Utrecht University

Disjoint Paths and Directed Steiner Tree on Planar Graphs with Terminals on Few Faces

Master’s Thesis in Computing Science

Hilde Verbeek

27 July 2022

Introduction

- Directed Steiner Tree
- Disjoint Paths
- Terminals on few faces

Fixed-parameter tractability and face cover number

Fixed-parameter tractability and face cover number

Parameterized algorithms

- NP-hard problems: exponential running time
- Finding exact solutions is impractical
- Solution: parameterized algorithms

Fixed-parameter tractability and face cover number

Parameterized algorithms

- Parameterized algorithm: runs in polynomial time if an input parameter k is fixed
- Allows one to design efficient algorithms if the parameter is bounded by a constant
- **FPT:** $f(k) \cdot \mathrm{poly}(n)$ time
  (*fixed-parameter tractable*)
- **XP:** $n^{f(k)}$ time
  (*slicewise polynomial*)

Fixed-parameter tractability and face cover number

Parameterized algorithms

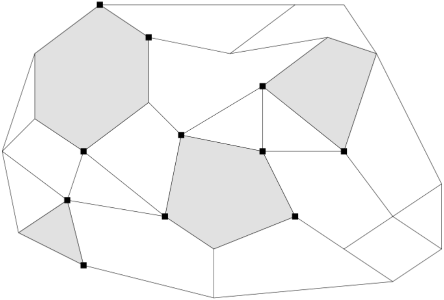
Examples:

- VERTEX COVER: $O(2^k \cdot n)$ time
  for a vertex cover of $\leq k$ vertices
- STEINER TREE: $O(2^k \cdot k^2 \cdot n^2)$ time
  on inputs with k terminals
- DOMINATING SET: $3^w \cdot w^{O(1)} \cdot n$ time
  on graphs of treewidth $\leq w$

Fixed-parameter tractability and face cover number

Face cover number of terminals

For a planar graph G and terminals $K \subseteq V(G)$, the *face cover number* $\gamma(G, K)$ is the number of faces needed to cover all vertices in K.



Fixed-parameter tractability and face cover number

Face cover number of terminals

- Computing a face cover of $\leq k$ faces: $c^k \cdot n$ time (FPT)[1]
  (for some constant c)
- $\gamma(G, K) \leq |K|$
- Useful in some applications

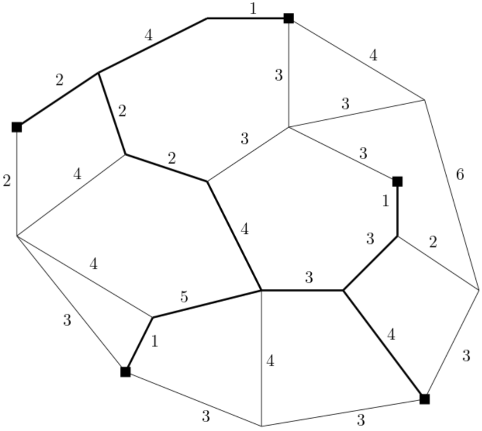
[1] Daniel Bienstock and Clyde L Monma. “On the complexity of covering vertices by faces in a planar graph”. In: *SIAM Journal on Computing* 17.1 (1988), pp. 53–76.

Directed Steiner Tree

Directed Steiner Tree

Introduction

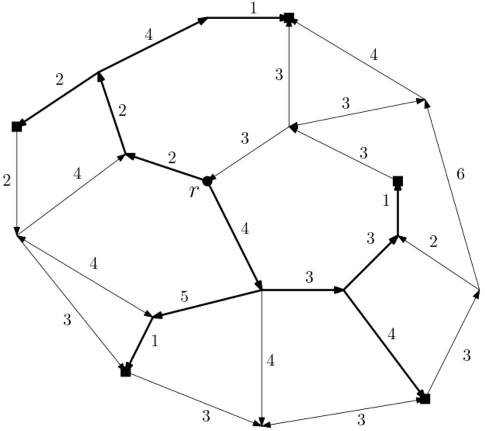
STEINER TREE: given weighted graph G and terminals $K \subseteq V(G)$, return minimum-weight tree connecting all terminals in K.



Directed Steiner Tree

Introduction

Directed Steiner Tree: graph is directed and contains root *r*; output tree should be rooted in *r* as well.



Directed Steiner Tree

Introduction

This section: two STEINER TREE algorithms, with parameterization by the face cover number on planar graphs.

Authors	General	Few faces
Dreyfus and Wagner	$O(2^k \cdot k^2 \cdot n^2)$[2]	$n^{O(k)}$[3]
Kisfaludi-Bak et al.	-	$2^{O(k)} \cdot n^{O(\sqrt{k})}$[4]

Both algorithms on undirected graphs originally, adapted by me.

[2] Stuart E Dreyfus and Robert A Wagner. "The Steiner problem in graphs". In: *Networks* 1.3 (1971), pp. 195–207.

[3] Marshall Bern. "Faster exact algorithms for Steiner trees in planar networks". In: *Networks* 20.1 (1990), pp. 109–120.

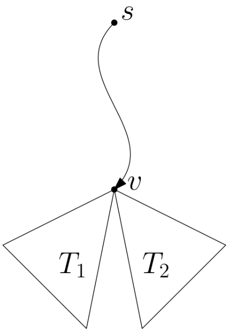
[4] Sándor Kisfaludi-Bak, Jesper Nederlof, and Erik Jan van Leeuwen. "Nearly ETH-tight algorithms for planar Steiner tree with terminals on few faces". In: *ACM Transactions on Algorithms (TALG)* 16.3 (2020), pp. 1–30.

Directed Steiner Tree

Based on Dreyfus-Wagner

Observe in the structure of a Steiner Tree on terminals K and root s:

- s: root
- v: first branching vertex
- P: shortest $s - v$ path
- T_1: min Steiner tree with terminals K_1 and root v
- T_2: min Steiner tree with terminals K_2 and root v



(barring edge cases $v = r$ and $v \in K$)

Directed Steiner Tree

Based on Dreyfus-Wagner

Basic idea: compute Steiner trees for all subsets of terminals and all root vertices using dynamic programming.

For every $D \subseteq K$ and $s \in V(G)$:
$A[D, s] =$ weight of minimum Steiner tree on terminals D and root s.

Then the final answer is simply $A[K, r]$.

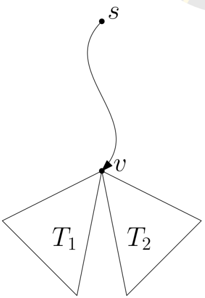
Directed Steiner Tree

Based on Dreyfus-Wagner

Recall the Steiner tree structure.

To compute $A[D, s]$:

- $w(P) = \text{dist}(s, v)$
- $w(T_1) = A[K_1, v]$
- $w(T_2) = A[K_2, v]$



By minimizing on D' and v:

$$A[D, s] = \min_{\substack{v \in V(G) \\ \emptyset \subset D' \subset D}} \{\text{dist}(s, v) + A[D', v] + A[D \setminus D', v]\}$$

Directed Steiner Tree

Based on Dreyfus-Wagner

$$A[D, s] = \min_{\substack{v \in V(G) \\ \emptyset \subset D' \subset D}} \{\mathrm{dist}(s, v) + A[D', v] + A[D \setminus D', v]\}$$

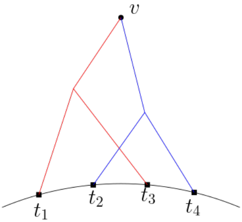
- Base case: $A[\{v\}, s] = \mathrm{dist}(s, v)$
- Compute for subsets of K of increasing sizes
- Final answer is $A[K, r]$
- Running time: $O(n3^k + n^2 2^k + n^2 \log n + nm)$ $(= O(3^k \cdot n^3))$
- Improved: $O(2^k \cdot k^2 \cdot n^2)$

  (using fast subset convolution and batching)

Directed Steiner Tree

Dreyfus-Wagner on few faces

Intuition: do not combine subtrees when they intersect, if their terminals are on the same face.



Directed Steiner Tree

Dreyfus-Wagner on few faces

- When terminals are on the same face, only consider intervals of those terminals.
- For p terminals: 2^p subsets, but only $O(p^2)$ subintervals.
- When selecting subsets $D \subseteq K$ or $D' \subseteq D$, ensure terminals of each face are on an interval.
- $n^{O(k)}$ time when k is the number of terminal faces.

Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Second algorithm: separating the graph by guessing some vertices, maintaining a balance between the terminal faces.

This section: summary of the algorithm, and how to adapted to directed graphs.

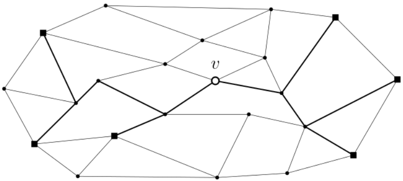
Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Consider a simple branching algorithm for terminals K:

1. Enumerate possibilities of $v \in V(G)$ and $K' \subset K$
2. Recurse on inputs $K' \cup \{v\}$ and $(K \setminus K') \cup \{v\}$ and return option with minimum total weight

Can something like this be applied to terminal faces?

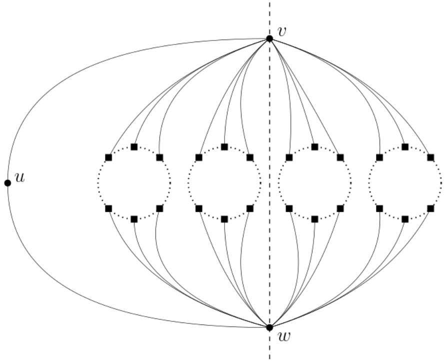


Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Problem: some faces end up in both subproblems.

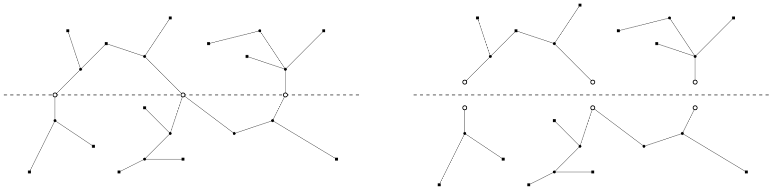
Solution: separate in multiple vertices.



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Separating in multiple vertices yields forests.



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

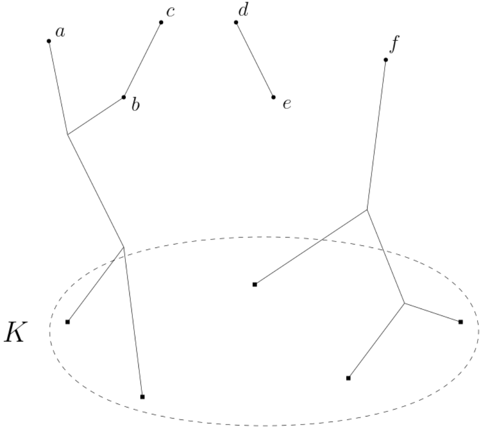
- Block: vertices B, encoding a component containing vertices B and maybe some terminals
- Block Steiner forest: for terminals $K \subseteq V(G)$ and blocks $\pi = \{B_1, \ldots, B_p\}$, a forest of which components are encoded by π and every terminal in K is connected to one component
- BSF with one block equals a Steiner tree

Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Example:

$\pi = \{\{a, b, c, \}, \{d, e\}, \{f\}\}$ and terminals K



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

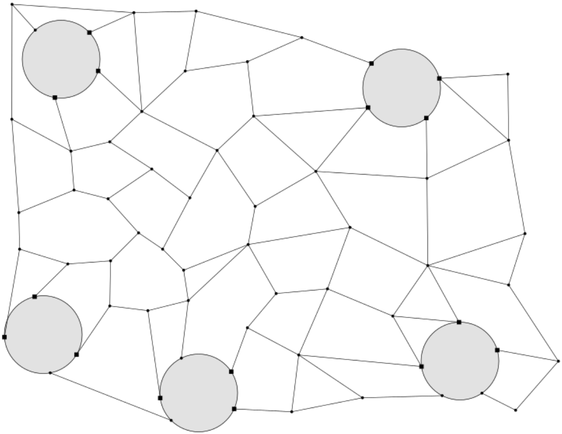
Algorithm:

- with few blocks and few terminal faces, divide terminals over blocks and use Dreyfus-Wagner
- otherwise: choose a separator, split the blocks and faces, recurse and take the optimal solution

Directed Steiner Tree

Based on Kisfaludi-Bak et al.

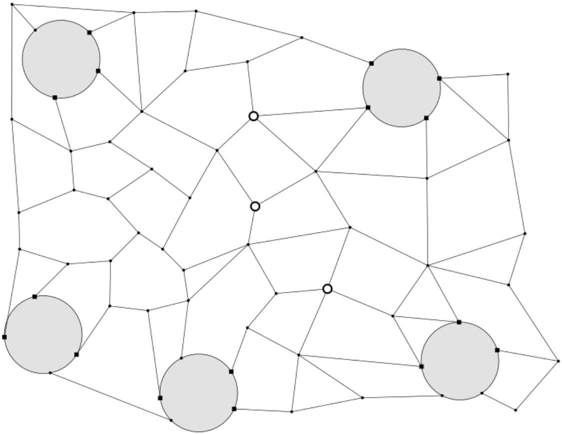
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

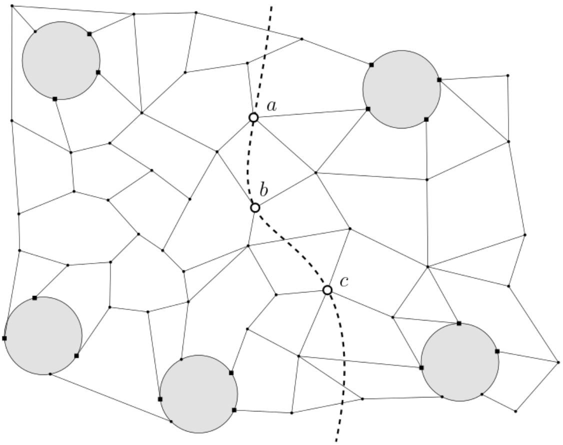
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

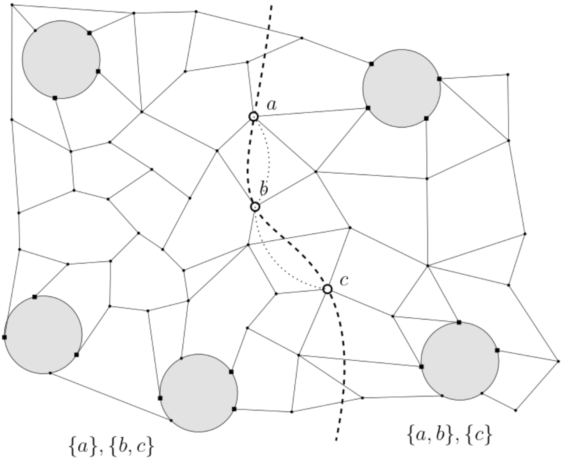
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

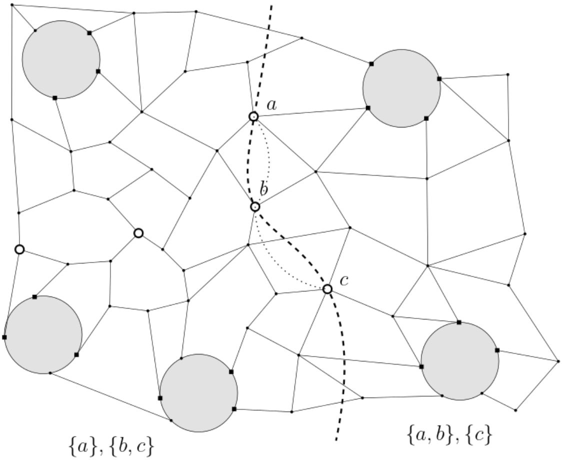
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

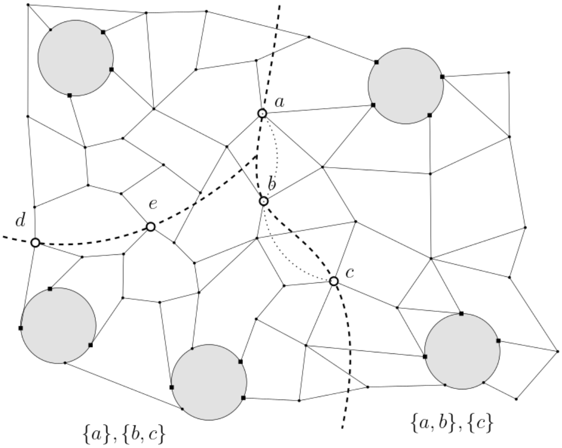
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

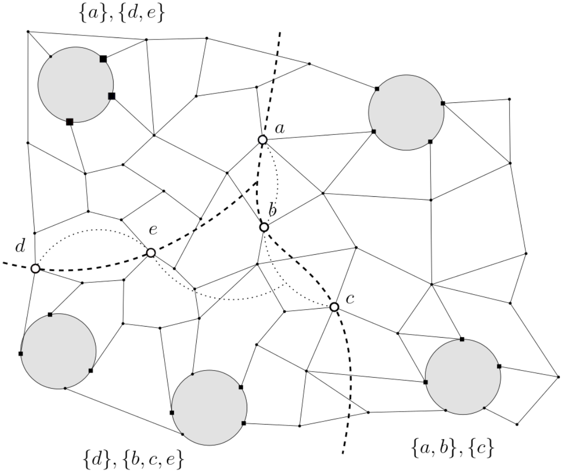
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

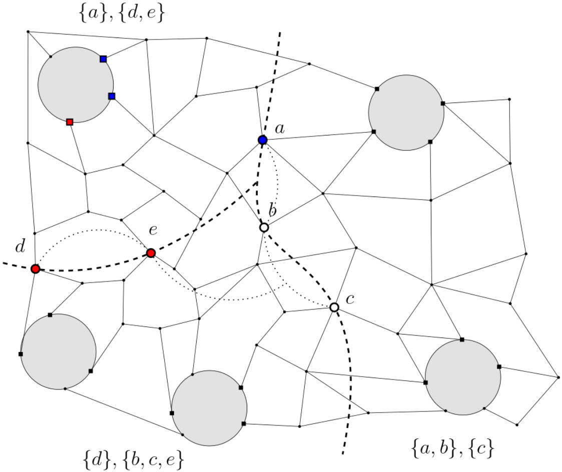
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

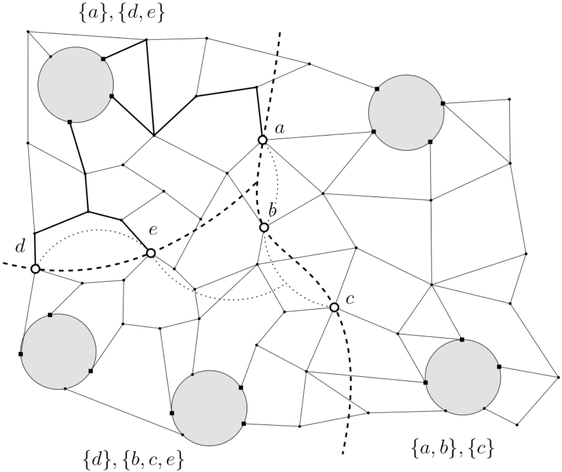
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

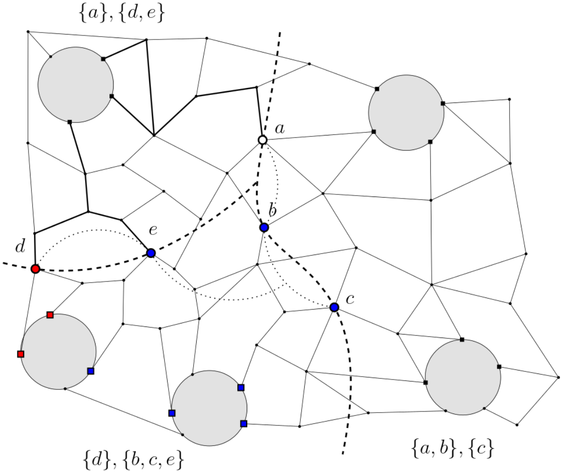
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

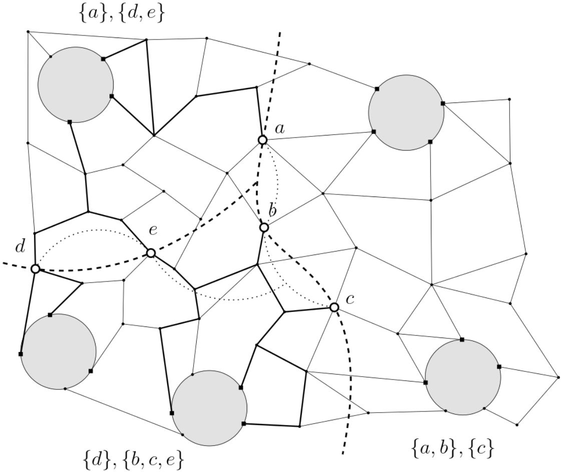
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

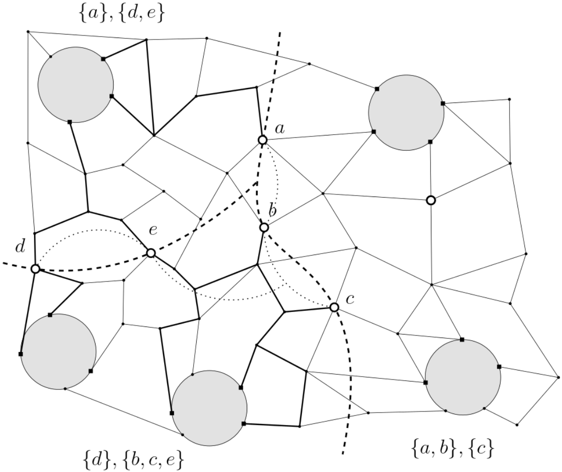
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

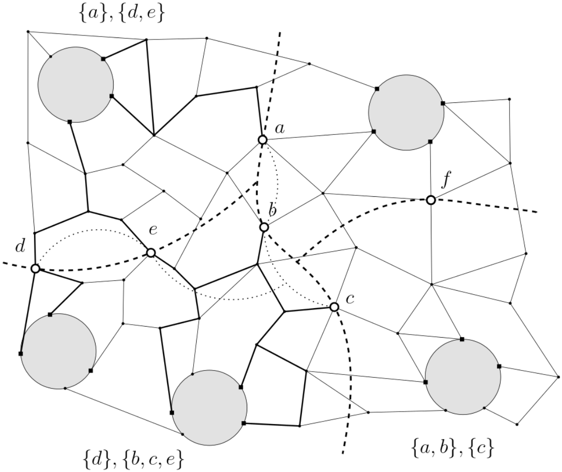
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

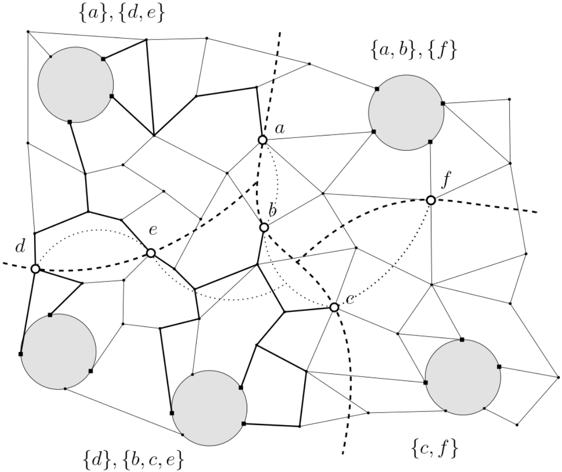
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

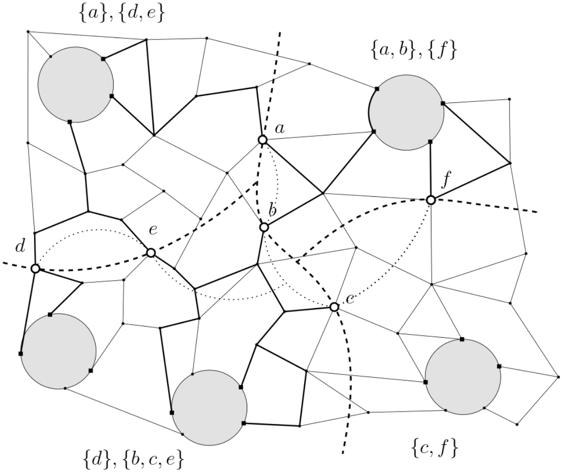
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

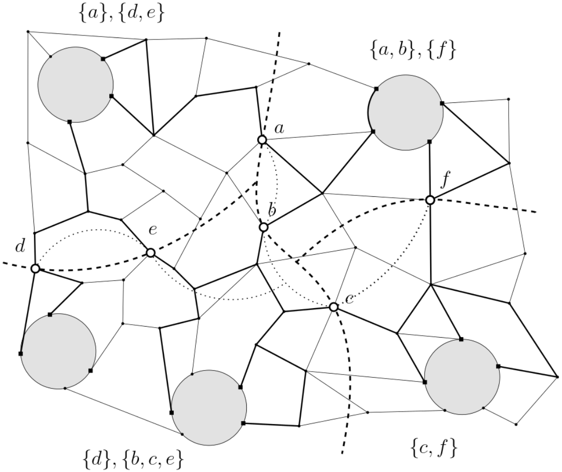
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

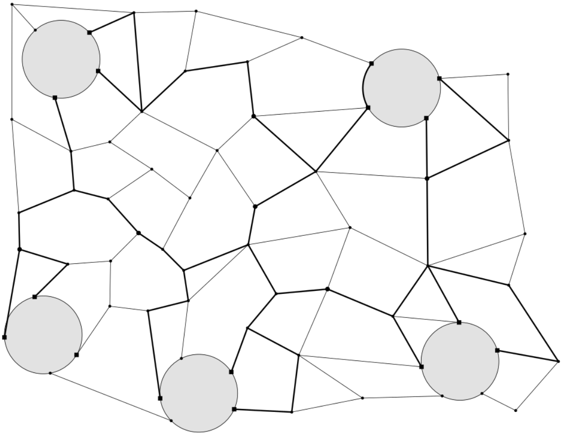
Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Demonstration:



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

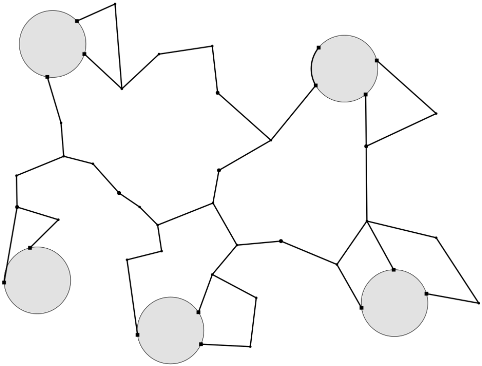
How large should the separator be?

Claim: only $O(\sqrt{k})$ vertices are needed to create a balanced separation between the terminal faces, if k is the number of terminal faces.

Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Observe the graph H which is the union of a solution block Steiner forest and the terminal faces:



Directed Steiner Tree

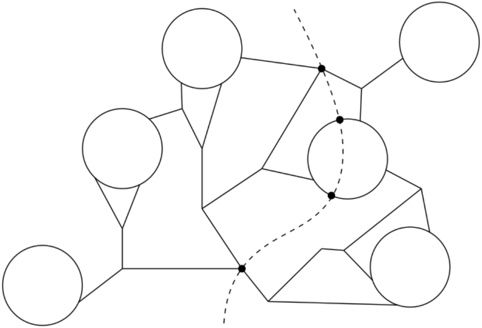
Based on Kisfaludi-Bak et al.

- Every face of H is either a terminal face or adjacent to one
- In other words, the dual graph H^* has a dominating set of size k
- DS of size $k \implies tw(H^*) \leq 15\sqrt{k}$
- $\text{tw}(H^*) \leq 15\sqrt{k} + 1 \implies \text{tw}(H) \leq 15\sqrt{k} + 1$
- $\text{tw}(H) \leq 15\sqrt{k} + 1 \implies H$ contains a balanced separation of size $\leq 15\sqrt{k} + 2$

Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Messy detail: some terminal faces are intersected by the separator. This is only the case for $O(\sqrt{k})$ faces, though.



Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Base case: when the blocks have b vertices and $k + b \leq c_0$ (constant), for every terminal face F:

- enumerate all $n^{O(b)}$ assignments of F's terminals to the b blocks
- for every assignment and every block, solve using Dreyfus-Wagner for the block and its assigned terminals
- take the result of minimum weight

Because there are $n^{O(bk)}$ assignments and Dreyfus-Wagner takes $n^{O(k+b)}$ time, and both parameters are bounded by a constant, this runs in polynomial time!

Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Algorithm:

Input: graph G, k terminal faces K, blocks π with b vertices

- if $k + b \leq c_0$, use base case algorithm
- enumerate all separators X of size $\leq 15\sqrt{k} + 2$
- enumerate all separations of the terminal faces into K_1 and K_2
- add vertices from X to the blocks, and enumerate sets of blocks π_1 and π_2 that can be combined to form π
- recurse on inputs (G, K_1, π_1) and (G, K_2, π_2)
- return the minimum result

Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Analysis: $2^{O(k)} \cdot n^{O(\sqrt{k})}$ time.

- $2^{O(k)}$ comes from separating the terminal faces
- $n^{O(\sqrt{k})}$ comes from picking a separator

Directed Steiner Tree

Based on Kisfaludi-Bak et al.

On directed graphs:

- how is the right connectivity maintained in block Steiner forests?
- how is the root vertex incorporated?
- can the proof for the separator size be adapted?
- how is the running time affected?

Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Connectivity in forests:

- maintain a root vertex for all blocks
- ensure proper connectivity when picking subproblem inputs; arithmetic on blocks
- topmost function call takes one block with the proper root

Directed Steiner Tree

Based on Kisfaludi-Bak et al.

Picking the separator:

- no significant change, as arc directions can be ignored in this part

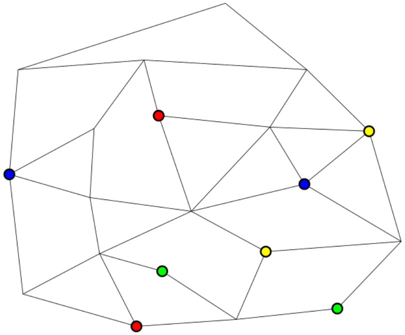
Running time: no change; adding roots does not affect asymptotic bound

Disjoint Paths

Disjoint Paths

Introduction

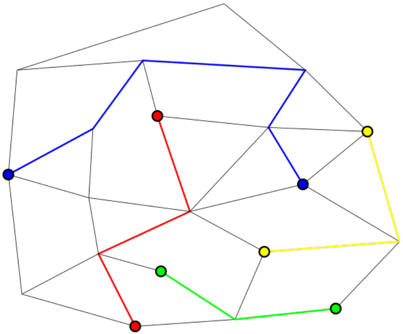
DISJOINT PATHS: given graph G and terminals $\{\{s_1, t_1\}, \ldots, \{s_k, t_k\}\}$, find k disjoint paths connecting each s_i to t_i.



Disjoint Paths

Introduction

DISJOINT PATHS: given graph G and terminals $\{\{s_1, t_1\}, \ldots, \{s_k, t_k\}\}$, find k disjoint paths connecting each s_i to t_i.



Disjoint Paths

Introduction

- VLSI chip design: connect terminals with wires on a chip
- Theoretical interest: Robertson and Seymour's graph minors project; ingredient for FPT Minor Testing algorithm
- Very hard to solve

Disjoint Paths

Irrelevant vertices technique

Robertson and Seymour: FPT algorithm using *irrelevant vertices*.

Vertex v is *irrelevant*:

(G, P) is has a solution $\iff$ $(G - v, P)$ has a solution

Disjoint Paths

Irrelevant vertices technique

Irrelevant vertices technique:

- If $tw(G) > g(k)$, then there must be an irrelevant vertex
- Remove irrelevant vertices until treewidth reaches $g(k)$
- Solve using tree decomposition algorithm

Robertson and Seymour: Disjoint Paths can be solved in $f(k) \cdot O(n^3)$ time[5]. This is FPT, but...

[5] Neil Robertson and Paul D. Seymour. "Graph Minors .XIII. The Disjoint Paths Problem". In: *Journal of Combinatorial Theory, Series B* 63.1 (1995), pp. 65–110.

Disjoint Paths

Galactic algorithms

- Robertson and Seymour: $\textsc{Disjoint Paths}$ can be solved in $f(k) \cdot O(n^3)$ time
- Kawarabayashi and Wollan: $f(k) = 2^{2^{2^{2^{\Omega(k)}}}}$ [6]
- Galactic algorithm: FPT, but in no way practical

[6] Ken-ichi Kawarabayashi and Paul Wollan. "A Shorter Proof of the Graph Minor Algorithm: The Unique Linkage Theorem". In: *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*. STOC '10. New York, NY, USA, 2010, pp. 687–694.

Disjoint Paths

Galactic algorithms

"For any instance $G = (V, E)$ that one could into the known universe, one would easily prefer $|V|^{70}$ to even *constant* time, if that constant had to be one of Robertson and Seymour's."

- David Johnson[7]

[7] David S Johnson. "The NP-completeness column: An ongoing guide". In: *Journal of algorithms* 8.2 (1987), pp. 285–303.

Disjoint Paths

Planar Disjoint Paths

Better results on planar graphs:

- Schrijver: $n^{O(k)}$ time using algebraic approach[8]
- Adler et al.: $2^{2^{O(k)}} \cdot n$ time using irrelevant vertices[9]
- Lokshtanov et al.: $2^{O(k^2)} \cdot n^{O(1)}$ time combining the two[10]

---

[8] Alexander Schrijver. "Finding k disjoint paths in a directed planar graph". In: *SIAM Journal on Computing* 23.4 (1994), pp. 780–788.

[9] Isolde Adler et al. "Irrelevant vertices for the planar disjoint paths problem". In: *Journal of Combinatorial Theory, Series B* 122 (2017), pp. 815–843.

[10] Daniel Lokshtanov et al. "An exponential time parameterized algorithm for planar disjoint paths". In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. 2020, pp. 1307–1316.

Disjoint Paths

Planar Disjoint Paths on few faces

What if all terminals lie on few faces?

Face cover number	Running time	Authors
1	$O(n)$	Robertson and Seymour[11]
2	$O(n)$	Ripphausen-Lipa et al.[12]
any k	$n^{f(k)}$	Schrijver[13]

[11] Neil Robertson and Paul D. Seymour. “Graph minors. VI. Disjoint paths across a disc”. In: *Journal of Combinatorial Theory, Series B* 41.1 (1986), pp. 115–138.

[12] Heike Ripphausen-Lipa, Dorothea Wagner, and Karsten Weihe. “Linear-time algorithms for disjoint two-face paths problems in planar graphs”. In: *International Journal of Foundations of Computer Science* 7.02 (1996), pp. 95–110.

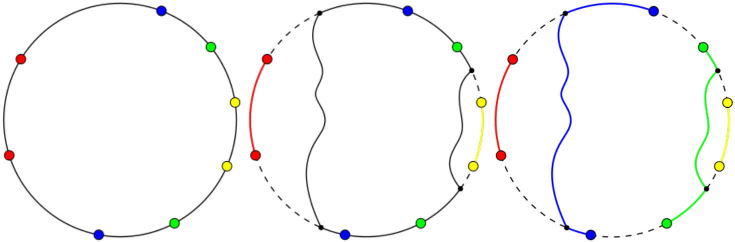
[13] Alexander Schrijver. “Disjoint homotopic paths and trees in a planar graph”. In: *Discrete & Computational Geometry* 6.4 (1991), pp. 527–574.

Disjoint Paths

Planar Disjoint Paths on one face

One face: greedy algorithm by ordering the terminals

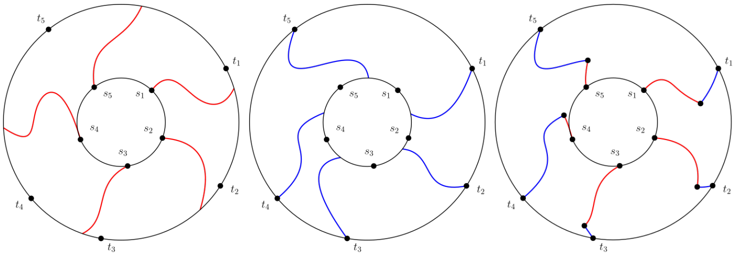
- Must be pair of terminals with no others between them
- Path along face border is always “free"



Disjoint Paths

Planar Disjoint Paths on two faces

Two faces: find two sets of paths trending in counterclockwise direction, combine to make full paths



Disjoint Paths

Planar Disjoint Paths on few faces

k faces:

- enumerate homotopy classes of connecting curves in the plane
- use linear programming to shift the curves and create disjoint paths in the graph

Concluding remarks

Concluding remarks

In summary:

- Face cover number potentially powerful as parameter
- Both problems are FPT by number of terminals, allow XP algorithm by face cover number
- Steiner Tree algorithms readily adapted to directed graphs

Concluding remarks

Future research:

- FPT algorithm or lower bound for Planar Disjoint Paths by face cover number
- Better Disjoint Paths algorithms in general?
- Planar Steiner Tree on one or two faces
- Complexity of Directed Steiner Tree