An Iterative Routing Algorithm for Energy Minimization in Coded Wireless Networks

Linyu Huang
Dept. of Electronic Engineering
City University of Hong Kong
linyhuang2@student.cityu.edu.hk

Chi Wan Sung
Dept. of Electronic Engineering
City University of Hong Kong
albert.sung@cityu.edu.hk

***Abstract*—Energy saving is important for many wireless devices. In a multi-hop wireless network with multiple sessions, XOR network coding can be applied to opposite traffic flows so as to reduce the number of packet transmissions, which in turn reduce transmission energy. Such a change in packet forwarding, however, impacts the design of traffic routing. Traditional routing algorithms, which typically aim at finding shortest paths between source and destination nodes, may no longer work well. In this paper, an iterative routing algorithm is proposed, which favors paths that can provide more pair-wise XOR network coding opportunities. Simulation results show that this algorithm integrates well with the XOR forwarding method and can reduce energy cost significantly when compared with traditional shortest-path routing, with and without network coding.**

I. Introduction

Energy efficiency is an important issue in the design of wireless networks. For some networks powered by batteries, replacement or re-charge of batteries are very difficult or even impossible. In such cases, how to use energy in an efficient way so that network lifetime can be prolonged is a big concern.

One way to improve energy efficiency is to apply XOR network coding to duplex flows. XOR network coding was first proposed in [1]. Its idea can be easily illustrated by the classical "Alice-Bob" information exchange example in [2]. By taking advantage of the broadcast nature of wireless medium, XOR network coding can reduce the total number of transmissions in information exchange between network entities. In some scenarios, when the total traffic of a network is fixed, reducing total number of transmissions is equivalent to reducing the total energy consumption.

In a general adhoc network, it is necessary to determine one or more paths from a source node to its destination nodes. When network coding is used, traditional routing algorithms may no longer be efficient. How routing should be done in a network-coded system has been studied by some researchers. In [3], Sengupta *et al.* considered the routing problems for a special XOR network coding system named COPE [2]. Lun *et al.* considered multicast routing problems focusing on minimum-cost in [4], where cost is a function related to average latency or energy consumption. Reddy *et al.* in [5] proposed multipath routing algorithm to find the splits of traffic for each source across its multiple paths in a distributed manner. ParandehGheibi *et al.* proposed an optimal multipath routing algorithm in [6]. Most of them focus on multicast problem or multipath routing.

Although multi-path routing may perform better in both throughput and reliability, single-path routing is preferable in some practical systems, since multi-path routing has a higher overhead in establishment and management of multiple paths than single-path routing does [7]. For example, when multi-path routing is used, it is hard to set the retransmission timer, as packets traversing different paths typically have different delays. Congestion control is also more difficult to perform, as it is likely that some paths are lightly loaded while others are congested. Besides, packet reassembly at destination is needed, which increases delay and incurs more control overhead.

In this paper, we consider the single-path routing problem in multi-hop XOR-coded wireless unicast networks, with the objective of minimizing total energy cost. We propose an iterative routing algorithm, which is applicable to general wireless network topologies. Simulation results show that our proposed algorithm can reduce total energy up to 32% when compared with shortest path routing without network coding.

II. Model and Problem Formulation

A. Network Model

The topology of a wireless network can be represented by an undirected graph $G(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of $|\mathcal{V}|$ network nodes and $\mathcal{E}$ is the set of $|\mathcal{E}|$ undirected edges. We assume each node has an infinite storage capacity. An edge $E_{ij} = (n_i, n_j) \in \mathcal{E}$, where $n_i, n_j \in \mathcal{V}$, is a wireless link between node n_i and node n_j. We decompose E_{ij} into two virtual directed edges $e_{ij} = \langle n_i, n_j \rangle$ and $e_{ji} = \langle n_j, n_i \rangle$, where e_{ij} denotes the transmission link from the initial node n_i to the terminal node n_j and e_{ji} denotes that from n_j to n_i. Each link e_{ij} has a capacity denoted by C_{ij}. Each packet transmission at node n_i takes a cost of ε_1 energy units for coded packet and ε_2 energy units for non-coded packet. Note that the ε_1 units of energy cost include the energy for packet transmission and for packet encoding at n_i, and the energy for packet decoding at the neighbors of n_i. Since packet encoding and decoding are not required for non-coded packets, we assume $\varepsilon_1 > \varepsilon_2$.

When $\varepsilon_1 \geq 2\varepsilon_2$, XOR network coding has no advantage in reducing energy cost. Therefore, in this paper, we only

This work was partially supported by a grant from the University Grants Committee of the Hong Kong Special Administrative Region, China (Project No. AoE/E-02/08).

consider the case where $\varepsilon_2 < \varepsilon_1 < 2\varepsilon_2$.

We assume the network supports some concurrent unicast flows $\mathcal{F} = \{f_1, f_2, f_3, \dots\}$, where each flow f_i is associated with a source s_i and a destination d_i. Each flow f_i supports a transmission rate denoted by r_i, which is assumed to be an integer. A path for flow f_i, denoted by $\mathcal{P}_i$, is defined as an ordered set of vertices $\{s_i, n_1, n_2, n_3, \dots, n_k, d_i\}$. Equivalently, a path can also be regarded as an ordered set of edges, with s_i being the initial node of the first edge and d_i the terminal node of the last edge. For each flow f_i, we assume there exists at least one path that can lead s_i to d_i, for otherwise f_i cannot be supported. Given a path $\mathcal{P}_i$ for flow f_i, we define an indicator $I_i(e)$ to represent whether edge e is used. If it is used, $I_i(e)$ is set to 1; otherwise, $I_i(e)$ is set to zero.

B. Coding Rules and Problem Formulation

When two consecutive edges are simultaneously in two or more paths and at least one path has different direction from others, XOR network coding can be applied. Consider the following situation: The edges $e_{(k-1)k}$ and $e_{k(k+1)}$ are used by path $\mathcal{P}_i$, and $e_{(k+1)k}$ and $e_{k(k-1)}$ are used by another path $\mathcal{P}_m$, i.e., $I_i(e_{(k-1)k})I_i(e_{k(k+1)}) = 1$ and $I_m(e_{(k+1)k})I_m(e_{k(k-1)}) = 1$. A network coding opportunity arises at node n_k and XOR network coding can be used. In general, $r_i \neq r_m$, which means that not all the packets can be coded at node n_k. Suppose $r_i > r_m$. During a period t, the number of packets from node n_{k-1} to node n_k is $r_i t$, which is larger than that from n_{k+1} to n_k, $r_m t$. In this case, all the packets from n_{k+1} to n_k can be coded at node n_k. But for the remaining $(r_i - r_m)t$ packets from n_{k-1} to n_k, they cannot be coded and will be forwarded directly to node n_{k+1}. Thus, during the period t, the total energy cost at node n_k is $\min(r_i, r_m)t\varepsilon_1 + |r_i - r_m|\, t\varepsilon_2$.

At node n_k, the packets can be divided into three types: 1) new packets generated by node n_k, 2) packets absorbed by n_k, and 3) relayed packets which flow through n_k. For type 1) packets, they can only be forwarded to the next hop without network coding. Hence, the energy cost for each packet is simply ε_2. For the second type, their energy costs (for decoding) are included in their transmissions from the previous node. Hence, the total energy cost of the network can be calculated as the summation of the energy used to transmit newly generated packets and the energy used to relay packets, where the summation is carried over all nodes in the network.

The energy cost of relayed packets at n_k can be calculated as follows: Denote the set of neighboring nodes of n_k by $\mathcal{N}_k$. Consider n_k and any two nodes from $\mathcal{N}_k$, say n_m and n_l. Denote the total traffic rate from n_m to n_l through n_k by R_{mkl}. It can be expressed as

$$R_{mkl} = \sum_{f_i \in \mathcal{F}} I_i(e_{mk}) I_i(e_{kl}) r_i. \tag{1}$$

The total cost $\mathbb{E}_{mkl}$ of link $\{E_{mk}, E_{kl}\}$ during one time unit can be expressed as:

$$\mathbb{E}_{mkl} = \min(R_{mkl}, R_{lkm})\varepsilon_1 + |R_{mkl} - R_{lkm}|\, \varepsilon_2. \tag{2}$$

Thus, the total energy cost $\mathbb{E}_k$ for relaying at node n_k can be expressed as:

$$\mathbb{E}_k = \sum_m \sum_l \mathbb{E}_{mkl}, \qquad m, l \in \mathcal{N}_k \text{ and } m < l. \tag{3}$$

For all the source nodes, the total energy used to transmit the newly generated packets in one time unit is:

$$\mathbb{E}_{new} = \sum_{f_i \in \mathcal{F}} r_i \varepsilon_2. \tag{4}$$

Thus, we can express the total energy cost of the network during one time unit as follows:

$$\mathbb{E}_{total} = \sum_{k \in \mathcal{V}} \mathbb{E}_k + \mathbb{E}_{new} = \sum_{k \in \mathcal{V}} \mathbb{E}_k + \sum_{f_i \in \mathcal{F}} r_i \varepsilon_2. \tag{5}$$

Our goal is to determine a path $\mathcal{P}_i$ for each of the flows $f_i \in \mathcal{F}$ so as to minimize the total energy cost $\mathbb{E}_{total}$. In other words, we want to solve the following optimization problem:

$$\min(\mathbb{E}_{total} = \sum_{k \in \mathcal{V}} \mathbb{E}_k + \sum_{f_i \in \mathcal{F}} r_i \varepsilon_2), \tag{6}$$

subject to

$$\sum_{f_i \in \mathcal{F}} I_i(e_{mk}) r_i \leq C_{mk}, \qquad \forall e_{mk} \in \mathcal{E}, \tag{7}$$

Since network coding is applied to multiple flows, any change of a path will affect the optimality of other paths. Hence, it is hard to find a global optimal routing solution. In this paper, we propose a suboptimal algorithm to solve the above problem. The idea is to sequentially choose a path for each flow, assuming that the paths for some flows have already been chosen. We try to create more network coding opportunities so that the energy cost for the corresponding flow can be minimized. This method will be described in the next section. Then we apply the same procedure iteratively for each flow, until no more improvement can be made. It is shown in Section IV that the algorithm always converges.

III. Admission of Single Flow

In this section, we consider the problem of admitting a new flow, f_i, into a network in which some flows are already there. We assume that the paths for existing flows have already been determined and cannot be changed. To minimize the total energy cost $\mathbb{E}_{total}$, we need to find a path which adds minimum energy cost to $\mathbb{E}_{total}$ after admitting f_i. We are going to show that it is equivalent to the shortest-path problem.

Before doing that, we first remove from the graph those edges that cannot afford the traffic rate of f_i. For edge e_{jk}, if the existing traffic on e_{jk} plus r_i is larger than its capacity C_{jk}, it will be temporarily deleted. If there does not exist any path from s_i to d_i, then f_i cannot be admitted. Otherwise, we apply a modification of Dijkstra's algorithm, to be described below, to find a minimum-cost path for f_i.

A. Edge Weight

Consider the flow f_i with rate r_i. Let the source node of this flow be n_0. For $j \in \mathcal{N}_0$, the weight of e_{0j}, denoted by w_{0j}^0, is defined as $r_i\varepsilon_2$, since network coding is not possible at n_0. For the weights of other edges, the definition is more convoluted, since the weight of an edge e_{kl} depends not only on the transmission of n_k, but also on the predecessor node of n_k, which determines whether network coding is possible.

Consider a candidate path, $\mathcal{P}$, for flow f_i. Suppose $\mathcal{P}$ consists of the edges e_{mk} and e_{kl}. Then the power consumption cost at n_k of transmitting a packet from n_m via n_k to n_l can be calculated and denoted as w_{kl}^m. This value can be regarded as the weight of e_{kl} given that the predecessor node is n_m. In other words, each edge has a set of weights, depending on which node is the predecessor node in a given path.

Detailed calculations of w_{kl}^m is shown as follows. Suppose some existing flows are transmitting packets from n_m via n_k to n_l with total rate r_1, and some flows are transmitting packets from n_l via n_k to n_m with total rate r_2. Let $\delta = r_2 - r_1$ be the rate of the reverse traffic.

$$w_{kl}^m = \begin{cases} r_i(\varepsilon_1 - \varepsilon_2), & \text{if } r_i \le \delta, \\ \delta(\varepsilon_1 - \varepsilon_2) + (r_i - \delta)\varepsilon_2, & \text{if } 0 < \delta < r_i, \\ r_i\varepsilon_2, & \text{if } \delta \le 0. \end{cases}$$

B. Modified Dijkstra's Algorithm

When admitting a new flow f_i, minimizing the total energy cost is equivalent to finding a path which adds minimum cost to the network. When treating energy cost at nodes as the weights of edges, the problem is transformed into finding a path from s_i to d_i whose total weights along its edges is minimized. To solve this problem, we modify the well-known Dijkstra's routing algorithm, whose details can be found, for example, in [8, Chapter 24] and [9, Chapter 5].

The main idea of Dijkstra's algorithm can be outlined as follows. In a weighted, directed graph $G(\mathcal{V}, \mathcal{E})$, the total weights from source n_0 to node n_k is denoted as D_k. The total weights from n_k to n_j is denoted as d_{kj}. It maintains a set $\mathcal{S}$ of nodes whose minimum total weights from the source s have already been determined. It repeatedly selects nodes n_k, where $n_k \in \mathcal{V} \setminus \mathcal{S}$, which is closest to the source, n_0. Then add n_k into $\mathcal{S}$ and update the total weights of those nodes leaving n_k. By using such a greedy strategy, it can finally find the shortest path until all the nodes are in $\mathcal{S}$.

Note that Dijkstra's algorithm cannot be directly applied to our problem, since in our graph, the weight of an edge is not a constant. Use ϕ_j to denote the current best path from the source node, n_0, to n_j. Our modified Dijkstra's algorithm can be summarized as follows:

Initialization: Set $\mathcal{S} := \{n_0\}$, $D_0 := 0$, and $\phi_0 := \{n_0\}$. For $n_k \in \mathcal{N}_0$, set $D_k := w_{0k}^0$ and $\phi_k = \{n_0, n_k\}$. For all other nodes, set $D_k := \infty$ and $\phi_k := \emptyset$.

Step 1: Find $n_p \in \mathcal{V} \setminus \mathcal{S}$ such that

$$D_p := \min_{n_j \notin \mathcal{S}} D_j.$$

Set $\mathcal{S} := \mathcal{S} \cup \{n_p\}$. If $\mathcal{S} = \mathcal{V}$, the algorithm halts and outputs its solution. Otherwise, proceed to step 2.

Step 2: For $n_j \in \mathcal{N}_p \setminus \mathcal{S}$, let

$$n_k := \arg \min_{n_{k'} \in \mathcal{N}_p \cap \mathcal{S}} [D_{k'} + w_{k'p}^{\alpha(k')} + w_{pj}^{k'}],$$

$$\Delta_j := \min\{D_k + w_{kp}^{\alpha(k)} + w_{pj}^k, D_p + w_{pj}^{\alpha(p)}\},$$

where $\alpha(k)$ is the index of the second last node in ϕ_k, i.e., $\phi_k = \{n_0, \ldots, n_{\alpha(k)}, n_k\}$, for $k \neq 0$, and $\alpha(0)$ is defined as 0.

If $\Delta_j < D_j$, set

$$D_j := \Delta_j,$$

and

$$\phi_j := \begin{cases} A(\phi_p, n_j), & \text{if } \Delta_j = D_p + w_{pj}^{\alpha(p)}, \\ A(\phi_k, n_p, n_j), & \text{otherwise} \end{cases}$$

where $A(\phi, n_1, n_2, \ldots, n_l)$ is the operation that appends $n_1, n_2, \ldots, n_l$ to ϕ. Then go to step 1.

This algorithm is able to find an optimal solution for the single-flow admission problem, following the same argument for the optimality of the original Dijkstra's algorithm.

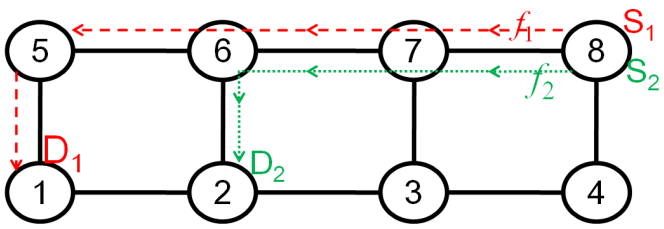

Fig. 1. An example.

C. An Example

We use an example to illustrate how the algorithm works. Consider Figure 1. Assume $\varepsilon_1 = 1.1\varepsilon_2$. The rates of existing flows f_1 and f_2 are R and $3R$, respectively. The goal is to select a minimum-cost path for a new flow f_3, which is from n_1 to n_8, with rate $4R$. The algorithm works as follows:

1) (Initialization) Set $\mathcal{S} = \{n_1\}$,$D_1 = 0$ and $\phi_1 = \{n_1\}$. For $k = 2, 5$, set $D_k = w_{1k}^1 = 4R\varepsilon_2$, $\phi_k = \{n_1, n_k\}$. For other nodes n_k, set $\phi_k = \emptyset$ and $D_k = \infty$.
2) Since $D_2 = D_5$, we break the tie arbitrarily and choose n_2. Set $\mathcal{S} = \{n_1, n_2\}$. $\mathcal{N}_2 \setminus \mathcal{S} = \{n_3, n_6\}$ and $\mathcal{N}_2 \cap \mathcal{S} = \{n_1\}$. $\Delta_3 = D_1 + w_{12}^1 + w_{23}^1 = 8R\varepsilon_2$. Since $\Delta_3 < D_3$, set $D_3 = \Delta_3 = 8R\varepsilon_2$ and set $\phi_3 = A(\phi_1, n_2, n_3) = \{n_1, n_2, n_3\}$. Similarly, $D_6 = 8R\varepsilon_2$ and $\phi_6 = \{n_1, n_2, n_6\}$.
3) Since $D_5 = \min\{D_3, D_5, D_6\}$, add n_5 into $\mathcal{S}$ and $\mathcal{S} = \{n_1, n_2, n_5\}$. Then $\mathcal{N}_5 \setminus \mathcal{S} = \{n_6\}$ and $\mathcal{N}_5 \cap \mathcal{S} = \{n_1\}$. Note that some packets of f_3 can be coded with f_1 at n_5 when calculating w_{56}^1. $\Delta_6 = D_1 + w_{15}^1 + w_{56}^1 =$

$0+4R\varepsilon_2+3.1R\varepsilon_2 = 7.1R\varepsilon_2$. Since $\Delta_6 < D_6$, set $D_6 = 7.1R\varepsilon_2$ and set $\phi_6 = A(\phi_1, n_5, n_6) = \{n_1, n_5, n_6\}$.

4) Since $D_6 = \min\{D_3, D_6\}$, add n_6 into $\mathcal{S}$, and $\mathcal{S} = \{n_1, n_2, n_5, n_6\}$. Then $\mathcal{N}_6 \setminus \mathcal{S} = \{n_7\}$ and $\mathcal{N}_6 \cap \mathcal{S} = \{n_2, n_5\}$. In w_{67}^2, f_1 can be encoded with f_2 at n_6. $\Delta_7 = \min\{(D_5 + w_{56}^1 + w_{67}^5), (D_2 + w_{26}^1 + w_{67}^2), (D_6 + w_{67}^2)\} = D_2 + w_{26}^1 + w_{67}^2 = 4R\varepsilon_2 + 4R\varepsilon_2 + 1.3R\varepsilon_2 = 9.3R\varepsilon_2$. Since $\Delta_7 < D_7$, set $D_7 = 9.3R\varepsilon_2$ and set $\phi_7 = A(\phi_2, n_6, n_7) = \{n_1, n_2, n_6, n_7\}$.
5) Since $D_3 = \min\{D_3, D_7\}$, add n_3 into $\mathcal{S}$ and get $\mathcal{S} = \{n_1, n_2, n_3, n_5, n_6\}$. Then $\mathcal{N}_3 \setminus \mathcal{S} = \{n_4, n_7\}$ and $\mathcal{N}_3 \cap \mathcal{S} = \{n_2\}$. $\Delta_4 = D_2 + w_{23}^1 + w_{34}^1 = 12R\varepsilon_2$. Since $\Delta_4 < D_4$, set $D_4 = 12R\varepsilon_2$ and $\phi_4 = A(\phi_2, n_3, n_4) = \{n_1, n_2, n_3, n_4\}$. $\Delta_7 = D_2 + w_{23}^1 + w_{37}^1 = 12R\varepsilon_2$. Since $\Delta_7 > D_7$, D_7 will not be updated.
6) Since $D_7 = \min\{D_4, D_7\}$, add n_7 into $\mathcal{S}$ and $\mathcal{S} = \{n_1, n_2, n_3, n_5, n_6, n_7\}$. Then $\mathcal{N}_7 \setminus \mathcal{S}$ and $\mathcal{N}_7 \cap \mathcal{S} = \{n_3, n_6\}$. In w_{78}^6, all packets of f_1 can be coded with f_1 and f_2. $\Delta_8 = \min\{(D_6 + w_{67}^5 + w_{78}^6), (D_3 + w_{37}^2 + w_{78}^3), (D_7 + w_{78}^6)\} = D_7 + w_{78}^6 = 9.3R\varepsilon_2 + 0.4R\varepsilon_2 = 9.7R\varepsilon_2$. Since $\Delta_8 < D_8$, set $D_8 = 9.7R\varepsilon_2$ and set $\phi_8 = A(\phi_7, n_8) = \{n_1, n_2, n_6, n_7, n_8\}$.
7) Since $D_8 = \min\{D_4, D_8\}$, add n_8 into $\mathcal{S}$ and $\mathcal{S} = \{n_1, n_2, n_3, n_5, n_6, n_7, n_8\}$. Then $\mathcal{N}_8 \setminus \mathcal{S} = \{n_4\}$ and $\mathcal{N}_8 \cap \mathcal{S} = \{n_7\}$. $\Delta_4 = \min\{(D_7 + w_{78}^6 + w_{84}^7), (D_8 + w_{84}^7)\} = D_8 + w_{84}^7 = 9.7R\varepsilon_2 + 4R\varepsilon_2 = 13.7R\varepsilon_2$. Since $\Delta_4 > D_4$, D_4 will not be updated.
8) Finally add the last node n_4 into $\mathcal{S}$. The algorithm halts and outputs solution D_8 and ϕ_8.

When the algorithm stops, the shortest path and total energy cost of the path can be obtained. In this example, the selected path for f_3 is $\{n_1, n_2, n_6, n_7, n_8\}$ and energy cost is $9.7R\varepsilon_2$.

IV. Iterative Algorithm

To solve the optimization problem in Section II, we propose the aforementioned single flow admission procedure be run in an iterative manner. We call this algorithm Iterative Single-Flow Admission (ISFA). We call it a *round* when each flow $f_i \in \mathcal{F}$ is optimized once. At the beginning of the first round, no flow is established in the network. Each flow is then admitted into the network sequentially using the single-flow admission procedure. From the second round onwards, the paths for the flows are optimized one by one until no more improvement can be made. When optimizing a flow, it can be regarded as re-routing the flow by applying single-flow admission procedure. ISFA can be described as follows:

Step 1: (First round) For $f_i \in \mathcal{F}$, sequentially use modified Dijkstra's algorithm to find $\mathcal{P}_i$ and add f_i into the network. After adding all f_i's, obtain the corresponding total energy cost $\mathbb{E}_{total}$. Proceed to step 2.

Step 2: (Subsequent rounds) For $f_i \in \mathcal{F}$, delete f_i from the network. Find a new path $\mathcal{P}_i^{'}$ for f_i without changing the paths for other flows. Calculate the total energy cost $\mathbb{E}_{total}^{'}$ when using $\mathcal{P}_i^{'}$. If $\mathbb{E}_{total}^{'} < \mathbb{E}_{total}$, add f_i into the network using $\mathcal{P}_i^{'}$. Otherwise, add f_i back using its original path $\mathcal{P}_i$. Move on to the optimization of next flow.

Step 3: (Halting condition) If all $f_i \in \mathcal{F}$ did not change their paths in the last round, the algorithm halts and outputs its solution. Otherwise go to Step 2.

The proposed ISFA always stops after a finite number of iterations. To see this, note that a flow selects a new shortest path to replace the old one only if the total energy cost $\mathbb{E}_{total}$ can be strictly reduced. It means that, after each round, the total energy cost $\mathbb{E}_{total}$ is strictly decreasing. Therefore, oscillation of the paths is impossible. As the network size is finite, there is only a finite number of path choices, implying that the algorithm must eventually stop. Indeed, our simulations show that the number of iterations required is quite small.

V. Performance Evaluation

To evaluate the performance of ISFA, we run simulations in Matlab and compare it with a traditional method, which assigns a shortest path (in terms of number of hops) to each flow and does not perform any network coding. We call it shortest-path routing (SPR), which will be used as our benchmark. We also consider applying network coding on top of SPR. In other words, the paths are chosen according to SPR, and when coding opportunities arise, network coding will be performed. We call this method shortest-path routing with network coding (SPR-NC).

We run simulations on both two-dimensional square lattice and random networks. A square lattice is a lattice which has the same number of nodes on the two dimensions, and each node, unless on the boundary, is directly connected to its four closest neighbors. Random networks are generated by placing nodes on a square uniformly at random. The area of the square is equal to the number of nodes times 1000 square units. Two nodes are considered in each other's transmission range if their distance is less than 25 units. When a network is connected, it can satisfy the requirement that each source-destination pair has at least one path. A generated network will be discarded if it is not connected. A new one will be re-generated until the condition is met. For both types of networks, flows are generated by randomly selecting a source and a destination, with a constant traffic rate of 100 units. In this paper, we do not study relationship between network coding and capacity, thus, we assume the capacity of each edge is infinite. We assume the energy costs of a coded packet and a non-coded packet are $\varepsilon_1 = 1.1$ and $\varepsilon_2 = 1$, respectively. For each simulation setting, we run 200 times and take the average as the final result.

Simulation results show that ISFA converges fast. We run simulations on different network scales with different traffic loads. In Table I, the first row reflects the network scale, where the numbers are both the number of nodes and the number of unicast flows, which are assumed equal in our simulation tests on the convergence speed of ISFA. The second and third rows show the average number of rounds that the algorithm takes before halts. We can see that ISFA converges fast.

To find the relationship between energy reduction and network scales, we run simulations on networks with fixed

TABLE I
CONVERGENCE ON DIFFERENT NETWORKS.

Network \ Scale	25	36	49	64	100	225	324
Lattice Network	1.91	2.07	2.45	2.53	3.11	4.15	4.37
Random Network	1.21	1.54	1.83	2.05	2.57	3.92	4.31

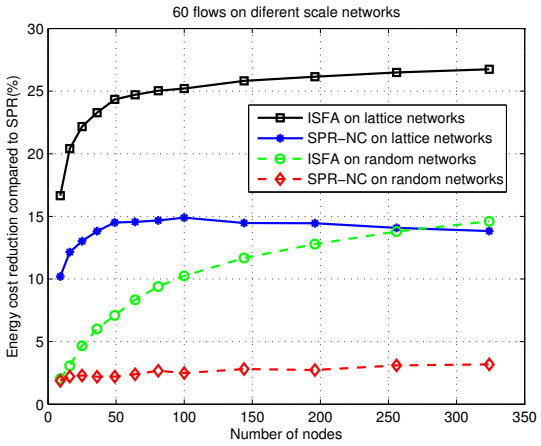

Fig. 2. Energy cost reduction on various network scales with 60 flows.

number of flows but various number of nodes. The number of nodes changes from 9 to 324 with 60 unicast flows. Results are ploted in Figure 2. We can see that when network scale is very small, the energy cost reduction is not large. The reason is that XOR can only be applied to paths whose length is longer than three. When the network is very small, most paths are very short and coding opportunities are rare. Thus, energy cost cannot be reduced greatly. When the network becomes larger, more and more long paths appear and coding opportunity increases. Energy cost reduction becomes more significant, and ISFA always yields a much larger gain than SPR-NC.

We are also interested in studying the relationship between energy cost reduction and traffic load. We run simulations on both lattice networks and random networks with 64 nodes and number of flows ranging from 10 to 400. The simulation results are shown in Figure 3. We can see that when the total number of flows is small, the total energy reduction is also very small, since there is not much chance to perform XOR coding between opposite flows. As there are more flows in the network, the chance of performing network coding increases. Therefore, the reduction of energy cost becomes much larger. Again ISFA significantly outperforms SPR-NC.

In our simulation, infinite buffers are assumed. However, the buffer capacity is limited in reality, which will reduce coding opportunities, so as to reduce the the gain of ISFA.

ISFA performs better on energy cost reduction, but leads to longer delays. In SPR and SPR-NC, all flows get shortest delays because of shortest paths. However, not all flows can get shortest paths in ISFA and leads to longer delays. Hence, SPR and SPR-NC outperform ISFA on delay.

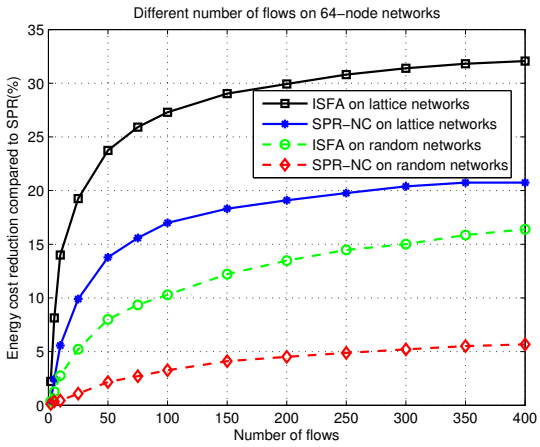

Fig. 3. Energy cost reduction on a 64-node network with different number of flows.

VI. CONCLUSION

A routing algorithm for XOR-coded wireless networks is proposed. The objective is to assign paths for traffic flows so as to minimize the total energy cost of the network. While globally optimal solution is difficult to find, we consider the subproblem of single-flow admission, whose optimal solution can be efficiently found. Based on that, an iterative algorithm is developed, which outputs sub-optimal solution to the original energy-cost minimization problem. To test its performance, we ran simulations for our proposed algorithm on both lattice and random networks with various scales and various traffic loads. Simulation results show that the proposed algorithm converges fast. When compared with shortest path routing, with and without network coding, the proposed algorithms always performs better in reducing total energy cost, and can achieve a reduction up to 32% for some scenarios.

REFERENCES

[1] R. Ahlswede, N. Cai, S. Y. R. Li and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, pp. 1204-1216, Aug. 2002.

[2] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard and J. Crowcroft, "XORs in the air: practical wireless network coding," *SIGCOMM Comput. Commun. Rev.*, vol. 36, pp. 243-254, Aug. 2006.

[3] S. Sengupta, S. Rayanchu and S. Banerjee, "An analysis of wireless network coding for unicast sessions: The case for coding-aware routing," in *IEEE Int. Conf. on Comput. Commun.*, 2007, pp. 1028-1036.

[4] 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/ACM Trans. Netw.*, vol. 14, no. SI. pp. 2608-2623, 2006.

[5] V. Reddy, S. Shakkottai, A. Sprintson and N. Gautam, "Multipath wireless network coding: a population game perspective," *Proc. IEEE Int. Conf. on Comput. Commun.*, pp. 1-9, Mar. 2010.

[6] A. ParandehGheibi, A. Ozdaglar, M. Effros and M. Médard, "Optimal reverse carpooling over wireless networks - A distributed optimization approach," in *44th Annu. Conf. on Inform. Sci. and Syst.*, pp. 1-6, 2010.

[7] Y. Bejerano, S. J. Han, K. T. Lee and A. Kumar, "Single-path routing for life time maximization in multi-hop wireless networks," *Wireless Networks*, vol. 17, no. 1, pp. 263-275, Jan. 2011.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, "Single-source shortest paths," in *Introduction to Algorithms*, 3rd ed. MA: The MIT press, 2009.

[9] D. Bertsekas and R. Gallager, "Routing in data networks," in *Data Networks*, 2nd ed. New Jersey: Prentice hall, 1992.