Exact Algorithms for Steiner Tree

Ondra Suchý

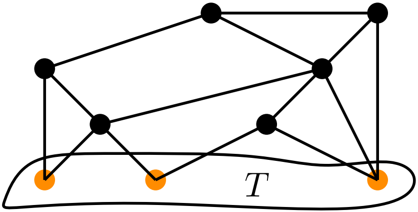
Faculty of Information Technology,
Czech Technical University in Prague,
Prague, Czech Republic.
ondrej.suchy@fit.cvut.cz

New Developments in Exact Algorithms and Lower Bounds,
IIT Delhi, 13th December 2014

Outline

1. Problem introduction and classical results
2. Exact algorithms for the general undirected case
3. Directed variants and algorithms for them
4. Algorithms for Steiner problems in sparse graphs

Steiner Tree

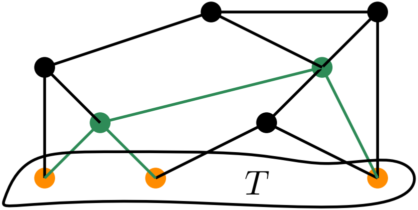


STEINER TREE

Given: Undirected graph $G = (V, E)$ and a set $T \subseteq V$

Find: A minimun size tree $H = (V', E')$ subgraph of G such that $T \subseteq V'$.

Steiner Tree

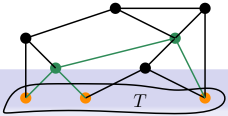


STEINER TREE

Given: Undirected graph $G = (V, E)$ and a set $T \subseteq V$

Find: A minimun size tree $H = (V', E')$ subgraph of G such that $T \subseteq V'$.

Steiner Tree



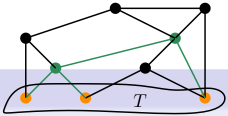
Steiner Tree

Given: Undirected graph $G = (V, E)$ and a set $T \subseteq V$

Find: A minimun size tree $H = (V', E')$ subgraph of G such that $T \subseteq V'$.

- The vertices in T are called *terminals*
- The vertices in $V \setminus T$ are called *Steiner points*
- Denote $n := |V|$, $m := |E|$, and $t := |T|$

Steiner Tree



Steiner Tree

Given: Undirected graph $G = (V, E)$ and a set $T \subseteq V$
Find: A minimun size tree $H = (V', E')$ subgraph of G such that $T \subseteq V'$.

- The vertices in T are called *terminals*
- The vertices in $V \setminus T$ are called *Steiner points*
- Denote $n := |V|$, $m := |E|$, and $t := |T|$

A minimum size:

- Vertex cardinality: $|V'|$ or rather $|S| := |V' \setminus T|$ (default)
- Edge cardinality: $|E'| = |V'| - 1$, this is equal to the above
- Node weighted: Given $w : V \to \mathbb{N}$ minimize $w(S)$
- Edge weighted: Given $w : E \to \mathbb{N}$ minimize $w(E')$

Steiner Tree (default decision variant)

Given: Undirected graph $G = (V, E)$, a set $T \subseteq V$, and $k \in \mathbb{N}$
Question: Is there a tree $H = (V', E')$, subgraph of G, such that $T \subseteq V'$ and $|V' \setminus T| \leq k$.

Importance

- Fundamental problem of network design
- Motivated by applications in, e.g.,
  - VLSI routing
  - phylogenetic tree reconstruction
  - network routing
- Several books devoted to Steiner Trees
  - Dietmar Cieslik: *Steiner minimal trees*, Kluwer Academic, 1989
  - Frank K. Hwang, Dana S. Richards, Pawel Winter: *The Steiner tree problem*, North-Holland, 1992
  - Alexandr O. Ivanov, Alexei A. Tuzhilin: *Minimal networks — the Steiner problem and its generalizations*, CRC Press, 1994
  - Hans J. Prömel, Angelika Steger: *The Steiner Tree Problem — A Tour through Graphs, Algorithms, and Complexity*, Springer, 2002
- 11th DIMACS Implementation Challenge (ending two weeks ago) devoted to Steiner Tree problems

Hardness and Approximability Results

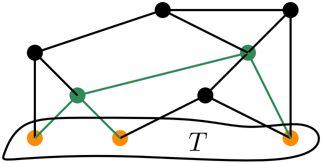
- NP-complete [Garey & Johnson 1979], even on planar graphs [Garey & Johnson, *SIAM J. Appl. Math* 1977]
- approximable to within $O(\log n)$ but not within $(1-\varepsilon)(\log t)$ unless NP $\subseteq$ DTIME$[N^{\mathrm{polylog} n}]$ [Klein & Ravi, Journal of Algorithms 1995];
- The edge weighted variant is APX-complete even on complete graphs with weights 1 and 2 [Bern, Plassmann, *Inf. Proc. Lett.* 1989]
- There are many approximation results on various variants of Steiner tree basically on every conference, e.g.,
  - Bateni, Hajiaghayi, Marx: *Approximation schemes for Steiner Forest on planar graphs and graphs of bounded treewidth STOC* 2010
  - Bateni, Checkuri, Ene, Hajiaghayi, Korula, Marx: *Prize-collecting Steiner problems on planar graphs SODA* 2011.
  - *STOC* 2014, *FOCS* 2013, and elsewhere
- An online compendium of approx. results: Hauptmann and Karpinski: *A Compendium on Steiner Tree Problems*, University of Bonn.

Exact Algorithms

In this talk:

1. Parameterized algorithms — exponential in some presumably small parameter:
   - number of Steiner points in the solution $k := |S| = |V' \setminus T|$
   - number of terminals t
   - total cardinality of the tree $t + k$
2. Kernelizations
3. Exact exponential time algorithms — exponential in the “input size”

Trivial Algorithms



Simple algorithm:

- Leaves of an optimal tree are terminals.
- An optimal tree contains at most t vertices of degree at least 3.
- Once the set T' of vertices of degree ≥ 3 is known, the optimal tree can be computed as minimum spanning tree of $T \cup T'$, where the lengths are distances in G.
- This gives $n^{O(t)}$ and $O(2^{\frac{2}{3}n} n^{O(1)}) = O(1.6181^n)$ algorithm for edge weighted variant of STEINER TREE.

Parameterization by the Solution Size

Theorem

Steiner Tree is W[2]-hard with respect to the number k of Steiner points in the solution.

- We show a parameterized reduction from

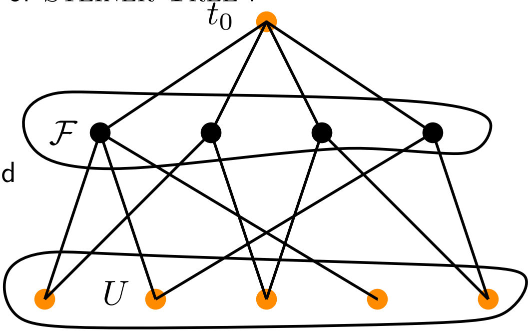
Set Cover

Given: A universe U, a family $\mathcal{F}$ of its subsets, and $k \in \mathbb{N}$
Question: Is there a subfamily $\mathcal{F}' \subset \mathcal{F}$, $|\mathcal{F}'| \leq k$ such that $\bigcup_{F \in \mathcal{F}'} F = U$?

- Set Cover was shown W[2]-hard with respect to k in Downey & Fellows 1999

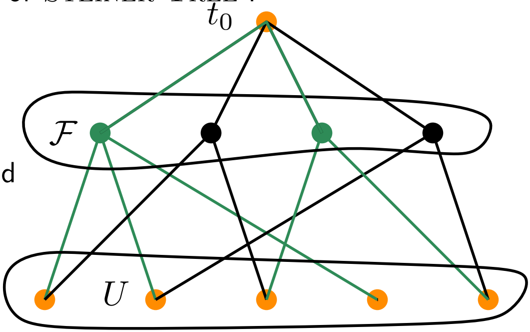
Hardness of Steiner Tree, continued

- For an instance $(U, \mathcal{F}, k)$ of Set Cover consider the following instance $(G = (V, E), T, k)$ of Steiner Tree :
- $V = U \cup \mathcal{F} \cup \{t_0\}$
- $E = \{\{u, F\} \mid u \in F \in \mathcal{F}\}$
  $\cup \{\{t_0, F\} \mid F \in \mathcal{F}\}$
- $T = U \cup \{t_0\}$
- Steiner points 1-1 correspond to the sets in $\mathcal{F}$
- we claim that
  the instances are equivalent



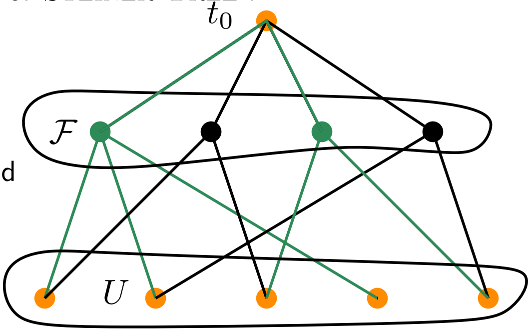
Hardness of Steiner Tree, continued

- For an instance $(U, \mathcal{F}, k)$ of Set Cover consider the following instance $(G = (V, E), T, k)$ of Steiner Tree :
- $V = U \cup \mathcal{F} \cup \{t_0\}$
- $E = \{\{u, F\} \mid u \in F \in \mathcal{F}\}$ $\cup\{\{t_0, F\} \mid F \in \mathcal{F}\}$
- $T = U \cup \{t_0\}$
- Steiner points 1-1 correspond to the sets in $\mathcal{F}$
- we claim that the instances are equivalent



Hardness of Steiner Tree, continued

- For an instance $(U, \mathcal{F}, k)$ of Set Cover consider the following instance $(G = (V, E), T, k)$ of Steiner Tree :
- $V = U \cup \mathcal{F} \cup \{t_0\}$
- $E = \{\{u, F\} \mid u \in F \in \mathcal{F}\}$
  $\cup \{\{t_0, F\} \mid F \in \mathcal{F}\}$
- $T = U \cup \{t_0\}$
- Steiner points 1-1 correspond to the sets in $\mathcal{F}$
- we claim that the instances are equivalent



Corollary

If for any $k \geq 3$ and $\varepsilon > 0$ Steiner Tree can be solved in time $O(n^{k-\varepsilon})$ then the Strong ETH fails.

- follows from the results of Patrascu and Williams [*SODA* 2010], since
- Dominating Set is a special case of Set Cover,

Parameterization by Terminals

Theorem [Dreyfus & Wagner, *Networks* 1971] and [Levin 1971]

Edge weighted $\textsc{Steiner Tree}$ can be solved in time $O(3^t \cdot n + 2^t \cdot n^2 + n(n \log n + m))$.

Proof:

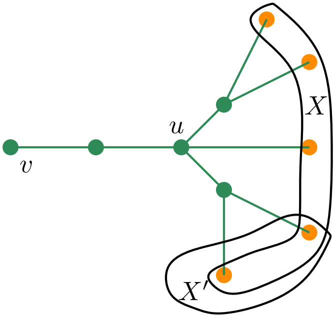
- The proof goes by dynamic programming.
- Pick any terminal t_0 and let $T' = T \setminus \{t_0\}$
- For every nonempty $X \subset T'$ and every $v \in V$ we compute:

$$\mathrm{ST}(X, v) = \text{ minimum edge weight of a Steiner tree for } (X \cup \{v\})$$

- Note that we allow $v \in X$
- The answer is stored in $\mathrm{ST}(T', t_0)$
- The trivial case: If $X = \{x\}$ for some $x \in T'$ then for every $v \in V$ we set $\mathrm{ST}(\{x\}, v) = \mathrm{dist}_G(x, v)$.

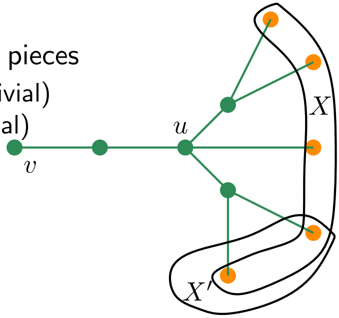
Dreyfus-Wagner Algorithm continued

- Now suppose $|X| \geq 2$
- Look at the tree from v
- Starting from v go along the tree until you reach either a vertex in X or a vertex of degree at least 3. Let us call it u. Possibly $u = v$.



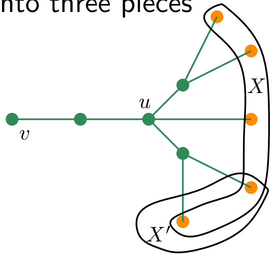
Dreyfus-Wagner Algorithm continued

- Now suppose $|X| \geq 2$
- Look at the tree from v
- Starting from v go along the tree until you reach either a vertex in X or a vertex of degree at least 3. Let us call it u. Possibly $u = v$.
- If $u \in X$ then we let $X' = \{u\}$.
- Otherwise we let X' be the vertices in X in one connected component of the tree with $\{u\}$ removed.
- In both cases we have $\emptyset \neq X' \subsetneq X$ and the tree can be split into three pieces
  - the path from v to u (possibly trivial)
  - a tree for u and X' (possibly trivial)
  - a tree for u and $X \setminus X'$



D-W Algorithm Recurrence

- We have $\emptyset \neq X' \subsetneq X$ and the tree can be split into three pieces
  - the path from v to u (possibly trivial)
  - a tree for u and X' (possibly trivial)
  - a tree for u and $X \setminus X'$



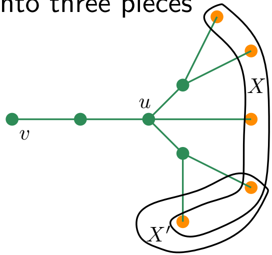
- The table can be computed using the following recurrence:

$$\mathrm{ST}(X, v) = \min_{v \in V}(\mathrm{dist}_G(v, u) + \min_{\emptyset \neq X' \subsetneq X}(\mathrm{ST}(X', u) + \mathrm{ST}(X \setminus X', u)))$$

- Both X' and $X \setminus X'$ are strictly smaller and nonempty

D-W Algorithm Recurrence

- We have $\emptyset \neq X' \subsetneq X$ and the tree can be split into three pieces
  - the path from v to u (possibly trivial)
  - a tree for u and X' (possibly trivial)
  - a tree for u and $X \setminus X'$



- The table can be computed using the following recurrence:

$$\mathrm{ST}(X, v) = \min_{v \in V}(\mathrm{dist}_G(v, u) + \min_{\emptyset \neq X' \subsetneq X}(\mathrm{ST}(X', u) + \mathrm{ST}(X \setminus X', u)))$$

- Both X' and $X \setminus X'$ are strictly smaller and nonempty

Running time:

- Each vertex of T' can be either in X', in $X \setminus X'$, or in $T' \setminus X$
- There are $3^{t-1}n^2$ evaluations of the recurrence.
- One can save by precomputing the second minimum.
- The running time follows

Improvements of the D-W Algorithm

- The 1971 D-W algorithm achieves time $O(3^t \cdot n + 2^t \cdot n^2 + n(n \log n + m))$
- This can be improved to $O(3^t \cdot n + 2^t(n \log n + m))$ by computing the distances more cleverly on demand [Erickson, Monma, Veinott *Mathematics of Operations Research* 1987]
- In 2007 Fuchs, Kern, and Wang [*Math. Meth. Oper. Res.*] improved this to $O(2.684^t n^{O(1)})$ and
- Mölle, Richter, and Rossmanith [*STACS* 2006] to $O((2+\varepsilon)^t n^{f(e^{-1})})$
- later the above two groups together [*Theory Comput. Syst.* 2007] improved the exponent to $O((\frac{\varepsilon}{-\ln \varepsilon})^{-\delta})$ for any $1/2 < \delta$ which gives, e.g., $O(2.5^t n^{14.2})$ or $O(2.1^t n^{57.6})$
- By using subset convolution and Möbius inversion, one can get to a running time of $\tilde{O}(2^t n^2 + nm)$ for the node weighted case with bounded weights [Björklund, Husfeldt, Kaski, Koivisto *STOC* 2007]

Improvements of the D-W Algorithm

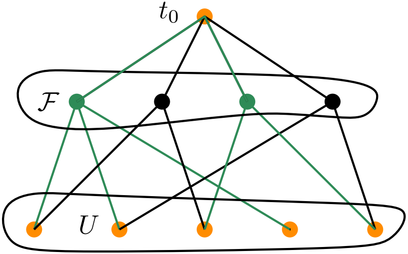
- The 1971 D-W algorithm achieves time $O(3^t \cdot n + 2^t \cdot n^2 + n(n \log n + m))$
- This can be improved to $O(3^t \cdot n + 2^t(n \log n + m))$ by computing the distances more cleverly on demand [Erickson, Monma, Veinott *Mathematics of Operations Research* 1987]
- In 2007 Fuchs, Kern, and Wang [*Math. Meth. Oper. Res.*] improved this to $O(2.684^t n^{O(1)})$ and
- Mölle, Richter, and Rossmanith [*STACS* 2006] to $O((2+\varepsilon)^t n^{f(e^{-1})})$
- later the above two groups together [*Theory Comput. Syst.* 2007] improved the exponent to $O((\frac{\varepsilon}{-\ln \varepsilon})^{-\delta})$ for any $1/2 < \delta$ which gives, e.g., $O(2.5^t n^{14.2})$ or $O(2.1^t n^{57.6})$
- By using subset convolution and Möbius inversion, one can get to a running time of $\tilde{O}(2^t n^2 + nm)$ for the node weighted case with bounded weights [Björklund, Husfeldt, Kaski, Koivisto *STOC* 2007]
- All these algorithms take exponential space.

Polynomial Space Algorithms

- First polynomial space algorithm: $O(6^t n^{O(\log t)})$ for edge weighted variant [Fomin, Grandoni, Kratsch *ESA 2008*]
- Combining with, e.g., D&W for $t < \log n$ one obtains $O^*(2^{O(t \log t)})$-time polynomial space algorithm for the edge weighted variant.
- Finally, Nederlof [*ICALP* 2009] gave a $2^t n^{O(1)}$-time polynomial space algorithm for edge weighted variant with bounded weights
- It is pretty simple and based on the inclusion-exclusion principle

Further Improvements

- Can we hope for an algorithm running in $(2-\varepsilon)^t n^{O(1)}$-time?
- This would imply an algorithm for Set Cover with running time $(2-\varepsilon)^{|U|}|\mathcal{F}|^{O(1)}$ (by the presented reduction)



Set Cover Conjecture (SeCoCo) [Cygan, Dell, Lokshtanov, Marx, Nederlof, Okamoto, Paturi, Saurabh, Wahlström *CCC* 2012]

There is no $(2-\varepsilon)^{|U|}|\mathcal{F}|^{O(1)}$-time algorithm for Set Cover.

- OPEN: Is there any relation between SeCoCo and SETH?

Kernelization

Theorem [folklore / Dom, Lokshtanov, Saurabh *ACM Transactions on Algorithms 2014* / *ICALP* 2009]

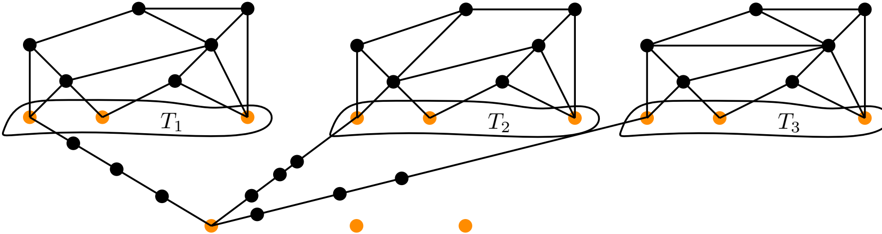
There is no polynomial kernel for Steiner Tree parameterized by $t + k$ unless NP$\subseteq$coNP/poly, which would imply the collapse of the Polynomial Hierarchy to the third level.

Proof:

- By the framework of Bodlaender, Downey, Fellows, and Hermelin [J. Comput. Syst. Sci. 2009/ ICALP 2008] we have to show that Steiner Tree is compositional with this parameterization
- Consider instances $(G_1, T_1, k_1), \ldots, (G_s, T_s, k_s)$
- We may assume that $|T_1| = |T_2| = \ldots = |T_s| = t$ and $k_1 = k_2 = \ldots = k_s = k$
- We denote $T_1 = \{t_1^1, \ldots, t_t^1\}$, etc.

Kernel Lower Bound continued

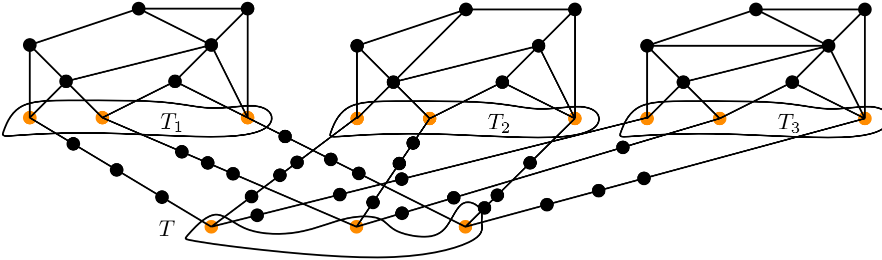
- We let $G = \bigcup_{i=1}^{s} G_i$
- For each $j \in \{1, \ldots, t\}$ we add to G a new vertex t_j and connect it to the vertices $t_j^1, t_j^2, \ldots, t_j^s$ by paths of length $k+2$.



- We let $T = \{t_1, \ldots, t_t\}$ and $k' = t \cdot (k+2) + k$.

Kernel Lower Bound continued

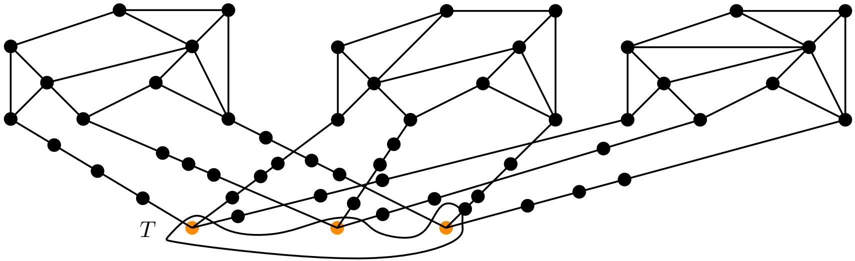
- We let $G = \bigcup_{i=1}^{s} G_i$
- For each $j \in \{1, \ldots, t\}$ we add to G a new vertex t_j and connect it to the vertices $t_j^1, t_j^2, \ldots, t_j^s$ by paths of length $k+2$.



- We let $T = \{t_1, \ldots, t_t\}$ and $k' = t \cdot (k+2) + k$.

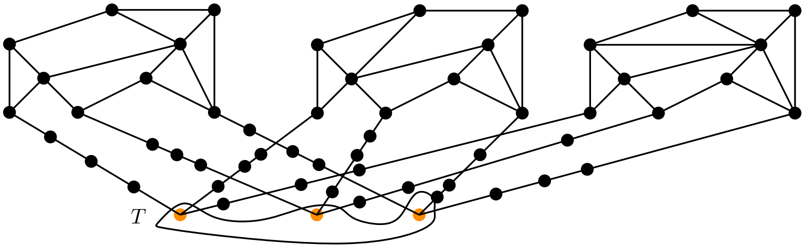
Kernel Lower Bound continued

- We let $G = \bigcup_{i=1}^{s} G_i$
- For each $j \in \{1, \ldots, t\}$ we add to G a new vertex t_j and connect it to the vertices $t_j^1, t_j^2, \ldots, t_j^s$ by paths of length $k + 2$.



- We let $T = \{t_1, \ldots, t_t\}$ and $k' = t \cdot (k + 2) + k$.

Kernel Lower Bound continued 2



- We claim that (G, T, k') is a yes instance if and only if $\exists i \in \{1, \ldots, s\}$ such that (G_i, T_i, k) is a yes-instance.
- If G_i is a yes-instance, then use the solution and add the paths to it.
- A solution for G, must contain at least one path for each t_j
- Hence, exactly one path.
- If they ended in different G_i's, then the graph would be disconnected.

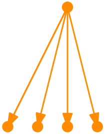
Exponential Time Algorithms

- the fast exponential algorithms are obtained by combining branching for large values of t and FPT algorithms for small t
- the fastest for weighted case is based on the algorithm of Mölle et al. achieving $O(1.42^n)$ in exponential space
- The only paper devoted to such algorithms is by Fomin, Grandoni, Kratsch, Lokshtanov, Saurabh [*Algorithmica* 2009 / *ESA 2008*]
- It uses more involved branching, quasi FPT algorithm, and analyses the running time by Measure & Conquer.
- It is polynomial space and originally achieved running time $O(1.59^n)$ for the weighted case and $O(1.55^n)$ for the cardinality case.
- Plugging in the $O^*(2^t)$ algorithm of Nederlof, the running time can be improved to $O(1.36^n)$ for the cardinality case.

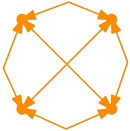
Steiner Problems in Directed Graphs

In directed graphs *"to be connected"* can mean several things:

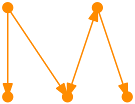
- Connect one distinguished root by directed paths to all other terminals
  DIRECTED STEINER TREE (DST)



- Connect all terminals among each other
  STRONGLY CONNECTED STEINER SUBGRAPH (SCSS)



- Connect given terminal vertex pairs in given directions
  DIRECTED STEINER NETWORK (DSN)

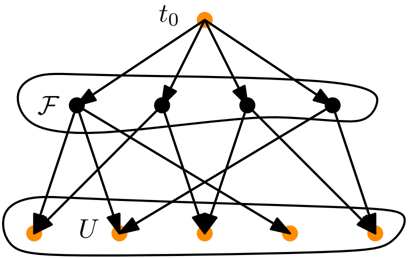


- DSN hard to approximate within $O(2^{log^{1-\epsilon} n})$ unless $NP \subseteq TIME(2^{polylog(n)})$ [Dodis, Khanna *STOC* 1999]

Parameterization by Steiner Points—Directed

With respect to the number k of Steiner points in the solution:

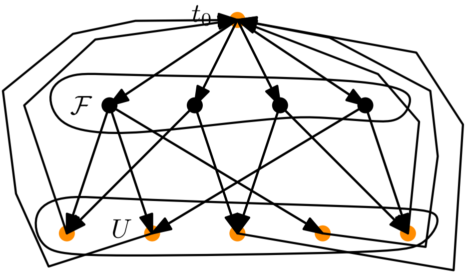
- The hardness reduction is easy to modify to show that both DIRECTED STEINER TREE and STRONGLY CONNECTED STEINER SUBGRAPH are W[2]-hard.
- It is enough to orientate all the edges "towards the universe"



Parameterization by Steiner Points—Directed

With respect to the number k of Steiner points in the solution:

- The hardness reduction is easy to modify to show that both DIRECTED STEINER TREE and STRONGLY CONNECTED STEINER SUBGRAPH are W[2]-hard.
- It is enough to orientate all the edges "towards the universe"
- and add backward arcs.
- Hence Directed Steiner Network is also W[2]-hard.



Parameterization by Terminals—Directed

With respect to the number t of terminals:

- All the FPT algorithms for STEINER TREE can be adapted to solve DIRECTED STEINER TREE.
- Feldman, Ruhl [*FOCS* 1999/ *SIAM J. Comput.* 2006]:
  - SCSS can be solved in $O(mn^{2t-3} + n^{2t-2} \log n)$ time
  - DSN can be solved in $O(mn^{4t-2} + n^{4t-1} \log n)$ time,
- Guo, Niedermeier, S.[*ISAAC* 2009/*SIAM J. Disc. Math.* 2011]:
  - SCSS is FPT with respect to t for the augmentation case — add arcs to existing graph to achieve the requested connectivity
  - SCSS is equivalent to TSP on the terminals if the graph is complete and the edge weights are 1 and 2.
  - SCSS is W[1]-hard for the cardinality case

Hardness of SCSS

Theorem [Guo, Niedemeier,S. 2009]

SCSS is W[1]-hard with respect to $k + t$ for the cardinality case.

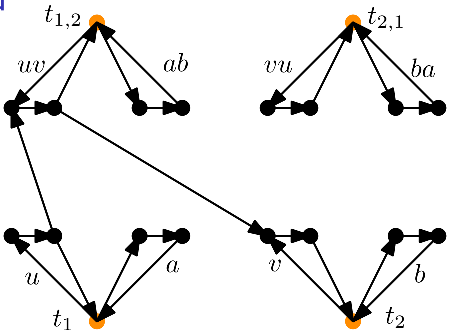
- We reduce

Multicolored Clique(MCC)

Given: A graph $G = (V, E)$, $k \in \mathbb{N}$ and a coloring $c : V \to \{1, \dots, k\}$.
Decide: Is there a clique in G taking exactly one vertex of each color?
Parameter: k

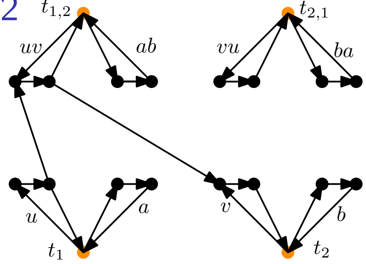
- MCC is W[1]-hard [Pietrzak, *JCSS* 2003]
- we use the edge representation strategy by Fellows, Hermelin, Rosamond and Vialette [*Theor. Comp. Sci.* 2009]

SCSS Hardness Proof ctd



- For each color i introduce a vertex t_i and for each vertex v of color i we introduce an oriented triangle t_i, v, v'
- For each pair of colors $i \neq j$, introduce two vertices $t_{i,j}$ and $t_{j,i}$.
- For each edge uv with $c(u) = i$ and $c(v) = j$, introduce two triangles $t_{i,j}, x_{uv}, x'_{uv}$ and $t_{j,i}, x_{vu}, x'_{vu}$
- add arcs (x'_{uv}, x_v), (x'_u, x_{uv}), (x'_{vu}, x_u), (x'_v, x_{vu})
- let $T = \{t_1, \ldots, t_k\} \cup \{t_{i,j} \mid i \neq j\}$ and $k' = 2k + 2\binom{k}{2}$

SCSS Hardness Proof Continued 2



- We claim that the instance (G, k, c) of MCC is equivalent to the constructed instance of SCSS

- Selecting a vertex or edge in G correspond to selecting the vertices of the corresponding triangle in the constructed digraph.
- If there is a mcc, then we can connect the terminals (easy to check).
- If the terminals are connected, then for each color there is one triangle selected, otherwise the terminal would be isolated.
- The same for pairs of colors.
- Exactly one triangle selected for each terminal.
- The selected edges must be between selected vertices.
- The selected vertices form a clique.

SCSS Running Time Lower Bound

Corollary

SCSS cannot be solved in $n^{o(t/\log t)}$ time unless ETH fails.

- the reduction can be done the same way with PARITIONED SUBGRAPH ISOMORPHISM
- the lower bound follows from Marx [*FOCS* 2007 / *Theory of Computing* 2010]

Sparse graphs

Sparse graph classes often studied:

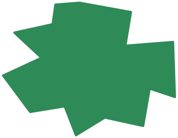
- Graphs of bounded treewidth
- planar graphs
- K_h-minor free — graphs that do not contain K_h as a minor
- K_h-topological minor free — for topological minor, one can only contract edge if one of the endpoints has degree 2
- d-degenerate $\Leftrightarrow$ every subgraph has vertex of degree at most d

Sparse directed = sparse underlying undirected

Graphs of Bounded Treewidth

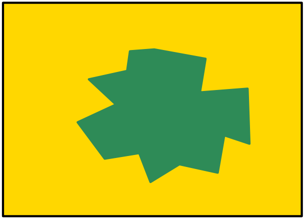
- Classical dynamic programming gives $2^{O(tw \log tw)} \cdot n$-time algorithm for Steiner Tree
- This was slightly improved to $O(B_{tw+2}^2 \cdot tw \cdot n)$, where B_k is the Bell number, by Chimani, Mutzel, Zey [*IWOCA* 2011/ *J. of Disc. Algor.* 2012]
- There is a $3^{tw} \cdot n^{O(1)}$ randomized algorithm, with no false positives and probability of false negative at most $\frac{1}{2}$ [Cygan, Nederlof, Pilipczuk, Pilipczuk, van Rooij, Wojtaszczyk *FOCS* 2011]
- It is based on a technique called "Cut & Count"

Cut and Count for Steiner Tree



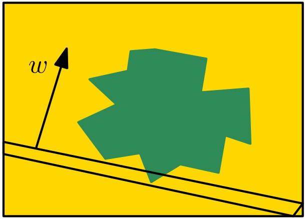
- We want to know, whether there is a *connected* subgraph of given size containing T.

Cut and Count for Steiner Tree



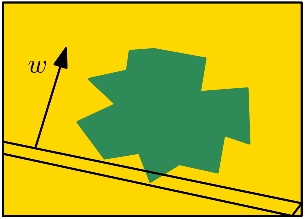
- We want to know, whether there is a *connected* subgraph of given size containing T.
- We count the number of *all* subgraphs of given size containing T.
- We do it in a way that the connected ones are counted once, while the others are counted an even number of times.
- If there was an odd number of connected ones, then the total number will be also odd.

Cut and Count for Steiner Tree



- We want to know, whether there is a *connected* subgraph of given size containing T.
- We count the number of *all* subgraphs of given size containing T.
- We do it in a way that the connected ones are counted once, while the others are counted an even number of times.
- If there was an odd number of connected ones, then the total number will be also odd.
- We use random weights on vertices to ensure that, with high probability, there is a unique Steiner tree of minimum weight.

Achieving Unique Solution



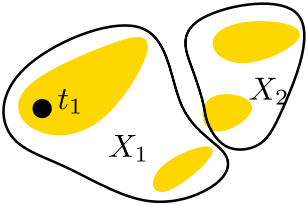
Isolation Lemma [Mulmuley, Vazirani, Vazirani *Combinatorica* 1987]

Let $\emptyset \neq \mathcal{F} \subseteq 2^U$. For each $u \in U$, choose a weight $w(u) \in \{1, \ldots, N\}$ uniformly and independently at random. Then the probability that a set of minimum weight in $\mathcal{F}$ is unique is at least $1 - \frac{|U|}{N}$.

- We use $N = 2n$.
- We guess the minimum weight, i.e., we compute the parity of the number of subgraphs for each possible weight.

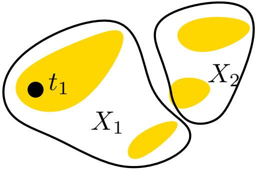
Cut and Count — What We Count

- We pick an arbitrary terminal $t_1 \in T$.
- We actually count the parity of the number of pairs of
  - a subgraph (X, F) with $T \subseteq X$ of given size and particular weight and
  - a cut (X_1, X_2) such that
    - ★ $X_1 \cap X_2 = \emptyset$,
    - ★ $X_1 \cup X_2 = X$
    - ★ $t_1 \in X_1$
    - ★ there is no edge with one endpoint in X_1 and one in X_2



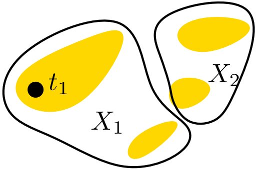
- For connected subgraphs only the cut $X_1 = X$ is possible.
- For a disconnected subgraph, there are $2^{cc(X)-1}$ possible cuts.
- The number can only become odd if there is a Steiner tree of the given size.

Cut and Count—Running Time



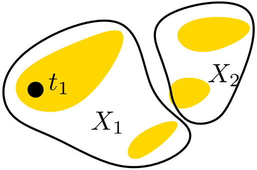
- There are three possible states of a vertex: in X_1, in X_2, outside X
- Correctness of the cut can be checked in polynomial times
- Using principle of inclusion and exclusion, we get to $3^{tw} \cdot tw^{O(1)} \cdot n$ for given size and weight
- The weights are between 1 and $2n^2$, sizes between 1 and n
- Total running time $3^{tw} \cdot n^{O(1)}$

Cut and Count—Running Time



- There are three possible states of a vertex: in X_1, in X_2, outside X
- Correctness of the cut can be checked in polynomial times
- Using principle of inclusion and exclusion, we get to $3^{tw} \cdot tw^{O(1)} \cdot n$ for given size and weight
- The weights are between 1 and $2n^2$, sizes between 1 and n
- Total running time $3^{tw} \cdot n^{O(1)}$
- In the same paper: If there is a constant $\varepsilon > 0$ and an $(3 - \varepsilon)^{pw} \cdot n^{O(1)}$ algorithm for Steiner Tree, then the Strong ETH fails.

Cut and Count—Running Time



- There are three possible states of a vertex: in X_1, in X_2, outside X
- Correctness of the cut can be checked in polynomial times
- Using principle of inclusion and exclusion, we get to $3^{tw} \cdot tw^{O(1)} \cdot n$ for given size and weight
- The weights are between 1 and $2n^2$, sizes between 1 and n
- Total running time $3^{tw} \cdot n^{O(1)}$
- In the same paper: If there is a constant $\varepsilon > 0$ and an $(3-\varepsilon)^{pw} \cdot n^{O(1)}$ algorithm for Steiner Tree, then the Strong ETH fails.
- There are more recent approaches which are deterministic and apply to weighted or counting variants [Bodlaender, Cygan, Kratsch, Nederlof *ICALP* 2013] [Fomin, Lokshtanov, Saurabh *SODA* 2014]
- OPEN: On directed graphs nothing better than $O^*(2^{tw^2})$ is known

Steiner Tree in Planar Graphs

- Consider STEINER TREE in planar graphs parameterized by the solution size k
- We can contract any edge connecting two terminals
- Hence, every other vertex on any path is Steiner
- Thus, the graph has diameter $\leq 2k$
- The treewidth is bounded by $O(k)$ and hence it is FPT wrt k.
- Recently Pilipczuk, Pilipczuk, Sankowski, van Leeuwen [*STACS* 2013] presented an algorithm running in time $O(2^{O(((k+t)\log(k+t))^{2/3})}n)$
- And even more recently this was improved to $O(2^{O(\sqrt{(k+t)\log(k+t)})}n)$ [Pilipczuk, Pilipczuk, Sankowski, van Leeuwen *FOCS* 2014]
- OPEN: Is there a subexponential algorithm parameterized only by t or only by k?

Directed Steiner Tree in Sparse Graphs

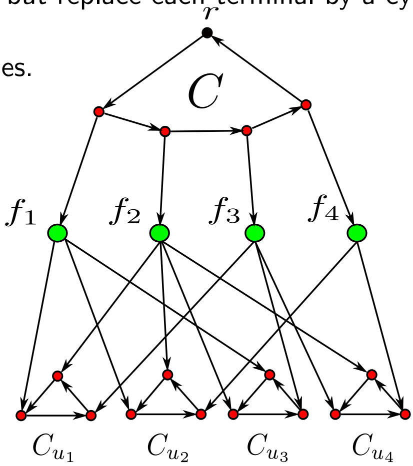
- Directed Steiner Tree in sprase graphs studied by Jones, Lokshtanov, Ramanujan, Saurabh, S. [*ESA* 2013]
- In directed planar graphs we can only contract strongly connected components formed by terminals.
- Hence we cannot bound the treewidth of the graph.
- Moreover, after contracting the strongly connected the graph may no longer be sparse.
- In fact, DST is W[2]-hard with respect to k even on 2-degenerate graphs.

DST Hardness in Degenerate Graphs

Observation

DST is W[2]-hard with respect to k even on 2-degenerate graphs.

- Use the reduction from setcover, but replace each terminal by a cycle of vertices of degree three.
- Subdivide the edges in these cycles.
- All new vertices are terminals



Directed Steiner Tree in Sparse Graphs continued

$D[T]$ arbitrary

- $O^*(3^{hk+o(hk)})$-time on K_h-minor free digraphs
  (the algorithm is based on a novel branching rule in combination with the Nederlof's algorithm)
- $O^*(f(h)^k)$-time on K_h-topological minor free digraphs
  (using the decomposition theorem of Grohe and Marx [*STOC* 2012])

$D[T]$ acyclic

- $O^*(3^{hk+o(hk)})$-time on K_h-topological minor free digraphs
- $O^*(3^{dk+o(dk)})$-time on d-degenerate graphs
  - DST is FPT wrt k on $o(\log n)$-degenerate graph classes
  - FPT algorithm for undirected Steiner Tree on d-degenerate
- For any constant $c > 0$, no $f(k)n^{o(\frac{k}{\log k})}$-time algorithm on graphs of degeneracy $c \log n$ unless ETH fails.
  - no $O^*(2^{o(d)f(k)})$-time algorithm unless ETH fails
- no $O^*(2^{f(d)o(k)})$-time algorithm unless ETH fails

Kernelization in Sparse Graphs

- STEINER TREE does not have a polynomial kernel with respect to $k + t$ even on 2-degenerate graphs, unless NP$\subseteq$coNP/poly [Cygan, Pilipczuk, Pilipczuk, Wojtaszczyk *Disc. App. Math.* 2012]
- STEINER TREE has an $O((k + t)^{142})$-size kernel on planar graphs [Pilipczuk, Pilipczuk, Sankowski, van Leeuwen *FOCS* 2014]
- a polynomial kernel also exists on bounded genus graphs and for the edge weighted variant
- OPEN: Kernel on planar graphs with respect to only k or only t?
- OPEN: Does it have a kernel on K_h-minor free or K_h-topological-minor free graphs?
- OPEN: Improve the size of the kernel.

SCSS in Sparse Graphs

- SCSS in planar graphs was recently studied by Chitnis, Hajiaghayi, Marx [*SODA* 2014]
- They show a $2^{O(t \log t)} \cdot n^{O(\sqrt{t})}$-time algorithm and also
- that an $f(k) \cdot n^{o(\sqrt{t})}$ algorithm would imply that ETH fails.
- OPEN: Can the algorithm for DIRECTED STEINER NETWORK be also somehow speed up on planar graphs?

Some Open Problems

The main open problems are (not repeating all already mentioned)

- Subexponential algorithm for STEINER TREE with respect to only t
- Improvement to the kernels for planar graphs
- generalizing the result to the higher connectivity settings (first results for SCSS obtained by Chitnis, Esfandiari, Hajiaghayi, Khandekar, Kortsarz, Seddighin [*IPEC* 2014])

Thank you for your attention!