Optimal Routing and Scheduling for a Simple Network Coding Scheme

Nathaniel M. Jones[*§], Brooke Shrader[*], and Eytan Modiano[§]

[*]Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, MA 02420
[§]LIDS, Massachusetts Institute of Technology, Cambridge, MA 02139

Abstract—We consider jointly optimal routing, scheduling, and network coding strategies to maximize throughput in wireless networks. While routing and scheduling techniques for wireless networks have been studied for decades, network coding is a relatively new technique that allows for an increase in throughput under certain topological and routing conditions. In this work we introduce k-tuple coding, a generalization of pairwise coding with next-hop decodability, and fully characterize the region of arrival rates for which the network queues can be stabilized under this coding strategy. We propose a dynamic control policy for routing, scheduling, and k-tuple coding, and prove that our policy is throughput optimal subject to the k-tuple coding constraint. We provide analytical bounds on the coding gain of our policy, and present numerical results to support our analytical findings. We show that most of the gains are achieved with pairwise coding, and that the coding gain is greater under 2-hop than 1-hop interference. Simulations show that under 2-hop interference our policy yields median throughput gains of 31% beyond optimal scheduling and routing on random topologies with 16 nodes.

I. Introduction

Network coding, originally introduced in [1], can increase network throughput by allowing intermediate nodes to combine or encode the data they receive, rather than simply replicating and forwarding it. The benefit of this approach for wireless transmissions was clearly demonstrated by COPE [6], an opportunistic network coding protocol that takes advantage of wireless multicast and allows encoding of packets between multiple unicast sessions using binary XOR operations. The authors combine their coding strategy with a modified MAC protocol to show significant throughput improvements versus a standard 802.11 MAC on a wireless testbed. While the original work on COPE [6] explored the interplay between coding and scheduling, subsequent work in [13] motivated the need for routing protocols to be aware of COPE-style network coding. The appropriate choice of routes can increase coding opportunities and [13] shows that significant throughput improvements are possible through such coding aware routing. In this work, we address the joint design and performance of routing, scheduling, and coding in a wireless network.

Numerous previous works have considered joint routing and scheduling in the absence of network coding. In their seminal

paper on network control [16], Tassiulas and Ephremides introduce the maximum weight scheduling and differential backlog routing policy to provide throughput optimal network control. The policy has an attractive property for dynamic control in that decisions rely only on current queue state information, without requiring knowledge of the long-term arrival rates. The authors are able to prove, using Lyapunov stability theory, that their policy can stabilize the network queues for any stochastic arrival process within the stability region of the network. Neely, Modiano, and Rohrs [9] extended this to jointly optimize for routing, scheduling, and power control in wireless networks with time-varying channels.

Network coding has been incorporated into the design of scheduling and routing schemes in recent work. A number of recent works, including [2], [7], [10], [12], develop joint scheduling and coding schemes in a network control framework, either for single-hop transmissions, or under the assumption that routes are fixed and specified *a priori*. In addressing the routing problem, [17] provides a linear optimization approach for identifying network coding opportunities on butterfly subgraphs with multiple unicast sessions, while [4] develops a policy for dynamic routing and scheduling to provide stability throughout the region from [17]. The *poison-remedy* approach introduced in [4] involves opportunistically identifying coding opportunities, creating *poisoned* or coded packets, and subsequently sending a request for *remedy* or uncoded packets to be sent to the destination node to allow for decoding. In a different approach, [3] provides a distributed backpressure routing and maximum weight scheduling policy for a generalized COPE coding scheme, making opportunistic coding decisions to increase throughput. The policy in [3] exploits the use of overhearing to provide coding opportunities, optimizing for a subset of coding opportunities to reduce complexity while allowing for distributed implementation. Finally, [14] formulates a linear program for the joint routing, scheduling and pairwise coding problem and evaluates results from a computational solution to the problem.

Network coding can be combined with overhearing to yield additional coding opportunities, as shown in [3], [6], [7], [10], [11], [12] and [13]. We may consider the addition of overhearing to our coding strategy in future research.

In this paper, we propose an inter-session network coding strategy that jointly optimizes for routing and scheduling of unicast traffic on wireless networks. The coding scheme we

consider does not require overhearing, but simply requires each node to keep a copy of packets it previously transmitted for some limited period of time. All coded packets must be decoded at the next hop, and when a coding opportunity is identified, the requisite conditions for decoding are already satisfied. The main contributions of this paper are as follows:

- introduces k-tuple coding, a generalization of pairwise inter-session network coding, and fully characterizes the stability region under this coding strategy;
- proposes a dynamic routing, scheduling, and k-tuple coding policy and proves that this policy is throughput optimal subject to the k-tuple coding constraint;
- proves analytical bounds on the throughput gain for k-tuple coding relative to optimal routing and scheduling without network coding;
- provides numerical results via simulation and linear program evaluation to give a sense of the performance of our policy under various settings.

A unique attribute of our policy is that it requires keeping track of which one-hop neighbor supplies each packet; this requirement shows up both in the characterization of the stability region and in the construction of weight calculations.

This paper is organized as follows. Section II describes our system models, and Section III characterizes the stability region under these models. In Section IV we design a control policy that combines scheduling, routing, and network coding to achieve the given stability region. We provide analytical results on coding gain in Section V, and describe the complexity of our coding operations in Section VI. In Section VII we give numerical results, and offer concluding remarks in Section VIII.

II. MODEL

A. Wireless Network

We consider a wireless network modeled as a directed hypergraph, $G = (\mathcal{N}, \mathcal{H})$, where $\mathcal{N}$ is the set of nodes in the network and $\mathcal{H}$ is the set of directed hyperedges supported by the network. Hyperedge $(a, \tilde{J})$ allows head node a to communicate directly with a set of tail nodes $\tilde{J}$ using a single transmission. Standard edge (a, b) is a special case of a hyperedge, where node b is the only tail node. Let $\mathcal{H}_k \subseteq \mathcal{H}$ be the set of hyperedges that contain exactly k tail nodes. We model the network as a hypergraph to capture the effects of wireless multicast transmissions, which are needed by our network coding strategy.

We consider unicast traffic. In this context, wireless multi-cast is used only for the transmission of network coded packets. We assume time to be slotted, and for simplicity assume unit rate links and packets of a fixed size corresponding to one packet per time slot. We refer to packets destined for node c as *commodity c* packets. We allow exogenous packet arrivals from an arbitrary process with finite second moment. Let λ_a^c be the average rate of exogenous arrivals at node a for commodity c, and let $\boldsymbol{\lambda} = (\lambda_a^c)$ be a vector of arrival rates for all sources a and commodities c.

We assume that non-interfering transmissions are reliable, but otherwise consider arbitrary interference constraints. However our numerical results consider two interference models of interest: 1-hop and 2-hop interference. In the context of wireless networks, the 1-hop interference model means that each node can receive from at most one neighbor at a time, and a node cannot receive while transmitting. The 2-hop interference model builds on the restrictions of 1-hop interference, adding a constraint such that simultaneous communications will interfere if connected by any standard edge in the network. These are natural extensions of the 1-hop and 2-hop interference models from [15], where we explicitly allow these models to make use of wireless multicast.

B. Routing and Scheduling

A wireless network requires mechanisms for *routing* packets along a series of hyperedges toward the destination node and for *scheduling* a set of hyperedges to be activated simultaneously without creating interference. Let schedule ℓ be a set of non-interfering hyperedges, and let $\mathcal{L}$ be the set of all such schedules. We consider a centralized control policy that dynamically chooses which hyperedges to activate during each time slot, and chooses which commodity to send over each hyperedge when active.

Tassiulas and Ephremides [16] provided a joint routing and scheduling policy that is throughput optimal; in the absence of network coding, their policy yields 100% throughput for all arrival rate vectors that can be supported by any policy. At each time slot $t \geq 0$ and for each edge (a, b), this policy calculates the edge weight $W_{ab}^*(t)$ as the maximum differential backlog over the edge:

$$W_{ab}^*(t) = \max_{c \in \mathcal{N}} \{U_a^c(t) - U_b^c(t)\} \tag{1}$$

where $U_a^c(t)$ is the backlog at node a of commodity c packets at time t. Their policy then chooses the schedule with maximum total weight $\ell^*(t)$:

$$\ell^*(t) = \arg\max_{\ell \in \mathcal{L}} \sum_{(a,b) \in \mathcal{H}_1} \ell_{ab} W_{ab}^*(t) \tag{2}$$

where $\ell_{ab} = 1$ if edge (a, b) is active in schedule ℓ, and is 0 otherwise. Finally, this policy serves the commodities that maximize Eqn. (1) for each active edge in schedule $\ell^*(t)$. While this policy optimizes for scheduling and routing, the policy as stated only considers standard edges in $\mathcal{H}_1$ and does not account for network coding. We extend this policy from Tassiulas and Ephremides to jointly optimize for scheduling, routing, and our simple network coding scheme.

C. Network Coding

We describe our k-tuple coding operations using a constructive approach by first considering the pairwise case of coding over 2 sessions, then extending this to the case of coding over 3 sessions, and finally generalizing to the case of coding over k sessions. We then motivate the use of coding by describing achievable throughput gains in simple scenarios. Our coding strategy depends on knowing the neighbor from which each
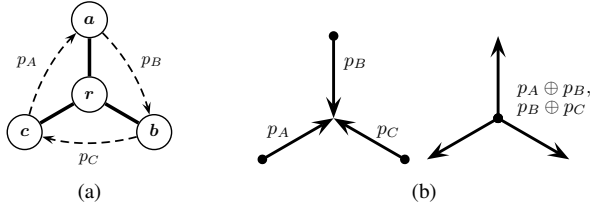
Fig. 1. 3-tuple coding operation at node r. (a) Standard edges shown with solid lines, with all hyperedges available; traffic demands shown with dashed arrows. (b) Activations shown with solid arrows.

packet is received. To accomplish this, nodes store packets in subqueues based on the one-hop source of each packet; one-hop subqueue d for commodity c holds commodity c packets received from neighbor d.

Our coding strategy considers ordered sets of hyperedge tail nodes and commodities. Let (a, J) be a hyperedge with ordered tail nodes, for $J \in \text{perms}(\tilde{J})$, where $\text{perms}(\tilde{J})$ is the set of all permutations of $\tilde{J}$. The tail node at the mth position in J is denoted $J(m)$, and with an abuse of notation $J(k+1) = J(1)$ for $|J| = k$. Let $s \in \mathcal{N}^k$ be an ordered set of k commodities, and let $\mathcal{S}_k$ be the set of all ordered commodity sets of size k. The commodity at the mth position of s is denoted $s(m)$, and again by abuse of notation, $s(k+1) = s(1)$.

Pairwise Coding: Consider node r that has received packet p_A for commodity a from neighbor b and packet p_B for commodity b received from neighbor a. Thus for hyperedge $(r, J), J = (a, b)$, and commodity set $s = (a, b)$, a packet for commodity $s(2) = b$ (*i.e.*, p_B) is in one-hop subqueue $J(1) = a$ and a packet for commodity $s(1) = a$ resides in one-hop subqueue $J(2) = b$. Node r can generate a coded packet $p_{AB} = p_A \oplus p_B$, where $\oplus$ is the binary XOR operation, and then send p_{AB} to nodes a and b in a single time slot using a wireless multicast transmission. Node a has previously seen packet p_B, and can recover $p_A = p_{AB} \oplus p_B$. Likewise, node b can recover packet p_B. Note that we do not require packets p_A, p_B, and p_{AB} to be transmitted in consecutive time slots, but require only that p_{AB} is transmitted after both p_A and p_B have been received at r.

The coding operation requires that each node maintain an extra buffer with uncoded copies of packets that it has previously transmitted; we call this the *side information buffer*. In the example above, upon transmitting to node r, node a keeps p_B and node b keeps p_A in their respective side information buffers. Additionally, node r adds p_A and p_B to its side information buffer upon transmitting p_{AB}. We discuss operations for removing packets from this buffer in Section VI-B. Note that coded packets can be discarded at the end of the coding operation, as only uncoded packets are stored in one-hop subqueues and side information buffers.

3-Tuple Coding: Now suppose that node r has received packet p_A for commodity a from neighbor c, packet p_B for commodity b from neighbor a, and packet p_C for commodity c from neighbor b. For hyperedge $(r, J), J = (a, b, c)$, and commodity set $s = (a, b, c)$, a packet for commodity $s(2) = b$ resides in the one-hop subqueue $J(1) = a$, a packet for

commodity $s(3) = c$ resides in subqueue $J(2) = b$, and a packet for commodity $s(1) = a$ resides in subqueue $J(3) = c$. Node r can encode packets p_A, p_B, and p_C using two coded packets: $p_{AB} = p_A \oplus p_B$ and $p_{BC} = p_B \oplus p_C$. Node r can then transmit coded packets p_{AB} and p_{BC} to neighbors a, b, and c using 2 wireless multicast transmissions. Each of the 3 neighbors can decode the packet destined for them using the 2 coded packets from r along with their side information copy of the uncoded packet that they respectively supplied to the encoding node. Note that even though nodes a, b, and c can decode all 3 packets, they each keep only the one packet that is destined for them and discard the rest. This scenario is shown in Fig. 1.

Definition 1: A **coding opportunity** $(s, (r, J))$ is formed by the combination of ordered hyperedge (r, J) and ordered set of commodities s held at node r for which: (a) $|s| = |J| = k$, and (b) for each $m = 1, 2, ..., k$, a packet for commodity $s(m+1)$ resides in the one-hop subqueue $J(m)$ at node r.

For the pairwise coding scenario, if p_A and p_B are the only packets in the one-hop subqueues at node r, then $s = (a, b)$ is the only set of commodities that forms a coding opportunity with hyperedge $(r, J), J = (a, b)$. Commodity set $s' = (b, a)$ does not form a coding opportunity with hyperedge $(r, J), J = (a, b)$, since at node r, there is no packet for commodity $s'(2) = a$ in one-hop subqueue $J(1) = a$, and no packet for commodity $s'(1) = b$ in one-hop subqueue $J(2) = b$. By assumption, a node will never transmit a packet destined for itself, so commodity set $s' = (b, a)$ and ordered hyperedge $(r, J), J = (a, b)$, will never satisfy the condition (b) for coding opportunities. However commodity set $s' = (b, a)$ and hyperedge $(r, J'), J' = (b, a)$, do form a coding opportunity, since s' and J' are formed by the same circular shift of s and J, respectively. Yet the coding operations and packets delivered for $(s', (r, J'))$ and $(s, (r, J))$ are identical. In general, for any coding opportunity $(s, (r, J))$, we can ignore equivalent circular shifts $(s', (r, J'))$ in constructing a routing and scheduling policy. Furthermore, consider hyperedge $(r, J), J = (a, b)$, on a more general topology, where the the transmit buffer at r contains packets in both one-hop subqueues a and b for commodities g and h. Then commodity sets $s_1 = (g, h)$, $s_2 = (h, g)$, $s_3 = (g, g)$, and $s_4 = (h, h)$ can each be combined with hyperedge (r, J) to form valid coding opportunities.

Coding Rule: A k-tuple coding operation can only be performed for a coding opportunity $(s, (r, J))$ that satisfies Definition 1. A packet for commodity $s(m+1)$ that resides in the one-hop subqueue $J(m)$ at node r is delivered to neighbor $J(m+1)$.

For the 3-tuple coding example, commodity set $s = (c, b, a)$ and ordered hyperedge $(r, J), J = (a, c, b)$ also form a valid coding opportunity. In this alternate coding opportunity, node r delivers packet p_C to node a, p_B to c, and p_A to b.

k-Tuple Coding: Generalizing further, a commodity set s and hyperedge (r, J), $|s| = |J| = k$, can form a k-tuple coding opportunity for $2 \le k \le \deg(r)$, and $\deg(r)$ is the degree of node r. The encoding operation requires r

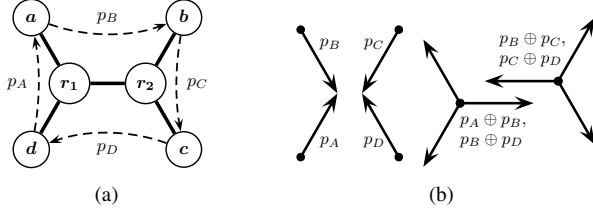Fig. 2. Pair of 3-tuple coding nodes, r_1 and r_2. (a) Standard edges shown with solid lines, with all hyperedges emanating from r_1 and r_2 available; traffic demands shown with dashed lines. (b) Activations shown with solid arrows.

to receive one packet from each of the k distinct neighbors in J, and then to transmit $k-1$ coded packets via wireless multicast to all k neighbors. To encode the uncoded packets $p_1, ..., p_k$ corresponding to commodities $s(1), ..., s(k)$, node r can generate $k-1$ coded packets as: $(p_1 \oplus p_2)$, $(p_2 \oplus p_3)$, ..., and $(p_{k-1} \oplus p_k)$. Each of the k neighbors already has in their side information buffer a copy of the packet that they respectively supplied to r. Upon receiving the $k-1$ coded packets from r, each of the k neighbors can then decode the packet destined for them. For example, assume node d supplied packet p_1 to r, and r sends packet p_k to d using a k-tuple code. Node d can recover packet p_k as: $p_2 = p_1 \oplus (p_1 \oplus p_2)$, $p_3 = p_2 \oplus (p_2 \oplus p_3)$, ... , and $p_k = p_{k-1} \oplus (p_{k-1} \oplus p_k)$. It follows that for all code sizes k, the use of binary XOR operations between pairs of packets is sufficient for both encode and decode operations for k-tuple coding.

Lemma 1: If k neighbors of a node each have in their respective side information buffers at most one packet from a k-tuple coding opportunity, then under any coding strategy, $k-1$ is the fewest number of packets that the coding node must transmit to exchange all k packets.

Proof: The proof follows because in order to solve for $k-1$ unknown packets, $k-1$ linearly independent equations are needed. ∎

Observation 1: A single k-tuple coding operation yields a throughput gain of $\frac{2k}{2k-1}$ when all hyperedges connected to the coding node mutually interfere.

The k-tuple coding operation requires a total of $2k-1$ time slots, while the same packet exchange without coding requires $2k$ time slots, yielding the observed result. For pairwise coding the throughput gain is $4/3$, and for 3-tuple coding the gain is $6/5$. While 3-tuple coding yields a lower gain than pairwise coding, notice that there is no pairwise coding opportunity in the 3-tuple coding scenario in Fig. 1.

Observation 2: There exists a setting where a throughput gain of $\frac{2k-1}{2k-2}$ is achievable.

Consider coding nodes r_1 and r_2 that share a single edge and where both coding nodes have degree k, with interference constraints such that all hyperedges mutually interfere. An example of this scenario is shown in Fig. 2 for the case of $k = 3$ under 2-hop interference. For the traffic demands shown, node r_1 is a tail node for the 3-tuple coding opportunity $(s_2, (r_2, J_2))$, $s_2 = (b, c, d)$ and $J_2 = (b, c, r_1)$, while r_2 is a tail node for the coding opportunity $(s_1, (r_1, J_1))$, where $s_1 = (a, b, d)$ and $J_1 = (a, r_2, d)$. The coding operation at

r_1 can deliver p_A to a, p_B to r_2 and p_D to d. The coding operation at r_2 can deliver p_B to b, p_C to c, and p_D to r_1. Without coding, it takes $2 + 3 + 2 + 3 = 10$ or $2(2k-1)$ time slots to deliver packets p_A, p_B, p_C, and p_D, while k-tuple coding can deliver the same set of packets in $4 + 2 + 2 = 8$ or $2(2k-2)$ time slots using the activations in Fig. 2(b). This yields the observed gain of $\frac{2k-1}{2k-2}$. For pairwise coding this is a throughput gain of $3/2$, while for 3-tuple coding the throughput gain is $5/4$. These throughput gains require a pipeline of coding operations, where nodes r_1 and r_2 are initialized with packets from a, b, c, and d, and the activations cycle between coding operations at r_1 and r_2.

III. Stability Region

The stability region Λ_{NC} of our k-tuple coding strategy is the set of all arrival rate vectors (λ_a^c) that can be supported while ensuring that all packet queues in the network remain finite.

Let $f_{ab}^{d,c}$ be the rate of uncoded flow of commodity c packets supplied by node d and sent over edge (a, b), and let f_{aJ}^s be the rate of flow over ordered hyperedge (a, J) for each commodity in set s, where $(s, (a, J))$ is a coding opportunity. For simplicity, we use the following $\hat{f}$ notation to represent a sum over a set of underlying flow variables. Notation $\{d, c\} \rightarrow b$ means commodity c from one-hop subqueue d is sent to node b. Let $\hat{f}_{ab}^{d,c}$ be the total uncoded and coded flow rate from node a to neighbor b for commodity c from the subqueue for one-hop neighbor d. Thus,

$$\hat{f}_{ab}^{d,c} = f_{ab}^{d,c} + \sum_{\left\{ \begin{array}{c} (a,J) \in \mathcal{H}_k, k \geq 2, s \in \mathcal{S}_k: \\ d, b \in J, c \in s, \{d,c\} \rightarrow b \end{array} \right\}} f_{aJ}^s, \quad \forall a, b, c, d \in \mathcal{N} \quad (3)$$

where the summation is over the set of coded flow variables f_{aJ}^s for all hyperedges (a, J) and commodity sets s that deliver commodity c packets from one-hop subqueue d to node b. Let $\hat{f}_{ab}^c$ be the total coded and uncoded flow rate from a to b for commodity c traffic from all one-hop subqueues.

$$\hat{f}_{ab}^c = \sum_d \hat{f}_{ab}^{d,c}, \quad \forall a, b, c \in \mathcal{N} \quad (4)$$

We start with some efficiency assumptions: nodes don't transmit to themselves and nodes don't transmit any traffic destined for themselves. Also, all flow variables are non-negative. Next, we define several constraints from our policy.

Flow Conservation: For each node a and for each commodity $c \neq a$, all commodity c flow that enters a must leave a. To maintain this flow conservation, the exogenous arrivals for commodity c must equal the difference between total network departures for commodity c and total network arrivals for commodity c.

$$\lambda_a^c = \sum_b \hat{f}_{ab}^c - \sum_d \hat{f}_{da}^c, \quad \forall a, c \in \mathcal{N} : a \neq c \quad (5)$$

Coding Constraint: Our coding strategy allows node a to encode packets for commodity c that have been received directly from neighbor d, where the total flow directly from d

to a for commodity c gives an upper bound on the total coded flow from a that can make use of commodity c packets in the side information buffer at neighbor d.

$$\sum_b \left(\hat{f}_{ab}^{d,c} - f_{ab}^{d,c} \right) \leq \hat{f}_{da}^c , \quad \forall a,c,d \in \mathcal{N} \qquad (6)$$

Hyperedge Rate Constraint: Let γ_ℓ be the fraction of time that schedule ℓ is active, and let $\ell_{a\tilde{J}} = 1$ if hyperedge $(a, \tilde{J})$ is active in schedule ℓ, and 0 otherwise. Let $R_{a\tilde{J}}$ be the fraction of time that hyperedge $(a, \tilde{J})$ is active. Then we find $R_{a\tilde{J}}$ as:

$$R_{a\tilde{J}} = \sum_{\ell \in \mathcal{L}} \ell_{a\tilde{J}} \gamma_\ell , \quad \forall (a, \tilde{J}). \qquad (7)$$

The set $(R_{a\tilde{J}})$ for all hyperedges must then be in the convex hull of the set of all schedules $\mathcal{L}$.

For uncoded traffic ($k = 1$), the fraction of time $R_{a\tilde{J}}$, $\tilde{J} = \{b\}$, that edge (a, b) is active gives an upper bound on the total flow of all commodities over that edge.

$$\sum_{d,c \in \mathcal{N}} f_{ab}^{d,c} \leq R_{a\tilde{J}} , \quad \forall (a,b) : \tilde{J} = \{b\} \qquad (8)$$

For coded traffic ($k \geq 2$), our coding strategy imposes a factor of $\frac{1}{k-1}$ to account for the $k-1$ time slots required to deliver one packet to each destination of a coded packet.

$$\sum_{J \in \mathrm{perms}(\tilde{J}), s \in \mathcal{S}_k} f_{aJ}^s \leq \frac{R_{a\tilde{J}}}{k-1} , \quad \forall (a, \tilde{J}) : |\tilde{J}| = k, k \geq 2 \quad (9)$$

The stability region for our k-tuple coding strategy is the convex polytope bounded by the set of constraints in Eqns. (5-9). It can be shown that Eqns. (5-9) are necessary for stability; due to space constraints, we omit the proof.

Additionally, we give the following two redundant constraints that are informative about our coding strategy.

$$\lambda_a^c = \sum_b f_{ab}^{a,c} , \quad \forall a,c \in \mathcal{N} : a \neq c \qquad (10)$$

$$0 = \left(\sum_b \hat{f}_{ab}^{d,c} \right) - \hat{f}_{da}^c , \quad \forall a,c,d \in \mathcal{N} : a \neq c, d \neq a \quad (11)$$

Eqn. (10) indicates that exogenous arrivals must be sent uncoded by the source node, while Eqn. (11) indicates that the total flow out of one-hop subqueue d for commodity c at node a must equal the total flow sent from d to a for commodity c. These two equations can be viewed as detailed flow conservation constraints for the one-hop subqueues. It can be shown that replacing Eqns. (5) and (6) with Eqns. (10) and (11) does not change the stability region Λ_{NC}.

IV. LCM-FRAME POLICY FOR ROUTING, SCHEDULING, AND k-TUPLE CODING

Our control policy performs scheduling, routing, and k-tuple coding for dynamic choice of k. The transmission of each k-tuple set requires $k-1$ time slots, and we schedule for fixed size frames such that all coding operations are performed within the frame boundary. Therefore, the frame size must be an integer multiple of $k-1$ for each code size $k \in \{1, ..., K\}$.

The least common multiple framing (LCM-Frame) policy uses a fixed frame size of length $T = \mathrm{lcm}\{1, ..., K-1\}$ time slots, where K is the maximum size of any k-tuple code used. Control decisions are made only at the beginning of each frame, so that for each hyperedge active within a frame, all packets sent over the active hyperedge contain packets using the same code size k. Let C_k represent a k-tuple coding set of $k \in \{2, ..., K\}$ packets, which are to be encoded to form $k - 1$ coded packets. For example, in the case of $K = 5$ all frames will have duration $T = \mathrm{lcm}\{1, 2, 3, 4\} = 12$ time slots and each active hyperedge can transmit 12 uncoded packets, 12 sets of C_2, 6 sets of C_3, 4 sets of C_4, or 3 sets of C_5.

At every time slot $t = nT$, for integer $n \geq 0$, the policy operates as follows.

1) For every standard edge $(a, b) \in \mathcal{H}_1$, calculate edge weight $W_{ab}^*(t)$ as below, and choose associated commodity $s_{ab}^*(t)$ and subqueue $d_{ab}^*(t)$. Let $U_a^{d,c}(t)$ represent the backlog at node a at time t for packets received from neighbor d and destined for commodity c. This is a slight modification from the policy in Eqn. (1), in that here we use the backlog of the one-hop subqueue of each commodity instead of the total backlog for each commodity.

$$W_{ab}^*(t) = \max_{c,d} \left\{ U_a^{d,c}(t) - U_b^{a,c}(t) \right\} \qquad (12)$$

2) For every hyperedge $(a, \tilde{J})$, for $k = |\tilde{J}| \geq 2$, calculate the weight as follows. For every ordered hyperedge (a, J), for $J \in \mathrm{perms}(\tilde{J})$, and every commodity set $s \in \mathcal{S}_k$ such that $(s, (a, J))$ is a coding opportunity, calculate the weight $W_{aJ}^s(t)$. First, evaluate the differential backlog for each commodity in s and the respective tail node from J as: $U_a^{J(m),s(m+1)}(t) - U_{J(m+1)}^{a,s(m+1)}(t)$, for each $m = 1, 2, ..., k$. If this differential backlog is non-positive for any position m, then it is not beneficial to encode, so set weight $W_{aJ}^s(t) = 0$. If the differential backlog is positive for all positions m, calculate the weight as:

$$W_{aJ}^s(t) = \frac{1}{k-1} \sum_{m=1}^k \left(U_a^{d,c}(t) - U_b^{a,c}(t) \right) , \qquad (13)$$

where $d = J(m), b = J(m+1)$, and $c = s(m+1)$. The factor $\frac{1}{k-1}$ accounts for $k - 1$ time slots required to transmit the coded set s. Then choose optimal weight $W_{a\tilde{J}}^*(t)$ for the unordered hyperedge as:

$$W_{a\tilde{J}}^*(t) = \max_{J,s} \left\{ W_{aJ}^s(t) \right\} \qquad (14)$$

The optimal commodity set $s_{a\tilde{J}}^*(t)$ and tail ordering $\tilde{J}_a^*(t)$ are chosen as the values of s and J, respectively, that yield the maximum weight in Eqn. (14).

3) Choose the maximum weighted schedule, generalizing Eqn. (2) to allow for hyperedges:

$$\ell^*(t) = \arg\max_{\ell \in \mathcal{L}} \sum_{(a,b)} \ell_{ab} W_{ab}^*(t) + \sum_{(a,\tilde{J})} \ell_{a\tilde{J}} W_{a\tilde{J}}^*(t) \quad (15)$$

4) Repeatedly activate the chosen hyperedges for the duration of the T-slot frame. If there are not enough packets in the subqueue for an active commodity, then null packets are used in place of that commodity for the remainder of the frame.

Definition 2: A queue $U(t), t \geq 0$, is *stable* if

$$\limsup_{t \to \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}[U(\tau)] < \infty \qquad (16)$$

A network is *stable* when all queues in the network are stable.

Theorem 1: The LCM-Frame policy stabilizes the network for all arrival rate vectors interior to stability region Λ_{NC}. The proof is given in the Appendix.

The policy probes all paths in the network, so it will encounter complex coding opportunities such as those in the example from Fig. 2. As the policy makes use of coding opportunities it reduces backpressure along those paths, thereby attracting more traffic to paths that offer coding opportunities.

V. k-TUPLE CODING GAIN

We require a meaningful metric to compare performance with coding versus without coding. We identify the maximum scaling of arrival rate vector subject to stability from Eqns. (5-9). Our metric of interest is the ratio of these stable scalings under k-tuple coding versus routing and scheduling without coding. Let Λ_{RS} be the stability region under routing-and-scheduling only.

Definition 3: Given $\boldsymbol{\lambda} \in \Lambda_{RS}$, let $f_{NC}(\boldsymbol{\lambda}) = \max\{\rho : \rho\boldsymbol{\lambda} \in \Lambda_{NC}\}$ and let $f_{RS}(\boldsymbol{\lambda}) = \max\{\rho : \rho\boldsymbol{\lambda} \in \Lambda_{RS}\}$, where ρ is a scalar. **Coding gain** is the ratio $f_{NC}(\boldsymbol{\lambda})/f_{RS}(\boldsymbol{\lambda})$.

Theorem 2: Considering all possible topologies, traffic demands, and interference constraints, and with dynamic and optimal choice of code size, the coding gain from k-tuple coding is upper bounded by 2.

We offer a sketch of the proof. First, the coding constraint is relaxed, removing the requirement that recipients of coded packets are part of a coding opportunity while still allowing encoding nodes to deliver k packets in $k-1$ time slots. Starting with Eqn. (6), take the sum over all neighbors d for both sides of the inequality, then add λ_a^c to the right side to yield the relaxed coding constraint below.

$$\sum_{b,d} \left(\hat{f}_{ab}^{d,c} - f_{ab}^{d,c} \right) \leq \lambda_a^c + \sum_d \hat{f}_{da}^c, \quad \forall a, c \in \mathcal{N} \qquad (17)$$

By this relaxation, we have $\Lambda_{NC} \leq \Lambda_{RCC}$, where Λ_{RCC} is the stability region under the relaxed coding constraint (17). Next, we compare to the flow conservation constraint (5) and note that the relaxed constraint (17) is degenerate and can be removed. With coding constraint (6) removed, any pair of packets can be encoded and sent over a standard edge in one time slot, allowing each edge to achieve a rate of 2. This implies that $\Lambda_{RCC} = 2\Lambda_{RS}$, which in turn gives the desired result $\Lambda_{NC} \leq 2\Lambda_{RS}$.

VI. COMPLEXITY AND SIDE INFORMATION

The gain in stable throughput provided by the LCM-Frame policy comes at the expense of additional complexity in computing weights and additional side information that must be stored in the network. In this section we quantify these aspects of the policy.

A. Complexity of Weight Computation

The policies that we consider require solving the maximum weight independent set (MWIS) problem at each time slot, which is known to be NP-Hard for general interference graphs. However, polynomial time solutions for MWIS are possible for certain classes of interference graphs, such as claw-free interference graphs [8]. For these classes of graphs with polynomial MWIS solutions, we focus on the complexity of calculating weights of hyperedges for k-tuple coding. Let N be the number of nodes in the network, and let each node represent a commodity.

Standard edges: There are $O(N)$ edges per node, with $O(N)$ commodities and $O(N)$ one-hop sources. The running time for the weight calculations of standard edges at each node is thus $O(N^3)$ per time slot.

Pairwise hyperedges: There are $O(N^2)$ pairwise hyperedges per node, and the required one-hop sources are given by the tail nodes of the hyperedge. Each pairwise hyperedge is composed of two standard edges, and for each of these component standard edges we can independently choose the commodity with maximum differential backlog. This gives a running time at each node of $2N^3 = O(N^3)$ per time slot.

General k-tuple hyperedges, $k \geq 2$: There are $O\binom{N}{k}$ subsets of k tail nodes at each node, and when we eliminate circular shifts, there are $(k-1)!$ circular permutations of each subset of tail nodes to consider. For each circular permutation, the choice of optimal commodity set requires a running time of $O(kN)$. The running time at each node is then $\frac{N!}{(N-k)!k!} Nk(k-1)! = O(N^{k+1})$ per time slot.

B. Upper Bound on Side Information

To decode coded packets, k-tuple coding requires each node to maintain a side information buffer, where uncoded copies of previously transmitted packets are stored.

Corollary 1: For every arrival rate vector (λ_a^c) strictly interior to the stability region Λ_{NC}, the LCM-Frame policy stabilizes the side information buffers in the network.

Proof: The k-tuple coding strategy requires each node a to keep a side information copy of each packet p sent from a to neighbor b only as long as p resides in b's queue. For the LCM-Frame policy, there is only one copy of each packet in the network, and each packet has a single one-hop source. Under centralized control, the activation schedule can alert nodes when to discard packets tracked as side information. The total network queue size then gives an upper bound on the total side information in the network, and by Theorem 1 the LCM-Frame policy stabilizes the network queues whenever $(\lambda_a^c) \in \Lambda_{NC}$. Therefore, the side information buffers are also stable. ∎
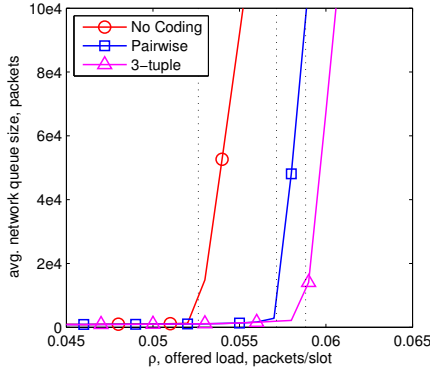
Fig. 3. Network queue size versus offered load for three configurations of the LCM-Frame policy on 16 node topology with 11 traffic demands. Dotted vertical lines indicate bounds of the stability region for each configuration. At each value of ρ between 0.045 and 0.065 at increments of 0.001, we evaluate the LCM-Frame policy for 10 million time slots.
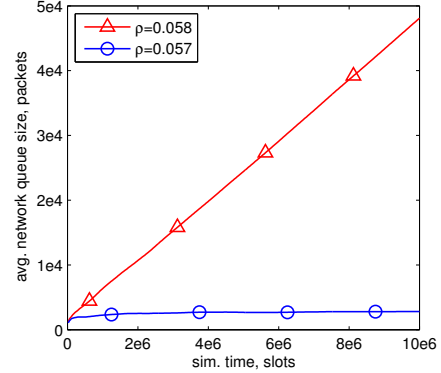

Fig. 4. Network queue size versus simulation time for the pairwise coding configuration from Fig. 3. Average network queue sizes shown for both stable $\rho = 0.057$ and unstable $\rho = 0.058$ values of offered load, with queue state recorded every 25 thousand time slots.

VII. NUMERICAL RESULTS

We use two approaches to study the LCM-Frame policy: a packet simulation to evaluate average queue size for the policy, and a linear program (LP) solver to evaluate the flow constraints of the policy to observe coding gain. For a scenario with N nodes, there are $N-1$ possible traffic demands at each node, for a total of $N(N-1)$ traffic demands in the network. This yields 56 and 240 possible demands for $N = 8$ and $N = 16$, respectively. We generate random arrival rate vectors $\boldsymbol{\lambda}$ by activating each of these demands with probability $1/2$, where demands are specified as 1 for active and 0 for inactive. Let ρ be a value by which we scale $\boldsymbol{\lambda}$ to specify the offered load; in effect, ρ is the offered load for each active demand.

A. Simulation Results

First we consider a random 16 node topology under 2-hop interference, and we choose an arrival rate vector $\boldsymbol{\lambda}$ with 11 active traffic demands, where we scale $\boldsymbol{\lambda}$ by ρ. Exogenous arrivals are generated for each active demand using an independent Bernoulli processes; since $\boldsymbol{\lambda}$ is a vector of 0's and 1's, the scalar ρ serves as the probability of packet arrival per time slot for each active demand. We compare three configurations of the policy: no coding ($K = 1$), pairwise coding ($K = 2$), and 3-tuple coding ($K = 3$).

Fig. 3 shows the time average of the total network queue size, over all nodes and commodities, as a function of offered load. Using the constraints of our stability region, Eqns. (5-9), we find the maximum stable values of offered load to be $\rho = 1/19 \approx 0.0526$ without coding, $\rho = 1/17.5 \approx 0.0571$ for pairwise coding, and $\rho = 1/17 \approx 0.0588$ for 3-tuple coding; these bounds are indicated on Fig. 3 with vertical dotted lines. For each configuration, the policy seems to maintain bounded average queue size within the stability region.

Fig. 4 shows average total network queue size as a function of time for pairwise coding. The queues are stable at an offered load of $\rho = 0.057$, just inside the stability region. Outside the stability region, with $\rho = 0.058$, the average network queue size grows linearly as a function of time.
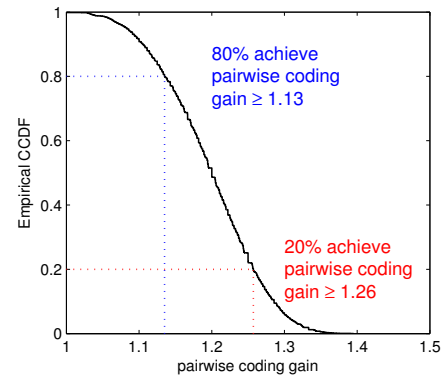
Fig. 5. Empirical CCDF of pairwise coding gain for random 8 node topologies under 2-hop interference.

B. Linear Program Results

We use an LP solver with constraints (5-9) of the stability region to evaluate the coding gain of our LCM-Frame policy. We consider random geometric topologies, with node placement drawn from a uniform distribution in a unit square. Node connectivity is given by a scaled unit disc model, such that two nodes are connected if they are within a certain connectivity radius of one another. In particular, we generate topologies with 8 nodes and a connectivity radius of 0.335, and topologies with 16 nodes with a connectivity radius of 0.273. For both 8 and 16 node cases, the median node degree is 3 among all nodes from the generated topologies. We consider only topologies that are connected.

First we evaluate coding gain for pairwise and 3-tuple coding. We consider 100 random topologies with 8 nodes each under 2-hop interference, and we evaluate coding gain for 100 arrival rate vectors per topology. Fig. 5 shows an empirical complementary cumulative distribution function (CCDF) of the observed pairwise coding gain, where 80% of the observations show gain of 1.13 or more and 20% show gain of 1.26 or more. Fig. 6 shows the ratio of 3-tuple coding gain versus pairwise coding gain. Here we see that 3-tuple coding yields additional gain in only 4% of our observations, and that this gain is limited to at most 6% and often much less.
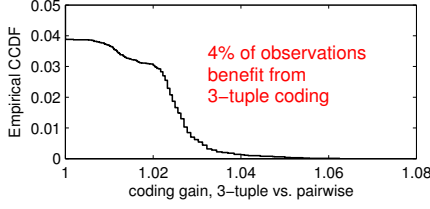
Fig. 6. Empirical CCDF of ratio of coding gain for 3-tuple versus pairwise coding. Same topologies and arrival rate vectors as Fig. 5.
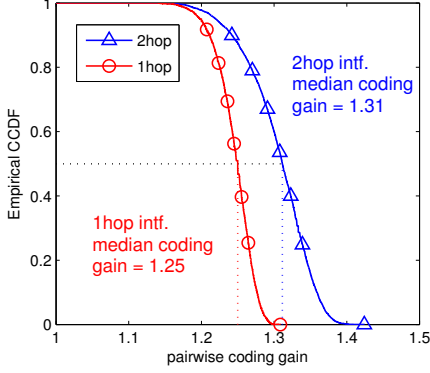

Fig. 7. Empirical CCDFs of pairwise coding gain for 16 node topologies under 1-hop and 2-hop interference.

Finally, we compare the gain of pairwise coding under 1-hop versus 2-hop interference. We evaluate 50 random topologies of 16 nodes each, with 100 random arrival rate vectors per topology. Fig. 7 shows empirical CCDFs of pairwise coding gain for both interference models. Here, pairwise coding performs reasonably well under 1-hop interference, with a median coding gain of 1.25, and performs even better under 2-hop interference, where the median has increased to 1.31. Comparing to the 8 node scenario, there is a noticeable improvement under 2-hop interference here with 80% of observations showing gain above 1.25.

VIII. CONCLUSION

In this paper we presented a technique that dynamically optimizes for routing, scheduling, and simple network coding for wireless networks. We introduced k-tuple coding, a generalization of pairwise network coding, and provided the LCM-Frame policy, which is throughput optimal subject to the k-tuple coding constraint. We have shown achievable coding gain on simple scenarios, provided simulation results for more complex scenarios, and gave an upper bound on k-tuple coding gain for all possible scenarios.

Our main conclusion is that pairwise coding provides most of the benefit of k-tuple coding for the scenarios considered. We evaluated the LCM-Frame policy via packet simulation and LP evaluation for pairwise and 3-tuple coding. Due to the topology and traffic structure required for k-tuple coding operations, we expect limited additional gain from increasing code size k on random topologies. Note that the reduced complexity in computing weights for pairwise coding becomes significant for larger networks. We observe that the LCM-Frame policy yields greater coding gains under 2-hop inter-

ference than under 1-hop interference. Future work of interest includes decentralized scheduling, suboptimal scheduling with reduced complexity, and full system implementation.

APPENDIX
PROOF OF STABILITY FOR LCM-FRAME POLICY

Proof: We use T-slot Lyapunov drift analysis to prove that our policy stabilizes the network for all arrival rate vectors strictly interior to the stability region. Let $A_i^{d,c}(t)$ be the number of exogenous arrivals for commodity c at node i during time slot t, where $A_i^{d,c}(t) = 0$ for any $d \neq i$. Let $\mathbf{U}(t)$ denote the matrix of queue backlogs $U_i^{d,c}(t)$ for all nodes and commodities.

At each decision time $t = nT, n \geq 0$, the LCM-Frame policy chooses which hyperedges and commodities to activate for the duration of the frame. At time $\tau \in \{t, ..., t+T-1\}$, let $\mu_{ab}^{d,c}(\tau)$ be the rate allocated for commodity c from one-hop source d over edge (a, b), where $\mu_{ab}^{d,c}(\tau) = 1$ if active and 0 otherwise. Also at time τ, let $\mu_{aJ}^s(\tau)$ be the rate allocated to each commodity of set s for k-tuple coded transmissions over ordered hyperedge (a, J), with traffic delivered according to the coding rule, where $\mu_{aJ}^s(\tau) = \frac{1}{k-1}, k = |J|$, if active and 0 otherwise. Like with flow variables, $\hat{\mu}$ represents a sum over rate allocation variables. Let $\hat{\mu}_{ab}^{d,c}(\tau)$ be the sum of rate allocation variables for coded and uncoded transmissions from a to b for commodity c packets from one-hop subqueue d:

$$\hat{\mu}_{ab}^{d,c}(\tau) = \mu_{ab}^{d,c}(\tau) + \sum_{\left\{ \substack{(a,J) \in \mathcal{H}_k, k \geq 2, s \in \mathcal{S}_k : \\ d, b \in J, c \in s, \{d,c\} \to b} \right\}} \mu_{aJ}^s(\tau) \qquad (18)$$

Let $\hat{\mu}_{a*}^{d,c}(\tau)$ be the sum of all transmissions at node a for packets from subqueue d for commodity c:

$$\hat{\mu}_{a*}^{d,c}(\tau) = \sum_b \hat{\mu}_{ab}^{d,c}(\tau) \qquad (19)$$

Let $\hat{\mu}_{ab}^{*,c}(\tau)$ be the sum of all network transmissions from a to b for commodity c packets from all one-hop subqueues:

$$\hat{\mu}_{ab}^{*,c}(\tau) = \sum_d \hat{\mu}_{ab}^{d,c}(\tau) \qquad (20)$$

The queueing dynamics of the network satisfy:

$$U_i^{d,c}(t_0 + T) \leq \left[U_i^{d,c}(t_0) - \sum_{\tau=t_0}^{t_0+T-1} \hat{\mu}_{i*}^{d,c}(\tau) \right]^+ + \sum_{\tau=t_0}^{t_0+T-1} \left(A_i^{d,c}(\tau) + \hat{\mu}_{di}^{*,c}(\tau) \right) \qquad (21)$$

where $[x]^+ = \max(x, 0)$. Next, we use the following result from [5, Lemma 4.3]: for $V, U, \mu, A \geq 0$ and $V \leq [U - \mu]^+ + A$, we have $V^2 \leq U^2 + \mu^2 + A^2 - 2U(\mu - A)$. Squaring both sides of Eqn. (21) and noting that $A_i^{d,c}(\tau)$, $\mu_{ab}^{d,c}(\tau)$, and $\mu_{aJ}^s(\tau)$ are all finite, we have:

$$\left(U_i^{d,c}(t_0 + T) \right)^2 - \left(U_i^{d,c}(t_0) \right)^2 \leq B_1 + 2U_i^{d,c}(t_0) \sum_{\tau=t_0}^{t_0+T-1} \left(A_i^{d,c}(\tau) + \hat{\mu}_{di}^{*,c}(\tau) - \hat{\mu}_{i*}^{d,c}(\tau) \right) \qquad (22)$$

where B_1 is a positive finite number.

We employ the quadratic Lyapunov function

$$L\big(\mathbf{U}(t)\big) = \sum_{i,c,d} \big(U_i^{d,c}(t)\big)^2 \qquad (23)$$

and the following T-slot Lyapunov drift argument from [5, Lemma 4.2]: If there exists a positive integer T such that $\mathbb{E}\{\mathbf{U}(\tau)\} < \infty$ for all $\tau \in \{0, ..., T-1\}$, and if there exists a positive B and θ such that for all time slots t_0 we have

$$\mathbb{E}\big\{L\big(\mathbf{U}(t_0+T)\big) - L\big(\mathbf{U}(t_0)\big)\big|\mathbf{U}(t_0)\big\}$$
$$\leq B - \theta \sum_{i,c,d} U_i^{d,c}(t_0) \quad (24)$$

then the network is stable according to Definition 2.

Using Eqn. (22) we have

$$L\big(\mathbf{U}(t_0+T)\big) - L\big(\mathbf{U}(t_0)\big)$$
$$\leq \sum_{i,c,d} \left[B_1 + 2U_i^{d,c}(t_0) \sum_{\tau=t_0}^{t_0+T-1} A_i^{d,c}(\tau) + \hat{\mu}_{di}^{*,c}(\tau) - \hat{\mu}_{i*}^{d,c}(\tau) \right]$$
$$(25)$$

$$\leq N^3 B_1 + 2 \sum_{i,c} U_i^{i,c}(t_0) \sum_{\tau=t_0}^{t_0+T-1} A_i^{i,c}(\tau)$$
$$- 2 \sum_{\tau=t_0}^{t_0+T-1} \sum_{i,b,c,d} \hat{\mu}_{ib}^{d,c}(\tau) \left[U_i^{d,c}(t_0) - U_b^{i,c}(t_0) \right] \quad (26)$$

At each decision time t_0, expected T-slot drift is given as

$$\Delta_T\big(\mathbf{U}(t_0)\big) \triangleq \mathbb{E}\big\{L\big(\mathbf{U}(t_0+T)\big) - L\big(\mathbf{U}(t_0)\big)\big|\mathbf{U}(t_0)\big\}$$
$$\leq N^3 B_1 + 2T \sum_{i,c} U_i^{i,c}(t_0)\lambda_i^c$$
$$- 2 \sum_{\tau=t_0}^{t_0+T-1} \sum_{i,b,c,d} \hat{\mu}_{ib}^{d,c}(\tau) \left[U_i^{d,c}(t_0) - U_b^{i,c}(t_0) \right] \quad (27)$$

For arrival rate matrix $\big(\lambda_i^c\big)$ strictly inside of stability region Λ_{NC}, there exists a small $\epsilon > 0$ such that $\big(\lambda_i^c + \epsilon\big)$ is also inside the stability region. By definition of the stability region, we can identify a flow vector of $f_{ab}^{c,d}$ and f_{iJ}^s terms that corresponds to $\big(\lambda_i^c + \epsilon\big)$ and satisfies the constraints of the stability region. For any decision time t_0 and any $\tau \in \{t_0, ..., t_0 + T - 1\}$, our policy satisfies the following inequality by choosing the set of rate allocation variables corresponding to $\hat{\mu}_{ab}^{d,c}(\tau)$ that maximize the term on the right:

$$\sum_{a,b,c,d} \hat{f}_{ab}^{d,c}\left[U_a^{d,c}(t_0) - U_b^{a,c}(t_0) \right]$$
$$\leq \sum_{a,b,c,d} \hat{\mu}_{ab}^{d,c}(\tau)\left[U_a^{d,c}(t_0) - U_b^{a,c}(t_0) \right] \quad (28)$$

Applying the result from Eqn. (28) to Eqn. (27), we have:

$$\Delta_T\big(\mathbf{U}(t_0)\big) \leq B_2 + 2T \sum_{i,c} U_i^{i,c}(t_0)\lambda_i^c$$
$$- 2T \sum_{i,b,c,d} \hat{f}_{ib}^{d,c}\left[U_i^{d,c}(t_0) - U_b^{i,c}(t_0) \right] \quad (29)$$

$$= B_2 + 2T \left[\sum_{i,c} U_i^{i,c}(t_0)\lambda_i^c - \sum_{i,c} U_i^{i,c}(t_0) \sum_b f_{ib}^{i,c} \right.$$
$$\left. - \sum_{i,c,d\neq i} U_i^{d,c}(t_0)\left(\sum_b \hat{f}_{ib}^{d,c} - \sum_g \hat{f}_{di}^{g,c} \right) \right] \quad (30)$$

$$= B_2 + 2T \left[\sum_{i,c} U_i^{i,c}(t_0)\lambda_i^c - \sum_{i,c} U_i^{i,c}(t_0)\left(\lambda_i^c + \epsilon\right) \right.$$
$$\left. - \sum_{i,c,d\neq i} U_i^{d,c}(t_0)(0) \right] \quad (31)$$

$$= B_2 - 2T\epsilon \sum_{i,c} U_i^{i,c}(t_0) \quad (32)$$

where $B_2 = N^3 B_1$ is finite. We have shown that our policy satisfies Eqn. (24), and thus satisfies the conditions of [5, Lemma 4.2]. The LCM-Frame policy therefore stabilizes the network for all arrival rate vectors strictly interior to the stability region. ∎

REFERENCES

[1] R. Ahlswede, N. Cai, S.Y.R. Li, and R.W. Yeung. Network information flow. *IEEE Trans. on Info. Theory*, 46(4):1204–1216, 2000.

[2] P. Chaporkar and A. Proutiere. Adaptive network coding and scheduling for maximizing throughput in wireless networks. In *Proc. MobiCom '07*, pages 135–146, 2007.

[3] T. Cui, L. Chen, and T. Ho. Energy efficient opportunistic network coding for wireless networks. In *Proc. INFOCOM 2008*, pages 361–365, 2008.

[4] A. Eryilmaz and D. S. Lun. Control for inter-session network coding. In *NetCod: Workshop on Network Coding*, 2007.

[5] L. Georgiadis, M. J. Neely, and L. Tassiulas. *Resource allocation and cross-layer control in wireless networks*, volume 1. Foundations and Trends in Networking, 2006.

[6] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft. XORs in the air: Practical wireless network coding. *IEEE/ACM Trans. on Networking*, 16(3):497–510, 2008.

[7] A. Khreishah, C.C. Wang, and N.B. Shroff. Cross-layer optimization for wireless multihop networks with pairwise intersession network coding. *IEEE Journal on Selected Areas in Comm.*, 27(5):606–621, 2009.

[8] D. Nakamura and A. Tamura. A revision of Minty's algorithm for finding a maximum weight stable set of a claw-free graph. *J. Oper. Res. Soc. Japan*, 44(2):194–204, 2001.

[9] M. J. Neely, E. Modiano, and C. E. Rohrs. Dynamic power allocation and routing for time-varying wireless networks. *IEEE Journal on Selected Areas in Comm.*, 23(1):89–103, 2005.

[10] G. S. Paschos, L. Georgiadis, and L. Tassiulas. Optimal scheduling of pairwise XORs under statistical overhearing and feedback. In *Proc. IEEE RAWNET*, 2011.

[11] S. Rayanchu, S. Sen, J. Wu, S. Banerjee, and S. Sengupta. Loss-aware network coding for unicast wireless sessions: design, implementation, and performance evaluation. In *Proc. ACM SIGMETRICS*, 2008.

[12] Y.E. Sagduyu, D. Guo, and R. Berry. Throughput and stability of digital and analog network coding for wireless networks with single and multiple relays. In *Proc. WICON '08*, 2008.

[13] S. Sengupta, S. Rayanchu, and S. Banerjee. An analysis of wireless network coding for unicast sessions: The case for coding-aware routing. In *Proc. INFOCOM 2007*, pages 1028–1036, 2007.

[14] S. Shabdanov, C. Rosenberg, and P. Mitran. Joint routing, scheduling, and network coding for wireless multihop networks. In *Proc. WiOpt 2011*, pages 33–40, 2011.

[15] G. Sharma, R.R. Mazumdar, and N.B. Shroff. On the complexity of scheduling in wireless networks. In *Proc. MobiCom*, 2006.

[16] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Trans. Auto. Control*, 37(12), Dec. 1992.

[17] 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. ISIT '06*, pages 1758–1762, 2006.