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

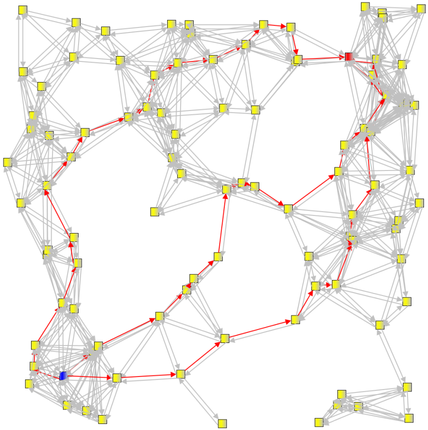
Markus Maier[1] *Steffen Mecke*[2] Dorothea Wagner[2]

[1]Max Planck Institute for Biological Cybernetics
Tübingen, Germany

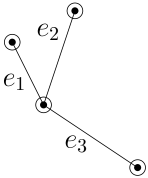
[2]Faculty of Informatics, Institute for Theoretical Informatics
Universität Karlsruhe (TH), Germany

SOFSEM 2007
January 24th, 2007
Harrachov, Czech Republic

Disjoint Shortest Paths



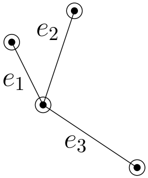
Wireless Multicast Advantage (WMA)

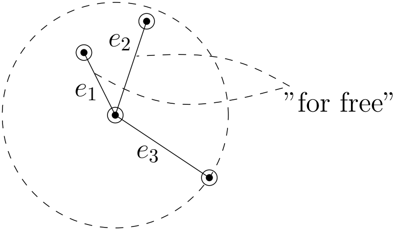


Classic:

$$W = w(e_1) + w(e_2) + w(e_3)$$

Wireless Multicast Advantage (WMA)





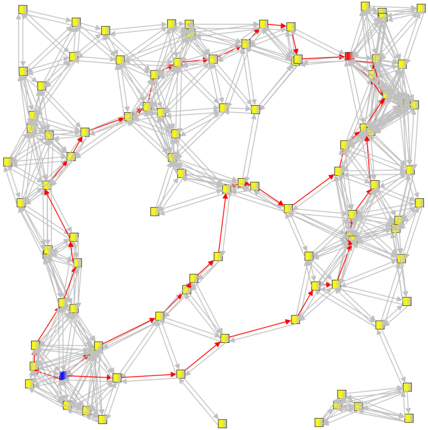
Classic:

$$W = w(e_1) + w(e_2) + w(e_3)$$

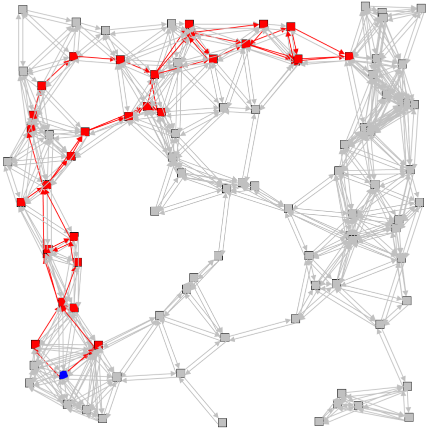
WMA:

$$\mathcal{E} = max\{w(e_1), w(e_2), w(e_3)\}$$

Disjoint Shortest Paths



Disjoint Shortest Paths



Overview

Previous Work

- *J.W. Suurballe: Disjoint paths in a network, 1974.*
  - k node-disjoint paths with minimum weight by k shortest path searches
  - can be extended to find edge disjoint paths
- Various: Minimum Range Assignment problems
  - assign transmission range to nodes such that the resulting graph has certain properties (connected, k-connected) etc.
- *A. Srinivas, E. Modiano: Minimum Energy Disjoint Path Routing in Wireless Ad-hoc Networks, 2003.*
  - min. energy disjoint paths with WMA
  - polynomial algorithm for node-disjoint case
  - polynomial algorithm for edge-disjoint paths and $k = 2$
  - three heuristics for $k > 2$

Network Model

- (directed) graph $D = (V, A)$ with edge weights $w(u, v)$
- weight of a set of paths P:

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

- energy of a set of paths P:

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

MEEP

Problem (Minimum Energy Edge-disjoint Paths)

- *Given:*
  - $D = (V, A), w : A \to \mathbb{R}^+$
  - $s, t \in V$
  - $k \geq 1$
  - $B > 0$
- *Are there k edge-disjoint paths P from s to t with $\mathcal{E}(P) \leq B$?*

NP-Completeness

Theorem

MEEP is NP-complete.

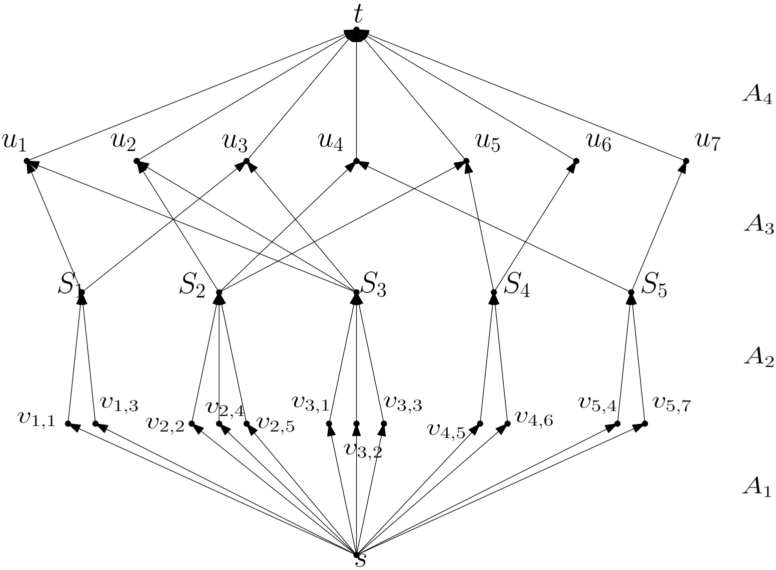
Proof.

Reduction from SET COVER. □

SET COVER

Definition (SET COVER)

- Given
  - a set $U = \{u_1, \ldots, u_k\}$,
  - 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?



B subsets cover U

$\iff$

there are $k := |U|$ edge-disjoint paths with $\mathcal{E}(P) = 1 + k + B + k$.

More on NP-Completeness

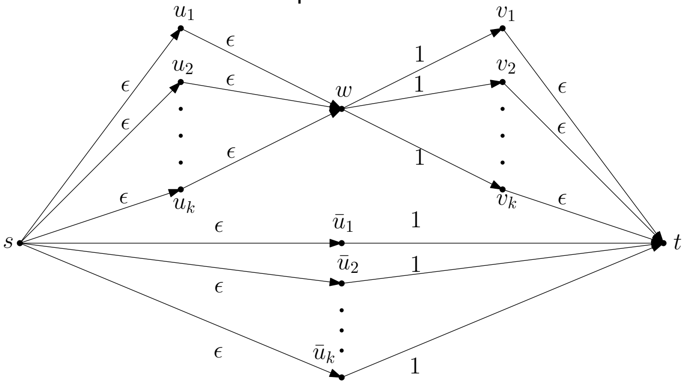
- holds also for binary case (edge weights $\equiv 1$)
- using result of Feige: no approximation algorithm for MEEP with appr. factor better than $(1 - o(1)) \ln k$ (for arbitrary edge weights)

A Lower Bound for a Known Algorithm

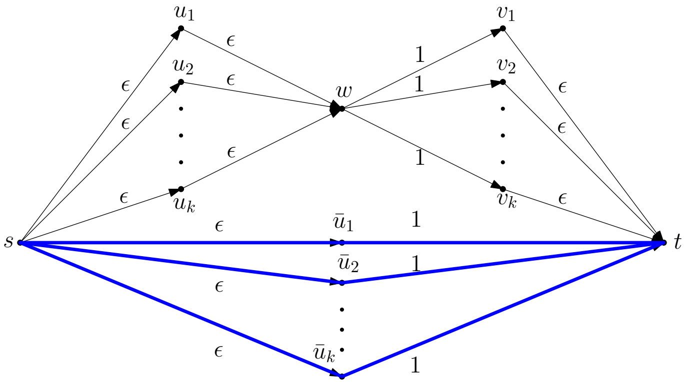
- LDMW from [SriMod03]: The k minimum-*weight* paths are a k-approximation for the k minimum-*energy* paths.

A Lower Bound for a Known Algorithm

- LDMW from [SriMod03]: The k minimum-*weight* paths are a k-approximation for the k minimum-*energy* paths.
- Here: this bound is sharp:

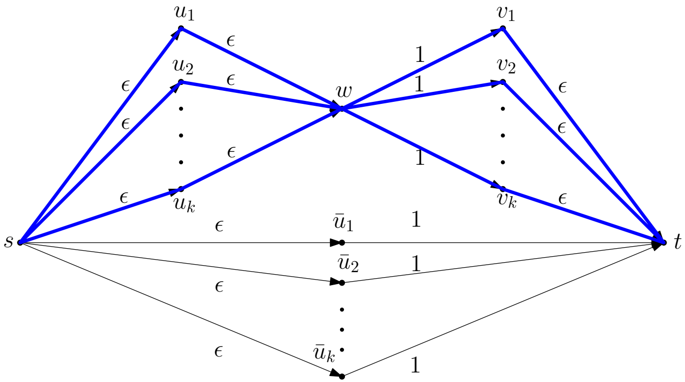


A Lower Bound for a Known Algorithm



- Minimum weight paths: $\mathcal{E}(P') = k + k\varepsilon$

A Lower Bound for a Known Algorithm



- Minimum weight paths: $\mathcal{E}(P') = k + k\varepsilon$
- Minimum energy paths: $\mathcal{E}(P^*) = 1 + 3k\varepsilon$

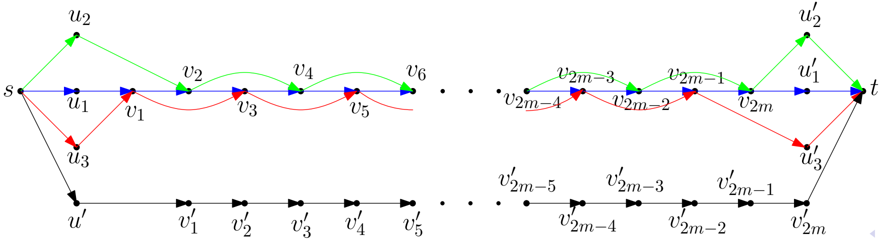
The Binary Case

- B(inary)MEEP: all edge weights are 1
- Two results:
  - The approx. factor of LDMW for BMEEP is in
    $$\Omega(\sqrt{k}) \ .$$
  - The approx. factor of LDMW for BMEEP is
    $$\leq 2\sqrt{k} \qquad (\text{for } k \geq 6) \ .$$
- For $k = 3$ the factor is exactly 3.

The Lower Bound

Theorem

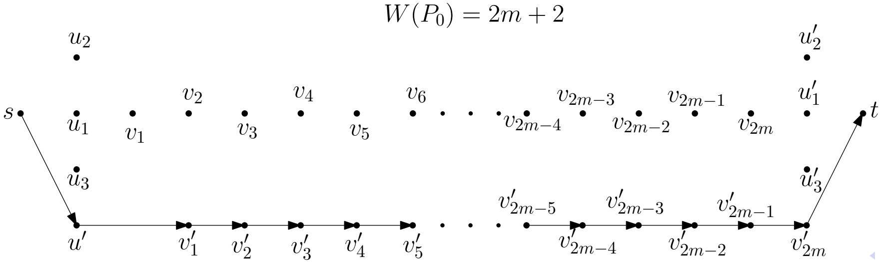
The approx. factor of LDMW is $\Omega(\sqrt{k})$ *(in the binary case).*



The Lower Bound

Theorem

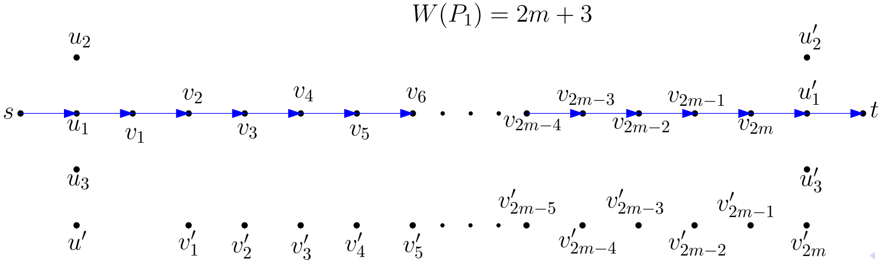
The approx. factor of LDMW is $\Omega(\sqrt{k})$ (in the binary case).



The Lower Bound

Theorem

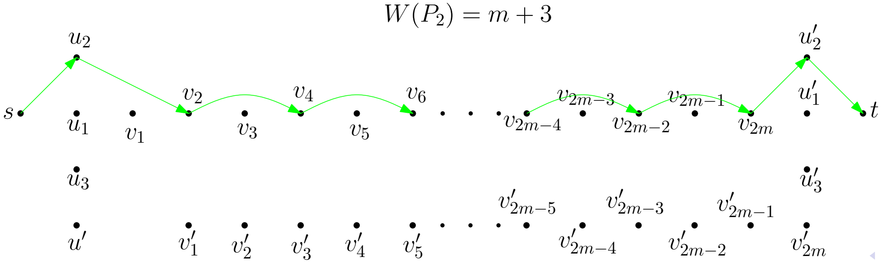
The approx. factor of LDMW is $\Omega(\sqrt{k})$ *(in the binary case).*



The Lower Bound

Theorem

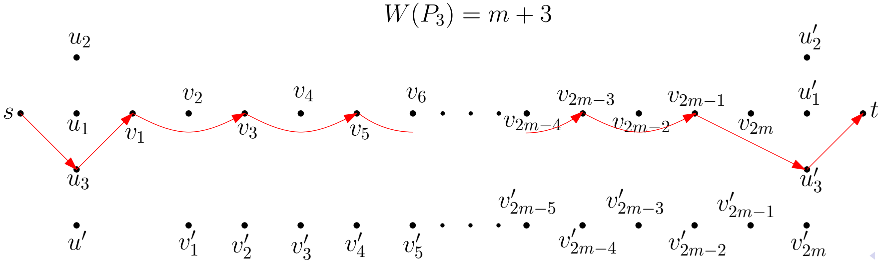
The approx. factor of LDMW is $\Omega(\sqrt{k})$ (in the binary case).



The Lower Bound

Theorem

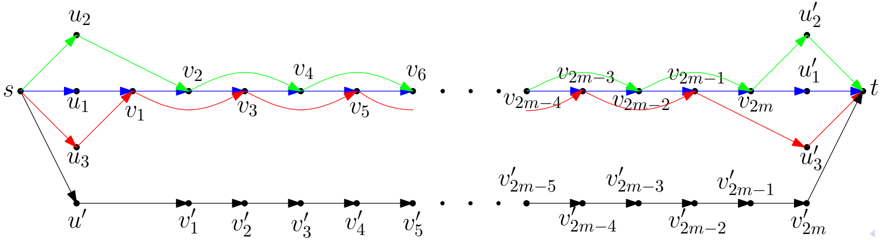
The approx. factor of LDMW is $\Omega(\sqrt{k})$ *(in the binary case).*



The Lower Bound

Theorem

The approx. factor of LDMW is $\Omega(\sqrt{k})$ *(in the binary case).*

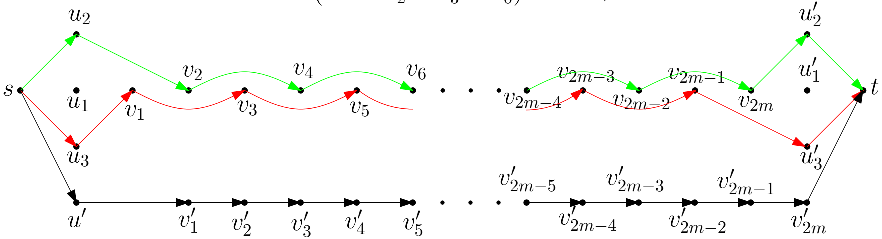


The Lower Bound

Theorem

The approx. factor of LDMW is $\Omega(\sqrt{k})$ (in the binary case).

$$\mathcal{E}(P' = P_2 \cup P_3 \cup P_0) = 4m + 6$$



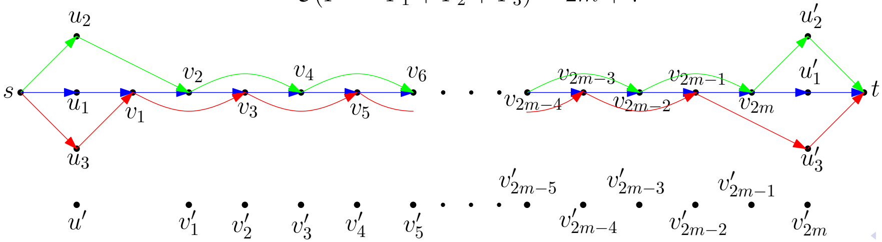
Minimum weight solution

The Lower Bound

Theorem

The approx. factor of LDMW is $\Omega(\sqrt{k})$ *(in the binary case).*

$$\mathcal{E}(P^* = P_1 + P_2 + P_3) = 2m + 7$$



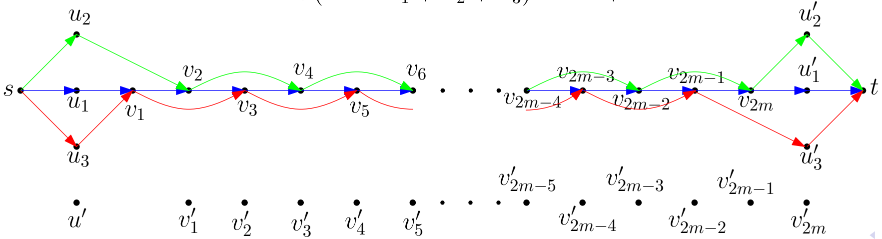
Minimum energy solution

The Lower Bound

Theorem

The approx. factor of LDMW is $\Omega(\sqrt{k})$ (in the binary case).

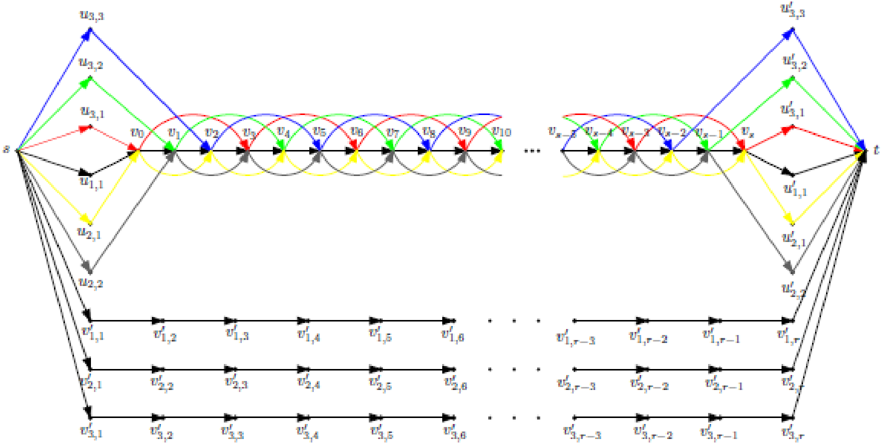
$$\mathcal{E}(P^* = P_1 + P_2 + P_3) = 2m + 7$$



$$\mathcal{E}(P') \geq 2\mathcal{E}(P^*)$$

The Lower Bound

Theorem
The approx. factor of LDMW is $\Omega(\sqrt{k})$ *(in the binary case).*



The Upper Bound

Theorem
The approx. factor of LDMW is $\leq 2\sqrt{k}$ *(in the binary case).*

Proof.

The Upper Bound

Theorem

The approx. factor of LDMW is $\leq 2\sqrt{k}$ *(in the binary case).*

Proof.

- P^*: the k minimum energy paths
- P': the k minimum weight paths

The Upper Bound

Theorem

The approx. factor of LDMW is $\leq 2\sqrt{k}$ (in the binary case).

Proof.

- P^*: the k minimum energy paths
- P': the k minimum weight paths
- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$

The Upper Bound

Theorem
The approx. factor of LDMW is $\leq 2\sqrt{k}$ *(in the binary case).*

Proof.

- P^*: the k minimum energy paths
- P': the k minimum weight paths
- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$
- $W(P^*)$ is number of edges, $\mathcal{E}(P^*)$ is (number of nodes $- 1$).

The Upper Bound

Theorem
The approx. factor of LDMW is $\leq 2\sqrt{k}$ (in the binary case).

Proof.

- P^*: the k minimum energy paths
- P': the k minimum weight paths
- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$
- $W(P^*)$ is number of edges, $\mathcal{E}(P^*)$ is (number of nodes $-$ 1).
- P^* is a DAG. Sort nodes such that edges go "to the right".

The Upper Bound

Theorem
The approx. factor of LDMW is $\leq 2\sqrt{k}$ *(in the binary case).*

Proof.

- P^*: the k minimum energy paths
- P': the k minimum weight paths
- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$
- $W(P^*)$ is number of edges, $\mathcal{E}(P^*)$ is (number of nodes $-$ 1).
- P^* is a DAG. Sort nodes such that edges go "to the right".
- Every path goes "to the right" and crosses a "distance" of $n - 1$ from s to t.

The Upper Bound

Theorem

The approx. factor of LDMW is $\leq 2\sqrt{k}$ *(in the binary case).*

Proof.

- P^*: the k minimum energy paths
- P': the k minimum weight paths
- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$
- $W(P^*)$ is number of edges, $\mathcal{E}(P^*)$ is (number of nodes $-$ 1).
- P^* is a DAG. Sort nodes such that edges go "to the right".
- Every path goes "to the right" and crosses a "distance" of $n - 1$ from s to t.
- Total distance crossed by all paths is $k(n-1)$.
- At most $n-1$ edges of length 1, $n-2$ edges of length 2, ...

The Upper Bound

Proof.

- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$
- $W(P^*)$ is number of edges, $\mathcal{E}(P^*)$ is (number of nodes $-$ 1).
- Total distance crossed by all paths is $k(n-1)$.
- Distance crossed by edges of length 1 to $2\sqrt{k}$ is

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

$$\geq k(n-1)$$

The Upper Bound

Proof.

- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$
- $W(P^*)$ is number of edges, $\mathcal{E}(P^*)$ is (number of nodes $-$ 1).
- Total distance crossed by all paths is $k(n-1)$.
- Distance crossed by edges of length 1 to $2\sqrt{k}$ is

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

$$\geq k(n-1)$$

The Upper Bound

Proof.

- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$
- $W(P^*)$ is number of edges, $\mathcal{E}(P^*)$ is (number of nodes $-$ 1).
- Total distance crossed by all paths is $k(n-1)$.
- Distance crossed by edges of length 1 to $2\sqrt{k}$ is

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

The Upper Bound

Proof.

- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$
- $W(P^*)$ is number of edges, $\mathcal{E}(P^*)$ is (number of nodes $-$ 1).
- Total distance crossed by all paths is $k(n-1)$.
- Distance crossed by edges of length 1 to $2\sqrt{k}$ is $\geq k(n-1)$.

The Upper Bound

Proof.

- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$
- $W(P^*)$ is number of edges, $\mathcal{E}(P^*)$ is (number of nodes $-$ 1).
- Total distance crossed by all paths is $k(n-1)$.
- Distance crossed by edges of length 1 to $2\sqrt{k}$ is $\geq k(n-1)$.
- *Number* of used edges is at most

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

$$\leq 2\sqrt{k}(n-1) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*) \ .$$

The Upper Bound

Proof.

- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$
- $W(P^*)$ is number of edges, $\mathcal{E}(P^*)$ is (number of nodes $-$ 1).
- Total distance crossed by all paths is $k(n-1)$.
- Distance crossed by edges of length 1 to $2\sqrt{k}$ is $\geq k(n-1)$.
- *Number* of used edges is at most

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

$$\leq 2\sqrt{k}(n-1) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*) \quad .$$

The Upper Bound

Proof.

- Need to show: $\mathcal{E}(P') \leq W(P') \leq W(P^*) \leq 2\sqrt{k} \cdot \mathcal{E}(P^*)$
- $W(P^*)$ is number of edges, $\mathcal{E}(P^*)$ is (number of nodes $-$ 1).
- Total distance crossed by all paths is $k(n-1)$.
- Distance crossed by edges of length 1 to $2\sqrt{k}$ is $\geq k(n-1)$.
- *Number* of used edges is at most

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

□

A (Exact, Polynomial) Algorithm for Acyclic Graphs

1. compute a layering (time $O(mn)$)
2. compute a properly layered graph with $O(mn)$ nodes and edges (time $O(mn)$)
3. compute k minimum energy paths by dynamic programming (time and space $O(m^k n^k)$)
4. find corresponding paths in original graphs (linear time)

Algorithm for Properly Layered Graphs

dynamic programming: for every layer maintain table with one entry for every (unordered) k-tuple of nodes

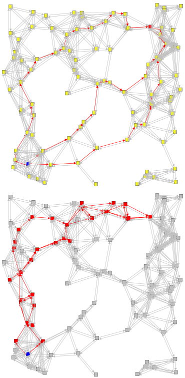
1. for every layer i from 1 to h do
2.   for every k-tuple of edges from layer i to $i+1$
3.   let $(u_1, v_1), \ldots, (u_k, v_k)$ be the current tuple
4.   update $\mathcal{E}_{\min}(v_1, \ldots, v_k)$

Heuristics for General Graphs

- Given: An arbitrary graph with node positions
- Create directed acyclic graph from G by removing edges (u, v) where $dist(v, t) > dist(u, t)$
- Apply algorithm for directed acyclic graphs
- For randomly placed nodes and an energy metric ($w(u, v) = dist(u, v)^2$) this heuristic usually outperforms LDMW.

Conclusion

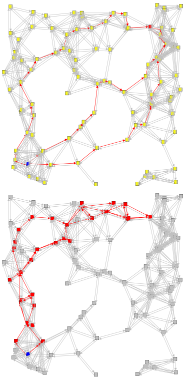
- Results:
  - NP-completeness
  - sharp bounds for minimum-weight solution
  - algorithm for acyclic graphs and
  - heuristics for geometric graphs
- Open:
  - complexity status for fixed $k > 2$
  - close the gap between "ln k" and "$\sqrt{k}$"
  - better heuristics (use geometry?)
  - distributed implementation?



Thank you!

Conclusion

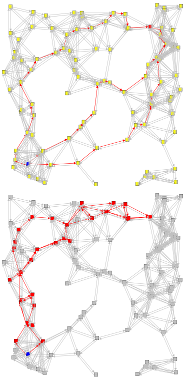
- Results:
  - NP-completeness
  - sharp bounds for minimum-weight solution
  - algorithm for acyclic graphs and
  - heuristics for geometric graphs
- Open:
  - complexity status for fixed $k > 2$
  - close the gap between "$\ln k$" and "$\sqrt{k}$"
  - better heuristics (use geometry?)
  - distributed implementation?



Thank you!

Conclusion

- Results:
  - NP-completeness
  - sharp bounds for minimum-weight solution
  - algorithm for acyclic graphs and
  - heuristics for geometric graphs
- Open:
  - complexity status for fixed $k > 2$
  - close the gap between "ln k" and "$\sqrt{k}$"
  - better heuristics (use geometry?)
  - distributed implementation?



Thank you!