Energy Efficient Opportunistic Network Coding for Wireless Networks

Tao Cui, Lijun Chen, and Tracey Ho
Division of Engineering and Applied Science
California Institute of Technology
Pasadena, CA, USA 91125
Email: {taocui, tho}@caltech.edu, chen@cds.caltech.edu

***Abstract*— We consider energy efficient network coding design in wireless networks with multiple unicast sessions. Our approach decomposes multiple unicast sessions into a superposition of multicast and unicast sessions, with coding occurring only within each session. We give an optimization approach that is more general than the existing poison-remedy optimization formulation. For the case of wireless, we consider XOR coding and give an achievable rate region for a primary interference model. To simplify network operation, we give an oblivious backpressure algorithm which does not optimize overhearing of transmissions, and a practical protocol called COPR based on the oblivious backpressure algorithm. Simulation experiments show that COPR largely reduces network power consumption over existing algorithms.**

I. Introduction

In this paper, we consider energy efficient cross-layer optimization for wireless networks by exploiting network coding and multiple-reception gain. We focus on network coding across multiple unicast sessions, or intersession network coding. Optimal intersession network coding design is an open problem; various suboptimal algorithms have been proposed, e.g. [1]–[4].

Our approach decomposes multiple unicast sessions into a superposition of multicast and unicast sessions, with coding occurring only within each session. For the case of wireless networks, we consider simple one-hop XOR coding as in COPE [1], where each node uses knowledge of what its neighbors have overheard to perform opportunistic network coding such that each encoded packet can be decoded immediately at the next hop. Reference [1] demonstrated substantial throughput gains for network coding that grow with the level of congestion. In this paper we consider the benefit of network coding for energy saving in power-constrained settings with less congestion. In doing so we develop general techniques that apply also to the case of throughput optimization and congestion control in wireless networks.

To exploit multiple-reception gain, we model the network as a directed hypergraph. The achievable rate region of one-hop XOR coding is determined under a primary interference model. It is difficult and complicated to design dynamic scheduling and coding algorithms to achieve the entire rate region as it typically requires optimization over overheard flows. To simplify network operation, an *oblivious backpressure* algorithm is proposed which does not optimize overheard flows. The link scheduling problem is found to be a maximum weighted hypergraph matching problem, which can be solved distributedly by using the algorithms in [5]. To further reduce the complexity of session scheduling, the algorithm optimizes only over coding opportunities for packets at the head of queues at each node. By using the suboptimal scheduling algorithm, a fully distributed COPR protocol is proposed. Our simulation experiments show that COPR achieves up to 25% power saving over pure routing, showing that exploiting multiple-reception gain and network coding can enhance overall network performance substantially.

Our contributions can be summarized as follows:

- A new optimization approach is proposed for intersession network coding, based on decomposition into a superposition of multicast and unicast sessions with intrasession network coding. This formulation includes the poison-remedy approach of [2]–[4] as a special case.
- The achievable rate region of one-hop wireless XOR coding is determined under a primary interference model.
- An oblivious backpressure algorithm is proposed for dynamic scheduling and one-hop wireless XOR coding. Note that COPE does not have a specially designed session and link scheduling algorithm. Moreover, we also consider exploiting multiple-reception gain. The oblivious backpressure based scheduling can also be combined with fixed path routing as in COPE.
- A fully distributed protocol, COPR, is proposed. By using specially designed packets' format, the overhead of sending reception reports is reduced.

II. Related Work

For brevity we do not list all works on network coding and on backpressure techniques in networking, but mention here a few that are most closely related to this work. Without network coding, joint congestion control, routing, and scheduling is studied in [6]. The impact of imperfect scheduling on cross-layer design is studied in [7]. In [8], the network capacity region is characterized, and a joint routing and power allocation policy is proposed to stabilize any input rates within the capacity region. This approach is extended in [9] to consider energy efficiency, and in [10] to consider multiuser diversity. Opportunistic routing protocol is proposed in [11] which makes use of multiuser diversity.

With intrasession network coding, in [12], Lun *et. al.* propose a dual subgradient method for the problem of minimum cost multicasting with network coding. For rate control, the approach in [6] is extended to wireline networks in [13]. In [14], the rate stability region for a wireless network with and without correlated sources is characterized. Crosslayer design with broadcast advantage is considered in [5], where the scheduling problem is formulated as a hypergraph matching problem.

This work has been supported in part by DARPA grant N66001-06-C-2020, Caltech's Lee Center for Advanced Networking and a gift from Microsoft Research.

With intersession network coding, opportunistic XOR coding is proposed in [1]. Constructive XOR coding across pairs of unicasts is considered in [2] using a linear optimization approach. Dynamic backpressure is applied in [3], [4]. In [15], coding-aware routing is considered, which requires central controller.

III. Preliminaries

A. Network Model

Wireless networks are considered in this paper. As in [5], [14], [16], the network is modeled as a directed hypergraph $\mathcal{H} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of hyperarcs. A hyperarc is a pair (i, J), where i, the start node, is an element of $\mathcal{N}$, and J, the set of end nodes, is a non-empty subset of $\mathcal{N}$. Each hyperarc (i, J) represents a broadcast link from node i to nodes in J, which indicates that a packet transmitted by node i may be received by nodes in J due to the broadcast nature of the wireless channel. When J only contains a single node j, the hypergraph reduces to the conventional graph model used in [6], [7]. A set of unicast sessions $\mathcal{U} = \{\tilde{u}_1, \ldots, \tilde{u}_{|\mathcal{U}|}\}$ is transmitted through the network. There are $|\mathcal{U}|$ commodities in the network corresponding to each unicast session. Let s_c and t_c denote the source and receiver of commodity c (or unicast session $\tilde{u}_c$, the terms commodity and session are used interchangeablely in this paper), and $\tilde{x}^c$ denote the flow rate of commodity c, $c = 1, \ldots, |\mathcal{U}|$. For a unicast session $\tilde{u}_c$ where the source s_c wants to transmit $\tilde{x}^c$ bits to the sink t_c, by the flow conservation condition, we have

$$\sum_{\{J|(i,J)\in\mathcal{A}\}} \sum_{j\in J} f^c_{iJj} - \sum_{j\in\mathcal{N}} \sum_{\{I|(j,I)\in\mathcal{A},\, i\in I\}} f^c_{jIi} = \tilde{\sigma}^c_i,\ \forall i \in \mathcal{N}, \quad (1)$$

where

$$\tilde{\sigma}^c_i = \begin{cases} \tilde{x}^c, & \text{if } i = s_c \\ -\tilde{x}^c, & \text{if } i = t_c \\ 0, & \text{otherwise,} \end{cases} \quad (2)$$

and f^c_{iJj} is the flow rate over hyperarc (i, J) intended to node $j \in J$.

The primary interference model is assumed in this paper, where each node is equipped with only a single transceiver. Therefore, links that share a common node cannot be active simultaneously. If we further assume that nodes use orthogonal CDMA or FDMA, links that do not share nodes can transmit at the same time. Under this interference model, it is easy to see that any feasible link schedule corresponds to a hypergraph matching [5], where a hypergraph matching is defined as a set of hyperarcs with no pair incident to the same node. Let Ξ denote the set of all hypergraph matchings with each hypergraph matching indexed by ξ. We represent a hypergraph matching as a 0-1 vector $\underline{\alpha}^\xi$, where the l-th entry is

$$\alpha^\xi_{iJ} = \begin{cases} 1 & \text{if } (i, J) \in \xi, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

The set of all the feasible scheduling vectors is defined as the convex hull of these 0-1 scheduling vectors

$$\Pi = \left\{ \underline{\alpha} \,\middle|\, \underline{\alpha} = \sum_{\xi\in\Xi} b_\xi \underline{\alpha}^\xi,\ b_\xi \geq 0,\ \sum_{\xi\in\Xi} b_\xi \leq 1 \right\}, \quad (4)$$

where $\underline{\alpha} = (\alpha_{iJ})$ and α_{iJ} denotes the frequency of (i, J) used in scheduling.

Different from previous work [5]–[7], where lossless link or hyperarc is assumed and Shannon channel capacity is used, here we assume each hyperarc is lossy, which means it experiences packet erasures, a more realistic assumption for practical systems such as IEEE 802.11. Let R_{iJ} denote node i's transmission rate on hyperarc (i, J), where R_{iJ} is assumed to be fixed. Node i allocates power P_{iJ} to (i, J), which is assumed to be fixed as P^{tot}_i (P^{tot}_i is the total power at node i). Note that the proposed algorithms can also be extended to the case that R_{iJ} is chosen from a set of discrete values corresponding to different modulations and P_{iJ} is chosen from a set of different power levels. Under the primary interference model, the packet reception probability at node j for a transmission on (i, J) is only determined by R_{iJ}, P_{iJ}, h_{ij} and is independent of other nodes' transmissions, i.e., the function $p_{iJj}(R_{iJ}, P_{iJ}, h_{ij})$, where h_{ij} represents the current state of channel from node i to node j. Denote 2^J as the power set of J and $\eta_{iS}(R_{iJ}, P_{iJ}, \underline{h}_{iJ})$ as the probability that $S \subseteq J$ is exactly the set of nodes that successfully receive a packet transmitted by node i. The dependence of p_{iJj} and η_{iS} on $R_{iJ}, P_{iJ}, \underline{h}_{iJ}$ is neglected in the following. Let $f^{S,c}_{iJj}$ denote the flow rate of commodity c from i to j when $S \subseteq J$ is exactly the set of nodes that receive a packet transmitted by node i successfully. By using scheduling $\underline{\alpha}$, we have the following constraints

$$f^c_{iJj} = \sum_{\{S|S\in 2^J,\, j\in S\}} f^{S,c}_{iJj},\ \forall i,\ j \in J, \quad (5)$$

$$\sum_{j\in S} f^{S,c}_{iJj} = \alpha_{iJ} \lambda^c_{iJ} \eta_{iS} R_{iJ},\ \forall S \in 2^J, \quad (6)$$

where $0 \leq \lambda^c_{iJ} \leq 1$ is the fraction of time that node i transmits traffic from commodity c over (i, J), and the right hand side of (6) is the average transmission rate from i to nodes in S.

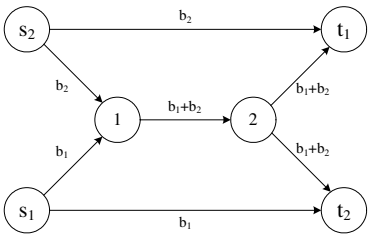

Fig. 1. The wireline butterfly network. Each link is with unit capacity. Two unicast sessions exist. s_i and t_i are source and receiver of session $\tilde{u}_i$, $i = 1, 2$.

B. Network Coding

In network coding, each node is allowed to perform algebraic operations on received packets. Network coding can be classified into intrasession network coding and intersession network coding. Intrasession network coding performs coding only across packets of the same session, while intersession network coding codes packets across different sessions. Intrasession network coding allows traffic for different sinks of a session to share network capacity. Typically random network coding is used for intrasession coding to ensure fully distributed network operating algorithms.

Intersession network coding is still in its infancy. State of art approach usually uses poison-remedy flow scheme [2]–[4]. For example, consider the wireline butterfly network in Fig. 1. If all the links in Fig. 1 have unit capacity, $\tilde{x}^1 = \tilde{x}^2 = 1$ is not feasible with routing, yet feasible with intersession network coding as shown in Fig. 1, where $\tilde{x}^i$ is the rate of session $\tilde{u}_i$, $i = 1, 2$. However, when the capacity of each link is arbitrary, it is not easy to determine the optimal intersession network

coding strategy. In [2]–[4], whenever the flows from the two sessions are coded (or poisoned) at node 1, a remedy request is sent to s_1 (s_2), and s_1 (s_2) sends remedy packets along the link (s_1, t_2) ((s_2, t_1)) to facilitate decoding the encoded packets at t_2 (t_1). This approach requires high overhead to send remedy request, it does not allow coding over remedy packets, and it only allows coding over two sessions each time.

IV. Intersession Network Coding via Session Decomposition

In this section we describe a new optimization approach for intersession network coding that is also inspired by Fig. 1 but is more general than the poison-remedy formulation. Our algorithm is based on the observation that in the coding scheme in Fig. 1, source s_i actually multicasts b_i to both t_1 and t_2, $i = 1, 2$. Therefore, the two unicast sessions can be considered as a single multicast session with two sources s_1, s_2 and two receivers t_1, t_2. When the capacity of each link in Fig. 1 is arbitrary, we can consider that three sessions exist, where two unicast sessions u_i, $i = 1, 2$ share the same source and sink as $\tilde{u}_i$, and one multicast session m has two sources s_1, s_2 and two receivers t_1, t_2. Let x^{u_i} denote the rate of session u_i and x^m denote the rate of session m. The coding scheme in Fig. 1 then becomes intrasession coding within the multicast session m. Note that there is no intersession coding in the new formulation.

This approach decomposes multiple unicast sessions into a superposition of multicast and unicast sessions, and can be generalized to coding across pairs of commodities on a general network as follows. The source s_c of each commodity $c \in \mathcal{U}$ partitions its exogenous packets into a unicast to receiver node t_c and $|\mathcal{U}| - 1$ multicast sessions each involving one other commodity. Each multicast session m involves two commodities c_m^1, c_m^2 and two corresponding receiver nodes t_m^1 and t_m^2. For $i = 1, 2$, t_m^i becomes an intermediate source for the commodity c_m^i packets it decodes, and discards packets that are not from c_m^i; like the original sources it partitions its decoded commodity c_m^i packets into a unicast to receiver node $t_{c_m^i}$ and $|\mathcal{U}| - 1$ multicast sessions each involving one other commodity. The commodity c packets that are unicast from the original and/or intermediate sources are treated as a single unicast session $\tilde{u}_c$. The packets from commodities c_m^1, c_m^2 that are multicast from the original and/or intermediate sources to a pair of corresponding receiver nodes t_m^1 and t_m^2 are treated as a single multicast session m. Intrasession network coding is applied within each session.

The network capacity constraints and flow conservation for each session are the same as for the intra-session network coding problem, e.g. [12]–[14], and the techniques in these works can be generalized to this case by adding the following additional constraints on flow conservation across sessions, where x_i^{mc} and $x_i^{u_c}$ denote the flow rate of commodity c from (new or intermediate) source node i for multicast session m and unicast session u_c respectively:

$$\sum_{\substack{\{m | t_m^1 = i \text{ and } c = c_m^1 \\ \text{or } t_m^2 = i \text{ and } c = c_m^2\}}} \sum_{j \in \mathcal{N}} x_j^{mc} + I(t_c = i) \sum_{j \in \mathcal{N}} x_j^{u_c} + I(s_c = i)\tilde{x}^c = \sum_m x_i^{mc} + x_i^{u_c} + I(t_c = i)\tilde{x}^c, \forall i \in \mathcal{N}, c, \tag{7}$$

where $I(\cdot)$ is the indicator function. The left hand side of (7) is the total flow rate of commodity c received at node i and the right hand side is the total flow of commodity c injected at node i for all the sessions.

There are $|\mathcal{U}|$ unicast sessions and up to $\binom{|\mathcal{U}|}{2}|\mathcal{N}|(|\mathcal{N}| - 1)$ multicast sessions. The complexity can be traded off flexibly against performance by constraining the set of potential multicast receiver nodes. The intermediate nodes may be chosen heuristically or randomly (like oblivious routing).

We can generalize this approach by allowing a multicast session to involve a set $\mathcal{C}$ of more than two commodities to be coded together, or more than two receiver nodes t_m^i. The commodities in $\mathcal{C}$ are partitioned among the receiver nodes so that each commodity's packets are kept at only one receiver which becomes an intermediate source for those packets.

Note that our formulation does not explicitly have remedy request and remedy flows. This allows more general coding of remedy flows with poison flows, and includes the XOR poison-remedy approach in [2]–[4] as a special case. Also there is no need for separate transmission of remedy requests.

V. Energy Efficient Opportunistic Backpressure Algorithms

A. Motivation and Key Idea

The algorithms in [2]–[4] for intersession network coding have high complexity and overhead, while the experimental results in [2] show that the gain of intersession network coding with the poison-remedy approach is fairly modest in wireline networks compared with optimal routing. On the other hand, it is reported in [1] that simple network coding can achieve large gains in the total throughput in wireless networks. One fundamental difference between the setup in [1] and those in [2]–[4] is that the former uses the broadcast nature of wireless communication. All these motivate us to consider using "simple" intersession network codes to exploit the multiple-reception gain in wireless networks.

We consider the one-hop XOR intersession network coding strategy that each coded packet is decoded at the immediate nexthop node, which is similar to that in [1]. However, our approach differs from [1] in that we carefully design session scheduling policy while [1] simply apply round-robin packet scheduling at each node. Also we consider opportunistic reception of both coded and uncoded packets while [1] only considers uncoded packets. For example, in Fig. 2, the packet successful reception probabilities from node 3 to nodes 4-7 are labeled besides each link. Nodes 4 and 5 receive b_1 from node 1, nodes 6 and 7 receive b_2 from node 2, and node 3 receives both b_1 and b_2. By using the strategy in [1], node 3 sends an encoded packet $b_1 + b_2$ to nodes 5 and 6, while our strategy also allows nodes 4-7 to receive the coded packet as all of them can decode the packet. The probability that at least one intended node receives the coded packet is 0.75 in the former case, while it increases to 0.9375 in the later case, which shows the potential benefit by using the proposed strategy. In our strategy, if both 4 and 5 receive the coded packet, a protocol is run to determined which one should keep the packet. Details of the protocol will be given in Section VI. When applying this strategy to a general wireless network, the key idea is to adopt the algorithm in Section IV by decomposing the network into superimposed wireless butterfly networks, each of which is similar to that in Fig. 2 centering around every node.

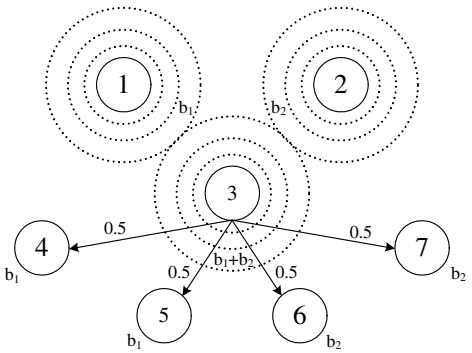

Fig. 2. Example of intersession coding with opportunistic reception. The packet successful reception probabilities from node 3 to nodes 4-7 are labeled besides each link. Nodes 4 and 5 receive b_1 from node 1, nodes 6 and 7 receive b_2 from node 2, and node 3 receives both b_1 and b_2.

B. Opportunistic Unicast without Network Coding

We start from the simple case of unicast without network coding, which gives the intuition for the algorithm with network coding in Section V-C. We want to minimize the average power cost of the whole network. Let λ_{iJ}^c denote the fraction of time node i transmits packets from commodity c to nodes in J, $\sum_c \lambda_{iJ}^c = t_{iJ}$, and let α_{iJ} be the frequency that (i,J) is used in scheduling as in (4). Clearly, the percentage of time that (i,J) is active is $\alpha_{iJ}t_{iJ}$ and the average power consumption on (i,J) is $\alpha_{iJ}t_{iJ}\Omega_{iJ}(P_{iJ})$, where $\Omega_{iJ}(\cdot)$ is a power cost function associated with each (i,J), with the property $\Omega_{iJ}(0)=0$.

We thus have the following joint opportunistic routing and scheduling problem

$$\begin{aligned}
\min_{f,\lambda,\alpha} \quad & \sum_{(i,J)\in\mathcal{A}} \alpha_{iJ}t_{iJ}\Omega_{iJ}\left(P_i^{\text{tot}}\right) \\
\text{subject to} \quad & \sum_{\{J|(i,J)\in\mathcal{A}\}}\sum_{j\in J} f_{iJj}^c - \sum_{j\in\mathcal{N}}\sum_{\{I|(j,I)\in\mathcal{A},\, i\in I\}} f_{jIi}^c = \tilde{\sigma}_i^c, \\
& f_{iJj}^c = \lambda_{iJ}^c z_{iJj}^c,\ \forall i,\ j\in J, \\
& z_{iJj}^c = \sum_{\{S|S\in 2^J,\, j\in S\}} z_{iJj}^{S,c},\ \forall i,\ j\in J, \\
& \sum_{j\in S} z_{iJj}^{S,c} = \alpha_{iJ}\eta_{iS}R_{iJ},\ \forall S\in 2^J, \\
& \sum_{c\in\mathcal{U}} \lambda_{iJ}^c = t_{iJ},\ 0\le t_{iJ}\le 1,\ \underline{\alpha}\in\Pi,
\end{aligned} \tag{8}$$

where the first constraint comes from (1), and the third and fourth constraints follow from (5) and (6). As network coding is not used, flow sharing is not available. Therefore, at any time slot, node i can only transmit packets from a particular session. z_{iJj}^c denotes the achievable flow rate when only commodity c exists at node i. By time sharing among all the sessions $\lambda_{iJ}^c z_{iJj}^c$ is flow rate for commodity c.

Note that in real networks, whether a packet is received successfully by a node is a random process. Thus, solving (8) by using dual decomposition directly may not give a scheduling not matched to current network state. However, the solution given by dual decomposition indeed sheds light on the optimal operation at each time slot. By relaxing only the first set of constraints in (8) and introducing Lagrange multipliers q_i^c at node i for commodity c, after simplification, we find that solving the dual problem is equivalent to solving

$$\begin{aligned}
\max_{f,\lambda,\alpha} \quad & \sum_{(i,J)}\sum_c \lambda_{iJ}^c \sum_{j\in J} z_{iJj}^c\left(q_i^c - q_j^c\right) \\
& \qquad - \sum_{(i,J)\in\mathcal{A}} \alpha_{iJ}t_{iJ}\Omega_{iJ}\left(P_i^{\text{tot}}\right) \\
\text{subject to} \quad & z_{iJj}^c = \sum_{\{S|S\in 2^J,\, j\in S\}} z_{iJj}^{S,c},\ \forall i,\ j\in J, \\
& \sum_{j\in S} z_{iJj}^{S,c} = \alpha_{iJ}\eta_{iS}R_{iJ},\ \forall S\in 2^J, \\
& \sum_{c\in\mathcal{U}} \lambda_{iJ}^c = t_{iJ},\ 0\le t_{iJ}\le 1,\ \underline{\alpha}\in\Pi.
\end{aligned} \tag{9}$$

By fixing $\underline{\alpha}$ and λ_{iJ}^c in (9) and considering only the set S in (i,J), we first solve the problem

$$\begin{aligned}
\max_{f} \quad & \sum_{j\in S} z_{iJj}^{S,c}\left(q_i^c - q_j^c\right) \\
\text{subject to} \quad & \sum_{j\in S} z_{iJj}^{S,c} = \alpha_{iJ}\eta_{iS}R_{iJ}.
\end{aligned} \tag{10}$$

Note that (10) is a linear program and the optimal solution is achieved at an extreme point. The maximum value of the objective function in (10) can be readily obtained as $\alpha_{iJ}\eta_{iS}R_{iJ}\max_j[q_i^c - q_j^c]^+$, where $[\cdot]^+$ denotes the projection onto $\mathbb{R}^+$. Therefore, we have

$$\begin{aligned}
\max_{f} \quad & \sum_{j\in J} z_{iJj}^c\left(q_i^c - q_j^c\right) \\
\text{subject to} \quad & z_{iJj}^c = \sum_{\{S|S\in 2^J,\, j\in S\}} z_{iJj}^{S,c},\ \forall i,\ j\in J, \\
& \sum_{j\in S} z_{iJj}^{S,c} = \alpha_{iJ}\eta_{iS}R_{iJ} \\
=& \alpha_{iJ}R_{iJ}\sum_{S\in 2^J}\eta_{iS}\max_{j\in S}[q_i^c - q_j^c]^+.
\end{aligned} \tag{11}$$

Let π_{iJ}^c[1] denote the permutation of nodes in J according to queue length difference such that

$$[q_i^c - q_{\pi(1)}^c]^+ \ge [q_i^c - q_{\pi(2)}^c]^+ \ge \cdots \ge [q_i^c - q_{\pi(|J|)}^c]^+. \tag{12}$$

We can rewrite (11) as

$$\alpha_{iJ}R_{iJ}\sum_j \chi_{i\pi(j)}^c[q_i^c - q_{\pi(j)}^c]^+ = \alpha_{iJ}\nu_{iJ}^c, \tag{13}$$

where $\nu_{iJ}^c = R_{iJ}\sum_j \chi_{i\pi(j)}^c[q_i^c - q_{\pi(j)}^c]^+$, and $\chi_{i\pi(j)}^c$ is the probability that a packet transmitted by node i is successfully received by node $\pi(j)$ and it is not received by any node in $\{\pi(1),\ldots,\pi(j-1)\}$ or equivalently any node has queue length difference greater than $\pi(j)$. Note that the representation of ν_{iJ}^c in (13) is much easier to compute than (11).

The optimization problem at each (i,J) then becomes

$$\max_{0\le t_{iJ}\le 1} \alpha_{iJ}t_{iJ}\left(\nu_{iJ}^* - \Omega_{iJ}\left(P_i^{\text{tot}}\right)\right), \tag{14}$$

where $\nu_{iJ}^* = \max_c \nu_{iJ}^c$. The solution to (14) is

$$t_{iJ}^* = \begin{cases} 0, & \text{if } \nu_{iJ}^* \le \Omega_{iJ}\left(P_i^{\text{tot}}\right), \\ 1, & \text{otherwise.} \end{cases} \tag{15}$$

Substituting (15) into (14) and summing over all hyperarcs, we obtain the following scheduling problem

$$\max_{\alpha}\sum_{(i,J)} \alpha_{iJ}\left[\nu_{iJ}^* - \Omega_{iJ}\left(P_i^{\text{tot}}\right)\right]^+, \text{ subject to } \quad \underline{\alpha}\in\Pi, \tag{16}$$

[1]we neglect supscript and subscript in π_{iJ}^c in the following for brevity. Its meaning should be clear from context.

which becomes a maximum weighted hypergraph matching problem with $[\nu_{iJ}^* - \Omega_{iJ}(P_i^{\text{tot}})]^+$ be the weight of hyperarc (i, J). In [5], several distributed maximum weighted hypergraph matching algorithms are proposed, which can be applied to solve (16) directly.

Remarks:

- The proposed algorithm can be readily modified for congestion control and minimizing the power consumption per bit. In the former case, we need to replace the objective function with $\max \sum_c U_c(\tilde{x}^c)$, where $U_c(\cdot)$ is the utility function for session c. In the latter case, the objective function is chosen as $\min \sum_c \frac{\sum_{(i,J)\in\mathcal{A}} \alpha_{iJ}\lambda_{iJ}^c \Omega_{iJ}(P_i^{\text{tot}})}{\tilde{x}^c}$. The same dual decomposition approach applies to both cases. The only modification is that each packet needs to track the average power consumption during its transmission (for example add the power consumption into the header of each packet). The receiver needs to feed back the average power per packet periodically to the corresponding source.
- If both the power and the transmission rate of each node can be varied, $p_{iJj}(R_{iJ}, P_{iJ}, h_{ij})$ should be substituted into (8) and the same algorithm can be applied. If the function $p_{iJj}(\cdot)$ is known at each node, the power consumption can be minimized by varying R_{iJ} and P_{iJ}.

C. Opportunistic Unicast with XOR Intersession Coding

1) Backpressure algorithm: Each node maintains a queue for each commodity. The lengths of these queues are used to make coding, routing and scheduling decisions. Each node also maintains a *side information buffer* containing decoded packets obtained via transmissions or overhearing. Analogously to Section IV, we choose among possible uncoded transmissions (unicast sessions) and coded transmissions (multicast sessions) at each hyperarc (i, J).

Consider a hyperarc (i, J). Each packet p in the commodity c queue at node i is associated with a *side information set* O_p consisting of those neighbors of i whose side information buffers contain p. Suppose a set $\mathcal{M}$ of packets, each from a different commodity, is coded together and transmitted on (i, J). Then

$$\Gamma_{iJ}^{\mathcal{M}p} = \bigcap_{\{p'\in\mathcal{M}|p'\neq p\}} O_{p'} \cap J - O_p \tag{17}$$

is the set of nodes in J, excluding those in O_p, that can decode packet p if they receive the coded combination. A set $\mathcal{M}$ is a valid coding set (multicast session) iff $\Gamma_{iJ}^{\mathcal{M}p}$ is nonempty for all $p \in \mathcal{M}$.

Let $Q_i^c(t)$ denote the queue length of commodity c at node i at time i. According to [6], [7], the dual variable q_i^c at time t can be written as $q_i^c(t) = \epsilon Q_i^c(t)$, where ϵ is a positive stepsize. The oblivious backpressure algorithm is described in detail in the following:

Algorithm 1: Oblivious backpressure algorithm for joint scheduling and XOR intersession coding

At time t:

• *Initialization:* At each node i, for each hyperarc (i, J), search through the queue of each commodity (or the head of line packets for each commodity) to find the valid coding sets $\mathcal{M}$. Let

$$\nu_{iJ}^{\mathcal{M}} = R_{iJ} \sum_c \sum_{\substack{\{S|S\in 2^J,\\ S\cap\Gamma_{iJ}^{\mathcal{M}p}\neq\emptyset\}}} \eta_{iS} \max_{j\in S\cap\Gamma_{iJ}^{mc}} [q_i^c(t) - q_j^c(t)]^+, \tag{18}$$

$$\nu_{iJ}^{u_c} = R_{iJ} \sum_{S\in 2^J} \eta_{iS} \max_{j\in S} [q_i^c - q_j^c]^+, \tag{19}$$

and $\nu_{iJ}^* = \max\{\max_{\mathcal{M}} \nu_{iJ}^{\mathcal{M}}, \max_{u_c} \nu_{iJ}^{u_c}\}$. Denote s_{iJ}^* as the session attaining ν_{iJ}^* (ties are broken randomly).

• *Link Scheduling:* A distributed hypergraph matching algorithm is executed to solve the scheduling problem (16) with ν_{iJ}^* obtained from the initialization step. If (i, J) is chosen by the matching algorithm or $\alpha_{iJ} = 1$, node i becomes active.

• *Session Scheduling, Network Coding, and Data Transmission:* For each node i that is active, if $\nu_{iJ}^* \geq \Omega_{iJ}(P_i^{\text{tot}})$, it decides to transmit and it then checks s_{iJ}^*. If s_{iJ}^* is a unicast session u_c, node i simply transmits a packet from commodity c at rate R_{iJ}. If s_{iJ}^* corresponds to a coding set (multicast session) $\mathcal{M}$, a coded packet is formed by XOR-ing together the packets in $\mathcal{M}$. The coded packet is then sent at rate R_{iJ}.

• *Packet Reception:* On receiving an uncoded packet from commodity c, for all nodes in J that receive the packet from node i, only the node j with the largest queue length difference $[q_i^c(t) - q_j^c(t)]^+$ puts the packet into its virtual queue corresponding to commodity c and the other nodes put the packet in their side information buffer. Moreover, the node keeping the packet tries to learn which nodes have also received this packet. On receiving a coded packet, for each commodity c in the coded packet, among all nodes in Γ_{iJ}^{mc}, only the node j with the largest queue length difference $[q_i^c - q_j^c]^+$ decodes the packet using overheard packets in its side information buffer. A node drops the packet if the packet is not decoded for any commodity.

• *Queue Update:* In the end, each node updates its queue length Q_i^c and broadcasts it to its neighbors.

The intuition behind the above algorithm is obtained from the dual decomposition in Section V-B. This algorithm is oblivious to overhearing because it makes use of overheard packets whenever possible but it does not optimize over overheard flows as indicated by (18) and (19). Note that this algorithm only requires nodes to communicate with direct neighbors. Thus, it is a desired distributed algorithm.

In Algorithm 1, having each node check all the packets in the virtual queue of commodity c to get all the possible coding sets causes two problems in practice. First, packets may be reordered because the scheduling prefers to transmit packets which are overheard by many nodes rather than to transmit the head of the queue. Second, it is complicated to search through the queue. Therefore, in practice the algorithm only checks the head of each virtual queue for potential coding sets.

In the packet reception component of Algorithm 1, we have assumed that both the node with overheard packet and the node performing XOR coding receives the packet from a common one-hop neighbor. For example, in Fig. 2, both node 3 and node 4 receive b_1 from the common one hop neighbor node 1. In this case, overheard packets are from nodes one hop away from the node performing XOR coding. We call this overhearing scenario as *one-hop overhearing*. But, in Fig. 2, node 4 may overhear b_1 from another node 8 not from node 1. In this case, overheard packets are from nodes more than

one hop away from the node performing XOR coding. We call this overhearing scenario as *multi-hop overhearing*. The derived scheduling policy can also be applied to multi-hop overhearing as long as each node knows that for each packet in its queue what other nodes have overheard this packet. We will discuss possible implementations for the proposed algorithms in Section VI.

2) Achievable Rate Region: We consider the achievable rate region when decisions are made taking into account overhearing probabilities. We take a fluid model approach. For each hyperarc (i, J), we consider as part of a single multicast session m all coding sets $\mathcal{M}$ whose corresponding packets p of each commodity c have the same side information sets $O_p = \Upsilon_{iJ}^{mc}$.

As (8), when node i only transmits packets from unicast session u_c, we have the following constraints

$$\sum_{j \in S} z_{iJj}^{S,u_c} = \alpha_{iJ}\eta_{iS}R_{iJ}, \forall S \in 2^J, \tag{20}$$

where $\alpha_{iJ}, \eta_{iS}, R_{iJ}, S$ are defined as in Section III-A. Note that in (20) we assume that the flow of commodity c is intended to all nodes in J. But we can restrict potential forwarders to be a subset of nodes $J^1 \subseteq J$ and all the nodes in $J^0 = J - J^1$ only overhear traffic from node i without forwarding. In this case, $S \in 2^J$ in (20) should be replaced by $S \in 2^{J^1}$. We adopt (20) in the following.

When node i only transmits packets from multicast session m over (i, J), we have

$$z_{iJj}^{mc} = \sum_{\{S|S\in 2^J,\, j\in S\cap\Gamma_{iJ}^{mc}\}} z_{iJj}^{S,mc}, \forall i,\, j \in J, \tag{21}$$

$$\sum_{j\in S\cap\Gamma_{iJ}^{mc}} z_{iJj}^{S,mc} = \alpha_{iJ}\eta_{iS}R_{iJ}, \forall S \in 2^J,\, c, \Upsilon_{iJ}^{mc} \neq \emptyset. \tag{22}$$

Note that (22) holds because with one-hop XOR network coding different commodities in each multicast session can share the capacity of each hyperarc. Let λ_{iJ}^m and $\lambda_{iJ}^{u_c}$ denote the fraction of time node i transmits packets from multicast session m and from unicast session u_c. By time sharing among all the sessions $f_{iJj}^{mc} = \lambda_{iJ}^m z_{iJj}^m$ is flow rate for session m and $f_{iJj}^{S,u_c} = \lambda_{iJ}^{u_c} z_{iJj}^{S,u_c}$ is flow rate for session u_c.

Let $g_{iJj}^{O_c}$ denote the total flow of commodity c over (i, J) received by node j, and is exactly overheard by nodes in O_c. If we have a virtual queue for each commodity c and each side information set O_c at each node, $g_{iJj}^{O_c}$ is the flow contribution from node i to the virtual queue corresponding to commodity c and side information set O_c at node j. Note that in unicast session u_c node i sends f_{iJj}^{S,u_c} to node j, where $S \subseteq J$ is defined to be the set of nodes that exactly receive a packet transmitted by node i successfully. All nodes in S overhear the flow f_{iJj}^{S,u_c}. Node i can also keep the flow f_{iJj}^{S,u_c} after sending it. Therefore, we have

$$f_{iJj}^{S,u_c} = g_{iJj}^{O_c},\, O_c = \{i\} \cup S - \{j\}. \tag{23}$$

Here, we do not make use of overheard coded packets as in [1]. As the forwarding node i has all the packets in the coded packet, in this case, the side information nodes set of any multicast traffic is simply $\{i\}$. We thus have

$$\sum_m f_{iJj}^{mc} = g_{iJj}^{O_c},\, O_c = \{i\}. \tag{24}$$

Denote θ_{iJ}^{mc} as the probability that at least one node in Γ_{iJ}^{mc} receives the coded packet from i when session m is active at i, and denote $\theta_{iJ}^{u_c}$ as the probability that at least one node in J receives the uncoded packet from i when session u_c is active at i. Transmitting flow from multicast session m will consume $\lambda_{iJ}^m \theta_{iJ}^{mc} \alpha_{iJ} R_{iJ}$ packets in the virtual queue corresponding to the side information set O_c if $\Upsilon_{iJ}^{mc} = O_c$. We then have the following flow constraints at node i:

$$\sum_{j\in\mathcal{N}} \sum_{\{I|(j,I)\in\mathcal{A},\, i\in I\}} g_{jIi}^{O_c} = \sum_{(i,J)\in\mathcal{A}} \left(\sum_{\{m|\Upsilon_{iJ}^{mc}=O_c\}} \lambda_{iJ}^m \theta_{iJ}^{mc} \right) \alpha_{iJ} R_{iJ} + y_i^{O_c}, \tag{25}$$

$$\sum_{O_c', O_c' \neq \emptyset} y_i^{O_c'} + \tilde{\sigma}_i^c = \sum_{(i,J)\in\mathcal{A}} \lambda_{iJ}^{u_c} \theta_{iJ}^{u_c} \alpha_{iJ} R_{iJ}, \tag{26}$$

where the left-hand side of (25) is the total incoming flow with side information set O_c at node i, which may contribute to the traffic of multicast sessions constituting O_c (the first term of the right-hand side of (25)) and contribute to the traffic of unicast session u_c (the second term of the right-hand side of (25)); $y_i^{O_c} \geq 0$ is the amount of flow with side information set O_c transmitted by unicast session u_c. The left-hand side of (26) is the total available flow for outgoing unicast session u_c at node i and the right-hand side of (26) is the total outgoing flow of unicast session u_c at node i; $\tilde{\sigma}_i^c$ is defined in (2).

The achievable rate region is defined by the constraints (20)-(26). Similar to unicast without network coding (8), we can formulate unicast with intersession network coding as

$$\begin{aligned} \min_{f,\lambda,\alpha} \quad & \sum_{(i,J)\in\mathcal{A}} \alpha_{iJ} t_{iJ} \Omega_{iJ} \left(P_i^{\text{tot}}\right) \\ \text{subject to} \quad & (20) - (26), \\ & t_{iJ} = \sum_m \lambda_{iJ}^m + \sum_{u_c} \lambda_{iJ}^{u_c},\, 0 \leq t_{iJ} \leq 1,\, \underline{\alpha} \in \Pi. \end{aligned} \tag{27}$$

3) Intuition and Performance Evaluation: We now show the algorithmic intuition behind Algorithm 1. It is difficult to solve (27) directly. For complexity reasons, in algorithm 1 we do not try to optimize the overhearing nodes, and the possible multicast sessions m are determined by network conditions. For example, at each time slot, each node knows what other nodes have overheard each packet in its queues, which can be achieved via reception reports as in [1], or piggybacking on other packets, or through the special packet acknowledgement mechanism in Section VI, and thus it can construct all possible multicast sessions m as in Algorithm 1. After summing (25) over all $O_c \neq \emptyset$ and combining with (26), and substituting $g_{iJj}^{O_c}$ in (23) and (24) into the resulting equation, we have the following modified problem

$$\begin{aligned} \min_{f,\lambda,\alpha} \quad & \sum_{(i,J)\in\mathcal{A}} \alpha_{iJ} t_{iJ} \Omega_{iJ} \left(P_i^{\text{tot}}\right) \\ \text{subject to} \quad & \sum_{\{J|(i,J)\in\mathcal{A}\}} \sum_{j\in J} \left(\sum_m f_{iJj}^{mc} + f_{iJj}^{u_c} \right) \\ & - \sum_{j\in\mathcal{N}} \sum_{\{I|(j,I)\in\mathcal{A},\, i\in I\}} \left(\sum_m f_{jIi}^{mc} + f_{jIi}^{u_c} \right) = \tilde{\sigma}_i^c, \\ & f_{iJj}^{mc} = \lambda_{iJ}^{mc} z_{iJj}^{mc},\, f_{iJj}^{u_c} = \lambda_{iJ}^{u_c} z_{iJj}^{u_c}, \\ & (20) - (22), \\ & t_{iJ} = \sum_m \lambda_{iJ}^m + \sum_{u_c} \lambda_{iJ}^{u_c},\, 0 \leq t_{iJ} \leq 1,\, \underline{\alpha} \in \Pi. \end{aligned} \tag{28}$$

Note that (28) is similar to (8) and it can also be solved following the same way as in Section V-B, which gives the oblivious backpressure algorithm at the beginning of this section. In general (28) is not equivalent to (27). Thus, the oblivious backpressure algorithm is not optimal to solve (27), and its achievable rate region is hard to obtain. We can however bound its performance as follows.

Let $\rho_i^c(t) = \sigma_i^{\tilde{u}_c} - \sum_{(i,J)} \sum_{j\in J} \left(\sum_m f_{iJj}^{mc} + f_{iJj}^{u_c}\right) + \sum_j \sum_{\{I|(j,I)\in\mathcal{A}, i\in I\}} \left(\sum_m f_{jIi}^{mc} + f_{jIi}^{u_c}\right)$ and $\underline{\rho}(t) = (\rho_i^c(t))$. If there exists G such that $\|\underline{\rho}(t)\|_2 \leq G$ for all t, we have the following theorem on the performance of Algorithm 1.

Theorem 1: Assume that $\tilde{x}^c$ lies in the rate region defined by the constraints in (8). Let $C(t)$ be the total power cost of the network at time t by using Algorithm 1, C_{R}^* be the optimal cost of (8) without network coding, and C_{XOR}^* be the optimal cost of (27) with intersession network coding. We have the following inequality

$$C_{\mathrm{XOR}}^* \leq \limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=1}^{t} E\{C(\tau)\} \leq C_{\mathrm{R}}^* + \frac{\epsilon G^2}{2}. \tag{29}$$

Proof (Outline): The proof basically follows that in [6]. The first inequality is trivial as the C_{XOR}^* is the optimal cost with XOR coding. Let $\underline{Q}(t) = (Q_i^c(t))$ and $\underline{q}(t) = (q_i^c(t))$. By the definition of $Q_i^c(t)$ and its relationship with $q_i^c(t)$, we have

$$\begin{aligned}
&E\left\{\|\underline{q}(t+1)\|_2^2|\underline{q}(t)\right\} \leq E\left\{\|\underline{q}(t) - \epsilon\underline{\rho}(t)\|_2^2|\underline{q}(t)\right\}\\
=&E\left\{\|\underline{q}(t)\|_2^2 - 2\epsilon C(t) + 2\epsilon C(t) - 2\epsilon\underline{\rho}^T(t)\underline{q}(t) + \epsilon^2\|\underline{\rho}(t)\|_2^2|\underline{q}(t)\right\}\\
\leq&\|\underline{q}(t)\|_2^2 - 2\epsilon E\{C(t)\} + 2\epsilon C_{\mathrm{R}}^* + \epsilon^2 G^2,
\end{aligned}$$

where in the last inequality we have used the fact that Algorithm 1 solves (14) and (16) optimally and the weight of each hyperarc by using Algorithm 1 is at least as large as that without network coding. Taking expectation on both sides of the above equation over $\underline{q}(t)$ and applying the resulting equation recursively, we obtain

$$E\left\{\|\underline{q}(t+1)\|_2^2\right\} \leq E\left\{\|\underline{q}(1)\|_2^2\right\} - 2\epsilon\sum_{\tau=1}^{t}\left(E\{C(\tau)\} - C_{\mathrm{R}}^*\right) + t\epsilon^2 G^2.$$

Since $E\left\{\|\underline{q}(t+1)\|_2^2\right\} \geq 0$, we obtain

$$\frac{1}{t}\sum_{\tau=1}^{t} E\{C(\tau)\} - C_{\mathrm{R}}^* \leq \frac{E\left\{\|\underline{q}(1)\|_2^2\right\} + t\epsilon^2 G^2}{2t\epsilon}.$$

By taking $t\to\infty$, we obtain the second inequality in (29). □

Remarks:

- If we apply Algorithm 1 to the downlink of a cellular network where a base station transmits data to K mobile users and assume all the users have the same queue length at base station all the time, it reduces to the scheduling algorithm in [17], which allows only the user with the best channel to transmit at any time.
- In this section, we focus on minimizing power consumption given fixed input rates. As in [5]–[7], our proposed algorithms can be readily extended to congestion control by maximizing a utility function.
- Note that algorithms proposed in this section can also be applied to applications with predetermined routes, such as shortest path.

Fig. 3. Data and ACK frame formats in COPR.

VI. COPR Protocol Design

In this section, we consider protocol design by using the opportunistic backpressure algorithms derived in Section V-C in synchronous slotted wireless networks using CDMA or FDMA. A protocol called COPR (Coding with Opportunistic Reception) is developed. We briefly outline the features of COPR. Details of COPR can be found in [18]. We first consider COPR with one-hop overhearing. The medium access control (MAC) layer of COPR runs on top of 802.11b MAC. Each time-slot is partitioned into three phases: contention period, data transmission, and packet acknowledgement.

Contention Period: At the beginning of each time slot, one of distributed hypergraph matching algorithms in [5] is executed during the contention period. 802.11b MAC is used to resolve contention during hypergraph matching.

Data Transmission: Data transmission, session scheduling, and network coding follow Algorithm 1. Data frame format is depicted in Fig. 3, which follows that in 802.11 standards. New `Pkt_Num`, `Pkt_ID`, and `Next_Hops` fields are added before the frame body, where `Pkt_Num` indicates the number of native packets XOR-ed together, `Pkt_ID` is the ID of one of the native packets, and `Next_Hops` includes the MAC addresses of all possible next hop nodes corresponding to the native packet with `Pkt_ID`. The addresses in `Next_Hops` are in decreasing order of queue difference. The maximum number of next hop nodes for each `Pkt_ID` is denoted as `max_next_num`.

The complexity of finding packets to XOR-ed together is exponential in the number of commodities. In [18], several low complexity and suboptimal algorithms are proposed.

Packet Acknowledgement: COPR reserves multiple acknowledgement slots. If a node hears a packet and it checks that its address is the j-th address in `Next_Hops` corresponding to the i-th `Pkt_ID`, it decodes the coded packet to obtain `Pkt_ID` and then waits for `SIFS`$+((i-1)\cdot$`max_next_num`$+j-1)\cdot$(`ack_tx_time`$+$`SIFS`) before sending its acknowledgement, where `ack_tx_time` is the time to transmit an ACK packet and SIFS denotes the short interframe space in 802.11. The ACK frame format is shown in Fig. 3. A new `Rp_Flag` field of `max_next_num` bits is added before the CRC. If the j-th bit in `Rp_Flag` is 1, it indicates that the j-th node in `Next_Hops` also receives `Pkt_ID`. Each node maintains a vector $\underline{r}$ of length `max_next_num`. At the beginning of each time slot, it sets $\underline{r}$ as an all zero vector. If it receives a packet and its address is the j-th address in `Next_Hops` corresponding to the `Pkt_ID`, it sets the j-th entry of $\underline{r}$, r_j to be 1. During packet acknowledgement period, whenever it overhears an ACK packet, it checks whether the `Pkt_ID` of this ACK is equal to the `Pkt_ID` it receives. If yes, it sets r_i to 1 if the i-th entry of `Rp_Flag` in the ACK is 1 (simply OR `Rp_Flag` and $\underline{r}$). At end of each time slot, if $r_i = 1$, it means that the i-th node in `Next_Hops` has also received the `Pkt_ID`. To get a tradeoff between performance improvement and overhead of sending ACK, in [18], we find the optimal value of `max_next_num` is 3 by using 802.11b parameters in fast Rayleigh fading channels. COPR also allows

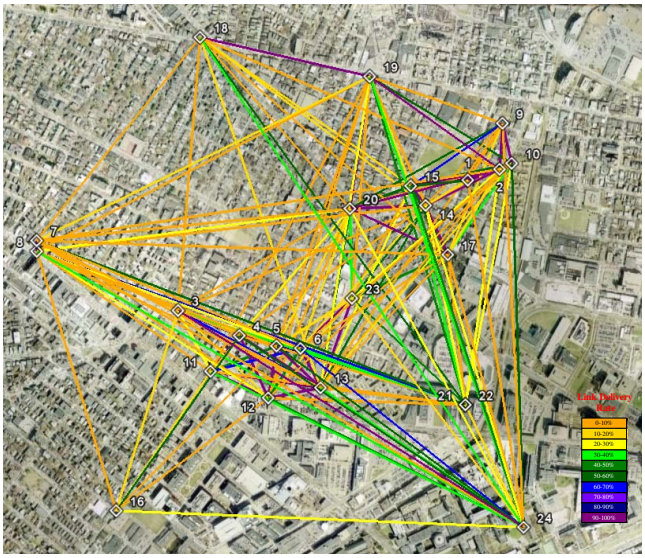

Fig. 4. Connectivity map of Roofnet in August 2004, with a diamond icon for each of the 24 nodes that participated in the experiments in this section (the .kml file of this map for Google earth is available online at `http://www.its.caltech.edu/~taocui/roof_net.kml`).

nodes that are not added into the hypergraph matching to overhear packets. In this case, reception report still needs to send as COPE. But from experiments, we find this case rarely happens.

Multiple-hop overhearing may be exploited to improve the network performance. We create a new field, Overheard_Node, in the header of a packet and put into the addresses of nodes that have overheard this packet. To reduce the overhead of this approach, we only keep those nodes that are K hops away from current node, or the number of overhearing node addresses is set to be $K\cdot$`max_next_num`. A tradeoff between overhead and performance can be attained through K.

VII. Experimental Results

In this section, we study the performance of different algorithms via packet level simulation on Roofnet [19], whose connectivity map in August 2004 is shown in Fig. 4. Even though Roofnet is not designed for energy efficient communication, we use this network to evaluate different algorithms because of the availability of experimental data on link loss rates in [19], which is labeled with different colors in Fig. 4. All nodes use identical 802.11b cards based on at Conexant (formerly Intersil) PRISM 2.5 chip-set. The cards are configured to transmit at 1 Mbit/s using BPSK modulation. The transmission power of each node is set to be 23 dBm. Each packet is of length 1 KB. Different algorithms are compared using average consumed power per bit. Each node starts with enough energy so that it will not run out of its energy during the simulations. Only power consumption during data transmission and ACK transmission is taken into account. Power consumption during contention period is not counted. The link delivery rate is assumed to be known at each node. We compare 4 different kinds of algorithms. The first one, called COPE_bp, employs the COPE opportunistic network coding algorithm with session scheduling performed by using the backpressure algorithm derived in [18], which is equivalent to setting `max_next_num`=1 in COPR, and with routing Djikstra's shortest path algorithm based on the ETT metric [20]. As this algorithm is different from COPE in [1], it is denoted as COPE_bp. The second one, COPR, performs session scheduling by searching only the head of queue[2]. The third one, opportunistic unicast without network coding, is denoted as ORouting (Note that this is different from the ExOR in [11]). The last one is simply shortest path routing.

As the network is large, it is difficult to find the maximum weighted hypergraph matching. Instead, in COPR, we use distributed greedy hypergraph matching algorithms in [5]. COPR_gdy1 chooses a locally heaviest hyperarc every time, which COPR_gdy2 chooses a locally heaviest hyperarc discounted by the size of the hyperarc every time. COPR with multi-hop overhearing is denoted as COPR_multi, which keeps in each packet the nodes overheard this packet in previous 3 hops. We choose $\epsilon = 0.025$ in all algorithms. U unicast sessions exist in the network, where each session picks sender and receiver randomly. For each U, 20 realizations of unicast sessions are generated. A 1 megabyte file is split evenly among all the U sessions and each session transmits one piece. The average power per bit is obtained after transferring the file over UDP, which is insensitive to packet losses and packet reordering. As backpressure based algorithms usually need to learn efficient routes first, we first transmit another 1 megabyte file before transmitting the test file. The realizations with the maximum, the minimum, and the median power saving by using COPR_gdy1 over that by using Routing are used to compare different algorithms.

Fig. 5 shows performance of different algorithms on Roofnet with 5 and 15 unicast sessions. For 5 unicast sessions, on average, ORouting achieves an 11.45% power saving over Routing for all 20 realizations, while this number increases to 12.59% by using COPR_gdy1. Interestingly, COPR_gdy2 performs worse than COPR_gdy1, which shows that energy efficient applications prefer large hyperarcs. By using COPR_multi, an additional 1.24% power saving can be attained. In the maximum power saving case, the average percentage of encoded packets at all nodes is 12.52% using COPR_gdy1 and is 13.53% with COPR_multi. COPE_bp only achieves a 1.43% power saving over Routing on average. In the maximum power saving case, on average, 9.11% packets are coded with COPE_bp. Similar results hold for 15 unicast sessions. The power saving by using ORouting and COPR_gdy1 over that by Routing becomes to 10.08% and 13.60%, respectively. By using COPR_multi, the gain increases to 14.59%. COPE_bp also attains a 2.27% gain. In the maximum power saving case, 28.94%, 30.49%, and 8.40% packets are coded together with COPR_gdy1, COPR_multi, and COPE_bp, respectively, and the power savings are 23.40%, 24.37%, and 8.11%, respectively. By increasing the number of sessions from 5 to 15, we find that the percentage of packets coded together increases from 7.67% to 19.98%. Therefore, increasing the number of sessions creates more coding opportunities. We do not find any 3 packets are XORed together in all 5-session realizations while only 0.02% of coded packets are XORed with 3 packets in all 15-session realizations. Opportunistic network coding with shortest path

[2]We do not compare with COPR by searching through the queue at each node as it is very complicated for a large network with many sessions. But from our simulation with 5 nodes wireless butterfly network, we indeed find that this scheme can improve performance especially in a very lossy network.

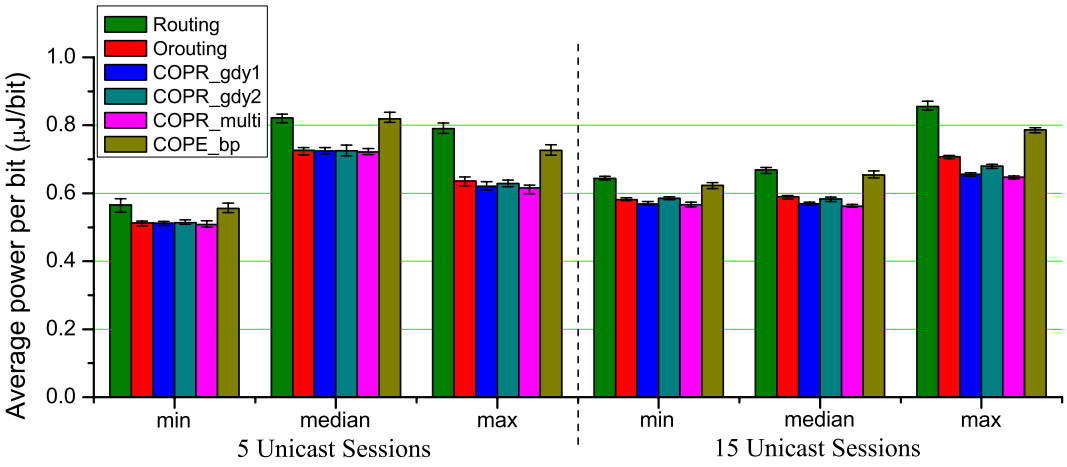

Fig. 5. Average power consumption per bit of different algorithms in Roofnet with 4 unicast sessions and 8 unicast sessions.

routing does not find many coding opportunities partly because the number of sessions is small in our simulation. In [1], a large number of sessions exist in a network results in high congestion and many coding opportunities. On the other hand, the proposed algorithm still works even though the number of sessions is small.

Our experimental results suggest the following:

- Backpressure based algorithms create more coding opportunities than shortest path based algorithms in low congestion and energy-constrained settings;
- Coding gain increases with the number of sessions.
- Multi-hop overhearing seems to be not helpful. It is good enough to use one-hop overhearing with reduced overhead.

VIII. Conclusion

In this paper, we have investigated energy efficient backpressure algorithms by exploiting multiple reception gain in wireless networks. Based on optimization framework, backpressure algorithms are proposed for unicast without network coding and XOR intersession network coding. Link scheduling problem is found to be a maximum weighted hypergraph matching problem, which can be solved distributedly by using the algorithms in [5]. The optimal session scheduling algorithm requires searching through all the queues at each node. To reduce the complexity of session scheduling, a suboptimal algorithm is proposed to search only the head of queues at each node. By using proposed algorithms, a COPR protocol is proposed for unicast with XOR intersession network coding. COPR uses a specially designed MAC, which runs on top of 802.11b MAC. Packets' format and parameters' settings are also discussed for COPR. Our experimental results show that COPR achieves up to 25% power saving over pure routing. As a future work, it is interesting to develop algorithms to reduce complexity of computing (18) and (19) in session scheduling.

References

[1] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: Practical wireless network coding," in *Proc. of ACM SIGCOMM*, Oct. 2006, pp. 243–254.

[2] 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. of IEEE ISIT*, July 2006, pp. 1758–1762.

[3] T. Ho, Y. Chang, and K. J. Han, "On constructive network coding for multiple unicasts," in *Proc. of Allerton Conf. on Comm., Contr. and Comput.*, Sept. 2006.

[4] A. Eryilmaz and D. S. Lun, "Control for inter-session network coding," in *Proc. of the Workshop on Network Coding, Theory and Applications*, 2007.

[5] T. Cui, L. Chen, and T. Ho, "Distributed optimization in wireless networks using broadcast advantage," in *Proc. of IEEE CDC*, Dec. 2007.

[6] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *Proc. of IEEE Infocom*, Apr. 2006.

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

[8] M. Neely, E. Modiano, and C. 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.

[9] M. Neely, "Energy optimal control for time varying networks," in *Proc. IEEE Infocom*, Apr. 2005.

[10] ——, "Optimal backpressure routing for wireless networks with multi-receiver diversity," in *Proc. of CISS*, Mar. 2006, pp. 18–25.

[11] S. Biswas and R. Morris, "ExOR: Opportunistic routing in multi-hop wireless networks," in *Proc. of ACM SIGCOMM*, Aug. 2005, pp. 133–144.

[12] D. S. Lun, N. Ratnakar, M. Médard, R. Koetter, D. R. Karger, T. Ho, E. Ahmed, and F. Zhao, "Minimum-cost multicast over coded packet networks," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2608–2623, June 2006.

[13] L. Chen, T. Ho, S. H. Low, M. Chiang, and J. C. Doyle, "Rate control for multicast with network coding," in *Proc. of IEEE Infocom*, Apr. 2007.

[14] T. Ho and H. Viswanathan, "Dynamic algorithms for multicast with intra-session network coding," in *Proc. of Allerton Conf. on Comm., Contr. and Comput.*, Sept. 2005.

[15] S. Sengupta, S. Rayanchu, and S. Banerjee, "An analysis of wireless network coding for unicast sessions: The case for coding-aware routing," in *Proc. of IEEE Infocom*, May 2007, pp. 1028–1036.

[16] D. S. Lun, M. Médard, and R. Koetter, "Efficient operation of wireless packet networks using network coding," in *Proc. International Workshop on Convergent Technologies*, Jun. 2005.

[17] R. Knopp and P. A. Humblet, "Information capacity and power control in single-cell multiuser communications," in *Proc. of IEEE ICC*, June 1995, pp. 331–335.

[18] T. Cui, L. Chen, and T. Ho, "Energy efficient opportunistic network coding for wireless networks," Caltech, Tech. Rep., July 2007.

[19] Roofnet, http://pdos.csail.mit.edu/roofnet/.

[20] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and evaluation of an unplanned 802.11b mesh network," in *Proc. of ACM MobiCom*, 2005, pp. 31–42.