Parameterized Approximation Schemes for Steiner Trees with Small Number of Steiner Vertices

Pavel Dvořák[1], Andreas Emil Feldmann[1], Dušan Knop[1,2], Tomáš Masařík[1], Tomáš Toufar[1], Pavel Veselý[1]

[1]Charles University,Prague, Czech Republic
[2]University of Bergen, Bergen, Norway

HALG 2018
Amsterdam, Netherlands

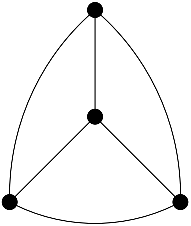
The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 616787.

Steiner Tree

- Input:
  - Graph $G = (V, E)$.
  - Edge weights: $w : E \to \mathbb{R}_0^+$.
  - Terminals $R \subseteq V$ (vertices in $V \setminus R$ are called Steiner vertices).
- Goal: find a Steiner tree $T \subseteq G$.
  - $R \subseteq V(T)$.
  - Minimize weight.

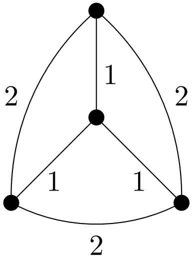
Steiner Tree

- Input:
  - Graph $G = (V, E)$.
  - Edge weights: $w : E \to \mathbb{R}_0^+$.
  - Terminals $R \subseteq V$ (vertices in $V \setminus R$ are called Steiner vertices).
- Goal: find a Steiner tree $T \subseteq G$.
  - $R \subseteq V(T)$.
  - Minimize weight.



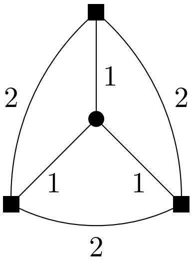
Steiner Tree

- Input:
  - Graph $G = (V, E)$.
  - Edge weights: $w : E \to \mathbb{R}_0^+$.
  - Terminals $R \subseteq V$ (vertices in $V \setminus R$ are called Steiner vertices).
- Goal: find a Steiner tree $T \subseteq G$.
  - $R \subseteq V(T)$.
  - Minimize weight.



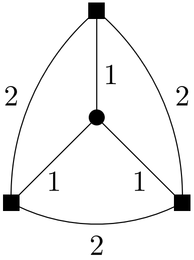
Steiner Tree

- Input:
  - Graph $G = (V, E)$.
  - Edge weights: $w : E \to \mathbb{R}_0^+$.
  - Terminals $R \subseteq V$ (vertices in $V \setminus R$ are called Steiner vertices).
- Goal: find a Steiner tree $T \subseteq G$.
  - $R \subseteq V(T)$.
  - Minimize weight.



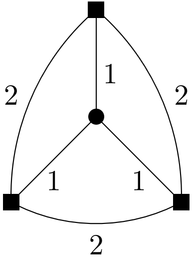
Steiner Tree

- Input:
  - Graph $G = (V, E)$.
  - Edge weights: $w : E \to \mathbb{R}_0^+$.
  - Terminals $R \subseteq V$ (vertices in $V \setminus R$ are called Steiner vertices).
- Goal: find a Steiner tree $T \subseteq G$.
  - $R \subseteq V(T)$.
  - Minimize weight.



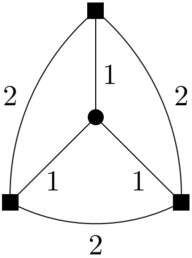
Steiner Tree

- Input:
  - Graph $G = (V, E)$.
  - Edge weights: $w : E \to \mathbb{R}_0^+$.
  - Terminals $R \subseteq V$ (vertices in $V \setminus R$ are called Steiner vertices).
- Goal: find a Steiner tree $T \subseteq G$.
  - $R \subseteq V(T)$.
  - Minimize weight.



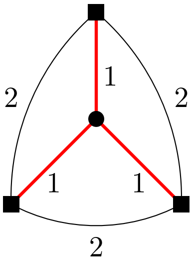
Steiner Tree

- Input:
  - Graph $G = (V, E)$.
  - Edge weights: $w : E \to \mathbb{R}_0^+$.
  - Terminals $R \subseteq V$ (vertices in $V \setminus R$ are called Steiner vertices).
- Goal: find a Steiner tree $T \subseteq G$.
  - $R \subseteq V(T)$.
  - Minimize weight.



Steiner Tree

- Input:
  - Graph $G = (V, E)$.
  - Edge weights: $w : E \to \mathbb{R}_0^+$.
  - Terminals $R \subseteq V$ (vertices in $V \setminus R$ are called Steiner vertices).
- Goal: find a Steiner tree $T \subseteq G$.
  - $R \subseteq V(T)$.
  - Minimize weight.



Known Results

Parameterized Complexity

- Number of terminals $|R|$.
  - FPT-algorithm [Dreyfus and Wagner '71, Mölle et al. '06].
  - $(1+\varepsilon)$-approximate polynomial size kernel [Lokshtanov et al. '16].
- Number of Steiner vertices in the optimum $|V(T) \setminus R| = p$.
  - W[2]-hard [folklore].

Known Results

Parameterized Complexity

- Number of terminals $|R|$.
  - FPT-algorithm [Dreyfus and Wagner '71, Mölle et al. '06].
  - $(1+\varepsilon)$-approximate polynomial size kernel [Lokshtanov et al. '16].
- Number of Steiner vertices in the optimum $|V(T) \setminus R| = p$.
  - W[2]-hard [folklore].

Known Results

Parameterized Complexity

- Number of terminals $|R|$.
  - FPT-algorithm [Dreyfus and Wagner '71, Mölle et al. '06].
  - $(1+\varepsilon)$-approximate polynomial size kernel [Lokshtanov et al. '16].
- Number of Steiner vertices in the optimum $|V(T) \setminus R| = p$.
  - W[2]-hard [folklore].

Known Results

Parameterized Complexity

- Number of terminals $|R|$.
  - FPT-algorithm [Dreyfus and Wagner '71, Mölle et al. '06].
  - $(1+\varepsilon)$-approximate polynomial size kernel [Lokshtanov et al. '16].
- Number of Steiner vertices in the optimum $|V(T) \setminus R| = p$.
  - W[2]-hard [folklore].

Known Results

Parameterized Complexity

- Number of terminals $|R|$.
  - FPT-algorithm [Dreyfus and Wagner '71, Mölle et al. '06].
  - $(1+\varepsilon)$-approximate polynomial size kernel [Lokshtanov et al. '16].
- Number of Steiner vertices in the optimum $|V(T) \setminus R| = p$.
  - W[2]-hard [folklore].

Known Results

Parameterized Complexity

- Number of terminals $|R|$.
  - FPT-algorithm [Dreyfus and Wagner '71, Mölle et al. '06].
  - $(1+\varepsilon)$-approximate polynomial size kernel [Lokshtanov et al. '16].
- Number of Steiner vertices in the optimum $|V(T) \setminus R| = p$.
  - W[2]-hard [folklore].

Known Results

Parameterized Complexity

- Number of terminals $|R|$.
  - FPT-algorithm [Dreyfus and Wagner '71, Mölle et al. '06].
  - $(1+\varepsilon)$-approximate polynomial size kernel [Lokshtanov et al. '16].
- Number of Steiner vertices in the optimum $|V(T) \setminus R| = p$.
  - W[2]-hard [folklore].

Approximation

- 96/95-approximation is NP-hard [Chlebík and Chlebíková '02].
- 1.39-approximation algorithm [Byrka et al. '13].

Known Results

Parameterized Complexity

- Number of terminals $|R|$.
  - FPT-algorithm [Dreyfus and Wagner '71, Mölle et al. '06].
  - $(1+\varepsilon)$-approximate polynomial size kernel [Lokshtanov et al. '16].
- Number of Steiner vertices in the optimum $|V(T) \setminus R| = p$.
  - W[2]-hard [folklore].

Approximation

- 96/95-approximation is NP-hard [Chlebík and Chlebíková '02].
- 1.39-approximation algorithm [Byrka et al. '13].

Known Results

Parameterized Complexity

- Number of terminals $|R|$.
  - FPT-algorithm [Dreyfus and Wagner '71, Mölle et al. '06].
  - $(1+\varepsilon)$-approximate polynomial size kernel [Lokshtanov et al. '16].
- Number of Steiner vertices in the optimum $|V(T) \setminus R| = p$.
  - W[2]-hard [folklore].

Approximation

- 96/95-approximation is NP-hard [Chlebík and Chlebíková '02].
- 1.39-approximation algorithm [Byrka et al. '13].

Known Results

Parameterized Complexity

- Number of terminals $|R|$.
  - FPT-algorithm [Dreyfus and Wagner '71, Mölle et al. '06].
  - $(1+\varepsilon)$-approximate polynomial size kernel [Lokshtanov et al. '16].
- Number of Steiner vertices in the optimum $|V(T) \setminus R| = p$.
  - W[2]-hard [folklore].

Approximation

- 96/95-approximation is NP-hard [Chlebík and Chlebíková '02].
- 1.39-approximation algorithm [Byrka et al. '13].

We study the parameter $p = |V(T) \setminus R|$ – good problem for **parameterized approximation**.

Our Results

Existence of

1. Efficient parameterized approximation scheme – it returns $(1+\varepsilon)$-approximation in time $f(p,\varepsilon) \cdot poly(n)$.

Our Results

Existence of

1. Efficient parameterized approximation scheme – it returns $(1+\varepsilon)$-approximation in time $f(p,\varepsilon)\cdot poly(n)$.
2. Polynomial size approximate kernelization scheme.

Our Results

Existence of

1. Efficient parameterized approximation scheme – it returns $(1+\varepsilon)$-approximation in time $f(p,\varepsilon)\cdot poly(n)$.
2. Polynomial size approximate kernelization scheme.

	Unweighted		Weighted	
Undirected	✓	✓	✓	✓
Directed	✓	✗ *	✗ **	✗ **

* Unless NP $\subseteq$ coNP/poly.
** Unless FPT$=$ W[2].

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.

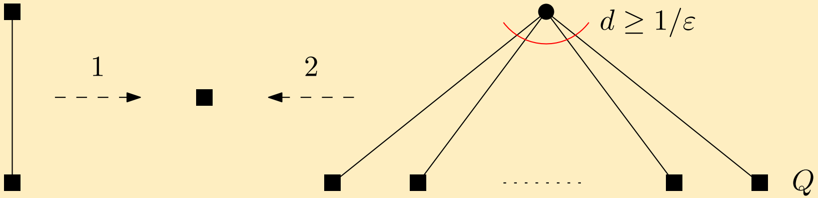
Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

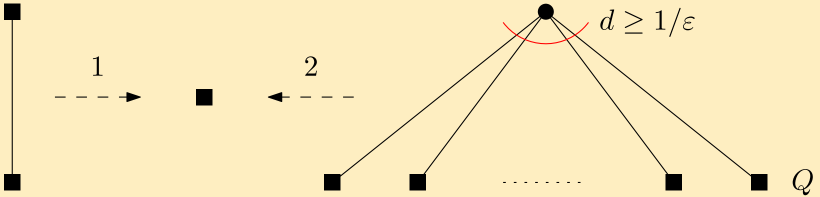
Undirected Unweighted Case



Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case

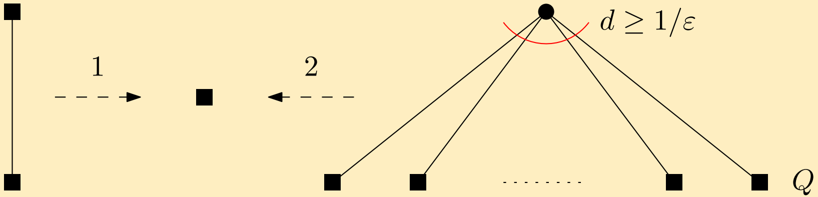


Reduction Rule 1: We can assume any such edge is in the optimal solution.

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case

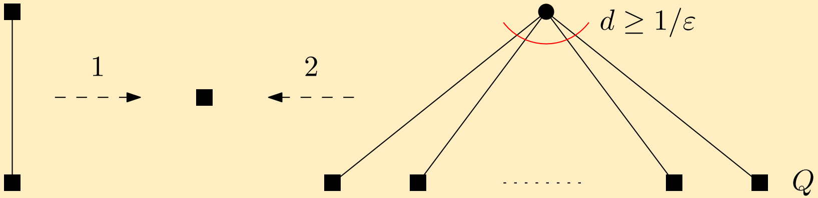


Reduction Rule 2: The optimal solution uses at least d edges to connect terminals in Q. Our solution uses at most $d + 1$ edges.

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case



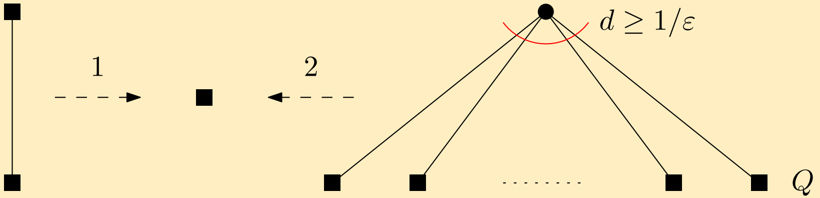
Reduction Rule 2: The optimal solution uses at least d edges to connect terminals in Q. Our solution uses at most $d + 1$ edges.

$$\frac{\text{ALG}}{\text{OPT}}$$

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case



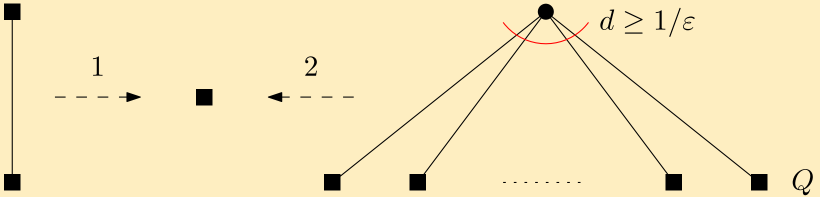
Reduction Rule 2: The optimal solution uses at least d edges to connect terminals in Q. Our solution uses at most $d + 1$ edges.

$$\frac{\text{ALG}}{\text{OPT}} = \frac{d+1}{d}$$

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case



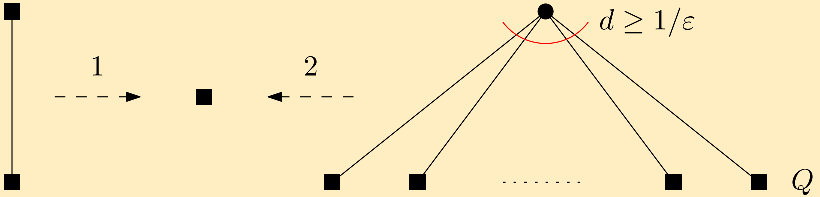
Reduction Rule 2: The optimal solution uses at least d edges to connect terminals in Q. Our solution uses at most $d+1$ edges.

$$\frac{\text{ALG}}{\text{OPT}} = \frac{d+1}{d} = 1 + \frac{1}{d}$$

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case



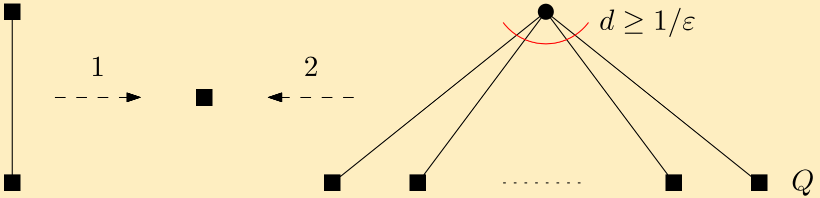
Reduction Rule 2: The optimal solution uses at least d edges to connect terminals in Q. Our solution uses at most $d + 1$ edges.

$$\frac{\text{ALG}}{\text{OPT}} = \frac{d+1}{d} = 1 + \frac{1}{d} \leq 1 + \varepsilon.$$

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case

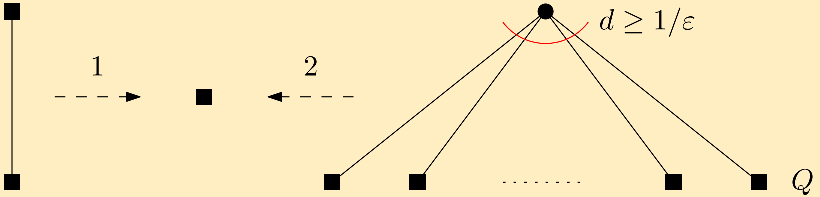


- If we can not apply any rule:

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case

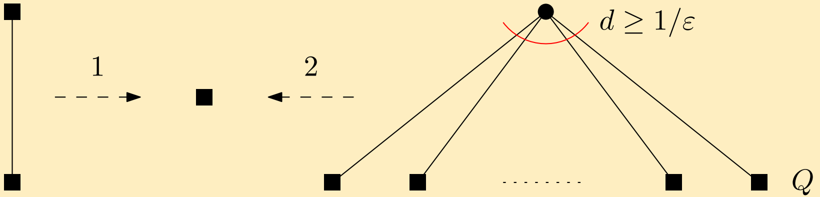


- If we can not apply any rule:
  1. There are only edges terminal–Steiner vertex.

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case

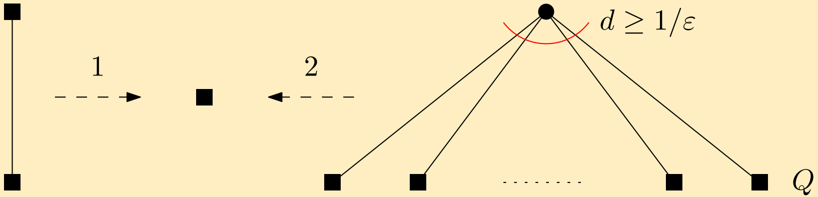


- If we can not apply any rule:
  1. There are only edges terminal–Steiner vertex.
  2. Each Steiner vertex is adjacent to at most $1/\varepsilon$ terminals.

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case

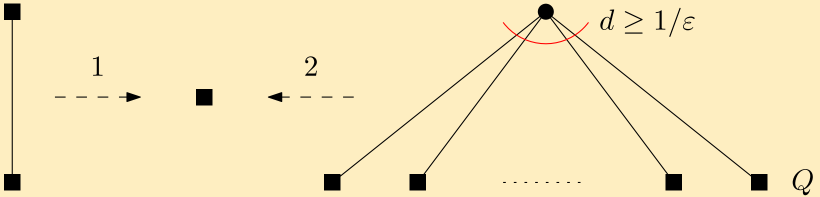


- If we can not apply any rule:
  1. There are only edges terminal–Steiner vertex.
  2. Each Steiner vertex is adjacent to at most $1/\varepsilon$ terminals.
  3. There is a Steiner tree with at most p Steiner vertices.

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case

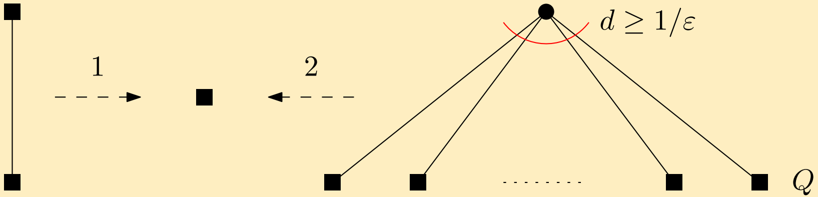


- If we can not apply any rule:
  1. There are only edges terminal–Steiner vertex.
  2. Each Steiner vertex is adjacent to at most $1/\varepsilon$ terminals.
  3. There is a Steiner tree with at most p Steiner vertices.
  4. There are at most p/ε terminals.

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Unweighted Case

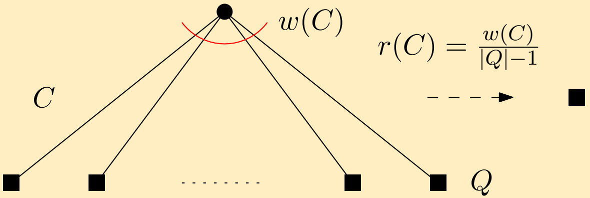


- The directed unweighted case is similar.

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

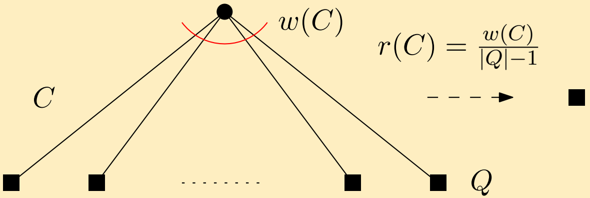
Undirected Weight Case



Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Weight Case

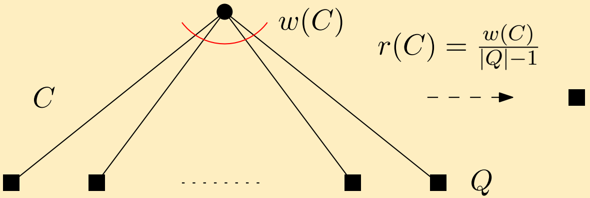


- In each step t we contract a star C_t of the smallest ratio $r(C_t)$.

Main Algorithmic Idea

1. Reduce the number of terminals under some bound $f(p, \varepsilon)$.
2. Use some existing algorithm or kernel for the parameter $|R|$.

Undirected Weight Case



- In each step t we contract a star C_t of the smallest ratio $r(C_t)$.

Thank you for your attention!