Fig. 11. An illustration of the iterative genie-free construction for the simple example.

is set to $2 + \delta_1$. The length of the detour using w_1 (the right triangle) is set to $\mathsf{Right.Half} + \delta_1$. The delay requirement value is set to $\mathsf{D} \triangleq 2 + 1 + \mathsf{Right.Half} + \delta_1$. One can easily check that such a D value will ensure that any deadline-respecting path can detour at most once. We also require that the last $\mathsf{D}_1 \triangleq \alpha_1 \cdot \mathsf{D}^\circ$ unit-capacity edges of the route from w_1 to d is replaced by a path with length D_1 and capacity $\frac{1}{\mathsf{gain}^\circ}$. Similarly, the last $\mathsf{D}_0 \triangleq \alpha_0 \cdot \mathsf{D}^\circ$ unit-capacity edges of the route from w_0 to d is replaced by a path with length D_0 and capacity $\frac{1}{\mathsf{gain}^\circ}$. See Fig. 11(a) for detailed illustration.

We now describe how to choose the four parameter values α_1, α_0, δ_1, and $\mathsf{Right.Half}$. In particular, we require them to satisfy the following conditions, for which the integer constants $\mathsf{D} \triangleq 2 + 1 + \mathsf{Right.Half} + \delta_1$, $\mathsf{D}_0 \triangleq \alpha_0\mathsf{D}^\circ$, and $\mathsf{D}_1 \triangleq \alpha_1\mathsf{D}^\circ$ have been defined previously.

- The Feasibility Condition:

$$\mathsf{Right.Half} + \delta_1 \ge \mathsf{D}_1 + 1 \tag{53}$$
$$\mathsf{Right.Half} \ge \mathsf{D}_0 + 2. \tag{54}$$

- Condition 2:

$$(2 + \delta_1) + 1 + (\mathsf{Right.Half} + \delta_1) - \mathsf{D}_1 + (\mathsf{D}_1 - \alpha_1\Delta_L^\circ) > \mathsf{D}. \tag{55}$$

- Condition 3:

$$\mathsf{D} - (2 + 1 + (\mathsf{Right.Half} + \delta_1) - \mathsf{D}_1) < \mathsf{D}_1 + \alpha_1\Delta_H^\circ \tag{56}$$
$$\mathsf{D} - (2 + 1 + \mathsf{Right.Half} - \mathsf{D}_0) < \mathsf{D}_0 + \alpha_0\Delta_H^\circ. \tag{57}$$

The feasibility conditions in (53)–(54) are to ensure that the path length values in Fig. 11(a) are consistent. Specifically, the detour using w_1 has length $\mathsf{Right.Half} + \delta_1$. We need it to be strictly larger than D_1 so that after replacing the last D_1 unit-capacity edges of the route from w_1 to d by the special path with length D_1 and capacity $\frac{1}{\mathsf{gain}^\circ}$, we still have part of the w_1-detour being the regular unit-capacity edges (the thick lines in Fig. 11(a)). Similarly, when not detouring in the right triangle, we need to have the direct path length $\mathsf{Right.Half}$ to be strictly larger than D_0+1 so that even after replacing the last D_0 unit-capacity edges of the route from B_1 to d, part of the B_1-to-d path still has some regular unit-capacity edges (the thick lines in Fig. 11(a)). In our construction, we implicitly assume that node B_1 is the immediate downstream neighbor of the node where paths su_0w_0d and su_0w_1d diverges. The constant 2 in (54) takes into account the edge from the diverging node to B_1 and the edge following B_1. Note that in (53), we only require that that the length from the diverging node through w_1 to the *entrance node* of the $\alpha_1 G^\circ$-subnetwork to be at least 1. The reason is that the detour using node w_1 does not have the intermediate node B_1 and thus the feasibility condition (53), which aims to preserve the topology of the original network, can be slightly looser for the upper detour in the right triangle. More specifically, to preserve the topology of Fig. 11(a) (and also Fig. 11(b)) the entrance node of subnetwork $\alpha_1 G^\circ$ can coincide with node w_1 but the entrance node of subnetwork $\alpha_0 G^\circ$ cannot coincide with node B_1.

The intuition behind Conditions 2 and 3 will be explained shortly after.

In Appendix F, we will prove that for any given T/D spread $(\mathsf{gain}^\circ, \Delta_L^\circ, \Delta_H^\circ)_{\mathsf{D}^\circ}$ value, we can always find four strictly positive integer parameter values α_1, α_0, δ_1, and $\mathsf{Right.Half}$ satisfying (53) to (57). For example, if $(\mathsf{gain}^\circ, \Delta_L^\circ, \Delta_H^\circ)_{\mathsf{D}^\circ} = (\frac{4}{3}, 1, 1)$, then we can choose $\alpha_1 = 1$, $\alpha_0 = 3$, $\delta_1 = 2$, and $\mathsf{Right.Half} = 20$, which satisfy (53) to (57).

After fixing the α_1, α_0, δ_1, and $\mathsf{Right.Half}$ values, the routing-equivalent network (Fig. 11(a)) is uniquely determined. We now convert the routing-equivalent network to a compound network by replacing each special path of length $\mathsf{D}_i = \alpha_i \mathsf{D}^\circ$ by an α_i-elongated version of G° for all $i \in \{0, 1\}$. See Fig. 11(b) for illustration. We now have the following lemma.

Lemma 5: The compound network Fig. 11(b) has delay-constrained NC capacity 2 packets per time slot, and its delay-constrained routing capacity is equal to that of Fig. 11(a).

Using this lemma, we can compute the delay-constrained routing capacity of Fig. 11(b) by solving the delay-constrained routing capacity of Fig. 11(a). By (52) with the value of $m = 2$, one can prove that, assuming $\mathsf{gain}^\circ \leq 1.5$, the routing capacity of Fig. 11(a) is $\frac{0.5}{\mathsf{gain}^\circ} + 1$. The new NC gain is $\mathsf{gain} = \frac{2\mathsf{gain}^\circ}{0.5+\mathsf{gain}^\circ}$. Our construction without the genie has the same NC gain as the genie-aided construction in Appendix D-B.

Proof of Lemma 5: The first half of Lemma 5 can be proven by the same NC scheme as discussed in Appendix D-B. Namely, we perform the NC scheme previously described in Section IV-B and then when traversing over the $\alpha_0 G^\circ$ and $\alpha_1 G^\circ$ subnetworks, instead of forwarding the packets, we perform the optimal NC solution associated to the constituent G°.

For the second half of Lemma 5, we will prove that Lemma 3 holds for our new construction as well. To that end, we notice that as long as Observations 1 to 3 in the proof of Lemma 3 hold, then Lemma 3 holds. One can easily see that Observation 1 holds naturally. In the following we prove that Observations 2 and 3 hold for Figs. 11(b) and 11(a) as well.

To prove Observation 2, we notice that the only deadline-violating path in the routing-equivalent network Fig. 11(a) is the path corresponding to su_1w_1d. Therefore, we need to prove that in the compound network Fig. 11(b), any path of the form $su_1w_1(\alpha_1 G^\circ)d$ must have length $> \mathsf{D}$. By the definition of Δ_L° in (43) and $\mathsf{D}_1 \triangleq \alpha_1 \mathsf{D}^\circ$, any path connecting the input/output nodes of $\alpha_1 G^\circ$ must have length no less than $\mathsf{D}_1 - \alpha_1 \Delta_L^\circ$ hops. Therefore, any path of the form $su_1w_1(\alpha_1 G^\circ)d$ in Fig. 11(b) must have at least $(2+\delta_1+1+\mathsf{Right.Half}+\delta_1-\mathsf{D}_1)+(\mathsf{D}_1-\alpha_1\Delta_L^\circ)$ hops where $(2+\delta_1+1+\mathsf{Right.Half}+\delta_1-\mathsf{D}_1)$ is the number of hops from s to the input node of $\alpha_1 G^\circ$ using the path $su_1w_1(\alpha_1 G^\circ)d$. Since (55) in Condition 2 is true, all those paths have length $> \mathsf{D}$. Observation 2 is thus proven.

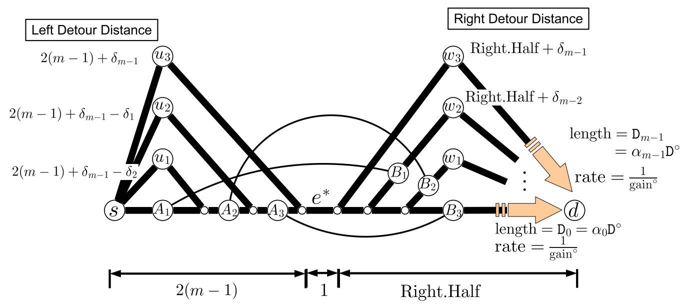
We now prove Observation 3. The first half of Observation 3 is straightforward since the two subnetworks $\alpha_0 G^\circ$ and $\alpha_1 G^\circ$ form a cut. To prove the second half, we first consider all the deadline-respecting paths (length $\leq \mathsf{D}$) in the compound network (Fig. 11(b)) that use the $\alpha_1 G^\circ$ network. Those paths must be of the form $su_0w_1(\alpha_1 G^\circ)d$ according to Observation 2. Since the partial path from s to the input node of $\alpha_1 G^\circ$ via su_0w_1d has $(2+1+(\mathsf{Right.Half}+\delta_1)-\mathsf{D}_1)$ number of hops, it implies that the sub-path within $\alpha_1 G^\circ$ must have length $\leq \mathsf{D}-(2+1+(\mathsf{Right.Half}+\delta_1)-\mathsf{D}_1)$. Since our construction satisfies (56), Observation 3 holds for all deadline-respecting paths in the compound network that use $\alpha_1 G^\circ$.

Let us now consider all the deadline-respecting paths (length $\leq \mathsf{D}$) in the compound network (Fig. 11(b)) that use the $\alpha_0 G^\circ$ network. We first quantify the shortest possible distance from s to the input terminal of $\alpha_0 G^\circ$ as follows. Since the (A_1, B_1) pipe has the same length as the distance from A_1 to B_1 in the primary component, we only need to consider the shortest path from s to the input terminal of $\alpha_0 G^\circ$ in the primary component without using the (A_1, B_1) pipe. One can easily see that such a shortest path must use the su_0w_0d path. As a result, the shortest possible distance from s to the input terminal of $\alpha_0 G^\circ$ is $(2+1+\mathsf{Right.Half}-\mathsf{D}_0)$ hops. This implies that for any deadline-respecting path in the compound network, the corresponding sub-path within $\alpha_0 G^\circ$ must have length $\leq \mathsf{D}-(2+1+\mathsf{Right.Half}-\mathsf{D}_0)$. Since our construction satisfies (57), Observation 3 holds for all deadline-respecting paths in the compound network that use $\alpha_0 G^\circ$.

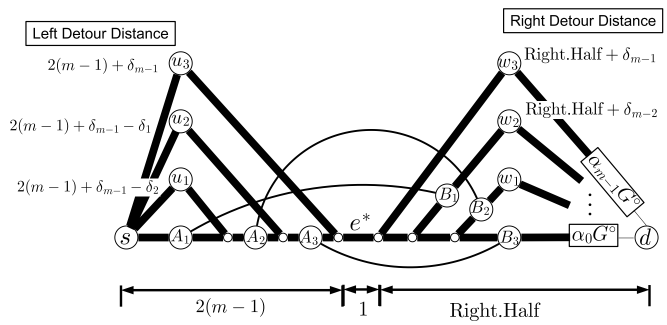
Since Observations 1 to 3 hold for our new construction, the second half of Lemma 5 is proven. ■

B. The generalized iterative construction without the help of a genie

Appendix E-A is a parallel version of Appendix D-B, where the latter is based on the help of a genie and the former is not. Similarly, Appendix E-B generalizes Appendix D-C in the sense that even without the help of a genie, we can still



(a) A modified version of the high-level description. $\mathsf{D} = 2(m-1) + 1 + \mathsf{Right.Half} + \delta_{m-1}$.



(b) The corresponding compound network.

Fig. 12. An illustration of the iterative genie-free construction for the general example with an arbitrarily given m value.

use some construction similar to the one in Appendix D-C to find a network instance with NC gain arbitrarily close to 2.

Our new construction is as follows.

Step 1: For any given $\epsilon > 0$, we find an m value such that $2^{-(m-1)} < \epsilon$ and fix that m value throughout our construction.

Step 2: Consider any arbitrarily given network with T/D spread $(\mathsf{gain}^\circ, \Delta_L^\circ, \Delta_H^\circ)$, which satisfies $\mathsf{gain}^\circ \leq 2 - 2^{-(m-2)}$, $\Delta_L^\circ \geq 1$ and $\Delta^\circ < \infty$. Without loss of generality, we also assume that for this G° we have $\mathcal{R}_{\text{NC}}(\mathsf{D}^\circ) = 1$ and $\mathcal{R}_{\text{route}}(\mathsf{D}^\circ) = \frac{1}{\mathsf{gain}^\circ}$. Using G°, we will construct a *routing-equivalent network* as follows.

We start from the high-level description for the m-th member (G_m, D_m) of the family described in Appendix C-B. Also see Fig. 7 for an illustration of $m = 4$. For any given $2m$ strictly positive integer values denoted by $\alpha_i, i \in [0, m-1]$, $\delta_j, j \in [1, m-1]$, and $\mathsf{Right.Half}$, we can modify Fig. 7 in a similar way as described in Fig. 11(a) and the resulting graph is described in Fig. 12(a).

Specifically, we set the length of the base of the left triangles to be $2(m-1)$. The reason is that the base of the left triangles contains $(m-1)$ segments, each is occupied by node A_i for $i \in [1, m-1]$. Assuming each segment is of 2 edges with A_i being the center node, the base of the left triangles is of length $2(m-1)$, see Fig. 12(a). The length of the base of the right triangles is set to $\mathsf{Right.Half}$.

The distance of the detour using w_i (the right triangle) is set to $\mathsf{Right.Half} + \delta_i$ for $i \in \{1, \cdots, m-1\}$, where we count the *detour distance* as the length from the head of bottleneck edge e^* to destination d using node w_i. For example, if $i = m-1$, then the length of the path along the upper edges of the biggest right triangle is $\mathsf{Right.Half} + \delta_{m-1}$. If $i = m-2$, then the detour path starts from the head of e^*, continues along the straight path to the first branching point, then diverts to w_{m-2}, and finally arrives at d. Our construction requires that such a detour path has length $\mathsf{Right.Half} + \delta_{m-2}$.

Because our construction has to satisfy $\mathsf{length}(su_iw_{m-1-i}d)$ being the same for all $i \in [0, m-1]$, see (36), we set the length of the left detour using u_i to be $2(m-1) + \delta_{m-1} - \delta_{m-1-i}$, where we define $\delta_0 \triangleq 0$ and we count the *detour distance* as the length from the source s to the tail of the bottleneck edge e^* using node u_i. For example, if $i = m-3$, then the detour path starts from s, uses the path corresponding to u_{m-3}, merges with the straight line su_0w_0d, and finally arrives at the tail of e^*. Our construction requires that such a detour path has length $2(m-1) + \delta_{m-1} - \delta_{m-1-i} = 2(m-1) + \delta_{m-1} - \delta_2$.

The delay requirement value is set to be

$$\begin{aligned} \mathsf{D} &\triangleq \mathsf{length}(su_iw_{m-1-i}d) \\ &= (2(m-1) + \delta_{m-1} - \delta_{m-1-i}) + 1 \\ &\quad + (\mathsf{Right.Half} + \delta_{m-1-i}) \\ &= 2(m-1) + 1 + (\mathsf{Right.Half} + \delta_{m-1}). \end{aligned} \tag{58}$$

See the high-level network description in (38) and see Fig. 12(a) for illustration.

Once the basic topology of the network is fixed, we also require that for all $i \in [1, m-1]$, the last $\mathsf{D}_i \triangleq \alpha_i \mathsf{D}^\circ$ unit-capacity edges of the route from w_i to d are replaced by a path with length D_i and capacity $\frac{1}{\mathsf{gain}^\circ}$. Similarly, the last $\mathsf{D}_0 \triangleq \alpha_0 \mathsf{D}^\circ$ unit-capacity edges of the straight route (directly from $\mathsf{head}(e^*)$ to d without using any detour) is replaced by a path with length D_0 and rate $\frac{1}{\mathsf{gain}^\circ}$. See Fig. 12(a) for illustration.

One can clearly see that the above construction is uniquely determined once the $2m$ integer values: $\{\alpha_i > 0 : i \in [0, m-1]\}$, $\{\delta_j > 0 : j \in [1, m-1]\}$, and $\mathsf{Right.Half} > 0$ are fixed. We now describe how to choose these parameter values. In particular, we require them to satisfy the following conditions, for which the integer constant D is defined in (58) and $\mathsf{D}_i \triangleq \alpha_i \mathsf{D}^\circ, \forall i \in [0, m-1]$.

- The Feasibility Condition:

$$\delta_{m-1} > \delta_{m-2} > \cdots > \delta_1 > \delta_0 = 0 \tag{59}$$

$$\mathsf{Right.Half} + \delta_{m-1} \geq \mathsf{D}_{m-1} + 1; \tag{60}$$

$$\mathsf{Right.Half} + \delta_i \geq \mathsf{D}_i + (m-i) + 1, \ \forall i \in [1, m-2]; \tag{61}$$

$$\mathsf{Right.Half} \geq \mathsf{D}_0 + (m-1) + 1. \tag{62}$$

- Condition 2: For all $j \in [1, m-1]$,

$$\begin{aligned} &(2(m-1) + \delta_{m-1} - \delta_{j-1}) + 1 + (\mathsf{Right.Half} + \delta_j - \mathsf{D}_j) \\ &\quad + (\mathsf{D}_j - \alpha_j \Delta_L^\circ) > \mathsf{D} \end{aligned} \tag{63}$$

$$\iff \delta_{j-1} < \delta_j - \alpha_j \Delta_L^\circ \tag{64}$$

where (64) follows from (63) by replacing D and D_i with their respective definitions.

- Condition 3: For all $i \in [0, m-1]$,

$$\begin{aligned} &\mathsf{D} - (2(m-1) + 1 + (\mathsf{Right.Half} + \delta_i - \mathsf{D}_i)) \\ &\quad < \mathsf{D}_i + \alpha_i \Delta_H^\circ \end{aligned} \tag{65}$$

$$\iff \delta_{m-1} - \alpha_i \Delta_H^\circ < \delta_i. \tag{66}$$

where (66) follows from (65) by replacing D and D_i with their respective definitions.

The feasibility conditions are to ensure that the path length values in Fig. 12(a) are consistent. Specifically, (59) ensures that the larger the triangle in the illustration, the longer the detour length. To explain (60), we consider the detour using w_{m-1}, which is of length $\mathsf{Right.Half} + \delta_{m-1}$. The feasibility condition (60) ensures after replacing the last D_{m-1} unit-capacity edges of the detour by a special path with length D_{m-1} and capacity $\frac{1}{\mathsf{gain}^\circ}$, we still have part of the w_{m-1} detour being the regular unit-capacity edges (the thick lines in Fig. 12(a)). Similarly, for all $i = m-2, m-3, \cdots, 1$, the detour using w_i has length $\mathsf{Right.Half} + \delta_i$. We then note that the path from the head of e^* to node B_{m-1-i} has $(m-i)$ hops. The feasibility condition (61) ensures that after replacing the last D_i unit-capacity edges of the route from w_i to d by a special path with length D_i and capacity $\frac{1}{\mathsf{gain}^\circ}$, we still have at least one edge after node B_{m-1-i} being the regular unit-capacity edges (the thick lines in Fig. 12(a)).

When not detouring at all in the right triangles, the direct path from $\mathsf{head}(e^*)$ to d has length $\mathsf{Right.Half}$. We also note that the path from the head of e^* to node B_{m-1} has $(m-1)$ hops. The feasibility condition (62) then ensures that after replacing the last D_0 unit-capacity edges of the route from B_{m-1} to d by a special path with length D_0 and capacity $\frac{1}{\mathsf{gain}^\circ}$, the B_{m-1}-to-d path still has at least one regular unit-capacity edge (the thick lines in Fig. 12(a)). Conditions 2 and 3 will be explained shortly after.

In Appendix F, we will prove that for any given T/D spread $(\mathsf{gain}^\circ, \Delta_L^\circ, \Delta_H^\circ)_{\mathsf{D}^\circ}$ value, we can always find $2m$ parameter values $\{\alpha_i > 0 : i = 0, \cdots, m-1\}$, $\{\delta_j > 0 : j = 1, \cdots, m-1\}$, and $\mathsf{Right.Half} > 0$ satisfying (59) to (66). Therefore, the construction of the routing-equivalent network (Fig. 12(a)) in Step 2 is always feasible.

Step 3: After constructing the routing-equivalent network (Fig. 12(a)) described in Step 2, we convert it to a compound network (Fig. 12(b)) by replacing each special path of length $\mathsf{D}_i = \alpha_i \mathsf{D}^\circ$ by an α_i-elongated version of G° for all $i \in [0, m-1]$ and keep the same delay requirement D value defined in (58). The description of the new compound network instance (G, D) is now complete.

The two networks constructed in Steps 1 to 3 satisfy the following lemma.

Lemma 6: The compound network Fig. 12(b) has delay-constrained NC capacity m packets per time slot, and its delay-constrained routing capacity is equal to that of Fig. 12(a).

Using this lemma, we can compute the delay-constrained routing capacity of Fig. 12(b) by solving the delay-constrained routing capacity of Fig. 12(a). By (52), one can prove that, assuming $\mathsf{gain}^\circ \leq 2 - 2^{-(m-1)}$, the routing capacity is $\frac{m-2+2^{-(m-1)}}{\mathsf{gain}^\circ} + 1$. The new NC gain is $\mathsf{gain} = \frac{m \cdot \mathsf{gain}^\circ}{m-2+2^{-(m-1)}+\mathsf{gain}^\circ}$. Our construction without the genie has the same NC gain as the genie-aided construction in Appendix D-C. By iteratively applying Steps 2 to 3 in this section, we can design a network with NC gain $> 2 - \epsilon$ for any given $\epsilon > 0$. Our construction is complete.

Proof of Lemma 6: The first half of Lemma 6 can be proven by the same NC scheme as discussed in Appendix D-C. Namely, we perform the NC scheme previously described in Appendix C-B and then when traversing over the $\alpha_i G^\circ$ subnetworks, instead of forwarding the packets, we perform the optimal NC solution associated to the constituent G°.

For the second half of Lemma 6, we will prove that Lemma 3 holds for our new construction as well. To that end, we notice that as long as Observations 1 to 3 in the proof of Lemma 3 hold, then Lemma 3 holds. One can easily see that Observation 1 holds naturally. In the following we prove that for any arbitrarily given $m \geq 2$, Observations 2 and 3 hold in our new genie-free construction. See Figs. 12(b) and 12(a) for illustration.

To prove Observation 2, we notice that the deadline-violating path in the routing-equivalent network Fig. 12(a) must be of the form su_iw_jd with $i + j \geq m$. As a result, we need to prove that any path of the form $su_iw_j(\alpha_j G^\circ)d$, $i + j \geq m$, in the compound network must have length $> \mathsf{D}$. Since the larger the triangle the longer the detour distance, we only need to prove that any path of the form $su_{m-j}w_j(\alpha_j G^\circ)d, j \in [1, m-1]$ in the compound network must have length $> \mathsf{D}$.

By the definition of Δ_L° in (43) and $\mathsf{D}_j \triangleq \alpha_j \mathsf{D}^\circ$, any path connecting the input/output nodes of $\alpha_j G^\circ$ must have length no less than $\mathsf{D}_j - \alpha_j \Delta_L^\circ$ hops. Therefore, any path of the form $su_{m-j}w_j(\alpha_j G^\circ)d$ in Fig. 12(b) must have at least $(2(m-1) + \delta_{m-1} - \delta_{j-1}) + 1 + (\mathsf{Right.Half} + \delta_j - \mathsf{D}_j) + (\mathsf{D}_j - \alpha_j \Delta_L^\circ)$ hops where $(2(m-1) + \delta_{m-1} - \delta_{j-1}) + 1 + (\mathsf{Right.Half} + \delta_j - \mathsf{D}_j)$ is the number of hops from s to the input node of $\alpha_j G^\circ$ using the path $su_{m-j}w_j(\alpha_j G^\circ)d$. Since (63) in Condition 2 is true, all those paths have length $> \mathsf{D}$. Observation 2 is thus proven.

We now prove Observation 3. The first half of Observation 3 is straightforward since the subnetworks $\alpha_j G^\circ, j \in [0, m-1]$ form a cut. To prove the second half, we first consider all the deadline-respecting paths (length $\leq \mathsf{D}$) in the compound network (Fig. 12(b)) that use the $\alpha_i G^\circ$ network for some $i \in [0, m-2]$. (The case in which $i = m-1$ is a degenerate case and follows similarly.) We first quantify the shortest possible distance from s to the input terminal of $\alpha_i G^\circ$ as follows. Since the (A_{m-1-i}, B_{m-1-i}) pipe has the same length as the distance from A_{m-1-i} to B_{m-1-i} in the primary component, we only need to consider the shortest path from s to the input terminal of $\alpha_i G^\circ$ in the primary component without using the (A_{m-1-i}, B_{m-1-i}) pipe. One can easily see that such a shortest path must use the su_0w_id path. As a result, the shortest possible distance from s to the input terminal of $\alpha_i G^\circ$ is $(2(m-1) + 1 + (\mathsf{Right.Half} + \delta_i) - \mathsf{D}_i)$ hops. This implies that for any deadline-respecting path in the compound network, the corresponding sub-path within $\alpha_i G^\circ$ must have length $\leq \mathsf{D} - (2(m-1) + 1 + (\mathsf{Right.Half} + \delta_i) - \mathsf{D}_i)$. Since our construction satisfies (65), Observation 3 holds for all deadline-respecting paths in the compound network that use $\alpha_i G^\circ$.

Since Observations 1 to 3 hold for our new construction, the second half of Lemma 6 is proven. ■

APPENDIX F
THE FEASIBILITY OF THE GENERAL COMPOUND NETWORK CONSTRUCTION IN APPENDIX E-B

In this section, we will prove that we can always find $2m$ integer values $\{\alpha_i > 0 : i \in [0, m-1]\}$, $\{\delta_j > 0 : j \in [1, m-1]\}$, and $\mathsf{Right.Half} > 0$ satisfying (59) to (66). We notice that (64) automatically implies (59) and we will thus focus only on (60) to (66) subsequently.

To that end, we first notice that Right.Half appears only on the left-hand sides of (60) to (62), not in (64) nor in (66). Therefore, whenever we have finished choosing the $\{\alpha_i\}$ and $\{\delta_j\}$ values, we can always choose a sufficiently large Right.Half to satisfy (60) to (62).

Before describing how to choose $2m-1$ integer values $\{\alpha_i > 0 : i \in [0, m-1]\}$ and $\{\delta_j > 0 : j \in [1, m-1]\}$, we relax the problem a bit and focus on finding $2m$ integer values $\{\alpha_i > 0 : i \in [0, m-1]\}$ and $\{\delta_j : j \in [0, m-1]\}$ that satisfy (64) and (66). Namely, instead of focusing on finding δ_1 to δ_{m-1} with the value of the dummy variable δ_0 hardwired to 0, we now are allowed to choose the δ_0 value. Furthermore, previously all δ_j values have to be strictly positive. In our relaxed problem, we allow negative δ_j. We will first prove that with such a relaxation, finding those $\{\alpha_i\}$ and $\{\delta_j\}$ values is always possible.

We prove the above claim by explicit construction. We first set $\delta_{m-1} = 0$ and $\alpha_{m-1} = 1$. Then for $i = m-2$ back to 0, we set the δ_i and α_i in the following sequential way. For any given i, we choose δ_i as an integer satisfying (64). This is always possible since we have already fixed our choices of δ_{i+1} and α_{i+1} in the previous round and we allow δ_i to take a negative value. After deciding the δ_i value, we choose the α_i value as a strictly positive integer satisfying (66). Again, this is always possible since δ_i has been decided already and by our definition of Δ_H° in (44) we always have $\Delta_H^\circ \geq 1$. The above procedure is repeated for all $i = m-2$ back to 0 and we have thus found the desired $\{\alpha_i > 0 : i \in [0, m-1]\}$ and $\{\delta_j : j \in [0, m-1]\}$ satisfying (64) and (66) simultaneously.

Now we describe how to incorporate the positivity constraint on δ_j and the dummy constant constraint $\delta_0 = 0$. To that end, we first use $\delta_{j,\text{old}}$ to denote the δ_j value we found in the previous construction. Define $x \triangleq \delta_{0,\text{old}}$. Then we keep the same $\{\alpha_i > 0 : i \in [0, m-1]\}$ values of our previous construction but choose the new $\delta_{j,\text{new}}$ by $\delta_{j,\text{new}} = \delta_{j,\text{old}} - x$ for all $j \in [0, m-1]$.

It is clear that $\delta_{0,\text{new}} = 0$ satisfies the dummy constant constraint. Also, the new $\delta_{j,\text{new}}$ and the previously constructed $\{\alpha_i > 0 : i \in [0, m-1]\}$ must satisfy both (64) and (66) since we now shift all the δ_j values by the same x amount. Finally, since the new $\delta_{j,\text{new}}$ satisfy (64), they are strictly increasing with respect to j. Therefore, $\delta_{j,\text{new}} > \delta_{0,\text{new}} = 0$ for all $j \in [1, m-1]$. The positivity condition on δ_j also holds.

The above explicit construction shows that we can always find the integer values $\{\alpha_i > 0 : i \in [0, m-1]\}$, $\{\delta_j > 0 : j \in [1, m-1]\}$, and $\mathsf{Right.Half} > 0$ satisfying (60) to (66). The proof is complete.

APPENDIX G
PROOFS OF PROPOSITION 4 AND COROLLARY 1

Proof of Proposition 4: Consider any delay-constrained rate R that is feasible. By Proposition 1, for any arbitrary T value, the time expanded graph $\overline{G}^{[T+\mathsf{D}]}$ can sustain T simultaneous unicast flows from $[s,t]$ to $[d, t+\mathsf{D}]$ for all $t \in [1,T]$ with individual rate R.

Consider any arbitrary T satisfying $T > L$. We define

$$\overline{E}_{h,T} \triangleq \{([u,t],[v,t+1]) : \forall e = (u,v) \in E, \\ \forall t \in [1, T+\mathsf{D}-1] \text{ s.t. } h(e, \bmod (t, L)) = 1\} \tag{67}$$

in the time expanded graph $\overline{G}^{[T+\mathsf{D}]}$, where $h(\cdot,\cdot)$ is a function satisfying the statements in Proposition 4. The difference between $\overline{E}_{h,T}$ and the previously defined $\overline{E}_h$ in (23) is that $\overline{E}_{h,T}$ is defined for arbitrarily large T while $\overline{E}_h$ is defined for a given L.

We now prove that $\overline{E}_{h,T}$ in (67) is an edge-cut in $\overline{G}^{[T+\mathsf{D}]}$ that separates $[s,t]$ from $\{[d, \tau+\mathsf{D}] : \forall \tau \in [1,t]\}$ for all $t \in [1,T]$. Suppose not. Then there exists $t_1 \leq T$ and $t_2 \leq t_1 + \mathsf{D}$ such that there exists a path from $[s,t_1]$ to $[d,t_2]$ in $\overline{G}^{[T+\mathsf{D}]}$ without using any edge in $\overline{E}_{h,T}$. Note that by the "causality" used in the construction of $\overline{G}^{[T+\mathsf{D}]}$, this implicitly implies $t_1 < t_2$.

We now argue that if such a path exists, there exists a path from $[s,\tilde{t}_1]$ to $[d,\tilde{t}_2]$ in $\overline{G}^{[L+\mathsf{D}]}$ without using $\overline{E}_h$ where $\tilde{t}_1 \triangleq \bmod (t_1 - 1, L) + 1$ and $\tilde{t}_2 \triangleq \tilde{t}_1 + t_2 - t_1$. The reason is that whatever the path form $[s,t_1]$ to $[d,t_2]$ in $\overline{G}^{[T+\mathsf{D}]}$ is, we can transcribe it to a path from $[s,\tilde{t}_1]$ to $[d,\tilde{t}_2]$ in $\overline{G}^{[T+\mathsf{D}]}$ by shifting the time indices of the intermediate nodes by $(t_1 - \tilde{t}_1)$. The time-shifted new path does not use any edge in $\overline{E}_{h,T}$ since the shift amount $t_1 - \tilde{t}_1$ is a multiple of L and $\overline{E}_{h,T}$ includes edges "periodically with period L." See (67). Furthermore, such a path is not only in $\overline{G}^{[T+\mathsf{D}]}$ but also in $\overline{G}^{[L+\mathsf{D}]}$ since by our construction $\tilde{t}_1 \leq L$ and $\tilde{t}_2 \leq L + \mathsf{D}$. By noting that $\overline{E}_{h,T}$ in (67) is a superset of $\overline{E}_h$ in (23), the time-shifted new path does not use $\overline{E}_h$, either.

However, the existence of such a path contradicts the statement in Proposition 4 that $\overline{E}_h$ forms an edge-cut separating $[s,t]$ from $[d,t+\mathsf{D}]$ for all $t \in [1,L]$ since one can now traverse from $[s,\tilde{t}_1]$ to $[d,\tilde{t}_2]$ and then traverse through $[d,\tilde{t}_2] \to [d,\tilde{t}_2+1] \to \cdots \to [d,\tilde{t}_1+\mathsf{D}]$ without using $\overline{E}_h$. This contradiction proves that $\overline{E}_{h,T}$ in (67) is an edge-cut in $\overline{G}^{[T+\mathsf{D}]}$ that separates $[s,t]$ from $\{[d,\tau+\mathsf{D}] : \forall \tau \in [1,t]\}$ for all $t \in [1,T]$.

Since $\overline{E}_{h,T}$ is an edge cut separating $[s,t]$ from $\{[d,\tau+\mathsf{D}] : \forall \tau \in [1,t]\}$ for all $t \in [1,T]$, by the *generalized network-sharing bound* in [21], we have

$$T \cdot R \leq \sum_{\overline{e} \in \overline{E}_{h,T}} c_{\overline{e}} \tag{68}$$

$$\leq \left\lceil \frac{T+\mathsf{D}-1}{L} \right\rceil \sum_e \left(\sum_{l=0}^{L-1} h(e,l) \right) c_e \tag{69}$$

where (68) follows from that the sum rate of the T coexisting flows is no larger than the generalized cut set value [21]; and (69) follows from the definition of $\overline{E}_{h,T}$ in (67) and from over-counting one extra time period of L time slots.

Ineq. (69) implies

$$R \leq \frac{L}{T}\left\lceil\frac{T+\mathsf{D}-1}{L}\right\rceil\sum_{e}\left(\frac{\sum_{l=0}^{L-1}h(e,l)}{L}\right)c_e, \quad \forall T.$$

By letting $T\to\infty$, we have proven (24). ■

Proof of Corollary 1: Specifically, consider any IP solution $\{y_e^*\}$ that leads to an upper bound in Proposition 3. By setting $L=1$ and the binary mapping to be $h(e,0)=y_e^*$ for all e, one can verify that the $\overline{E}_h$ in (23) is an edge-cut in $\overline{G}^{[1+\mathsf{D}]}$ that separates $[s,1]$ from $[d,1+\mathsf{D}]$ since $\{y_e^*\}$ corresponds to a cut in G severing all paths of length $\leq \mathsf{D}$. One can easily verify that the resulting upper bound in (24) for this particular choice of $L=1$ and $h(e,0)=y_e^*$ is identical to (17). As a result, any upper bound in Proposition 3 can be used to construct an upper bound in Proposition 4 with the same value.

Conversely, for $L=1$ and any binary mapping $h(e,0)$, if $\overline{E}_h$ in (23) is an edge cut separating $[s,1]$ from $[d,1+\mathsf{D}]$, then the choice of $y_e^*=h(e,0)$ will be a cut severing all paths of length $\leq\mathsf{D}$. As a result, any upper bound in Proposition 4 can be used to construct an upper bound in Proposition 3 with the same value. The equivalence is thus proven. ■

Appendix H
The Generalized Cut Set Verification

This section considers exclusively Fig. 3(b). To show that the $\overline{E}_h$ in (23) generated by $L=2$ and the $h(e,t)$ in (25)–(26) is an edge cut separating $[s,t]$ from $\{[d,\tau+\mathsf{D}]:\forall\tau\in[1,t]\}$ for all $t\in[1,2]$, we notice that there are only 4 types of paths that connect $[s,t]$ to $[d,t+\mathsf{D}]$, where $\mathsf{D}=6$. They are

Type 1: The path $[s,t]\to[v_1,t+1]\to[v_2,t+2]\to[v_3,t+3]\to[v_7,t+4]\to[v_8,t+5]\to[d,t+6]$. If $t=2$, then the edge $[s,t]\to[v_1,t+1]$ will be included in $\overline{E}_h$ by (25). If $t=1$, then the edge $[v_2,t+2]\to[v_3,t+3]$ will be included in $\overline{E}_h$ by (26).

Type 2: The path $[s,t]\to[v_5,t+1]\to[v_6,t+2]\to[v_2,t+3]\to[v_3,t+4]\to[v_4,t+5]\to[d,t+6]$. If $t=2$, then the edge $[v_2,t+3]\to[v_3,t+4]$ will be included in $\overline{E}_h$ by (26). If $t=1$, then the edge $[v_4,t+5]\to[d,t+6]$ will be included in $\overline{E}_h$ by (25).

Type 3: The path $[s,t]\to[v_1,t+1]\to[v_9,t+2]\to[v_{11},t+3]\to[v_{10},t+4]\to[v_4,t+5]\to[d,t+6]$. If $t=2$, then the edge $[s,t]\to[v_1,t+1]$ will be included in $\overline{E}_h$ by (25). If $t=1$, then the edge $[v_4,t+5]\to[d,t+6]$ will be included in $\overline{E}_h$ by (25).

Type 4: Type 4 corresponds to the paths of the form $sv_1v_2v_3v_4d$. Since the path $sv_1v_2v_3v_4d$ in the original graph G has length 5, it means that if we go directly from $[s,t]$ to $[d,t+6]$ in the time expanded graph $\overline{G}^{[L+\mathsf{D}]}$, the packet will arrive in 5 time slots rather than 6. Therefore, we can "wait" and stay idle in one of the six nodes $\{s,v_1,v_2,v_3,v_4,d\}$. For example, if we stay in v_1, then the corresponding path in $\overline{G}^{[L+\mathsf{D}]}$ becomes $[s,t]\to[v_1,t+1]\to[v_1,t+2]\to[v_2,t+3]\to[v_3,t+4]\to[v_4,t+5]\to[d,t+6]$. If we stay in d, then the corresponding path in $\overline{G}^{[L+\mathsf{D}]}$ becomes $[s,t]\to[v_1,t+1]\to[v_2,t+2]\to[v_3,t+3]\to[v_4,t+4]\to[d,t+5]\to[d,t+6]$. Totally there are 6 paths of type-4, each corresponding to staying in one of the six nodes $\{s,v_1,v_2,v_3,v_4,d\}$, respectively.

We now consider two cases. Case 1: If we stay idle in one of $\{v_3,v_4,d\}$, then the first 3 edges of the paths must be $[s,t]\to[v_1,t+1]\to[v_2,t+2]\to[v_3,t+3]$. If $t=2$, then the edge $[s,t]\to[v_1,t+1]$ will be included in $\overline{E}_h$ by (25). If $t=1$, then the edge $[v_2,t+2]\to[v_3,t+3]$ will be included in $\overline{E}_h$ by (26).

Case 2: If we stay idle in one of $\{s,v_1,v_2\}$, then the last 3 edges of the paths must be $[v_2,t+3]\to[v_3,t+4]\to[v_4,t+5]\to[d,t+6]$. If $t=2$, then the edge $[v_2,t+3]\to[v_3,t+4]$ will be included in $\overline{E}_h$ by (26). If $t=1$, then the edge $[v_4,t+5]\to[d,t+6]$ will be included in $\overline{E}_h$ by (25).

The above analysis shows that $\overline{E}_h$ separates $[s,t]$ from $[d,t+6]$ for all $t\in[1,2]$. Our proof is thus complete.

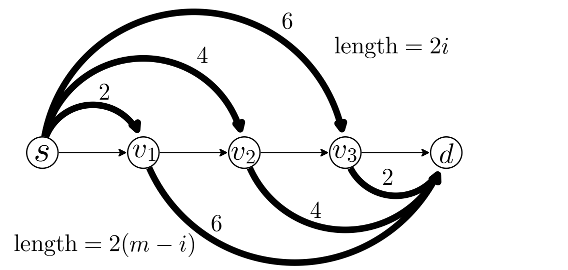


Fig. 13. An illustration of the proposed construction with $m=4$ and $\mathsf{D}=2m-1=7$.

Appendix I
A Proof of Proposition 5

For any $m\geq 2$, we construct a network instance as follows. See Fig. 13 for illustration. The network contains three major components.

Component 1: A direct path of length m connecting s to d. We denote the intermediate nodes by $v_i, \forall i\in[1,m-1]$.

Component 2: A set of $m-1$ paths connecting s and v_i for $i\in[1,m-1]$. Each path is of length $2i$. Component 2 is illustrated by the paths in the upper half of Fig. 13. For future references, the path connecting s and v_i is denoted by U_i. Namely, the i-th Upper path.

Component 3: A set of $m-1$ paths connecting v_i and d for $i\in[1,m-1]$. Each path is of length $2(m-i)$. Component 3 is illustrated by the paths in the lower half of Fig. 13. For future references, the path connecting v_i and d is denoted by L_i. Namely, the i-th Lower path.

All edges/paths are of unit capacity, i.e., $c_e=1,\forall e\in E$. The delay requirement is set to $\mathsf{D}=2m-1$.

We first show that $R^*_{\text{route}}=m$ for the above network. The reason is that the following m paths

$$sv_1L_1d$$
$$sU_iv_iv_{i+1}L_{i+1}d, \forall i\in[1,m-2]$$
$$sU_{m-1}v_{m-1}d$$

all have length $2m-1$ and are edge-disjoint. As a result, we have $R^*_{\text{route}} = m$ since the min-cut value from s to d is m.

We now prove that $R^*_{\text{RLNC}} = 1$. To that end, we first consider the *transfer function* of the network, which takes into account the local coding coefficients in (29), but omits the impact of the precoding operations in (28). More specifically, suppose at time t source s sends a coded symbol $W_1(t)$ directly to v_1 and sends coded symbols $W_{i+1}(t)$ through the path U_i for $i \in [1, m-1]$. And also suppose in the *end* of time t destination d receives $Y_m(t)$ directly from v_{m-1} and receives $Y_i(t)$ through the path L_i for $i \in [1, m-1]$. Then the transfer function between the input vector $\vec{W}(t) = (W_1(t), \cdots, W_m(t))^{\mathrm{T}}$ and the output vector $\vec{Y}(t) = (Y_1(t), \cdots, Y_m(t))^{\mathrm{T}}$ can be written as

$$\vec{Y}(t) = \sum_{\tau=1}^{t} F_{t,\tau} \cdot \vec{W}(\tau)$$

where $F_{t,\tau}$ is the transfer matrix from $\vec{W}(\tau)$ to $\vec{Y}(t)$, which is a function of the local coding coefficients of the network nodes. Since we assume that the local coding coefficients are fixed and do not change over time, $F_{t,\tau}$ is a function of the time difference $\Delta = t - \tau$. We can thus define $F_\Delta = F_{\tau+\Delta,\tau}$.

Using the above notation of the transfer matrix F_Δ, [14] proves the following results regarding the delay-constrained throughput.

Proposition 8 ([14, Theorem 2]): For any given set of local coding coefficients, we consider its transfer matrix F_Δ. For any given integer $x \in [0, \infty)$, define the following matrix

$$\overline{F}_x \triangleq \begin{bmatrix} F_0 & \mathbf{0} & \cdots & \cdots & \mathbf{0} \\ F_1 & F_0 & \mathbf{0} & \cdots & \mathbf{0} \\ F_2 & F_1 & F_0 & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ F_x & F_{x-1} & F_{x-2} & \cdots & F_0 \end{bmatrix}. \tag{70}$$

Then the largest supportable rate under the delay constraint D, assuming both the precoding vectors (how we generate $\vec{W}(t)$) and the decoding operations at d are chosen in the optimal way, is characterized by

$$R^* = \mathrm{Rank}(\overline{F}_{\mathsf{D}-1}) - \mathrm{Rank}(\overline{F}_{\mathsf{D}-2}). \tag{71}$$

We will use Proposition 8 to prove that $R^*_{\text{RLNC}} = 1$ in Fig. 13. To that end, we first assume that all the interior nodes of paths U_i and L_i only perform pure relaying. Such an assumption is without loss of generality since with a sufficiently large finite field $\mathsf{GF}(q)$, the RLNC performed by any node u with $|\mathrm{In}(u)| = 1$ is equivalent to "relaying" with close-to-one probability. And we denote the local coding coefficients of node v_i, $i \in [1, m-1]$, by

$$\beta^{[i]}_{H,H},\ \beta^{[i]}_{U,H},\ \beta^{[i]}_{H,L},\ \text{and}\ \beta^{[i]}_{U,L}. \tag{72}$$

Namely, $\beta^{[i]}_{H,H}$ is the coding coefficient used by v_i when going from the Horizontal input edge to the Horizontal output edge of v_i; $\beta^{[i]}_{U,H}$ is the coding coefficient used by v_i when going from the Upper path U_i to the Horizontal output edge; $\beta^{[i]}_{H,L}$ is the coding coefficient used by v_i when going from the Horizontal input edge to the Lower path L_i; and $\beta^{[i]}_{U,L}$ is the coding coefficient used by v_i when going from the Upper path U_i to the Lower path L_i.

Using the local coding coefficients in (72), we will characterize the expressions of F_Δ in Fig. 13. We first notice that since the shortest path from s to d is of length m, we have $F_\Delta = \mathbf{0}$ if $\Delta \le m-2$. We now characterize F_Δ when $\Delta \in [m-1, \mathsf{D}-1] = [m-1, 2m-2]$. Specifically, we can write down the transfer matrix F_Δ by

$$F_\Delta \triangleq \left(f^{(\Delta)}_{i,j}\right)$$

where $f^{(\Delta)}_{i,j}$ denotes the entry at the intersection of the i-th row and the j-th column. By tracing the impact of each local coding coefficients in (72), we have that for any $i, j \in [1, m]$ and any $\Delta \in [m-1, 2m-2]$,

$$f^{(\Delta)}_{i,j} = \begin{cases} \beta^{[j-1]}_{U,H} \left(\prod_{l=j}^{i-1} \beta^{[l]}_{H,H}\right) \beta^{[i]}_{H,L} & \text{if } i-j = 2m-2-\Delta \\ 0 & \text{otherwise} \end{cases}. \tag{73}$$

Here we use the convention that $\beta^{[0]}_{U,H} = 1 = \beta^{[m]}_{H,L}$ since the coding coefficients in (72) are defined only for $i \in [1, m-1]$.

Define

$$\forall j \in [1, m-1],\ \gamma^{[j]} \triangleq \frac{\beta^{[j]}_{U,H}}{\beta^{[j-1]}_{U,H} \beta^{[j]}_{H,H}} \text{ and}$$

$$\Gamma \triangleq \begin{bmatrix} \gamma^{[1]} & 0 & \cdots & 0 \\ 0 & \gamma^{[2]} & \cdots & 0 \\ \vdots & \vdots & \vdots & 0 \\ 0 & 0 & 0 & \gamma^{[m-1]} \end{bmatrix}.$$

Since all the coding coefficients in (72) are chosen randomly, $\gamma^{[i]}$ and Γ are well-defined with close-to-one probability when the underlying finite field $\mathsf{GF}(q)$ is sufficiently large.

By (73), one can easily verify that

$$\begin{bmatrix} F_{m-1} \\ F_m \\ F_{m+1} \\ \vdots \\ F_{2m-2} \end{bmatrix} \begin{bmatrix} \mathbf{0}_{1\times(m-1)} \\ \mathbf{I}_{(m-1)\times(m-1)} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ F_{m-1} \\ F_m \\ \vdots \\ F_{2m-3} \end{bmatrix} \begin{bmatrix} \Gamma \\ \mathbf{0}_{1\times(m-1)} \end{bmatrix} \tag{74}$$

where $\mathbf{0}_{1\times(m-1)}$ is a zero row vector of dimension $m-1$ and $\mathbf{I}_{(m-1)\times(m-1)}$ is the $(m-1)$-by-$(m-1)$ identity matrix.

If we remove the first column of $\overline{F}_{\mathsf{D}-1}$ and denote the remaining $(m\mathsf{D})\times(m\mathsf{D}-1)$ matrix as $\overline{F}'_{\mathsf{D}-1}$, we will then have $\mathrm{Rank}(\overline{F}'_{\mathsf{D}-1}) = \mathrm{Rank}(\overline{F}_{\mathsf{D}-2})$ since (i) $F_\Delta = \mathbf{0}$ if $\Delta \le m-2$; (ii) by (70) and (74) one can see every column of $\overline{F}'_{\mathsf{D}-1}$ can be written as a linear combination of the columns in $\overline{F}''_{\mathsf{D}-2}$ where

$$\overline{F}''_{\mathsf{D}-2} \triangleq \begin{bmatrix} \mathbf{0}_{m\times(m(\mathsf{D}-1))} \\ \overline{F}_{\mathsf{D}-2} \end{bmatrix};$$

and (iii) we also have $\mathrm{Rank}(\overline{F}''_{\mathsf{D}-2}) = \mathrm{Rank}(\overline{F}_{\mathsf{D}-2})$. The fact that $\mathrm{Rank}(\overline{F}'_{\mathsf{D}-1}) = \mathrm{Rank}(\overline{F}_{\mathsf{D}-2})$ then implies

$$\mathrm{Rank}(\overline{F}_{\mathsf{D}-1}) - \mathrm{Rank}(\overline{F}_{\mathsf{D}-2}) \le 1$$

whenever Γ is well-defined, which is of close-to-one probability.

Finally, since the $(m,1)$-th entry of F_{m-1} being $\prod_{i=1}^{m-1} \beta_{H,H}^{[i]}$ is non-zero with close-to-one probability, we have $\text{Rank}(\overline{F}_{\mathsf{D}-1}) - \text{Rank}(\overline{F}_{\mathsf{D}-2}) = 1$ with close-to-one probability. By Proposition 8, the largest supportable $R^*_{\text{RLNC}} = 1$ even with the optimal precoder and decoder designs.

APPENDIX J
FIG. 4(A) HAS $R^*_{\text{RLNC}} = 2$

In this section we show that with high probability, RLNC is able to support an integer rate $R = 2$ in Fig. 4(a). The proof that RLNC is not able to support $R = 3$ is similar to our discussion in Section II-C and is thus omitted.

For the ease of exposition, we assume that the message symbols for each time t are $X(t)$ and $Y(t)$. Since the symbols are precoded in (28), we assume

$$\begin{aligned} M_{sv_2}^{(t)} &= a_1X(t) + b_1Y(t) \\ M_{sv_1}^{(t)} &= a_2X(t) + b_2Y(t) \\ M_{sv_4}^{(t)} &= a_3X(t) + b_3Y(t) \end{aligned}$$

where the coefficients a_i and b_i are randomly chosen. Assuming nodes $\{v_1, v_4, v_5, v_3, v_6, v_7\}$ only perform pure relaying, we will have

$$\begin{aligned} M_{v_2d}^{(t)} =& \beta_{s,d}(a_1X(t-1) + b_1Y(t-1)) \\ &+ \beta_{v_1,d}(a_2X(t-2) + b_2Y(t-2)) \\ &+ \beta_{v_5,d}(a_3X(t-3) + b_3Y(t-3)) \end{aligned} \tag{75}$$

where for any $u_1 \in \text{In}(v_2)$ and $u_2 \in \text{Out}(v_2)$, the scalar β_{u_1,u_2} is shorthand for the local coding coefficient from edge (u_1, v_2) to edge (v_2, u_2) at node v_2. Similarly, we have

$$\begin{aligned} M_{v_3d}^{(t)} =& \beta_{s,v_3}(a_1X(t-2) + b_1Y(t-2)) \\ &+ \beta_{v_1,v_3}(a_2X(t-3) + b_2Y(t-3)) \\ &+ \beta_{v_5,v_3}(a_3X(t-4) + b_3Y(t-4)) \end{aligned} \tag{76}$$

$$\begin{aligned} M_{v_7d}^{(t)} =& \beta_{s,v_6}(a_1X(t-3) + b_1Y(t-3)) \\ &+ \beta_{v_1,v_6}(a_2X(t-4) + b_2Y(t-4)) \\ &+ \beta_{v_5,v_6}(a_3X(t-5) + b_3Y(t-5)). \end{aligned} \tag{77}$$

Recall that d would like to decode $X(t-3)$ and $Y(t-3)$ by the end of time t since $\mathsf{D} = 4$. By delaying the $M_{v_2d}^{(t)}$ coded symbol in (75), destination d can compute

$$\begin{aligned} & M_{v_3d}^{(t)} - \frac{\beta_{s,v_3}}{\beta_{s,d}} M_{v_2d}^{(t-1)} \\ &= \left(\beta_{v_1,v_3} - \frac{\beta_{s,v_3}}{\beta_{s,d}}\beta_{v_1,d}\right)(a_2X(t-3) + b_2Y(t-3)) \\ &\quad + \left(\beta_{v_5,v_3} - \frac{\beta_{s,v_3}}{\beta_{s,d}}\beta_{v_5,d}\right)(a_3X(t-4) + b_2Y(t-4)) \end{aligned} \tag{78}$$

We now notice that in (77) and (78) the terms corresponding to $(X(t-4), Y(t-4))$ and $(X(t-5), Y(t-5))$ have already been decoded in the past, and they can thus be removed. Destination d can then decode $(X(t-3), Y(t-3))$, with close-to-one probability, from the remaining terms of (77) and (78), which are

$$\begin{aligned} &\beta_{s,v_6}(a_1X(t-3) + b_1Y(t-3)) \\ &\text{and } \left(\beta_{v_1,v_3} - \frac{\beta_{s,v_3}}{\beta_{s,d}}\beta_{v_1,d}\right)(a_2X(t-3) + b_2Y(t-3)). \end{aligned}$$

Rate $R = 2$ is thus supported by RLNC.

REFERENCES

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, July 2000.

[2] F. Bai, T. Elbatt, G. Hollan, H. Krishnan, and V. Sadekar, "Towards characterizing and classifying communication-based automotive applications from a wireless networking perspective," in *Proceedings of IEEE Workshop on Automotive Networking and Applications (AutoNet)*, 2006, pp. 1–25.

[3] X. Chen, M. Chen, B. Li, Y. Zhao, Y. Wu, and J. Li, "Celerity: a low-delay multi-party conferencing solution," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 9, September 2013.

[4] R. Dougherty, C. Freiling, and K. Zeger, "Networks, matroids, and non-Shannon information inequality," *IEEE Trans. Inf. Theory*, vol. 53, no. 6, pp. 1949–1969, June 2007.

[5] E. Drinea, C. Fragouli, and L. Keller, "Delay with network coding and feedback," in *Proc. IEEE Int'l Symp. Inform. Theory*. Seoul, Korea, June 2009.

[6] J. Edmonds, "Edge-disjoint branchings," in *Combinatorial Algorithms*, R. Rustin, Ed. Academic Press: New York, 1973, pp. 91–96.

[7] J. Edmonds and R. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *Journal of the ACM (JACM) JACM*, vol. 19, no. 2, pp. 248–264, April 1972.

[8] E. Erez, "Convolutional network codes," in *Proc. IEEE Int'l Symp. Inform. Theory*, 2004.

[9] E. Erez, M. Effros, and T. Ho, "Network codes with deadlines," in *Proc. 46th Annual Allerton Conf. on Comm., Contr., and Computing*. Monticello, Illinois, USA, September 2008.

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

[11] L. Ford and D. Fulkerson, "Maximal flow through a network," *Canadian Journal of Mathematics*, vol. 8, no. 3, pp. 399–404, 1956.

[12] A. Goldberg, "Recent developments in maximum flow algorithms," NEC Research Institute, Inc., Technical Report 98-045, 1998.

[13] A. Goldberg and R. Tarjan, "A new approach to the maximum-flow problem," *Journal of the ACM*, vol. 35, no. 4, pp. 921–940, October 1988.

[14] W. Guo, N. Cai, and Q. Sun, "Time-variant decoding of convolutional network codes," *IEEE Communications Letter*, vol. 16, no. 10, pp. 1656–1659, October 2012.

[15] P. Gupta and P. Kumar, "The capacity of wireless networks," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 388–404, March 2000.

[16] J. Han and C.-C. Wang, "General capacity region for the fully-connected 3-node packet erasure network," in *Proc. IEEE Int'l Symp. Inform. Theory*. Hong Kong, China, June 2015.

[17] N. Harvey, R. Kleinberg, and A. Lehman, "On the capacity of information network," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2345–2364, June 2006.

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

[19] S. Kamath, S. Kannan, P. Viswanath, and C. Chekuri, "Delay-constrained unicast and the triangle-cast problem," in *Proc. IEEE Int'l Symp. Inform. Theory*. Hong Kong, China, June 2015, pp. 804–808.

[20] S. Kamath, D. Tse, and C.-C. Wang, "Two-unicast is hard," in *Proc. IEEE Int'l Symp. Inform. Theory*. Honolulu, Hawaii, USA, 2014.

[21] S. Kamath, D. Tse, and V. Anantharam, "Generalized network sharing outer bound and the two-unicast problem," in *Proc. 7th Workshop on Network Coding, Theory, & Applications (NetCod)*. Beijing, China, July 2011, pp. 1–6.

[22] A. Khreishah, C.-C. Wang, and N. Shroff, "Optimal rate control with inter-session network coding," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 816–829, June 2010.

[23] H. Kim, Y.-K. Chia, and A. El Gamal, "A note on the broadcast channel with stale state information at the transmitter," *IEEE Trans. Inf. Theory*, vol. 61, no. 7, pp. 3622–3631, July 2015.

[24] M. Kodialam and T. Lakshman, "On allocating capacity in networks with path length constrained routing," in *Proc. 40th Annual Allerton Conf. on Comm., Contr., and Computing*, 2002.

[25] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Networking*, vol. 11, no. 5, pp. 782–795, October 2003.

[26] D. Koutsonikolas, C.-C. Wang, Y. 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," *Elsevier Performance Evaluation (PEVA)*, vol. 68, no. 11, November 2011.

[27] W.-C. Kuo and C.-C. Wang, "Two-flow capacity region of the COPE principle for two-hop wireless erasure networks," *IEEE Trans. Inf. Theory*, vol. 59, no. 11, pp. 7553–7575, November 2013.

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

[29] S.-Y. Li and Q. Sun, "Network coding theory via commutative algebra," *IEEE Trans. Inf. Theory*, vol. 57, no. 1, pp. 403–415, January 2011.

[30] S.-Y. Li and R. Yeung, "On convolutional network coding," in *Proc. IEEE Int'l Symp. Inform. Theory*. Seattle, USA, June 2006.

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

[32] X. Li, C.-C. Wang, and X. Lin, "Optimal immediately-decodable inter-session network coding (IDNC) schemes for two unicast sessions with hard deadline constraints," in *Proc. 49th Annual Allerton Conf. on Comm., Contr., and Computing*. Monticello, Illinois, USA, September 2011, pp. 784–791.

[33] ——, "Throughput and delay analysis on uncoded and coded wireless broadcast with hard deadline constraints," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 5, May 2011.

[34] C. Ngai and R. Yeung, "Network coding gain of combination networks," in *Proc. IEEE Inform. Theory Workshop*. San Antonio, Texas, October 2004, pp. 283–287.

[35] Q. Sun, S. Jaggi, and S.-Y. R. Li, "Delay invariant convolutional network codes," in *Proc. IEEE Int'l Symp. Inform. Theory*. St. Petersburg, Russia, 2011, pp. 2492–2496.

[36] C.-C. Wang, "Pruning network coding traffic by network coding — a new class of max-flow algorithms," *IEEE Trans. Inf. Theory*, vol. 56, no. 4, pp. 1909–1929, April 2010.

[37] ——, "Capacity of 1-to-K broadcast packet erasure channels with channel output feedback," *IEEE Trans. Inf. Theory*, vol. 58, no. 2, pp. 957–988, February 2012.

[38] ——, "Capacity region of two symmetric nearby erasure channels with channel state feedback," in *Proc. IEEE Inform. Theory Workshop*. Lausanne, Switzerland, September 2012.

[39] ——, "On the capacity of wireless 1-hop intersession network coding — a broadcast packet erasure channel approach," *IEEE Trans. Inf. Theory*, vol. 58, no. 2, pp. 931–956, February 2012.

[40] 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, September 2014.

[41] C.-C. Wang and N. Shroff, "Pairwise intersession network coding on directed networks," *IEEE Trans. Inf. Theory*, vol. 56, no. 8, pp. 3879–3900, August 2010.

[42] Y. Wu, M. Chiang, and S.-Y. Kung, "Distributed utility maximization for network coding based multicasting: A critical cut approach," in *Proc. 2nd Workshop on Network Coding, Theory, & Applications (NetCod)*. Boston, Massachusetts, April 2006.

[43] Y. Wu, K. Jain, and S.-Y. Kung, "A unification of network coding and tree-packing (routing) theorems," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2398–2409, June 2006.

[44] Y. Wu and S.-Y. Kung, "Distributed utility maximization for network coding based multicasting: A shortest path approach," *IEEE J. Sel. Areas Commun.*, vol. 24, pp. 1475–1488, August 2006.

[45] L. Ying, S. Shakkottai, A. Reddy, and S. Liu, "On combining shortestpath and back-pressure routing over multihop wireless networks," *IEEE/ACM Trans. Netw.*, vol. 19, no. 3, pp. 841–854, June 2011.

PLACE
PHOTO
HERE

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

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

PLACE
PHOTO
HERE

Minghua Chen (S'04, M'06, SM'13) received his B.Eng. and M.S. degrees from the Department of Electronic Engineering at Tsinghua University in 1999 and 2001, respectively. He received his Ph.D. degree from the Department of Electrical Engineering and Computer Sciences at University of California at Berkeley in 2006. He spent one year visiting Microsoft Research Redmond as a Postdoc Researcher. He joined the Department of Information Engineering, the Chinese University of Hong Kong in 2007, where he is currently an Associate Professor. He is also an Adjunct Associate Professor in Institute of Interdisciplinary Information Sciences, Tsinghua University. He received the Eli Jury award from UC Berkeley in 2007 (presented to a graduate student or recent alumnus for outstanding achievement in the area of Systems, Communications, Control, or Signal Processing) and The Chinese University of Hong Kong Young Researcher Award in 2013. He also received several best paper awards, including the IEEE ICME Best Paper Award in 2009, the IEEE Transactions on Multimedia Prize Paper Award in 2009, and the ACM Multimedia Best Paper Award in 2012. He is currently an Associate Editor of the IEEE/ACM Transactions on Networking. He serves as a TPC Co-Chair of ACM e-Energy 2016 and a General Co-Chair of ACM e-Energy 2017. His current research interests include energy systems (e.g., smart power grids and energy-efficient data centers), energy-efficient transportation system, distributed optimization, multimedia networking, wireless networking, network coding, and delay-constrained network information flow.