Algorithmic Aspects of Minimum Energy Edge-Disjoint Paths in Wireless Networks

Markus Maier, Steffen Mecke, and Dorothea Wagner

Universität Karlsruhe (TH), Germany

Abstract. The problem of finding k minimum energy, edge-disjoint paths in wireless networks (MEEP) arises in the context of routing and belongs to the class of range assignment problems. A polynomial algorithm which guarantees a factor-k-approximation for this problem has been presented before, but its complexity status was open. In this paper we prove that MEEP is NP-hard and give new lower and upper bounds on the approximation factor of the k-approximation algorithm. For MEEP on acyclic graphs we introduce an exact, polynomial algorithm which is then extended to a heuristic for arbitrary graphs.

1 Introduction

Links between nodes of a wireless network are less reliable than connections in wired networks, because of effects like fading, interference, or obstructions. For reliable routing in wireless networks it can therefore be desirable to communicate not only over one path but over various, disjoint paths. This can help to achieve connections that are more reliable, have less latency, or higher bandwidth. Energy is a sparse resource in ad hoc and especially in sensor networks, therefore it is usually vital to achieve the connectivity goal with a minimum energy usage. The advantage of a wireless networks node, in this respect, is their ability to do *multicasts* to all of their neighbors, using only energy for one transmission. If several paths have a node in common, energy can be saved by doing just one multicast at this node instead of several unicast transmissions.

In [1] Srinivas and Modiano showed several algorithms for finding sets of k edge-disjoint or node-disjoint paths in wireless ad hoc networks. They gave a polynomial time algorithm for finding an energy-minimal set of k node-disjoint paths and a polynomial time algorithm for finding an energy-minimal pair of edge-disjoint paths. The node-disjoint case is less complex in the sense that, as the paths share no nodes except the start node s, only s can save energy by doing a multicast to its neighbors. For the edge-disjoint case a k approximation algorithm was presented in [1] (the LDMW algorithm). However, the complexity of the problem has remained unknown. Therefore, in this paper we concentrate on the edge-disjoint case. Here, energy can be saved also at intermediate nodes. The disadvantage of paths that are merely edge-disjoint is that they may not protect against node failures. One main difference between node and edge failures is that the reasons for the latter are often only temporary (e.g., interference or obstruction) whereas reasons for node failures are often permanent (e. g., power

Jan van Leeuwen et al. (Eds.): SOFSEM 2007, LNCS 4362, pp. 410–421, 2007.
© Springer-Verlag Berlin Heidelberg 2007

loss or mobility). The permanent interruption of a path, therefore, has to be dealt with differently, namely by establishing a new path, whereas, in case of a transient failure, the system might just use the alternative paths until the failing link becomes available again.

1.1 Related Work and Overview

Energy-efficient routing has been looked at many times before. One of the first works was probably [2], followed by many more. Disjoint path routing is already present in [3] and has been rediscovered in the context of wireless sensor and ad hoc networks, e.g. in [4] or [5]. But [1] seems to be the first work on the combined problem of finding energy-minimal disjoint paths (namely MEEP). It is closely related to work on finding energy-efficient strongly connected ([6]) or strongly k-connected ([7]) subgraphs in the context of topology control. These two problems and the k-MEEP problem belong to the class of *range assignment* problems. One of the first works on this type of problem was [8], and [9] was another predecessor of [1]. Range assignment has been studied widely in the meantime, see also [10] for a survey. In a successive paper ([11]), MEEP is extended and analyzed under lifetime aspects.

As Srinivas and Modiano already pointed out, the problem of finding minimum energy disjoint paths is more focused than finding k-connected subgraphs, in the sense that it is only concerned with finding disjoint paths between one pair of nodes and therefore does not need to maintain routes between nodes that may never have to communicate with each other at all. Whereas they gave polynomial time algorithms for finding minimum energy node-disjoint paths and pairs of edge-disjoint paths, our first result (Sect. 3) is that the MEEP problem is NP-complete, at least, if k is part of the input. It was shown in [1] that an algorithm for finding k minimum *length* edge disjoint paths (like, for example Suurballe's algorithm, [3]) provides at least k-approximation. In Sect. 4 we show that the factor of k is exact. However if we assume all edge weights to be equal we can prove a asymptotically tight bound of $\Theta(\sqrt{n})$ for the approximation factor of this algorithm. Finally, in Sect. 5, we present an exact, polynomial time algorithm for acyclic graphs. Based on this algorithm, we describe a simple heuristic for arbitrary graphs in Sect. 6.

2 Network Model

We use a slightly more general network model than that of [1]. A network consists of n nodes. Each node v has a maximum transmission power $\mathcal{E}_{\max}(v)$ and can transmit at any power in the interval $[0, \mathcal{E}_{\max}(v)]$. For each (ordered) pair (u, v) of distinct nodes we are given a weight $w(u, v)$. The node u can establish a link to a node v if it transmits with power greater than or equal to $w(u, v)$. These weights need not be symmetric, nor does the triangle inequality need to hold. This includes energy metrics (with $d(u, v) = \delta(u, v)^\alpha$ for $\alpha \in \{0, 1, 2, ...\}$) as well as other, more general metrics.

Clearly, such a network can be modeled by a weighted directed graph where the nodes are the nodes of the network and there is an edge from a node u to a node v if $\mathcal{E}_{\max}(u) \geq w(u,v)$, i. e. if there can be a link from u to v. In this case the corresponding edge weight is $w(u,v)$.

As mentioned above, we can use the wireless character of a network to save energy in a multicast setting (the so-called wireless multicast advantage (WMA)). Consequently, the cost of a set of paths should not be measured by the sum of the edge weights. Instead we define the following cost function which we call energy:

Definition 1 (Energy). *The* energy $\mathcal{E}(P)$ *of a set P of paths in a directed graph is defined as*

$$\mathcal{E}(P) = \sum_{u \in V(P)} \max_{(u,v) \in A(P)} w(u,v)$$

where $V(P)$ denotes the nodes and $A(P)$ the edges in the path and we set for convenience $\max_{\emptyset} = 0$. *The* weight *of P is*

$$w(P) = \sum_{(u,v) \in A(P)} w(u,v) \ .$$

Formally, the decision version of the problem of finding k minimum energy edge-disjoint paths can be stated as follows:

Definition 2 (MEEP). *Given a directed acyclic graph $D = (V, A)$ with weights $w : A \to \mathbb{R}^+$, two nodes $s, t \in V$, $B \in \mathbb{N}$ and the number $k \in \mathbb{N}$ of paths. Are there k edge-disjoint paths P from s to t with $\mathcal{E}(P) \leq B$?*

3 Complexity

There are polynomial time algorithms for finding k edge-disjoint paths of minimum weight (i. e. sum of edge weights) in a graph [3]. However, since in MEEP a different cost function is used, the problem becomes NP-complete for general k (i. e. for k being part of the input). We will show this in the following by reduction of SET COVER to MEEP.

Theorem 1. *Given a directed graph $D = (V, A)$ with weights $w : A \to \mathbb{R}^+$, start node $s \in V$, end node $t \in V$, $k \in \mathbb{N}$ and a threshold $B \in \mathbb{N}$. Then it is NP-complete to decide if there is a set P of k edge-disjoint paths from s to t with $\mathcal{E}(P) \leq B$.*

Proof. We will show the theorem by a reduction of the classic SET COVER problem to MEEP. Let us first remind the reader of the definition of SET COVER:

Definition 3 (SET COVER). *Given a set $U = \{u_1, \ldots, u_n\}$, a family $F = \{S_1, \ldots, S_s\}$ of subsets of U and an integer B. Can we select B (or less) subsets from F such that every element of U is in at least one of the selected subsets?*

It is a well-known fact that SET COVER is NP-complete [12]. Given an instance of SET COVER, we can construct a directed acyclic graph $D = (V, A)$ in polynomial time such that there is a correspondence between n edge-disjoint paths in D and the set covers. The construction is as follows: The nodes V of D consist of the elements of U and F, two nodes s and t, and for every set $S_i \in F$ with n_i elements, we have nodes $v_{i,1}, \ldots, v_{i,|S_i|}$. From s there is an edge to every $v_{i,j}$ $(i \in \{1, \ldots, s\}, j \in \{1, \ldots, |S_i|\})$. From the nodes $v_{i,j}$ there are edges to the corresponding sets S_i. From each set S_i there is an edge to every node $u \in U$ with $u \in S_i$. Finally, there are edges from all the nodes in U to t. All edges are assigned a weight of 1. Fig. 1 shows an example of this reduction.

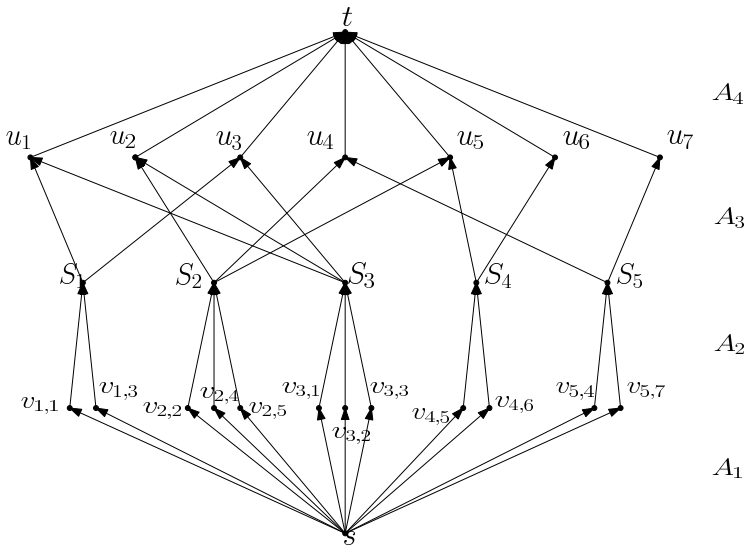


Fig. 1. Reduction of an instance of SET COVER with $U = \{u_1, \ldots, u_7\}$, $F = \{S_1, \ldots, S_5\}$, $S_1 = \{u_1, u_3\}$, $S_2 = \{u_2, u_4, u_5\}$, $S_3 = \{u_1, u_2, u_3\}$, $S_4 = \{u_5, u_6\}$, $S_5 = \{u_4, u_7\}$ to MEEP

Clearly, the size of this graph is polynomial in the size of the SET COVER problem and it can be constructed in polynomial time. We will show that there is a set cover of size less than or equal to B if and only if there are n edge-disjoint paths P from s to t with $\mathcal{E}(P) \leq B + 2n + 1$.

Given a set cover of size B – w. l. o. g. the sets $S_1 \ldots, S_B$ – we can construct n edge-disjoint paths from s to t as follows: For every element of U we can find a set in $S_1, \ldots, S_B$ that covers the element. Let $n_i \leq |S_i|$ $(i \in \{1, \ldots, B\})$ denote the number of elements that are thus associated with set S_i. Clearly $n_1 + \ldots + n_B = n$. Now we construct n_i edge-disjoint paths from s to S_i (via the nodes $v_{i,1}, \ldots, v_{i,n_i}$) and n_i edge-disjoint paths from S_i to t (via the elements that are covered by S_i. Together, we have constructed n edge-disjoint paths from s to t. Since all edge weights are 1, the energy consumed by the paths is equal to

the number of nodes (except t), namely $\mathcal{E}(P) = 1 + n + B + n = B + 2n + 1$, as desired.

Given n edge-disjoint paths from s to t they must visit exactly n of the nodes $v_{i,j}$ and n of the nodes $u_1, \ldots, u_n$, by construction of the graph. Thus, the energy of the paths is $B + 2n + 1$, where B is now the number of used nodes in $S_1, \ldots, S_s$. The paths easily induce a set cover by taking for every element $u \in U$ the predecessor on the path visiting u (one of $S_1, \ldots, S_s$). As a result, we have found a set cover with B or less sets.

We have shown that there is a set cover of size B if and only if there is a set of n edge-disjoint paths from s to t in D with energy $1 + 2n + B$, which implies NP-completeness. □

Remark 1. From the NP-completeness proof we can derive a result about the best possible approximation factor for MEEP using a result of Feige: Feige could show in [13] that there cannot exist an approximation algorithm for SET COVER with an approximation factor better than $O(1-o(1)) \log n$ unless NP has slightly superpolynomial time algorithms. Using this result one can easily show that under the same conditions the same bound holds for the approximation of MEEP.

4 Approximation

As we have seen in the previous section we cannot expect to find an approximation algorithm with an approximation factor of less than $O(\log k)$ for MEEP. In [1] the so-called Link-Disjoint Minimum-Weight (LDMW) algorithm was proposed and shown to possess an approximation factor of less than or equal to k. In this algorithm, the k paths of minimum weight are computed instead of the paths of minimum energy (e. g. using Suurballe's algorithm [3]). The example in Fig. 2 shows, that the approximation factor of the LDMW algorithm is exactly k.

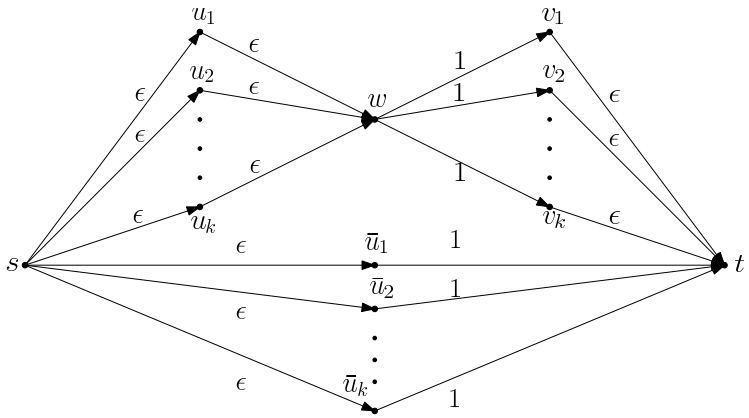


Fig. 2. An example that shows an approximation factor of k for the LDMW approximation algorithm

In this example, all of the upper paths (via w) have a weight of $1+3\varepsilon$, whereas the lower paths have a weight of $1+\varepsilon$. Thus, the k paths found by the LDMW algorithm are the lower paths which need an energy of $k+\varepsilon$. The energy of the upper paths, however, is $1+(2k+1)\varepsilon$. For $\varepsilon \to 0$ the quotient of the LDMW solution and the optimal solution approaches k.

4.1 The Binary Case

The example in Fig. 2 works because of great differences in edge weights. It is an interesting question if we can attain a better approximation factor if no such great differences can occur. We studied the case where all edges have the same weight (w. l. o. g. 1) and called this restricted problem BMEEP (for Binary MEEP), because the nodes can either send or not, but not send at different energy levels. Note that our proof of NP-completeness works for BMEEP as well. As explained above, however, the example that shows an approximation factor of k for the LDMW algorithm applied to MEEP does not work for BMEEP. Can we expect LDMW to work better on BMEEP?

We can give an example that shows for $k \in \mathbb{N}$ of the form $k = 1+2+\ldots+l$ for some $l \in \mathbb{N}$ that the approximation factor cannot be better than $1+\frac{l}{2}$ which is $\Omega(\sqrt{k})$. Fig. 3 shows the example for $k = 1+2 = 3$. We will discuss this example and briefly show how it can be extended to the general case.

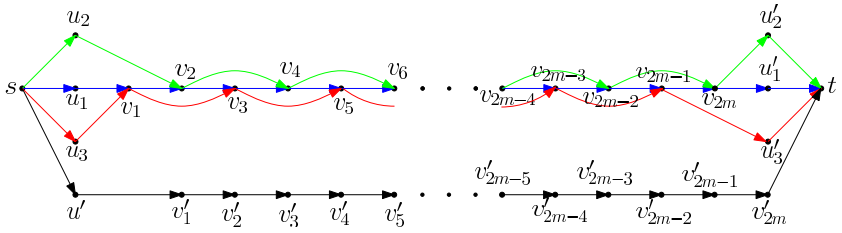


Fig. 3. A lower bound on the approximation factor of LDMW applied to BMEEP

$$\begin{aligned} \text{Let} \quad & P_1 = (s, u_1, v_1, v_2, \ldots, v_{2m}, u_1', t), \\ & P_2 = (s, u_2, v_2, v_4, \ldots, v_{2m}, u_2', t), \\ & P_3 = (s, u_3, v_1, v_3, \ldots, v_{2m-1}, u_3', t), \quad \text{and} \\ & P_1' = (s, u', v_1', \ldots, v_{2m}', t) \end{aligned}$$

denote four s-t-paths. Clearly, the weights of the paths are $W(P_1) = 2m+3$, $W(P_2) = W(P_3) = m+3$ and $W(P_1') = 2m+2$. Thus, the three paths of minimum weight are P_1', P_2 and P_3 with an energy of $\mathcal{E}(P_1', P_2, P_3) = 4m+6$. On the other hand $\mathcal{E}(P_1, P_2, P_3) = 2m+7$. Thus the factor between the LDMW solution and the optimal solution is $\frac{4m+6}{2m+7}$ which approaches 2 for $m \to \infty$.

The idea of the general case is to use i paths of "step size" i where $i \in \{1, \ldots, l\}$ as optimal paths (P_1 with step size 1; P_2 and P_3 with step size 2

above) and $m = \frac{l(l-1)}{2}$ parallel paths that are slightly shorter than the paths of "step size" $l-1$. Then one can show the factor of $1+\frac{l}{2}$ between the LDMW solution and the optimal solution.

The next lemma shows that this approximation factor is asymptotically tight.

Lemma 1. *For $k \geq 6$ the approximation factor of the LDMW algorithm is at least $2\sqrt{k}$.*

Proof. Let P^* be the set of k edge-disjoint paths with minimum energy and P' the set of k edge-disjoint paths with minimum weight. It is sufficient to show that $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$ where all but the last inequality is obvious. Let's have a look at the graph that is induced by P^*. It is the union of k shortest (directed) s-t-paths and therefore (w.l.o.g.) a directed acyclic graph of n nodes (if there are cycles, they can be removed). There exists an order $s = v_1, v_2, v_3, \ldots, v_{n-1}, v_n = t$ of the nodes of this graph (for instance the topological order) such that every edge goes "upward". The weight of P^* is the number of edges of this graph. The energy is the number of nodes (minus 1). The more edges the graph has, the shorter they have to be (where the length of an edge (v_i, v_j) is $|j-i|$). There are at most $n-1$ edges of length 1, $n-2$ edges of length 2 and so on. Every path leads from node s to node t, so the "distance" it crosses is $n-1$. Even if P^* uses only the shortest possible edges, we claim that the total distance is "used up" after edges of length $2\sqrt{k}$ are used because it is at least

$$\begin{aligned}\sum_{i=1}^{2\sqrt{k}} i(n-i) &= \sum in - \sum i^2 = 2nk + n\sqrt{k} - 8/3k\sqrt{k} - 6\sqrt{k} - 1 \\ &\geq k(n-1)\end{aligned}$$

for $n \geq k \geq 6$. Therefore the *number* of used edges is at most

$$W(P^*) \leq \sum_{i=1}^{2\sqrt{k}} (n-i) = 2n\sqrt{k} - 2k - \sqrt{k} \leq 2\sqrt{k}E(P^*) \ . \qquad \square$$

For the case $k = 3$ we could show that the approximation factor is exactly k. The proof is rather long and technical and cannot be given here. It can be found in other works by the authors. For the case of general k, the lower bound of $1+\frac{l}{2}$ (for $k = \frac{l(l+1)}{2}$) asymptotically matches the upper bound of $2\sqrt{k}$.

5 An Algorithm for Acyclic Graphs

In Sect. 3 we have shown that MEEP is NP-complete for weighted directed graphs and the number k of paths part of the input. In this section we will show that there is an exact, polynomial time algorithm if we restrict the graphs to be considered to acyclic graphs and fix a certain $k \in \mathbb{N}$. The algorithm relies on a notion from graph drawing that has to be presented first, so-called layerings.

We will first give an algorithm for properly layered graphs and then show briefly how we can transform an acyclic graph to a properly layered graph. Combining these steps we get a polynomial time algorithm for acyclic (directed) graphs.

5.1 Algorithm for Properly Layered Graphs

Layerings are a well-studied problem in graph drawing. The following definitions are from [14].

Definition 4. *A* layering *of an acyclic digraph $D = (V, A)$ is a partition of V into subsets (layers) $L_1, \ldots, L_h$, such that if $(u, v) \in A$, $u \in L_i$ and $v \in L_j$, then $i > j$. The* span *$d(e)$ of an edge $e = (u, v)$ where $u \in L_i$ and $v \in L_j$ is defined as $d(e) = j - i - 1$. A layering is called* proper *if $d(e) = 0$ for all edges $e \in A$.*

For every acyclic digraph a layering can be computed in linear time, e.g. by longest path layering [14]. From now on, we confine ourselves to finding k edge-disjoint paths in properly layered graphs for fixed k.

Theorem 2. *Given a weighted acyclic digraph $D = (V, A)$ with weights $w : A \to \mathbb{R}^+$, a proper layering into layers $L_0, \ldots L_h$, start node s and end node t, we can compute k minimum-energy edge-disjoint paths from s to t in time $O(n^k m^k)$.*

Proof. W. l. o. g. we can assume that $L_0 = \{s\}$ and $L_h = \{t\}$.

Since we have a proper layering, all edges go from one layer to the next. Thus the set of edges can be divided into layers as well. Let

$$A_i = \{(u, v) \in A | u \in L_{i-1}, v \in L_i\}$$

for $i = 1, \ldots, h$. Clearly A is the disjoint union of $A_1, \ldots, A_h$.

Obviously, a set of k edge-disjoint paths from s to t must use exactly k (different) edges from each edge layer. In the following we will consider k edge-disjoint paths from the source s to k-combinations of nodes from the same layer. For a k-combination (without repetitions) $(e_1, \ldots, e_k)$ of edges from a layer A_i let $\phi(e_1, \ldots, e_k)$ be the combination of the start nodes and $\psi(e_1, \ldots, e_k)$ the combination of the end nodes.

Let $(u_1, \ldots, u_k)$ be a k-combination (with repetitions) of nodes from layer L_i $(i = 0, \ldots, h-1)$. $\mathcal{E}(u_1, \ldots, u_k)$ denotes the minimum energy of k edge-disjoint paths from s to $(u_1, \ldots, u_k)$ and $\mathcal{E}_{(e_1,\ldots,e_k)}(u_1, \ldots, u_k)$ the minimum energy of k edge-disjoint paths from s to the node combination $(u_1, \ldots, u_k)$ using the edge combination $(e_1, \ldots, e_k)$.

Given a k-combination with repetitions $(u_1, \ldots, u_k)$ of nodes from a layer L_i $(i = 1, \ldots, h)$, let $A_{(u_1,\ldots,u_k)}$ denote the set of k-combinations of edges leading to $(u_1, \ldots, u_k)$. Then

$$\begin{aligned}\mathcal{E}(u_1, \ldots, u_k) &= \min_{(e_1,\ldots,e_k) \in A_{(u_1,\ldots,u_k)}} \mathcal{E}_{(e_1,\ldots,e_k)}(u_1, \ldots, u_k) \\ &= \min_{(e_1,\ldots,e_k) \in A_{(u_1,\ldots,u_k)}} \mathcal{E}(\phi(e_1, \ldots, e_k)) + \Delta_{(e_1,\ldots,e_k)} \ .\end{aligned}$$

Here, $\Delta_{(e_1,\ldots,e_k)}$ is the increase in energy when a set of k paths to a combination of nodes of a layer L_i is extended by edges $e_1,\ldots,e_k$ to a combination of nodes of layer L_{i+1}. Due to the multicast advantage, we get the following formula:

$$\Delta_{(e_1,\ldots,e_k)} = \sum_{u\in\phi(e_1,\ldots,e_k)} \max_{(u,v)\in(e_1,\ldots,e_k)} w(u,v) \ .$$

This leads to a dynamic programming approach: In order to compute the k minimum-energy paths to a combination $(v_1,\ldots,v_k)$ of nodes in layer L_{i+1}, we use the minimum-energy paths to all combinations of nodes in layer L_i: We enumerate all possible k-combinations (without repetitions) of edges from edge layer A_{i+1} leading to $(v_1,\ldots,v_k)$ and pick the combination with minimum total energy. The energy $\mathcal{E}(t,\ldots,t)$ is the energy of k minimum-energy edge disjoint s-t-paths and the paths themselves can be found by backtracking: For every combination of nodes we have to store the k edges on the minimum-energy paths leading there. The predecessor edges on the k edge-disjoint minimum-energy paths to a combination $(u_1,\ldots,u_k)$ are denoted by $pred(u_1,\ldots,u_k)$.

In summary we can give the following dynamic programming algorithm:

- Initialization for all combinations (with repetitions) $(v_1,\ldots,v_k)$ of nodes from layers $L_0,\ldots,L_h$:
  - $\mathcal{E}_{\min}(v_1,\ldots,v_k) = \begin{cases} 0 & \text{if } v_1 = v_2 = \ldots = v_k = s \\ \infty & \text{otherwise} \end{cases}$
- For all edge layers $A_i = A_1,\ldots,A_h$ do
  - For all combinations (without rep) $(e_1,\ldots,e_k)$ of edges from A_i
    - If $\mathcal{E}_{\min}(\psi(e_1,\ldots,e_k)) > \mathcal{E}_{\min}(\phi(e_1,\ldots,e_k)) + \Delta(e_1,\ldots,e_k)$
      - $\mathcal{E}_{\min}(\psi(e_1,\ldots,e_k)) = \mathcal{E}_{\min}(\phi(e_1,\ldots,e_k)) + \Delta(e_1,\ldots,e_k)$
      - $\text{pred}(\psi(e_1,\ldots,e_k)) = (e_1,\ldots,e_k)$
- $\mathcal{E}(t,\ldots,t)$ is the energy of k minimum energy edge-disjoint paths.
- k minimum-energy edge-disjoint paths are found by backtracking.

The running time of the algorithm above is determined by the total number of edge combinations to be considered. If we set $m_i = |A_i|$ $(i = 1,\ldots,h)$ this number is

$$\sum_{i=1}^{h} \binom{m_i}{k} \le \binom{m}{k} \in O(m^k) \ .$$

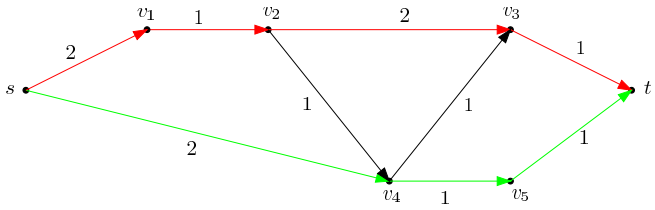
The algorithm needs to hold in memory a table of predecessor edges and the energy of a minimum-energy path for every combination of nodes from the same layer. Setting $n_i = |L_i|$ the number of combinations is

$$\sum_{i=0}^{h} \binom{n_i + k - 1}{k} \le \sum_{i=0}^{h} n_i^k \in O(n^k) \ ,$$

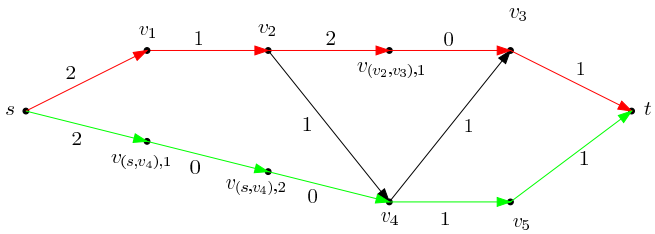
since the number of combinations with repetitions can clearly be bounded by the number of permutations with repetitions. □

5.2 Algorithm for Acyclic Graphs

For every acyclic digraph, a layering can be computed in linear time, e.g. by longest path layering [14]. From there, we can easily construct a proper layering by introducing new nodes for all edges $e = (u, v) \in A$ that span more than one layer. If, for example, $u \in L_i$, $v \in L_j$ and $j > i + 1$, we introduce $j - i - 1$ new nodes $v_{e,1}, \ldots, v_{e,j-i-1}$ and replace e by the path $(u, v_{e,1}, v_{e,2}, \ldots, (v_{e,j-i-1}, v))$. The weights of the new edges are set to $w'(u, v_{e,1}) = w(e)$ and $w'(e') = 0$ for all other introduced edges e'. An example of the transformation of a layered graph to a properly layered graph and for a mapping of paths in one graph to the other graph can be seen in Fig. 4.



(a) Two edge-disjoint paths in a layered acyclic graph



(b) The corresponding paths in the properly layered graph constructed by our algorithm

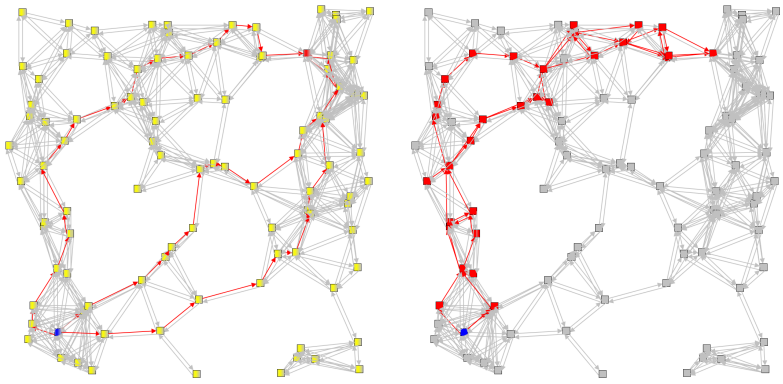
Fig. 4. An example for the transformation of layered graphs to properly layered graphs and corresponding paths

Combining the algorithms we can derive an algorithm for general acyclic graphs. Given an acyclic graph $D = (V, A)$ with n nodes and m edges, we first compute a layering. Then we compute in time $O(nm)$ a properly layered graph $D' = (V', A')$ with $O(mn)$ nodes and edges. Applying the algorithm for properly layered graphs to D' we can compute k edge-disjoint minimum-energy paths in D' in time $O(|A'|^k) = O(m^k n^k)$ and with space in $O(|V'|^k) = O(m^k n^k)$. Finally, we have to find the corresponding paths in D, which can be done in linear time (given appropriate data structures, e. g. pointers from the edges in A' to the corresponding edges in A). For a fixed $k \in \mathbb{N}$ we can thus find k minimum-energy edge-disjoint paths in polynomial time.

6 A Heuristic for General Graphs

Most graphs that arise of real-world networks are not acyclic, e. g. if we assume symmetry of our weights $w(u, v)$ and the maximum energy of the nodes is equal we get a symmetric graph. However, we can apply our algorithm for acyclic graphs to derive a heuristic for the general case: In the first step we compute an appropriate acyclic subgraph and use our exact algorithm in the acyclic subgraph. One natural way of doing this assumes that the coordinates of the nodes are known (i. e. we have information about the geometry of the network). Then we can just remove any edge whose end point is further from the target than the starting point (in terms of euclidean distance). Edges adjacent to s are treated differently: All edges leaving s remain in the graph, whereas edges leading to s are removed.

We did some experiments with graphs of different sizes and randomly created layouts. We placed nodes uniformly at random in a square of a given size and computed the LDMW paths in the original graph and the exact solution in the acyclic subgraph. We assumed that the enrgy (i.e., the edge lengths) depend only on the euclidian distances of nodes (i. e. we used the network model of [1]). Due to the high running time and memory requirements of the algorithm we could only make comparisons for $k = 3$. They showed that our heuristic usually outperformed the LDMW algorithm. Energy savings were up to 40 % and the average was between 10 % and 15 %, depending on the "density" of the graph. We also found that removing edges in order to get an acyclic graph did not decrease the number of edge-disjoint s-t-paths dramatically. In summary we could show that the paths found by the LDMW approximation algorithm usually are far from optimal. Thus it would be worth searching for better approximation algorithms.



(a) three paths with $\mathcal{E}(P) = 415381$ (b) three paths with $\mathcal{E}(P) = 359295$

Fig. 5. Comparison between the LDMW heuristc (left) and acyclic graph heuristic (right)

7 Conclusion

We have seen that MEEP is NP-complete in the general case where k is not bounded (but part of the input). The complexity of MEEP for a fixed $k \in \mathbb{N}$ is still unknown. If we restrict our problem to graphs of equal edge weights, there remains a small gap between the $\Omega(\sqrt{k})$ lower bound and the $2\sqrt{k}$ upper bound for $k > 6$ for the approximation factor of the LDMW algorithm. It is also worth searching for better approximation algorithms, as we are still far away from the theoretical lower bound of around $\log(k)$. And there is still no satisfying (heuristic or approximative) distributed algorithm for finding energy-optimal disjoint paths.

References

1. Srinivas, A. and Modiano, E.: Minimum Energy Disjoint Path Routing in Wireless Ad-Hoc Networks. In: Proc. Int. Conf. on Mobile Computing and Networking, Mobicom'03, ACM Press (2003) 122–133
2. Singh, S., Woo, M., and Raghavendra, C.S.: Power-Aware Routing in Mobile Ad Hoc Networks. In: Proc. Int. Conf. on Mobile computing and networking, MobiCom'98, ACM Press (1998) 181–190
3. Suurballe J.W.: Disjoint Paths in a Network. Networks **4** (1974) 125–145
4. Ganesan, D., Govindan, R., Shenker, S., and Estrin, D.: Highly-Resilient, Energy-Efficient Multipath Routing in Wireless Sensor Networks. SIGMOBILE Mob. Comput. Commun. Rev. **5** 4 (2001) 11–25
5. Nasipuri, A. and Das, S.: On-Demand Multipath Routing for Mobile Ad Hoc Networks. In: Proc. Int. Conf. on Computer Communications and Networks, ICCCN'99 (1999) 64–70
6. Chen, W.T., Huang, N.F.: The Strongly Connecting Problem on Multihop Packet Radio Networks. IEEE Transactions on Communications **37**(3) (1989) 293–295
7. Lloyd, E.L., Liu, R., Marathe, M.V., Ramanathan, R., and Ravi, S.S.: Algorithmic Aspects of Topology Control Problems for Ad Hoc Networks. Mob. Netw. Appl. **10**(1-2) (2005) 19–34
8. Kirousis, L.M., Kranakis, E., Krizanc, D., and Pelc, A.: Power Consumption in Packet Radio Networks (Extended abstract). In: Proc. Symp. on Theoretical Aspects of Computer Science, STACS'97, Springer-Verlag (1997) 363–374
9. Wieselthier, J.E., Nguyen, G.D., and Ephremides, A.: Energy-Efficient Broadcast and Multicast Trees in Wireless Networks. Mob. Netw. Appl. **7**(6) (2002) 481–492
10. Clementi, A., Huiban, G., Penna, P., Rossi, G., and Verhoeven, Y.: Some Recent Theoretical Advances and Open Questions on Energy Consumption in Ad-Hoc Wireless Networks. In: Proc. Workshop on Approximation and Randomization Algorithms in Communication Networks, ARACNE (2002) 23–38
11. Tang, J. and Xue, G.: Node-Disjoint Path Routing in Wireless Networks: Tradeoff between Path Lifetime and Total Energy. In: Proc. IEEE International Conference on Communications **7** (2004) 3812–3816
12. Papadimitriou, C.H.: Computational Complexity. Addison-Wesley (1995)
13. Feige, U.: A Threshold of $\ln n$ for Approximating Set Cover. J. ACM **45** 4 (1998) 634–652
14. di Battista, G., Eades, P., Tamassia, R., and Tollis, I.G.: Graph Drawing: Algorithms for the Visualization of Graphs. Prentice Hall (1999)