Rate Control with Pairwise Inter-session Network Coding

Abdallah Khreishah, Chih-Chun Wang, Ness B. Shroff

***Abstract*—In this paper we develop a distributed rate control algorithm for networks with multiple unicast-sessions when network coding is allowed across different sessions. Building on recent flow-based characterization of *pairwise inter-session network coding*, the corresponding optimal rate-control problem is formulated as a convex optimization problem. The formulation exploits pairwise coding possibilities between any pair of sessions, where any coded symbol is formed by coding over at most two original symbols. The objective function is the sum of the utilities based on the rates supported by each unicast session. Working on the Lagrangian of the formulated problem, a distributed algorithm is developed with little coordination among intermediate nodes. Each unicast session has the freedom to choose its own utility function. The only information exchange required by the source is the weighted sum of the queue length of each link, which can be piggy-backed to the acknowledgment messages. In addition to the optimal rate control algorithm, we propose a decentralized *pairwise random coding* scheme that decouples the decision of coding from that of rate-control, which further enhances the distributiveness of the proposed scheme. The convergence of the rate control algorithm is proven analytically and verified by extensive simulations. Simulation results also demonstrate the advantage of the proposed algorithm over the state-of-the-art in terms of both throughput and fairness.**

***Index Terms*—Inter-session network coding, multiple-unicast-sessions problem, rate control, distributed algorithm, capacity region, fairness.**

I. Introduction and Related Work

Over the last several years, there has been tremendous interest in the study of network optimization techniques to maximize network performance, while at the same time achieving fairness across flows (see, for example, [1]–[5]).

More recently, a new area of research has emerged called network coding that has the potential to further increase the achievable throughput by packet mixing at intermediate nodes [6]. There are two types of network coding: *inter-session network coding* where coding is permitted between the packets of different sessions and *intra-session network coding* where coding is restricted to be performed between the packets of the same session. Both intra and inter-session network coding have demonstrated significant capacity improvement. In this work we focus on inter-session network coding with multiple unicast sessions, of which a key challenge is to provide a solution that not only satisfies optimality considerations but is distributed and easy to implement. To that end, in this paper, we use inter-session network coding to develop a *distributed algorithm* that maximizes network performance and enhances fairness, under the constraint that two symbols/packets from any arbitrary pairs of the coexisting sessions can be mixed together.

In the literature, beginning with the seminal paper [6], network coding for a single multicast session (intra-session network coding) have been extensively studied. Follow-up works include [7], which shows that linear network coding is sufficient for a single multicast session. In [8], the authors develop a useful algebraic approach to network coding and [9] provides a distributed implementation based on random linear network coding. Several other related works of intra-session network coding can be found in [10]–[12].

Nonetheless, most network traffic is unicast. Intra-session network coding, although providing strict throughput improvement for multicast sessions, has no throughput gain over the non-coded solution when only unicast traffic is present. Therefore, in order to enhance the capacity and fairness for multiple unicast sessions, inter-session network coding is needed.

To see that inter-session network coding can be used to enhance fairness, consider the following classical butterfly topology, shown in Fig. 1(a). In this simple butterfly configuration, we assume that each link can sustain a throughput of at most 1 packet per second (in subsequent discussions we drop the units). Sources s_1 and s_2 want to send packets to destinations t_1 and t_2, respectively. The shaded regions in Fig. 1(b) represent the capacity regions for the butterfly network with and without network coding. The capacity regions represent all the possible rates R_1 for session (s_1, t_1) and R_2 for session (s_2, t_2) that can be supported. As shown by the capacity regions, if the objective is to achieve strict fairness (i.e., transmissions at equal rates from both sources), network coding doubles the rate of both sessions. Another example is the so-called grail topology shown in Fig. 2(a) introduced in [13]. Again, each link has a unit capacity and two unicast sessions (s_1, t_1) and (s_2, t_2) coexist in the network. For the grail topology the capacity regions for the coded and non-coded solutions are in Fig. 2(b). As is evident from Fig. 2(b), network coding provides a bigger capacity region, which enhances the fairness and improves the throughput of the network.

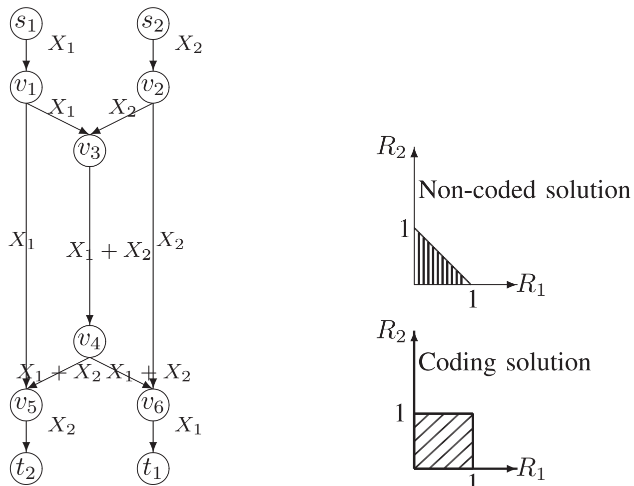
Inter-session network coding for multiple unicast/multicast sessions has been less studied. *In [14] it was shown that linear*

A. Khreishah and C.-C. Wang are with Center for Wireless Systems and Applications (CWSA), School of Electrical and Computer Engineering, Purdue University (emails: {akhreish,chihw}@purdue.edu.)

N.B. Shroff is with the Departments of Electrical and Computer Engineering and Computer Science and Engineering, The Ohio State University, (email: shroff@ece.osu.edu).

Part of this work was presented in IEEE Infocom 2008.

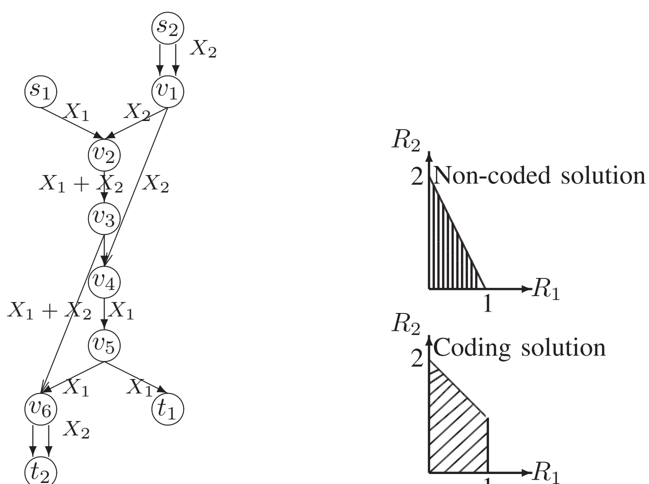
This work has been sponsored in part by NSF grants CNS 0626703, CCF 0635202, and CCF 0845968, the ARO MURI award W911NF-08-1-0238, and the AFOSR award FA 9550-07-1-0456.



(a) The full butterfly.

(b) Capacity regions.

Fig. 1. The butterfly topology and its capacity regions with and without network coding.



(a) The grail.

(b) Capacity Regions.

Fig. 2. The grail topology and its capacity regions with and without network coding.

coding is no longer capable of achieving the optimal capacity region in the multiple unicast/multicast case in contrast to the case of the single multicast session. Since then, many studies have targeted charactizing the capacity region of inter-session network coding and suboptimal solutions to the multi-session unicast problem using inter-session network coding as in [15]–[19]. For wireless networks, the nature of the links further enriches the coding possibility, as broadcasting can be achieved without power penalty. Therefore, a much smaller number of transmissions is necessary when compared to its wired counterpart. Based on the broadcast nature of wireless networks, opportunistic exclusive OR (XOR) coding is introduced in [20], and further improved and analyzed in [21].

The main subject of this work is for wireline networks, while the same principle can be applied to wireless networks in a similar way, as in [22], [23]. For wireline networks, an achievable butterfly-based capacity region was introduced in [24], which searches for any possible butterfly structure in the network. Two distributed algorithms for stabilizing the butterfly-based capacity region were provided in [25] and [26] using back-pressure techniques. In these two algorithms, the queue length information has to be exchanged among intermediate nodes. The purpose of the queue-length exchange is to determine the location of the encoding, decoding, and remedy generating nodes in [25], and for remedy packets requests in [26]. Like most back-pressure algorithms, these two algorithms focus on stabilizing the given network load instead of dynamic rate control for fairness improvement.

In this paper, our main contributions are as follows:

1) The development of a distributed algorithm with rate control and utility maximization for *inter-session* network coding for multiple unicast flows, which can be easily generalized for the case of multiple multicast flows [27]. Our results show that the utility-optimization-based rate-control algorithm, originally designed for non-coded transmissions [2]–[5] and later generalized for *intra-session network coding* [12], [28]–[32] can be extended to *pairwise inter-session network coding* for the first time. This is a non trivial generalization considering the characteristic difference between inter and intra-session network coding.
2) Our result is developed based on finding good paths rather than finding specific structures in the network (such as the butterfly structures in [24]). This enables more efficient solutions since one can leverage upon existing work on how to choose good paths through the network. Further, we show empirically that the capacity region obtained via our approach can be considerably larger than those obtained via the pattern search algorithm [24]–[26].
3) A pairwise random coding scheme is proposed, which is a modified version of the random linear coding scheme in [9]. The pairwise random coding scheme decouples the coding and rate-control decisions and facilitates the development of a fully distributed algorithm. Combining the distributed rate control and the decentralized coding scheme, we eliminate unnecessary queue length information exchange among intermediate nodes, which results in improved efficiency of the overall scheme compared to the back-pressure algorithms in [25] and [26].

The rest of the paper is organized as follows. In Section II, the graph theoretic characterization of pairwise inter-session network coding is reviewed for completeness. In Section III, we describe the system settings and the formulation of our optimization problem. In Section IV, we solve the dual problem to obtain the optimal distributed rate control algorithm for pairwise inter-session network coding. Several practical implementation issues are in Section V including the pairwise random coding scheme. In Section VI we propose two approaches to reduce the complexity of the rate control algorithm. Section VII is devoted to simulation results. We conclude the paper in Section VIII.

II. Settings and Preliminaries

A. System Settings

We model the network by a directed acyclic graph (DAG) $G = (V, E)$, where V and E are the sets of all nodes and links, respectively. We use $\mathrm{In}(v)$ to represent the set of all incoming links to node v and $\mathrm{Out}(v)$ to represent the set of all outgoing links from node v. Two types of graphs are considered depending on the corresponding edge capacity: graphs with integral edge-capacity and graphs with fractional edge-capacity. For the former type, each edge has unit capacity and carries either one or zero packet per unit time. (No fractional packets are allowed.) For the latter type, each edge e has a fractional capacity, denoted by C_e, and can transmit at any rate between 0 and C_e. An integral graph models the packet-based transmission in a network, for which a high-rate link is represented by parallel edges. On the other hand, the fractional graph can be viewed as a time-averaged version of the integral graph, which focuses on the "transmission rates" rather than the packet-by-packet behavior. For the rate-control algorithm in this paper, we use the fractional graph model. When discussing the detailed coding operations among different packets, we use the integral graph model.

For networks modelled by fractional graphs, the rate control problem is defined by the set of tuples $(s_i, t_i, U_i(R_i))$ $i \in 1, 2, \ldots, N$, where N is the number of coexisting unicast sessions. s_i and t_i are the source and destination nodes of session i and $U_i(\cdot)$ is the utility function of session i that is concave and monotonically increasing. R_i is the transmission rate supported in the i-th session.

Some graph-theoretic definitions will also be used in this work. We use $P_{v,w}$ or $Q_{v,w}$ to denote paths from nodes v to w. Here, we used two different notations to describe a path from u to v to make the characterization in Theorem 1 easier to understand. We use $\mathcal{P}$ to represent a set of paths, and the Number of Coinciding Paths at link e, $\mathsf{ncp}_{\mathcal{P}}(e)$, is defined as the number of paths in the set $\mathcal{P}$ that use link e.

B. Preliminary Results

In our previous work [13], we have studied the problem of network coding with two simple unicast sessions for integral DAGs. The main result in [13] is summarized as follows.

Theorem 1: For an integral DAG with two coexisting unicast sessions between the source-sink pairs (s_1,t_1), (s_2,t_2), there exists a linear network coding scheme that can transmit two packets (one for each session) simultaneously if and only if one of the following two conditions holds.

- [Condition 1] There exists two edge disjoint paths (EDPs) between (s_1, t_1) and (s_2, t_2), i.e a collection $\mathcal{P}$ of two paths P_{s_1,t_1} and P_{s_2,t_2}, such that $\max_{e\in E} \mathsf{ncp}_{\mathcal{P}}(e) \leq 1$.
- [Condition 2] There exist a collection $\mathcal{P}$ of three paths $\{P_{s_1,t_1}, P_{s_2,t_2}, P_{s_2,t_1}\}$ and a collection $\mathcal{Q}$ of three paths $\{Q_{s_1,t_1}, Q_{s_2,t_2}, Q_{s_1,t_2}\}$, such that $\max_{e\in E} \mathsf{ncp}_{\mathcal{P}}(e) \leq 2$ and $\max_{e\in E} \mathsf{ncp}_{\mathcal{Q}}(e) \leq 2$.

Remark: Theorem 1 focuses on the problem of mixing two packets, one from every source. It does not characterize mixing of more than one packet from each source.

If condition 1 is satisfied, the problem is feasible even by non-coded solutions. If only condition 2 is satisfied, then only network-coding-based schemes can transmit two packets simultaneously for both sessions. For example, the butterfly network in Fig. 1(a) can transmit two packets simultaneously and satisfies only condition 2 with the following choices of paths $P_{s_1,t_1} = s_1v_1v_3v_4v_6t_1$, $P_{s_2,t_2} = s_2v_2v_3v_4v_5t_2$, $P_{s_2,t_1} = s_2v_2v_6t_1$, $Q_{s_1,t_1} = s_1v_1v_3v_4v_6t_1$, $Q_{s_2,t_2} = s_2v_2v_3v_4v_5t_2$, and $Q_{s_1,t_2} = s_1v_1v_5t_2$. Another example where only condition 2 is satisfied is the grail network in Fig. 2(a) with

$$\begin{aligned}&P_{s_1,t_1} = s_1v_2v_3v_4v_5t_1, P_{s_2,t_2} = s_2v_1v_2v_3v_6t_2,\\ &P_{s_2,t_1} = s_2v_1v_4v_5t_1, Q_{s_1,t_1} = s_1v_2v_3v_4v_5t_1,\\ &Q_{s_2,t_2} = s_2v_1v_4v_5v_6t_2, Q_{s_1,t_2} = s_1v_2v_3v_6t_2. \qquad (1)\end{aligned}$$

Two packets can thus be transmitted simultaneously using intersession network coding for the grail structure. In this paper we call any collection of six paths that satisfy condition 2 of Theorem 1 a *Pairwise Inter-session network Coding Configuration* (PICC).

C. Superposition Approach

Theorem 1 serves as the building foundation of inter-session network coding over pairs of unicast sessions.

Consider N source-&-sink pairs and each source s_i would like to transmit at rate R_i packets per unit time to the corresponding sink t_i over a fractional DAG. The rate vector $(R_1, \cdots, R_N)$ is feasible if the original graph G can be viewed as the superposition of one graph G' and many PICCs such that (i) non-coded transmission is performed for every (s_i, t_i) pair in G', (ii) pairwise linear network coding across (s_i, t_i) and (s_j, t_j), $i \neq j$ is performed in each PICC individually, and (iii) the transmission rates $(R_1, \cdots, R_N)$ can be supported. Here, R_i is the sum of the non-coding transmission rate between s_i and t_i through G' and all the rates supported in any PICC where inter-session network coding is performed between session i and some other session j.

Based on this superposition principle the above construction describes the achievable rate region of *Pairwise Inter-session Network Coding* (PINC). In the next section we will describe the corresponding PINC achievable rate region by a set of constraints.

III. Problem Formulation

Since in the PINC region, the rate R_i is expressed as the sum of rates with/without inter-session network coding, two sets of parameters and variables will be used in our formulation. Some parameters and variables are for the non-coded transmission and the others capture the inter-session network coding performed on the PICCs. For the non-coded transmission, we define the parameters $\mathcal{P}_i$ and $H_i^k(e)$, and the variable x_i^k. Let $\mathcal{P}_i$ represent the collection of all paths between s_i and t_i. If link e is used by the k-th path between s_i and t_i, where k ranges from 1 to $|\mathcal{P}_i|$, then the indicator function $H_i^k(e) = 1$. Otherwise it is set to zero. x_i^k represents the uncoded transmission rate supported through the k-th path between s_i and t_i in G'. For the coded transmission through the PICCs, we define the parameters $\mathcal{P}(i,j)$, and $\mathcal{E}^p_{ij}(e)$, and the variable x^{pm}_{ij}. $\mathcal{P}(i,j)$ is the set of all tuples containing all possible choices of paths $\{P_{s_i,t_i}, P_{s_j,t_j}, P_{s_j,t_i}\}$. Because each PICC contains two sets of 3 paths, therefore, any PICC between sessions i and j can be indexed by p and m jointly, where the p and m means that the p-th tuple in $\mathcal{P}(i,j)$ and the m-th tuple in $\mathcal{P}(j,i)$ are used to generate the PICC of interest. The rate supported for sessions i and j over that PICC is denoted by x^{pm}_{ij}. We also define $\overrightarrow{x}$ as a column vector containing x^k_i, $\forall i,k$ and $x^{pm}_{ij}, \forall i,j,p,m$. Therefore, the total supported rate for session i becomes $R_i = \sum_{k=1}^{|\mathcal{P}_i|} x^k_i + \sum_{j:j\neq i}\sum_{p=1}^{|\mathcal{P}(i,j)|}\sum_{m=1}^{|\mathcal{P}(j,i)|} x^{pm}_{ij}$.

Consider a specific link e. The capacity consumed by pure routing traffic is: $\sum_{i=1}^{N}\sum_{k=1}^{|\mathcal{P}_i|} H^k_i(e)x^k_i$. For the PICC between sessions i and j, indexed by p and m, the capacity consumed by the path selection $\mathcal{P}$ is $\mathcal{E}^p_{ij}(e)x^{pm}_{ij}$, where $\mathcal{E}^p_{ij}(e)$ is defined in the following manner:

$$\mathcal{E}^p_{ij}(e) = \begin{cases} 0 & \text{if no path in the } p\text{-th tuple in } \mathcal{P}(i,j) \\ & \text{uses link } e \\ 1 & \text{if 1 or 2 paths in the } p\text{-th tuple in} \\ & \mathcal{P}(i,j) \text{ use link } e \\ 2 & \text{if 3 paths in the } p\text{-th tuple in } \mathcal{P}(i,j) \\ & \text{use link } e. \end{cases}$$

This is because by Theorem 1, successful pairwise network coding requires that $\mathsf{ncp}_{\mathcal{P}}(e) \leq 2$. If all three paths in $\mathcal{P}$ use link e, then the traffic along these three paths must use two parallel edges instead of a single one. Otherwise, $\mathsf{ncp}_{\mathcal{P}}(e) = 3$, which violates the necessary condition for pairwise inter-session network coding. The same argument holds for the traffic along the paths in $\mathcal{Q}$, the m-th tuple in $\mathcal{P}(j,i)$, for which the network coded traffic consumes $\mathcal{E}^m_{ji}(e)x^{pm}_{ij}$. From the above reasoning, the total capacity consumed by inter-session coding for the PICC between sessions i and j, indexed by p and m is the maximum of the two which is formally expressed as $\max(\mathcal{E}^p_{ij}(e), \mathcal{E}^m_{ji}(e))x^{pm}_{ij}$. Summing over all pairs of sessions $i \neq j$, and all p-th and m-th tuples of $\mathcal{P}(i,j)$ and $\mathcal{P}(j,i)$, the total capacity consumed by inter-session network coding becomes $\sum_{(i,j):i<j}\sum_{p=1}^{|\mathcal{P}(i,j)|}\sum_{m=1}^{|\mathcal{P}(j,i)|}\max(\mathcal{E}^p_{ij}(e), \mathcal{E}^m_{ji}(e))x^{pm}_{ij}$.

Let $\mathcal{PICC}_{ij}$ represent the collection of all PICCs between sessions i and j. For simplicity we use x^l_{ij} instead of x^{pm}_{ij}, where l is the index indicating that the l-th PICC of $\mathcal{PICC}_{ij}$ is used. Since any union of the p-tuple and the m-th tuple of $\mathcal{P}(i,j)$ and $\mathcal{P}(j,i)$ can be mapped to the l-th PICC between i and j. We can also define

$$H^l_{ij}(e) = \frac{1}{2}\max(\mathcal{E}^p_{ij}(e), \mathcal{E}^m_{ji}(e)). \tag{2}$$

From the above discussion, the following constraints represent the PINC capacity region.

$$\sum_{i=1}^{N}\sum_{k=1}^{|\mathcal{P}_i|} H^k_i(e)x^k_i + 2\sum_{(i,j):i<j}\sum_{l=1}^{|\mathcal{PICC}_{ij}|} H^l_{ij}(e)x^l_{ij} \leq C_e, \quad \forall e \in E \tag{3}$$

$$x^l_{ij} = x^l_{ji}, \quad \forall i<j, l. \tag{4}$$

Thus, our optimization problem becomes:

$$\max_{\overrightarrow{x}\geq 0}\sum_{i=1}^{N} U_i\Big(\sum_{k=1}^{|\mathcal{P}_i|} x^k_i + \sum_{j:i\neq j}\sum_{l=1}^{|\mathcal{PICC}_{ij}|} x^l_{ij}\Big) \tag{5}$$

subject to $\overrightarrow{x}$ satisfying (3) and (4).

By change of variable indices i and j we have

$$\sum_{(i,j):i<j}\sum_{l=1}^{|\mathcal{PICC}_{ij}|} H^l_{ij}(e)x^l_{ij} = \sum_{(i,j):j<i}\sum_{l=1}^{|\mathcal{PICC}_{ij}|} H^l_{ji}(e)x^l_{ji}.$$

Since $x^l_{ij} = x^l_{ji}$ according to (4), the constraints in (3) can be rewritten as:

$$\sum_{i=1}^{N}\sum_{k=1}^{|\mathcal{P}_i|} H^k_i(e)x^k_i + \sum_{(i,j):i\neq j}\sum_{l=1}^{|\mathcal{PICC}_{ij}|} H^l_{ij}(e)x^l_{ij} \leq C_e, \quad \forall e \in E \tag{6}$$

For the following we focus on the rate control problem satisfying constraints (4) and (6) with the objective function being (5).

IV. The Rate Control Algorithm

Note that even if every utility function $U_i(\cdot)$ is strictly concave, the objective function in (5) may not be strictly concave due to the presence of the linear terms $\sum_{k=1}^{|\mathcal{P}_i|} x^k_i + \sum_{j:i\neq j}\sum_{l=1}^{|\mathcal{PICC}_{ij}|} x^l_{ij}$. Thus, a direct application of standard convex optimization techniques might lead to multiple solutions, for which the output of an iterative method may oscillate. However, we can apply the *"proximal method"* described in [33] page 233 to ensure convergence. The idea behind the proximal method is to solve a series of problems, each of which has a strictly concave objective function. The limit of the series approaches a single solution of the original problem. A detailed description of the proximal method is in [33]. To implement the proximal method, we now introduce auxiliary variables $\overrightarrow{y} = \{y^k_i, y^l_{ij}\}$ with the same size of $\overrightarrow{x}$. The intermediate optimization problem of the proximal method becomes:

$$\max_{\overrightarrow{x}\geq 0}\sum_{i=1}^{N} U_i\Big(\sum_{k=1}^{|\mathcal{P}_i|} x^k_i + \sum_{j:i\neq j}\sum_{l=1}^{|\mathcal{PICC}_{ij}|} x^l_{ij}\Big) - \sum_{i=1}^{N}\sum_{k=1}^{|\mathcal{P}_i|}\frac{\gamma_i}{2}(x^k_i - y^k_i)^2 - \sum_{(i,j):i\neq j}\sum_{l=1}^{|\mathcal{PICC}_{ij}|}\frac{\gamma_i}{2}(x^l_{ij} - y^l_{ij})^2 \tag{7}$$

subject to $\overrightarrow{x}$ satisfying (4) and (6), where γ_i is a positive constant.

In the following, we focus on the dual of the intermediate maximization problem. Since the Slater condition holds (see for reference [34]), there is no duality gap between the primal and the dual problems. Hence, we can use the dual approach to solve the problem.

Associate Lagrange multiplier λ_e with each link e, and μ^l_{ij} with the l-th PICC between sessions i and j. Also, let $\overrightarrow{\lambda}$ and $\overrightarrow{\mu}$ be two column vectors with elements λ_e and

μ_{ij}^l, respectively. The Lagrange function of the above primal intermediate problem is:

$$L(\overrightarrow{x}, \overrightarrow{\lambda}, \overrightarrow{\mu}, \overrightarrow{y}) = \sum_{i=1}^{N} U_i(\sum_{k=1}^{|\mathcal{P}_i|} x_i^k + \sum_{j:i\neq j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} x_{ij}^l)$$
$$- \sum_{i=1}^{N}\sum_{k=1}^{|\mathcal{P}_i|} \frac{\gamma_i}{2}(x_i^k - y_i^k)^2 - \sum_{(i,j):i\neq j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} \frac{\gamma_i}{2}(x_{ij}^l - y_{ij}^l)^2$$
$$- \sum_e \lambda_e \left\{ \sum_{i=1}^{N}\sum_{k=1}^{|\mathcal{P}_i|} H_i^k(e)x_i^k + \sum_{(i,j):i\neq j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} H_{ij}^l(e)x_{ij}^l \right\}$$
$$+ \sum_e \lambda_e C_e - \sum_{(i,j):i<j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} \mu_{ij}^l x_{ij}^l + \sum_{(i,j):i<j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} \mu_{ij}^l x_{ji}^l$$

Since $\sum_{(i,j):i<j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} \mu_{ij}^l x_{ji}^l = \sum_{(i,j):j<i} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} \mu_{ji}^l x_{ij}^l$, by a simple change of variables the Lagrange function is *separable* and we can rewrite it as:

$$L(\overrightarrow{x}, \overrightarrow{\lambda}, \overrightarrow{\mu}, \overrightarrow{y}) \;=\; \sum_{i=1}^{N} B_i(\overrightarrow{x}, \overrightarrow{\lambda}, \overrightarrow{\mu}, \overrightarrow{y}) + \sum_e \lambda_e C_e.$$

Here,

$$B_i(\overrightarrow{x}, \overrightarrow{\lambda}, \overrightarrow{\mu}, \overrightarrow{y}) = U_i(\sum_{k=1}^{|\mathcal{P}_i|} x_i^k + \sum_{j:i\neq j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} x_{ij}^l)$$
$$- \sum_{k=1}^{|\mathcal{P}_i|} \frac{\gamma_i}{2}(x_i^k - y_i^k)^2 - \sum_{j:i\neq j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} \frac{\gamma_i}{2}(x_{ij}^l - y_{ij}^l)^2$$
$$- \sum_{k=1}^{|\mathcal{P}_i|} \left(\sum_e H_i^k(e)\lambda_e\right) x_i^k - \sum_{j:i\neq j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} \left(\sum_e H_{ij}^l(e)\lambda_e\right) x_{ij}^l$$
$$- \sum_{j:i<j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} \mu_{ij}^l x_{ij}^l + \sum_{j:i>j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} \mu_{ji}^l x_{ij}^l.$$

The objective function of the dual problem is

$$D(\overrightarrow{\lambda}, \overrightarrow{\mu}, \overrightarrow{y}) = \max_{\overrightarrow{x}\geq 0} L(\overrightarrow{x}, \overrightarrow{\lambda}, \overrightarrow{\mu}, \overrightarrow{y}),$$

and the dual problem is:

$$\min_{\overrightarrow{\lambda}\geq 0, \overrightarrow{\mu}} D(\overrightarrow{\lambda}, \overrightarrow{\mu}, \overrightarrow{y}).$$

The dual optimization problem can be solved using the gradient method.

Based on the above discussion we have the following distributed rate control algorithm (Algorithm $\mathcal{A}$).

Algorithm $\mathcal{A}$:

- Initialization phase: Find all paths between all sources and destinations. This can be done using any routing protocol that finds multiple paths in a distributed way as in [35], [36]. After this, sources send control messages to every link e to set the values of $H_i^k(e)$ and $H_{ij}^l(e)$. Each link sets its corresponding $\lambda_e(0)$ to zero, each destination t_i sets its corresponding $\mu_{ij}^l(0)$ to zero, and each source s_i chooses the values of $y_i^k(0)$, $y_{ij}^l(0)$, $x_i^k(0)$ and $x_{ij}^l(0)$ arbitrarily.
- Iteration phase: At the τ-th iteration:
  1) Fix $\overrightarrow{\lambda}(\tau,0) = \overrightarrow{\lambda}(\tau)$, $\overrightarrow{\mu}(\tau,0) = \overrightarrow{\mu}(\tau)$, and $\overrightarrow{x}(\tau,0) = \overrightarrow{x}(\tau)$.
  2) perform the following steps sequentially for $\kappa = 0,\ldots,K-1$.
     – Update the dual variables at each link e by:
     $$\lambda_e(\tau,\kappa+1) = \left[\lambda_e(\tau,\kappa) + \alpha_e\left(\sum_{i=1}^{N}\sum_{k=1}^{|\mathcal{P}_i|} H_i^k(e)x_i^k(\tau,\kappa) + \sum_{(i,j):i\neq j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} H_{ij}^l(e)x_{ij}^l(\tau,\kappa) - C_e\right)\right]^+. \quad (8)$$
     Here, $[.]^+$ is a projection on $[0,\infty)$ and α_e is a positive step size. Also, $(\sum_{i=1}^{N}\sum_{k=1}^{|\mathcal{P}_i|} H_i^k(e)x_i^k(\tau,\kappa) + \sum_{(i,j):i\neq j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} H_{ij}^l(e)x_{ij}^l(\tau,\kappa) - C_e)$ is the queue length change at link e during the time from the κ-th to the $(\kappa+1)$-th step.
     – Set
     $$\mu_{ij}^l(\tau,\kappa+1) = \mu_{ij}^l(\tau,\kappa) + \alpha_{ij}^l(x_{ij}^l(\tau,\kappa) - x_{ji}^l(\tau,\kappa)), \qquad \forall i<j. \quad (9)$$
     This can be implemented at each destination t_i, where α_{ij}^l is a positive step size.
     – Let $\overrightarrow{x}(\tau,\kappa+1) = \arg\max_{\overrightarrow{x}\geq 0} L(\overrightarrow{x}, \overrightarrow{\lambda}(\tau,\kappa+1), \overrightarrow{\mu}(\tau,\kappa+1), \overrightarrow{y}(\tau))$. This can be computed in a distributed way at each source since the L function is separable. It is worth noting that computing $\overrightarrow{x}(\tau,\kappa+1)$ needs the values of (i) $\sum_e H_{ij}^l(e)\lambda_e(\tau,\kappa+1), \;\; \forall i<j,l,m$, which can be computed along the paths, (ii) $\mu_{ij}^l(\tau,\kappa+1)$, $\forall i<j,l$, and (iii)$\mu_{ji}^l(\tau,\kappa+1)$, $\forall i>j,l$. All of this information can be sent back to the source using an acknowledgment message as will be explained in Section V-A.
  3) Let $\overrightarrow{\lambda}(\tau+1) = \overrightarrow{\lambda}(\tau,K)$ and $\overrightarrow{\mu}(\tau+1) = \overrightarrow{\mu}(\tau,K)$. Set
  $$\overrightarrow{y}(\tau+1) = \overrightarrow{x}(\tau,K)$$
  and
  $$\overrightarrow{x}(\tau+1) = \overrightarrow{x}(\tau,K).$$

For sufficiently large K and sufficiently large number of iterations, $\overrightarrow{x}(\tau)$ converges to the optimizing $\overrightarrow{x}^*$ for the original problem with the objective function in (5) and the constraints (4) and (6).

Theorem 2: As $K \longrightarrow \infty$, with the step sizes $(\alpha_e, \alpha_{ij}^l)$ satisfying the following:
$(\mathcal{L}\cdot\max_e \alpha_e + 2\max_{\{i,j,l\}} \alpha_{ij}^l) < 2\min_i(\gamma_i)$, where

$$\mathcal{L} = \sum_e \Big(\sum_{i=1}^{N}\sum_{k=1}^{|\mathcal{P}_i|} H_i^k(e) + \sum_{(i,j):i\neq j} \sum_{l=1}^{|\mathcal{PICC}_{ij}|} (H_{ij}^l(e))^2\Big),$$

Algorithm $\mathcal{A}$ converges to the optimal solution of (5) subject to the constraints (4) and (6).

The proof is provided in Appendix D. For the case when K is bounded away from infinity, the convergence of Algorithm $\mathcal{A}$ is verified by simulations. Similar proofs to those in [5] can be used to rigorously prove the convergence of Algorithm $\mathcal{A}$ with fixed K and with noisy and delayed measurements. This makes Algorithm $\mathcal{A}$ suitable for practical implementation.

V. Implementation Details

In this section, we discuss several practical issues that may impact the implementation of our algorithm. In Section V-A we show how to collect the implicit costs needed for Algorithm $\mathcal{A}$. The pairwise random coding scheme is introduced in Section V-B, followed by a discussion of the transient behavior before Algorithm $\mathcal{A}$ converges in Section V-C. A brief discussion of how to deal with non-concave objective functions for real-time traffic is in Section V-D.

A. Collecting implicit costs

Each source s_i needs to collect $\sum_e \lambda_e H_{ij}^l(e)$, $\forall j \neq i$, $l \in \{1, \dots, |\mathcal{PICC}_{ij}^l|\}$ in order to compute the update rate x_{ij}^l. To do so, special control messages $\mathcal{S}_{ij}^l(u,e)$ and $\mathcal{S}_{ij}^l(e,v)$ are used. $\mathcal{S}_{ij}^l(u,e)$ is the control message sent from node u to link e to collect $\sum_e \lambda_e H_{ij}^l(e)$. Similarly, $\mathcal{S}_{ij}^l(e,v)$ is the control message sent from link e to node v to collect $\sum_e \lambda_e H_{ij}^l(e)$. More explicitly, collecting $\sum_e \lambda_e H_{ij}^l(e)$, $\forall j \neq i$ is done according to the following.

- Each source s_i sets $\mathcal{S}_{ij}^l(s_i,e) = 0$ to all of it's outgoing links e that satisfy $H_{ij}^l(e) \neq 0$.
- Assuming $e = (u,v)$, then at link e, $\mathcal{S}_{ij}^l(e,v) = \mathcal{S}_{ij}^l(u,e) + \lambda_e H_{ij}^l(e)$.
- At every intermediate node v, let $\text{In}_{ij}^l(v)$ be the set of incoming links to node v such that $H_{ij}^l(e) \neq 0$, and $\text{Out}_{ij}^l(v)$ be the set of outgoing links from node v such that $H_{ij}^l(e) \neq 0$. Then node v arbitrarily chooses one $e_v \in \text{Out}_{ij}^l(v)$ and sets $\mathcal{S}_{ij}^l(v,e_v) = \sum_{e \in \text{In}_{ij}^l(v)} \mathcal{S}_{ij}^l(e,v)$ and $\mathcal{S}_{ij}^l(v,e) = 0$ for all links $e \in \text{Out}_{ij}^l(v) \backslash e_v$.

The third step avoids overcounting the implicit costs. In the end, $\sum_{e \in \text{In}_{ij}^l(t_i)} \mathcal{S}_{ij}^l(e,t_i) + \sum_{e \in \text{In}_{ij}^l(t_j)} \mathcal{S}_{ij}^l(e,t_j) = \sum_e \lambda_e H_{ij}^l(e)$. The first term of the left hand side can be obtained at t_i while the second can be obtained at t_j. Both of them can be sent back using the acknowledge messages and s_i can obtain $\sum_e \lambda_e H_{ij}^l(e)$.

B. The Coding Scheme

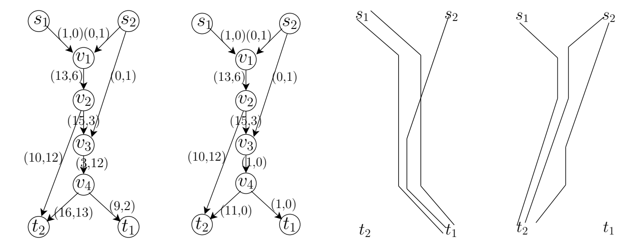
The optimization problem and the solution described thus far allocate rates at each link so that the utility function can be optimized subject to $\overrightarrow{x}$ being in the PINC region. The next question is what is the network coding scheme that can achieve the optimal rate assignment? In this section, we propose the use of a scheme we call the pairwise random coding scheme. Suppose rate x_{ij}^l is sustained along the l-th PICC between sessions i and j. From a packet-by-packet perspective it means that every $\frac{1}{x_{ij}^l}$ unit time one packet will be sent from s_i to t_i and another packet will be sent form s_j to t_j. Therefore, we can focus on coding over those two packets (every $\frac{1}{x_{ij}^l}$ unit time) along the corresponding PICC. Let the integral graph G'' represent the underlying PICC. Without loss of generality we assume that G'' is for the session pair (s_1,t_1) and (s_2,t_2). We further assume that the packets for the unicast sessions (s_1,t_1), (s_2,t_2) are X_1, X_2, respectively.

One choice of the coding scheme that is widely used is the random linear coding scheme, as in [9]. Unfortunately directly using random network coding for *pairwise inter-session network coding* without modification is infeasible. Take Fig. 3(a) for example, which is a typical choice of random network coding over GF(17) where the vector (θ_1,θ_2) at edge e in the figure represents that the packet at link e contains $\theta_1 X_1 + \theta_2 X_2$. In this case t_1 will not be able to decode both X_1 and X_2, as random mixing is performed at v_3 and t_1 will receive $9X_1 + 2X_2$. If both t_1, t_2 have min-cut max-flow values being ≥ 2, random network coding is sufficient for PINC, because both t_1, t_2 can decode both symbols. The infeasibility of random network coding is caused by the min-cut max-flow value from s_1 and s_2 to either t_1 or t_2 being 1. If the min-cut max-flow value from s_1 and s_2 to either t_1 or t_2 is 1, either the paths in the set $\mathcal{Q}_1 = \{P_{s_1,t_1}, P_{s_2,t_1}, Q_{s_1,t_1}\}$ or the paths in the set $\mathcal{Q}_2 = \{P_{s_2,t_2}, Q_{s_1,t_2}, Q_{s_2,t_2}\}$ share the same edge in G'' based on the path selection in (1). For example, in Fig. 3(c) all paths in the set $\mathcal{Q}_1$ share edge (v_3,v_4). Motivated by this observation, the pairwise random coding scheme performs pure routing and random network coding on most part of the network and performs decoding on only two nodes. The pairwise random network coding is described as follows.

Find the furthest edge $e_1 = (u_1,v_1)$ from t_1 such that (i) $\mathsf{ncp}_{\mathcal{Q}_1}(e_1) = 3$. (ii) For all paths in $\mathcal{Q}_1$ the segments from v_1 to t_1 are edge disjoint from the path P_{s_2,t_2}. Also find the furthest edge $e_2 = (u_2,v_2)$ from t_2 such that (a) $\mathsf{ncp}_{\mathcal{Q}_2}(e_2) = 3$. (b) For all paths in $\mathcal{Q}_2$ the segments from v_2 to t_2 are edge disjoint from the path P_{s_1,t_1}. After that perform random linear network coding through all the edges of G'' except edges e_1 and e_2. Decode X_1 on e_1 and forward it to t_1 through the segment of path Q_{s_1,t_1} that goes from u_1 to t_1, decode X_2 on e_2 and forward it to t_2 through the segment of path P_{s_2,t_2} that goes from u_2 to t_2. For example, if we use pairwise random coding in Fig. 3(b), (v_3,v_4) will be the first edge that satisfies the conditions for e_1 in the pairwise coding scheme as is clear from Fig. 3(c). Therefore, v_3 will decode X_1 instead of random mixing and forward it to t_2.

Theorem 3: Given that pairwise network coding is feasible on PICC G'' as in Theorem 1, the probability that the pairwise random coding scheme is able to transmit X_1 and X_2 successfully for sessions (s_1,t_1) and (s_2,t_2), is lower bounded by $P_r(success) \geq (1 - \frac{2}{q})^{|E''|}$. Here, q is the field size and $|E''|$ is the number of edges in G''.

Proof: $Pr(success)$ is lower bounded by the probability that both u_1 and u_2 recover both X_1 and X_2 successfully. Because (i) G'' is directed acyclic, (ii) the min-cut from s_1 and s_2 to u_1 is ≥ 2, (iii) the min-cut from s_1 and s_2 to u_2 is ≥ 2, we have three cases. Case 1: There is no path from v_1 to



(a) Random network coding (b) Pairwise random coding (c) The set $\mathcal{Q}_1$ (d) The set $\mathcal{Q}_2$

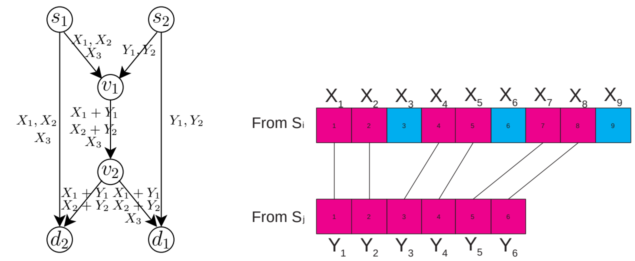
Fig. 3. Applying both the random network coding and the pairwise random network coding to the grail structure.

u_2 nor there is a path from v_2 to u_1. The problem is the same as multicasting both X_1 and X_2 to both u_1 and u_2 when all the coding coefficients are random and the inequality holds. Case 2: There exists a path from v_1 to u_2. Here we construct another graph F'' from G'' by removing all outgoing edges from v_1 and replacing them by new edges from s_1 to the same vertices that the removed edges were going to. We send X_1 through these edges and perform pairwise random coding through the rest of the edges in F''. The probability that both u_1 and u_2 recover both X_1 and X_2 on G'' is the same of that on F'', which satisfies the inequality. Case 3: There exists a path from v_2 to u_1. This case is symmetric to case 3 and so we remove the outgoing edges of v_2 and replace them by new edges from s_2 to show that the inequality holds. ■

The pairwise random coding scheme can be implemented in a distributed way. Two trace messages can be sent back by the destinations t_1 and t_2 during the initialization phase to identify edges e_1 and e_2 to perform decoding. Furthermore, by Theorem 3, we can see that the success probability of pairwise random coding scheme approaches one when the size of the finite field is sufficiently large. In practice [37] moderately-sized $q = 2^{16}$ or $q = 2^8$ is sufficient without incurring too much overhead (generally 3-6%).

C. Coding Scheme when $x^l_{ij} \neq x^l_{ji}$

The above pairwise random coding scheme assumes that two sessions (s_i, t_i) and (s_j, t_j) share the same pairwise coding rates $x^l_{ij} = x^l_{ji}$, which is achieved after the convergence of the Algorithm $\mathcal{A}$ (as proven in Theorem 2). However, during the transient time before convergence, we might have unequal cross-coding rates assigned by each individual session respectively. Furthermore, it is difficult to know when Algorithm $\mathcal{A}$ converges. To overcome these difficulties the coding scheme can be modified in the transient state when $x^l_{ij} \neq x^l_{ji}$. The basic idea is if $x^l_{ij}(\tau) > x^l_{ji}(\tau)$, we perform pairwise network coding at the smaller rate $x^l_{ji}(\tau)$ and send uncoded packets at rate $(x^l_{ij}(\tau) - x^l_{ji}(\tau))$. Therefore, t_j can receive coded packets at rate x^l_{ji} while t_i can receive coded packets at rate x^l_{ji} and uncoded packets at rate $x^l_{ij} - x^l_{ji}$ (the total rate is still x^l_{ij}). For example, assume that $x^l_{ij}(\tau) = 3$ and $x^l_{ji}(\tau) = 2$, then packets with sequence numbers 1,2,4,5,7,8,10,11 of s_i will be coded with packets with sequence numbers 1,2,3,4,5,6,7,8 of s_j. Packets with sequence numbers 3,6,9,12 of s_i will be forwarded as shown in Fig. 4(b). Take the well studied butterfly structure as an example, the assignment of the code on the edges of this PICC in the transient state is represented in Fig. 4(a). In the figure we assume that the rate is 3 for s_1 and 2 for s_2. Therefore, in one time slot s_1 will send X_1, X_2, and X_3 and s_2 will send Y_1 and Y_2. Packets X_1 and X_2 will be coded with packets Y_1 and Y_2, respectively, and packet X_3 will be forwarded without coding. In this way t_1 is able to receive at rate 3 and t_2 is able to receive at rate 2.



(a) (b)

Fig. 4. (a) The assignment of the codes on the butterfly as an example of the l-th PICC between sessions i and j with unequal rates $x^l_{ij} = 3$ and $x^l_{ji} = 2$ during the transient state. (b) The upper sequence represents the sequence numbers of packets sent by s_i through the l-th PICC between sessions i and j, and the lower sequence represents the sequence numbers of packets sent by s_j through the same PICC. If a packet in the upper sequence is to be coded with another one in the lower sequence, there is a link between them. Packets without links are those for which no coding is performed.

D. Dealing with non-concave objective functions for real-time traffic

Our assumption here is that the utility function is concave which does not hold for transmission scenarios like real-time traffic. Real-time traffic can be modelled by sigmoidal functions. In this case Algorithms $\mathcal{A}$ can be modified as in [38] to achieve the optimal rate control when the number of sessions is sufficiently large. This is possible because of the path based formulation we use.

VI. Complexity Reduction

The complexity of Algorithm $\mathcal{A}$ depends on the number of PICCs in the network. This is because for the l-th PICC between sessions i and j we assign two primal variables x^l_{ij} and x^l_{ji}. Also every link e has to maintain variables of the form $H^l_{ij}(e)$ and each destination t_i has to maintain variables of the form μ^l_{ij}. Furthermore, to compute x^l_{ij} in every update, source s_i has to collect $\sum_e \lambda_e H^l_{ij}(e)$ as explained in Section V-A. Using the approach in Section III, the number of PICCs between session i and j is $\left(|\mathcal{P}_{ii}|^2 \cdot |\mathcal{P}_{jj}|^2 \cdot |\mathcal{P}_{ij}| \cdot |\mathcal{P}_{ji}|\right)$, and so the total number of PICCs in the network is $\sum_{(i,j):i \neq j} |\mathcal{P}_{ii}|^2 \cdot |\mathcal{P}_{jj}|^2 \cdot |\mathcal{P}_{ij}| \cdot |\mathcal{P}_{ji}|$. Here, $|\mathcal{P}_{ij}|$ represents the number of paths between s_i and t_j. From the above discussion, reducing the number of PICCs in the network plays a major role in the practical implementation of the proposed algorithm. In this section we provide two approaches to reduce the complexity of Algorithm $\mathcal{A}$. The first approach reduces the number of PICCs without sacrificing performance, while the second chooses paths and PICCs adaptively and may sacrifice performance.

A. Excluding redundant PICCs in the initialization step

By construction, the l-th PICC between sessions i and j satisfies condition 2 of Theorem 1 and supports the coded traffic rate x_{ij}^l. In this section we provide rules to see whether condition 1 of Theorem 1 is also satisfied on that PICC at rate x_{ij}^l. If so, we can remove x_{ij}^l and $H_{ij}^l(e)$, $\forall e$ from the optimization problem (5) and still achieve the same optimal solution. This is because the 2EDPs in that PICC can be used to send uncoded traffic which has already been characterized by x_i^k, the uncoded data rate in (3). The rules also help detecting whether a given PICC contains redundant links such that there exists another PICC whose links are a proper subset of the links used by the given PICC. Since for these PICCs, we can use a strictly smaller part of the PICC while still providing the same throughput improvement, we term those PICCs redundant PICCs. The following rules remove redundant PICC and result in a tremendous reduction in the complexity of Algorithm $\mathcal{A}$. In the following we assume the same simplification as in section V-B by considering the integral graph G'' for session pair $(s_1, t_1), (s_2, t_2)$. To identify the PICCs that should be removed from the optimization problem, we have the following rules:

- Rule 1: For G'', if neither $P_{s_1,t_1} = Q_{s_1,t_1}$ nor $P_{s_2,t_2} = Q_{s_2,t_2}$, then G'' is a redundant PICC.
- Rule 2: For a given path P, let $E(P)$ be the set of edges in path P. If the two sets of edges $E(P_{s_1,t_1}) \bigcup E(Q_{s_1,t_1})$ and $E(P_{s_2,t_2}) \bigcup E(Q_{s_2,t_2})$ are disjoint, then G'' is a redundant PICC.
- Rule 3: If $H_{ij}^l(e) = 1$ for some link e, then the l-th PICC between sessions i and j is redundant.

If a PICC is classified as redundant by a rule, we say that the PICC is declared redundant by that rule. Otherwise, we say that the PICC passes that rule.

Theorem 4: Rules 1-3 identify redundant PICCs. All redundant PICCs can be removed without sacrificing the achievable rate.

Based on Theorem 4, we have the following reduced-complexity algorithm which achieves the same capacity region as by Algorithm $\mathcal{A}$.

(*Algorithm* $\mathcal{B}$):

1) *Path Finding*: Every source finds a set of paths to every destination, and announces the sizes of these sets to the links in these paths and to other sources. Every PICC that passes Rule 1 will have a unique ID number which can be computed locally at every link in the network.
2) Source s_i sends trace message through all paths $P_{s_i t_j}$. This message includes i, j, and the path number.
3) Link e sets up $H_{ij}^l(e)$ for all PICCs that pass Rule 1.
4) The following steps are executed at every link e in parallel
   - If P_{s_i,t_i}^l or Q_{s_i,t_i}^l share link e with P_{s_j,t_j}^l or Q_{s_j,t_j}^l a notification message of type $\mathcal{X}$ is sent back to s_i, s_j, with the ID number of the l-th PICC between sessions i and j. This means that the PICC passes Rule 2. Here, P_{s_i,t_i}^l represents the path from s_i to t_i in the set $\mathcal{P}$ in Theorem 1 for the l-th PICC between sessions i and j. $Q_{s_i,t_i}^l, Q_{s_j,t_j}^l$, and P_{s_j,t_j}^l are defined in the same way.
   - If $H_{ij}^l(e) = 1$, link e sends a notification message of type $\mathcal{Y}$ to s_i, s_j, with the ID number of the l-th PICC between sessions i and j. This means the l-th PICC between i and j is redundant by Rule 3.
5) Sources s_i and s_j delete all PICC for which a notification message of type $\mathcal{Y}$ is received or no notification message of type $\mathcal{X}$ is received.
6) Run Algorithm $\mathcal{A}$ on the non deleted PICCs.

Steps (1)-(5) are initialization steps and executed only once. Also, they can be executed in a distributed manner.

B. Adaptive Algorithm

The reductions in Section VI-A reduce the number of PICCs in the initialization step without sacrificing the performance by eliminating redundant PICCs. To further reduce the complexity, we propose another type of reduction. As observed by our simulations the rates on some PICCs converge to zero very quickly (generally after only a few iterations), which means that network coding over those PICCs provides no positive gain when compared to the optimal rate-control solution. This may be due to that non-coded solution is sufficient for those PICCs because they satisfy both conditions of Theorem 1. It may also be because the links used by the PICCs can be used by more significant PICCs. We termed those PICCs as *insignificant* PICCs. The adaptive scheme we propose in this section works initially on a small number of PICCs. While the algorithm is run on these PICCs, insignificant PICCs among them will be deleted and new PICCs formed by the newly found paths will be added adaptively. This approach reduces the variable space of the optimization problem. Fig. 5 contains a detailed description of the above scheme with an adaptive path search mechanism.

In the flow chart in Fig. 5, every source maintains a collection of paths $\mathcal{P}_{found}$ and every source pair maintains collections of PICCs $\mathcal{PICC}_{found}$ and $\mathcal{PICC}_{active}$. Every time new paths are found, the source puts them in $\mathcal{P}_{found}$ which is executed in parallel to the steps in Fig. 5. When $\mathcal{PICC}_{found}$ becomes empty, all possible PICCs that can be formed by the paths in the $\mathcal{P}_{found}$ and have not been used by the algorithm yet, are put in $\mathcal{PICC}_{found}$. Each pair of sessions i and j is assigned a value ϕ_{ij}. The constant ϕ_{ij} represents the maximum number of PICCs (for sessions i and j) that can be included in the maximization problem simultaneously. If the rate of a given PICC converges to zero, it is identified as an insignificant one. Convergence to zero is detected when the rate of the PICC goes below a threshold value. Every time insignificant PICCs (for sessions i and j) are removed from the optimization problem, the complexity of the algorithm reduces and we can afford to include one new PICC when solving the optimization problem. Therefore, s_i moves some of the PICCs (for sessions i and j) from $\mathcal{PICC}_{found}$ to $\mathcal{PICC}_{active}$. This is done in a way such that the total number of PICCs for i and j in $\mathcal{PICC}_{active}$ does not exceed ϕ_{ij}. For distributed implementation one source of each pair is assigned

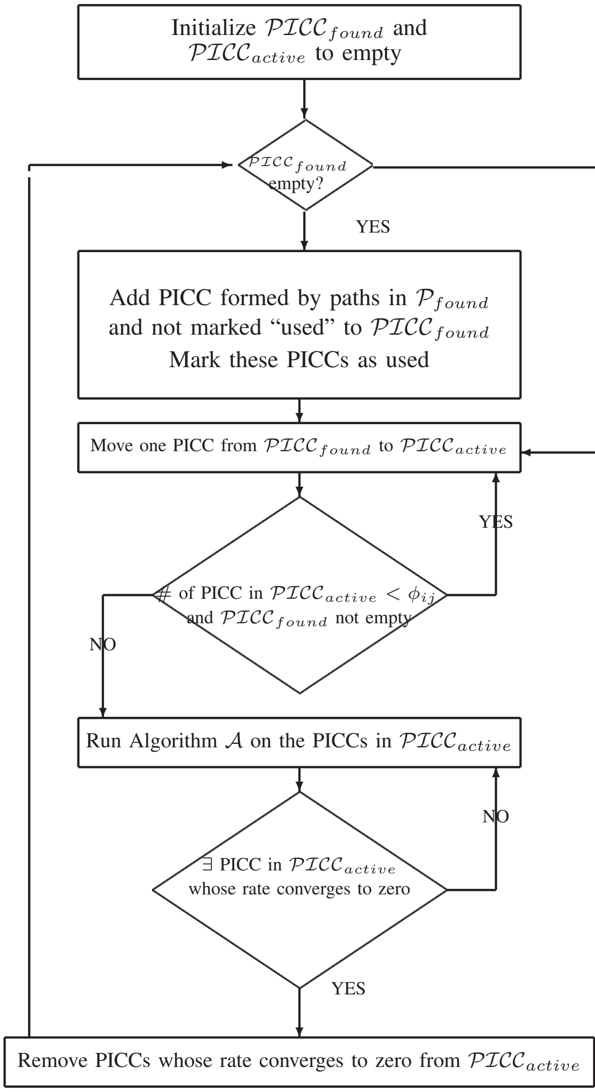


Fig. 5. Flow chart for the adaptive algorithm.

as a controller to ensure the consistency of $\mathcal{PICC}_{found}$ and $\mathcal{PICC}_{active}$ for the two sources.

VII. Simulation Results

The objectives of the simulations are to verify the convergence of Algorithms $\mathcal{A}$ and $\mathcal{B}$, and to show the benefits of the proposed inter-session network coding solution in terms of throughput, fairness, and its complexity advantage over existing inter-session coding rate-control solutions.

A. Convergence

To study the convergence of Algorithms $\mathcal{A}$ and $\mathcal{B}$, we run simulations on the so called grail topology in Fig. 2(a) with the utility function of each source s_i being $\log_2(R_i)$. As is evident from the grail topology (Fig. 2(a)), there are three paths connecting (s_2,t_2), two paths connecting (s_1,t_2), two paths connecting (s_2,t_1), and one path connecting (s_1,t_1). Therefore, there are six different path collections $\mathcal{P}$ and six different path collections $\mathcal{Q}$. Totally, there are 36 possible PICCs. We assign the initial rates of each PICC randomly, and vary α_e, α_{ij}^l and the number of proximal iterations K to test the speed of convergence of Algorithm $\mathcal{A}$.

The optimal solution for the grail topology is to assign unit rate to the optimal PICC that uses the paths in (1), and zero rates to all of the other PICCs. These are the paths that satisfy condition 2 in Theorem 1 as explained in section II. In Fig. 6 we show the rates for the optimal PICC with different step sizes. Every outer iteration contains K proximal iterations. Our algorithm converges even with a very small number of proximal iterations. As expected, increasing the step size up to a specific value will make the algorithm converge faster. A bigger topology in Fig. 7 with 36 nodes and unit capacity links is used in our simulations. This topology has four unicast sessions. The convergence results for one of the optimal PICCs and one of the insignificant PICCs in the topology in Fig. 7 are shown in Fig. 8. In Fig. 8, the rate of the insignificant PICC converges quickly to zero.

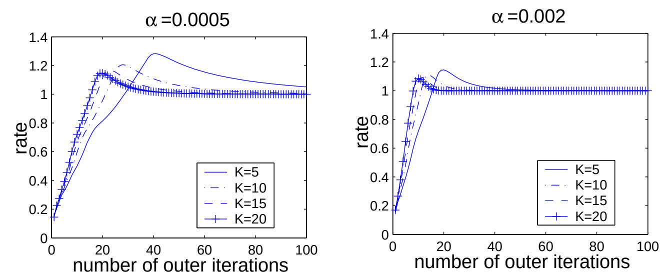


Fig. 6. Convergence results for s_1 in the grail topology with different step sizes and K, the number of proximal iterations. Here, the rate corresponds to the optimal PICC.

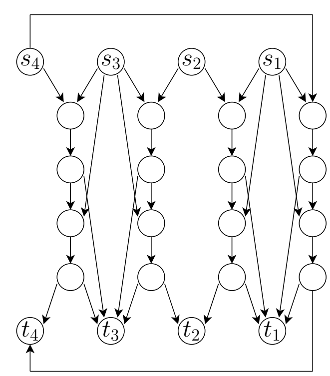


Fig. 7. Topology contains four source-sink pairs.

B. Gain and Fairness

We compare Algorithm $\mathcal{A}$ with existing algorithms and quantify the benefits of inter-session network coding over non-coded solutions. The simulation is conducted on a graph depicted in Fig. 7. For this topology, the butterfly-based work [24] and its distributed implementation in [25] and [26] cannot realize any throughput benefits of network coding and the performance of these algorithms is the same as that of non-coded solutions since there is no butterfly substructure in Fig. 7. It is worth noting that the distributed implementations

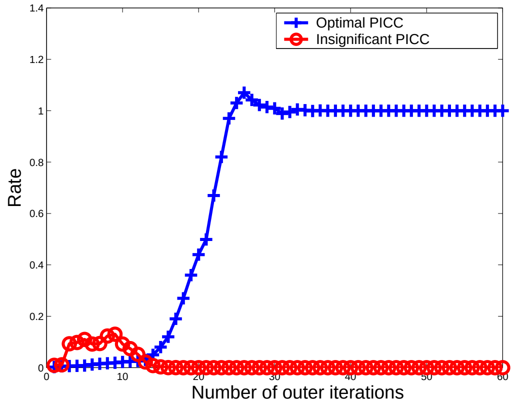


Fig. 8. Convergence results for the topology in Fig. 7 with $\alpha = 0.01$, and the number of proximal iterations $K = 5$. We plot the convergence rate vs. iterations for the optimal PICC and one insignificant PICC.

of the butterfly-based region in [25], [26] focus on stabilizing the given traffic load instead of maximizing the utility function. We define the utility gain of pairwise inter-session network coding PINC, $\mathcal{UG}$ as

$$\mathcal{UG} = \frac{Utility(\text{PINC}) - Utility(\text{non-coded})}{Utility(\text{non-coded})}.$$

We denote the total throughput of the network when the optimal utility is achieved under the PINC and the non-coded solutions by $\sum_i R_i(\text{PINC})$ and $\sum_i R_i(\text{non-coded})$, respectively. The throughput gain, $\mathcal{TG}$ is defined as

$$\mathcal{TG} = \frac{\sum_i R_i(\text{PINC}) - \sum_i R_i(\text{non-coded})}{\sum_i R_i(\text{non-coded})}.$$

We evaluate the gains of Algorithm $\mathcal{A}$ using different utility functions presented in [1] and [2]. The first type of utility function is $\log_2(\delta + R_i)$, where δ is a constant in the range $[0, 1]$. The second type of utility function is of the form $\frac{R_i^{1-\sigma}}{1-\sigma}$, where σ is a constant in the range $(0, 1)$. The results are shown in Figs. 9 and 10. Algorithm $\mathcal{A}$ provides strict performance gains over both non-coded and butterfly-based capacity region on this topology. Moreover, the largest throughput gain happens when fairness is the design criteria for the network, i.e, when δ is small and when σ is large. This is the same conclusion drawn from the capacity regions in Figs. 1(b) and 2(b).

We also run simulations on the grid topology in Fig. 11. In this topology there are four paths between s_1 and t_1, four paths between s_2 and t_2, and only one path between s_3 and t_3. Also, the path between s_3 and t_3 overlaps with all the other eight paths. Therefore, without network coding the rates of sessions 1 and 2 are about 3.7 times the rate of session 3 when δ is small 0.1 (fairness is of high priority) as shown in Table I. Using network coding the rate ratio is reduced to about 2.2 times with a small decrease in rates R_1 and R_2 and a considerable increase in R_3 from 0.29 to 0.47. When δ is relatively large 0.6, the rate ratio without network coding is about 21, because less emphasis is put on the smaller rate (rate of session 3 in this case.) Surprisingly with network coding the rate ratio is reduced to 4. This decrease in the ratio is due to that network coding resolves the bottlenecks. The fairness is thus improved as network coding removes the bottleneck for the smallest-rate session 3.

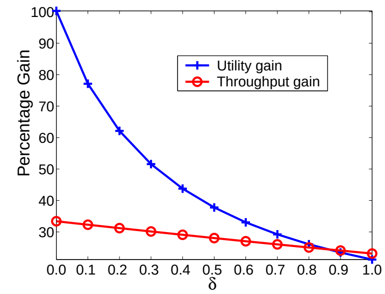


Fig. 9. Gain for the topology in Fig. 7 with the objective function $\sum_i \log_2(\delta + R_i)$ and different values of δ.

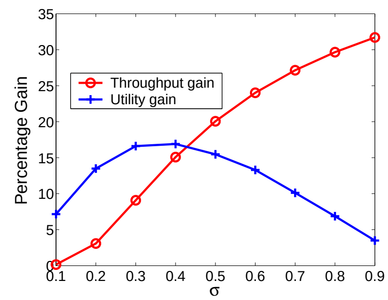


Fig. 10. Gain for the topology in Fig. 7 with the objective function $\sum_i \frac{R_i^{1-\sigma}}{1-\sigma}$ and different values of σ.

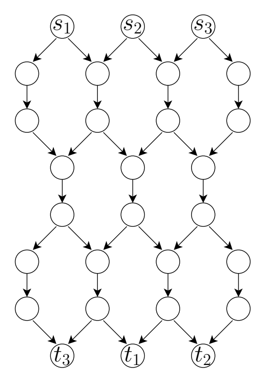


Fig. 11. Grid topology with three sessions

C. Complexity

In terms of computational complexity of Algorithm $\mathcal{A}$ and the existing butterfly-based method [24], the path based Algorithm $\mathcal{A}$ solves a maximization problem of 2,328 variables and 36 constraints in a distributed way for the topology in Fig. 7, while the pattern-search-based optimization problem [24] has more than 31,104 variables and 31,104 constraints. Furthermore, if we use Algorithm $\mathcal{B}$, we can reduce the number of variables to 22. Also, for the topology in Fig. 11 the number of variables using the pattern search algorithm is more than 17,000 and the number of constraints is more than 30,000. The number of variables is reduced to about 3000 using Algorithm

	$\delta=0.1$			$\delta=0.2$		
	R_1	R_2	R_3	R_1	R_2	R_3
Non-coded	1.066	1.066	0.289	1.133	1.133	0.244
PINC	1.033	1.033	0.467	1.067	1.0667	0.433
	$\delta=0.4$			$\delta=0.6$		
	R_1	R_2	R_3	R_1	R_2	R_3
Non-coded	1.266	1.266	0.155	1.399	1.399	0.066
PINC	1.133	1.133	0.367	1.200	1.200	0.300

TABLE I
RATE R_i ASSIGNED FOR EACH SESSION IN FIG. 11 USING ROUTING AND INTER-SESSION NETWORK CODING WITH THE OBJECTIVE FUNCTION $\sum_i \log_2(\delta+R_i)$ AND DIFFERENT VALUES OF δ

$\mathcal{A}$ and to 300 using Algorithm $\mathcal{B}$ and the number of constraints is reduced to 35 using both Algorithms $\mathcal{A}$ and $\mathcal{B}$. In sum, the flexible choice of utility functions, decentralized rate control capability, superior performance in terms of utility/throughput gains, fairness and manageable complexity with an adaptive path search mechanism, demonstrate the efficacy of the path-based Algorithms $\mathcal{A}$ and $\mathcal{B}$.

VIII. Conclusion

In this paper we develop a distributed rate control algorithm for the multiple-unicast-sessions problem. The algorithm supports rates in the PINC achievable rate region that allows for inter-session network coding. We also propose a distributed pairwise random coding scheme suitable for online implementation. Our algorithm improves both throughput and fairness among flows in information networks.

Appendix A
Notations used for the proof of Theorem 2

Let

$$V(\overrightarrow{x})=\begin{cases}\sum_{i=1}^{N}U_i(\sum_{k=1}^{|\mathcal{P}_i|}x_i^k+\\ \qquad\sum_{j\neq i}\sum_{l=1}^{|\mathcal{PICC}_{ij}|}x_{ij}^l) & \overrightarrow{x}\geq 0\\ -\infty & \text{else.}\end{cases}$$

Then V is the extended concave objective function, and we can write our problem in the following matrix form:

$$\max V(\overrightarrow{x})$$

subject to: $\mathbf{A}\overrightarrow{x}\leq\overrightarrow{C}$ and $\mathbf{B}\overrightarrow{x}=0$, and the Lagrangian can be written as:

$$L(\overrightarrow{x},\overrightarrow{\lambda},\overrightarrow{\mu},\overrightarrow{y})=V(\overrightarrow{x})-\overrightarrow{x}^T\begin{bmatrix}\mathbf{A}^T\mathbf{B}^T\end{bmatrix}\begin{bmatrix}\overrightarrow{\lambda}\\ \overrightarrow{\mu}\end{bmatrix} -\frac{1}{2}(\overrightarrow{x}-\overrightarrow{y})^T\mathbf{C}(\overrightarrow{x}-\overrightarrow{y}),\quad(10)$$

where $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are constructed as follows. Let $M(i)=\sum_{j:j\neq i}|\mathcal{PICC}_{i,j}|$, $\mathbf{A}$ is a matrix with $|E|$ rows and $(\sum_i|\mathcal{P}_i|+2M(i))$ columns, where the e-th row is filled with $H_i^k(e)$ and $H_{ij}^l(e)$ in the same order as the corresponding x_i^k and x_{ij}^l appear in $\overrightarrow{x}$. Matrix $\mathbf{B}$ is defined as a matrix with $\sum_i M(i)$ rows and $(\sum_i|\mathcal{P}_i|+2M(i))$ columns, such that the (a,b) entry in $\mathbf{B}$ is 1 if the b-th entry in $\overrightarrow{x}$ is a variable of the form x_{ij}^l and the a-th entry in $\overrightarrow{\mu}$ is μ_{ij}^l. The (a,b) entry in $\mathbf{B}$ is -1 if the b-th entry in $\overrightarrow{x}$ is a variable of the form x_{ij}^l and the a-th entry in $\overrightarrow{\mu}$ is μ_{ji}^l. For all other cases, the (a,b) entry in $\mathbf{B}$ is 0. We define $\mathbf{C}$ as a diagonal matrix of size $(\sum_i|\mathcal{P}_i|+2M(i))$. If the a-th variable of $\overrightarrow{x}$ is x_i^k or x_{ij}^l for some i, the a-th diagonal elements of $\mathbf{C}$ will be γ_i.

Throughout the proof we use the following norm $\|A\|_{\mathbf{D}}=A^T\mathbf{D}^{-1}A$, where $\mathbf{D}=\begin{bmatrix}\mathbf{E}&0\\0&\mathbf{F}\end{bmatrix}$. Here, $\mathbf{E}$ is an $|E|$ elements diagonal matrix with the e-th entry being α_e. Matrix $\mathbf{F}$ is another $(\sum_i M(i))$ diagonal matrix, in which the a-th diagonal element is α_{ij}^l where i,j,l are the same indices of the a-th element of $\overrightarrow{\mu}$.

Appendix B
Lemma 1

Lemma 1: Fix $\overrightarrow{y}$. Let $\begin{bmatrix}\overrightarrow{\lambda_1}\\ \overrightarrow{\mu_1}\end{bmatrix}$ and $\begin{bmatrix}\overrightarrow{\lambda_2}\\ \overrightarrow{\mu_2}\end{bmatrix}$ be two implicit cost vectors and let $\overrightarrow{x_1}^*$ and $\overrightarrow{x_2}^*$ be the corresponding maximizers of the Lagrangian, then:

$$\begin{bmatrix}(\overrightarrow{\lambda_1}-\overrightarrow{\lambda_2})^T & (\overrightarrow{\mu_1}-\overrightarrow{\mu_2})^T\end{bmatrix}\begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix}(\overrightarrow{x_1}^*-\overrightarrow{x_2}^*)\leq -(\overrightarrow{x_1}^*-\overrightarrow{x_2}^*)^T\mathbf{C}(\overrightarrow{x_1}^*-\overrightarrow{x_2}^*).$$

Proof: Let $\overrightarrow{x_0}^*=\arg\max_{\overrightarrow{x}}L(\overrightarrow{x},\overrightarrow{\lambda},\overrightarrow{\mu},\overrightarrow{y})$. By taking the subgradient of (10) with respect to $\overrightarrow{x}$, we can see that there must exist a subgradient of (10) at $\overrightarrow{x_0}^*$ such that:

$$\nabla V(\overrightarrow{x_0}^*)-\begin{bmatrix}\mathbf{A}^T & \mathbf{B}^T\end{bmatrix}\begin{bmatrix}\overrightarrow{\lambda}\\ \overrightarrow{\mu}\end{bmatrix}-\mathbf{C}(\overrightarrow{x_0}^*-\overrightarrow{y})=0.\quad(11)$$

Substituting $\begin{bmatrix}\overrightarrow{\lambda}\\ \overrightarrow{\mu}\end{bmatrix}$ in (11) by $\begin{bmatrix}\overrightarrow{\lambda_1}\\ \overrightarrow{\mu_1}\end{bmatrix}$ and $\begin{bmatrix}\overrightarrow{\lambda_2}\\ \overrightarrow{\mu_2}\end{bmatrix}$, respectively, and taking the difference, we have:

$$\begin{bmatrix}\mathbf{A}^T & \mathbf{B}^T\end{bmatrix}\begin{bmatrix}\overrightarrow{\lambda_1}-\overrightarrow{\lambda_2}\\ \overrightarrow{\mu_1}-\overrightarrow{\mu_2}\end{bmatrix}=\begin{bmatrix}\nabla V(\overrightarrow{x_1}^*)-\nabla V(\overrightarrow{x_2}^*)\end{bmatrix} -\mathbf{C}(\overrightarrow{x_1}^*-\overrightarrow{x_2}^*).$$

Since V is concave we have:

$$\begin{bmatrix}\nabla V(\overrightarrow{x_1}^*)-\nabla V(\overrightarrow{x_2}^*)\end{bmatrix}^T(\overrightarrow{x_1}^*-\overrightarrow{x_2}^*)\leq 0.$$

Hence

$$\begin{aligned}&\begin{bmatrix}(\overrightarrow{\lambda_1}-\overrightarrow{\lambda_2})^T & (\overrightarrow{\mu_1}-\overrightarrow{\mu_2})^T\end{bmatrix}\begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix}(\overrightarrow{x_1}^*-\overrightarrow{x_2}^*)\\ &=\begin{bmatrix}\nabla V(\overrightarrow{x_1}^*)-\nabla V(\overrightarrow{x_2}^*)\end{bmatrix}^T(\overrightarrow{x_1}^*-\overrightarrow{x_2}^*)\\ &\quad-(\overrightarrow{x_1}^*-\overrightarrow{x_2}^*)^T\mathbf{C}(\overrightarrow{x_1}^*-\overrightarrow{x_2}^*)\\ &\leq-(\overrightarrow{x_1}^*-\overrightarrow{x_2}^*)^T\mathbf{C}(\overrightarrow{x_1}^*-\overrightarrow{x_2}^*).\end{aligned}$$

■

Appendix C
Lemma 2

Lemma 2: Let $\mathcal{L}=\sum_e\Big(\sum_{i=1}^N\sum_{k=1}^{|\mathcal{P}_i|}H_i^k(e)+\sum_{(i,j):i\neq j}\sum_{l=1}^{|\mathcal{PICC}_{ij}|}(H_{ij}^l(e))^2\Big)$. The sufficient condition for $2\mathbf{C}-\mathbf{A}^T\mathbf{E}\mathbf{A}-\mathbf{B}^T\mathbf{F}\mathbf{B}$ to be positive definite is that the step sizes α_e, α_{ij}^l fall in the following region:
$(\mathcal{L}\cdot\max_e\alpha_e+2\max_{i,j,l}\alpha_{ij}^l)<2\min_i(\gamma_i)$

$$\left\|\begin{bmatrix}\overrightarrow{\lambda}(\tau,\kappa+1)-\overrightarrow{\lambda_0}\\ \overrightarrow{\mu}(\tau,\kappa+1)-\overrightarrow{\mu_0}\end{bmatrix}\right\|_{\mathbf{D}}=\left\|\begin{bmatrix}[\overrightarrow{\lambda}(\tau,\kappa)+\mathbf{E}(\mathbf{A}\overrightarrow{x}(\tau,\kappa)-\overrightarrow{C})]^{+}-[\overrightarrow{\lambda_0}+\mathbf{E}(\mathbf{A}\overrightarrow{x_0}-\overrightarrow{C})]^{+}\\ \overrightarrow{\mu}(\tau,\kappa)+\mathbf{FB}\overrightarrow{x}(\tau,\kappa)-\overrightarrow{\mu_0}+\mathbf{FB}\overrightarrow{x_0}\end{bmatrix}\right\|_{\mathbf{D}}$$

$$\leq\left\|\begin{bmatrix}\overrightarrow{\lambda}(\tau,\kappa)+\mathbf{E}(\mathbf{A}\overrightarrow{x}(\tau,\kappa)-\overrightarrow{C})-\overrightarrow{\lambda_0}+\mathbf{E}(\mathbf{A}\overrightarrow{x_0}-\overrightarrow{C})\\ \overrightarrow{\mu}(\tau,\kappa)+\mathbf{FB}\overrightarrow{x}(\tau,\kappa)-\overrightarrow{\mu_0}+\mathbf{FB}\overrightarrow{x_0}\end{bmatrix}\right\|_{\mathbf{D}}=\left\|\begin{bmatrix}\overrightarrow{\lambda}(\tau,\kappa)-\overrightarrow{\lambda_0}+\mathbf{EA}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})\\ \overrightarrow{\mu}(\tau,\kappa)-\overrightarrow{\mu_0}+\mathbf{FB}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})\end{bmatrix}\right\|_{\mathbf{D}} \quad (12)$$

Proof: If $2\mathbf{C}-\mathbf{A}^T\mathbf{E}\mathbf{A}-\mathbf{B}^T\mathbf{F}\mathbf{B}$ is positive definite, then

$$\overrightarrow{\delta x}^T(2\mathbf{C}-\mathbf{A}^T\mathbf{E}\mathbf{A}-\mathbf{B}^T\mathbf{F}\mathbf{B})\overrightarrow{\delta x}>0,$$

for all nonzero column vectors $\overrightarrow{\delta x}$, which is equivalent to

$$2\overrightarrow{\delta x}^T\mathbf{C}\overrightarrow{\delta x}>\overrightarrow{\delta x}^T(\mathbf{A}^T\mathbf{E}\mathbf{A}+\mathbf{B}^T\mathbf{F}\mathbf{B})\overrightarrow{\delta x}.$$

We use δx_i to refer to the i-th element of $\overrightarrow{\delta x}$. By the Cauchy-Schwartz Inequality, we have

$$\begin{aligned}
&\overrightarrow{\delta x}^T(\mathbf{A}^T\mathbf{E}\mathbf{A})\overrightarrow{\delta x}+\overrightarrow{\delta x}^T(\mathbf{B}^T\mathbf{F}\mathbf{B})\overrightarrow{\delta x}\\
=\;&\sum_e\alpha_e(\sum_i\sum_k H_i^k(e)\delta x_i^k\\
&+\sum_{(i,j):i\neq j}\sum_{l=1}^{|\mathcal{PICC}_{ij}|}(H_{ij}^l(e))\delta x_{ij}^l)^2\\
&+\sum_{(i,j):i<j}\sum_l\alpha_{ij}^l[(\delta x_{ij}^l-\delta x_{ji}^l)^2]\\
\leq\;&\sum_e\alpha_e(\sum_i\sum_k H_i^k(e)\\
&+\sum_{(i,j):i\neq j}\sum_{l=1}^{|\mathcal{PICC}_{ij}|}(H_{ij}^l(e))^2)\sum_i\delta x_i^2\\
&+\sum_{(i,j):i<j}\sum_l(\alpha_{ij}^l(2\delta x_{ij}^l)^2+2(\delta x_{ji}^l)^2)\\
\leq\;&\max_e\alpha_e(\sum_e(\sum_i\sum_k H_i^k(e)\\
&+\sum_{(i,j):i\neq j}\sum_{l=1}^{|\mathcal{PICC}_{ij}|}(H_{ij}^l(e))^2))\sum_i\delta x_i^2\\
&+(\max_{i,j,l}\alpha_{ij}^l)2\sum_i\delta x_i^2\\
=\;&(\mathcal{L}\cdot\max_e\alpha_e+2\max_{i,j,l}\alpha_{ij}^l)\sum_i\delta x_i^2
\end{aligned}$$

Therefore if the inequality that

$$(\mathcal{L}\cdot\max_e\alpha_e+2\max_{i,j,l}\alpha_{ij}^l)\sum_i\delta x_i^2<2\min_i(\gamma_i)\sum_i\delta x_i^2$$

holds, then we have

$$2\overrightarrow{\delta x}^T\mathbf{C}\overrightarrow{\delta x}\quad>\quad\overrightarrow{\delta x}^T(\mathbf{A}^T\mathbf{E}\mathbf{A}+\mathbf{B}^T\mathbf{F}\mathbf{B})\overrightarrow{\delta x}$$

which in turn implies that $2\mathbf{C}-\mathbf{A}^T\mathbf{E}\mathbf{A}-\mathbf{B}^T\mathbf{F}\mathbf{B}$ is positive definite. The proof is complete. ■

Appendix D
Proof of *Theorem 2*

In this section, we prove Theorem 2 the convergence of Algorithm $\mathcal{A}$ for the case that K tends to infinity first and then let the number of iterations goes to infinity.

Proof: We will prove the convergence of Algorithm $\mathcal{A}$ when $K\longrightarrow\infty$. To do so, we will prove the convergence of the first step during the proximal iteration. The convergence of the whole algorithm follows from [33] page 233. Fix $\overrightarrow{y}(\tau)$. Let $\overrightarrow{\lambda_0},\overrightarrow{\mu_0}$ be a stationary point of (8) and (9), and let $\overrightarrow{x_0}$ be the corresponding primal variable. $\overrightarrow{x_0}$ is unique since $L(\cdot)$ is strictly concave with respect to $\overrightarrow{x}$, and $\overrightarrow{x_0}=\arg\max_{\overrightarrow{x}}L(\overrightarrow{x},\overrightarrow{\lambda},\overrightarrow{\mu},\overrightarrow{y})$. By the projection theorem in [33] page 211 we have the inequality in (12) which gives:

$$\begin{aligned}
&\left\|\begin{bmatrix}\overrightarrow{\lambda}(\tau,\kappa+1)-\overrightarrow{\lambda_0}\\ \overrightarrow{\mu}(\tau,\kappa+1)-\overrightarrow{\mu_0}\end{bmatrix}\right\|_{\mathbf{D}}\\
\leq&\left\|\begin{bmatrix}\overrightarrow{\lambda}(\tau,\kappa)-\overrightarrow{\lambda_0}\\ \overrightarrow{\mu}(\tau,\kappa)-\overrightarrow{\mu_0}\end{bmatrix}\right\|_{\mathbf{D}}+\begin{bmatrix}\mathbf{EA}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})\\ \mathbf{FB}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})\end{bmatrix}^T\\
&\mathbf{D}^{-1}\begin{bmatrix}\mathbf{EA}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})\\ \mathbf{FB}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})\end{bmatrix}\\
&+2\begin{bmatrix}\overrightarrow{\lambda}(\tau,\kappa)-\overrightarrow{\lambda_0}\\ \overrightarrow{\mu}(\tau,\kappa)-\overrightarrow{\mu_0}\end{bmatrix}^T\mathbf{D}^{-1}\begin{bmatrix}\mathbf{EA}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})\\ \mathbf{FB}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})\end{bmatrix}\\
=&\left\|\begin{bmatrix}\overrightarrow{\lambda}(\tau,\kappa)-\overrightarrow{\lambda_0}\\ \overrightarrow{\mu}(\tau,\kappa)-\overrightarrow{\mu_0}\end{bmatrix}\right\|_{\mathbf{D}}+(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})^T\begin{bmatrix}\mathbf{A}^T & \mathbf{B}^T\end{bmatrix}\\
&\begin{bmatrix}\mathbf{E}^T & 0\\ 0 & \mathbf{F}^T\end{bmatrix}\mathbf{D}^{-1}\begin{bmatrix}\mathbf{E} & 0\\ 0 & \mathbf{F}\end{bmatrix}\begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})\\
&+2\begin{bmatrix}\overrightarrow{\lambda}(\tau,\kappa)-\overrightarrow{\lambda_0}\\ \overrightarrow{\mu}(\tau,\kappa)-\overrightarrow{\mu_0}\end{bmatrix}^T\\
&\mathbf{D}^{-1}\begin{bmatrix}\mathbf{E} & 0\\ 0 & \mathbf{F}\end{bmatrix}\begin{bmatrix}\mathbf{A} & 0\\ 0 & \mathbf{B}\end{bmatrix}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})\\
=&\left\|\begin{bmatrix}\overrightarrow{\lambda}(\tau,\kappa)-\overrightarrow{\lambda_0}\\ \overrightarrow{\mu}(\tau,\kappa)-\overrightarrow{\mu_0}\end{bmatrix}\right\|_{\mathbf{D}}+(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})^T\begin{bmatrix}\mathbf{A}^T & \mathbf{B}^T\end{bmatrix}\\
&\mathbf{D}\mathbf{D}^{-1}\mathbf{D}\begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})\\
&+2\begin{bmatrix}\overrightarrow{\lambda}(\tau,\kappa)-\overrightarrow{\lambda_0}\\ \overrightarrow{\mu}(\tau,\kappa)-\overrightarrow{\mu_0}\end{bmatrix}^T\mathbf{D}^{-1}\mathbf{D}\begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix}(\overrightarrow{x}(\tau,\kappa)-\overrightarrow{x_0})
\end{aligned}$$

By *Lemma 1* we have:

$$\left\|\begin{bmatrix}\overrightarrow{\lambda}(\tau,\kappa+1)-\overrightarrow{\lambda_0}\\ \overrightarrow{\mu}(\tau,\kappa+1)-\overrightarrow{\mu_0}\end{bmatrix}\right\|_{\mathbf{D}}$$

$$\leq \left\| \begin{bmatrix} \overrightarrow{\lambda}(\tau,\kappa) - \overrightarrow{\lambda_0} \\ \overrightarrow{\mu}(\tau,\kappa) - \overrightarrow{\mu_0} \end{bmatrix} \right\|_{\mathbf{D}} + (\overrightarrow{x}(\tau,\kappa) - \overrightarrow{x_0})^T \begin{bmatrix} \mathbf{A}^T & \mathbf{B}^T \end{bmatrix} \mathbf{D} \begin{bmatrix} \mathbf{A} \\ \mathbf{B} \end{bmatrix} (\overrightarrow{x}(\tau,\kappa) - \overrightarrow{x_0}) - 2(\overrightarrow{x}(\tau,\kappa) - \overrightarrow{x_0})^T \mathbf{C} (\overrightarrow{x}(\tau,\kappa) - \overrightarrow{x_0})$$

$$= \left\| \begin{bmatrix} \overrightarrow{\lambda}(\tau,\kappa) - \overrightarrow{\lambda_0} \\ \overrightarrow{\mu}(\tau,\kappa) - \overrightarrow{\mu_0} \end{bmatrix} \right\|_{\mathbf{D}} - (\overrightarrow{x}(\tau,\kappa) - \overrightarrow{x_0})^T \mathbf{L} (\overrightarrow{x}(\tau,\kappa) - \overrightarrow{x_0}),$$

where $\mathbf{L} = 2\mathbf{C} - \begin{bmatrix} \mathbf{A}^T & \mathbf{B}^T \end{bmatrix} \mathbf{D} \begin{bmatrix} \mathbf{A} \\ \mathbf{B} \end{bmatrix}$. When $\mathbf{L}$ is positive definite, we have:

$$\left\| \begin{bmatrix} \overrightarrow{\lambda}(\tau,\kappa+1) - \overrightarrow{\lambda_0} \\ \overrightarrow{\mu}(\tau,\kappa+1) - \overrightarrow{\mu_0} \end{bmatrix} \right\|_{\mathbf{D}} \leq \left\| \begin{bmatrix} \overrightarrow{\lambda}(\tau,\kappa) - \overrightarrow{\lambda_0} \\ \overrightarrow{\mu}(\tau,\kappa) - \overrightarrow{\mu_0} \end{bmatrix} \right\|_{\mathbf{D}}. \quad (13)$$

Therefore, if the step sizes α_e and α_{ij}^l satisfy the condition in Theorem 2, then by *Lemma 2*, $\mathbf{L}$ is positive definite. Accordingly $\left\| \begin{bmatrix} \overrightarrow{\lambda}(\tau,\kappa) - \overrightarrow{\lambda_0} \\ \overrightarrow{\mu}(\tau,\kappa) - \overrightarrow{\mu_0} \end{bmatrix} \right\|_{\mathbf{D}}$ will be a nonnegative and decreasing sequence. Therefore, as $K \longrightarrow \infty$, $\overrightarrow{x}(\tau,K) \longrightarrow \overrightarrow{x_0}$.

■

APPENDIX E
PROPOSITION 1

Recall the following theorem from [27].

Theorem 5: Define two sets of integral graphs, $\mathcal{G}_b$ and $\mathcal{G}_g$, as follows.

1) $\mathcal{G}_b$ contains the full butterfly as described in Fig. 1(a), and all graphs obtained from the full butterfly via edge contraction. See [39] for the definition of edge contraction and subdivision.
2) $\mathcal{G}_g$ contains the full grail and all graphs obtained from the full grail via edge contraction. The full grail can be obtained by removing one of the parallel edges between s_2 and v_1, and one of the parallel edges between v_6 and d_2 in Fig. 2(a).

Suppose there exists a network coding solution to the two unicast-session problem. Then one of the following two conditions must hold.

- There exist two EDPs connecting (s_1, t_1) and (s_2, t_2).
- G'' contains an integral subgraph $F = (V^F, E^F)$ such that (i) $\{s_1, s_2, t_1, t_2\} \in V^F$ and (ii) there exists a $G_q \in \mathcal{G}_b \bigcup \mathcal{G}_g$ such that F is a subdivision of G_q. Namely, F can be obtained from G_q by replacing each edge of G_q with an interior-vertex-disjoint path, also known as an *independent path*.

Proposition 1: For F as defined in Theorem 5, if F contains two edges that connect the same pair of vertices, then F contains 2 EDPs connecting (s_1, t_1) and (s_2, t_2)

Proof: F is a subdivision of G_q. If F contains two edges that connect the same pair of vertices then so does G_q. It is easy to check that for all subgraphs in $\mathcal{G}_b$ or in $\mathcal{G}_g$, if there are two edges connecting the same pair of vertices, there exist two edge-disjoint paths connecting (s_1,t_1) and (s_2,t_2). The proof is complete. ■

APPENDIX F
PROOF OF THEOREM 4

Proof: For any PICC if there exist 2EDPs between (s_1,s_2) and (s_1,s_2), we can send packets through these paths and achieve the required rate without network coding, which means that the PICC of interest is redundant. We exploit this observation in the following to show that a PICC is redundant.

If G'' is declared "redundant" by rule 2, we have 2EDPs between (s_1, t_1) and (s_2, t_2), because of the disjointness of $E(P_{s_1,t_1}) \bigcup E(Q_{s_1,t_1})$ and $E(P_{s_2,t_2}) \bigcup E(Q_{s_2,t_2})$. In the following, we will prove rule 3. Without loss of generality, we can use the integral graph G'' to represent the l-th PICC between sessions i and j. G'' always satisfy condition 2 of Theorem 1. If condition 1 is satisfied, there is no need to include G'' in the optimization problem because there are 2 EDPs. Assume condition 1 is not satisfied. We have two cases. Case 1: F in Theorem 5 is the same as G''. By (2), if $H_{ij}^l(e) = 1$, then link e is modelled as two edges connecting the same pair of vertices. G'' contains 2 EDPs by Proposition 1. Case 2: F is a proper subgraph of G''. The edges and vertices in F are subsets of the edges and vertices in G'', and F satisfies the necessary and sufficient conditions for pairwise linear network coding. Therefore, the same solution in G'' can be achieved in F by consuming fewer resources and hence G'' is a redundant PICC. Rule 1 follows by using the same technique. ■

REFERENCES

[1] T. Bonald and L. Massoulie, "Impact of fairness on Internet performance," in *Proc. of ACM Joint International Conference on Measurement and Modeling of Computer Systems (Sigmetrics), Cambridge, MA*, June 2001.

[2] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. on Networking*, vol. 8, no. 5, pp. 556–567, 2000.

[3] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237–252, 1998.

[4] X. Lin and N. B. Shroff, "An optimization based approach for quality-of-service routing in high-bandwidth networks," *IEEE/ACM Trans. on Networking*, vol. 14, no. 6, pp. 1348–1361, 2006.

[5] ——, "Utility maximization for communication networks with multi-path routing," *IEEE Trans on Automatic Control*, vol. 51, no. 5, pp. 766–781, may 2006.

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

[7] R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. on Information Theory*, vol. 49, no. 2, pp. 371–381, 2003.

[8] R. Koetter and M. Medard, "Beyond routing: An algebraic approach to network coding," in *Proc. of IEEE Conference on Computer Communications (INFOCOM), New York*, June 2002.

[9] T. Ho, R. Koetter, M. Medard, M. Effros, J. Shi, and D. Karger, "A random linear network coding approach to multicast," *IEEE Trans. on Information Theory Theory*, vol. 52, no. 10, pp. 4413–4430, october 2006.

[10] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. on Information Theory*, vol. 51, no. 6, pp. 1973–1982, 2005.

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

[12] D. S. Lun, N. Ratnakar, R. Koetter, M. Medard, E. Ahmed, and H. Lee, "Achieving minimum-cost multicast: A decentralized approach based on network coding," in *Proc. of IEEE Conference on Computer Communications (INFOCOM), Miami, FL*, March 2005.

[13] C.-C. Wang and N. B. Shroff, "Beyond the butterfly – a graph-theoretic characterization of the feasibility of network coding with two simple unicast sessions," in *Proc. IEEE Int'l Symp. Inform. Theory*, June 2007.

[14] R. Dougherty, C. Freiling, and K. Zeger, "Insufficiency of linear coding in network information flow," *IEEE Trans. on Information Theory*, vol. 51, no. 8, pp. 2745–2759, 2005.

[15] Z. Li and B. Li, "Network coding: the case of multiple unicast sessions," in *42rd Allerton Conf., Monticello, IL*, Sep 2004.

[16] K. Jain, V. Vazirani, R. Yeung, and G. Yuval, "On the capacity of multiple unicast sessions in undirected graphs," in *Proc. IEEE Int'l Symp. Inform. Theory*, July 2006.

[17] J. Price and T. Javidi, "Network coding games with unicast flows," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 7, pp. 1302–1316, sep 2008.

[18] A. H. Mohsenian-Rad, J. Huang, V. Wong, S. Jaggi, and R. Schober, "A game-theoretic analysis of inter-session network coding," in *Proc. IEEE Int'l Conf. Comm. (ICC), Germany*, June 2009.

[19] M. Kim, M. Medard, U. O'Reilly, and D. Traskov, "An evolutionary approach to inter-session network coding," in *IEEE Conference on Computer Communications (INFOCOM), Rio de Janeiro, Brazil*, April 2009.

[20] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard, and J. Crowcroft, "XORs in the air: Practical wireless network coding," in *Proc. ACM Special Interest Group on Data Commun. (SIGCOMM), Pisa, Italy*, Sept 2006.

[21] S. Sengupta, S. Rayanchu, and S. Banerjee, "An analysis of wireless network coding for unicast sessions: The case for coding-aware routing," in *Proc. of IEEE Conference on Computer Communications (INFOCOM), Anocharage, AK*, May 2007.

[22] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Trans. on Networking*, vol. 14, no. 2, pp. 302–315, 2006.

[23] X. Lin, N. B. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1452–1463, 2006.

[24] D. Traskov, N. Ratnakar, D. S. Lun, R. Koetter, and M. Medard, "Network coding for multiple unicasts: An approach based on linear optimization," in *Proc. IEEE Int'l Symp. Inform. Theory*, 2006.

[25] A. Eryilmaz and D. S. Lun, "Control for inter-session network coding," in *Proc. of Workshop on Network Coding, Theory, & Applications (NetCod), San Diego*, Jan 2007.

[26] T. Ho, Y. Chang, and K. J. Han, "On constructive network coding for multiple unicasts," in *44th Allerton Conf., Monticello, IL*, Sept 2006.

[27] C.-C. Wang and N. B. Shroff, "Intersession network coding for two simple multicast sessions," in *45th Allerton Conf., Monticello, IL*, Sep 2007.

[28] L. Chen, T. Ho, S. H. Low, M. Chiang, and J. C. Doyle, "Optimization based rate control for multicast with network coding," in *Proc. of IEEE Conference on Computer Communications (INFOCOM), Anocharage, AK*, May 2007.

[29] Y. Wu and S.-Y. Kung, "Distributed utility maximization for network coding based multicasting: a shortest path approach," *IEEE J. on Selected Areas in Communications*, vol. 24, no. 8, pp. 1475–1488, Apr 2006.

[30] Y. Wu, M. Chiang, and S.-Y. Kung, "Distributed utility maximization for network coding based multicasting: a critical cut approach," in *Proc. Workshop on Network Coding, Theory, & Applications (NetCod)*, Apr 2006.

[31] Y. Xi and E. M. Yeh, "Distributed algorithms for minimum cost multicast with network coding in wireless networks," in *Second Workshop on Network Coding, Theory, and Applications, Boston, MA*, April 2006.

[32] ——, "Distributed algorithms for minimum cost multicast with network coding," in *43rd Allerton Conf., Monticello, IL*, Sep 2005.

[33] D. Bertsekas and J. N. Tsitsikalis, *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[34] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[35] J. Chen, P. Drushel, and D. Subramanian, "An efficient multi-path forwarding method," in *IEEE Conference on Computer Communications (INFOCOM)*, March 1998.

[36] W. T. Zaumen and J. J. Garcia-Luna-Aceves, "Loop-free multipath routing using generalized diffusing computations," in *Proc. of IEEE Conference on Computer Communications (INFOCOM), San Francisco, CA*, March 1998.

[37] P. Chou, Y. Wu, , and K. Jain, "Practical network coding," in *Proc. of 41st Allerton Conf., Monticello, IL*, October 2003.

[38] J. W. Lee, R. R. Mazumdar, and N. B. Shroff, "Non-convex optimization and rate control for multi-class services in the Internet," *IEEE/ACM Trans. on Networking*, vol. 13, no. 4, pp. 827–840, Aug 2005.

[39] R. Diestel, *Graph Theory*. New York: Springer-Verlag, 2005.

Abdallah Khreishah received the B.S. degree from Jordan University of Science and Technology (JUST), Irbid, Jordan in 2004, and the M.S. degree from the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, in 2006. He is currently working toward the Ph.D. degree at Purdue University. His research interests include network coding, congestion control, and cross layer design in wireless networks.

Chih-Chun Wang joined the School of Electrical and Computer Engineering in January 2006 as an Assistant Professor. 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 during in 2000 and spent the summer of 2004 with Flarion Technologies, New Jersey. In 2005, he held a post-doc position in the Electrical Engineering Department of Princeton University. He received the National Science Foundation Faculty Early Career Development (CAREER) Award in 2009. His current research interests are in the graph-theoretic and algorithmic analysis of iterative decoding and of network coding. Other research interests of his fall in the general areas of optimal control, information theory, detection theory, coding theory, iterative decoding algorithms, and network coding.

Ness B. Shroff received his Ph.D. degree from Columbia University, NY, in 1994 and joined Purdue university as an Assistant Professor. At Purdue, he became Professor of the School of Electrical and Computer Engineering in 2003 and director of CWSA in 2004, a university-wide center on wireless systems and applications. In July 2007, he joined The Ohio State University as the Ohio Eminent Scholar of Networking and Communications, a chaired Professor of ECE and CSE. His research interests span the areas of wireless and wireline communication networks. He is especially interested in fundamental problems in the design, performance, pricing, and security of these networks.

Dr. Shroff is a past editor for IEEE/ACM Trans. on Networking and the IEEE Communications Letters and current editor of the Computer Networks Journal. He has served as the technical program co-chair and general co-chair of several major conferences and workshops, such as the IEEE INFOCOM 2003, ACM Mobihoc 2008, IEEE CCW 1999, and WICON 2008. He was also a co-organizer of the NSF workshop on Fundamental Research in Networking, held in Arlie House Virginia, in 2003.

Dr. Shroff is a fellow of the IEEE. He received the IEEE INFOCOM 2008 best paper award, the IEEE INFOCOM 2006 best paper award, the IEEE IWQoS 2006 best student paper award, the 2005 best paper of the year award for the Journal of Communications and Networking, the 2003 best paper of the year award for Computer Networks, and the NSF CAREER award in 1996 (his INFOCOM 2005 paper was also selected as one of two runner-up papers for the best paper award).