Network Coding-Aware Queue Management for Unicast Flows over Coded Wireless Networks

Hulya Seferoglu, Athina Markopoulou
EECS Dept, University of California, Irvine
{hseferog, athina}@uci.edu

***Abstract*—We are interested in unicast flows over wireless networks with intersession network coding (such as COPE [1]). TCP flows over coded wireless networks do not fully exploit network coding opportunities due to their bursty behavior and to the fact that TCP is agnostic to the underlying network coding. In this paper, we take the following steps. First, we formulate congestion control for unicast flows over coded wireless networks as a network utility maximization problem and we present a distributed solution. Second, by mimicking the structure of the optimal solution, we propose a "network-coding aware" queue management scheme (NCAQM) at intermediate nodes. We make no changes to TCP or MAC protocols. We demonstrate, via simulation, that TCP over NCAQM performs significantly better than TCP over COPE.**

***Index Terms*—Network coding, wireless networks, congestion control, transport protocols, queue management.**

I. Introduction

Wireless environments lend themselves naturally to network coding, thanks to the inherent broadcast and overhearing capabilities. We are particularly interested in wireless mesh networks with opportunistic network coding, which have been extensively studied in theory and practice [1], [2]. We consider unicast flows (particularly TCP, which is the dominant traffic type today) transmitted on top of such networks.

In this setting, it has been demonstrated that network coding can significantly increase throughput [1]. However, it has also been observed [1] that TCP does not exploit the full potential of the underlying network coding, mainly due to its bursty behavior. Rate mismatch between flows can significantly reduce the coding opportunities, as there may not be enough packets from different flows at an intermediate node to code together. One possible solution is to artificially delay packets at intermediate nodes [3], until more packets arrive and can be coded together. However, the throughput increases with small delay (due to more coding opportunities) but decreases with large delay (which reduces the TCP rate); the optimal delay depends on the network topology and the background traffic and also may change over time. We consider the same problem, but propose a different solution. Because the mismatch between flow rates is due to the bursty nature of TCP, the problem can be eliminated by making modifications to congestion control mechanisms (at the end-points) and/or to queue management schemes (at intermediate nodes) to make them network coding-aware.

This work was supported by NSF CAREER grant 0747110 and by AFOSR MURI award FA9550-09-1-0643

First, we formulate the congestion control problem over wireless networks with intersession network coding within the network utility maximization (NUM) framework [4], [5]. We consider that a constructive network coding scheme is deployed in a wireless mesh network according to some predetermined rules, *e.g.*, COPE in [1] for one-hop network coding. The optimal solution of the NUM problem decomposes into several parts, each of which has an intuitive interpretation, namely rate control, queue management, and scheduling.

Second, motivated by the analysis, we explore modifications to congestion control mechanisms, so as to mimic the optimal solution of the NUM problem and fully exploit the potential of network coding. It turns out that the optimal solution dictates minimal and intuitive implementation changes. We propose a network coding-aware queue management scheme at intermediate nodes (*NCAQM*), which stores coded packets and drops packets based on both congestion and network coding information. We note that the queues, which are already used for network coding, are a natural place to implement such changes. In contrast, we do not modify TCP or MAC (802.11) protocols, which enables the practical deployment of our proposal. Finally, we evaluate our proposal via simulation in GloMoSim and we show that TCP over the proposed scheme (*NCAQM*) outperforms significantly TCP over COPE (*e.g.,* it doubles the throughput improvement in some scenarios).

The rest of the paper is organized as follows. Section II discusses related work. Section III presents the system model. Section IV presents the optimization problem and solution. Section V presents the network coding-aware protocol design. Section VI presents simulation results and Section VII concludes the paper.

II. Our Work in Perspective

Our work relies on COPE [1] to do the underlying network coding and provide the available coded and uncoded flows to higher layers. We then seek to optimize the treatment of these flows at the end-points and/or at intermediate nodes so as to maximize network coding opportunities. COPE [1] has also noticed the problem with TCP performance due to rate mismatch, which motivated this study. As discussed in the introduction, [3] addressed this problem by delaying packets to code with others. We propose a different solution via queue management and congestion control.

In terms of analysis, our NUM formulation is within the classic framework [5]. Several optimization problems have

already been studied for networks with network coding. In [7], minimum cost multicast over network coded wireline and wireless networks was studied. This work was extended for rate control in [8] for wireline networks. The rate region of multicast flows when network coding is used is studied in [9], [10]. Resource allocation problems have also been considered for intersession and intrasession network coding for unicast flows. Rate control, routing, and scheduling for generation-based intrasession network coding over wireless networks is considered in [11]. Optimal scheduling and optimal routing for COPE are considered in [12] and [13], respectively. NUM is used in [14] for end-to-end pairwise intersession network coding. Energy efficient opportunistic intersession network coding over wireless is proposed in [15]. Compared to prior work, we focus on the congestion control problem for multiple unicast flows over wireless with a given intersession network coding scheme. The most similar formulation is probably [8] for intra-session network coding; we consider intersession network coding for multiple unicast flows.

In terms of implementation, to the best of our knowledge, our work is the first to take the step from theory (optimization) to practice (protocol design), specifically for the problem of congestion control over intersession network coding. We propose implementation changes, which have a number of desired features: they are justified and motivated by analysis, they perform well (double the throughput in simulations), and they are minimal (only queue management is affected, while TCP and MAC remain intact). The extended version of this paper can be found in [20].

III. System Model

Sources/Flows. Let $\mathcal{S}$ be the set of flows between some source-destination pairs. Each flow $s \in \mathcal{S}$ is associated with a rate x_s and a utility function $U_s(x_s)$, which we assume to be a strictly concave function of x_s.

Wireless Network. A hyperarc $(i, \mathcal{J})$ is a collection of links from node $i \in \mathcal{N}$ to a non-empty set of next-hop nodes $\mathcal{J} \subseteq \mathcal{N}$. A hypergraph $\mathcal{H} = (\mathcal{N}, \mathcal{A})$ represents a wireless mesh network, where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of hyperarcs. For simplicity, $h = (i, \mathcal{J})$ denotes a hyperarc, $h(i)$ denotes node i and $h(\mathcal{J})$ denotes node $\mathcal{J}$, *i.e.*, $h(i) = i$ and $h(\mathcal{J}) = \mathcal{J}$. We use these representations interchangeably.

We consider the protocol model of interference [16], according to which, each node can either transmit or receive at the same time and all transmissions in the range of the receiver are considered as interfering. Given a hypergraph $\mathcal{H}$, we can construct the conflict graph $\mathcal{C} = (\mathcal{A}, \mathcal{I})$, whose vertices are the hyperarcs of $\mathcal{H}$ and edges indicate interference between hyperarcs. A clique $\mathcal{C}_q \subseteq \mathcal{A}$ consists of several hyperarcs, at most one of which can transmit simultaneously without interference.

Network Coding: We assume that intermediate nodes use COPE [1] for one-hop opportunistic network coding. Each node i listens all transmissions in its neighborhood, stores the overheard packets in its virtual buffer, and periodically advertises the content of its virtual buffer to its neighbors.

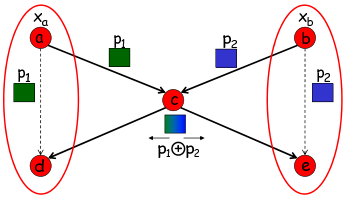


Fig. 1. "X topology". a transmits a flow with rate x_a to e and b transmits a flow with rate x_b to d over c. a and b transmit their packets p_1 and p_2, in two time slots, and c receives them. Furthermore, d overhears p_1 and e overhears p_2, because $a-d$ and $b-e$ are in the same transmission range and they can overhear each other. In the next time slot, c broadcasts the network coded packet, $p_1 \oplus p_2$ over hyperarc $(c, \{d, e\})$. Since d and e have overheard p_1 and p_2, they can decode their packets p_2 and p_1, respectively.

Then, when a node i wants to transmit a packet, it checks or estimates the contents of the virtual buffer of its neighbors. If there is a network coding opportunity, the node combines the relevant packets using simple coding operations (XOR) and broadcasts the combination to $\mathcal{J}$. Note that it is possible to construct more than one network code over a hyperarc $(i, \mathcal{J})$. Let $\mathcal{K}_{i,\mathcal{J}}$ be the set of network codes over a hyperarc $(i, \mathcal{J})$. Let $\mathcal{S}_k \subseteq \mathcal{S}$ be the set of flows, whose packets are coded together using code $k \in \mathcal{K}_{i,\mathcal{J}}$ and broadcast over $(i, \mathcal{J})$.

In this paper, we primarily assume one-hop network coding based on COPE [1]. However, our formulations are general enough to also apply to butterfly structures [6], and in general to multi-hop constructive coding schemes for multiple unicasts, as long as the underlying coding scheme is considered known. We present the multi-hop extensions in [20].

Routing: Each flow $s \in \mathcal{S}$ follows a single path $\mathcal{P}_s \subseteq \mathcal{N}$ from the source to the destination. This path is pre-determined by a routing protocol, *e.g.*, OLSR or AODV, and given as input to our problem. However, note that several different hyperarcs may connect two consecutive nodes along the path. We define $H_{i,\mathcal{J}}^{s,k} = 1$ if s is transmitted through hyperarc $(i, \mathcal{J})$ using network code $k \in \mathcal{K}_{i,\mathcal{J}}$; and $H_{i,\mathcal{J}}^{s,k} = 0$, otherwise.

Example 1: The example shown in Fig. 1 illustrates the problem we consider. Since c can transmit $p_1 \oplus p_2$ in one time slot, instead of p_1, p_2 in two time-slots, network coding has the potential to improve throughput. However, if there is mismatch between the rates x_a, x_b of the two flows, c may not have packets from the two flows to code together at all times, and thus does not exploit the full potential of network coding. We confirmed this intuition through simulations in this example topology. When the buffer size was set to 10 packets at each node and the bandwidth was $1Mbps$ for each link, we observed that 50% of the time, there were no packets from the two flows at the same time at c to code together. For smaller queue sizes and larger transmission rates, there were even fewer coding opportunities. This means that there is potential for improvement by updating the protocols so as to mitigate the rate mismatch between TCP flows. This is the observation that motivates this paper. □

IV. Optimal Congestion Control

A. Problem Formulation

Our objective is to maximize the total utility function by optimizing the flow rates x_s at sources $s \in \mathcal{S}$, their traffic splitting parameter $\alpha_h^{s,k}$ (following the terminology of [8]) into network codes $k \in \mathcal{K}_h$ over hyperarc h at intermediate nodes, and the percentage of time τ_h each hyperarc is used:

$$\begin{aligned}
&\max_{\boldsymbol{x},\boldsymbol{\alpha},\boldsymbol{\tau}} \sum_{s\in\mathcal{S}} U_s(x_s) \\
&\text{s.t.} \sum_{k\in\mathcal{K}_h} \max_{s\in\mathcal{S}_k}\{H_h^{s,k}\alpha_h^{s,k}x_s\} \le R_h\tau_h,\ \forall h\in\mathcal{A} \\
&\sum_{h(\mathcal{J})|h\in\mathcal{A}} \sum_{k\in\mathcal{K}_h|s\in\mathcal{S}_k} \alpha_h^{s,k} = 1,\ \forall s\in\mathcal{S}, i\in\mathcal{P}_s \\
&\sum_{h\in\mathcal{C}_q} \tau_h \le \tau,\ \forall \mathcal{C}_q \subseteq \mathcal{A} \qquad (1)
\end{aligned}$$

The first constraint is the capacity constraint. $H_h^{s,k}\alpha_h^{s,k}x_s$ indicates the part of flow rate x_s allocated to the k-th network code over hyperarc h. The rate of the k-th network code is the maximum rate among flows $s \in \mathcal{S}_k$ coded together in code k: $\max_{s\in\mathcal{S}_k}\{H_h^{s,k}\alpha_h^{s,k}x_s\}$ [7]. Different network codes $k \in \mathcal{K}_h$ over h share the available capacity $R_h\tau_h$, where R_h is the transmission capacity of h; since h is a set of links, R_h is the minimum: $R_h = \min_{j\in h(\mathcal{J})}\{R_{i,j}\xi_{i,j}\}$ where $R_{i,j}$ is the capacity of link (i,j), and $\xi_{i,j}$ is the probability of successful transmission over link (i,j). The second constraint is the flow conservation constraint: at every node i on the path $\mathcal{P}_s$ of source s, the sum of $\alpha_h^{s,k}$ over all network codes and hyperarcs should be equal to 1. Indeed, when a flow enters a particular node i, it can be transmitted to its next hop j as part of different network coded and uncoded flows. The third constraint is due to interference. As mentioned, τ_h is the percentage of time h is used. Its sum over all hyperarcs in a clique should be less than an over-provisioning factor, $\gamma \le 1$, because all hypearcs in a clique interfere and should time-share the medium.

B. Solution

By relaxing the capacity constraint in Eq. (1), we have

$$\begin{aligned}
&L(\boldsymbol{x},\boldsymbol{\alpha},\boldsymbol{\tau},\boldsymbol{q}) = \\
&\sum_{s\in\mathcal{S}} U_s(x_s) - \sum_{h\in\mathcal{A}} q_h\Big(\sum_{k\in\mathcal{K}_h} \max_{s\in\mathcal{S}_k}\{H_h^{s,k}\alpha_h^{s,k}x_s\} - R_h\tau_h\Big), \qquad (2)
\end{aligned}$$

where q_h is the Lagrange multiplier, which can be interpreted as the queue size at hyperarc h, as discussed later. To decompose the Lagrangian, we rewrite $\max_{s\in\mathcal{S}_k}\{H_h^{s,k}\alpha_h^{s,k}x_s\}$ as $\max_{m_h^{s,k}} \sum_{s\in\mathcal{S}_k} H_h^{s,k}\alpha_h^{s,k}x_s m_h^{s,k}$ *s.t.* $\sum_{s\in\mathcal{S}_k} m_h^{s,k} = 1$, where $m_h^{s,k}$ is a new variable, which we call the *dominance indicator*. It indicates whether the source s has the maximum rate among all flows coded together in the k-th network code, or not. In the next section, we will see that only the dominant flow in a network code needs to back-off during congestion.

The Lagrange function in Eq. (2) is not strictly concave in $m_h^{s,k}$ and this causes oscillation in its solution. We use the proximal method [18] to eliminate oscillations;

$$\begin{aligned}
&\max_{\boldsymbol{m}} \sum_{s\in\mathcal{S}_k} (H_h^{s,k}\alpha_h^{s,k}x_s m_h^{s,k} - c(m_h^{s,k} - \mu_h^{s,k})^2) \\
&\text{s.t.} \sum_{s\in\mathcal{S}_k} m_h^{s,k} = 1, \qquad (3)
\end{aligned}$$

where c is a constant and $\mu_h^{s,k}$ is an artificial variable of the proximal method [18]. Its value is set to $m_h^{s,k}$ periodically. Let $(m_h^{s,k})^*$ be the solution to this problem.

By rewriting the summation $\sum_{k\in\mathcal{K}_h}\sum_{s\in S_k}$ as $\sum_{s\in\mathcal{S}}\sum_{k\in K_h|s\in\mathcal{S}_k}$, the Lagrange function in Eq. (2) can be expressed as: $L(\boldsymbol{x},\boldsymbol{\alpha},\boldsymbol{\tau},\boldsymbol{q}) = \sum_{h\in\mathcal{A}} q_h R_h\tau_h + \sum_{s\in\mathcal{S}}\Big(U_s(x_s) - x_s\sum_{h\in\mathcal{A}}\sum_{k\in\mathcal{K}_h|s\in\mathcal{S}_k} q_h H_h^{s,k}\alpha_h^{s,k}(m_h^{s,k})^*\Big)$. Now, we can decompose the Lagrangian into the following intuitive problems: rate control, traffic splitting, scheduling, and parameter update (queue management).

Rate Control. First, we solve the Lagrangian w.r.t x_s:

$$x_s = (U_s')^{-1}\left(\sum_{h\in\mathcal{A}}\sum_{k\in\mathcal{K}_h|s\in\mathcal{S}_k} q_h H_h^{s,k}\alpha_h^{s,k}(m_h^{s,k})^*\right), \qquad (4)$$

where $(U_s')^{-1}$ is the inverse function of the derivative of U_s. If we define $w_h^s = \sum_{k\in\mathcal{K}_h|s\in\mathcal{S}_k} H_h^{s,k}\alpha_h^{s,k}(m_h^{s,k})^*$ and $q_{h(i)}^s = \sum_{h(\mathcal{J})|h\in\mathcal{A}} q_h w_h^s$, the rate x_s can be expressed as $x_s = (U_s')^{-1}(\sum_{i\in\mathcal{P}_s} q_i^s)$, noting that $i = h(i)$.

In the special case where proportional fairness is desired, $U_s(x_s) = \log(x_s), \forall s \in \mathcal{S}$, leading to $x_s = \left(\sum_{i\in\mathcal{P}_s} q_i^s\right)^{-1}$, *i.e.*, x_s is inversely proportional to the total network coded queue sizes over the path of flows s, which we will be explained later.

Traffic Splitting. Second, we solve the Lagrangian for $\alpha_h^{s,k}$: At each node i along the path (*i.e.*, $i \in \mathcal{P}_s$), the traffic splitting problem can be expressed as

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} \sum_{h(J)|h\in\mathcal{A}} \sum_{k\in K_h|s\in\mathcal{S}_k} q_h H_h^{s,k}(m_h^{s,k})^*\alpha_h^{s,k} \\
&\text{s.t.} \sum_{h(J)|h\in\mathcal{A}} \sum_{k\in K_h|s\in\mathcal{S}_k} \alpha_h^{s,k} = 1,\ \forall i\in\mathcal{P}_s \qquad (5)
\end{aligned}$$

Similar to Eq. (3), we also use the proximal method [18] to solve the optimization problem in Eq. (5).

Scheduling. Third, we solve the Lagrangian for τ_h. This problem is solved for every hyperarc and every clique for the conflict graphs in the hypergraph.

$$\max_{\boldsymbol{\tau}} \sum_{h\in\mathcal{A}} q_h R_h\tau_h \qquad \text{s.t.} \qquad \sum_{h\in\mathcal{C}_q} \tau_h \le \tau,\ \forall \mathcal{C}_q \subseteq A. \qquad (6)$$

Parameter (Queue Size) Update. We find the Lagrange multipliers (queue sizes) q_h, using gradient descent; $q_h(t+1) = \{q_h(t) + c_t[\sum_{k\in\mathcal{K}_h}\sum_{s\in\mathcal{S}_k} H_h^{s,k}\alpha_h^{s,k}(m_h^{s,k})^*x_s - R_h\tau_h]\}^+$. Equivalently;

$$q_h(t+1) = \{q_h(t) + c_t[\sum_{k\in\mathcal{K}_h} \max_{s\in\mathcal{S}_k}\{H_h^{s,k}\alpha_h^{s,k}x_s\} - R_h\tau_h]\}^+ \qquad (7)$$

where t is the iteration number, c_t is a small constant, and the $^+$ operator makes the Lagrange multipliers positive. q_h can be interpreted as the queue size at hyperarc $\forall h \in \mathcal{A}$. Indeed, in Eq. (7), q_h is updated with the difference between the incoming $\sum_{k\in\mathcal{K}_h} \max_{s\in\mathcal{S}_k}\{H_h^{s,k}\alpha_h^{s,k}x_s\}$ and outgoing $R_h\tau_h$ traffic at h. Therefore, we call q_h the *hyperarc-queue*, or *h-queue* for brevity. We confirmed the convergence of q_h's via numerical calculations as in [20].

V. Network Coding-Aware Implementation

In the previous section, we saw that the NUM problem decomposed into Eqs. (4)-(7), each of which has an intuitive interpretation. In this section, we mimic the properties of the optimal solution and we propose modifications to the corresponding protocols to make them network coding-aware. It turns out that changes limited to queue management at intermediate nodes are sufficient, while TCP and scheduling can remain intact without loss in performance. This makes our proposal well suited for practical deployment.

A. Queue Management at Intermediate Nodes (NCAQM)

1) Summary of Proposed Scheme: We refer to our "Network Coding-Aware Queue Management" scheme as *NCAQM*. NCAQM builds on and extends COPE [1]. Its goal is to interact with TCP congestion control in a way that matches the rates of TCP flows coded together and thus increases network coding opportunities. It achieves its goal through the following changes at intermediate nodes. *NCAQM* stores coded packets in the output queue $\mathcal{Q}_i$. *NCAQM* maintains state per hyperarc queue q_h and per network code transmitted over each hyperarc $k \in \mathcal{K}_h$; this is feasible in the setting of wireless mesh with limited number of flows. During congestion at a node, a packet is dropped from the flow that has the largest number of packets, where this number is computed only over h-queues where this flow is dominant. Essentially, NCAQM is "longest-flow first" policy, thus balancing the flow lengths, but the "length" of each flow is calculated so as to take into account a key feature of inter-session network coding. We note that intermediate nodes perform already network coding operations and are a natural place to implement these additional changes.

2) Detailed Description of Proposed Scheme:

Maintaining Queues: A wireless node i maintains a single physical output queue, $\mathcal{Q}_i$, which stores all packets (coded and uncoded depending on the opportunities) passing through it. On the other hand, motivated by the fact that Lagrange multiplier (h-queue), we maintain h-queue virtually for each hyperarc, which keeps track of packets that are network coded and broadcast over h. The size of an h-queue is Q_h and how it is determined in practice will be explained later.

Network Coding (Alg. 1): Motivated by the fact that the incoming traffic in Eq. (7) is the sum of the network coded flows over h, we code packets when they are inserted to output queues. If a network coding opportunity does not exist when the packet arrives at node i, we just store it in $\mathcal{Q}_i$ in a FIFO way. Periodically, Alg. 1 runs to check all packets in the queue for network coding. Let $\mathcal{Q}_i = \{p_1, p_2, ..., p_l\}$ where p_1 is the first and p_l is the last packet in the queue; $l \leq L$, where L is the buffer size, *i.e.*, the maximum number of packets that can be stored in $\mathcal{Q}_i$. First, p_1 is picked for network coding. Since $\mathcal{Q}_i$ stores network coded packets, p_1 may be already a coded. Independently of whether p_1 is network coded or not, it can be further coded with other packets in the queue beginning from p_2, if the following two conditions are satisfied; (i) the packets constructing p_1 and p_2 should be from different flows, and (ii) $p_1 \oplus p_2$ should be decodable at the next hop of all packets that construct the network code. If these conditions are satisfied, we say that the network code is an eligible network code, and p_1 is replaced by $p_1 \oplus p_2$. Then $p_1 \oplus p_3$ is checked for network coding, etc. After all packets are checked for network coding, the output queue $\mathcal{Q}_i$ is updated: (i) the final packet p_1 is stored in the first slot of the output queue, and (ii) the memory allocated to other packets are freed. Then, the same algorithm is run for packet p_2, etc. When a transmission opportunity arises, the first packet from the output queue is checked for network coding again and broadcast over the hyperarc.

Algorithm 1 Network coding in output queue Q_i at node i

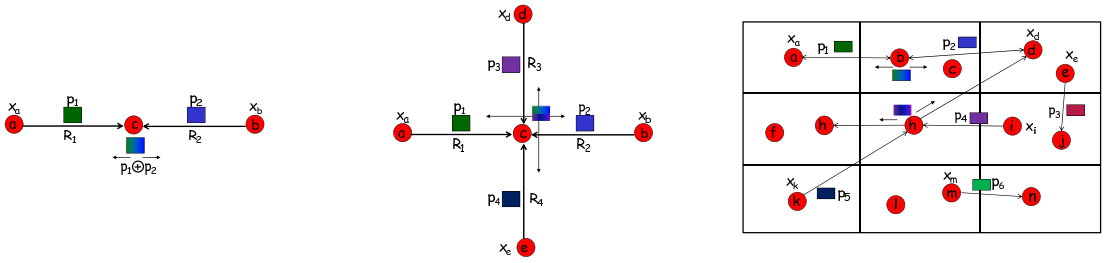
```
for m = 1...L do
    if ∃p_m ∈ Q_i then
        for n = (m+1)...L do
            if p_m ⊕ p_n is eligible then
                p_m ← p_m ⊕ p_n
            end if
        end for
    end if
    Update Q_i
end for
```

Algorithm 2 Packet dropping at node i during congestion

```
Initialization: Φ_i^s = 0, ∀s ∈ S, S_i' = ∅
if l > L then
    for ∀s ∈ S do
        Calculate Φ_i^s = Σ_{h(J)|h∈A} Q_h w̃_h^s
    end for
    S_i' = arg max_{s∈S}{Φ_i^s}
    Choose a flow s' ∈ S_i' randomly
    if ∃p_n ∈ Q_i, n = 1..l, from flow s' then
        Drop p_n
    else
        Drop p_l
    end if
end if
```

Let the number of packets from flow s in node i be Q_i^s. Q_i^s captures the difference between the incoming and outgoing traffic for flow s at node i. Since an h-queue captures the difference between the incoming and outgoing traffic over a hyperarc, we calculate its size using the following heuristic: $Q_h = \sum_{k\in\mathcal{K}_h} \max_{s\in\mathcal{S}_k}\{H_h^{s,k}\breve{\alpha}_h^{s,k}Q_i^s\}$, where $\breve{\alpha}_h^{s,k}$ is the approximate traffic splitting, explained next.

Through numerical calculations, we made the following observation: each $\alpha_h^{s,k}$ converges to the percentage of time that packets from flow s are transmitted with the k-th network code over h at node i. At each packet transmission, we calculate the probability that a network code k over hyperarc h can be used for flow s, over a time window. The average over this window



(a) Alice-and-Bob Topology (b) Cross Topology (c) Grid Topology

Fig. 2. Topologies and traffic scenarios used in simulations.

gives an estimate of the traffic splitting parameter, $\check{\alpha}_h^{s,k}$.

Packet Dropping (Alg. 2): When a node is congested, it decides which packet to drop. In order to eliminate the potential of rate mismatch between flows coded together, we propose that the node compares the number of all (coded and uncoded) packets of each flow, in queues where the flow is dominant ($m_h^{s,k} = 1$). This is motivated by the optimal rate control in Eq. (4). Specifically, for each flow s, we calculate $\Phi_i^s = \sum_{h(\mathcal{J})|h\in\mathcal{A}} Q_h \check{w}_h^s$, where $\check{w}_h^s = \sum_{k\in\mathcal{K}_h|s\in\mathcal{S}_k}$ and $H_h^{s,k}\check{\alpha}_h^{s,k}\check{m}_h^{s,k}$. Upon congestion, the Φ_i^s's are compared and a packet from the flow with the largest Φ_i^s is dropped, preferably the last uncoded packet. If all packets from the selected flow are coded, a newcoming packet(s) is dropped instead.

To estimate the dominance indicator $\check{m}_h^{s,k}$ needed in Alg. 2, we compute heuristically an estimate $\check{m}_h^{s,k}$ as follows. If $H_h^{s,k}\check{\alpha}_h^{s,k}Q_i^s < H_h^{s',k}\check{\alpha}_h^{s',k}Q_i^{s'}$ s.t. $\exists s' \in \mathcal{S}_k - \{s\}$, then $\check{m}_h^{s,k} = 0$. Otherwise, $\check{m}_h^{s,k} = (|\mathcal{S}_k^{max}|)^{-1}$ where $\mathcal{S}_k^{max} = \{s|s\in\mathcal{S}_k \wedge H_h^{s,k}\check{\alpha}_h^{s,k}Q_i^s = \max\{H_h^{s',k}\check{\alpha}_h^{s',k}Q_i^s \mid s'\in\mathcal{S}_k\}\}$.

B. Rate Control at the Sources

The optimal rate x_s is inversely proportional to the sum of the queue sizes q_i^s across all nodes i on the path $\mathcal{P}_s$ of flow s according to Eq. (4) for logarithmic utility (*i.e.*, $x_s = (\sum_{i\in\mathcal{P}_s} q_i^s)^{-1}$. q_i^s). However, it is impractical to feed back to the source the full information $\sum_{i\in\mathcal{P}_s} q_i^s$. Instead, when a queue is congested, a packet is dropped or marked [4]. The source uses this binary information as a signal to reduce its rate, mimicking the inverse relationship. The exact adaptation of the flow rate depends on the TCP version used. In the simulations, we used TCP-SACK without any modification. The only change we propose is the packet dropping scheme at the queue. TCP still reacts to drops but these drops are caused according to network coding requirements (Alg. 2).

C. Scheduling

The scheduling part in Eq. (6) has two parts: intra- and inter-scheduling that determine which packet to transmit from a node and which node should transmit, respectively. Both have difficulties in practice. Intra-scheduling causes packet reordering at TCP receivers. Inter-scheduling requires centralized knowledge and it is NP hard and hard to approximate [5]. Given these difficulties and our goal to make minimal changes, we limit our modifications to the queue management.

VI. Performance Evaluation

In this section, we evaluate the throughput of TCP over our proposed scheme (*NCAQM*) in various topologies and traffic scenarios. We compare it to TCP over the following baseline schemes: no network coding (*noNC*), which uses FIFO without network coding; *COPE* [1], which stores native packets in a FIFO and decides which packets to code together at each transmission opportunity; and the optimal control.

A. Simulation Setup

We used the GloMoSim simulator [19], which is well suited for wireless. We implemented from scratch the modules for one-hop network coding over wireless mesh networks (COPE) as well as for our proposed scheme (NCAQM).

1) Topologies: We simulated four illustrative topologies shown in Fig. 1 and Fig. 2. In X, Alice-and-Bob (a and b transmit to each other via the relay c), and cross (a, b and d, e communicate through the relay c) topologies, c is placed in the center of a circle with $90m$ radius over $200m \times 200m$ terrain and all other nodes are placed around the circle. In the grid topology nodes are distributed over a $300m \times 300m$ terrain, divided into 9 cells of equal size. 15 nodes are divided into sets consisting of 1 or 2 nodes and each set is assigned to a different cell. If both sender and receiver are in the same cell or in neighboring cells, there is a direct transmission; otherwise, a node in a neighboring cell acts as a relay. If there are more than one neighboring cells, one is chosen at random.

2) MAC: In the MAC layer, we simulated IEEE 802.11 with RTS/CTS enabled and with the modifications proposed in [1] for network coding.

3) Wireless Channel: We used the two-ray path loss model and Rayleigh fading (with good link quality) in GloMoSim.

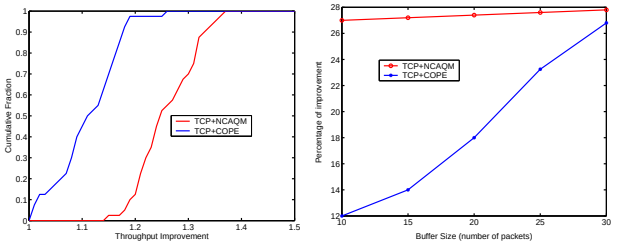
4) TCP Traffic: We consider FTP/TCP traffic on top of the wireless network. In the Alice-and-Bob, X, and cross topologies, TCP flows start at random times within the first $5sec$ and live until the end of the simulation. In the grid topology, TCP flows arrive according to a Poisson distribution with average 6 flows per $30sec$. The sender and the receiver of a TCP flow are chosen randomly.

B. Simulation Results

In this section, we present simulation results for the four topologies. We compare: (i) TCP over NCAQM (*TCP+NCAQM*) to (ii) TCP over COPE (*TCP+COPE*) as

TABLE I
AVERAGE THROUGHPUT IMPROVEMENT COMPARED TO NO-NC.

	Optimal	TCP+NCAQM	TCP+COPE
Alice-and-Bob Topology	33%	27%	12%
Cross Topology	60%	31%	16%
X Topology	33%	22%	10%
Grid Topology	-	19%	8%



(a) CDF (b) Throughput improvement

Fig. 3. (a) CDF, buffer size is 10 packets. (b) The effect of buffer size (in # of packets). Average throughput improvement compared to *noNC* in several scenarios over the Alice-and-Bob topology.

well as to (iii) the optimal solution working together with the optimal queue management in Eq. (7). We report the average throughput of each scheme as % improvement over the throughput of the baseline *TCP+noNC*.

Table I presents the results for the following parameters: the buffer size at each intermediate node is 10 packets; the packet size of $1000B$; the channel capacity is $1Mbps$; the simulation duration is $1min$. The results are averaged over 10 simulations. The first observation is that our scheme (*TCP+NCAQM*) doubles the throughput improvement compared to *TCP+COPE* in all four topologies. We note that the buffer size was purposely chosen to be limited to make network coding opportunities scarce. *TCP+NCAQM* has two advantages in this challenging scenario: (i) it stores network coded, instead of native, packets thus saving buffer and (ii) it drops packets to increase network coding opportunities. The second observation is that *TCP+NCAQM* performs close to the optimal for the Alice-and-Bob and X topologies. For the cross topology, there is still a gap due to the very limited buffer size for 4 flows at the relay (rather than 2 flows in Alice-and-Bob and X topologies).

Fig. 3(a) shows the cumulative distributed function (CDF) for the Alice-and-Bob topology, buffer size of 10 packets and 40 simulations. The CDF of *TCP+NCAQM* is shifted to significantly higher throughput levels compared to *TCP+COPE*. *E.g., TCP+NCAQM* improves throughput more than 20% in more than 90% of the realizations.

Fig. 3(b) shows the throughput improvement vs. buffer size. When buffer sizes are small, the difference between *TCP+NCAQM* and *TCP+COPE* is significant. The throughput of *TCP+COPE* increases when buffer sizes increase, which is intuitively expected. The problem addressed in this paper was the mismatch between rates of flows coded together, due to the bursty nature of TCP, which reduces coding opportunities. However, when buffer sizes increase, there are more packets available in queues for coding. Fig. 3(b) demonstrates that our scheme is particularly beneficial in harsh conditions.

VII. CONCLUSION

In this paper, we showed how to improve the TCP performance over wireless networks with a given intersession network coding scheme. The key intuition was to eliminate the rate mismatch between flows that are coded together, through a synergy of rate control and queue management. First, we formulated the NUM problem and derived a distributed solution. Then, we proposed minimal modifications to queue management (NCAQM) that mimic the structure of the optimal solution; TCP and MAC protocols remained intact. Simulation results showed that the proposed scheme significantly improves TCP performance compared to network coding-unaware protocols. The extended version of this paper, including convergence analysis and the extension to multi-hop network coding, can be found in [20].

REFERENCES

[1] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, J. Crowcroft, "XORs in the Air: Practical Wireless Network Coding," *IEEE/ACM Trans. on Networking*, Vol. 16, No. 3, pp. 497-510, June 2008.

[2] Y. Wu, P. A. Chou, and S. Y. Kung, "Information exchange in wireless network coding and physical layer broadcast," *in CISS 2005 in Proc. of IEEE CISS*, Baltimore, MD, March 2005.

[3] Y. Huang, M. Ghaderi, D. Towsley, and W. Gong, "TCP performance in coded wireless mesh networks," *in Proc. of IEEE SECON 2008*.

[4] R. Srikant, "The Mathematics of Internet Congestion Control", Birkhauser, 2003.

[5] M. Chiang, S. T. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: A mathematical theory of network architectures," *Proceedings of the IEEE*, vol. 95(1), pp. 255-312, Jan. 2007.

[6] S. Omiwade, R. Zheng, and C. Hua. "Butterflies in the mesh: lightweight localized wireless network coding," *in Proc. of NetCod*, Jan. 2008.

[7] D. S. Lun, N. Ratnakar, M. Medard, R. Koetter, D. R. Karger, T. Ho, E. Ahmed, and F. Zhao, "Minimum-cost multicast over coded packet networks," *in IEEE ToIT*, vol. 52(6), June 2006.

[8] L. Chen, T. Ho, S. Low, M. Chiang, and J. C. Doyle, "Optimization based rate control for multicast with network coding," *in Proc. of Infocom*, 2007.

[9] J. Yuan, Z. Li, W. Yu, and B. Li, "A cross-layer optimization framework for multi-hop multicast in wireless mesh networks," *in IEEE JSAC*, vol. 24(11), Nov. 2006.

[10] Z. Li, B. Li, and M. Wang, "Optimization models for streaming in multihop wireless networks," *in Proc. of ICCCN*, Honolulu, HI, Aug. 2007.

[11] B. Radunovic, C. Gkantsidis, P. Key, P. Rodriguez, and W. Hu, "An optimal framework for practical multipath routing in wireless mesh networks," *in Proc. of Infocom*, April 2008.

[12] P. Chaporkar and A. Proutiere, "Adaptive network coding and scheduling for maximizing througput in wireless networks," *in Proc. of ACM Mobicom*, Montreal, Canada, Sep. 2007.

[13] S. Sengupta, S. Rayanchu, and S. Banarjee, "An Analysis of Wireless Network Coding for Unicast Sessions: The Case for Coding-Aware Routing," *in Proc. of Infocom*, 2007.

[14] A. Khreishah, C. C. Wang, and N. B. Shroff, "Cross-layer optimization for wireless multihop networks with pairwise intersession network coding," *in IEEE JSAC*, vol. 27(5), June 2009.

[15] T. Cui, L. Chen, and T. Ho, "Energy Efficient Opportunistic Network Coding for Wireless Networks," *in Proc. of Infocom*, April 2008.

[16] P. Gupta and P. R. Kumar, "The Capacity of Wireless Network," *in IEEE Trans. on Information Theory*, vol. 34(5), pp. 910-917, 2000.

[17] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgement options," *RFC 2018, IETF*, Oct. 1996.

[18] D. P. Bertsekas and J. N. Tsitsiklis, "Parallel and Distributed Computation: Numerical Methods," NJ, Prentice Hall, 1989.

[19] GloMoSim Version 2.0 , "Global Mobile Information Systems Simulation Library," *available at http://pcl.cs.ucla.edu/projects/glomosim/.*

[20] H. Seferoglu, A. Markopoulou, "Improving the Performance of TCP over Coded Wireless Networks," *in arXiv:cs.NI:1002.4885*, Feb. 2010.