Minimum-Energy Topology Control Algorithms in Ad Hoc Networks[1]

Joseph Y. Halpern
Dept. of Computer Science
Cornell University
Ithaca NY 14853
halpern@cs.cornell.edu

Li (Erran) Li
Center for Networking Research
Bell Labs, Lucent
Holmdel, NJ 07733
erranlli@dnrc.bell-labs.com

August 7, 2004

0.1 Introduction

Wireless ad hoc networks can be deployed in many settings such as environment monitoring, disaster relief, and battlefield situations. In these settings, wireless devices such as sensors are often powered by an on-board battery. Many of these networks are expected to function for an extended period of time. In order to accomplish this without a renewable energy source, energy conservation is the key.

We consider how to adjust a node's transmission power to minimize its energy consumption and improve network performance in terms of network lifetime and throughput. We refer to this problem as the *topology-control* problem. Our focus here is on maximizing the time that the network is able to function, i.e., the *network lifetime*. We discuss below how network lifetime can be increased, the subtleties of defining it precisely, and the difficulties of achieving optimal network performance in practice.

In an ad hoc network, network lifetime can be increased by energy reduction in the hardware, the software (operating systems and applications), and the communication protocols. To reduce the energy consumption of hardware, low-power CPUs such as the Intel embedded StrongARM 1100 processor and low-power displays have been developed. To reduce the energy consumption of software, low-energy software can be developed through various techniques, including reducing the number of operations through code optimization and the use of multiple fidelity algorithms [SN99]. The synergy of hardware and application software can also be exploited by operating systems to reduce energy consumption. For example, CPU energy consumption can be reduced through dynamic voltage scaling if the computation workload decreases [PLS01, FRM01]. In addition, a disk can be spun down to reduce its idle-time energy consumption.

We focus on the design of energy-efficient communication protocols. A radio consumes energy at all times when sending, when receiving, and when idle. (Studies have shown that power consumption during the idle state cannot be ignored [SK97].) This suggests two complementary approaches to reducing radio energy consumption: (1) minimizing energy consumption due to idle time or due to passively listening to transmissions not addressed to a node itself and (2) minimizing energy consumption due to communication. Protocols that minimize idle-time energy consumption have been proposed in [SR98, XHE01, CJBM01]. We restrict our attention to minimizing energy consumption due to communication.

Ideally one would like to design a general-purpose communication protocol that maximizes network lifetime. However, the notion of network lifetime is application dependent. There are a number of reasonable notions. For example, for event-monitoring applications, one wants to maximize the time network monitoring centers are able to receive information about events happening in the field. For data-gathering applications, one may wants to maximize the time until a certain percentage of nodes cannot deliver data to the data-gathering centers. For mission-critical applications, one might want to maximize the time until the first message cannot be delivered. Because of the application-dependent nature of the definition of network lifetime, it seems doubtful that there will be a general solution that is appropriate for all settings.

Of course, network lifetime is only one of several network-performance metrics of interest. Other metrics, such as throughput and latency, are also important. Optimizing one metric can adversely impact another metric. For example, to maximize network lifetime, energy-efficient routes tend to be chosen. An energy-efficient route has more hops in general than the corresponding shortest route. This may lead to longer latency.

The problem of optimizing network performance is perhaps best viewed in terms of decision theory. Suppose we assume that, for each possible outcome of the algorithm, we can associate a utility. This utility would trade off the various features of an outcome, such as latency, throughput, and lifetime. If there is a probability distribution on outcomes, an optimal protocol is one that maximizes the expected utility. Because utilities and probability distributions on outcomes are difficult to obtain in practice, instead of trying to achieve application-specific optimal solutions, we focus on general heuristics for reducing communication-energy consumption.

To reduce energy consumption, it is typically better to relay messages through intermediate nodes rather than sending a message directly to the intended recipient. This is because radio-signal attenuation is inversely proportional to the nth power of the distance a signal propagates [Rap96], where n is between 2 and 6. Thus, relaying through intermediate nodes can reduce total power consumption. In addition, if a node sends a message directly to a distant receiver, it must use greater power and is more likely to interfere with the transmissions of other nodes.

While reducing broadcast power reduces power consumption and minimizes interference, we do not want to lose routes in the process. Suppose that each node u broadcasts with power $p(u)$. The resulting communication graph has an edge from u to v iff u can reach v when broadcasting with power $p(u)$. Since we do not want to lose routes, a minimal requirement on the choice of $p(u)$ is that if there is a route between a pair of nodes in the communication graph that results if each node broadcasts with maximum power, then there is a route in the communication graph that results if each node u broadcasts with power $p(u)$. But choosing $p(u)$ to satisfy this minimal constraint may not be the best choice in terms for reducing power consumption.

Let $p(u, v)$ denote the minimal power needed to send a point-to-point message between u and v. A *minimum-energy path* between a pair u and v of nodes is the path that requires the least amount of energy to send a message between u and v, provided that power $p(u', v')$ is used to transmit messages between neighboring nodes u' and v' on the path. To minimize power consumption for unicast (i.e., point-to-point) messages, it is typically best if each node broadcasts with enough power so that the minimal-energy path for any given node pair still exists in the resulting communication graph. A protocol for determining the broadcast power with this property is said to have the *minimum-energy property*. If a protocol has the minimum-energy property, then a suitable routing protocol can be used to find the minimum-energy path between any node pair. In this chapter, we present topology-control algorithms based on finding minimum-energy paths.

The rest of the chapter is organized as follows. Section 0.2 gives the network model. Section 0.3 identifies a condition necessary and sufficient for achieving the minimum-energy property. This characterization is used in Section 0.4 to construct the SMECN protocol. We prove that it has the minimum-energy property and that it constructs a network smaller than that constructed by Rodoplu and Meng [RM99] if the broadcast region is circular. Our SMECN requires location information which is usually obtained from a GPS unit. In Section 0.5, we show how SMECN can be used to deal with topology changes as well. In Section 0.6, we give the results of simulations showing the energy savings obtained by using the network constructed by SMECN. We summarize in Section 0.7.

0.2 The Model

We assume that a set V of nodes is deployed in a two-dimensional region, where no two nodes are in the same physical location. Each node has a GPS receiver on board, so knows it own location.

It does not necessarily know the location of other nodes. Moreover, the location of nodes will in general change over time.

A transmission between node u and v takes power $p(u, v) = td(u, v)^n$ for some appropriate constant t, where $n \geq 2$ is the path-loss exponent of outdoor radio-propagation models [Rap96], and $d(u, v)$ is the distance between u and v. A reception at the receiver takes power c. This power expenditure at the receiver is referred to as the receiver power. Computational power consumption is ignored.

Suppose there is some maximum power p_{max} at which the nodes can transmit. Thus, there is a graph $G_R = (V, E_R)$ where $(u, v) \in E_R$ if u can reach v when using power p_{max}. Clearly, if $(u, v) \in E_R$, then $td(u, v)^n \leq p_{max}$. However, we do not assume that a node u can transmit to all nodes v such that $td(u, v)^n \leq p_{max}$. For one thing, there may be obstacles between u and v that prevent transmission. Even without obstacles, if a unit transmits using a directional transmit antenna, then only nodes in the region covered by the antenna (typically a cone-like region) will receive the message. Rodoplu and Meng [RM99] implicitly assume that every node can transmit to every other node. Here we take a first step in exploring what happens if this is not the case. However, we do assume that the graph G_R is connected, so that there is a potential communication path between every pair of nodes in V.

Because the power required to transmit between a pair of nodes increases as the nth power of the distance between them, for some $n \geq 2$, it may require less power to relay information than to transmit directly between two nodes. As usual, a *path* $r = (u_0, \ldots, u_k)$ in a graph $G = (V, E)$ is defined to be an ordered list of nodes such that $(u_i, u_{i+1}) \in E$. The *length* of $r = (u_0, \ldots, u_k)$, denoted $|r|$, is k. The total power consumption of a path $r = (u_0, u_1, \cdots, u_k)$ in G_R is the sum of the transmission and receiver power consumed, i.e.,

$$C(r) = \sum_{i=0}^{k-1} (p(u_i, u_{i+1}) + c).$$

A path $r = (u_0, \ldots, u_k)$ is a *minimum-energy path* from u_0 to u_k if $C(r) \leq C(r')$ for all paths r' in G_R from u_0 to u_k. For simplicity, we assume that $c > 0$. (Our results hold even without this assumption, but it makes the proofs a little easier.) A subgraph $G = (V, E)$ of G_R has the *minimum-energy property* if, for all $u, v \in V$, there is a path r in G that is a minimum-energy path in G_R from u to v.

0.3 A Characterization of Minimum-Energy Communication Networks

Our goal is to find a minimal subgraph G of G_R that has the minimum-energy property. Note that a graph G with the minimum-energy property must be connected since, by definition, it contains a path between every pair of nodes.

The intention is to have the nodes communicate using the links in G. To do this, it must be possible for each of the nodes in the network to construct G (or, at least, the relevant portion of G from their point of view) in a distributed way. In this section, we provide a condition that is necessary and sufficient for a subgraph of G_R to be minimal with respect to the minimum-energy property. In the next section, we use this characterization to provide an efficient distributed

algorithm for constructing a graph G with the minimum-energy property that, while not necessarily minimal, still has relatively few edges.

Clearly if a subgraph $G = (V, E)$ of G_R has the minimum-energy property, an edge $(u, v) \in E$ is *redundant* if there is a path r from u to v in G such that $|r| > 1$ and $C(r) \leq C(u, v)$. Let $G_{\min} = (V, E_{\min})$ be the subgraph of G_R such that $(u, v) \in E_{\min}$ iff there is no path r from u to v in G_R such that $|r| > 1$ and $C(r) \leq C(u, v)$. As the next result shows, $G_{\min}$ is the smallest subgraph of G_R with the minimum-energy property.

Theorem 0.3.1 *A subgraph G of G_R has the minimum-energy property iff it contains $G_{\min}$ as a subgraph. Thus, $G_{\min}$ is the smallest subgraph of G_R with the minimum-energy property.*

Proof: We first show that $G_{\min}$ has the minimum-energy property. Suppose, by way of contradiction, that there are nodes $u, v \in V$ and a path r in G_R from u to v such that $C(r) < C(r')$ for any path r' from u to v in $G_{\min}$. Suppose that $r = (u_0, \dots, u_k)$, where $u = u_0$ and $v = u_k$. Without loss of generality, we can assume that r is the longest minimal-energy path from u to v. Note that r has no repeated nodes because any cycle can be removed to give a path that requires strictly less power. Thus, the length of a minimum-length path is bounded by $|V|$. Since $G_{\min}$ has no redundant edges, for all $i = 0, \dots, k-1$, it follows that $(u_i, u_{i+1}) \in E_{\min}$. For otherwise, there is a path r_i in G_R from u_i to u_{i+1} such that $|r_i| > 1$ and $C(r_i) \leq C(u_i, u_{i+1})$. But then it is immediate that there is a path r^* in G_R such that $C(r^*) \leq C(r)$ and r^* is longer than r, contradicting the choice of r.

To see that $G_{\min}$ is a subgraph of every subgraph of G_R with the minimum-energy property, suppose that there is some subgraph G of G_R with the minimum-energy property that does not contain the edge $(u, v) \in E_{\min}$. Thus, there is a minimum-energy path r from u to v in G. It must be the case that $C(r) \leq C(u, v)$. Since (u, v) is not an edge in G, we must have $|r| > 1$. But then $(u, v) \notin E_{\min}$, a contradiction. ∎

This result shows that in order to find a subgraph of G with the minimum-energy property, it suffices to ensure that it contains $G_{\min}$ as a subgraph.

0.4 A Power-Efficient Protocol for Finding a Minimum-Energy Communication Network

Checking if an edge (u, v) is in $E_{\min}$ may require checking nodes that are located far from u. This may require a great deal of communication, possibly to distant nodes, and thus require a great deal of power. Since power-efficiency is an important consideration in practice, we consider here an algorithm for constructing a communication network that contains $G_{\min}$ and can be constructed in a power-efficient manner rather than trying to construct $G_{\min}$ itself.

Say that an edge $(u, v) \in E_R$ is *k-redundant* if there is a path r in G_R such that $|r| = k$ and $C(r) \leq C(u, v)$. Notice that $(u, v) \in E_{\min}$ iff it is not k-redundant for all $k > 1$. Let E_2 consist of all and only edges in E_R that are not 2-redundant. In our algorithm, we construct a graph $G = (V, E)$ where $E \supseteq E_2$; in fact, under appropriate assumptions, $E = E_2$. Clearly $E_2 \supseteq E_{\min}$, so G has the minimum-energy property.

There is a trivial algorithm for constructing E_2. Each node u starts the process by broadcasting a "Hello" message at maximum power p_{max}, stating its own position. If a node v receives this message, it responds to u with a *Ack* message stating its location. Let $M(u)$ be the set of nodes that respond to u and let $N_2(u)$ denote u's neighbors in E_2. Clearly $N_2(u) \subseteq M(u)$. Moreover, it is easy to check that $N_2(u)$ consists of all those nodes $v \in M(u)$ other than u such that there is no $w \in M(u)$ such that $C(u, w, v) \leq C(u, v)$. Since u has the location of all nodes in $M(u)$, $N_2(u)$ is easy to compute.

The problem with this algorithm is in the first step, which involves a broadcast using maximum power. While this expenditure of power may be necessary if there are relatively few nodes, so that power close to p_{max} will be required to transmit to some of u's neighbors in E_2, it is unnecessary in denser networks. In this case, it may require much less than p_{max} to find u's neighbors in E_2. We now present a more power-efficient algorithm for finding these neighbors than the one proposed by Rodoplu and Meng [RM99] (We refer to their protocol as MECN for *Minimum Energy Communication Network*) . Let $F(u, p)$ be the region that u can reach if it broadcasts with power p. For this algorithm, we assume that u knows $F(u, p)$. If there are no obstacles and the antenna is omni-directional, then $F(u, p)$ is just a circle of radius d_p such that $td_p^n = p$. We are implicitly assuming that even if there are obstacles or the antenna is not omni-directional, a node u knows the terrain and the antenna characteristics well enough to compute $F(u, p)$.

Before presenting the algorithm, it is useful to define a few terms.

Definition 0.4.1 Given a node v, let *Loc*(v) denote the physical location of v. The *relay region* of the transmit-relay node pair (u, v) is the physical region $R_{u \to v}$ such that relaying through v to any point in $R_{u \to v}$ takes less power than direct transmission. Formally,

$$R_{u \to v} = \{(x, y) : C(u, v, (x, y)) \leq C(u, (x, y))\},$$

where we abuse notation and take $C(u, (x, y))$ to be the cost of transmitting a message from u to a virtual node whose location is (x, y). That is, if there were a node v' such that *Loc*$(v') = (x, y)$, then $C(u, (x, y)) = C(u, v')$; similarly, $C(u, v, (x, y)) = C(u, v, v')$. Note that, if a node v is in the relay region $R_{u \to w}$, then the edge (u, v) is 2-redundant. Moreover, since $c > 0$, $R_{u \to u} = \emptyset$.

Given a region F, let

$$N_F = \{v \in V : Loc(v) \in F\};$$

if F contains u, let

$$R_F(u) = \bigcap_{w \in N_F} \left(F(u, p_{max}) - R_{u \to w} \right). \tag{1}$$

Intuitively, N_F consists of the nodes in region F, while $R_F(u)$ consists of those points that can be reached by u transmitting at maximum power other than those for which routing through some node in N_F would be more energy efficient than direct communication.

The following proposition gives a useful characterization of $N_2(u)$.

Proposition 0.4.2 *Suppose that F is a region containing the node u. If $F \supseteq R_F(u)$, then $N_{R_F(u)} \supseteq N_2(u)$. Moreover, if F is a circular region with center u and $F \supseteq R_F(u)$, then $N_{R_F(u)} = N_2(u)$.*

Proof: Suppose that $F \supseteq R_F(u)$ and that $N_{R_F(u)} \supseteq N_2(u)$. Suppose that $v \in N_2(u)$. Then clearly $Loc(v) \notin \cup_{w \in V} R_{u \to w}$ and $Loc(v) \in F(u, p_{max})$. Thus, $Loc(v) \in R_F(u)$, so $v \in N_{R_F(u)}$. Since v was chosen arbitrarily, it follows that $N_2(u) \subseteq N_{R_F}(u)$.

Now suppose that F is a circular region with center u and $F \supseteq R_F(u)$. We now show that $N_{R_F(u)} \subseteq N_2(u)$. Suppose that $v \in N_{R_F(u)}$. If $v \notin N_2(u)$, then there exists some w such that $C(u, w, v) \leq C(u, v)$. Since transmission costs increase with distance, it must be the case that $d(u, w) \leq d(u, v)$. Since $v \in N_{R_F(u)} \subseteq N_F$ and F is a circular region with center u, it follows that $w \in N_F$. Since $C(u, w, v) \leq C(u, v)$, it follows that $Loc(v) \in R_{u \to w}$. Thus, $v \notin R_F(u)$, contradicting our original assumption. Thus, $v \in N_2(u)$. ∎

The algorithm for node u constructs a set F such that $F \supseteq R_F(u)$, and tries to do so in a power-efficient fashion. By Proposition 0.4.2, the fact that $F \supseteq R_F(u)$ ensures that $N_{R_F(u)} \supseteq N_2(u)$. Thus, the nodes in $N_{R_F(u)}$ other than u itself are taken to be u's neighbors. By Theorem 0.3.1, the resulting graph has the minimum-energy property.

Essentially, the algorithm for node u starts by broadcasting a "Hello" message with some initial power p_0, getting Acks from all nodes in $F(u, p_0)$, and checking if $F(u, p_0) \supseteq R_{F(u,p_0)}(u)$. If not, it transmits with more power. It continues increasing the power p until $F(u, p) \supseteq R_{F(u,p)}(u)$. It is easy to see that $F(u, p_{max}) \supseteq R_{F(u,p_{max})}(u)$, so that as long as the power increases to p_{max} eventually, then this process is guaranteed to terminate. We do not investigate here how to the initial power p_0, nor do we investigate how to increase the power at each step. We simply assume some function *Increase* such that *Increase*$^k(p_0) = p_{max}$ for sufficiently large k. An obvious choice is to take *Increase*$(p) = 2p$. If the initial choice of p_0 is less than the power actually needed, then it is easy to see that this guarantees that u's estimate of the transmission power needed to reach a node v will be within a factor of 2 of the minimum transmission power actually needed to reach v.[1]

Thus, the protocol run by node u is simply

$p = p_0$;
while $F(u, p) \not\supseteq R_{F(u,p)}(u)$ do *Increase*(p);
$N(u) = N_{R_{F(u,p)}}$

A more careful implementation of this algorithm is given in Figure 1. Note that we also compute the minimum power $p(u)$ required to reach all the nodes in $N(u)$. In the algorithm, A is the set of all the nodes that u has found so far in the search and M consists of the new nodes found in the current iteration. In the computation of η in the second-last line of the algorithm, we take $\cap_{v \in M}(F(u, p_{max}) - R_{u \to v})$ to be $F(u, p_{max})$ if $M = \emptyset$. For future reference, we note that it is easy to show that, after each iteration of the while loop, we have that $\eta = \cap_{v \in A}(F(u, p_{max}) - R_{u \to v})$.

Define the graph $G = (V, E)$ by taking $(u, v) \in E$ iff $v \in N(u)$, as constructed by the algorithm in Figure 1. It is immediate from the earlier discussion that $E \supseteq E_2$. Thus, the following theorem holds.

Theorem 0.4.3 *G has the minimum-energy property.*

---

[1]Note that, in practice, a node may control a number of directional transmit antennae. Our algorithm implicitly assumes that they all transmit at the same power. This was done for ease of exposition. It would be easy to modify the algorithm to allow each antenna to transmit using different power. All that is required is that after sufficiently many iterations, all antennae transmit at maximum power.

Algorithm SMECN
$p = p_0;$
$A = \emptyset;$
$NonNbrs = \emptyset;$
$\eta = F(u, p_{max});$
while $F(u, p) \not\supseteq \eta$ do
    $p = Increase(p);$
    Broadcast "Hello" message with power p and gather $Acks$;
    $M = \{v \mid Loc(v) \in F(u, p), v \notin A, v \neq u\};$
    $A = A \bigcup M;$
    for each $v \in M$ do
        for each $w \in A$ do
            if $Loc(v) \in R_{u \to w}$ then $NonNbrs = NonNbrs \bigcup \{v\};$
            else if $Loc(w) \in R_{u \to v}$ then $NonNbrs = NonNbrs \bigcup \{w\};$
    $\eta = \eta \cap \bigcap_{v \in M} (F(u, p_{max}) - R_{u \to v});$
$N(u) = A - NonNbrs;$
$p(u) = \min\{p : F(u, p) \supseteq \eta\}$

Figure 1: Algorithm SMECN running at node u.

We next show that SMECN dominates MECN. MECN is described in Figure 2. For easier comparison, we have made some inessential changes to MECN to make the notation and presentation more like that of SMECN. The main difference between SMECN and MECN is the computation of the region η. As we observed, in SMECN, $\eta = \cap_{v \in A}(F(u, p_{max}) - R_{u \to v})$ at the end of every iteration of the loop. On the other hand, in MECN, $\eta = \cap_{v \in A - NonNbrs}(F(u, p_{max}) - R_{u \to v})$. Moreover, in SMECN, a node is never removed from $NonNbrs$ once it is in the set, while in MECN, it is possible for a node to be removed from $NonNbrs$ by the procedure $Flip$. Roughly speaking, if a node $v \in R_{u \to w}$, then, in the next iteration, if $w \in R_{u \to t}$ for a newly discovered node t, but $v \notin R_{u \to t}$, node v will be removed from $NonNbrs$ by $Flip(v)$. In [RM99], it is shown that MECN is correct (i.e., it computes a graph with the minimum-energy property) and terminates (and, in particular, the procedure $Flip$ terminates). Here we show that, at least for circular search regions, SMECN does better than MECN.

Theorem 0.4.4 *If the search regions considered by the algorithm SMECN are circular, then the communication graph constructed by SMECN is a subgraph of the communication graph constructed by MECN.*

Proof: For each variable x that appears in SMECN, let x_S^k denote the value of x after the kth iteration of the loop; similarly, for each variable in MECN, let x_M^k denote the value of x after the kth iteration of the loop. It is almost immediate that SMECN maintains the following invariant: $v \in NonNbrs_S^k$ iff $v \in A_S^k$ and $Loc(v) \in \cup_{w \in A_S^k} R_{u \to w}$. Similarly, it is not hard to show that MECN maintains the following invariant: $v \in NonNbrs_M^k$ iff $v \in A_S^k$ and $Loc(v) \in \cup_{w \in A_M^k - NonNbrs_M^k} R_{u \to w}$. (Indeed, the whole point of the $Flip$ procedure is to maintain this invariant.) Since it is easy to check that $A_S^k = A_M^k$, it is immediate that $NonNbrs_S^k \supseteq NonNbrs_M^k$. Suppose that SMECN terminates after k_S iterations of the loop and MECN terminates after k_M iterations of the loop. Hence $\eta_S^k \subseteq \eta_M^k$ for all $k \leq \min(k_S, k_M)$. Since both algorithms use the

Algorithm MECN
$p = p_0$;
$A = \emptyset$;
$NonNbrs = \emptyset$;
$\eta = F(u, p_{max})$;
while $F(u, p) \not\supseteq \eta$ do
      $p = Increase(p)$;
      Broadcast "Hello" message with power p and gather *Acks*;
      $M = \{v \mid Loc(v) \in F(u, p), v \notin A, v \neq u\}$;
      $A = A \bigcup M$;
      $NonNbrs = NonNbrs \bigcup M$;
      for each $v \in M$ do $Flip(v)$;
      $\eta = \bigcap_{v \in (A - NonNbrs)} (F(u, p_{max}) - R_{u \to v})$;
$N(u) = A - NonNbrs$;
$p(u) = \min\{p : F(u, p) \supseteq \eta\}$

Procedure $Flip(v)$
      if $v \notin NonNbrs$ then $NonNbrs = NonNbrs \bigcup \{v\}$;
          for each $w \in A$ such that $Loc(w) \in R_{u \to v}$ do $Flip(w)$;
      else if $Loc(v) \notin \cup_{w \in A - NonNbrs} R_{u \to w}$ then $NonNbrs = NonNbrs - \{v\}$;
          for each $w \in A$ such that $Loc(w) \in R_{u \to v}$ do $Flip(w)$;

Figure 2: Algorithm MECN running at node u.

condition $F(u, p) \supseteq \eta$ to determine termination, it follows that SMECN terminates no later than MECN; that is, $k_S \leq k_M$.

Since the search region used by SMECN is assumed to be circular, by Proposition 0.4.2, $A_S^{k_S} - NonNbrs_S^{k_S} = N_2(u)$. Moreover, even if we continue to iterate the loop of SMECN (ignoring the termination condition), then $F(u, p)$ keeps increasing while η keeps decreasing. Thus, by Proposition 0.4.2 again, we continue to have $A_S^k - NonNbrs_S^k = N_2(u)$ even if $k \geq k_S$. That means that if we were to continue with the loop after SMECN terminates, none of the new nodes discovered would be neighbors of u. Since the previous argument still applies to show that $NonNbrs_S^{k_M} \supseteq NonNbrs_M^{k_M}$, it follows that $N_2(u) = A_S^{k_M} - NonNbrs_S^{k_M} \subseteq A_M^{k_M} - NonNbrs_M^{k_M}$. That is, the communication graph constructed by SMECN has a subset of the edges of the communication graph constructed by MECN. ∎

In the proof of Theorem 0.4.4, we implicitly assumed that both SMECN and MECN use the same value of initial value p_0 of p and the same function *Increase*. In fact, this assumption is not necessary, since the neighbors of u in the graph computed by SMECN are given by $N_2(u)$ independent of the choice of p_0 and *Increase*, as long as $F(u, p_0) \not\supseteq F(u, p_{max})$ and $Increase^k(p_0) \geq p_{max}$ for k sufficiently large. Similarly, the proof of Theorem 0.4.4 shows that the set of neighbors of u computed by MECN is a superset of $N_2(u)$, as long as *Increase* and p_0 satisfy these assumptions.

Theorem 0.4.4 shows that the neighbor set computed by MECN is a superset of $N_2(u)$. As the following example shows, it may be a strict superset (so that the communication graph computed by SMECN is a strict subgraph of that computed by MECN).

Example 0.4.1 *Consider a network with 4 nodes t, u, v, w, where $Loc(v) \in R_{u \to w}$, $Loc(w) \in R_{u \to t}$, and $Loc(v) \notin R_{u \to t}$. As shown in Figure 3, it is not hard to choose power functions and locations for the nodes that have this property. It follows that $N_2(u) = \{t\}$. (It is easy to check that $Loc(t) \notin R_{u \to v} \cup R_{u \to w}$.) On the other hand, suppose that Increase is such that t, v, and w are added to A in the same step. Then all of them are added to $NonNbrs$ in MECN.*
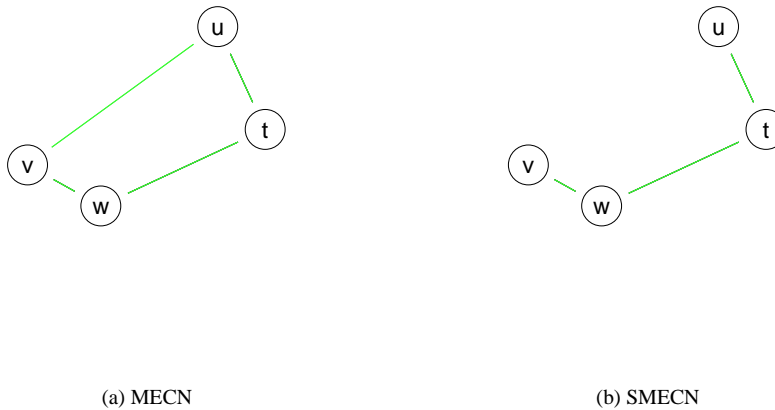

(a) MECN   (b) SMECN

Figure 3: A network where SMECN dominates MECN.

0.5 Reconfiguration

In a multi-hop wireless network, nodes can be mobile. Even if nodes do not move, nodes may die if they run out of energy. In addition, new nodes may be added to the network. We assume that each node uses a Neighbor Discovery Protocol (NDP), a periodic message that provides all its neighbors with its current position (according to the GPS) in order to detect changes in the topology of the network. A node u sends out the message with just enough power to reach all the nodes that it currently considers to be its neighbors (i.e., the nodes in $N_2(u)$). Once a node detects a change, it may need to update its set of neighbors. This is done by a *reconfiguration protocol*. Rodoplu and Meng [RM99] do not provide an explicit reconfiguration protocol. Rather, they deal with changes in network topology by running MECN periodically at every node. While this will work, it is inefficient. If a node does not detect any changes, then there is no obvious need to run MECN. We now present a reconfiguration protocol where, in a precise sense, we run SMECN only when necessary (in the sense that it is run only when not running it may result in a network that does not satisfy the minimum-energy property).

There are three types of events that trigger the reconfiguration protocol: *leave events*, *join events*, and *move events*:

- A *leave$_u$(v)* event happens when a node v that was in u's neighborhood is detected to no longer be in the neighborhood (since its beaconing message is not received). This may happen because v is faulty or dies or because it has in fact moved away.

- A $join_u(v)$ event happens when a node v is detected to be within u's neighborhood by the NDP.

- A $move_u(v, L)$ event happens when u detects that v has moved from the previous location to the current location L. (Node v's location L is relative to u's location, so the event could be due to u's own movement.)

It is straightforward to see how to update the neighbor set if u detects a single change. Suppose p^* is u's current power setting (that is, the final power setting used in the last invocation of SMECN by u); let $F^* = F(u, p^*)$ be the last region searched by u. Let A^* consist of all the nodes in F^* (that is, the set of all nodes discovered by the algorithm).

- If a single $leave_u(v)$ or a $move_u(v, L)$ is detected, let $A' = A^* - \{v\}$ if $leave_u(v)$ is detected, and let $A' = A^*$ if $move(v, L)$ is detected. Let $R'_F = \bigcap_{w \in A'}(F(u, p_{max}) - R_{u \to w})$, where the new location for v is used in the computation if $v \in A'$. (Note that R'_F is defined essentially in the same way as $R_F(u)$ in Equation (1).) If $F^* \supseteq R'_F$, then take u's updated neighbor set to be $N_{R'_F}$; otherwise, run SMECN taking $p_0 = p^*$.

- If a single $join_u(v)$ is detected, recompute the neighbor set as follows. Let $A' = A^* \cup \{v\}$. Let $R'_F = \bigcap_{w \in A'}(F(u, p_{max}) - R_{u \to w})$. Take u's updated neighbor set to be $N_{R'_F}$. Then let $p' = \min\{p : F(u, p) \supseteq \bigcap_{w \in A'}(F(u, p_{max}) - R_{u \to w})\}$.

The following proposition is almost immediate from our earlier results.

Proposition 0.5.1 *Suppose that a graph G has the minimum-energy property. If the nodes in G observe a sequence of single changes and update their edge sets as above, the resulting graph $G^*(V, E^*)$ still has the minimum-energy property for the new topology. Moreover, if $F(u, p)$ is a circular region for all p, then $E^* = E_2$.*

In general, there may be more than one change event that is detected at a given time by a node u. (For example, if u moves, then there will in general several leave and move events detected by u.) If more than one change event is detected by u, we consider the events observed in some order. If we can perform all the updates without rerunning SMECN, we do so; otherwise, we rerun SMECN starting from p^*. By rerunning SMECN, we can deal with all the changes simultaneously.

Up to now we have assumed that no topology changes are detected while SMECN itself is being run. If changes are in fact detected while SMECN is run, then it is straightforward to incorporate the update into SMECN. For example, if u detects a $join_u(v)$ event, then v is added to the set A in the algorithm, while if u detects a $leave_u(v)$ event, u is dropped from A and η is recomputed. We leave the details to the reader.

As we mentioned earlier, there is no reconfiguration protocol given in [RM99]. However, it is easy to modify the reconfiguration algorithm protocol given above for SMECN so that it works for MECN. If a $leave_u(v)$ or $move_u(v, L)$ is detected, then the same approach works (except that MECN rather than SMECN is called with $p_0 = p^*$). Similarly, if a $join_u(v)$ is detected, we update the neighbor set using the approach of MECN rather than SMECN.

Note that we have assumed a perfect MAC layer in our reconfiguration discussion. Our reconfiguration works fine even with a MAC layer that drops packets. The reason is as follows. If the

Ack message of some nodes get dropped, then the final power setting $p_a{}^*$ using an imperfect MAC layer will be bigger than the corresponding p^* using a perfect MAC layer. Since NDP beaconing with p^* reaches all nodes in $N_2(u)$, beaconing with a bigger power $p_a{}^*$ will still reach all nodes in $N_2(u)$. Eventually all the nodes in $N_2(u)$ whose Acks are lost will be detected by u through NDP beacons. Thus, the neighbor set computed using an imperfect MAC layer converges to a superset of $N_2(u)$. If the final search region is circular, then the neighbor set converges to the set $N_2(u)$.

0.6 Simulation Results and Evaluation

How can using the subnetwork computed by (S)MECN help performance? Clearly, sending messages on minimum-energy paths is more efficient than sending messages on arbitrary paths, but the algorithms are all local; that is, they do not actually find the minimum-energy path, they just construct a subnetwork in which it is guaranteed to exist.

There are actually two ways that the subnetwork constructed by (S)MECN helps. First, when sending periodic beaconing messages, it suffices for u to use power $p(u)$, the final power computed by (S)MECN. Second, the routing algorithm is restricted to using the edges $\cup_{u \in V} N(u)$. While this does not guarantee that a minimum-energy path is used, it makes it more likely that the path used is one that requires less energy consumption.

To measure the effect of focusing on energy efficiency, we compared the use of MECN and SMECN in a simulated application setting.

Both SMECN and MECN were implemented in ns-2 [Pro], using the wireless extension developed at Carnegie Mellon [Gro99]. We generated 20 random networks, each with 100 nodes. The nodes were placed uniformly at random in a rectangular region of 1500 by 1500 meters. (There has been a great deal of work on realistic placement, e.g. [ZCB96, CDZ97]. However, this work has the Internet in mind. Since the nodes in a multihop network are often best viewed as being deployed in a somewhat random fashion and move randomly, we believe that the uniform random placement assumption is reasonable in many large multihop wireless networks.)

We assume that the path-loss exponent for outdoor radio propagation models is 4. The carrier frequency is 914 MHz and transmission raw bandwidth is 2 MHz. We further assume that each node has an omni-directional antenna with 0 dB gain and is placed at 1.5 meter above the node. The receive threshold is -94 dBW, the carrier sense threshold is -108 dBW, and the capture threshold is 10 dB. These parameters simulate the 914 MHz Lucent WaveLAN DSSS radio interface. Given these parameters, the t parameter in the equation $p(u, v) = t d(u, v)^n$ in Section 0.2 is -101 dBW. In WaveLAN radio, it has been measured that radio receiver power can be quite significant [SK97] and accounts for 75% of the fixed transmission power. However, techniques for reducing the power consumption of radio electronics are fast improving. A radio typically consists of transmitter electronics, receiver electronics, and a transmit amplifier. Low-power circuit designs and signal processing reduce the power expended in the transmitter and receiver electronics. As a result, the receiver power of future radios is likely to be quite small. However, the power needed by the transmit amplifier is constrained by the rapid radio attenuation in space. Therefore, transmission power is expected to dominate receiver power in the future. Because radio-receiver power varies from radio to radio and has an impact on the computation of the minimal-energy path, we vary the receiver power c to study its effect on MECN and SMECN.

Each node in our simulation has an initial energy of 1 Joule. We would like to evaluate the effect of using SMECN on network performance. To do this, we need to simulate the network's application traffic. We used the following application scenario. All nodes periodically send UDP traffic to a sink node situated at the boundary of the network. The sink node is viewed as the master data-collection site. The application traffic is assumed to be CBR (constant bit rate); application packets are all 128 bytes. The sending rate is 0.25 packets per second. This application scenario has also been used before [HCB00]. Although this application scenario does not seem appropriate for telephone networks and the Internet (cf. [PF95, PF97]), it does seem reasonable for ad hoc networks, for example, in environment-monitoring sensor applications. In this setting, sensors periodically transmit data to a data-collection site, where the data is analyzed.

To find routes along which to send messages, we use AODV [PR99]. However, as mentioned above, we restrict AODV to finding routes that use only edges in $\cup_{u \in V} N(u)$. There are other routing protocols, such as LAR [KV98], GSPR [KK00], and DREAM [BCSW98], that take advantage of GPS hardware. We used AODV because it is readily available in our simulator and it is well studied. Since we would like to optimize with respect to the minimum-energy path metric, we modify the ns-2 AODV implementation to use the minimum-energy path metric instead of using the current shortest-path metric. Although different routing protocols may result in different network performance, we do not believe that using a different routing protocol would significantly affect the relative merits of SMECN and MECN we present here.

In order to simulate the effect of power control, we made changes to the physical layer of the ns-2 simulation code. Specifically, when simulating SMECN (resp., MECN), a node u broadcasts to its neighbors using the final transmission power $p(u)$ of its neighbor-discovery process with SMECN (resp., MECN). Similarly, a node u sends a point-to-point message to a neighbor v using the minimum power required to reach v, as determined during the neighbor-discovery process. A node's energy reserve is then subtracted by the appropriate amount for each transmission or reception.

We assumed that each node in our simulation had an initial energy of 1 Joule and then ran the simulation for 1600 simulation seconds, using both SMECN and MECN. Each data point represents an average of 20 randomly-generated networks. For the sake of fairness, identical traffic scenarios are used for both MECN and SMECN. We did not actually simulate the execution of SMECN and MECN. Rather, we assumed the neighbor set $N(u)$ and power $p(u)$ computed by (S)MECN each time it is run were given by an oracle. (Of course, it is easy to compute the neighbor set and power in the simulation, since we have a global picture of the network.) Thus, in our simulation, we did not take into account one of the benefits of SMECN over MECN, that it stops earlier in the neighbor-search process. Since a node's available energy is decreased after each packet reception or transmission, nodes in the simulation die over time. After a node dies, the network must be reconfigured. In [RM99], this is done by running MECN periodically. In our simulation, the NDP triggers the reconfiguration protocol. (When running MECN, we use the same reconfiguration protocol as the one we use for SMECN, with the appropriate modifications, as discussed in Section 0.5.) The NDP beacon for MECN and SMECN is sent with a period of 1 second and uses the power $p(u)$ computed by the neighbor-discovery process of SMECN (resp., MECN).

For simplicity, we simulated only a static network (that is, we assumed that nodes did not move), although some of the effects of mobility—that is, the triggering of the reconfiguration protocol—can already be observed with node deaths.

In this setting, we are interested in network lifetime, as measured by two metrics: (1) the number of nodes that are still alive over time and (2) the number of nodes that are still connected

to the sink. As we argued in the introduction, these are reasonable metrics. Of course, if we have more knowledge of the application, the definition of network lifetime can be made even more application-specific. For example, in a sensor network, it may be more appropriate to define network lifetime as the time that the sensors completely cover the deployment region.

We first report the experimental results when the receiver power is 0. Before describing the performance, we consider some features of the subnetworks computed by MECN and SMECN. Since the search regions will be circular with an omni-directional antenna, Theorem 0.4.4 assures us that the network used by SMECN will be a subnetwork of that used by MECN, although it does not say how much smaller the subnetwork will be. The initial network in a typical execution of the MECN and SMECN is shown in Figure 4. The average number of neighbors of MECN and SMECN in the 20 networks are initially 3.21 and 2.71 respectively. Thus, each node running MECN has roughly 19% more links than the same node running SMECN. This makes it likely that the final power setting computed will be higher for MECN than for SMECN. In fact, our experiments show that it is roughly 38% higher, so more power will be used by nodes running MECN when sending messages. Moreover, AODV is unlikely to find routes that are as energy efficient with MECN.
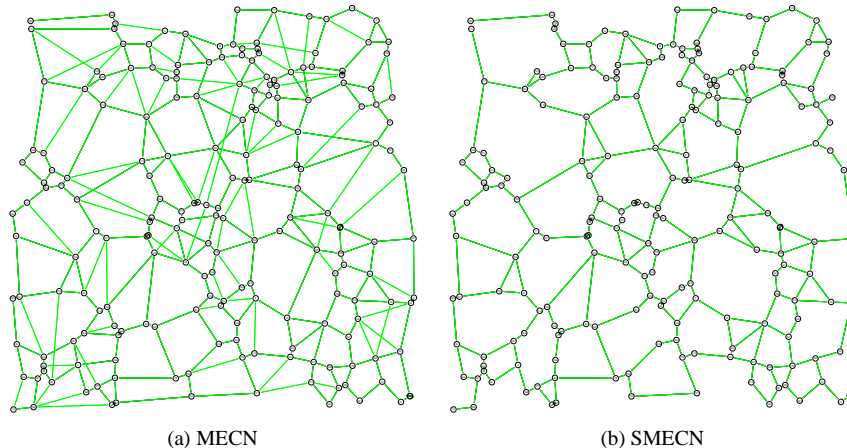

(a) MECN   (b) SMECN

Figure 4: Initial network computed by MECN and SMECN with $c = 0$mW.

As nodes die (due to running out of power), the network topology changes due to reconfiguration. Nevertheless, as shown in Figure 5, the average number of neighbors stays roughly the same over time, thanks to the reconfiguration protocol.

Turning to the network-lifetime metrics discussed above, as shown in Figure 6, SMECN performs consistently better than MECN for both. The number of nodes still alive and the number of nodes still connected to the sink decrease much more slowly in SMECN than in MECN. For example, in Figure 6(b), at time 800, 66.5% of the nodes have disconnected from the sink for MECN while only 36.4% of the nodes have disconnected from the sink for SMECN.

Finally, we collected data on average energy consumption per node at the end of the simulation, on the total number of packets delivered, and on end-to-end delay. MECN uses 21% more energy per node than SMECN. SMECN delivers more than 110% more packets than MECN by the end of the simulation, MECN's delivered packets have an average end-to-end delay that is 2% lower than
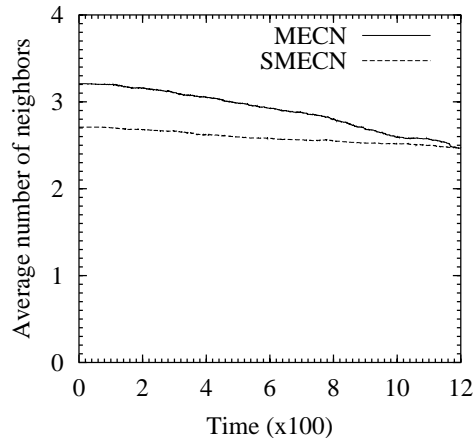
Figure 5: Average number of neighbors over time with $c = 0$mW.

SMECN. Overall, it is clear that the performance of SMECN is significantly better than MECN if the receiver power is negligible.

We now vary the receiver power c to study its impact on MECN and SMECN. As we discussed earlier in this section, the receiver power of a radio is expected to be small in the future. Hence, we set c to a small value (20mW). A typical network topology maintained by MECN and SMECN is shown in Figure 7. Comparing Figures 4 and 7, it is clear that there tend to be more direct links with $c = 20$mW than with $c = 0$mW. The average number of neighbors and broadcast power using MECN and SMECN are quite similar with $c = 20$mW. As a result, it is not surprising that the performance of the two algorithms is quite similar in this case. This is further substantiated by experimental results using the average number of neighbors metric (shown in Figure 8) and the two network-lifetime metrics (shown in Figure 9).

0.7   Summary

In this chapter, we presented a protocol SMECN that computes a network with the minimum-energy property. In the case of a circular search space, SMECN computes the set E_2 consisting of all edges that are not 2-redundant. Our protocol is localized in the sense that each node needs to know only about its local neighborhood (that is, those nodes that are a small number of hops away). In addition, we presented an energy-efficient reconfiguration protocol that maintains the minimum-energy path property despite changes in the network topology. The localized nature of our protocol makes it easy to deal with reconfiguration. We have shown by simulation that SMECN performs significantly better than MECN, while being computationally simpler.

There are a number of other localized topology-control algorithms [LHB+01, WL03, JRS03, WZ03]. CBTC [LHB+01] was the first algorithm that simultaneously achieved a variety of useful properties, such as symmetry (only symmetric links are used), sparseness (bounded degree), and good routes; CBTC achieves this without requiring each node to know its location; in particular, unlike SMECN, a GPS unit (or other means for knowing the location) is not required. However, the network CBTC computes does not have the minimum-energy property. Consequently the total
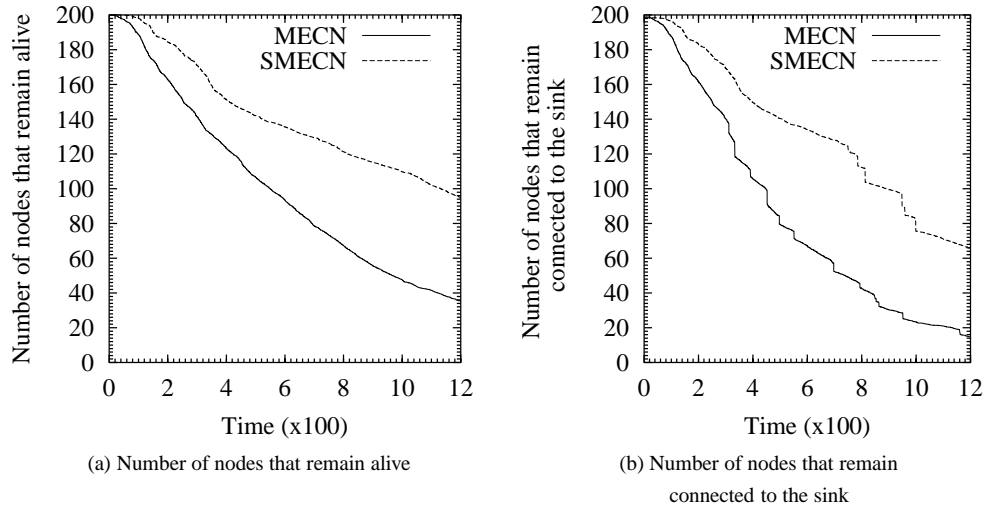
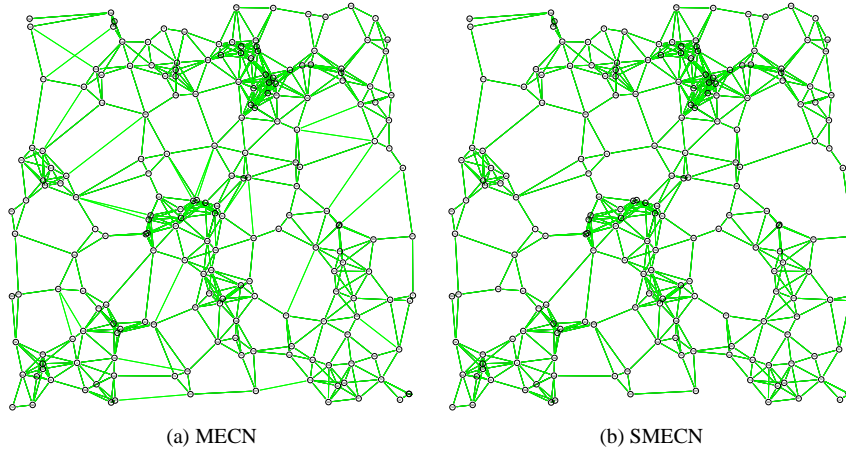Figure 6: Network lifetime for two different metrics with $c = 0$mW.

Figure 7: Initial network computed by MECN and SMECN with $c = 20$mW.

energy consumed in the network constructed by CBTC is likely to be greater than that in the network constructed by SMECN (see [LHB+01] for a detailed comparison of CBTC and SMECN). The subgraph G' of G_R constructed by Wang and Li's recent algorithm [WL03] has bounded degree and is a *k-spanner*, for a relatively small k, so that for every pair of nodes u and v, there is a path connecting them in G' whose length is no more than k times that of the shortest path from u to v in G_R. However, the network computed in [WL03] does not have the minimum-energy property and thus is unlikely to be as energy efficient as SMECN. The topology-control algorithm analyzed by Jia, Rajaraman, and Scheideler [JRS03] constructs a graph with constant degree and constant energy-stretch (the minimum-energy path for any given pair of nodes in the subnetwork is within a constant factor of the minimum energy path in the original network). XTC [WZ03] is similar in spirit to SMECN, but it does not assume any specific radio-propagation model. When running SMECN, each node u must know enough about the radio-propagation model to compute the sets $R_{u \to v}$, this knowledge is not required in XTC. Thus, XTC can be used in settings where

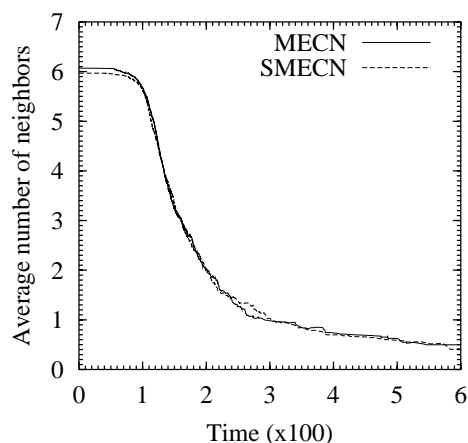Figure 8: Average number of neighbors over time with $c = 20$mW.

(a) Number of nodes that remain alive
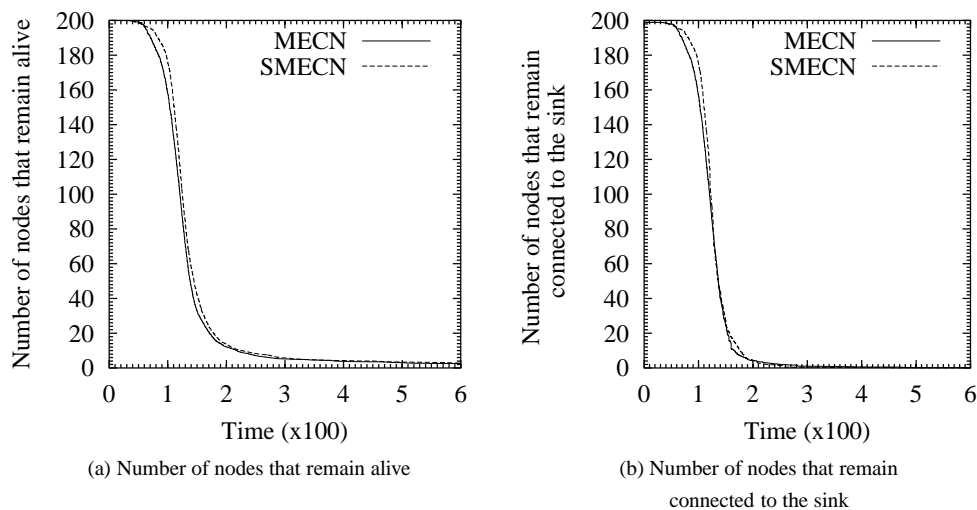
(b) Number of nodes that remain connected to the sink

Figure 9: Network lifetime for two different metrics with $c = 20$mW.

radio-propagation model is unknown. On the other hand, XTC must use maximum power in the neighbor-discovery process, so it is likely to be less energy efficient than SMECN in settings where the radio-propagation model is known.

With all these alternatives, it is clear that more work needs to be done to understand what the most appropriate algorithm is as a function of the demands of a specific application. We have focused here only on energy minimization, but there are clearly other relevant metrics as well, which further complicates the decision.

References

[BCSW98] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward. A distance routing effect algorithm for mobility (DREAM). In *Proc. Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 76–84, 1998.

[CDZ97] K. Calvert, M. Doar, and E. W. Zegura. Modeling Internet topology. *IEEE Communications Magazine*, 35(6):160–163, June 1997.

[CJBM01] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *Proc. Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 85–96, 2001.

[FRM01] K. Flautner, S. Reinhardt, and T. Mudge. Automatic performance-setting for dynamic voltage scaling. In *Proc. Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 260–271, 2001.

[Gro99] CMU Monarch Group. Wireless and mobility extensions to ns-2. http://www.monarch.cs.cmu.edu/cmu-ns.html, October 1999.

[HCB00] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless micro-sensor networks. In *Proc. IEEE Hawaii Int. Conf. on System Sciences*, pages 4–7, January 2000.

[JRS03] L. Jia, R. Rajaraman, and C. Scheideler. On local algorithms for topology control and routing in ad hoc networks. In *Proc. SPAA*, pages 220–229, 2003.

[KK00] B. Karp and H. T. Kung. Greedy perimeter stateless routing (GPSR) for wireless networks. In *Proc. Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 243–254, 2000.

[KV98] Y. B. Ko and N. H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. In *Proc. Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 66–75, 1998.

[LHB+01] L. Li, J. Y. Halpern, P. Bahl, Y. M. Wang, and R. Wattenhofer. Analysis of distributed topology control algorithms for wireless multi-hop networks. In *Proc. ACM Symposium on Principle of Distributed Computing (PODC)*, pages 264–273, 2001.

[PF95] V. Paxson and S. Floyd. Wide-area traffic: the failure of Poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, 1995.

[PF97] V. Paxson and S. Floyd. Why we don't know how to simulate the Internet. *Proc. 1997 Winter Simulation Conference*, pages 1037–1044, 1997.

[PLS01] J. Pouwelse, K. Langendoen, and H. Sips. Dynamic voltage scaling on a low-power microprocessor. In *Proc. Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 251–259, 2001.

[PR99] C. E. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing. In *Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, 1999.

[Pro] VINT Project. The UCB/LBNL/VINT network simulator-ns (Version 2). http://www.isi.edu/nsnam/ns.

[Rap96] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 1996.

[RM99] V. Rodoplu and T. H. Meng. Minimum energy mobile wireless networks. *IEEE J. Selected Areas in Communications*, 17(8):1333–1344, 1999.

[SK97] M. Stemm and R. H. Katz. Measuring and reducing energy consumption of network interfaces in hand-held devices. *IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Science, Special Issue on Mobile Computing*, E80-B(8):1125–1131, 1997.

[SN99]   M. Satyanarayanan and D. Narayanan. Multifidelity algorithms for interactive mobile applications. In *Proc. 3th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (Dial M for Mobility)*, pages 1–6, 1999.

[SR98]   S. Singh and C. S. Raghavendra. PAMAS – power aware multi-access protocol with signaling for ad hoc networks. *ACM Computer Communications Review*, pages 5–26, July 1998.

[WL03]   Y. Wang and X.Y. Li. Localized construction of bounded degree and planar spanner for wireless ad hoc networks. In *Proc. ACM DIALM-POMC Joint Workshop on Foundations of Mobile Computing*, pages 59–68, 2003.

[WZ03]   R. Wattenhofer and A. Zollinger. XTC: A practical topology control algorithm for ad-hoc networks. Technical Report 407, Computer Science Department, ETH Zurich, 2003.

[XHE01]   Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad-hoc routing. In *Proc. Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 70–84, 2001.

[ZCB96]   E. W. Zegura, K. Calvert, and S. Bhattacharjee. How to model an Internetwork. In *Proc. IEEE Infocom*, volume 2, pages 594–602, 1996.