Energy Efficient Opportunistic Routing in Wireless Networks

Xufei Mao [‡] Xiang-Yang Li[‡], Wen-Zhan Song[†], Ping Xu[‡] Kousha Moaveni-Nejad[‡],

***Abstract*—Opportunistic routing [1], [2] was shown to improve the network throughput greatly. The core idea is to allow any node in the *forwarder list*, which overhears the transmission and is closer to the destination to participate in forwarding the packet. The nodes in *forwarder list* is prioritized and the lower priority forwarder will discard the packet if the packet has been forwarded by a higher priority forwarder. One open problem is how to select and prioritize forwarder list efficiently. In the paper, we investigate how to select and prioritize forwarder list to minimize energy consumptions. We study the case where the transmission power of each node is fixed (known as *non-adjustable transmission model*) as well as the case where each node is able to adjust its transmission power for each transmission (known as *adjustable transmission model*). Energy optimum algorithms to select and prioritize forwarder list in both cases are presented and analyzed. Worth to mention that, our methods do not assume any geometrical properties or energy models, they apply to practical and dynamic wireless networks. In addition, we conducted extensive simulations in TOSSIM to study the performance of the proposed routing protocol by comparing it with ExOR [1].**

I. Introduction

Routing protocols have been studied extensively in wireless networks. In general, design of those protocols are guided by two essential requirements: minimize energy cost and maximize network throughput. The traditional routing protocols in wired networks choose the best sequence of nodes between the source and destination, and forward each packet through that sequence until next routing update period. In the past, the majority routing protocols in multi-hop wireless networks have typically followed this convention, including those multi-path routing protocols. However, this convention did not take advantages of the broadcast nature of wireless communication: a node's transmission can be heard by any node within its transmission range. In addition, the lossy and dynamic wireless links make traditional routing protocol unsuitable as well, *i.e.*, in multihop wireless networks, various factors, like fading, interference, multi-path effects, and collisions, can temporarily lead to heavy packet losses [9] in the pre-selected *good* path, even not mentioning whether that link is actually good or not.

In contrast, opportunistic routing, like ExOR [1] and MORE [2], allows any node (in the forwarder list) that overhears the transmission and is closer to the destination to participate in forwarding the packet. The routing path is selected opportunistically based on the current link quality situations. However, opportunistic routing introduces a difficult challenge at the same time: multiple nodes may hear a broadcasted packet and unnecessarily forward the same packet. ExOR deals with this issue by tying the MAC to the routing, imposing a strict scheduler on routers' access to the medium. The scheduler goes in rounds. Forwarders transmit in order, and only one forwarder is allowed to transmit at any given time. The others listen the transmissions to learn which packets were overheard by each node. In contrast to ExOR's highly structured scheduler, MORE addresses the above challenge with randomness and is MAC-independent Opportunistic Routing and Encoding. MORE randomly mixes packets before forwarding them. This ensures that routers that hear the same transmission do not forward the same packet. Indeed, the probability that such randomly coded packets are the same is proven to be exponentially low. As a result, MORE does not need a special scheduler; it runs directly on top of 802.11.

There is an open problem in the opportunistic routing design: how to select the appropriate forwarder list such that the expected energy cost is minimized? In this paper, we investigate this problem and propose our *Energy Efficient Opportunistic Routing (EEOR)* protocol for wireless networks through rigorous theory analysis as well as extensive simulations. We study the case where the transmission power of each node is fixed (known as *non-adjustable transmission model*) as well as the case where each node can adjust its transmission power for each transmission (known as *adjustable transmission model*). Optimum algorithms to select and prioritize forwarder list in both cases are presented and analyzed. Worth to mention that, our analysis do not assume any geometrical properties or energy models, and apply to dynamic multi-hop wireless networks. We conducted extensive simulations in TOSSIM to study the performance of the proposed protocol by comparing it with ExOR [1]. It shows that both energy efficiency and network performance is higher than ExOR.

The rest of the paper is organized as follows. In Section II we explain how to calculate the expected energy cost using EEOR and present the algorithm to select energy cost optimum forwarder list. In Section III, we present our system design details of our implementation in TinyOS. We report our simulation results in Section IV. We review previous works on some related energy efficient and opportunistic routing protocols in Section V and conclude in Section VI.

II. Theory Analysis

A. Network Model

We consider a wireless ad hoc (or mesh) network and assume that all wireless nodes have distinctive identities (i.e., $i \in [1, n]$) and each wireless node u has a maximum transmission power W. In some cases, we assume that each node can adjust its transmission power to any value between 0 and W. Let w denote such adjusted transmission power. The multihop

wireless network is then modeled by a communication graph $G = (V, E)$, where V is a set of $n = |V|$ wireless nodes and E is a set of directed links. Each directed link (u, v) has a non-negative *weight*, denoted by $w(u, v)$, which is the minimum transmission power required by node u to send a packet to node v. Our methods work with any weight assignment, in other words, the weights assigned to the communication links could be derived from any power attenuation model.

The number of neighboring nodes of a node u changes with the power consumed at node u when sending a packet. Let $N_w(u)$ denote the neighboring nodes of a node u when node u transmits with the power w. For simplicity, when the subscript w is not mentioned, we mean that the node is using its maximum power *i.e.*, $N(u) = N_W(u)$. Also each link (u, v) has an *error probability*, denoted by $e(u, v)$, which is the probability that a transmission over link (u, v) is not successful. In other words, node u must consume at least $w(u, v)$ power to have a chance of $1 - e(u, v)$ to transmit a packet to node v. No transmission is possible otherwise.

To see how we take advantage of wireless multicast advantage (WMA), consider the contrived scenario in Figure 1. The error probability from the source to each node v_i is e and the error probability from each node v_i to the target node is 0. Traditional routing would route all the data through the same node during a routing-update period, so the packet has to be sent $\frac{1}{1-e}$ times before being received by the intended node v_i. However, by taking advantage of WMA property, the packet has to be sent only $\frac{1}{1-e^n}$ times. The difference is more noticeable when e is close to 1 and n is a big number. In other words, the advantage of using WMA property is more noticeable in dense graphs with high error probabilities.

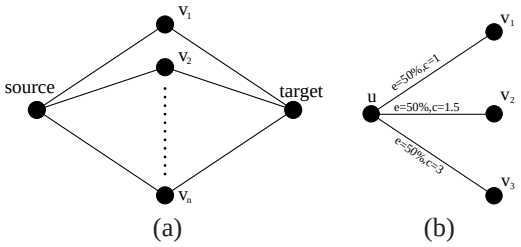

Fig. 1. (a) Wireless Multicast Advantage, (b) Calculating the expected cost.

The basic idea of opportunistic routing is to take advantage of WMA property during packet forwarding. Consider that a source node and a target node that are far apart, where the packet from the source node to a target node must be routed through a multi-hop path. Traditionally, the packet will be routed through a predefined multi-hop path during a routing-updated period, and the packet sent at each step by the source node is intended for only one node, in other words, unicast transmission for each step. It makes sense in wired networks as nodes are connected through dedicated wires, however, it did not take the wireless multicast advantage. As proposed in ExOR [1], the source node selects a subset of its neighboring nodes as a forwarder list. The forwarder list is prioritized by the source node to indicate which nodes have higher priority to forward the packet. Then one or more nodes in the forwarder list, which received the packet successfully, will opportunistically act as the new source node(s) and route the packet to the target node. This is implemented by sending ACK messages in different priorities among the nodes in the forwarder list which receive the packet correctly. Details has been described in section III. In this paper, we continue to use the concept and main idea of *Forwarder List* and will propose the different strategy to construct *Forwarder List* more efficiently and usefully in order to minimize total energy consumption of wireless networks.

Let $C_u(\mathrm{Fwd})$ denote the expected cost needed by an opportunistic routing to send a packet from node u to the target node when the forwarder list is Fwd. For simplicity, we use C_u to denote the expected cost of node u if there are no confusions. Initially the expected cost of target is 0 and the costs of other nodes are ∞. Using the similar mechanism of distance vector routing, the calculations will be carried out periodically and every node updates its expected cost and forwarder list periodically. Since nodes may or may not have the ability to adjust their transmission power, here we consider both assumptions and provide algorithms to select the forwarder list for both cases.

B. Non-Adjustable Transmission Power

In this section, we consider the non-adjustable transmission power model, where each node, say u, uses a fixed transmission power. Note that in this model the weight of the link connecting the node u to its neighbors does not matter. One may think that the best forwarder list in this case is $N(u)$, but, surprisingly, it is not always true. At the end of this section, we will show an example, based on the Figure 1, such that the best forwarder list only contains part of nodes in $N(u)$.

Now we study how to calculate the expected cost for each node and how to find the forwarder list. Consider a node u and its neighboring nodes. We find the expected cost of node u and the forwarder list of node u based on the expected cost of the neighbors which are already assigned an expected cost. In other words, here we want to choose a subset of neighboring nodes $N(u)$ as forwarder list of node u such that the expected cost of sending a packet for node u to the target is minimized. Given a set of nodes S, let S^* denote the sorted list of S based on their expected cost in increasing order. Let $\mathrm{Fwd}(u)$ denote the forwarder list of node u. To find the expected cost at node u, we first sort the forwarder list $\mathrm{Fwd}^*(u)$ in increasing order of the expected cost, *i.e.*, $\mathrm{Fwd}^*(u) = \{v_1, v_2, ..., v_{|\,\mathrm{Fwd}(u)|}\}$, where $i < j \Rightarrow C_{v_i} \leq C_{v_j}$. Let α denote the probability that a packet sent by node u is not received by any node in $\mathrm{Fwd}^*(u)$. Clearly,

$$\alpha = \prod_{i=1}^{|\,\mathrm{Fwd}^*(u)|} e_{uv_i} \tag{1}$$

Let ρ denote the probability that a packet sent by node u is received by at least one node in $\mathrm{Fwd}^*(u)$. Then $\rho = 1 - \alpha$. Let $C_u^h(\mathrm{Fwd}^*)$ denote the expected energy that node u must

consume to send a packet to at least one node in the forwarder list Fwd^*. $C_u^h(\mathrm{Fwd}^*)$ can be calculated as follows:

$$C_u^h(\mathrm{Fwd}^*) = \frac{w}{\rho} \tag{2}$$

When at least one of the nodes in the forwarder list has received the packet, we need to calculate the expected cost for one of nodes in the forwarder list to forward the packet. Here we assume that only one node from the forwarder list will forward the packet. Although this assumption is very optimistic but, as we will explain later, in most cases it is true. The expected cost that we calculate here could be slightly lower than the cost when multiple nodes from forwarder list could forward the data packet.

Let $C_u^f(\mathrm{Fwd}^*)$ denote the expected total cost for node(s) in Fwd to relay the packet to the target when u's current forwarder list is Fwd. Assume that the prioritized forwarder list is $\mathrm{Fwd}^* = \{v_1, v_2, \cdots, v_{|\mathrm{Fwd}^*|}\}$. The probability that node v_1 forwards the packet is $1 - e(u, v_1)$ in which case the cost will be C_{v_1}, then node v_2 will forward the packet with probability $e(u, v_1) \cdot (1 - e(u, v_2))$ in which case the cost will be C_{v_2}. Basically node v_i forwards the packet if it receives the packet and nodes $v_j, 0 < j < i$ did not receive the packet correctly and in this case it will cost C_{v_j}. So we can sum these costs as follows:

$$\beta = \sum_{i=1}^{|\mathrm{Fwd}^*|} \prod_{j=1}^{i-1} e_{uv_j} \cdot (1 - e_{uv_i}) \cdot C_{v_i} \tag{3}$$

Since β is computed under condition that a forwarder node got the packet, then we have

$$C_u^f(\mathrm{Fwd}^*) = \frac{\beta}{\rho} \tag{4}$$

After $C_u^h(\mathrm{Fwd}^*)$ and $C_u^f(\mathrm{Fwd}^*)$ are calculated for a given forwarder list, $C_u(\mathrm{Fwd}^*)$ is computed as follows:

$$C_u(\mathrm{Fwd}^*) = C_u^h(\mathrm{Fwd}^*) + C_u^f(\mathrm{Fwd}^*) \tag{5}$$

So far we introduced a method to calculate the expected cost for a given node and a given forwarder list. But we did not discuss how to choose the forwarder list. Consider there are k nodes in $N(u)$ for which an expected cost is already assigned, then there are $(2^k - 1)$ choices to select the forwarder list. Finding the expected cost pertaining to each forwarder list is not practical. Here we study the properties of the forwarder list and the expected cost first and then we explain how to efficiently choose the optimal forwarder list.

To simplify our arguments, let us introduce a property known as *prefix*. A set X is called a prefix of an ordered set Y if X are the set of first k elements of Y. So each set Y has $(|Y| + 1)$ prefixes. Now consider node u and its neighboring nodes $N(u)$. Sort the nodes in $N(u)$ based on their expected cost, and get $N^*(u) = \{v_1, v_2, ..., v_{|N(u)|}\}$ such that $|N(u)| \geq i > j > 0 \Rightarrow C_{v_i} > C_{v_j}$. First we show that the optimum forwarder list of node u is a prefix of $N^*(u)$.

Theorem 1: The optimum forwarder list of node u must be a prefix of $N^*(u)$.

Proof: We prove this theorem by contradictions. Assume that the optimum forwarder list, say Opt, has the lowest expected cost and is not a prefix of $N^*(u) = \{v_1, v_2, \cdots, v_{|N^*(u)|}\}$. Then there must be two nodes $v_k, v_{k+1} \in N^*(u)$ such that $v_k \notin Opt$ and $v_{k+1} \in Opt$. Assume v_k is the first node satisfies conditions above when we check nodes in Opt^* (sorted list of Opt by increasing order) one by one. We will show that, if v_k is added to Opt, the expected cost decreases, i.e., $C_u(Opt^+) \leq C_u(Opt)$, and hence Opt cannot be the optimum forwarder list. Here $Opt^+ = Opt \bigcup \{v_k\}$.

First, let's examine a way to compute an upper bound, say Δ, of the expected cost for the node set Opt^+ as follows. If node v_{k+1} receives the packet and node v_k doesn't, then node v_{k+1} forwards the packet with cost $C_{v_{k+1}}$. If both v_{k+1} and v_k receive the packet, then node v_k forwards the packet with cost C_{v_k}. In other words, node v_k forwards data only if both node v_k and v_{k+1} receive the packet. It is obvious that this change increases the expected cost of node u, assuming same forwarder list Opt^+. In other words, $\Delta \geq C_u(Opt^+)$.

Then, we compare Δ and $C_u(Opt)$. Let us consider the expected cost to forward the packet by one of the nodes in the forwarder list. The only time that the cost is different is when node v_k and node v_{k+1} both receive the packet and no node $v_i, i < k$ receives the packet. In the calculation of Δ, the cost is C_{v_k}; while in the calculation of $C_u(Opt)$, the cost is $C_{v_{k+1}}$ since $v_k \notin Opt$. According to the assumption, $C_{v_k} \leq C_{v_{k+1}}$, hence $\Delta \leq C_u(Opt)$.

Consequently, we have $C_u(Opt^+) \leq \Delta \leq C_u(Opt)$. That is to say, Opt can not be the optimum forwarder list, if it is not a prefix of $N^*(u)$. This completes the proof. ■

We further study the properties of forwarder list, by introducing two more theorems. The first theorem, Theorem 2, shows that if a node, whose expected cost is less than the expected cost of a prefix forwarder list, is added to the forwarder list, then the expected cost of the newly created forwarder list will decrease but it will still be greater than the expected cost of the newly added node. The second theorem, Theorem 3, shows the opposite: if a node, whose expected cost is greater than the expected cost of a prefix forwarder list, is added to the forwarder list, then the expected cost of the newly created forwarder list will increase.

Theorem 2: Consider a node u, a prefix forwarder list Fwd^*, and a node $v_k \in N(u) \setminus \mathrm{Fwd}^*$. If $C_{v_k} < C_u(\mathrm{Fwd}^*)$, then $C_{v_k} < C_u(\mathrm{Fwd}^* \bigcup \{v_k\}) < C_u(\mathrm{Fwd}^*)$

Proof: Assume node u uses energy w to send the packet. For simplicity let $e = e_{uv_k}$, $C_1 = C_u(\mathrm{Fwd}^*)$, and $C_2 = C_u(\mathrm{Fwd}^* \bigcup \{v_k\})$. Using Equations (1), (3), and (5) we have

$$\begin{cases} (1 - \alpha) \cdot C_1 = w + \beta \\ (1 - \alpha \cdot e) \cdot C_2 = w + \beta + \alpha(1 - e)C_{v_k} \end{cases} \tag{6}$$

Subtract and reorder, we have $(1 - \alpha \cdot e) \cdot C_2 = (1 - \alpha) \cdot C_1 + \alpha(1 - e)C_{v_k}$. We know that $C_{v_k} < C_1$, so $(1 - \alpha \cdot e) \cdot C_2 < (1 - \alpha) \cdot C_1 + \alpha(1 - e)C_1$, thus, $C_2 < C_1$.

Now we show that $C_{v_k} < C_2$. In Equation (6), we replace $(w + \beta)$ in the second equation with $((1 - \alpha) \cdot C_1)$ from the

first equation. Then we have $(1-\alpha\cdot e)\cdot C_2 = (1-\alpha)\cdot C_1 + \alpha(1-e)C_{v_k}$. We know that $C_{v_k} < C_1$, thus, $(1-\alpha\cdot e)\cdot C_2 > (1-\alpha)\cdot C_{v_k} + \alpha(1-e)C_{v_k}$. This implies that $C_2 > C_{v_k}$. ■

Theorem 2 proves that the expected cost of each node is higher than the expected cost of every node in its forwarder list. This property enables us to take a greedy approach in routing, which will be discussed later.

Theorem 3: Consider a node u, a prefix forwarder list Fwd^*, and a node $v_k \in N(u) \setminus \mathrm{Fwd}^*$. If $C_{v_k} > C_u(\mathrm{Fwd}^*)$, then $C_u(\mathrm{Fwd}^* \bigcup \{v_k\}) > C_u(\mathrm{Fwd}^*)$.

Proof: The proof is similar to that of Theorem 2. In Equation (6), we subtract and reorder to get $(1-\alpha\cdot e)\cdot C_2 = (1-\alpha)\cdot C_1 + \alpha(1-e)C_{v_k}$. We know that $C_{v_k} > C_1$, thus, $(1-\alpha\cdot e)\cdot C_2 > (1-\alpha)\cdot C_1 + \alpha(1-e)C_1$. This implies that $C_2 > C_1$. ■

Having these three properties, the forwarder list for a wireless node can be selected easily. The following Algorithm 1 finds the optimum forwarder list and calculates the expected cost for a wireless node u. First it calculates $N^*(u)$ and then adds nodes in $N(u)$ to the forwarder list as long as the expected cost of u is decreasing. Just before the cost starts to increase, it terminates. Clearly, u knows whether it will increase or decrease its expected cost before it adds a neighbor node to its forwarder list due to Theorem 2. Note that based on the theorems aforementioned, it is clear that Algorithm 1 finds the optimum forwarder list for each wireless node u. Basically Algorithm 1 can be used to find the forwarder list and compute the expected cost for a wireless node under the non-adjustable transmission power model.

Algorithm 1 ExpectedCostFixedPower$(u, N(u), C_u, \mathrm{Fwd})$

Input: the expected cost of all its neighboring nodes
Output: the cost C_u and forwarder list Fwd.

1: Set $C_u = \infty$, $\mathrm{Fwd} = \emptyset$.
2: Sort the neighboring nodes $N^*(u) = \{v_1, v_2, ..., v_{|N(u)|}\}$ based on their expected cost.
3: **for** $(i = 1; i \le |N(u)|; i = i + 1)$ **do**
4: **if** $(C_u > C_{v_i})$ **then**
5: Set $\mathrm{Fwd} = \mathrm{Fwd} \bigcup v_i$ and $C_u = C_u(\mathrm{Fwd})$

Consider a network example illustrated by Figure 1(b). Assume node u consumes one unit of energy (i.e. $w = 1$) to send a packet and $N_1(u) = \{v_1, v_2, v_3\}$. For simplicity let e_i denote $e(u, v_i)$ and c_i denote the expected cost at node v_i. The forwarder list and the expected cost of node u will be calculated as follows. First we add node v_1 to the forwarder list. The expected cost ($\mathrm{Fwd}(u) = \{v_1\}$) will be $\frac{w+(1-e_1)\cdot c_1}{1-e_1} = 3$. Next, the expected cost of node v_2 is 1.5 and based on Theorem 2 adding node v_2 to the forwarder list will decrease the expected cost of node u. Then add v_2 and the current expected cost ($\mathrm{Fwd}(u) = \{v_1, v_2\}$) becomes $\frac{w+(1-e_1)\cdot c_1+e_1(1-e_2)\cdot c_2}{1-e_1e_2} = 2.5$. In turn, we check node v_3 with expected cost 3, based on Theorem 3 adding node v_3 will increase the expected cost at node u. For example, the expected cost ($\mathrm{Fwd}(u) = \{v_1, v_2, v_3\}$) will be $\frac{w+(1-e_1)\cdot c_1+e_1(1-e_2)\cdot c_2+e_1e_2(1-e_3)\cdot c_3}{1-e_1e_2e_3}$, which is equal to $\frac{18}{7} > 2.5$. So the optimum forwarder list is $\{v_1, v_2\}$ and the expected cost at node u is 2.5. This would serve as a good example that an optimum forwarder list is not necessarily $N(u)$, as mentioned in the beginning of this section.

Figure 2 illustrates another example of the optimum forwarder list calculations. The graph has 6 nodes and the link weights and error probabilities are shown. The forwarder list of each node and also expected cost of each node is shown in the figure (the expected cost is rounded by two decimal places). Note that not all the neighbor nodes are used in forwarder lists; the vertical link shown by a dashed line is an example.

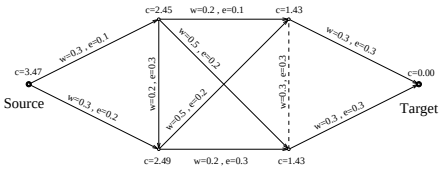

Fig. 2. An example for non-adjustable transmission power case. The solid direct link means the destination node of this link is selected into the forwarder list of the other.

C. Adjustable Transmission Power

In the power non-adjustable case (Section II-B), the power that a node u consumes is fixed and therefore $N(u)$ is fixed too. In this section we consider the case where a node can adjust its power to any value $w \in [0, W]$. Note that for a given forwarder list, if we decrease w to the weight of the farthest link in $\mathrm{Fwd}(u)$ then C_u^h (see Equation 2) may decrease while C_u^f (see Equation 4) will remain the same, so using adjustable transmission ranges will give us some marginal improvement. As another example consider Figure 3. Assume node u consumes power w, has the expected cost C_u, and $W > w(u, v) > w$. As can be seen in Figure 3, if node u consumes power w, node v will not receive packets sent by node u. Should we increase the transmission power of node u to include node v in its transmission power? If $C_v > C_u$, based on Theorem 3, adding node v will increase the expected cost at node u even if no more additional power is needed. But if $C_v < C_u$, it is a trade-off. Adding node v in one hand increases the power C_u^h that node u must consume and in the other hand decreases C_u^f, this may or may not decrease the expected cost at node u.

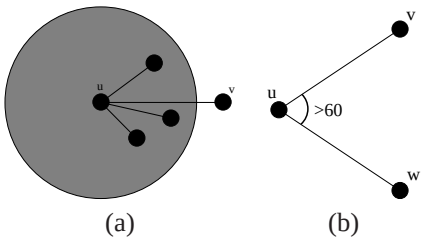

Fig. 3. (a) Calculating the expected cost in adjustable transmission power model; (b) The case where more than one node forwards the packet.

To find the expected cost in adjustable transmission power model, we sort the nodes in $N(u)$ based on the weight of the link that connects that node to u. Then we keep increasing the power at node u such that the number of nodes in $N_w(u)$ increases by one at each step. Then for each w and each $N_w(u)$, using the Algorithm 1, we calculate the expected cost and pick the one that induces the minimum cost.

Algorithm 2 ExpectedCostAdjustPower(u, C_u, Fwd)

Set $C_u = \infty$, $\mathrm{Fwd} = \emptyset$
Sort nodes in $N(u)$ based on $weight$
Let $N(u) = \{v_1, v_2, ..., v_{|N(u)|}\}$
for $(i = 1; i \leq |N(u)|; i = i + 1)$ **do**
  Set $w = w(u, v_i)$
  Run ExpectedCostFixedPower$(u, N_w(u), Cr, \mathrm{CFwd})$
  if $C_u > Cr$ **then**
    Set $C_u = Cr$ and $\mathrm{Fwd} = \mathrm{CFwd}$.

D. The Main Algorithm

So far we have learned how to build the forwarder list for an individual node based on the expected costs already assigned to neighboring nodes. Now we provide an algorithm that builds the forwarder list for each node in the graph.

Basically we calculate an *expected cost* for each node u to send a packet to the target node t. Let $C_{u,t}$ denote this expected cost. Assume that the cost for a node to send a packet to itself to be zero (i.e., $C_{t,t} = 0$). Given a set V of nodes, a source node s, and a target node t, the following algorithm computes the expected energy cost needed to relay a packet from any node to the target node t using opportunistic routing.

Algorithm 3 Expected Cost by Opportunistic Routing

Input: target node t, source node s, power $w(u, v)$ and link reliability for each link uv.

Set $S_1 = V - \{t\}$, $S_2 = \{t\}$ and $\forall u \in V$, set $C_u = \infty$, let $C_t = 0$
$\forall u \in N(t)$ run Algorithm 1 or 2 to compute C_u.
repeat
  Let v be the node in S_1 that has the minimum cost.
  Let $S_1 = S_1 - \{v\}$ and $S_2 = S_2 \cup \{v\}$.
  For each $u \in N(v) \cap S_2$, run Algorithm 1 or 2 to compute C_u.
until $s \in S_2$

Algorithm 3 works as follows. First the set of nodes V is divides into two sets S_1 and S_2. Initially set $S_1 = V - \{t\}$ and $S_2 = \{t\}$. Then we find the node in S_1 that has the least expected cost to send a packet to the target node t. We remove that node from $V \setminus S$ and add it to Set S. The algorithm continues till the source node s is in the set S_2. Let *Expected Cost Graph* denote the directed subgraph that includes a directed edge uv from the original communication graph if v is in the forwarder list of u.

Theorem 4: Expected Cost Graph is loop-free.

Proof: We prove this by induction. At the beginning there is only one node in S_2, so S_2 is loop-free. Now consider that after k steps, S_2 is still loop-free and at $(k+1)^{th}$ step node u is added to S_2. Since at every step each node chooses its forwarder list from the nodes in S_2, there is no link in Expected Cost Graph to node u, so adding node u will not generate any loop. This completes the proof. ■

Theorem 5: Algorithm 3 assigns the optimum expected cost to each node.

Proof: We prove this theorem by induction. Algorithm 3 adds a node to S_2 at each step. We show that the expected cost assigned to the newly added node is optimum. We start with the target node that has expected cost 0 which is obviously optimum. Now consider that after k steps, the expected cost for every node in S_2 has been calculated and at $(k+1)^{th}$ step node u is added to S_2. Based on the algorithm, node u selects its forwarder list from the nodes in S_2 only. It suffices to show that node u will never choose a node in S_1 in its forwarder list. Based on Theorem 2, the expected cost of the node that is added to set S_2 is greater than the expected cost of any node that is in S_2. Since C_u has the smallest expected cost among the nodes in S_1, the expected cost of the nodes in S_1 will be always greater than C_u and hence those nodes will never be in the forwarder list of node u. This finishes the proof. ■

III. System Design

In this section, we present our design details of our EEOR protocol in TinyOS-based wireless sensor network system. The design faces several key challenges. Firstly, all nodes selected into the same forwarder list must agree on next operation, *i.e.*, based on the priorities coming with the packet, which one(s) will finally act as the relay node(s) in order to save energy and increase the throughput. Since agreement involves communication and thus increases the overhead of the wireless network, we must guarantee the increased overhead will not overwhelm the performance gain. Secondly, the EEOR protocol should be able to handle the network traffic efficiently, *i.e.*, be able to handle with congestion, to avoid bottleneck in order to decrease packet loss ratio and save the energy cost at the same time. To solve this issue, we need to consider many aspects. For example, the ongoing traffic flows from all source nodes should not exceed the capacity bound of the wireless networks. In other words, all source nodes need to have the ability to adjust their network flows according to the current network situation such that the ongoing flows in the wireless network are stable. For example, a source node could push more flow to the network if the latter does not reach its capacity; otherwise, a node should decrease its flow. Thirdly, a single packet could arrive at the destination through multiple paths, thus involves more wireless nodes, consumes more energy and increases the traffic burden of wireless networks. Thus, it is necessary to introduce penalty to EEOR if a node is too selfish such that it chooses too many nodes as potential forwarders. For example, when a wireless node finds the packets from its neighbor contains too much nodes in the forwarder list, it could increase its expected cost to quit the forwarder list

next time or drop this packet. Fourthly, the broadcast nature of wireless communication not just add the complexity of interferences, but also add benefits by enabling snooping or overhearing. A node can utilize overheard messages to reduce control overhead like ACK messages. Actually, to utilize these snooped information to avoid duplication is one important strategy in our design and simulation results indicate that this strategy can improve the system performance well. Algorithm 4 presents the strategy of packet handling of our EEOR protocol in details. In the algorithm, we use *PD*, *PF*, *PE* and *PO* to denote the corresponding operations of a wireless node u after it receives a packet p. *PD* means u dropped p, *PF* means u is forwarding or will forward p, *PE* means u has forwarded p, and *PO* means u is the source node of p.

Table I lists the data structures used in algorithm 4. Worth to mention that $u.fwdList(p)$ in the table has the same meaning as $Fwd(u)$ used before, and emphasizes the forwarder list is constructed for packet p.

TABLE I
DATA STRUCTURES

$u.id$	the unique identity of wireless node u
$p.sdr$	current sender node's identity of packet p
$p.des$	destination node identity of p
$ACK(p)$	ACK message corresponding to p
$u.fwdList(p)$	forwarder list that u constructs for p
$u(p).pri$	priority of u to forward p, this is indicated in the forwarder list contained in p
$u.pktList()$	Packet list of u, which records information of all data packets u received during a period of time

The example shown in the following Fig. 4 can be considered as a snapshot of a part of a wireless network gives us a simple case which can help us to understand and illustrate Algorithm 4. In Fig, 4, node u_1 could receive the same packet,

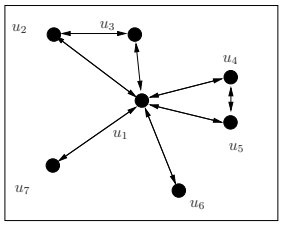

Fig. 4. A simple example in which node u_1 could receive same packet (p) for many times. Here, node $u_1 \in u_2.fwdList(p) \bigcap u_7.fwdList(p)$, $u_3 \in u_2.fwdList(p)$ and all three nodes u_4,u_5 and u_6 is belong to $u_2.fwdList$. The bidirectional arrows of a link means two end points of the link are neighbors of each other.

say p for many times. For instance, after node u_2 broadcasts p to u_1 and u_3, if $u_3(p).pri \geq u_1(p).pri$, u_3 will broadcast p to u_3's forwarder list first. Then u_1 could receive p from u_3 again. In addition, if u_7 has p (maybe due to multiple paths), u_7 will send p to u_1 sooner or later because u_1 is in the forwarder list of u_7. Moreover, after u_1 becomes a new source of p and forwards p to u_4, u_5, u_6, if u_1 is fortunate enough, u_1 could receive p from one of $\{u_4.u_5, u_6\}$ (the one has highest priority) only once if u_4, u_5, u_6 agree on which one of them will forward p. If they cannot agree with each other due to ACK loss or node u_6 cannot receive signal from u_4 or u_5 (showed in this case), u_1 will receive p again from u_6 and from either u_4 or u_5, or all of them due to ACK loss. By using Algorithm 4, clearly, some of transmission could be avoided. For example, if u_7 receive the p from u_1 when u_1 forwarded p to u_4, u_5, u_6, u_7 will never forward p to u_1. Or, if $u_4(p).pri \geq u_5(p).pri$, u_4 will forward p to next hop first, u_5 will not forward p again if it receives p again from u_4 even if the $ACK(p)$ sent by u_4 is lost. In addition, our simulation results show this packet handling mechanism can mitigate the duplications of packets extremely in most of cases.

Algorithm 4 EEOR Packet Handling Protocol

Input: Receiving a packet p

1: **if** p is a data packet **then**
2: **if** this is the first time for u to receive p **then**
3: **if** $u.id == p.des$ **then**
4: $broadcast(ACK(p))$; $pktList(p).status = PD$; save data packet p;
5: **if** $u.id == p.sdr)$ **then**
6: $Drop(p)$; $pktList(p).status = PD$;
7: **if** u is in the forwarder list of p **then**
8: $broadcast(ACK(p))$; $putPacketInBuffer(p)$; and $pktList(p).status = PF$;
9: **else**
10: $pktList(p).status = PD$;
11: **else**
12: **if** $pktList(p).status == PD$ **then**
13: $Drop(p)$;
14: **if** $(pktList(p).status == PO) \bigvee (pktList(p).status == PE)$ **then**
15: $Drop(p)$; $pktList(p).status = PD$;
16: **if** $pktList(p).status == PF$ **then**
17: **if** $p.sdr.pri \geq u(p).pri$ **then**
18: $Drop(p)$; $pktList(p).status = PD$;
19: **else**
20: $Drop(p)$;
21: **else**
22: **if** p is $ACK(q)$ **then**
23: **if** $(q \notin u.pktList) \bigvee (u.pktList(q).status == PD)$ **then**
24: Drop(p);
25: **else**
26: **if** $(u.id == q.sdr) \bigvee ((p.sdr)(q).pri > u(q).pri)$ **then**
27: $u.pckList(q).status = PD$; $Drop(q)$;
28: **else**
29: Update information based on beacon message p;

IV. PERFORMANCE EVALUATIONS

To illustrate the feasibility and performance of our protocol. We implemented both EEOR and ExOR on TOSSIM, TinyOS 2.0.2. on Ubuntu 7.0.4. and conducted extensive tests

based on different network environment. We implement ExOR following the descriptions in [1]. To compare two protocols fairly, we use same max forwarder list size for both and let batch size in ExOR equal to 1. We compared our simulation results with ExOR [1] for unicast case with respect to energy consumption, packet loss ratio, end-to-end delay and packet duplication ratio, etc. The experimental results showed that the performance of our protocol is better than ExOR. In addition, the simulation also examines some of the individual design and implementation decisions in our energy efficient opportunistic routing protocol, explores the consistency of our algorithms' performance and identified areas for improvement.

A. Network Description

We randomly deploy 100 wireless nodes with transmission range 50 feet in a 300×300 $feet^2$ square region. The node software uses default CSMA MAC protocol in TinyOS.

We randomly pick 18 pairs of wireless node as source/destination pairs and for each source/destination pair nodes u and v, u will generate a new packet per second, and send to v. Notice, the speed of generating new packet could change when the source node finds congestion in the network. We call the number of sending packets as *data size*. Considering the limited storage capacity of wireless sensor nodes, we set the buffer size to 50 bytes. After the buffer of a node is full, it will either drop new packet or replace old packet with new one according to different priorities of packets.

B. Performance Comparison

We compared our protocol with ExOR with respect to the total energy consumption, packet loss rate, end-to-end delay and packet duplication ratio. Due to different motes have different energy consumption parameters, we first considered and compared several operations of nodes which consume most of a node's energy, like sending and receiving. The Fig. 5 and 6 show the count of transmission and receiving (including receiving, snooping, intercepting) of all wireless nodes with EEOR and ExOR respectively.

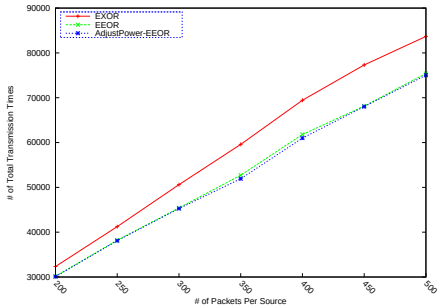

Fig. 5. Total transmission count for all nodes.

As we can see from the figures, both transmission and receiving count of ExOR are larger than EEOR's. This is because of the following reasons. First, for a node u in ExOR, it will always choose more neighbors (all neighbors sorted by ETX [16]) into forwarder list for a packet under the constraint of penalty. However, in EEOR, when a node u chooses forwarder list for a packet, it will not only considers the expected cost of sorted neighbors, but also considers the increment cost to add a node to the forwarder list such that u will stop to add a new neighbor to the forwarder list if doing so will lead the expected cost of u increasing. Second, a wireless node u's expected cost (ETX in ExOR) only depends on the neighbor which has smallest ETX value. However, the expected cost of a wireless node u in EEOR is determined both by the current selected forwarder list and by link error rates between u and nodes in the forwarder list, which is more reasonable. These two differences aforementioned between EEOR and ExOR make the average forwarder list size of the former is smaller than latter's in most of cases, thus EEOR involves fewer intermediate nodes. In addition, we also studied and compared the total transmission times and receiving times for each source/destination pair separately for both EEOR and ExOR. Let us take node pair $90 - 10$ as an example. Here, the hop number of the shortest path between nodes 90 and 10 is 5. When we let source node 90 send up to 500 packets to target node 10, the total receiving times of all nodes (involved in the transmissions between pair $90 - 10$) is 23080 by ExOR and is 20914 by EEOR separately. And the total transmission times for pair $90 - 10$ is 9396 by ExOR and 8357 by EEOR separately.

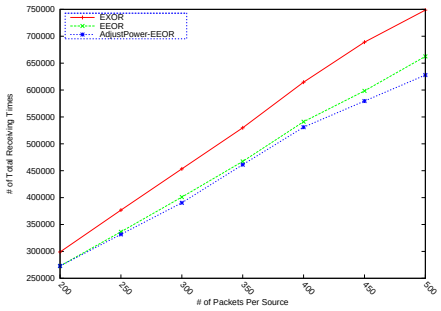

Fig. 6. Total receiving count for all nodes

Next, we measure the total energy consumption for both protocols based on the energy consumption parameters of *Tmote Sky* sensor node. For example, the energy consumption for one time transmission and receiving for Tmote Sky sensor node is $17.4mA$ and $19.7mA$ respectively. The Fig. 7 illustrates the total energy consumption for EEOR, AdjustablePower-EEOR and ExOR three protocols when we let the data size of each source node change from 200 to 500 given a fixed randomly topology and randomly chosen 18 source/destination pairs. As we can see, the total energy consumption for each protocol is increased with the data size of each source node. And for each case, the performance of our protocols is better than ExOR's.

To compare the packet loss rate, we set the data size of each source node equal to 500 and compared 18 source/destination pairs one by one for both protocols. The comparison results is shown in Fig. 8. As we can see, the average packet loss rate of each pair increases as the hop count increases between a

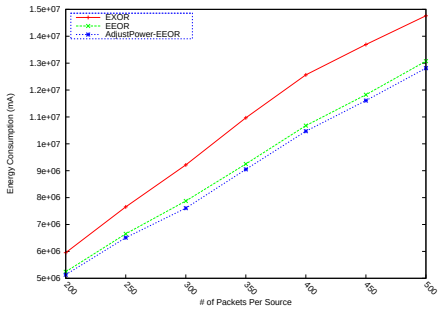

Fig. 7. Energy consumption for three protocols.

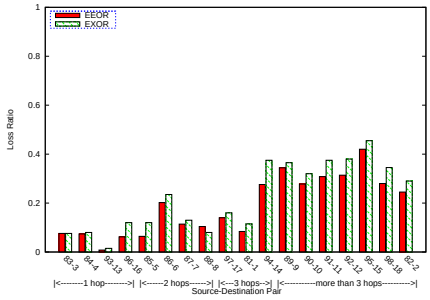

Fig. 8. Packet loss ratio.

source and a destination node. In addition, in most cases, the packet loss rate of EEOR is less than ExOR's.

The next comparison property is the end-to-end delay. We still let each source node send up to 500 packets towards its destination. We measure both average and max end-to-end delay time for each source/destination pair. Here, the definition of end-to-end delay of a packet is the time duration from a source node sent the packet to a destination received this packet. The average delay of each pair is illustrated in Fig.9 and the maximum delay for each pair is described in Fig. 10.

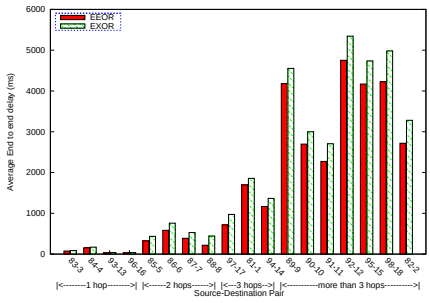

Fig. 9. Average delay for each pair.

As we can see, the end-to-end delay of EEOR is shorter than EXOR's. This is mainly because in ExOR, a wireless node u sorts the neighbors nodes only by ETX when it chooses the forwarder list for a packet, *i.e.*, the relatively further neighbors will be chosen if they have smaller ETX. However, the further distance between v and u increases the unreliability of used links as well, thus decrease the probability of u to transmit packets to v successfully. In EEOR, for a wireless node u,

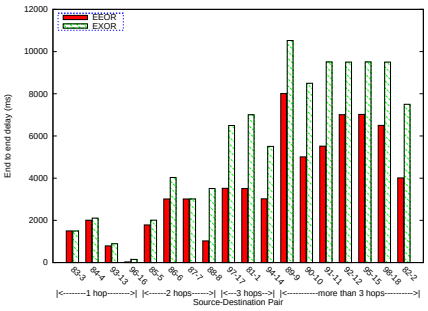

Fig. 10. Max delay for each pair.

we considered both the expected cost of a neighbor node v and the link error rate between u and v, in other words, we combined both the distance and reliability of the link between u and v, this will lead u to choosing further nodes with good link quality to the forwarder list.

The last property we compared our protocol with ExOR is the packet duplication ratio. Here the main reason for us to test packet duplication ratio is because both EEOR and ExOR are multi-path routing protocols such that in most of cases they cannot avoid the fact that same packets will be relayed to the destination node through multiple paths, thus increases the overhead of wireless networks. Thus, multi-path property for unicast on the one hand decrease the packet loss ratio and energy consumption to some extent, on the other hand increase the overhead of the whole network. Fortunately, through our simulation results, the overhead increased by multi-path property is not much. The reason is because for both protocols, a forwarder list for each node constraints the area in which a packet can travel in the network, and at most times, these multiple paths will converge to some node(s) or at least cross with each other. The result of duplication packet ratio is shown in Fig.11. Here, the definition of repeat times is

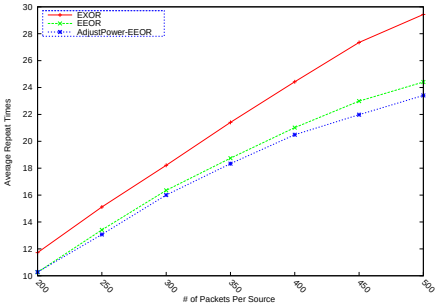

Fig. 11. Packet duplication count.

the average times that a wireless node is required to forward how many duplicated packets for each source/destination pair.

V. Previous Works

A number of energy efficient routing protocols [4], [10], [11] have been proposed recently combining with a variety techniques (dynamic transmission power adjustment, adaptive sleeping, topology control, multi-path routing, directional

antennas, etc). Many existing power-aware routing protocols often assumed that the wireless links of a wireless network are reliable and then try to theoretically prove performance guarantees [5], [12], [13]. Srinivas and Modiano [11] investigate the problem of minimum energy node/link disjoint paths routing in multi-hop wireless networks. Clearly, such schemes result in increased energy consumption, compared with the minimum energy single path. More importantly, they do not consider link error rates. Those simplified network model helps to understand the fundamental theory limits, however, it is clearly too optimistic in practice: the wireless communications are unreliable and often unpredictable.

A number of protocols have been proposed recently to remedy or even utilize the unreliability of the wireless channels such as using multi-path routing [7], [8], building reliable backbone [13], [6], and using energy efficient reliable routing structure [3], [15]. In [3], Dong and Banerjee addressed the problem of energy-efficient reliable wireless communication in the presence of unreliable or lossy wireless link layers in multi-hop wireless networks. The main focus of the paper is on single path routing. They first presented two centralized algorithms, BAMER and GAMER, that optimally solve the minimum energy reliable communication problem in presence of unreliable links. Subsequently they presented a distributed algorithm, DAMER, that approximates the performance of the centralized algorithm and leads to significant performance improvement over existing single-path or multi-path based techniques. Banerjee and Misra [15] explored the effect of lossy links on energy efficient routing and solved the problem of finding the minimum energy paths in the hop-by-hop retransmission model. Let w and p denote the transmission power and the error rate of a hop-by-hop retransmission link, respectively. They proposed the link cost to be $\frac{w}{1-p}$, which is actually the expected energy consumption of delivering a packet over that link. For the hop-by-hop retransmission model, it is then straightforward to use a traditional shortest path algorithm (e.g. Dijkstra's algorithm) to compute minimum energy paths. [17] studied power assignment problem for wireless networks with unreliable links in order to save energy.It assigns transmission power to each node based on a given topology and link error rates for all links such that the total energy consumption is minimized;

However, these methods all follow a conventional design principle in network layer of wired networks: after the best path(s) between a source and destination is calculated, all data flows from source to destination follow the selected path(s) until the path is updated after certain routing update period. ExOR [1] challenges this conventional design principle in network layer. ExOR broadcasts each packet, choosing a receiver to forward only *after* learning the set of nodes which actually received the packet. Delaying forwarding decisions until after reception allows ExOR to try multiple long but radio lossy links concurrently, resulting in high expected progress per transmission. The key challenge in realizing ExOR is to ensure that only the best receiver of each packet forwards it, in order to avoid duplication. ExOR operates on batches of packets in order to reduce the communication cost of agreement. The source node includes in each packet a list of candidate forwarders prioritized by *closeness* to the destination. Receiving nodes buffer successfully received packets and await the end of the batch. The highest priority forwarder then broadcasts the packets in its buffer, including its copy of the "batch map" in each packet. The remaining forwarders then transmit in order, but only send packets which were not acknowledged in the batch maps of higher priority nodes. The forwarders continue to cycle through the priority list until the destination has 90% of the packets. The remaining packets are transferred with traditional routing. In contrast, our EEOR protocol uses energy cost to rank forwarders and we are able to prove that the expected cost for routing a packets by all possible nodes is minimized. MORE [2] presents a MAC-independent opportunistic routing protocol. MORE randomly mixes packets before forwarding them. This randomness ensures that routers that hear the same transmission do not forward the same packets. Thus, MORE needs no special scheduler to coordinate routers and can run directly on top of 802.11. Experimental results from a 20-node wireless testbed show that MORE's median unicast throughput is 22% higher than ExOR, and the gains rise to 45% over ExOR when there is a chance of spatial reuse. For multicast,MORE's gains increase with the number of destinations, and are $35 - 200\%$ greater than ExOR.

VI. Conclusion and Future Work

In the paper, we investigated how to select and prioritize forwarder list to minimize energy consumptions for unicast. We study both cases that the transmission power of each node is fixed or changeable. The algorithms to select and prioritize forwarder list in both cases are presented and analyzed. Worth to mention that, our methods do not assume any geometrical properties or energy models. We conducted extensive simulations in TOSSIM to study the performance of proposed algorithms by comparing it with ExOR [1].

Several problems can be the possible future directions. For instance, how to select optimum forwarder list for multicast and broadcast; and, how to combine opportunistic routing with network coding and link layer error corrections [14] to improve throughput and reduce retransmissions.

References

[1] Sanjit Biswas and Robert Morris. Exor: opportunistic multi-hop routing for wireless networks. In *ACM SIGCOMM*, pages 133–144, 2005.

[2] Szymon Chachulski, Michael Jennings, Sachin Katti, and Dina Katabi. Trading structure for randomness in wireless opportunistic routing. In *ACM SIGCOMM*, 2007.

[3] Qunfeng Dong, Suman Banerjee, Micah Adler, and Archan Misra. Minimum energy reliable paths using unreliable wireless links. In *ACM MobiHoc*, pages 449–459, 2005.

[4] Robin Kravets and P. Krishnan. Power management techniques for mobile communication. In *ACM MobiCom*, 1998.

[5] Xiang-Yang Li, Wen-Zhan Song, and Weizhao Wang. A unified energy-efficient topology for unicast and broadcast. In *ACM MobiCom*, pages 1–15, 2005.

[6] Manki Min, Feng Wang, Ding-Zhu Du, and Panos M. Pardalos. A reliable virtual backbone scheme in mobile ad-hoc networks. In *IEEE MASS*, 2004.

[7] A. Nasipuri, R. Castaneda, and S. R. Das. Performance of multipath routing for on-demand protocols in ad hoc networks. *ACM/Kluwer Mobile Networks and Applications (MONET)*, 6(4):339–349, 2001.

[8] J. Raju and J. Garcia-Luna-Aceves. A new approach to on-demand loop-free multipath routing. In *IEEE ICCCN*, pages 522–527, 1999.

[9] T.S. Rappaport. *Wireless Communications: Principles and Practices*. Prentice Hall, 1996.

[10] Volkan Rodoplu and Teresa H. Meng. Minimum energy mobile wireless networks. In *IEEE ICC*, volume 3, 1998.

[11] Anand Srinivas and Eytan Modiano. Minimum energy disjoint path routing in wireless ad-hoc networks. In *ACM MobiHoc*, pages 122–133. 2003.

[12] Peng-Jun Wan, G. Calinescu, Xiang-Yang Li, and Ophir Frieder. Minimum-energy broadcast routing in static ad hoc wireless networks. *ACM Wireless Networks*, 2002, and IEEE INFOCOM 2000.

[13] Yu Wang, WeiZhao Wang, and Xiang-Yang Li. Distributed low-cost backbone formation for wireless ad hoc networks. In *ACM MobiHoc*, pages 2–13, 2005.

[14] Henri Dubois-Ferriere, Deborah Estrin and Martin Vetterli. Packet Combining in Sensor Networks In *ACM SENSYS*, 2005.

[15] Suman Banerjee and Archan Misra. Minimum energy paths for reliable communication in multi-hop wireless networks. In *ACM MobiHoc*, pages 146–156, 2002.

[16] D. De Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *ACM MobiCom*, 2003.

[17] Xiang-Yang Li, YanTai Shu, HaiMing Chen, XiaoWen Chu and YanWei Wu. Energy Efficient Routing With Unreliable Links in Wireless Networks. In *The Third IEEE MASS*, 2006.