Wireless Networks 11, 401–417, 2005

Finding Minimum Energy Disjoint Paths in Wireless Ad-Hoc Networks

ANAND SRINIVAS* and EYTAN MODIANO
Massachusetts Institute of Technology, Cambridge, MA 02139

Abstract. We develop algorithms for finding minimum energy disjoint paths in an all-wireless network, for both the node and link-disjoint cases. Our major results include a novel polynomial time algorithm that optimally solves the minimum energy 2 link-disjoint paths problem, as well as a polynomial time algorithm for the minimum energy k node-disjoint paths problem. In addition, we present efficient heuristic algorithms for both problems. Our results show that link-disjoint paths consume substantially less energy than node-disjoint paths. We also found that the incremental energy of additional link-disjoint paths is decreasing. This finding is somewhat surprising due to the fact that in general networks additional paths are typically longer than the shortest path. However, in a wireless network, additional paths can be obtained at lower energy due to the broadcast nature of the wireless medium. Finally, we discuss issues regarding distributed implementation and present distributed versions of the optimal centralized algorithms presented in the paper.

Keywords: wireless ad-hoc networks, energy efficiency, disjoint paths, multipath routing, graph theory, distributed algorithms

1. Introduction

In this paper, we address the problem of finding minimum energy disjoint paths in wireless ad-hoc networks. An ad-hoc network is an infrastructure-less network, where every node assumes the role of both host and router. Generally, nodes in an ad-hoc network are mobile as well, though in this paper we are primarily concerned with relatively static ad-hoc networks, a prevalent example of which are sensor networks.

The motivation for the minimum energy disjoint paths problem considered in this paper is two-fold. The first is the need for reliability in wireless networks. This need stems from the unpredictable nature of the wireless environment, which unlike its wired counterpart is more easily prone to link failures (e.g. due to channel fading or obstructions) and resulting path failures and data loss. Additionally, node failures (e.g. due to power loss or mobility) are also common to ad-hoc networks. Therefore from this perspective, a potential application of our work, i.e. simultaneous routing along multiple disjoint paths, can result in increased resiliency against such failures. This is especially apparent in the case of real-time data transmission, whereby if one routes along a single path, just one node (or link) failure is sufficient to cause path failure and transmission interruption. In contrast, routing along k disjoint paths makes failure much less likely, as all k disjoint paths must become disconnected in order for transmission to be interrupted. We consider both node and link-disjoint path routing in this paper. Node-disjoint paths are more resilient to failures than link-disjoint paths; as they protect against both node and link failures. However, as will be seen later in this paper, link-disjoint paths are much more energy efficient than node-disjoint paths. Moreover, in a wireless network, link-disjoint paths can protect against link failures that may result from mobility, fading, or obstructions. Hence, in many cases, individual links may fail while the node remains operational.

The second motivation is the importance of energy efficiency in wireless networks. Wireless nodes, especially sensors, tend to use small batteries for energy supply that are in many instances non-replenishable. Therefore, energy conservation is a vital factor in prolonging network lifetime. It was shown in [12] that wireless nodes often expend most of their energy in communications. As such, our objective is to minimize the aggregate transmission power (energy) used by nodes to route data along multiple paths.

Our approach to energy-efficient routing is similar to that discussed in [29] in that it differs in a key aspect from the conventional layered structure. In our treatment of routing (a network layer function), we also incorporate transmission power level variations (hence network connectivity, a physical layer function). Traditional research on routing in ad-hoc networks decouples these two layers by restricting nodes to constant transmission ranges, leading to a "static" (node mobility notwithstanding) network topology. These networks are subsequently modelled as "disk graphs", and routing is done to minimize a link-based metric (e.g. shortest hop, minimum weight). In recent years however, it has been argued that a decoupled approach, while well-suited for wired networks, does not capture many salient properties of wireless networks. This is especially true for transmission energy usage, where joint consideration of the network and physical layer issues can result in significant energy savings.

The combined problem of minimum energy disjoint path routing has not been looked at before. However, when taken as separate problems, considerable work has been done on energy efficient routing in wireless networks [1,2,5,7,10,14,22,28–30] as well as disjoint path routing in both wired and wireless networks [4,9,13,16–18,21,24,26,27]. The energy efficiency aspect of our work builds upon that of Wieselthier et al. [29] on energy-efficient broadcasting and multicasting in wireless

* Corresponding author.
E-mail: anand3@mit.edu

networks. Although they present only heuristic solutions to the problem (the problem was subsequently proven to be NP-Hard [1,5,10,14]), their work elucidates many of the fundamental aspects of energy-efficient routing in wireless networks that are used in this paper.

Other relevant work in the area of energy-efficiency in wireless networks includes work by Chen and Huang [8] on the minimum energy strongly connecting problem (i.e. there exists a path between every node pair) for packet radio networks (also proven to be NP-Hard). Along the same lines are the minimum energy topology control problems considered in [6,15,19], where the minimum energy strongly connecting problem is generalized to variants of the minimum energy k-strongly connecting problem (i.e. there exists k-node (link) disjoint paths between every node pair).

The distinction between these problems and the disjoint paths problem considered in this paper is that instead of k-disjoint paths between *every* node pair, our problem requires k disjoint paths between just two nodes—the source and destination. In the minimum energy k-strongly connecting problems, transmission ranges are assigned to all nodes such that the resulting network topology contains k disjoint paths between *every* node pair, and the aggregate transmission energy for the entire network is minimum. However, this type of optimization needlessly minimizes energy usage over nodes that may not even be transmitting, and yields sub-optimal aggregate energy usage for the specific nodes that *are* actively involved in transmission, namely the nodes belonging to the k disjoint paths between a specific source-destination pair. In this regard, finding minimum energy k disjoint paths is the more focused problem, as the energy optimization is done only over pertinent nodes. Furthermore, while most of the minimum energy k-strongly connecting problems have been proven to be NP-complete [6,8,15,19], we present *polynomial time algorithms* that optimally solve the minimum energy k node-disjoint paths problem, as well as the minimum energy 2 link-disjoint paths problem.

The problem of finding k node (link) disjoint source-destination paths in a network, is a well studied problem in graph theory. Polynomial $O(kN^2)$ running time algorithms that find minimum-weight k node (link) disjoint source-destination paths have existed for decades [4,24,26]. While these algorithms do not address the minimum energy disjoint paths problem, they serve as basic building blocks for the algorithms developed in this paper.

The remainder of the paper focuses on developing optimal polynomial running time algorithms for finding minimum energy disjoint paths. We start by introducing our network model as well as some basic concepts pertaining to wireless transmission that will be used throughout the paper. We next discuss the problem of finding k minimum energy node-disjoint source-destination paths, and follow with the link-disjoint variant. We then present a short section on alternative heuristic algorithms with lower computational complexity, but sub-optimal performance. This is followed by results, including performance comparisons between several energy-efficient algorithms. We conclude with a short section regarding distributed implementation.

2. Network model

We consider a wireless network consisting of N nodes that have omnidirectional antennas and can dynamically vary their transmission power. Specifically, each node has a maximum transmission power level $\mathcal{E}_{\max}$, and we assume that transmissions can take place at any power level in the range $[0, \mathcal{E}_{\max}]$. We assume a commonly used wireless propagation model [20] whereby the received signal power attenuates as $r^{-\alpha}$, where r is the transmission range and α is the loss constant, typically between 2 and 4 depending on the wireless medium.

Based on this model, we can clarify the concept of a wireless link, which is quite different from the traditional wired link. In wired networks the definition is clear: A "link" exists between two nodes if they can communicate via a physical medium (e.g. a wire) between them. By contrast, a wireless link is more of a "soft" concept, where it can be said that a "link" exists between two wireless nodes if the transmitting node transmits with sufficiently high power such that the "signal-to-interference-plus-noise-ratio" (SINR) at the receiving node is greater than a given threshold value θ. The threshold value θ is chosen to achieve a desired bit-error-rate for the given modulation scheme and data rate. Without loss of generality, we normalize all values such that the power required to support a wireless link at a given data rate between node i and node j is given by,

$$\mathcal{E}_{ij} = r_{ij}^{\alpha} \tag{1}$$

where r_{ij} is the distance between nodes i and j. We say that node i can "reach" node j if and only if node i transmits at a power greater than or equal to r_{ij}^{α}.

The first observation based on this model is that the network topology is entirely dependent on the range at which nodes transmit. Links can be added or removed by a node changing its transmission range. The second observation is that this model severely penalizes (from an energy standpoint) longer range transmissions. As can be seen from (1), the energy required to support such transmissions increases according to a power function. In fact, the solution to the energy efficient *single path* routing problem is based primarily on the concept that shorter hops are preferred to longer ones. The actual solution, consisting of two main steps is quite simple and is illustrated in figure 1. The first step, consisting of a basic graph transformation is one that we use quite extensively in all our algorithms, and is as follows: Given a network of N nodes and co-ordinates for each node, construct a graph $G = (V, E)$ such that $(i, j) \in E \iff r_{ij}^{\alpha} \leq \mathcal{E}_{\max}$ and $w_{ij} = r_{ij}^{\alpha}$ (where w_{ij} is the weight of link (i, j)). The new graph, that we will hereby refer to as the *energy cost graph*, provides information about all possible network topologies, in accordance with characteristics of the wireless environment and node power constraints. The second and final step is simply to run a

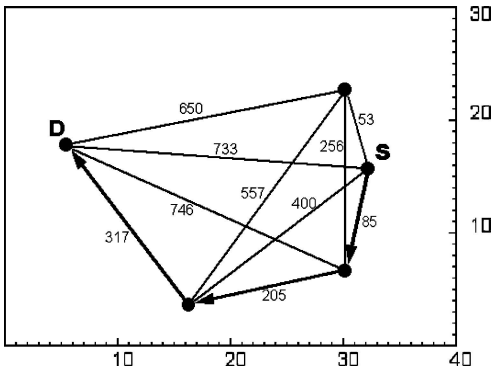


Figure 1. Example of algorithm that finds the minimum energy source-destination path (with $\alpha = 2$ and $\mathcal{E}_{\max} = 70^2$). Shown is the key step, consisting of a graph transformation that we continually refer to this paper as the "Energy Cost Graph". The minimum energy path is highlighted in bold, and has aggregate energy cost 607.

shortest path algorithm (e.g. Dijkstra, Bellman Ford) on the energy cost graph, and the resultant path is the minimum energy path.

In the case of energy efficient *multicast* and *multipath* routing, however, we see that long range transmissions can actually be used to extract energy savings. Specifically, due to the use of omnidirectional antennas, when node i transmits at a power r^α, the transmission is simultaneously received by all nodes j that are a distance less or equal to than r from node i. In figure 2, we see that for node i to multicast a message to both nodes j and k, it has three options: (a) Transmit the message to j, and have j transmit that message to k, (b) Transmit the message to j, and then re-transmit the same message to k, or (c) Transmit the message once at a range $\max(r_{ij}, r_{ik})$, thereby ensuring both j and k receive the message simultaneously. Note that without the use of omnidirectional antennas, only options (a) and (b) would be possible. However, omnidirectional antennas allow the possibility of option (c), which is clearly more energy efficient than option (b) (i.e. the transmission at range $\min(r_{ij}, r_{ik})$ in option (b) is redundant). The energy savings that option (c) provides over option (b) is referred to in [29] as the "Wireless Multicast Advantage" (WMA).

It should be noted that Wieselthier et al. [29] apply the energy saving potential of the WMA only to the minimum energy broadcast and multicast problems. In this paper, we show that the WMA can also be exploited to provide energy efficient reliability in the form of minimum energy multipath transmission.

While it is clear that exploiting the WMA for maximum energy savings is desirable, it should be noted that incorporating the WMA (i.e. allowing option (c) from figure 2) into minimum energy routing problems makes finding optimal solutions very difficult. As mentioned earlier, the majority of minimum energy topological problems [1,5,6,10,14,15,19,29] have been shown to be NP-complete. To understand in more detail the complications that the WMA adds to these problems, we must examine the relative energy cost functions with and without the WMA.

Consider an arbitrary directed subgraph of the energy cost graph P (i.e. an achievable topology). Let us first consider the case without the WMA. We can express the aggregate energy cost for this subgraph as simply the sum of all the weights on all links belonging to P. That is,

$$W(P) = \sum_{(i,j)\in P} w_{ij} \tag{2}$$

where w_{ij} is the energy cost of transmitting from node i to node j, given in (1).

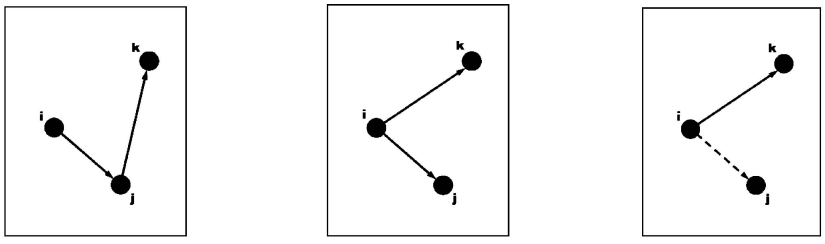
Under this cost function and in the absence of the WMA, finding k minimum energy disjoint paths between a source-destination pair corresponds to finding a minimum energy subgraph P such that P is made up of the edges belonging to these k disjoint paths. One can find such a subgraph by solving the traditional minimum weight k disjoint paths problem on the energy cost graph using standard disjoint paths algorithms [4,24,26].

With the wireless multicast advantage, the energy cost function becomes a function of a node-based metric, where due to the WMA, only maximum weight outgoing edges contribute to the aggregate energy cost. That is,

$$\mathcal{E}(P) = \sum_{x\in P} T(x) \tag{3}$$

where $T(x)$ is the transmission power of node x, i.e. $T(x) = \max\{w_{xj} : (x, j) \in P\}$.

This node-based cost function is different from the usual link-based cost functions for which traditional graph algorithms were developed. Additionally, in the context of the k disjoint paths problem, the solution found no longer corresponds exactly to the k disjoint S-D paths, P. In general, depending on the transmission powers assigned to each node,



(a) Message Forwarding (b) Separate Transmissions (c) Wireless Multicast

Figure 2. Examples of different ways to multicast a message to neighbouring nodes in a wireless network. The dashed edge in 2(c) indicates an edge obtained for "free" due to the wireless multicast advantage.

our solution is actually a subgraph of the energy cost graph that *contains* P, where due to the WMA, various edges in the subgraph may not contribute any additional energy cost. It is this property that we exploit to lower the overall energy, $\mathcal{E}(P)$.

For the remainder of the paper, we refer to the quantity in (2) as aggregate *weight*, and the quantity in (3) as aggregate *energy*. The distinction between weight and energy is an important one, as it underscores a major difference between general networks and wireless networks. In graph terms, weight is an edge-based metric, that assumes that the addition of any edge (i, j) into a solution topology P contributes w_{ij} to $W(P)$, regardless of its endpoint nodes i and j. Calculating $W(P)$ is therefore tantamount to simply summing the weights of all edges in P. Energy however, is a node-based metric, in that the cost contributed to $\mathcal{E}(P)$ by the addition of an edge (i, j) into P, depends both on the transmitting node i and the weights of its outgoing edges already in P. This is due to the WMA, whereby nodes need only expend energy corresponding to the maximum weight outgoing edge (i.e. the transmission power). All other edges are obtained for "free".

3. Minimum energy node-disjoint paths

The minimum energy k node-disjoint S-D paths problem can be stated as follows: Given an *Energy Cost Graph* $G = (V, E)$ with weights w_{ij} and source-destination pair $S, D \in V$, find a set of k node-disjoint S-D paths, $P = \{p_1, p_2, \ldots, p_k\}$, such that $\mathcal{E}(P)$ is minimized.

An example of a k node-disjoint path topology is shown in figure 3. Observe that since the k paths are node-disjoint, all nodes in P other than S and D have exactly one outgoing edge and S has exactly k outgoing edges. Hence it is clear that the source node is the only node at which the wireless multicast advantage (WMA) can be exploited for energy savings. Thus the energy cost equation from (3) can be re-written in the following manner:

$$\begin{aligned}\mathcal{E}(P) &= T(S) + \sum_{x \in P, x \neq S} T(x) \\ &= T(S) + \sum_{(i,j) \in P, i \neq S} w_{ij} \end{aligned} \tag{4}$$

where $T(S)$ is the transmission power of the source node.

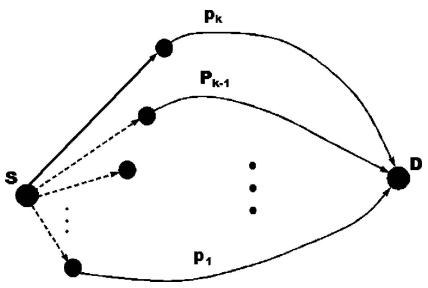


Figure 3. Example of k node-disjoint source-destination paths. Dashed lines indicate edges achieved for "free".

The form of this equation exposes the fact that this problem is closely related to the minimum *weight* k-node-disjoint paths problem discussed earlier. In particular, let us set the source transmission power, $T(S)$, to be a constant value, $T_S < \mathcal{E}_{\max}$. This is reflected in the energy cost graph by removing all edges between the source and nodes that cannot be "reached" with a transmission power of T_S. Moreover, since we have already expended the transmission energy cost of T_S, the WMA indicates that all edges between the source and nodes that can be "reached" contribute no additional energy cost. We reflect this change in the energy cost graph by setting the weights of these edges to 0.

Once we apply these changes, it is clear that *given* a source transmission power $T(S) = T_S$, the problem of finding k node-disjoint paths that minimize (4) amounts to running a minimum weight k node-disjoint paths algorithm (e.g. Suurballe's algorithm [24]) on the modified energy cost graph. What remains is to determine the optimal value of $T(S)$, that results in the overall minimum energy solution. The STPS algorithm presented below is an algorithm that searches over all relevant values of $T(S)$, evaluating (4) at each step. Finally, the overall minimum energy solution is extracted, which are the minimum energy k node-disjoint paths.

3.1. Source transmit power selection (STPS) algorithm

The STPS algorithm takes as input an energy cost graph $G = (V, E)$, the number of desired node-disjoint paths k, and a source-destination pair, $S, D \in V$. Moreover, assume S has M outgoing edges[1] $m_1, m_2, \ldots, m_M$, ordered such that $w(m_i) > w(m_j) \Leftrightarrow i > j$, where $w(m_i)$ is the weight of the edge m_i. Its output is the set of k minimum energy node-disjoint paths, $P_{\min}$.

Initialize: Let $T_i(S)$ represent the current iteration source transmission power, corresponding to the i closest nodes "reached" by S. Initialize $i = k$ and thus $T_i(S) = w(m_k)$. Note that starting with $i < k$ would be fruitless, as the existence of k node-disjoint paths requires at least k outgoing edges from the source. Finally, let $\mathcal{E}_{\min}$ represent the overall energy cost of the k minimum energy node-disjoint paths, $P_{\min}$. Initialize $\mathcal{E}_{\min}$ to ∞.

Step 1: Construct a new graph G_i, where G_i is a modified version of the energy cost graph that reflects all possible network topologies given the current iteration source transmission power, $T_i(S)$. Accordingly, let G_i be equal to G, except remove the edges $m_{i+1}, m_{i+2}, \ldots, m_M$, and set the weights of the edges $m_1, m_2, \ldots, m_i$ equal to 0.

Step 2: Run a minimum weight k node-disjoint S-D paths algorithm on G_i. Let P_i and $W(P_i)$ represent the solution k paths found by the algorithm and their aggregate weight, respectively. If given the current $T_i(S)$, k-disjoint paths cannot be found by the minimum-weight algorithm, then set $W(P_i) = \infty$ and continue.

[1] $M \leq N - 1$, with equality if and only if $\mathcal{E}_{\max}$ is large enough such that S can directly reach every node in the graph.

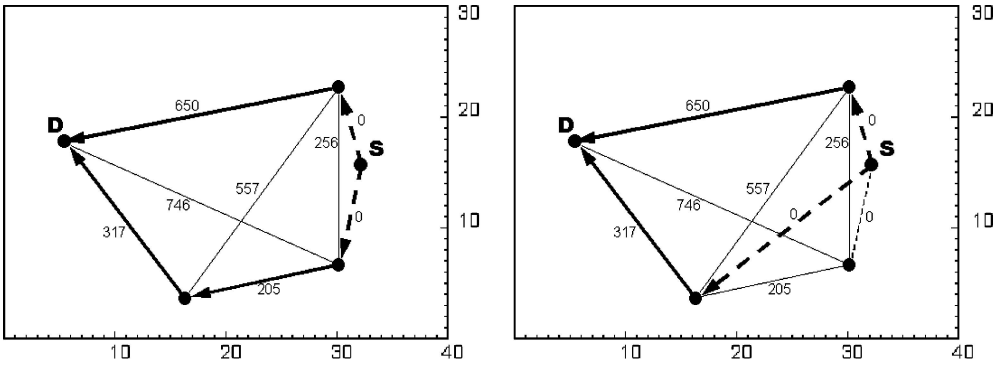
Step 3: Evaluate the following condition: If $W(P_i) + T_i(S) < \mathcal{E}_{\min}$, then set $\mathcal{E}_{\min} = W(P_i) + T_i(S)$ and $P_{\min} = P_i$. This ensures that $\mathcal{E}_{\min}$ and $P_{\min}$ always correspond to the overall minimum energy k node-disjoint paths.

Step 4: Increment $i = i + 1$ and correspondingly increase the source transmission power, i.e. $T_{i+1}(S) = w(m_{i+1})$. Repeat steps 1–4 until $i > M$, at which point all relevant $T(S)$ will have been considered, and the overall minimum energy k node-disjoint paths, $P_{\min}$ determined.

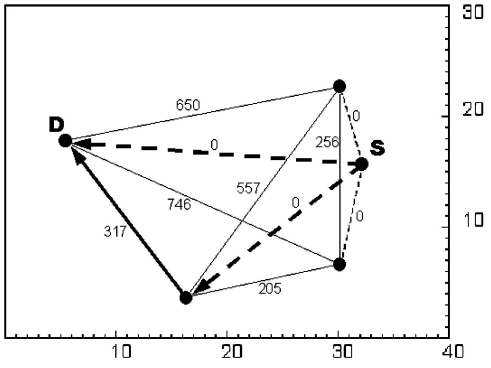
The proof that the STPS algorithm actually finds an optimal set of minimum energy k node-disjoint paths follows from (4), as we basically perform a brute force search over all relevant $T(S)$. Clearly the only relevant values of $T(S)$ are ones that can be used to reach its neighbouring nodes, i.e. the weights of its outgoing edges in G.

A visual example of the operation of the algorithm with $k = 2$, run on the energy cost graph of figure 1, is shown in figure 4. The first iteration of the algorithm is illustrated in figure 4(a), in which the modified energy cost graph, reflective of the initial source transmission power $T_2(S) = 85$ is shown. Also shown are the node-disjoint paths found by the minimum weight algorithm *given* the particular value of $T(S)$. In figure 4(c) we see the minimum energy node-disjoint paths are found when we set $T(S)$ such that we reach the destination in one hop. This is an excellent example of using long range transmissions (i.e. WMA) to extract energy savings, as even though we pay a heavy energy cost (i.e. 733) to achieve the direct link between the source and destination, we realize that by doing so we obtain the high cost (i.e. 400) first link on the second path for "free".

Of course setting $T(S)$ to its maximum value does not always work, and it is important to clarify why we must indeed iterate over all relevant values of $T(S)$. The key factor here is the tradeoff between the current value of $T(S)$ and the aggregate *weight* of the paths found by the minimum weight algorithm in step 2 of the STPS algorithm (i.e. *given* the current $T(S)$ value). Consider two different values of $T(S)$, T_a and T_b, such that $T_a < T_b$. We know that given $T(S) = T_b$, we can always find the exact same paths that we could given $T(S) = T_a$, as edges are *added* to the modified energy cost graph when $T(S)$ is increased. Moreover, since given $T(S) = T_b$ the corresponding energy cost graph has a "richer" topology than if $T(S) = T_a$, we may even be able to find "better" (i.e. lower aggregate weight) paths. This may lead to the false reasoning that increasing $T(S)$ can only decrease the overall aggregate *energy*. However, higher values of $T(S)$ *can* result in higher energy consumption. An example where increasing $T(S)$ does not lower the aggregate energy can be seen in figures 4(a) and (b), where the overall energy of the paths found with $T(S) = 400$ is actually *higher* than those found with $T(S) = 85$ (i.e 1367 vs. 1257). This is despite the fact that the aggregate weight of



(a) $i = 2, T_2(S) = 85, \mathcal{E}(P_2) = 85 + 1172 = 1257$ (b) $i = 3, T_3(S) = 400, \mathcal{E}(P_3) = 400 + 967 = 1367$



(c) $i = 4, T_4(S) = 733, \mathcal{E}(P_4) = 733 + 317 = 1050$

Figure 4. Operation of STPS algorithm when run on the energy cost graph of figure 1. In this example, the minimum energy node-disjoint S-D paths are those in figure 4(c).

the paths found with $T(S) = 400$ is *lower* than those found with $T(S) = 85$ (i.e. 967 vs. 1172).

We conclude this section by addressing the issue of complexity. The worst case complexity of the STPS algorithm, as presented above, is $O(kN^3)$. This is because the algorithm iterates $M - k + 1$ times, where $M = N - 1$ in the worst case (i.e. $\mathcal{E}_{\max}$ is sufficiently high such that the source can reach all nodes in the graph in one hop), and in each iteration we run a minimum weight node-disjoint paths algorithm whose complexity is $O(kN^2)$. The result is an overall worst case complexity of $O(kN^3)$.

It should be noted that certain modifications can be made to improve the running time of the STPS algorithm. For example, a straightforward improvement would be to initialize $i = M$, and work our way down to $i = k$. By doing this, in step 3, if at any point k node-disjoint paths did not exist given the current source transmission range, we could immediately terminate the algorithm and declare the optimal solution as the current $P_{\min}$. A more involved modification yields an elegant 2 minimum energy node-disjoint paths algorithm that is, on average, faster than the STPS algorithm. We refer to this as the Enhanced Source Transmit Power Select (E-STPS) algorithm, and describe it in the Appendix. We have not found any modification, however, that improves the worst-case running time below $O(kN^3)$.

4. Minimum energy link-disjoint paths

The minimum energy k link-disjoint S-D paths problem can be stated similarly to the minimum energy k node-disjoint S-D paths problem, as follows: Given an *Energy Cost Graph* $G = (V, E)$ with weights w_{ij} and source-destination pair $S, D \in V$, find a set of k link-disjoint S-D paths, $P = \{p_1, p_2, \ldots, p_k\}$, such that $\mathcal{E}(P)$ is minimized.

We start by noting that finding minimum energy link-disjoint paths is a much harder problem than the node-disjoint variant. The main reason for this is the difference in complexity of the aggregate energy cost functions, which in both cases is given by (3). However, recall that in the node-disjoint case, as we saw in (4), $T(x) = max\{w_{xj} : (x, j) \in P\}$ simplified to $T(x) = w_{xj}$ for all nodes x other then the source node. This reduced the minimum energy node-disjoint paths problem to one of finding the optimal source transmission power, $T(S)$.

In the case of link-disjoint paths however, any node in the resultant topology P can have up to k outgoing edges. The implication of this is that energy savings can be realized at potentially many nodes (i.e. any node with multiple outgoing edges), and we therefore need to find the optimal transmission power, $T(x)$, for *every* node x in P. Clearly, we cannot use the approach of searching over all relevant transmission powers for every node, as this type of brute force search would be exponentially complex and thus intractable.

We therefore need an alternative approach to finding k minimum energy link-disjoint paths in polynomial time. To this end, we start with $k = 2$, and try and simplify the problem by exploiting properties of a *pair* of link-disjoint paths,

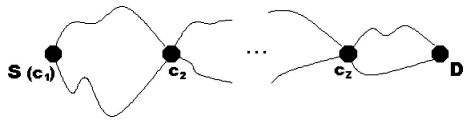


Figure 5. Example of a pair of link-disjoint paths expressed as the union of a set of node-disjoint path pairs.

$P = \{p_1, p_2\}$. We first define the notion of a "common node", which is a node that is "common" to both paths and therefore has exactly 2 outgoing edges. Next, we define the ordered set of common nodes, $C(P) = \{c_1, c_2, \ldots, c_Z\}$ as follows: If we trace along either of the paths in P, starting from S towards D, the first common node (after S, which we define as c_1) encountered is c_2, the next is c_3, and so forth. As a matter of semantics, the destination node is not considered a common node per se, but for notational convenience is defined as c_{Z+1}. This is because even though it belongs to both paths, it has no outgoing edges. This means that it does not transmit, and can be ignored in our energy calculations. It is important to note that it is only at the common nodes where we can exploit the WMA to realize energy savings. If we apply the common node analogy to the node-disjoint problem, clearly the source node is the only "common node", i.e. $C(P) = \{S\}$.

We can now make the critical observation that *any set of two link-disjoint source-destination paths can be represented as the union of node-disjoint path pairs between successive common nodes*. This is shown in figure 5, where we see that the pair of link-disjoint paths P can be broken into the corresponding set of 2 node-disjoint paths between successive common nodes. We use the notation $\gamma_P^{i,j}$ to represent the pair of node-disjoint paths between node i and node j belonging to P. We can thus re-express P, i.e. $P = \bigcup_{i=1}^{Z} \gamma_P^{c_i, c_{i+1}}$, where $c_1, c_2 \ldots, c_Z$ are the common nodes. Moreover, we can also re-express the aggregate energy cost of P, as

$$\mathcal{E}(P) = \sum_{i=1}^{Z} \mathcal{E}\left(\gamma_P^{c_i, c_{i+1}}\right) \tag{5}$$

These observations, coupled with the following theorem make up what we refer to as the *Common Node Decomposition*, and it forms the basis of our solution to finding the pair of minimum energy link-disjoint S-D paths.

Theorem 1. Let $P^* = \{p_1^*, p_2^*\}$ be a pair of optimal minimum energy link-disjoint S-D paths with corresponding set of common nodes, $C(P^*) = \{c_1^*, c_2^*, \ldots, c_Z^*\}$. Then, $\forall i, i = 1, 2, \ldots, Z$, the $\gamma_{P^*}^{c_i^*, c_{i+1}^*}$ node-disjoint path pairs are minimum energy node-disjoint path pairs.

Proof. Consider a pair of successive common nodes in P^*, c_i^* and c_{i+1}^*. Suppose $\gamma_{P^*}^{c_i^*, c_{i+1}^*}$ is not a pair of minimum energy node-disjoint paths, i.e. there exists $\gamma_{P'}^{c_i^*, c_{i+1}^*}$, such that $\mathcal{E}(\gamma_{P'}^{c_i^*, c_{i+1}^*}) < \mathcal{E}(\gamma_{P^*}^{c_i^*, c_{i+1}^*})$. Hence, replacing $\gamma_{P^*}^{c_i^*, c_{i+1}^*}$ with $\gamma_{P'}^{c_i^*, c_{i+1}^*}$ will reduce the aggregate energy cost of the paths.

In order to complete our proof, we must also show that the new S-D paths that result from replacing $\gamma_{P^*}^{c_i^*, c_{i+1}^*}$ with $\gamma_{P'}^{c_i^*, c_{i+1}^*}$ are also link-disjoint. This subtlety arises because the pair of

node-disjoint paths, $\gamma_{P'}^{c_i^*,c_{i+1}^*}$ could potentially intersect with some of the other node-disjoint path pairs $\gamma_{P^*}^{c_j^*,c_{j+1}^*}$, $j \neq i$ comprising P^*. However, we show that if such an intersection took place, then a cycle would form that could be removed to further reduce the aggregate energy cost of the link-disjoint S-D paths. This contradicts the assertion that P^* are minimum energy link-disjoint S-D paths; and the Theorem is shown.

To see this, suppose such an intersection exists. That is, a node $\mathbf{w}$ exists such that $\mathbf{w} \in \gamma_{P'}^{c_i^*,c_{i+1}^*}$ and $\mathbf{w} \in \gamma_{P^*}^{c_j^*,c_{j+1}^*}$, $j \neq i$. Let P' be the new set of paths that result from replacing $\gamma_{P^*}^{c_i^*,c_{i+1}^*}$ with $\gamma_{P'}^{c_i^*,c_{i+1}^*}$. Starting from the source, we can trace two paths in P' towards the destination; p_1' and p_2'. Without loss of generality, let p_1' take the form, $p_1' = \{S, \ldots, c_i, \ldots, \mathbf{w}, \ldots, c_{i+1}, \ldots, c_j, \ldots, \mathbf{w}, \ldots, c_{j+1}, \ldots, D\}$, and p_2' the remaining edges in P'. We first note that both p_1' and p_2' are S-D paths, except that p_1' contains a cycle starting from node $\mathbf{w}$ that can be removed. The result is the new pair of link-disjoint S-D paths with the cycle in p_1' removed. Since the energy cost of the cycle must be strictly positive, its removal further reduces the cost of the S-D path pair. Here it should be noted that the energy cost of the cycle could not have been masked by the WMA, since the WMA only applies to links outgoing from a common node, while the cycle must contain at least one node that is not a common node. Similarly, it can be shown that if multiple intersections occur between $\gamma_{P'}^{c_i^*,c_{i+1}^*}$ and a subset of the $\gamma_{P^*}^{c_j^*,c_{j+1}^*}$, they form multiple cycles that can similarly be eliminated by removing each cycle individually. □

The common node decomposition reduces the minimum energy link-disjoint path pair problem in the following way. Instead of looking for optimal transmission powers for every node, the problem is reduced to finding the optimal ordered set of common nodes and minimum energy node-disjoint paths between them. We know that there are $N(N-1)$ distinct node pairs in the graph and from our discussion in the previous section, we know how to find minimum energy node-disjoint paths between them in polynomial time. Therefore, all that remains is to find the optimal ordered set among these $N(N-1)$ minimum energy *node*-disjoint path pairs, whose *union* results in the pair of minimum energy *link*-disjoint source-destination paths. This can be accomplished by a brute force search over all possible combinations of minimum energy node-disjoint path pairs. However, such a search would be computationally difficult, as there are $O(2^{N^2})$ such combinations.

Fortunately, Theorem 1 and (5) allow us to express the aggregate energy cost of a pair of *minimum energy* link-disjoint S-D paths, as the sum of the energy costs of minimum energy node-disjoint paths between the common nodes. Hence we can efficiently find the optimal common node decomposition, using a graph-based approach as follows. We define a new graph where the weight of an edge (i, j) corresponds to the *energy cost of the minimum energy node-disjoint path pair between nodes i and j*. We then note that because (5) expresses the aggregate energy of a pair of link-disjoint S-D paths in the new graph as an additive link-based metric, the optimal common node decomposition can be found by running a simple shortest path algorithm (e.g. Dijkstra) on the new graph. Note that an edge must be defined for every node pair (i, j), as any node in the graph could potentially belong to the optimal common node decomposition. Thus finding the shortest path from S to D in this new graph corresponds to finding the set of minimum energy node-disjoint path pairs whose union result in a pair of minimum energy link-disjoint S-D paths. Additionally, the nodes belonging to the shortest S-D path, $H = \{S, h_2, \ldots, h_Z, D\}$, *are* the ordered set of optimal common nodes. The last step, constructing the optimal link-disjoint solution P, is done by concatenating the appropriate $h_i - h_{i+1}$ node-disjoint path pairs. The algorithm is detailed below.

4.1. Optimal common node decomposition (OCND) algorithm

The OCND algorithm takes as input an "Energy Cost Graph" $G = (V, E)$, and a source-destination pair, $S, D \in V$. Its output are the minimum energy 2 link-disjoint S-D paths, $P = \{p_1, p_2\}$.

Step 1: Construct a graph $G^* = (V, E^*)$ such the weight of every edge $(i, j) \in E^*$, $i \neq j$ is equal to $\mathcal{E}(\gamma_{\text{opt}}^{i,j})$, i.e. the aggregate energy cost of the minimum energy 2 node-disjoint i-j paths. This amounts to running the STPS algorithm for every distinct node pair in G.

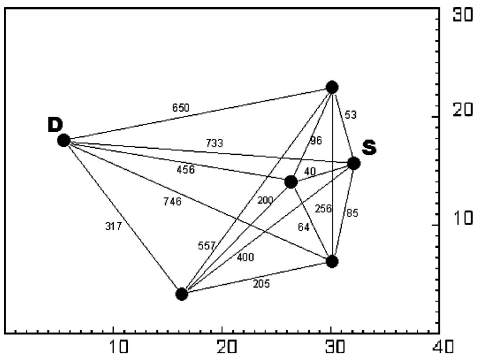
Step 2: Run a minimum weight (shortest) S-D path algorithm (e.g. Dijkstra) on G^*, resulting in a minimum weight path $H = \{h_1, h_2, \ldots, h_Z, D\}$, where $h_1 = S$. The set H represents the ordered set of common nodes that make up the optimal common node decomposition.

Step 3: Construct the solution minimum energy link-disjoint source-destination paths, $P = \{p_1, p_2\}$, by concatenating the minimum energy node-disjoint h_i-h_{i+1} path pairs, $i = 1, 2, \ldots, Z$.

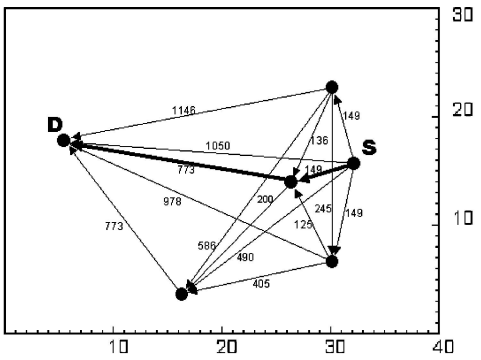
The optimality of the OCND algorithm follows directly from Theorem 1 and the following lemma.

Lemma 1. The minimum energy node-disjoint h_i-h_{i+1} path pairs picked by the OCND algorithm never intersect.

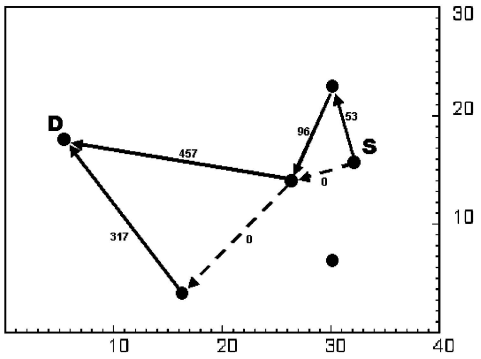
Proof. Suppose 2 of the path pairs picked by the OCND algorithm intersected, e.g. a node $\mathbf{w}$ exists such that $\mathbf{w} \in \gamma_P^{h_i,h_{i+1}}$ and $\mathbf{w} \in \gamma_P^{h_j,h_{j+1}}$, $j \neq i$. However, as we saw in the proof of Theorem 1, this intersection causes a cycle to appear, which can be removed. Specifically, upon removal of the cycle, we end up with link-disjoint S-D paths $P' = \{p_1', p_2'\}$, where $p_1' = \{S, \ldots, h_i, \ldots, h_{j+1}, \ldots, D\}$ and $p_2' = \{S, \ldots, h_i, \ldots, h_{i+1}, \ldots, h_j, \ldots, h_{j+1}, \ldots, D\}$, and where in P', h_i and h_{j+1} are now successive common nodes. Moreover, we have that $\mathcal{E}(P') < \mathcal{E}(P)$ and $\mathcal{E}(\gamma_{P'}^{h_i,h_{j+1}}) < \sum_{m=i}^{j} \mathcal{E}(\gamma_P^{h_m,h_{m+1}})$. However if this was the case, then to get from node h_i to node h_{j+1} in G^* (step 2), the shortest path algorithm would have picked the lower cost path, i.e. the edge (h_i, h_{j+1}) instead of the edges



(a) Original Network and corresponding "Energy Cost Graph"



(b) Shortest S-D path in Transformed Graph (i.e. Optimal Common Node Decomposition). The weight of each edge corresponds to the energy cost of the minimum energy node-disjoint path pair between its two end points.



(c) Minimum Energy link-disjoint S-D Path Pair. $\mathscr{E}(P) = 149 + 773 = 922$ for these paths

Figure 6. Operation of OCND algorithm, with $\alpha = 2$ and $\mathcal{E}_{\max} = 70^2$.

(h_i, h_{i+1}), $(h_{i+1}, h_{i+2}), \ldots, (h_j, h_{j+1})$ that caused the intersection in the first place. Thus the OCND algorithm always chooses "disjoint" minimum energy node-disjoint path pairs, and the Lemma is shown. □

An example of its operation, run on the energy cost graph of figure 6(a) is illustrated in figure 6. The construction of G^* in the first step of the algorithm is illustrated in figure 6(b), along with the shortest S-D path in G^* from step 2. It is important to note that not all edges in G^* are shown. This was done for legibility, but in general G^* is a complete graph, where edges are defined in both directions. Finally, figure 6(c) shows the solution minimum energy link-disjoint paths, whose aggregate energy in this case is 922.

We next address the issue of complexity of the OCND algorithm. Step 1 is clearly the most complex step, as we must run the STPS algorithm $N(N-1)$ times. This results in an overall complexity of $O(N^5)$ which is high, but a vast improvement over an exponentially complex brute force search approach. Through a slightly more complicated implementation of step 1, we can actually lower the complexity of the OCND algorithm to $O(N^4)$; We present this implementation as the Enhanced Optimal Common Node Decomposition (E-OCND) algorithm, in the appendix.

Note that the notion of common node decomposition cannot be easily extended to $k > 2$ disjoint paths. This is because when $k > 2$ a node may be common to a *subset* (as opposed to exactly 2, for $k = 2$) of the paths. The result of this is that in general, k link-disjoint paths cannot be decomposed into a concatenation of k node-disjoint paths. We were not able to find an optimal polynomial time algorithm for the minimum energy link-disjoint problem for $k > 2$, however in the following section we present efficient heuristic algorithms that find energy-efficient link-disjoint paths for general k.

5. Lower complexity heuristics

Although both the STPS and OCND algorithms find minimum energy solutions in polynomial time, their respective running times of $O(kN^3)$ and $O(N^5)$ are still quite high. Moreover, the OCND algorithm only finds a *pair* of minimum energy link-disjoint paths, which is not sufficient when a greater number of link-disjoint paths are required. To address these concerns, we present three sub-optimal heuristic algorithms that find

energy-efficient disjoint paths in $O(kN^2)$ running time. All three algorithms have extremely similar node and link-disjoint versions, but for brevity only the link-disjoint versions are presented.

5.1. Heuristic 1: Naive Dijkstra algorithm

This algorithm is a very basic algorithm that finds link-disjoint paths. It entails running Dijkstra's shortest path algorithm k times on the energy cost graph G, where after each run, links belonging to the last path found are removed, ensuring link-disjointness among the k paths. As a final step, we remove redundant transmissions at every common node of the paths found by applying the WMA (i.e. nodes with multiple outgoing edges need only expend transmission power once, corresponding to the weight of the maximum weighted outgoing edge). Note that the algorithm does not take into account the benefits of the WMA in searching for the paths. Although, after finding the disjoint paths the WMA is applied to reduce the energy cost of the paths.

5.2. Heuristic 2: Link-Disjoint Min-Weight (LD-MW) algorithm

This algorithm uses a minimum weight k link-disjoint S-D paths algorithm on the energy cost graph G, to find k link-disjoint paths, $P = \{p_1, p_2, \ldots, p_k\}$. The final step is the removal of redundant transmissions at every common node belonging to the paths. What is key to note here is that the LD-MW algorithm (similar to the Naive Dijkstra algorithm) *does not* consider the WMA when finding paths. However, once the paths are found, they are post-processed and any incidental WMA benefit is realized. An interesting property of both node and link-disjoint versions of this heuristic is that they produce solutions whose resultant overall energy is k-approximate to the optimal minimum energy solution; the proof for this is given in the appendix. As an example of its operation, when run on the Energy Cost Graph of figure 6(a), the pair of disjoint paths found by the LD-MW algorithm are shown in figure 7.

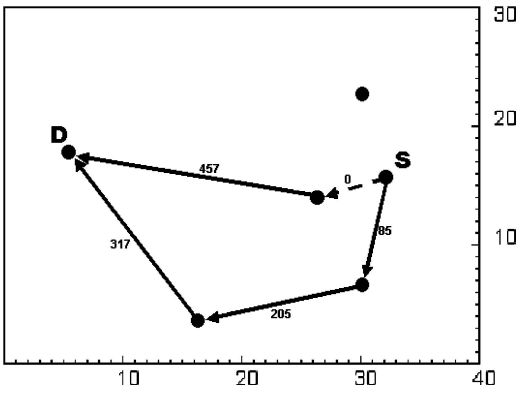


Figure 7. Solution paths found by LD-MW algorithm run on energy cost graph of figure 6(a). $\mathcal{E}(P) = 1063$ for these paths.

Note the difference in energy cost with respect to the optimal solution in figure 6(c), i.e. 1063 vs. 922.

5.3. Heuristic 3: WMA enhanced link-disjoint shortest path (LD-ESP) algorithm

The LD-ESP algorithm is an enhancement to the Naive Dijkstra algorithm discussed above. The enhancement is as follows. After each iteration i, for every node v along the last path found, p_i, modify its outgoing edges to all neighbours j, (v, j), as follows: $w^i_{vj} = \max\{0, \min\{w^{i-1}_{vj}, w^0_{vj} - w^0_{vk}\}\}$, where w^i_{vj} refers to the weight of edge (v, j) after the i^{th} iteration, w^0_{vj} refers to the original weight (i.e. from the original energy cost graph) of the edge (v, j), and (v, k) is the outgoing edge from node v which belongs to p_i.

This enhancement allows the algorithm to incorporate the WMA after choosing a path in the current iteration. It does this by modifying the weights on the outgoing edges from the nodes along the last path found, such that they represent the new *incremental* power (i.e. $w^0_{vj} - w^0_{vk}$) needed to add those edges in a future iteration. The solution paths found by the LD-ESP algorithm when run on the energy cost graph of figure 6(a) are identical to those found by the OCND algorithm, shown in figure 6(c). However, while in this specific example the LD-ESP found the optimal solution, in general the LD-ESP does not find optimal solutions. In the specific case of $k = 2$, it can be shown that if the path selected in the first iteration belongs to the optimal solution, then the LD-ESP algorithm is guaranteed to find the optimal solution. However, if the initial path is not in the optimal solution, the LD-ESP (similar to the Naive Dijkstra) algorithm can have *arbitrarily bad* performance.

6. Results

In this section we compare the performance of the algorithms discussed in this paper. We focus on three main aspects: (a) The performance difference between the optimal algorithms and the sub-optimal heuristics, (b) The energy cost of multipath routing along link-disjoint paths vs. node-disjoint paths, and (c) The incremental energy cost of adding paths (i.e. additional reliability).

We simulate networks of a varying number of nodes, N, placed randomly within a 50×50 plane. We use $\alpha = 2$ and $\mathcal{E}_{\max} = 100^2$. Note that setting $\mathcal{E}_{\max}$ in this way results in every node being able to reach every other node in one hop (if it transmits at a sufficiently high power level). Finally, for each plot shown, the results are averaged over 100 randomly generated network instances.

We begin with the evaluation of the various node-disjoint algorithms (we refer to the node-disjoint versions of LD-MW and LD-ESP as ND-MW and ND-ESP respectively). Figure 8 shows the average energy cost of the various algorithms vs. the number of nodes in the graph. We first observe that both incarnations of the dijkstra algorithm (i.e. node-disjoint naive dijkstra and ND-ESP) are the least energy efficient. We expect

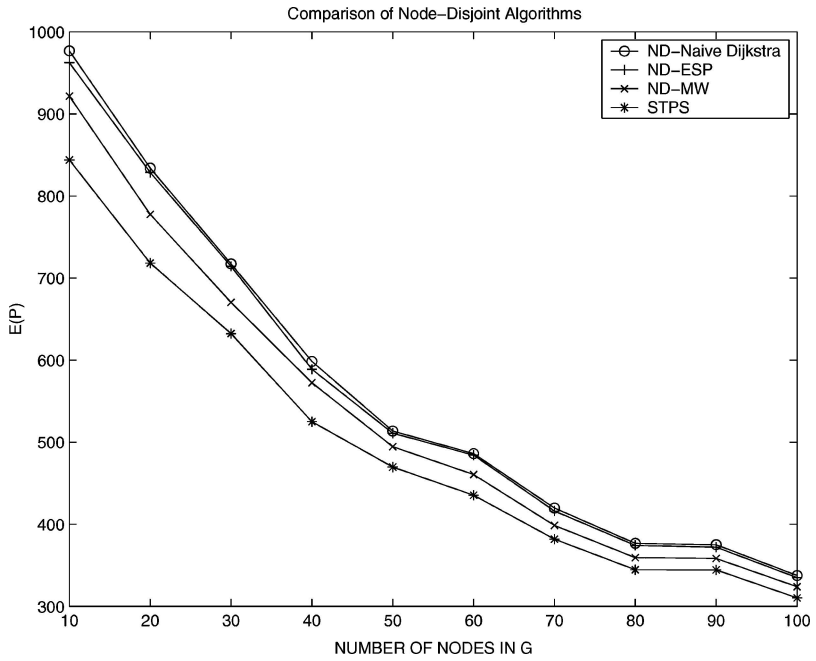


Figure 8. Comparison between energy-efficient node-disjoint algorithms.

bad performance from the naive dijkstra algorithm because it does not attempt to capture the wireless multicast advantage in its search for disjoint paths. The ND-ESP algorithm however, takes into account the WMA at the source node, but like the naive dijkstra algorithm does not minimize the aggregate paths weight. Therefore in the node-disjoint case, even though the ND-ESP may achieve maximum energy savings at the source node, we see that in general this energy savings is far lower than the additional energy expended due to the (weight) sub-optimal paths it finds. Finally, we see that the performance gain of the optimal STPS algorithm over the ND-MW algorithm is highest for low values of N. This is because in "sparse" (in terms of number of nodes per unit area) graphs, it is more likely that every node, including the source, will be forced to take longer range hops, resulting in a greater overall expenditure of energy (this can be seen in figure 8 as $\mathcal{E}(P)$ for all algorithms decreases with increasing N). The consequence of this is that for such graphs, the STPS algorithm can maximally exploit energy savings at both the source node (WMA) as well as along the paths (weight).

We next explore the performance of the link-disjoint algorithms, shown in figure 9. For the same reasons as in the node-disjoint case, the link-disjoint version of the naive dijkstra algorithm has the worst performance. However, in contrast to the node-disjoint case, the LD-ESP algorithm actually *outperforms* the LD-MW algorithm. The reason for this is that with link-disjoint paths, there are more opportunities for the LD-ESP algorithm to exploit the WMA (i.e. at the common nodes). Therefore, while in the node-disjoint case the energy saved at the source node was less than the additional energy spent on weight sub-optimal paths, we see that the opposite is true for link-disjoint paths. Moreover, we see that with increasing N, the gap between the LD-ESP and LD-MW algorithms widens, as with more nodes there are even more potential common nodes where energy savings can be realized. We also see this with the performance of the OCND algorithm, as its relative performance also increases with larger N.

Figure 10 shows an energy cost comparison between optimal pairs of node and link-disjoint paths. Clearly, link-disjoint paths are far more energy efficient than node-disjoint paths, with the difference widening drastically with increasing N (e.g. for $N = 50$, the optimal node-disjoint path pair consumes 25% more energy than the optimal link-disjoint path pair). This obviously has great consequences when one considers this in the context of reliability. While transmission along node-disjoint paths is, from a reliability perspective, more desirable, figure 10 shows that it is much more energy efficient to transmit along link-disjoint paths.

We finally explore the "cost of additional reliability". Figure 11 shows an energy cost comparison between a single path, found by dijkstra's algorithm, up to 4 *node* disjoint paths, found using the optimal STPS algorithm. Figure 12 shows an energy cost comparison between a single path up to 4 *link* disjoint paths, where the 2 disjoint paths are found using the optimal OCND algorithm, and the 3 and 4 disjoint paths are found using the sub-optimal LD-ESP algorithm. Note that our intuition about the WMA tells us that the greater the number of paths, the more it can be exploited for energy savings. However, this is counter-balanced by the fact that additional paths tend to be longer than the shortest path. In the node-disjoint case we see from figure 11 that 4 node-disjoint paths seem to cost on average well over 4 times the energy cost of a single path. This can be explained by the fact that the energy savings attained at the source node by additional exploitation of the

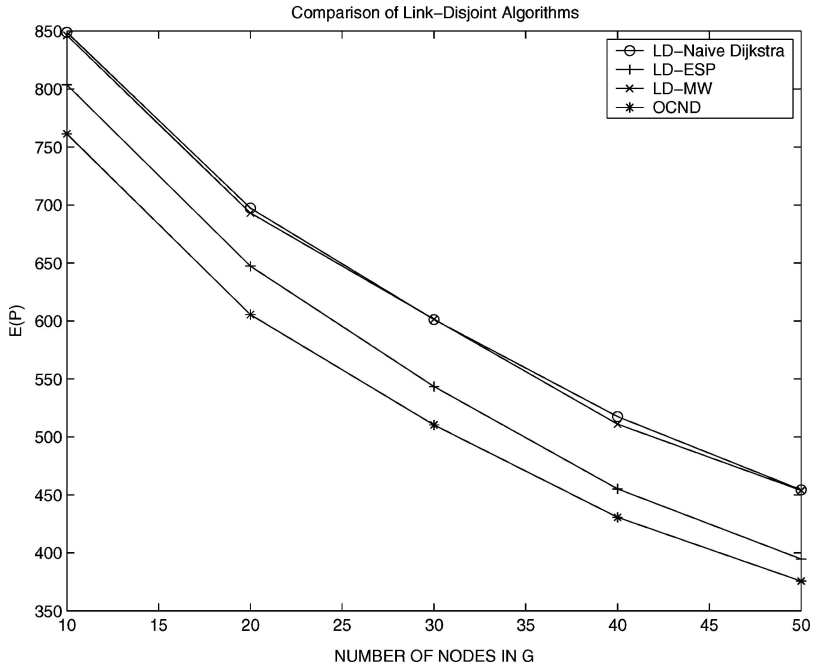


Figure 9. Comparison between energy-efficient link-disjoint algorithms.

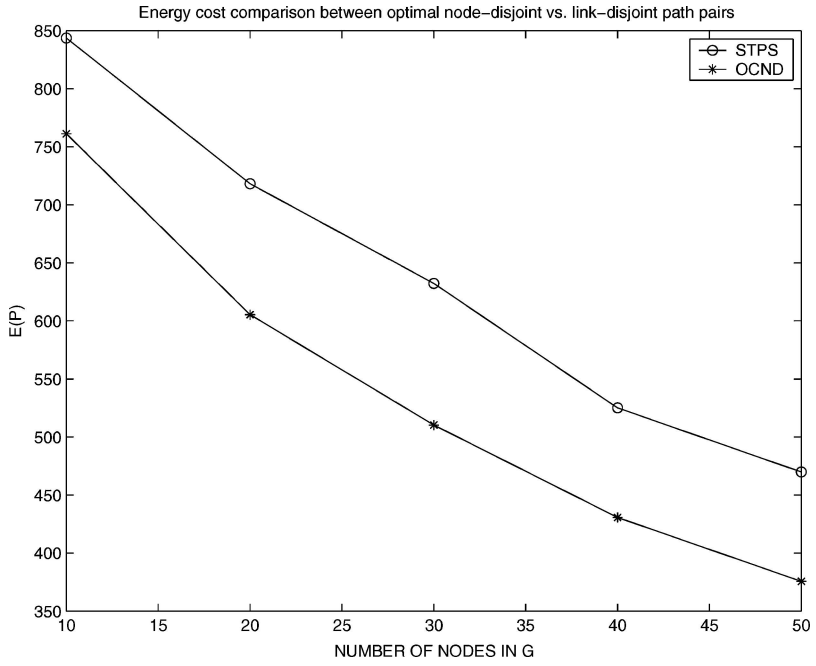


Figure 10. Comparison between pair of optimal node-disjoint vs. link-disjoint paths.

WMA is counter-acted by the additional cost of using longer and longer node-disjoint paths (i.e. the second shortest path is longer than the shortest path, etc.). In the case of link-disjoint paths however, we see from figure 12 that the path pairs found by the OCND algorithm are on average less than twice the cost of a single path (e.g. for $N = 50$, the cost of the minimum energy path pair is only 1.6 times the cost of the shortest path). Moreover, for larger N, the savings seem to increase (albeit marginally) as the number of paths increases.

7. Distributed implementation

In this section, we discuss issues regarding distributed implementation of the centralized algorithms presented in this paper. Such a discussion is important for most practical situations where global topology knowledge is not immediately available to all nodes in the network. Moreover, distributed implementation is important in instances where the topology may be changing frequently. For the purposes of this discussion, we

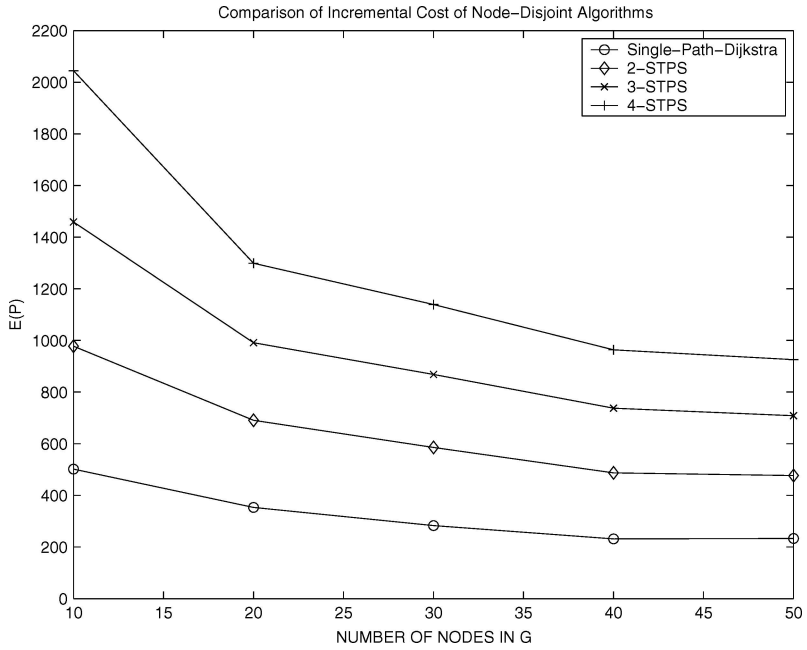


Figure 11. Incremental cost of adding additional node-disjoint paths.

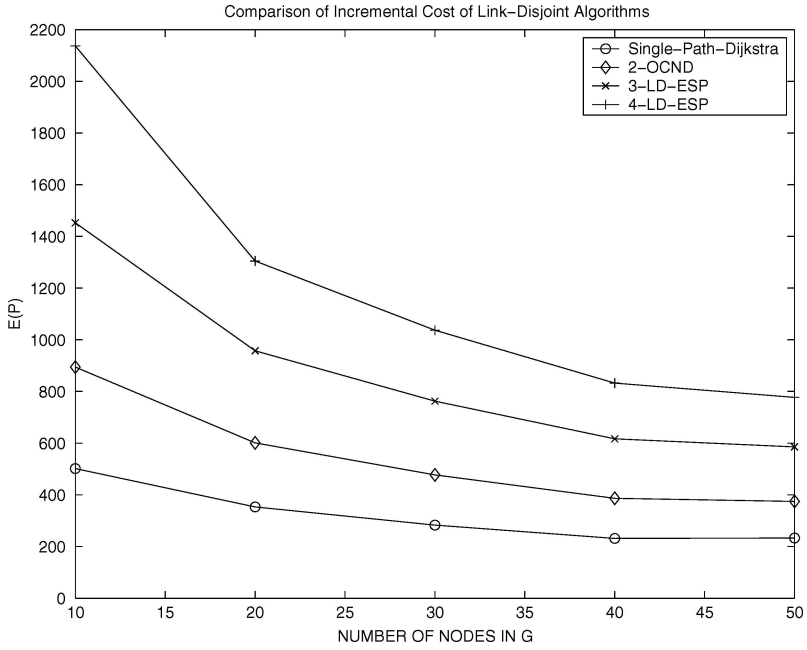


Figure 12. Incremental cost of adding additional link-disjoint paths.

assume that nodes have only local topology knowledge, i.e. the weights of the outgoing edges in the energy cost graph. For example, this can be easily found by each node employing a physical-layer probing mechanism using incremental power level increases [28].

First, we note that the algorithms can be made "distributed" in the sense that any centralized algorithm can be made distributed via some global topology dissemination mechanism (e.g. flooding or broadcast). Moreover, such a distributed algorithm can be made robust to topology change by periodically re-disseminating the topology information, re-running the algorithms locally upon change or when appropriate. Of course, there may be situations where one may not want to rely on such a dissemination mechanism, and a "truly" distributed implementation, where nodes need only to exchange information with their neighbours, is desirable.

Fortunately, the algorithms presented in this paper lend themselves to such a distributed implementation. To see this, note that optimal algorithms for both the shortest paths and minimum weight k disjoint paths problems have efficient

distributed implementations [3,9,17,18]. As discussed previously, the centralized versions of these algorithms serve as basic building blocks for the centralized algorithms presented in this paper. Similarly, the distributed versions of these building block algorithms can be used to construct distributed analogs to the STPS and OCND algorithms. A brief high level description of those algorithms follow:

Distributed STPS: Similar to the centralized STPS algorithm, run a distributed minimum weight k node-disjoint paths algorithm $M-k+1$ times, in each iteration adding/removing outgoing edges from the source node. After each iteration, the algorithm keeps the lowest energy paths found thus far as the current estimation of the minimum energy node-disjoint paths. The algorithm both converges and terminates after $M-k+1$ iterations. Note that to conserve total running time (at a cost of additional bandwidth), all $M-k+1$ instances of the distributed minimum weight disjoint paths algorithm are independent, and can thus be run simultaneously. This would result in a total convergence time equal to that of a single execution of the distributed minimum weight k disjoint paths algorithm.

Distributed OCND: Over time, each node x collects information regarding the minimum energy node-disjoint path pair between x and all other nodes y (e.g. by running a distributed STPS algorithm between x and y). Based on the current information node x has, it can individually set new edge weights on its outgoing edges (x, y) equal to the energy cost of the minimum energy node-disjoint path pair between x and y (analogous to the construction of the graph G^* in the centralized OCND algorithm). Finally, a distributed shortest paths algorithm is periodically run on the current G^*, resulting in a current estimation of the optimal common node decomposition, and thus the minimum energy link-disjoint paths.

Similarly, it should be clear that the distributed implementation of the heuristic algorithms presented earlier follow directly from the optimal distributed shortest paths and minimum weight k disjoint paths algorithms.

Dealing with Topology Changes: In a wireless ad-hoc network, the topology may change frequently. In part, the disjoint paths algorithms developed in this paper are designed to provide some resilience against such topological changes. When a link or a node "fails", the alternate paths are there to keep the connection active.

However, once a link or node has failed, the connection, while still active, is no longer supported by all of the original disjoint paths. It is therefore necessary to "recompute" the failed paths. One simple way to accomplish this is to find a new set of disjoint paths. While this solution may not be the most elegant, it is certainly feasible; especially because the connection is still active and hence there is no urgency in finding the new paths. An alternative approach, albeit (energy) sub-optimal, is to simply find new additional paths that consume the minimum amount of incremental energy. An example of this approach are the two heuristics presented in Section 5 based on the shortest path algorithms. This approach is computationally efficient as it only involves applications of a shortest-path algorithm. Moreover, it is also energy efficient as we observed in Section 5. In particular, the LD-ESP algorithm, which finds energy efficient link-disjoint paths sequentially, performed very close to the optimal algorithm.

8. Conclusion

In this paper, we presented a novel polynomial time algorithm that finds a pair of minimum energy link-disjoint paths in a wireless network. In addition, we presented an optimal algorithm that solves the minimum energy k node-disjoint paths problem in polynomial time, as well as fast, but sub-optimal heuristics for both problems. Our results show that link-disjoint paths consume substantially less energy than node-disjoint paths. We also found that the incremental energy of additional link-disjoint paths is decreasing. This finding is somewhat surprising due to the fact that in general graphs additional paths are typically longer than the shortest path. We determined that for the case of node-disjoint paths, the energy savings due to the use of the optimal algorithm (over a sub-optimal heuristic) was most notable in sparse graphs (i.e., N small); while for the link-disjoint case the energy savings were most notable in dense graphs.

It should be noted that the algorithms presented in this paper work for *general graphs*, as long as the objective is to minimize a node based aggregate metric of the form $C(x) = \max\{w_{xj} : (x, j) \in E\}$. The general nature of these algorithms makes them applicable to other wireless environments where the energy radiation may not be symmetric and the path losses between the nodes are not just a function of the distance between them (e.g., due to the physical terrain variations).

Lastly, although the algorithms presented in this paper are centralized, they lend themselves to distributed implementation as well. We presented distributed versions of the STPS and OCND algorithms. Further study of issues related to distributed implementation remain an important area for future work.

Appendix I: Enhanced source transmit power select (E-STPS) algorithm

The E-STPS Algorithm improves on the STPS algorithm for the specific case of $k = 2$, by performing a more efficient search over the source transmission ranges, $T(S)$. Before proceeding, we first need the following two lemmas, which form the basis for the algorithm.

Lemma 2. Consider a set of 2 node-disjoint S-D paths, $P = \{p_1, p_2\}$ with corresponding "source edges" (i.e. edges outgoing from the source) $\{m_1, m_2\}$, $w(m_1) \leq w(m_2)$, found by running a minimum weight 2 node-disjoint paths algorithm on a graph G. Next, consider a different set of 2 node-disjoint

paths, $P' = \{p'_1, p'_2\}$, $P' \neq P$. Then,

$$\forall \text{ such } P', \text{ if } w(m'_1) < w(m_1), \ \mathcal{E}(P') > \mathcal{E}(P) \tag{6}$$

Proof. Express the aggregate energy of a node-disjoint path pair, $P = \{p_1, p_2\}$, as $\mathcal{E}(P) = W(p_1) + W(p_2) - w(m_1)$, where $W(p_i)$ is the sum over the weights of all edges in the path p_i. Then,

$$\begin{aligned}\mathcal{E}(P') - \mathcal{E}(P) = [(W(p'_1) + W(p'_2)) - (W(p_1) + W(p_2))] \\ + [w(m_1) - w(m'_1)]\end{aligned} \tag{7}$$

The first square brackets term is non-negative, as $(W(p_1) + W(p_2))$ is minimum. Therefore, $\mathcal{E}(P') - \mathcal{E}(P) > 0$, and $\mathcal{E}(P') > \mathcal{E}(P)$. □

Lemma 3. Consider a set of 2 node-disjoint S-D paths, $P = \{p_1, p_2\}$, found by running a minimum weight 2 node-disjoint S-D paths algorithm on a transformed graph G^*, where G^* is equal to G, except all source edges are given weight 0. Define the residual path, R_{p_i}, of a path p_i as equal to $p_i - m_i$ (i.e. the path with the source edge removed). Again, consider a different set of 2 node-disjoint paths, $P' = \{p'_1, p'_2\}$, $P' \neq P$. Then,

$$\forall \text{ such } P', \text{ if } w(m'_2) > w(m_2), \text{ then } \mathcal{E}(P') > \mathcal{E}(P) \tag{8}$$

Proof. Express the energy of a node-disjoint path pair $P = \{p_1, p_2\}$ in a form similar to (4), i.e. $\mathcal{E}(P) = W(R_{p_1}) + W(R_{p_2}) + w(m_2)$, where $T(S) = w(m_2)$. We then have that:

$$\begin{aligned}\mathcal{E}(P') - \mathcal{E}(P) = \big[\big(W\big(R_{p'_1}\big) + W\big(R_{p'_2}\big)\big) \\ - \big(W\big(R_{p_1}\big) + W\big(R_{p_2}\big)\big)\big] + [w(m'_2) - w(m_2)]\end{aligned} \tag{9}$$

The first square brackets term is positive, as $(W(R_{p_1}) + W(R_{p_2}))$ is minimum. Therefore, $\mathcal{E}(P') - \mathcal{E}(P) > 0$, and $\mathcal{E}(P') > \mathcal{E}(P)$. □

Lemmas 2 and 3 give us a way of intelligently deciding which $T(S)$ values to search over. Lemma 2 tells us that once we have discovered a set of 2 node-disjoint S-D paths, P, as defined in the theorem, then we immediately know that any different set of paths that includes a source edge whose weight is *less* than the weight of the *minimum weight source edge* of P, cannot possibly have lower overall energy. Thus, we can eliminate all such source edges from the search space. Similarly, Lemma 3 allows us to eliminate from the search space, source edges with weight *greater* than the weight of the *maximum weight source edge* of the set of paths as defined in Lemma 3. These results lead directly to an elegant minimum energy 2 node-disjoint S-D paths algorithm, which we now present.

The E-STPS algorithm takes as input an energy cost graph $G = (V, E)$, and a source-destination pair, $S, D \in V$. Moreover, assume S has M outgoing edges $m_1, m_2, \ldots, m_M$, ordered such that $w(m_i) > w(m_j), \Leftrightarrow i > j$. Its output is the set of 2 minimum energy node-disjoint paths, $P_{\min}$.

Initialize: Let G_1 and G_2 represent two graphs, both equal to G, except the source edges in G_2 are given weight 0. Maintain two pointers, *LEFT* and *RIGHT*, initialized to 1 and M respectively, where $[m_{LEFT}, m_{RIGHT}]$ represents the range of source edges that we allow the minimum weight algorithm to use in any iteration.

Step 1: Run a minimum weight 2 node-disjoint paths algorithm on G_1 to obtain $P_1 = \{p_1, p_2\}$ with corresponding source edges $\{m_x, m_y\}$ as the minimum weight paths, where the integers x and y index the source edges with respect to the ordered source edges of the original graph, $LEFT \leq x < y \leq RIGHT$. Set $\mathcal{E}(P_1) = W(p_1) + W(p_2) - w(m_x)$ and increase *LEFT*, $LEFT = x + 1$.

Step 2: Run a minimum weight 2 node-disjoint paths algorithm on G_2 to obtain $P_2 = \{p'_1, p'_2\}$ with corresponding source edges $\{m_u, m_v\}$ as the minimum weight paths, where u and v are integers defined similar to x and y in step 2. Set $\mathcal{E}(P_2) = W(p'_1) + W(p'_2) + w(m_v)$ and decrease *RIGHT*, $RIGHT = v - 1$.

Step 3: Remove all source edges from both G_1 and G_2 except for those in the range $[m_{LEFT}, \ldots, m_{RIGHT}]$. This is the step where we narrow the search space in accordance with the results of Lemmas 2 and 3.

Step 4: Evaluate the minimum energy condition, $\mathcal{E}_{\min} = \min\{\mathcal{E}_{\min}, \mathcal{E}(P_1), \mathcal{E}(P_2)\}$, and update $P_{\min}$ accordingly.

Step 5: Repeat steps 2 to 5 until $LEFT \geq RIGHT$, at which point we would have exhausted the $T(S)$ search space. Moreover, at any iteration, if in step 2 there do not exist 2 minimum weight node-disjoint paths, we can exit the algorithm and conclude that the current $P_{\min}$ is the optimal minimum energy solution. This is because successive iterations would remove more source edges, which would only further inhibit the ability of the minimum weight 2 node-disjoint S-D paths algorithm to find node-disjoint paths.

The correctness of the E-STPS algorithm follows directly from Lemmas 2 and 3, as all we are basically doing is perform an "intelligent" brute force search over the $T(S)$ values. Note that the E-STPS algorithm terminates after at most $\lceil \frac{M-1}{2} \rceil$ iterations, since after each iteration the pointers *LEFT* and *RIGHT* are incremented/decremented by at least 1.

A final note about the lemmas, is that while they can be generalized to any k, the subsequent results do not seem to give us an intuitive way to proceed as they do for $k = 2$.

Appendix II: LD-MW *k*-approximateness proof

Theorem 2. Let $P = \{p_1, p_2, \ldots, p_k\}$ be the set of k link-disjoint S-D paths found by running the LD-MW algorithm on G, and let $P^* = \{p^*_1, p^*_2, \ldots, p^*_k\}$ be a set of optimal minimum energy k link-disjoint S-D paths. Then,

$$\forall G, \mathcal{E}(P) < k\mathcal{E}(P^*) \tag{10}$$

Proof. Let $N(P)$ represent the total weight of edges in the solution set P that we obtain "for free", i.e. the aggregate

energy savings at the common nodes. We upper bound the total energy of the algorithm solution, $\mathcal{E}(P)$, by using the fact that $N(P) > 0$; this is true because at minimum the weight of at least one outgoing edge from the source will be saved by the WMA.

$$\mathcal{E}(P) = \sum_{i=1}^{k} W(p_i) - N(P) < \sum_{i=1}^{k} W(p_i) \tag{11}$$

Let k link-disjoint paths $p_1, p_2, \ldots, p_k$ be ordered such that $W(p_i) > W(p_j) \Leftrightarrow i > j$. We now establish a lower bound on the optimal minimum *energy* solution, $\mathcal{E}(P^*)$, noting that in the best case scenario $N(P^*)$ will account for maximum energy savings at the common nodes, which will at most be the weight of all paths other then the "maximum weight path" (i.e. in this best case scenario, this path corresponds to both the maximum weight path as well as the path consisting of the maximum weight outgoing edges from the common nodes), i.e. $N(P^*) < \sum_{i=1}^{k-1} W(p_i^*)$.

$$\begin{aligned}\mathcal{E}(P^*) &= \sum_{i=1}^{k} W(p_i^*) - N(P^*) \\ &> W(p_k^*) \\ &\geq \frac{1}{k} \sum_{i=1}^{k} W(p_i)\end{aligned} \tag{12}$$

Where the last line follows comes from the following observation, based on the LD-MW solution, P, being minimum *weight* (different from energy!):

$$\begin{aligned}\sum_{i=1}^{k} W(p_i^*) \geq \sum_{i=1}^{k} W(p_i) &\Leftrightarrow k W(p_k^*) \\ \geq \sum_{i=1}^{k} W(p_i) &\Leftrightarrow W(p_k^*) \geq \frac{1}{k} \sum_{i=1}^{k} W(p_i)\end{aligned} \tag{13}$$

It should be noted that the the result of equation 13 follows from the general fact that if the sum of k numbers is greater than some value z, then at least one (e.g. the maximum) of the k numbers must be greater than or equal to the average (e.g. z/k). Finally, combining the results of equations (11) and (12), we have the following relations, and the result is shown.

$$\frac{1}{k} \sum_{i=1}^{k} w(p_i) < \mathcal{E}(P^*) \leq \mathcal{E}(P) < \sum_{i=1}^{k} w(p_i) \tag{14}$$

□

The above result applies to the ND-MW algorithm as well, since node-disjoint paths are simply link-disjoint paths with 1 common node, namely the source node.

Appendix III: Enhanced optimal common node decomposition (E-OCND) algorithm

Thus far, we have only used the basic single-source single destination minimum weight k disjoint paths algorithms as the main building blocks for the minimum energy algorithms described in this paper. However, there exist very efficient algorithms [25,26] that find minimum weight disjoint path pairs between a single-source and *all other nodes* in a single shot; these algorithms solve the *single-source N-destination minimum weight 2 disjoint paths problem*. It turns out we can use these algorithms to significantly reduce the complexity of our minimum energy 2 link-disjoint S-D paths algorithm (i.e. the OCND algorithm).

To this end, we first note that the $O(N^5)$ complexity of the OCND algorithm is concentrated in step 1, where by comparison steps 2 and 3 take just $O(N^2)$ time. Therefore, reducing the complexity of step 1 is the key to reducing the complexity of the OCND algorithm.

Next, we note that the function of step 1 is to find minimum energy 2 node-disjoint paths between all distinct node pairs in the network. In the original OCND algorithm, we did this by simply running our minimum energy node-disjoint paths algorithm (i.e. the STPS algorithm) $N(N-1)$ times; once for each distinct node pair. However, we can do this more efficiently by changing our implementation of the STPS algorithm, such that in step 2 (of the STPS) we employ a single-source N-destination minimum weight node-disjoint paths algorithm, instead of a single-source *single*-destination minimum weight node-disjoint paths algorithm. Our complete modification of step 1 of the OCND algorithm, incorporating the above change to the STPS implementation, is presented below; the resulting algorithm is referred to as the Enhanced Optimal Common Node Decomposition (E-OCND) algorithm.

Modified Step 1a: Consider a node v, and assume v has M outgoing edges $m_1, m_2, \ldots, m_M$, ordered such that $w(m_i) > w(m_j) \Leftrightarrow i > j$. Let $P_{\min}^{v,w}$ represent the current minimum energy node-disjoint path pair between node v and node w, and $\mathcal{E}_{\min}^{v,w}$ their aggregate energy cost. Initialize an integer variable $c = 2$.

Modified Step 1b: Remove edges $m_{c+1}, \ldots, m_M$ from the graph. Set $w(m_1), w(m_2), \ldots, w(m_c)$ equal to 0. Run a single-source N-destination minimum weight 2 node-disjoint paths algorithm on the modified graph, where v is the source. Let $P^{v,w}$ represent the solution paths between v and w found by the algorithm, and $W(P^w)$ their aggregate weight.

Modified Step 1c: For every node w, evaluate the following condition: if $W(P^{v,w}) + w(m_c) < \mathcal{E}_{\min}^{v,w}$, then set $\mathcal{E}_{\min}^{v,w} = W(P^{v,w}) + w(m_c)$ and $P_{\min}^{v,w} = P^{v,w}$.

Modified Step 1d: Increment $c = c+1$. Repeat steps 1b through 1d until $c > M$, at which point for all nodes w, $P_{\min}^{v,w}$ will represent the minimum energy node-disjoint path pair between v and w.

Modified Step 1e: Repeat steps 1a through 1d for all nodes v.

Steps 2 and 3 are kept the same as in the original OCND algorithm. Note that the modified step 1 is correct since for every source v, it performs the exact same brute force search over all relevant $T(v)$ values as in the original STPS algorithm.

We next address the complexity of the E-OCND algorithm. First, we observe that steps 1a–c take $O(N^2)$ time (e.g. using Suurballe and Tarjan's implementation of the single-source N-destination minimum weight 2 node-disjoint paths algorithm [26]). Next we note that steps 1a–1c are executed $(M-1)(N-1)$ times (where $M = N-1$ in the worst case), which results in an overall complexity for the E-OCND algorithm of $O(N^4)$; much better than the $O(N^5)$ complexity of the original OCND algorithm.

References

[1] A. Ahluwalia, E. Modiano and L. Shu, On the complexity and distributed construction of energy-efficient broadcast trees in static ad hoc wireless networks, in: *Proc. 36th Annual Conference on Information Sciences and Systems (CISS)* (March 2002).

[2] S. Banerjee and A. Misra, Minimum energy paths for reliable communication in multi-hop wireless networks, in: *Proc. 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)* (June 2002) pp. 146–156.

[3] D. Bertsekas and R. Gallager, *Data Networks* (Prentice-Hall Upper Saddle River, NJ, 1992).

[4] R. Bhandari, Optimal physical diversity algorithms and survivable networks, in: *Proc. Second IEEE Symposium on Computers and Communications* (July 1997) pp. 433–441.

[5] M. Cagali, J.P. Hubaux and C. Enz, Minimum energy broadcast in all-wireless networks: NP-completeness and distribution issues, in: *Proc. Ninth ACM International Conference on Mobile Networking and Computing (MOBICOM)* (September 2002) pp. 172–182.

[6] G. Calinescu, I.I. Mandoiu and A. Zelikovsky, Symmetric connectivity with minimum power consumption in radio networks, in: *Proc. 2nd IFIP International Conference on Theoretical Computer Science* (August 2002) pp. 119–130.

[7] J.H. Chang and L. Tassiulas, Energy conserving routing in wireless ad-hoc networks, in: *Proc. IEEE INFOCOM* (March 2000) pp. 22–31.

[8] W.T. Chen and N.F. Huang, The strongly connecting problem on multihop packet radio networks, IEEE Transactions on Communications 37 (1989) 293–295.

[9] C. Cheng, S. Kumar and J.J. Garcia-Luna-Aceves, An efficient distributed algorithm for routing over k-disjoint paths of minimum total length, in: *Proc. 28th Annual Allerton Conference on Communication, Control, and Computing* (October 1990).

[10] A.E.F. Clementi, P. Crescenzi, P. Penna and P.V.G. Rossi, On the complexity of computing minimum energy consumption broadcast subgraphs, in: *Proc. 18th Annual Symposium on Theoretical Aspects of Computer Science* (February 2001) pp. 121–131.

[11] A. Ephremides, Energy concerns in wireless networks, IEEE Wireless Communications 9 (August 2002) 48–59.

[12] L.M. Feeney and M. Nilson, Investigating the energy consumption of a wireless network interface in an ad hoc networking environment, in: *Proc. IEEE INFOCOM* (April 2001) pp. 1548–1557.

[13] S.J. Lee and M. Gerla, split multipath routing with maximally disjoint paths in ad hoc networks, in: *Proc. IEEE ICC* (June 2001) pp. 3201–3205.

[14] W. Liang, Constructing minimum energy broadcast trees in wireless ad hoc networks, in: *Proc. 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)* (June 2002) pp. 112–122.

[15] E. Lloyd, R. Liu, M. Marathe, R. Ramanathan and S.S. Ravi, Algorithmic aspects of topology control problems for ad hoc networks, in: *Proc. 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)* (June 2002) pp. 123–134.

[16] A. Nasipuri and S.R. Das, On-demand multi-path routing for mobile ad hoc networks, in: *Proc. IEEE ICCCN* (October 1999) pp. 64–70.

[17] R.G. Ogier, V. Rutenburg and N. Shacham, Distributed algorithms for computing shortest pairs of disjoint paths, IEEE Transaction on Information Theory 39 (May 1993) 173–182.

[18] Richard G. Ogier and Nachum Shacham, A distributed algorithm for finding shortest pairs of disjoint paths, in: *Proc. IEEE INFOCOM* (April 1989) pp. 173–182.

[19] R. Ramanathan and R. Rosales-Hain, Topology control of multihop wireless networks using transmit power adjustment, in: *Proc. IEEE INFOCOM* (March 2000) pp. 404–413.

[20] T.S. Rappaport, *Wireless Communications: Principles and Practices* (Prentice-Hall Upper Saddle River, NJ, 1996).

[21] D. Sidhu, R. Nair and S. Abdallah, Finding disjoint paths in networks, in: *Proc. ACM SIGCOMM* (September 1991) pp. 43–51.

[22] S. Singh, M. Woo and C.S. Raghavendra, Power-aware routing in mobile ad hoc networks, in: *Proc. Fifth ACM International Conference on Mobile Networking and Computing (MOBICOM)* (Oct. 1998) pp. 181–190.

[23] A. Srinivas and E. Modiano, Minimum energy disjoint path routing in wireless ad hoc networks, in: *Proc. Ninth ACM International Conference on Mobile Networking and Computing (MOBICOM)* (September 2003) pp. 122–133.

[24] J.W. Suurballe, Disjoint paths in a network, Networks 4 (1974) 125–145.

[25] J.W. Suurballe, The single-source, all terminals problem for disjoint paths, Unpublished technical memorandum, Bell Laboratories (1982).

[26] J.W. Suurballe and R.E. Tarjan, A quick method for finding shortest pairs of disjoint paths, Networks 14 (1984) 325–336.

[27] S. Vutukury and J.J. Garcia-Luna-Aceves, MDVA: A distance-vector multipath routing protocol, in: *Proc. IEEE INFOCOM* (April 2001) pp. 557–564.

[28] R. Wattenhofer, L. Li, P. Bahl and Y. Wang, Distributed topology control for power efficient operation in multihop wireless ad-hoc networks, in: *Proc. IEEE INFOCOM* (April 2001) pp. 1388–1397.

[29] J.E. Wieselthier, G.D. Nguyen and A. Ephremides, On the construction of energy-efficient broadcast and multicast trees in wireless networks, in: *Proc. IEEE INFOCOM* (March 2000) pp. 585–594.

[30] J.E. Wieselthier, G.D. Nguyen and A. Ephremides, The energy efficiency of distributed algorithms for broadcasting in ad hoc networks, in: *Proc. The 5th International Symposium on Wireless Personal Multimedia Communications* (Oct. 2002) pp. 499–503.

Anand Srinivas is currently a PhD candidate in the Laboratory for Information and Decision Systems (LIDS) at MIT. He recieved his Masters of Science in EECS from MIT in 2004, and his Bachelors of Applied Science in Computer Engineering from the University of Toronto in 2001. In 2004 he also received a Masters of Science in Aerospace Engineering from MIT. His current research interests include reliability and energy-efficiency in wireless ad-hoc networks, routing and network optimization, graph theory, and the design of efficient algorithms.
E-mail: anand3@mit.edu

Eytan Modiano received his B.S. degree in Electrical Engineering and Computer Science from the University of Connecticut at Storrs in 1986 and his M.S. and Ph.D. degrees, both in Electrical Engineering, from the University of Maryland, College Park, MD, in 1989 and 1992 respectively. He was a Naval Research Laboratory Fellow between 1987 and 1992 and a National Research Council Post Doctoral Fellow during 1992–1993 while he was conducting research on security and performance issues in distributed network protocols.

Between 1993 and 1999 he was with the Communications Division at MIT Lincoln Laboratory where he designed communication protocols for satellite, wireless, and optical networks and was the project leader for MIT Lincoln Laboratory's Next Generation Internet (NGI) project. He joined the MIT faculty in 1999, where he is presently an Associate Professor in the Department of Aeronautics and Astronautics and the Laboratory for Information and Decision Systems (LIDS). His research is on communication networks and protocols with emphasis on satellite, wireless, and optical networks.

He is currently an Associate Editor for Communication Networks for IEEE Transactions on Information Theory and for The International Journal of Satellite Communications. He had served as a guest editor for IEEE JSAC special issue on WDM network architectures; the Computer Networks Journal special issue on Broadband Internet Access; the Journal of Communications and Networks special issue on Wireless Ad-Hoc Networks; and for IEEE Journal of Lightwave Technology special issue on Optical Networks. He is the Technical Program co-chair for Wiopt 2006 and vice- chair for Infocom 2007.

E-mail: modiano@mit.edu