Solving Planar k-Terminal Cut in $O(n^{c\sqrt{k}})$ time

Philip N. Klein[1] Dániel Marx[2]

[1]Computer Science Department,
Brown University
Providence, RI

[2]Computer and Automation Research Institute,
Hungarian Academy of Sciences (MTA SZTAKI)
Budapest, Hungary

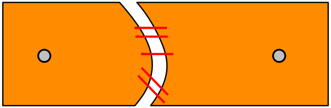
ICALP 2012
Warwick, UK
July 13, 2012

A classical problem

$s - t$ Cut

Input: A graph G, an integer p, vertices s and t

Output: A set S of at most p edges such that removing S separates s and t.



Fact

A minimum $s - t$ cut can be found in polynomial time.

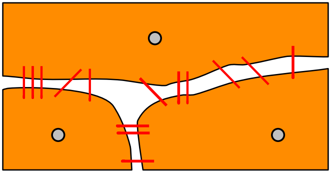
What about separating more than two terminals?

More than two terminals

Multiway Cut

Input: A graph G, an integer p, and a set T of terminals
Output: A set S of at most p edges such that removing S separates any two vertices of T

Note: Also called Multiterminal Cut or k-Terminal Cut.



Theorem [Dalhaus et al. 1994]

NP-hard already for $|T| = 3$.

Planar graphs

Theorem [Dalhaus et al. 1994] [Hartvigsen 1998] [Bentz 2012]

k-Terminal Cut can be solved in time $n^{O(k)}$ on **planar graphs.**

Main result

k-Terminal Cut can be solved in time $c^k \cdot n^{O(\sqrt{k})}$ on **planar graphs.**

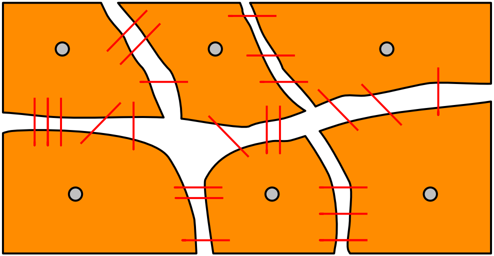
The improvement in the exponent is best possible:

Previous talk

Assuming ETH, k-Terminal Cut on planar graphs cannot be solved in time $f(k) \cdot n^{o(\sqrt{k})}$ for any computable function $f(k)$.

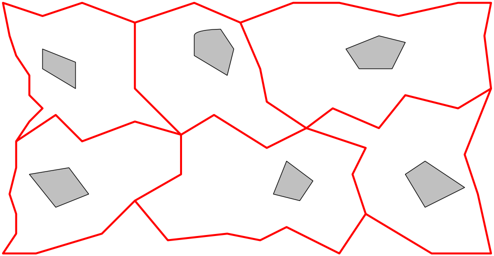
Dual graph

The previous algorithms (as well as ours) look at the solution in the dual graph



Dual graph

The previous algorithms (as well as ours) look at the solution in the dual graph

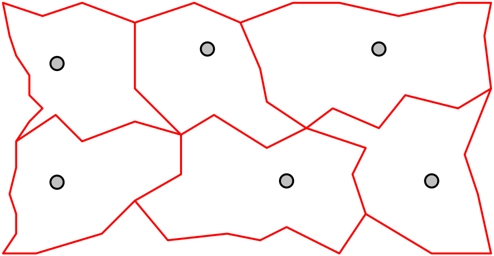


Recall:

Primal graph		Dual graph
vertices	$\Leftrightarrow$	faces
faces	$\Leftrightarrow$	vertices
edges	$\Leftrightarrow$	edges

Dual graph

The previous algorithms (as well as ours) look at the solution in the dual graph



Recall:

Primal graph		Dual graph
vertices	$\Leftrightarrow$	faces
faces	$\Leftrightarrow$	vertices
edges	$\Leftrightarrow$	edges

We slightly transform the problem in such a way that the terminals are represented by **vertices** in the dual graph (instead of faces).

Previous approaches

[Dalhaus et al. 1994] [Hartvigsen 1998] [Bentz 2012]

1. The dual solution has $O(k)$ branch vertices.
2. Guess the location of branch vertices ($n^{O(k)}$ guesses).
3. Deep magic to find the paths connecting the branch vertices (shortest paths are not necessarily good!)

New idea:

Fact

A planar graph with k vertices has treewidth $O(\sqrt{k})$.

The dual solution has treewidth $O(\sqrt{k})$, so instead of guessing, let's find the vertices in a dynamic programming on the tree decomposition.

Problem: How to implement the deep magic in a DP?

Previous approaches

[Dalhaus et al. 1994] [Hartvigsen 1998] [Bentz 2012]

1. The dual solution has $O(k)$ branch vertices.
2. Guess the location of branch vertices ($n^{O(k)}$ guesses).
3. Deep magic to find the paths connecting the branch vertices (shortest paths are not necessarily good!)

New idea:

Fact

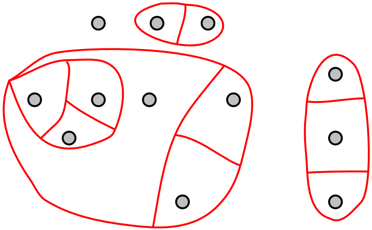
A planar graph with k vertices has treewidth $O(\sqrt{k})$.

The dual solution has treewidth $O(\sqrt{k})$, so instead of guessing, let's find the vertices in a dynamic programming on the tree decomposition.

Problem: How to implement the deep magic in a DP?

2-connectivity

In general, the dual solution is not 2-connected.



2-connected problem

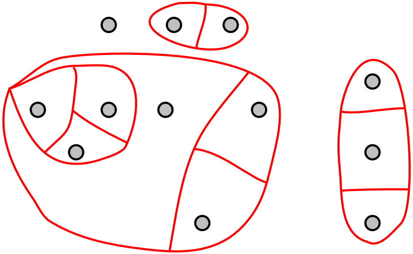
Find a 2-connected dual solution that separates a subset X of terminals from each other and from every other terminal.

A simple DP reduces the original problem to the 2-connected problem.

2-connectivity

$a(X)$: cost of separating the terminals in X from each other.
$b(X)$: cost of separating X from each other and from every other terminal with a solution that is 2-connected in the dual.

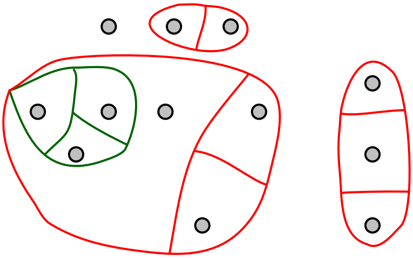
$$a(T) = \min_{\emptyset \neq X \subseteq T} (b(X) + a(T \setminus X))$$



2-connectivity

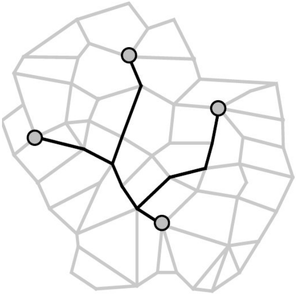
$a(X)$: cost of separating the terminals in X from each other.
$b(X)$: cost of separating X from each other and from every other terminal with a solution that is 2-connected in the dual.

$$a(T) = \min_{\emptyset \neq X \subseteq T} (b(X) + a(T \setminus X))$$



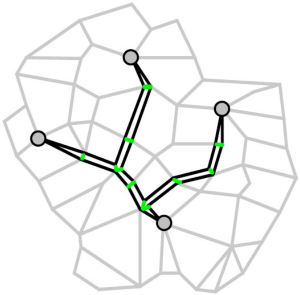
The Steiner tree

We find a minimum cost Steiner tree T of the terminals in the **dual** and cut open the graph along the tree.
(Steiner tree: $3^k \cdot n^{O(1)}$ time by [Dreyfus-Wagner 1972] or $2^k \cdot n^{O(1)}$ time by [Björklund 2007])



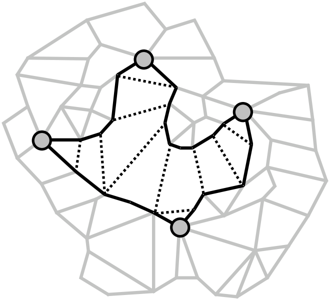
The Steiner tree

We find a minimum cost Steiner tree T of the terminals in the **dual** and cut open the graph along the tree.
(Steiner tree: $3^k \cdot n^{O(1)}$ time by [Dreyfus-Wagner 1972] or $2^k \cdot n^{O(1)}$ time by [Björklund 2007])



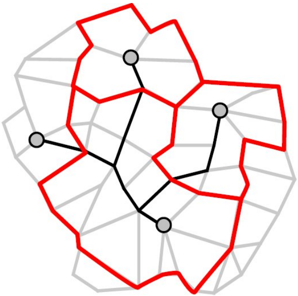
The Steiner tree

We find a minimum cost Steiner tree T of the terminals in the **dual** and cut open the graph along the tree.
(Steiner tree: $3^k \cdot n^{O(1)}$ time by [Dreyfus-Wagner 1972] or $2^k \cdot n^{O(1)}$ time by [Björklund 2007])



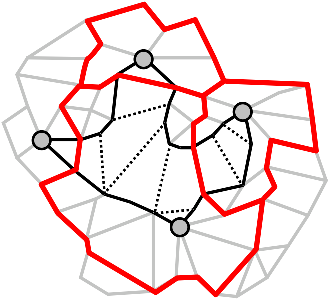
The Steiner tree

We find a minimum cost Steiner tree T of the terminals in the **dual** and cut open the graph along the tree.
(Steiner tree: $3^k \cdot n^{O(1)}$ time by [Dreyfus-Wagner 1972] or $2^k \cdot n^{O(1)}$ time by [Björklund 2007])



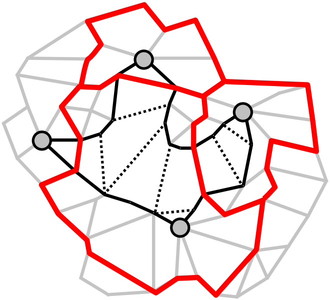
The Steiner tree

We find a minimum cost Steiner tree T of the terminals in the **dual** and cut open the graph along the tree.
(Steiner tree: $3^k \cdot n^{O(1)}$ time by [Dreyfus-Wagner 1972] or $2^k \cdot n^{O(1)}$ time by [Björklund 2007])



The Steiner tree

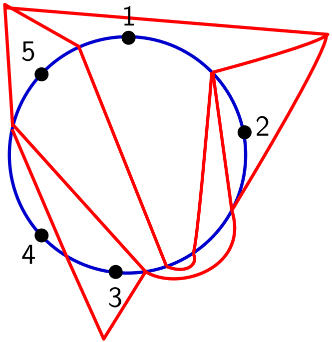
We find a minimum cost Steiner tree T of the terminals in the **dual** and cut open the graph along the tree.
(Steiner tree: $3^k \cdot n^{O(1)}$ time by [Dreyfus-Wagner 1972] or $2^k \cdot n^{O(1)}$ time by [Björklund 2007])



Key idea: the paths of the dual solution between the branch points/crossing points can be assumed to be shortest paths.

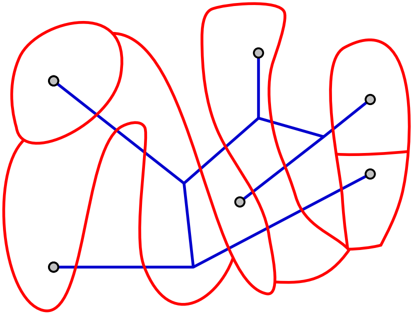
Topology

Key idea: the paths of the dual solution between the branch points/crossing points can be assumed to be shortest paths.



Thus a solution can be completely described by the location of these points and which of them are connected.
A "topology" just describes the connections without the locations.

A combinatorial lemma



Lemma

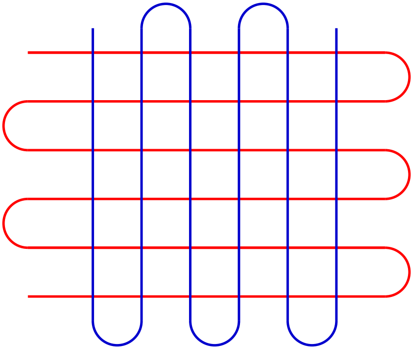
There is an optimum dual solution S that has $O(k)$ branch vertices and “crosses the tree” $O(k)$ times.

Proof uses

- the minimality of T,
- the minimality of S,
- the 2-connectivity of S,
- Euler’s formula.

A proof idea

Why this cannot happen?

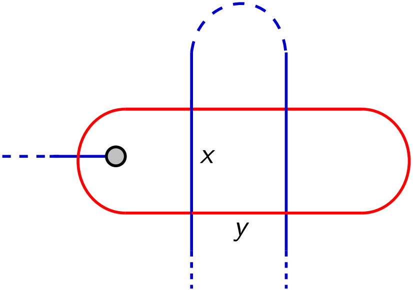


There are no red-blue-red-blue faces:

- If $x < y$, then we can get a better solution S.
- If $x > y$, then we can get a better Steiner tree T.

A proof idea

Why this cannot happen?

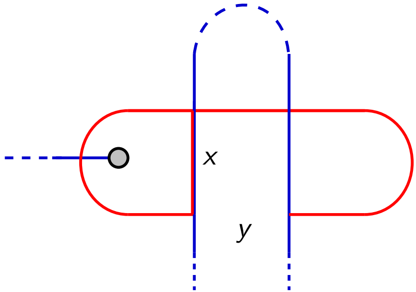


There are no red-blue-red-blue faces:

- If $x < y$, then we can get a better solution S.
- If $x > y$, then we can get a better Steiner tree T.

A proof idea

Why this cannot happen?

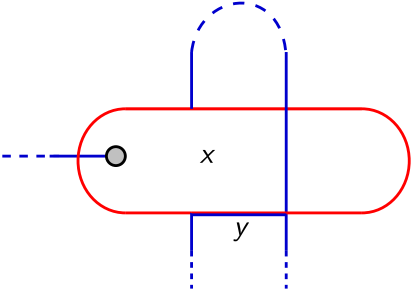


There are no red-blue-red-blue faces:

- If $x < y$, then we can get a better solution S.
- If $x > y$, then we can get a better Steiner tree T.

A proof idea

Why this cannot happen?



There are no red-blue-red-blue faces:

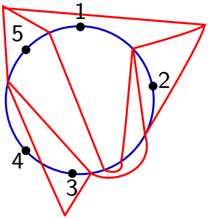
- If $x < y$, then we can get a better solution S.
- If $x > y$, then we can get a better Steiner tree T.

Realizing a topology

Lemma

Given a topology of size p, we can find a minimum cost realization in time $n^{O(\sqrt{p})}$.

- p branch points/crossing points $\Rightarrow$ treewidth is $O(\sqrt{p})$.
- Fairly standard DP on the tree decomposition.
- In each bag of the tree decomposition, we have to keep track of the location of $O(\sqrt{p})$ points $\Rightarrow n^{O(\sqrt{p})}$ possibilities.
- We need that the crossing points and the terminals are in the right order, but that is easy.



Algorithm

For the 2-connected problem:

1. Find the Steiner tree T ($2^k \cdot n^{O(1)}$ time).
2. Cut along T.
3. Guess a "topology" of size $O(k)$ (c^k guesses).
4. Find a minimum cost realization of the topology using DP on the tree decomposition ($n^{O(\sqrt{k})}$ time).

For the general problem:

1. Solve 2^k instances of the 2-connected problem.
2. Solved the general problem for every subset using DP.

Conclusions

- A $c^k \cdot n^{O(\sqrt{k})}$ time algorithm for k-terminal planar Multiway Cut.
- Is there an $n^{O(\sqrt{k})}$ time algorithm?
- Eventually boils down to the $O(\sqrt{n})$ treewidth bound on planar graphs, but not just a trivial application of bidimensionality.
- It seems hard to prove lower bounds better than $\Omega(\sqrt{k})$ for planar problems. There should be $O(\sqrt{k})$ algorithms for **all** these problems!