Control for Inter-session Network Coding

Atilla Eryilmaz[†]
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
E-mail: eryilmaz@mit.edu

Desmond S. Lun[†]
The Broad Institute of MIT and Harvard
Cambridge, MA 02142, USA
E-mail: dslun@mit.edu

***Abstract*— We propose a dynamic routing-scheduling-coding strategy for serving multiple unicast sessions when linear network coding is allowed across sessions. Noting that the set of stabilizable throughput levels in this context is an open problem, we prove that our strategy supports any point in the non-trivial region of achievable rates recently characterized by Traskov et al. [1]. This work also provides a theoretical framework in which the gains of intersession network coding and pure routing can be compared.**

I. Introduction

Coding in packet networks can be classified into two types: intra-session coding (where coding is restricted to packets belonging to the same session or connection) and inter-session coding (where this restriction is lifted and coding is allowed among packets belonging to possibly different sessions). The former, which is also referred to as superposition coding [2], has been extensively studied. It is well-known that intra-session coding improves the throughput of lossless multicast sessions (see, for example, [3], [4], [5]) and of lossy sessions—unicast or multicast (see, for example, [6], [7]). It is also known, however, that intra-session coding is suboptimal [2]: *inter*-session coding is necessary to achieve optimal throughput in general.

Unfortunately, performing inter-session coding is difficult. To perform inter-session coding optimally, linear coding operations are not sufficient [8], and, even if we limit ourselves to a particular class of linear coding operations, deciding what operations to perform is an NP-hard problem [4]. This situation motivates us to develop methods for inter-session coding that, though not optimal, achieve significant throughput gains over intra-session coding. There is good reason to believe that such gains can be found without the use of sophisticated inter-session coding—a simulation study and testbed implementation by Katti et al. [9], [10] found significant throughput gains for multi-hop wireless networks with a rudimentary inter-session coding scheme that generalizes the "physical piggybacking" discussed in [11].

This paper continues the work along the line of suboptimal, yet improved, methods for inter-session coding, which includes [12], [13], [1], [14]. The defining characteristic of this paper is that, rather than proposing an algorithm that operates on given flow rates (or ones it measures), we propose a dynamic routing-scheduling-coding strategy that operates solely on queue-state information. Thus, although the algorithm described in [14] (which is the result of independent work by Ho et al.) bears some similarities to our strategy, it nevertheless differs in this defining aspect. Dynamic strategies such as ours do not require flow rates as an input and can be run "on-line". They will generally take some time to find the desired operating point, but they are robust to dynamics because they react to present circumstances as measured by the state of the queues.

Our strategy extends that of Katti et al. [9], [10] and can be seen, moreover, as an extension of the dynamic routing-scheduling strategies of Tassiulas and Ephremides [15], and others (e.g. [16], [17], [18], [19], [20]), which do not allow for coding, and of the dynamic routing-scheduling-coding strategy of Ho and Viswanathan [21], which allows for only intra-session coding. It is, however, not a straightforward extension of these dynamic strategies: as will become apparent, allowing for inter-session coding requires an approach with significant differences. Our main result is that our strategy stably supports any throughput that lies strictly within the non-trivial region of achievable rates for multiple unicast sessions given by Traskov et al. [1]—a region that we refer to as the TRLKM region.

Our strategy applies to both lossless wireline and lossless wireless networks. We model a wireline network as a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{E}$ is a set of directed edges that represent point-to-point links. We model a wireless network as a directed hypergraph $\mathcal{H} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{E}$ is a set of directed hyperedges[1] that represent broadcast links. We suppose that the set of achievable link rates of the network is represented by the set Γ. If $\nu_{(m,n)}$ is the rate at which packets are injected on edge e, then the vector ν, consisting of $\nu_{(m,n)}$, $(m,n) \in \mathcal{E}$, represents a set of injection rates within the capacity of the network only if $\nu \in \Gamma$. For wireline networks, it is generally the case that the capacity of separate links are independent, and Γ is the Cartesian product of $|\mathcal{E}|$ closed intervals, each extending from 0 to some non-negative capacity $\gamma_{(m,n)}$. For wireless networks, however, this is generally not the case,

This work was supported by the Control-Based Mobile Ad-Hoc Networking (CBMANET) Program under DARPA subcontract no. 060786 and by the National Science Foundation under grant no. CCR-0325673.

[†]Author names are given in alphabetical order.

[1]A hyperedge is a generalization of an edge that starts at a single node and ends at possibly more than one node.

and the capacities of separate links are generally dependent because of interference. In this paper, we present only the wireline case in detail. The wireless case is conceptually similar. Readers interested in the details of the wireless case are referred to [22].

II. The Butterfly Network Case

We first describe our algorithm for the well-known butterfly network. We take this approach to explain the essential components of, and give general intuition for, our algorithm without complicated notation. In Section III, we extend our results to general wireline networks and give an algorithm that stabilizes all rates in the TRLKM region. The TRLKM region is essentially obtained by decomposing a general wireline network into superimposed butterfly networks and thus, though Section III appears much more complicated, the conceptual extension that is required is minor.

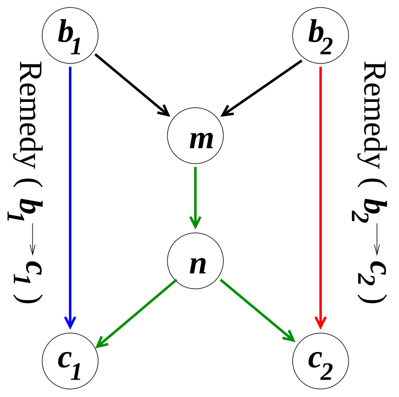
b_1
b_2
m
n
c_1
c_2
Remedy ($b_1 \rightarrow c_1$)
Remedy ($b_2 \rightarrow c_2$)

Fig. 1. The butterfly network with two unicast flows: Flow-f, from b_1 to c_2; Flow-g, from b_2 to c_1.

Consider the butterfly network shown in Figure 1. The system operates in equal length time slots. Suppose there are two unicast flows: Flow-f from node b_1 to c_2, and Flow-g from node b_2 to c_1. The exogenous arrivals for these flows have mean rates $\lambda^{(f)}$ and $\lambda^{(g)}$ packets/slot, respectively. The only other constraint on the arrival processes is that they have finite second moments. The capacity of the links can be asymmetric and randomly varying in time. We let $\gamma_{(m,n)}$ denote the average link rate for link $(m,n) \in \mathcal{E}$. The statistics of the arrival or link quality are not known. The only required knowledge is that of the link state of incident links at each node.

Without network coding, the set of achievable $(\lambda^{(f)}, \lambda^{(g)})$ is clear:

Definition 1 (Butterfly Capacity Region with Routing): With routing (i.e., without coding), we can achieve all $(\lambda^{(f)}, \lambda^{(g)})$ that satisfy

$$\lambda^{(f)} \leq \min(\gamma_{(b_1,m)}, \gamma_{(k,c_2)}), \quad (1)$$

$$\lambda^{(g)} \leq \min(\gamma_{(b_2,m)}, \gamma_{(k,c_1)}), \quad (2)$$

$$\lambda^{(f)} + \lambda^{(g)} \leq \gamma_{(m,n)}, \quad (3)$$

$$\lambda \geq 0. \quad (4)$$

When network coding is allowed, the set of achievable $(\lambda^{(f)}, \lambda^{(g)})$ increases. In particular, we can, at node m, take one packet from Flow-f and one packet from Flow-g, XOR[2] them together into a coded packet, and multicast this coded packet to both nodes c_1 and c_2. To allow the coded packet to be decoded, we require two *remedy* packets to be sent, one from b_1 to c_1, and one from b_2 to c_2. Let $\lambda^{(f,g)}$ be the rate at which packets are coded in this way. Then, we can define the capacity region with coding as follows.

Definition 2 (Butterfly Capacity Region with Coding): Recalling that $\lambda^{(f,g)}$ denotes the rate of coded packets, with coding, we can achieve all $(\lambda^{(f)}, \lambda^{(g)})$ that satisfy, for some $\lambda^{(f,g)}$, (1), (2), (4), and

$$\lambda^{(f)} + \lambda^{(g)} - \lambda^{(f,g)} \leq \gamma_{(m,n)}, \quad (5)$$

$$\lambda^{(f,g)} \leq \gamma_{(b_1,c_1)}, \quad (6)$$

$$\lambda^{(f,g)} \leq \gamma_{(b_2,c_2)}, \quad (7)$$

$$0 \leq \lambda^{(f,g)} \leq \min(\lambda^{(f)}, \lambda^{(g)}). \quad (8)$$

It is not difficult to see that this region cannot be increased any further and, thus, the achievable region for $(\lambda^{(f)}, \lambda^{(g)})$ is in fact the capacity region.

If both the exogenous arrival rates, $\lambda^{(f)}$ and $\lambda^{(g)}$, are known, then, at some centralized point with complete network information, $\lambda^{(f,g)}$ can be found for any $(\lambda^{(f)}, \lambda^{(g)})$ in the capacity region, thus allowing the rate pair to be achieved. In many network settings, however, $\lambda^{(f)}$ and $\lambda^{(g)}$ are not known, and we moreover do not have a centralized point with complete network information. Also, the arrivals and link states are stochastically varying. Therefore, we wish to make decisions on coding, routing, and scheduling on-the-fly in a decentralized way. This is the type of dynamic policy that we seek.

Intuitively, a good place to make the coding decision is at node m. If node m observes that its instantaneous packet queue has many packets queued for Flow-f and many packets queued for Flow-g, then it is likely that node m represents a bottleneck. In this case, we could alleviate the congestion at node m by coding, which introduces remedy packets at node b_1 and b_2. We assume that node m is capable of sending small, *remedy request*, protocol messages[3] to nodes b_1 and b_2, requesting that these additional packets be sent. If the links (b_1, c_1) and (b_2, c_2) are themselves congested, however, it may not be a good idea for node m to code. So it is not clear what the decision rule must be to exploit the network coding advantage while guaranteeing decodability at the receivers.

In this section, we give a dynamic policy that yields coding decisions based on the occupancies of neighboring queues. These queue-lengths must be maintained so that they serve as a measure of decodability of the coded packets. Let $Q_k^{(d)}[t]$ denote the length of the queue at the beginning of slot t, maintained at node k, holding packets destined for node d.

[2]The XOR operation is performed for each aligned bit pair in the two packets.

[3]Note that these messages are simple signals much shorter than packet lengths. We assume that their consumption of link capacity is negligible.

In addition, let $Q_k^{(c_1,\{c_1,c_2\})}[t]$ denote the number of coded packets at node k that are destined for node c_1. At each time slot, Flow-f packets arrive at node b_1 and are placed into queue $Q_{b_1}^{(c_2)}$, and Flow-g packets arrive at node b_2 and are placed into queue $Q_{b_2}^{(c_1)}$.

We consider each of the nodes in turn.

• Node b_1 maintains two queues, $Q_{b_1}^{(c_1)}$ and $Q_{b_1}^{(c_2)}$, and its policy is straightforward: At each time slot, it uses whatever capacity is available on link (b_1, m) to serve $Q_{b_1}^{(c_2)}$, removing served packets from the queue and placing them into $Q_m^{(c_2)}$, and it uses whatever capacity is available on link (b_1, c_1) to serve $Q_{b_1}^{(c_1)}$, removing served packets from the queue, which then reach their destination. The situation at node b_2 is similar to that at node b_1.

• Node n maintains four queues, $Q_n^{(c_1)}$, $Q_n^{(c_2)}$, $Q_n^{(c_1,\{c_1,c_2\})}$, and $Q_n^{(c_2,\{c_1,c_2\})}$. It checks to see if $Q_n^{(c_1)}$ or $Q_n^{(c_1,\{c_1,c_2\})}$ is greater, and serves the greater of the two using whatever capacity it has on link (n, c_1); likewise, it checks to see if $Q_n^{(c_2)}$ or $Q_n^{(c_2,\{c_1,c_2\})}$ is greater, and serves the greater of the two using whatever capacity it has on link (n, c_2). Nodes c_1 and c_2 are final destination nodes and do not maintain queues.

• The coding decision of node m is based on

$$\begin{aligned}
\rho_{(m,n)}^{(c_1)}[t] &\triangleq \left(Q_m^{(c_1)}[t] - Q_n^{(c_1)}[t]\right)^+, \\
\rho_{(m,n)}^{(c_2)}[t] &\triangleq \left(Q_m^{(c_2)}[t] - Q_n^{(c_2)}[t]\right)^+, \\
\sigma_{(m,n)}^{(\{c_1,c_2\})}[t] &\triangleq Q_m^{(c_1)}[t] - (Q_n^{(c_1,\{c_1,c_2\})}[t] + Q_{b_1}^{(c_1)}[t]) \\
&\quad + Q_m^{(c_2)}[t] - (Q_n^{(c_2,\{c_1,c_2\})}[t] + Q_{b_2}^{(c_2)}[t]),
\end{aligned}$$

where $(y)^+ \triangleq \max(0, y)$. If $\sigma_{(m,n)}^{(\{c_1,c_2\})}[t]$ is greater than $\max(\rho_{(m,n)}^{(c_1)}[t], \rho_{(m,n)}^{(c_2)}[t])$, then coding is performed: Node m removes one packet from $Q_m^{(c_1)}$ and one packet from $Q_m^{(c_2)}$, forms a single coded packet from the XOR of the two, and transmits the coded packet on link (m, n). Upon reception at node n, the coded packet is placed into both queues $Q_n^{(c_1,\{c_1,c_2\})}$ and $Q_n^{(c_2,\{c_1,c_2\})}$. As well as transmitting the coded packet on link (m, n), node m transmits two remedy request protocol messages, one to b_1 and one to b_2. These remedy request protocol messages ultimately result in a remedy packet being placed into each queue $Q_{b_1}^{(c_1)}$ and queue $Q_{b_2}^{(c_2)}$. Node m repeatedly forms coded packets and sends remedy request protocol messages for them for as much capacity is available on link (m, n) in time slot t.

If either $\rho_{(m,n)}^{(c_1)}[t]$ or $\rho_{(m.n)}^{(c_2)}[t]$ is greater than $\sigma_{(m,n)}^{(\{c_1,c_2\})}[t]$, then coding is not performed; rather, if $\rho_{(m,n)}^{(c_1)}[t] > \rho_{(m,n)}^{(c_2)}[t]$, then $Q_m^{(c_1)}$ is served using all the available capacity of link (m, n), otherwise $Q_m^{(c_2)}$ is served using all the available capacity of link (m, n). ⋄

We can understand the policy employed on node m as an extension of *differential backlog* (see [15], [19], [18]): $\rho_{(m,n)}^{(c_1)}$ and $\rho_{(m,n)}^{(c_2)}$ give the traditional differential backlog associated with c_1 and c_2, respectively, and the factor $\sigma_{(m,n)}^{(\{c_1,c_2\})}$ represents the differential backlog associated with coding. To calculate the latter correctly, we need to account for the following two effects of coding: first, by coding, we effectively serve two packets for the price of one, removing a packet from both $Q_m^{(c_1)}$ and $Q_m^{(c_2)}$ while transmitting only a single packet on link (m, n); second, we have to pay for this advantage of coding with remedy packets, which create packets in $Q_{b_1}^{(c_1)}$ and $Q_{b_2}^{(c_2)}$, one for each flow. The first effect causes $Q_m^{(c_1)}[t]-Q_n^{(c_1,\{c_1,c_2\})}$ to be summed with $Q_m^{(c_2)}[t] - Q_n^{(c_2,\{c_1,c_2\})}$ when calculating the differential backlog, and the second effect causes $Q_{b_1}^{(c_1}[t]$ and $Q_{b_2}^{(c_2)}[t]$ to be subtracted from the differential backlog, finally yielding $\sigma_{(m,n)}^{(\{c_1,c_2\})}$ as the correct differential backlog associated with coding.

Our main result, in Section III-C, states that the policy we describe above will stabilize all exogenous arrival rates, $\lambda^{(f)}$ and $\lambda^{(g)}$, that lie strictly in the interior of the capacity region given by Definition 2

III. Extension to general wireline networks

In the previous section, we have described a dynamic, decentralized policy that achieves all exogenous arrival rates that lie strictly in the interior of the capacity region for the butterfly network. In this section, we extend the algorithm to be implemented in more general networks. For more general wireline networks, we can consider superimposing or overlaying butterfly networks into the network to extend the butterfly network case. In general, this kind of superimposing of butterfly networks will not achieve the capacity region, but it will at least expand the region that is achievable by routing. The region that can be achieved in this way has been established by Traskov et al. [1], and we refer to the region as the TRLKM region.

In this paper, we refrain from discussing the TRLKM region in any depth. We note that it essentially considers all possible ways in which butterfly networks can appear in a general wireline network, and, for each butterfly network, it allows a coded packet to be transmitted on the center link (or path) as long as remedy packets are transmitted on the side links (or paths). We refer the reader to [22] for the details of the region as expressed using our notation.

A. System model

We can model any wireline network using the graphical model $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, with $\mathcal{N}$ and $\mathcal{E}$ representing the set of nodes and edges, respectively. We consider unicast flows that are described by a pair of beginning-end nodes of the flow with no a priori fixed routes. As before, we let $\lambda^{(f)}$ be the average number of exogenous packets that enter the network for Flow-f. The algorithm we propose makes routing, scheduling and *coding* decisions based on properly maintained buffer occupancy levels. In the algorithm, each node proactively seeks opportunities to create coded packets by generating remedy packets elsewhere in the network.

As discussed in Section II, when packets of two flows are linearly coded at a node, a *remedy packet* must be generated for each flow at another node in the network. Such remedy packets are needed because, from the perspective of one of the coded flows, the coded packet is *poisoned* by the other flow. Thus, in order to extract the desired packet at a future node, a remedy must also be sent to it. The remedy packet can only be generated at one of the nodes that the coded packet had traversed. This implies that even after the transmission of packets, nodes need to store them for a while at the possibility of a future remedy transmission request. This can be achieved by maintaining a finite size memory for this purpose. Another practical consideration is for the coding node to know which nodes have the remedy packet. This issue can be resolved by including in each packet the set of nodes that are capable of generating it.

In addition to the remedy packet generation, the coding node is also allowed to choose the *decoding nodes* for the coding being performed by it. In particular, if flows f and g, coded at node m, are to be decoded at nodes c_1 and c_2 with remedy packets generated at nodes b_1 and b_2, then we can view the system to be composed of one *multicast* session generated at node m with receivers c_1 and c_2, and two unicast sessions for the remedy flows, one from b_1 to c_1 and the other from b_2 to c_2 (see Figure 2).

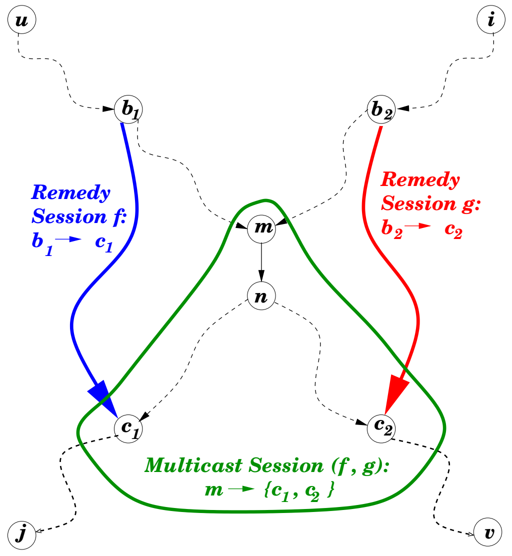
u
i
b1
b2
Remedy Session f: b1 → c1
Remedy Session g: b2 → c2
m
n
c1
c2
Multicast Session (f, g): m → {c1, c2}
j
v

Fig. 2. Flow-f goes from u to v and Flow-g goes from i to j, both traversing link (m, n). The dashed lines indicate paths composed of multiple links. A decision to perform inter-session coding across flows f and g at node m with remedy nodes (b_1, b_2) and decoding nodes (c_1, c_2) results in: two unicast sessions for remedy packets $[b_1 \to c_1$ and $b_2 \to c_2]$; and one multicast session $[m \to \{c_1, c_2\}]$.

The above multicast session is allowed to be coded again in its path. Thus, in general, coding can be performed across two multicast sessions destined for the set of nodes D_1 and D_2, respectively. A packet is said to be of *type* D if it is part of a multicast session destined for the set of nodes in D.

We maintain a separate queue for each type of packet and for each receiver expecting that type of packet. In particular, we let $Q_n^{(d,D)}[t]$ be the length, at the beginning of time slot t, of the queue at node n, holding packets of type D with destination $d \in D$. The coding service, which applies to packets of two types simultaneously, is made available by the sending of packets formed from the XOR of one packet of type D_1 with one packet of type D_2; remedy packets are injected at nodes b_1 and b_2 and decoding is done at nodes c_1 and c_2 as shown in Figure 2.

The precise evolution of queues is determined by the routing-scheduling-coding strategy described in the next section. Due to space constraints, we leave the details of this evolution to the extended version of the paper [22].

B. The Routing-Scheduling-Coding Strategy

The purpose of the strategy is to decide whether coding is to be performed across sessions and, if so, which sessions to code; if not, which queues to serve. We now describe the operation of the strategy in greater detail.

*Definition 3 (Routing-Scheduling-Coding (*RSC*) Algorithm):* At every time slot t, for each link $(m, n) \in \mathcal{E}$, the following two sets of weights, one corresponding to intra-session and the other corresponding to inter-session coded packets, are computed at node m:

$$\rho_{(m,n)}^{(D)}[t] \triangleq \sum_{d \in D} \left(Q_m^{(d,D)}[t] - Q_n^{(d,D)}[t] \right)^+ ,$$

$$\begin{aligned} &\sigma_{(m,n)}^{((D_1,b_1,c_1),(D_2,b_2,c_2))}[t] \\ &\quad \triangleq \sum_{d \in D_1} \left(Q_m^{(d,D_1)}[t] - Q_{c_1}^{(d,D_1)}[t] \right)^+ - Q_n^{(c_1,\{c_1,c_2\})}[t] \quad (9) \\ &\quad + \sum_{d \in D_2} \left(Q_m^{(d,D_2)}[t] - Q_{c_2}^{(d,D_2)}[t] \right)^+ - Q_n^{(c_2,\{c_1,c_2\})}[t] \\ &\quad \quad (10) \\ &\quad - Q_{b_1}^{(c_1)}[t] - Q_{b_2}^{(c_2)}[t], \quad (11) \end{aligned}$$

where we write $Q_n^{(d)}[t]$ as shorthand for $Q_n^{(d,\{d\})}[t]$. Here, $\rho_{(m,n)}^{(D)}[t]$ represents the weight associated with serving packets of type D over link (m, n) without coding across sessions, while $\sigma_{(m,n)}^{((D_1,b_1,c_1),(D_2,b_2,c_2))}[t]$ is the weight associated with coding packets of type D_1 and D_2 with remedies created at b_1, b_2, and decoding to be performed at c_1, c_2.

Once these weights are computed, the maximizing D for $\rho_{(m,n)}^{(D)}[t]$, denoted by $D_{(m,n)}^\star[t]$, and the maximizing $((D_1, b_1, c_1), (D_2, b_2, c_2))$ for $\sigma_{(m,n)}^{((D_1,b_1,c_1),(D_2,b_2,c_2))}[t]$, denoted by $((D_1^\star, b_1^\star, c_1^\star), (D_2^\star, b_2^\star, c_2^\star))_{(m,n)}[t]$, are computed. We maximize over $b \in B_1$ and $b \in B_2$, where B_1 and B_2 are chosen using the information carried by the candidate packets for coding to ensure that appropriate remedy packets can be generated at b_1 and b_2. We let

$$\sigma_{(m,n)}^\star[t] \triangleq \sigma_{(m,n)}^{((D_1^\star,b_1^\star,c_1^\star),(D_2^\star,b_2^\star,c_2^\star))_{(m,n)}[t]}[t],$$

$$\rho_{(m,n)}^\star[t] \triangleq \rho_{(m,n)}^{(D_{(m,n)}^\star[t])}[t],$$

which represent the weights associated with the best decisions with and without inter-session coding, respectively. The final decision is performed based on the comparison between

$\rho^\star_{(m,n)}[t]$ and $\sigma^\star_{(m,n)}[t]$:

• if $\rho^\star_{(m,n)}[t] > \sigma^\star_{(m,n)}[t]$, then no inter-session coding is performed at node m, and random intra-session coding is performed only within packets of session $D^\star_{(m,n)}[t]$ as follows: the head of the line packets for those $d \in D^\star_{(m,n)}$ satisfying

$$\left(Q_n^{(d,D^\star_{(m,n)})}[t] - Q_m^{(d,D^\star_{(m,n)})}[t]\right) \quad > \quad 0 \qquad (12)$$

are removed from their queues at node n and linearly combined with random coefficients. Then, the resulting randomly generated packet is transmitted over link (m,n) along with the random coefficients and enqueued at each of the queues at node m for which (12) was positive.

• if $\rho^\star_{(m,n)}[t] < \sigma^\star_{(m,n)}[t]$, then inter-session coding is to be performed at node m across packets of type $D^\star_1$ and $D^\star_2$ through the following steps:

(1) *Intra-session Coding:* For each $i = 1, 2$, the head of the line packets for those $d \in D^\star_i$ satisfying $\left(Q_m^{(d,D^\star_i)}[t] - Q_{c^\star_i}^{(d,D^\star_i)}[t]\right) > 0$ are removed from their corresponding queues at node m and are linearly combined with random coefficients.

(2) *Inter-session Coding:* The inter-session coding is performed by adding (XORing) the packets generated by the previous intra-session coding operation. Then, the final coded packet is transmitted over link (m,n) and is enqueued at both $Q_n^{(c_1,\{c_1,c_2\})}$, and $Q_n^{(c_2,\{c_1,c_2\})}$.

(3) *Remedy Packet Generation:* Upon coding operation, remedy request protocol messages are transmitted to nodes $b^\star_1$ and $b^\star_2$, which in turn reproduce packets [generated during Step (1)] of type $D^\star_2$ and $D^\star_1$, respectively, and then enqueue them at $Q_{b^\star_1}^{(c^\star_1)}$ and $Q_{b^\star_2}^{(c^\star_2)}$ for transmission to $c^\star_1$ and $c^\star_2$, respectively.

• if $\rho^\star_{(m,n)}[t] = \sigma^\star_{(m,n)}[t]$, then choose randomly, with equal probabilities, one of the above two modes of operations and implement it.

This completes the description of the RSC algorithm. $\diamond$

Remark 1: The term $\rho^{(D)}_{(m,n)}[t]$ is a generalization of the concept of *differential backlog* introduced in [15] to multicast sessions. This term dynamically establishes routes by steering packets in the largest differential backlog direction. The same term appears in a recent work in [21], where the authors study routing-scheduling-coding strategies for multicast sessions without inter-session coding. However, $\sigma^{((D_1,b_1,c_1),(D_2,b_2,c_2))}_{(m,n)}[t]$ is introduced for the first time in this work and has a form that provides significant insight. In particular, it contains two differential backlog terms between the coding node m and the decoding nodes c_1 and c_2, (see (9) and (10)) along with the occupancy at the neighboring node, n. Thus, these terms intuitively measure the decodability at c_1 and c_2. Also, (11) includes the occupancy level of the nodes with remedy packets. Thus, if these are high, it implies that the remedies cannot reach the decoding nodes efficiently and the coding decision must be discouraged.

Remark 2: The decodability of the packets formed by intra-session coding is shown in [21]. That the packets formed by inter-session coding are decodable is established by the way in which the inter-session coding decision is made. When two packets, x_1 and x_2, are XORed together at node m, x_1 must be available at node b_1 and x_2 must be available at node b_2. Provided that x_1 is eventually communicated to node c_1 and x_2 is eventually communicated to node c_2, which is ensured by the setting up of the two remedy sessions, the inter-session coding operation can be undone at node c_1 (to recover x_2) and at node c_2 (to recover x_1).

Remark 3: The policy requires the knowledge of the occupancy levels of those nodes at which decoding and remedy packet generation is to be performed. In practice, such information may be available only for those nodes in a local neighborhood of each node. Although the performance of the policy will improve as the span of this information increases, it has been observed in empirical studies [9], [10] that even a one-hop neighborhood knowledge improves the achievable throughput considerably. Our model is general enough to accommodate the extreme scenarios of more practical implementation with weaker but still good performance, and less practical implementation with better performance.

C. Analysis

The tight connection between the stability of stochastic networks and the associated fluid models has been observed in many works [23], [24], [25], [18], [26]. The study of fluid models not only simplifies the analysis, but also facilitates the understanding of the main behavior of the system. In this light, we also use a heuristic fluid model of our network under the RSC policy, and then study the performance of the resulting network. We use tools from Lyapunov Stability Theory [27] to prove the global asymptotic stability of the fluid model for any throughput vector lying within the TRLKM region. This finding, in turn, suggests the stability of the original system.

In the fluid model: the discrete-time parameter, $[t]$, is replaced with the continuous-time parameter, (t); the stochastic arrival and link state processes are replaced with their means; and the discrete-time queue-length evolution of the stochastic system is replaced by a differential equation. We use $\mathbf{q}$ to indicate queue-length vector in the fluid model. Then, we have the following result.

Theorem 1: For any $\varepsilon > 0$, if the flow rates $\{\lambda^{(f)}\}_f$ are such that $\{\lambda^{(f)} + \varepsilon\}_f$ lies in the TRLKM region, then the RSC algorithm is globally asymptotically stable, i.e., for any $\mathbf{q}(0)$, $\mathbf{q}(t) \to 0$ as $t \to \infty$.

Proof: The proof uses Lyapunov Stability Theorem and LaSalle's invariance principle along with the description of the TRLKM region to show the stability of the network under the RSC Algorithm. We omit the details due to space limitations. They can be found in [22]. ■

Under the assumption of finite variance of exogenous arrival processes, it is possible to utilize the result of Theorem 1 to prove positive recurrence and stability of the Markov chain $\{\mathbf{Q}[t]\}_{t\geq 0}$ (see, for example, [18], [28]). This states that the RSC Algorithm stabilizes the queues for any arrival rate λ that lies in the interior of the TRLKM region.

IV. Conclusion

In this paper, we have introduced a dynamic routing-scheduling-coding strategy for inter-session network coding, which can be seen both as a generalization of dynamic routing-scheduling strategies based on differential backlog (e.g. [15], [19], [18]) and as a generalization of the 2-way coding policy of Katti et al. [9]. Our strategy decides whether two previously-independent flows should be coded together at a node and, if so, which flows. We did not consider allowing for coding operations that involve more than two flows, but generalizing the ideas of this paper to allow for such coding operations is, at least conceptually, not difficult.

Our main result was to show that this strategy, the RSC algorithm, stably supports any throughput that lies strictly within the TRLKM region. This result is conservative. The RSC algorithm should support rates outside the TRLKM region because the RSC algorithm allows for a wider range of coding operations to be performed. In particular, it allows for a coded packet to be coded again at a downstream node. We do not in fact know the stability region of the RSC algorithm, and we suspect that it cannot be easily characterized.

But uncharacterized rate regions may simply have to be accepted to proceed with inter-session coding in a meaningful way. It is well-known that the general rate region for inter-session coding is very difficult to characterize [8], [29] and attempts to describe rate regions, such as the TRLKM region, have not yielded the gains observed in empirical studies (e.g., [9], [10]). In this work, we have described a policy rather than a region, and it may be that any accurate characterization of achievable rates—especially in scenarios pertinent to practice—will have to come from measurements. Nevertheless, the RSC algorithm is grounded in a solid theoretical framework and, since it is a generalization of the 2-way coding policy of Katti et al., it should perform at least as well; measurements that describe its true capabilities have yet to be performed.

Acknowledgments

The authors would like to thank Ralf Koetter, Muriel Médard, Asuman Ozdaglar, and Edmund Yeh for helpful discussions and comments.

References

[1] D. Traskov, N. Ratnakar, D. S. Lun, R. Koetter, and M. Médard, “Network coding for multiple unicasts: An approach based on linear optimization,” in *Proc. IEEE International Symposium on Information Theory (ISIT)*, July 2006, pp. 1758–1762.

[2] R. W. Yeung, “Multilevel diversity coding with distortion,” *IEEE Trans. Inform. Theory*, vol. 41, no. 2, pp. 412–422, Mar. 1995.

[3] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, “Network information flow,” *IEEE Trans. Inform. Theory*, vol. 46, no. 4, pp. 1204–1216, July 2000.

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

[5] S. Deb, M. Effros, T. Ho, D. R. Karger, R. Koetter, D. S. Lun, M. Médard, and N. Ratnakar, “Network coding for wireless applications: A brief tutorial,” in *Proc. International Workshop on Wireless Ad-hoc Networks (IWWAN) 2005*, May 2005, invited paper.

[6] A. F. Dana, R. Gowaikar, R. Palanki, B. Hassibi, and M. Effros, “Capacity of wireless erasure networks,” *IEEE Trans. Inform. Theory*, vol. 52, no. 3, pp. 789–804, Mar. 2006.

[7] D. S. Lun, M. Médard, R. Koetter, and M. Effros, “Further results on coding for reliable communication over packet networks,” in *Proc. IEEE International Symposium on Information Theory (ISIT)*, Sept. 2005, pp. 1848–1852.

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

[9] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Médard, “The importance of being opportunistic: Practical network coding for wireless environments,” in *Proc. 43rd Annual Allerton Conference on Communication, Control, and Computing*, Sept. 2005.

[10] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, “Network coding made practical,” MIT CSAIL, Technical Report 2006-009, Feb. 2006.

[11] Y. Wu, P. A. Chou, and S.-Y. Kung, “Information exchange in wireless networks with network coding and physical-layer broadcast,” in *Proc. Conference on Information Sciences and Systems (CISS)*, 2005.

[12] D. S. Lun, M. Médard, T. Ho, and R. Koetter, “Network coding with a cost criterion,” in *Proc. International Symposium on Information Theory and its Applications (ISITA)*, Oct. 2004, pp. 1232–1237.

[13] N. Ratnakar, D. Traskov, and R. Koetter, “Approaches to network coding for multiple unicasts,” in *Proc. International Zurich Seminar on Communications (IZS)*, 2006, pp. 70–73, invited paper.

[14] T. Ho, Y.-H. Chang, and K. Han, “On constructive network coding for multiple unicasts,” in *Proc. 44th Annual Allerton Conference on Communication, Control, and Computing*, Sept. 2006.

[15] L. Tassiulas and A. F. Ephremides, “Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks,” *IEEE Trans. Automat. Contr.*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.

[16] M. Andrews, K. Kumaran, K.Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting, “Providing quality of service over a shared wireless link,” February 2001.

[17] S. Shakkottai and A. Stolyar, “Scheduling for multiple flows sharing a time-varying channel: The exponential rule,” *Translations of the AMS*, A volume in memory of F. Karpelevich, vol. 207, pp. 185–202, 2002.

[18] A. Eryilmaz, R. Srikant, and J. R. Perkins, “Stable scheduling policies for fading wireless channels,” *IEEE/ACM Trans. Networking*, vol. 13, no. 2, pp. 411–424, Apr. 2005.

[19] N. J. Neely, E. Modiano, and C. E. Rohrs, “Dynamic power allocation and routing for time-varying wireless networks,” *IEEE J. Select. Areas Commun.*, vol. 23, no. 1, pp. 89–103, Jan. 2005.

[20] X. Lin and N. Shroff, “The impact of imperfect scheduling on cross-layer rate control in multihop wireless networks,” in *Proceedings of IEEE Infocom*, Miami, FL, March 2005.

[21] T. Ho and H. Viswanathan, “Dynamic algorithms for multicast with intra-session network coding,” in *Proc. 43rd Annual Allerton Conference on Communication, Control, and Computing*, Sept. 2005.

[22] A. Eryilmaz and D. S. Lun, “Control for inter-session network coding,” MIT LIDS, Technical Report 2722, Aug. 2006.

[23] J. G. Dai, “On the positive Harris recurrence for multiclass queueing networks: A unified approach via fluid limit models,” *Ann. Appl. Probab.*, pp. 49–77, 1995.

[24] J. G. Dai and S. P. Meyn, “Convergence and moments of networks and their fluid models,” *IEEE Trans. Automat. Contr.*, vol. 40, pp. 1889–1904, 1995.

[25] A. Stolyar, “On the stability of multiclass queueing networks: A relaxed sufficient condition via limiting fluid processes,” *Markov Processes and Related Fields*, pp. 491–512, 1995.

[26] A. Eryilmaz and R. Srikant, “Resource allocation of multi-hop wireless networks,” in *Proc. International Zurich Seminar on Communications (IZS)*, Feb. 2006.

[27] H. Khalil, *Nonlinear Systems*. NY: John Wiley & Sons, 2002.

[28] L. Bui, A. Eryilmaz, R. Srikant, and X. Wu, “Joint asynchronous congestion control and distributed scheduling for wireless networks,” in *Proc. IEEE Infocom*, 2006.

[29] R. Dougherty, C. Freiling, and K. Zeger, “The Vámos network,” in *Proc. 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, 2006.