Approximation Algorithms

Lecture 3: SteinerTree and MultiwayCut

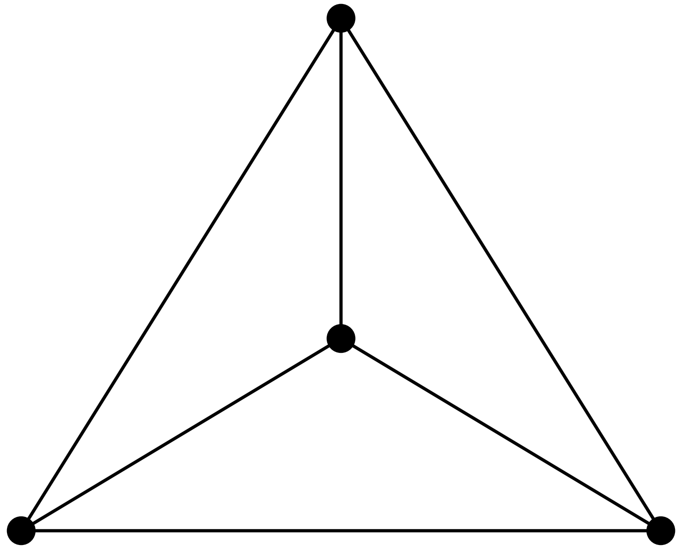
Part I: SteinerTree

Joachim Spoerhase

Winter 2021/22

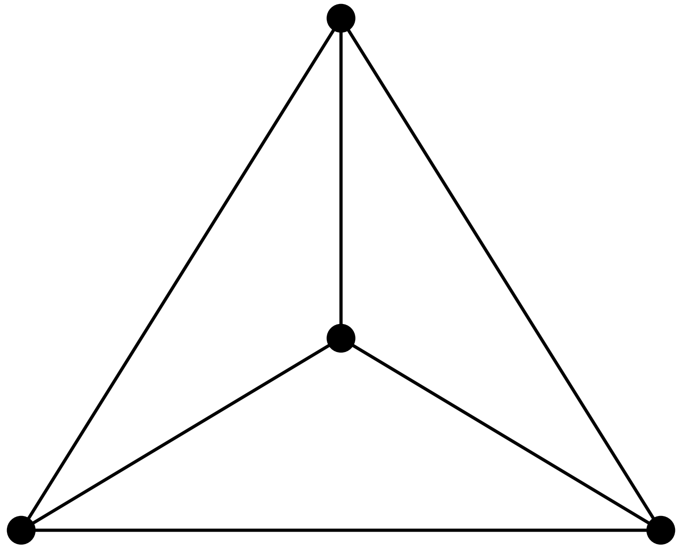
SteinerTree

Given: A graph $G = (V, E)$



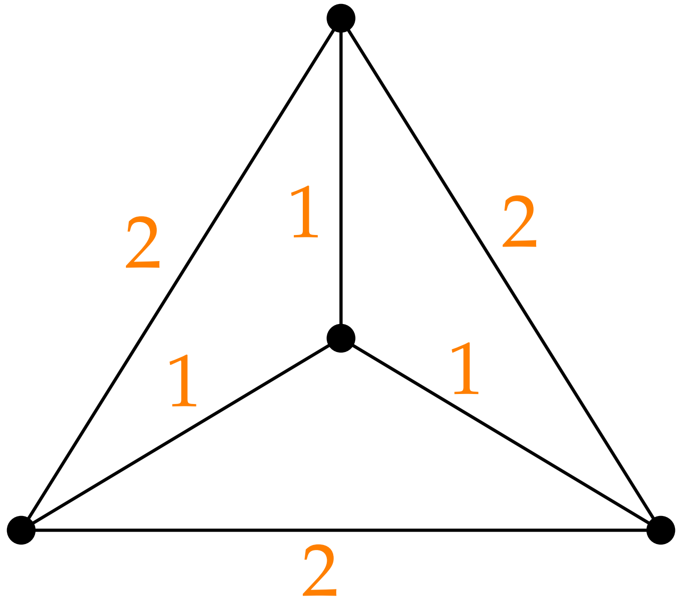
SteinerTree

Given: A graph $G = (V, E)$ with edge weights $c\colon E \to \mathbb{Q}^+$



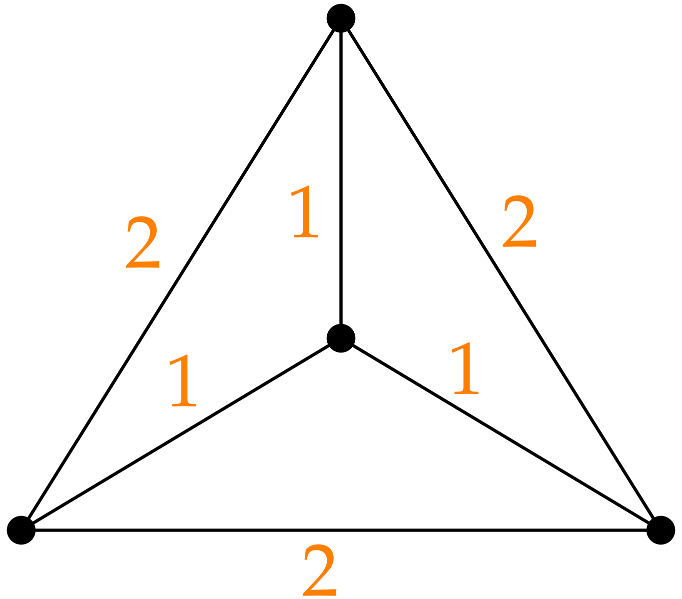
SteinerTree

Given: A graph $G = (V, E)$ with edge weights $c\colon E \to \mathbb{Q}^+$



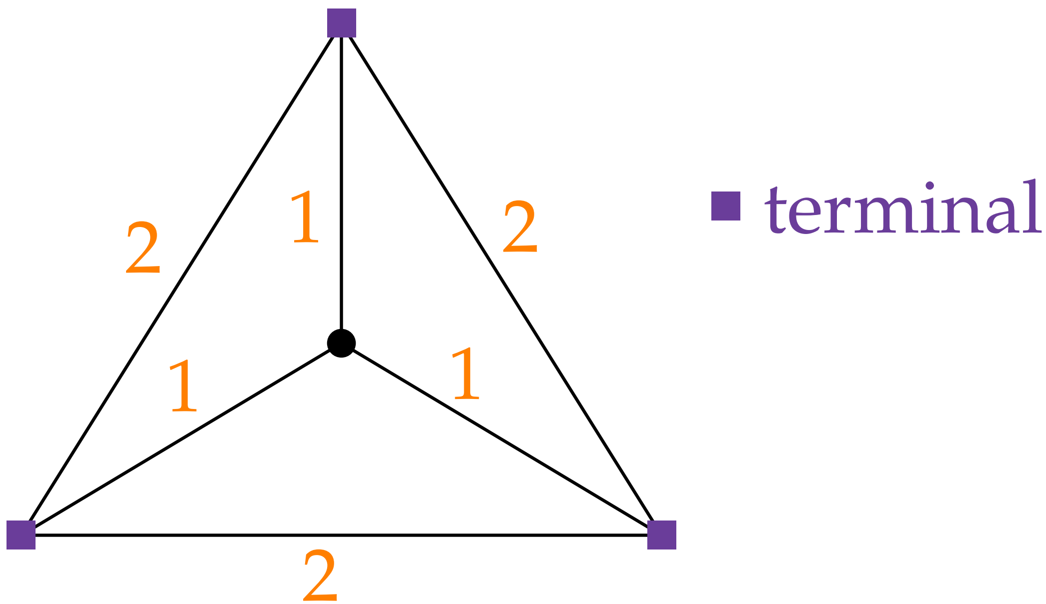
SteinerTree

Given: A graph $G = (V, E)$ with edge weights $c\colon E \to \mathbb{Q}^+$ and a partition of V into a set T of **terminals** and a set S of **Steiner vertices**.



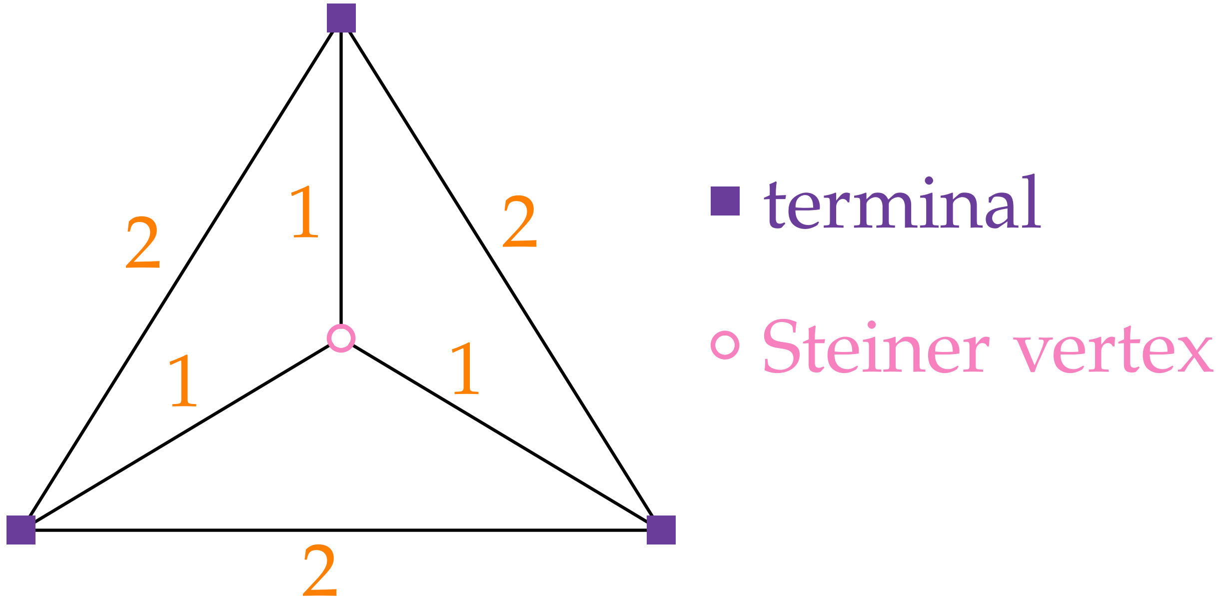
SteinerTree

Given: A graph $G = (V, E)$ with edge weights $c\colon E \to \mathbb{Q}^+$ and a partition of V into a set T of **terminals** and a set S of **Steiner vertices**.



SteinerTree

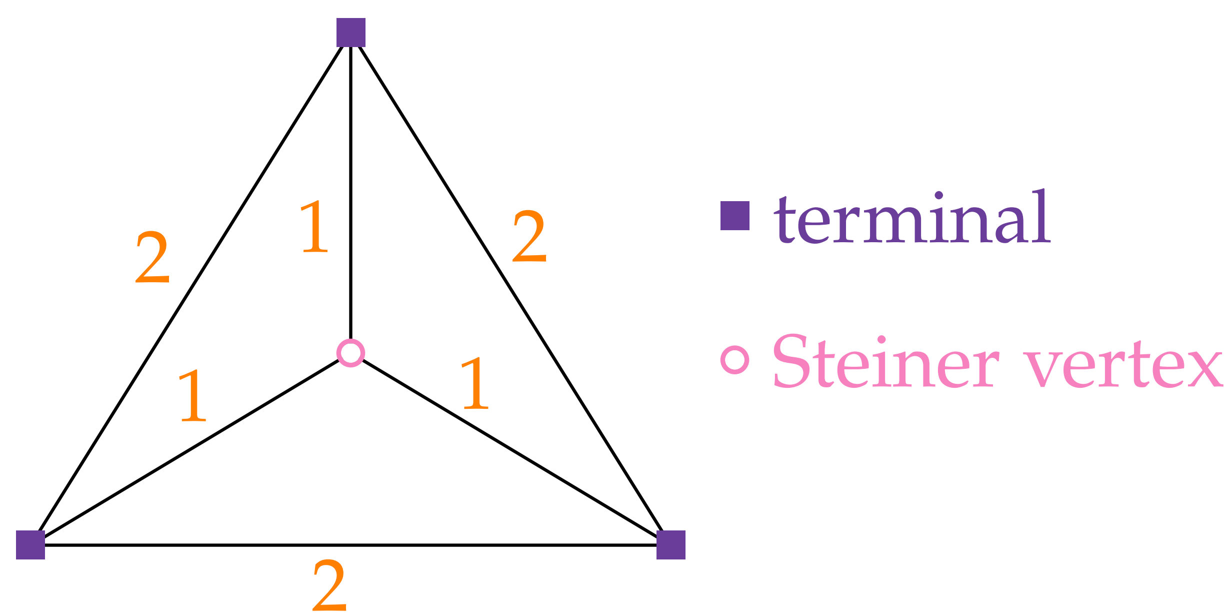
Given: A graph $G = (V, E)$ with edge weights $c \colon E \to \mathbb{Q}^+$ and a partition of V into a set T of **terminals** and a set S of **Steiner vertices**.



SteinerTree

Given: A graph $G = (V, E)$ with edge weights $c\colon E \to \mathbb{Q}^+$ and a partition of V into a set T of **terminals** and a set S of **Steiner vertices**.

Find: A subtree $B = (V', E')$ of G that contains all terminals, i.e., $T \subseteq V'$.

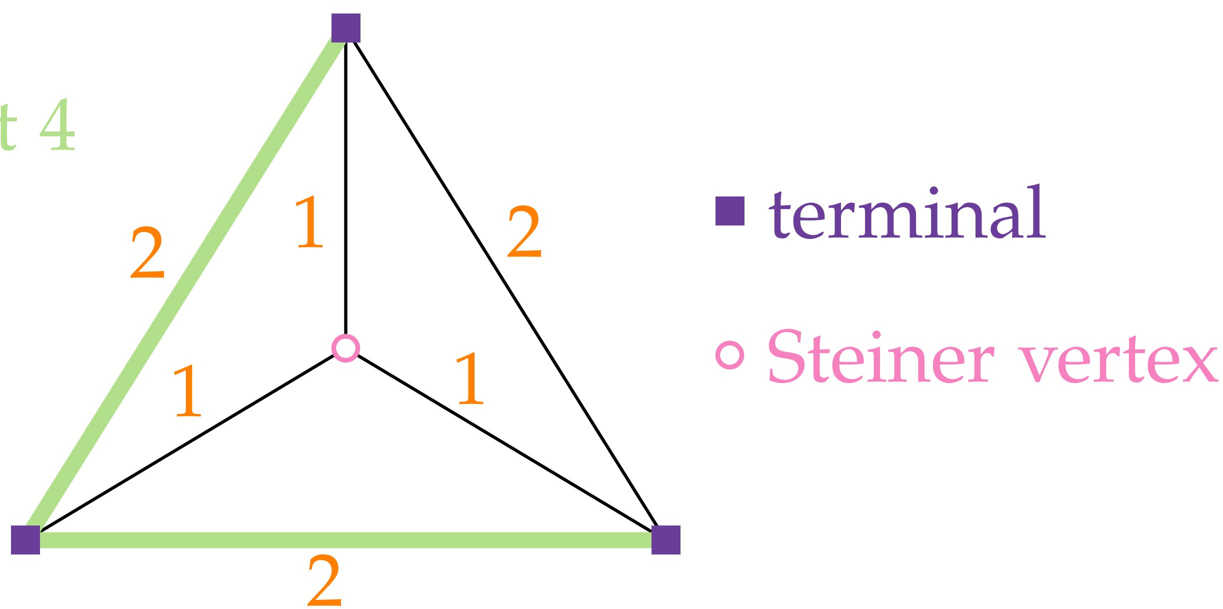


SteinerTree

Given: A graph $G = (V, E)$ with edge weights $c \colon E \to \mathbb{Q}^+$ and a partition of V into a set T of **terminals** and a set S of **Steiner vertices**.

Find: A subtree $B = (V', E')$ of G that contains all terminals, i.e., $T \subseteq V'$

valid solution with cost 4

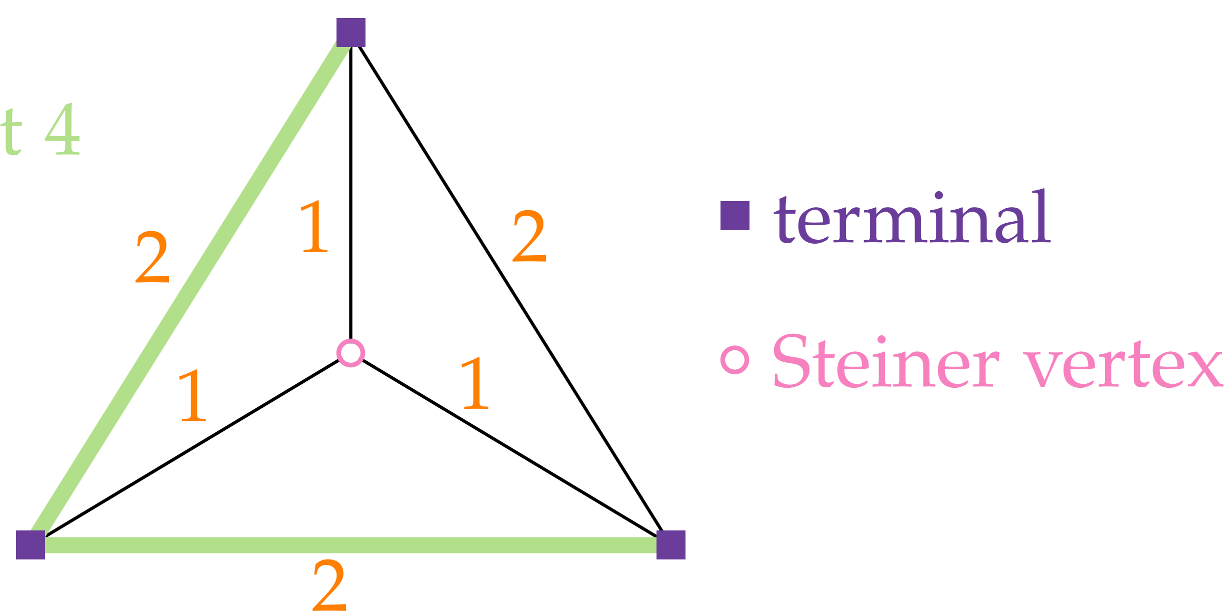


SteinerTree

Given: A graph $G = (V, E)$ with edge weights $c\colon E \to \mathbb{Q}^+$ and a partition of V into a set T of **terminals** and a set S of **Steiner vertices**.

Find: A subtree $B = (V', E')$ of G that contains all terminals, i.e., $T \subseteq V'$, and has minimum cost $c(E') := \sum_{e \in E'} c(e)$ among all subtrees with this property.

valid solution with cost 4



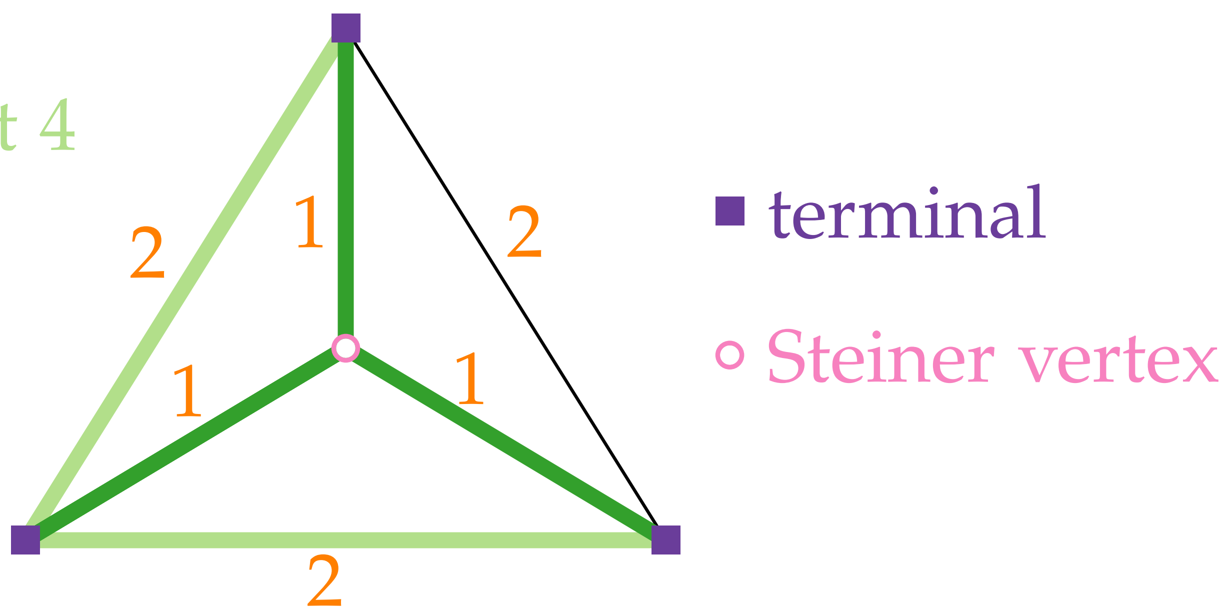
SteinerTree

Given: A graph $G = (V, E)$ with edge weights $c\colon E \to \mathbb{Q}^+$ and a partition of V into a set T of **terminals** and a set S of **Steiner vertices**.

Find: A subtree $B = (V', E')$ of G that contains all terminals, i.e., $T \subseteq V'$, and has minimum cost $c(E') := \sum_{e \in E'} c(e)$ among all subtrees with this property.

valid solution with cost 4

optimum solution with cost 3



MetricSteinerTree

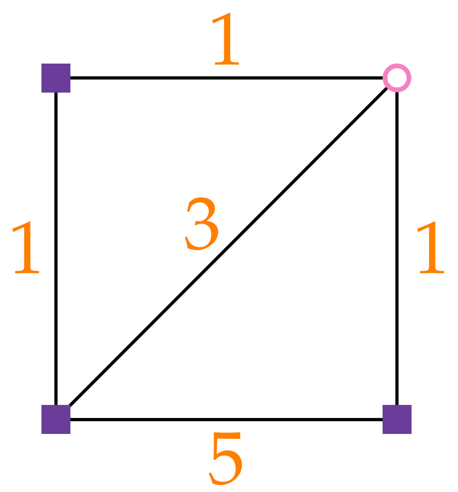
Restriction of SteinerTree where the graph G is complete and the cost function is **metric**

MetricSteinerTree

Restriction of SteinerTree where the graph G is complete and the cost function is **metric**, i.e., for every triple u, v, w of vertices, we have $c(u, w) \leq c(u, v) + c(v, w)$.

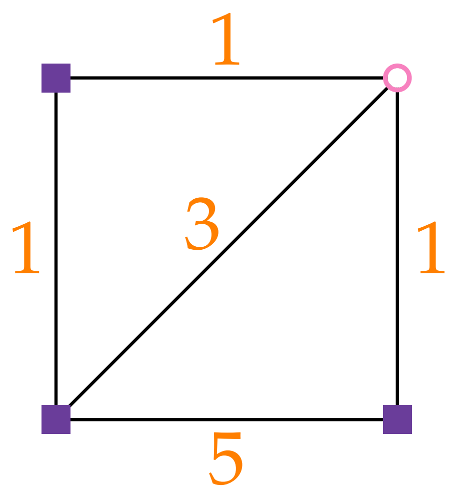
MetricSteinerTree

Restriction of SteinerTree where the graph G is complete and the cost function is **metric**, i.e., for every triple u, v, w of vertices, we have $c(u,w) \leq c(u,v) + c(v,w)$.



METRICSTEINERTREE

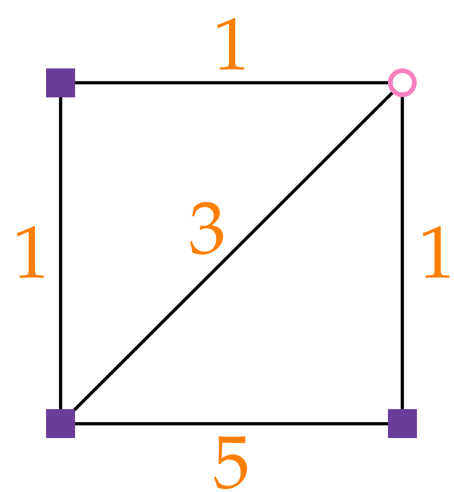
Restriction of STEINERTREE where the graph G is complete and the cost function is **metric**, i.e., for every triple u, v, w of vertices, we have $c(u, w) \leq c(u, v) + c(v, w)$.



not complete

MetricSteinerTree

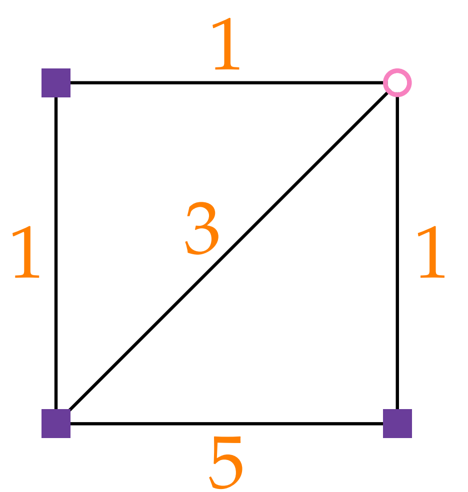
Restriction of SteinerTree where the graph G is complete and the cost function is **metric**, i.e., for every triple u, v, w of vertices, we have $c(u, w) \leq c(u, v) + c(v, w)$.



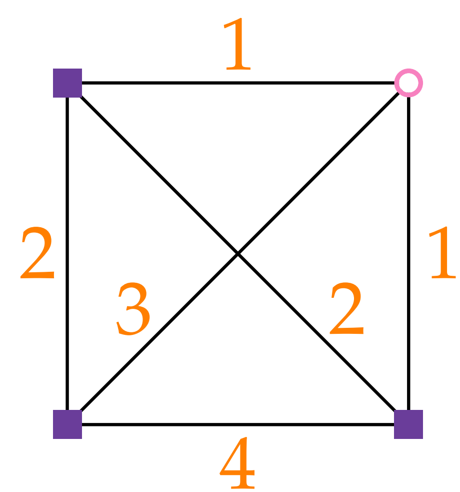
not complete
not metric

METRICSTEINERTREE

Restriction of STEINERTREE where the graph G is complete and the cost function is **metric**, i.e., for every triple u, v, w of vertices, we have $c(u,w) \leq c(u,v) + c(v,w)$.

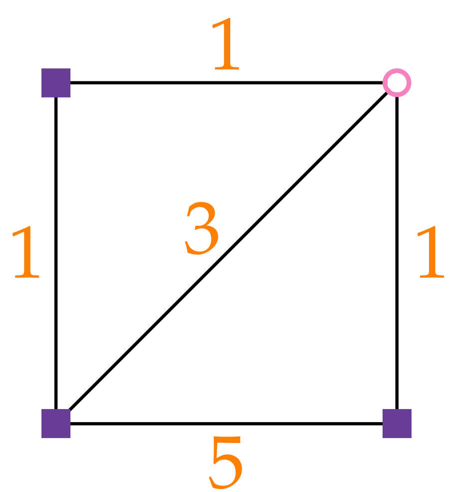


not complete
not metric

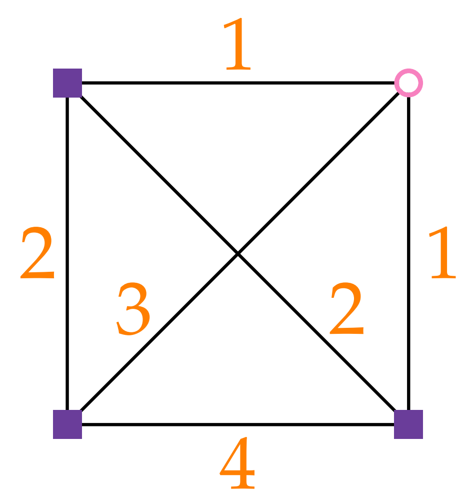


MetricSteinerTree

Restriction of SteinerTree where the graph G is complete and the cost function is **metric**, i.e., for every triple u, v, w of vertices, we have $c(u, w) \leq c(u, v) + c(v, w)$.



not complete
not metric



complete
metric

Approximation Algorithms

Lecture 3:
SteinerTree and MultiwayCut

Part II:
Approximation Preserving Reduction

Joachim Spoerhase

Winter 2021/22

Approximation Preserving Reduction

Let Π_1, Π_2 be minimization problems.

Approximation Preserving Reduction

Let Π_1, Π_2 be minimization problems.

problems Π_1 Π_2

Approximation Preserving Reduction

Let Π_1, Π_2 be minimization problems. An **approximation preserving reduction** from Π_1 to Π_2 ist a tuple (f, g) of poly-time computable functions with the following properties.

problems Π_1 Π_2

Approximation Preserving Reduction

Let Π_1, Π_2 be minimization problems. An **approximation preserving reduction** from Π_1 to Π_2 ist a tuple (f, g) of poly-time computable functions with the following properties.

(i) For each instance I_1 of Π_1, $I_2 := f(I_1)$ is an instance of Π_2 with $\mathrm{OPT}_{\Pi_2}(I_2) \leq \mathrm{OPT}_{\Pi_1}(I_1)$.

problems Π_1 Π_2

Approximation Preserving Reduction

Let Π_1, Π_2 be minimization problems. An **approximation preserving reduction** from Π_1 to Π_2 ist a tuple (f, g) of poly-time computable functions with the following properties.

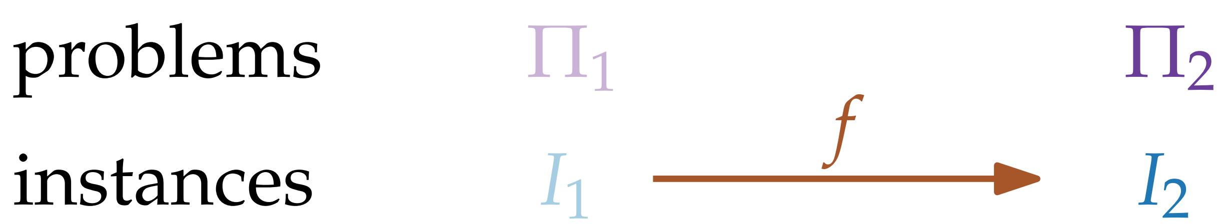
(i) For each instance I_1 of Π_1, $I_2 := f(I_1)$ is an instance of Π_2 with $\mathrm{OPT}_{\Pi_2}(I_2) \leq \mathrm{OPT}_{\Pi_1}(I_1)$.

problems	Π_1	Π_2
instances	I_1	

Approximation Preserving Reduction

Let Π_1, Π_2 be minimization problems. An **approximation preserving reduction** from Π_1 to Π_2 ist a tuple (f, g) of poly-time computable functions with the following properties.

(i) For each instance I_1 of Π_1, $I_2 := f(I_1)$ is an instance of Π_2 with $\mathrm{OPT}_{\Pi_2}(I_2) \leq \mathrm{OPT}_{\Pi_1}(I_1)$.

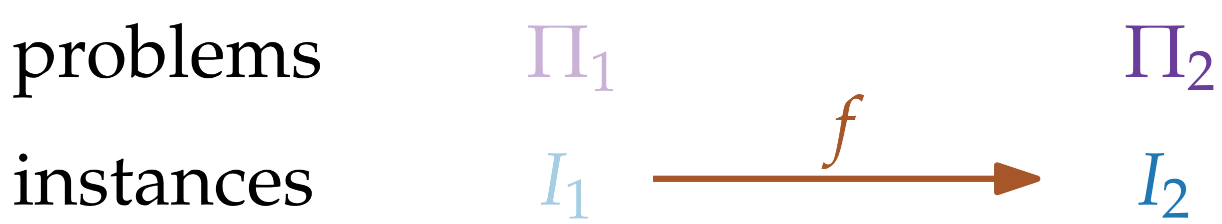


Approximation Preserving Reduction

Let Π_1, Π_2 be minimization problems. An **approximation preserving reduction** from Π_1 to Π_2 ist a tuple (f, g) of poly-time computable functions with the following properties.

(i) For each instance I_1 of Π_1, $I_2 := f(I_1)$ is an instance of Π_2 with $\mathrm{OPT}_{\Pi_2}(I_2) \le \mathrm{OPT}_{\Pi_1}(I_1)$.

(ii) For each feasible solution t of I_2, $s := g(I_1, t)$ is a feasible solution of I_1 with $\mathrm{obj}_{\Pi_1}(I_1, s) \le \mathrm{obj}_{\Pi_2}(I_2, t)$.

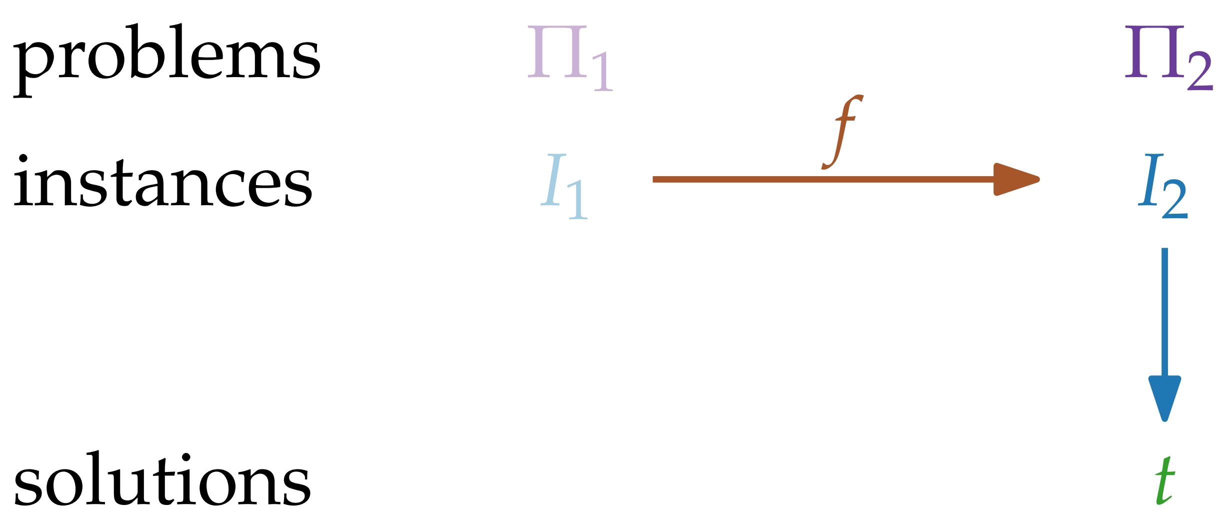


Approximation Preserving Reduction

Let Π_1, Π_2 be minimization problems. An **approximation preserving reduction** from Π_1 to Π_2 ist a tuple (f, g) of poly-time computable functions with the following properties.

(i) For each instance I_1 of Π_1, $I_2 := f(I_1)$ is an instance of Π_2 with $\mathrm{OPT}_{\Pi_2}(I_2) \leq \mathrm{OPT}_{\Pi_1}(I_1)$.

(ii) For each feasible solution t of I_2, $s := g(I_1, t)$ is a feasible solution of I_1 with $\mathrm{obj}_{\Pi_1}(I_1, s) \leq \mathrm{obj}_{\Pi_2}(I_2, t)$.

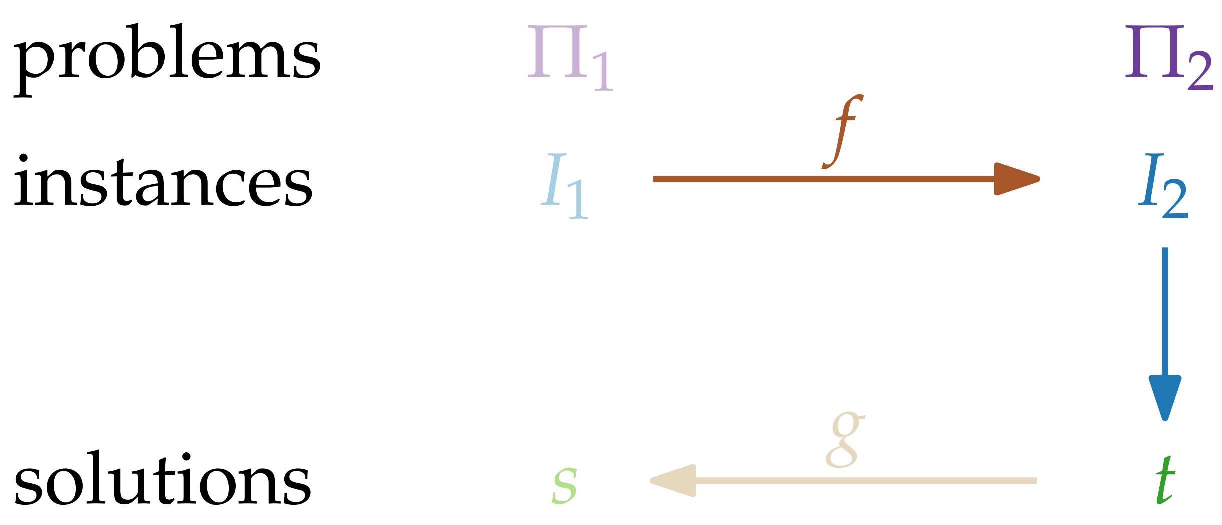


Approximation Preserving Reduction

Let Π_1, Π_2 be minimization problems. An **approximation preserving reduction** from Π_1 to Π_2 ist a tuple (f, g) of poly-time computable functions with the following properties.

(i) For each instance I_1 of Π_1, $I_2 := f(I_1)$ is an instance of Π_2 with $\mathrm{OPT}_{\Pi_2}(I_2) \leq \mathrm{OPT}_{\Pi_1}(I_1)$.

(ii) For each feasible solution t of I_2, $s := g(I_1, t)$ is a feasible solution of I_1 with $\mathrm{obj}_{\Pi_1}(I_1, s) \leq \mathrm{obj}_{\Pi_2}(I_2, t)$.

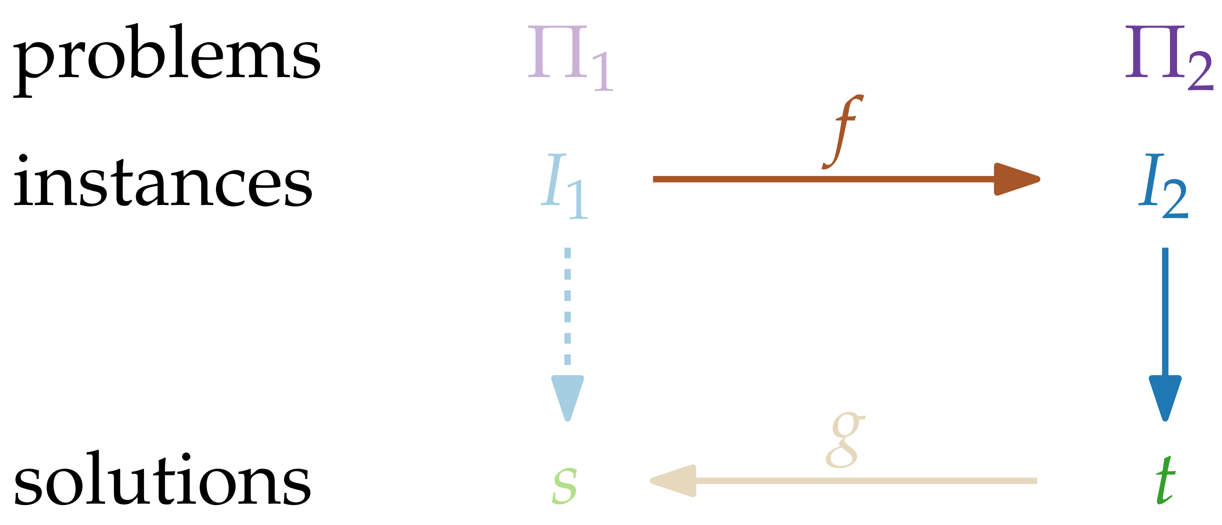


Approximation Preserving Reduction

Let Π_1, Π_2 be minimization problems. An **approximation preserving reduction** from Π_1 to Π_2 ist a tuple (f, g) of poly-time computable functions with the following properties.

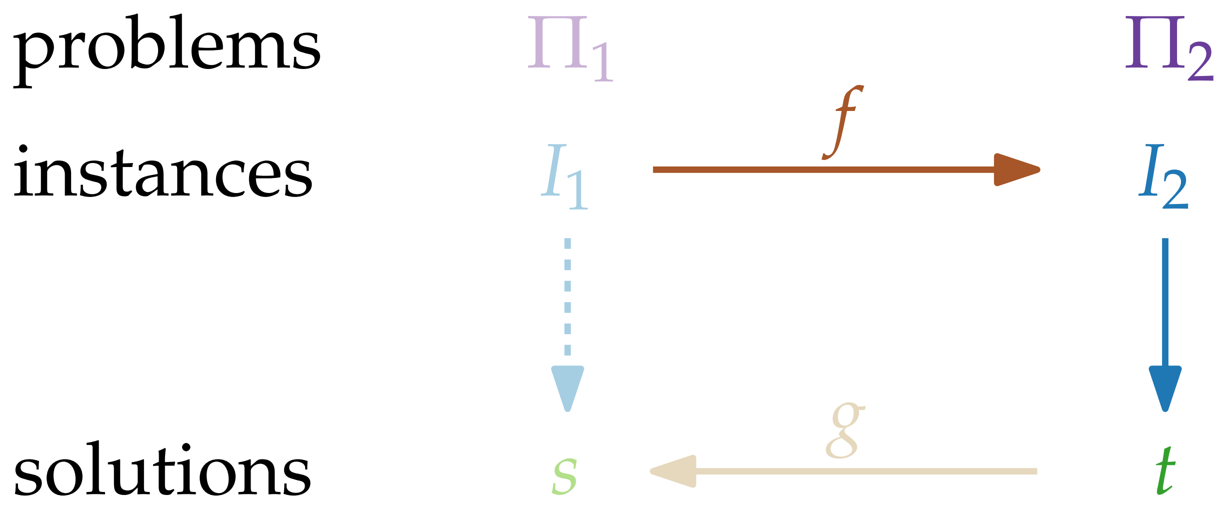
(i) For each instance I_1 of Π_1, $I_2 := f(I_1)$ is an instance of Π_2 with $\mathrm{OPT}_{\Pi_2}(I_2) \leq \mathrm{OPT}_{\Pi_1}(I_1)$.

(ii) For each feasible solution t of I_2, $s := g(I_1, t)$ is a feasible solution of I_1 with $\mathrm{obj}_{\Pi_1}(I_1, s) \leq \mathrm{obj}_{\Pi_2}(I_2, t)$.



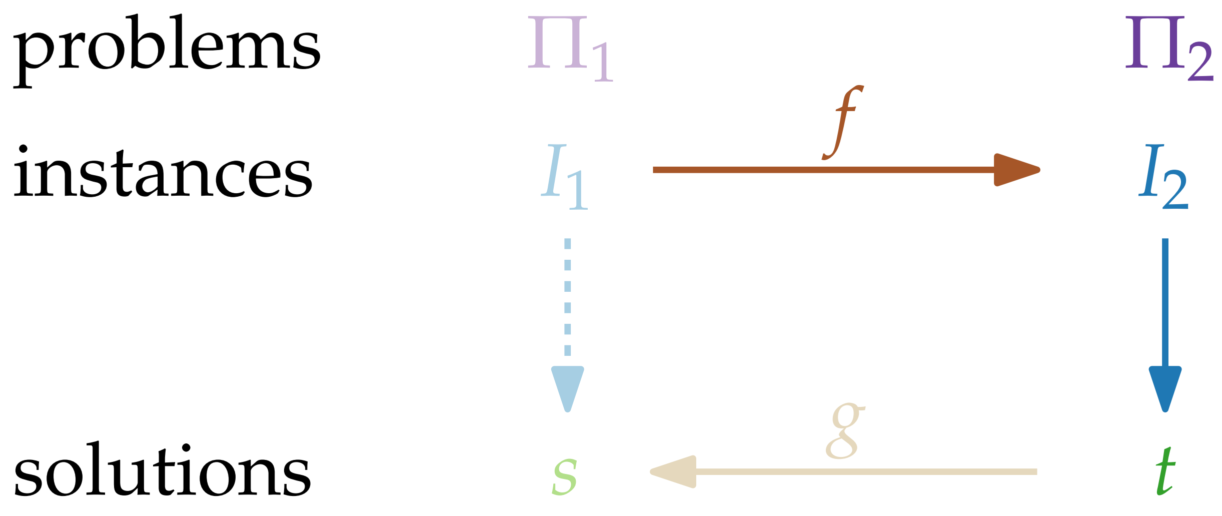
Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2.



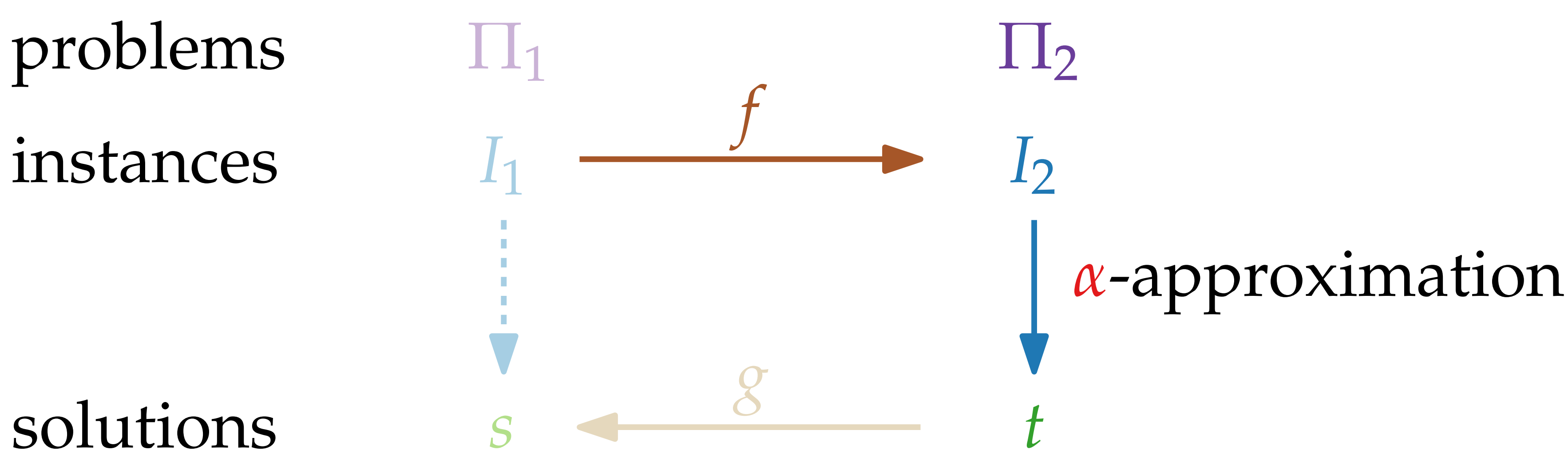
Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor ? approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.



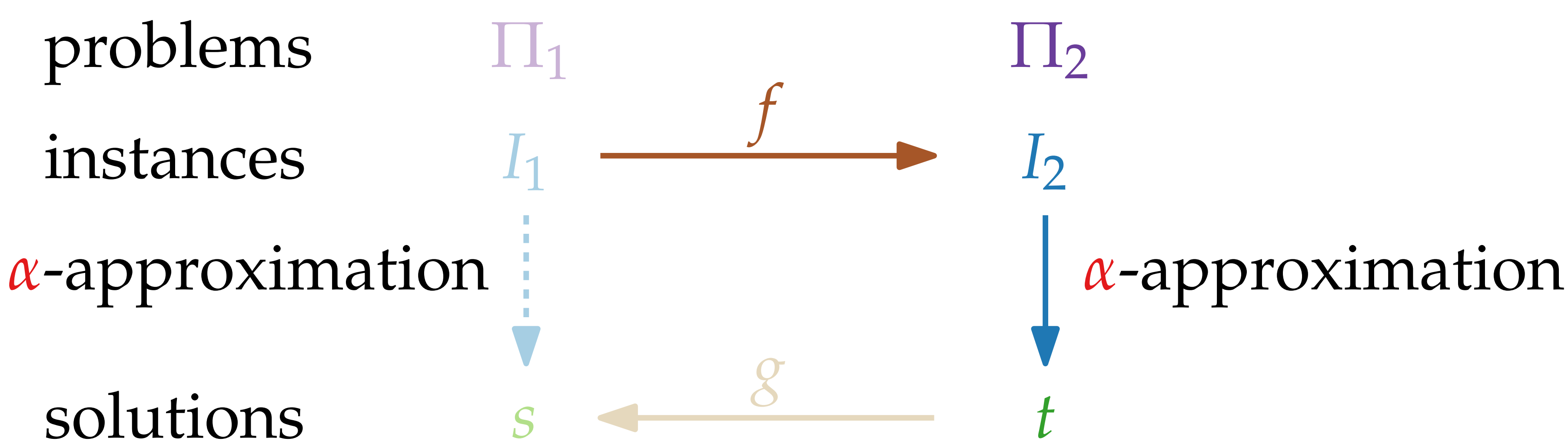
Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor **?** approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.



Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor-α-approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.

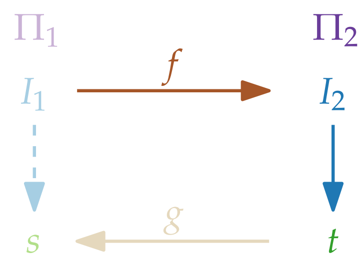


Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor-α-approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.

Proof.

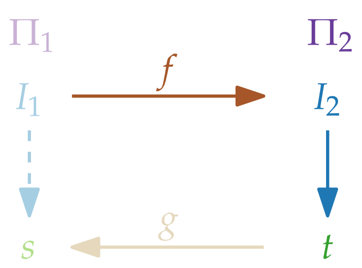
Let A be a factor-α-approx. alg. for Π_2.



Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor-α-approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.

Proof.



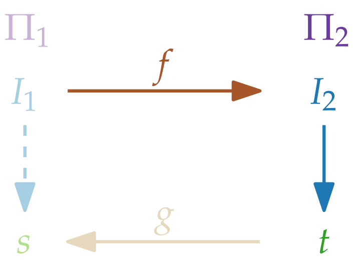
Let A be a factor-α-approx. alg. for Π_2.

Let I_1 be an instance of Π_1.

Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor-α-approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.

Proof.



Let A be a factor-α-approx. alg. for Π_2.

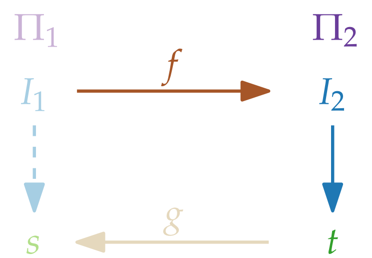
Let I_1 be an instance of Π_1.

Set $I_2 :=$, $t :=$ and $s :=$.

Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor-α-approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.

Proof.



Let A be a factor-α-approx. alg. for Π_2.

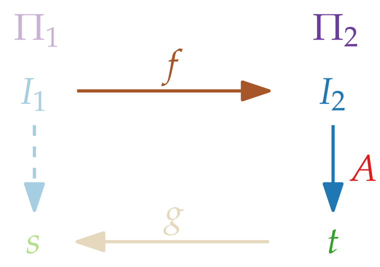
Let I_1 be an instance of Π_1.

Set $I_2 := f(I_1)$, $t :=$ and $s :=$.

Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor-α-approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.

Proof.



Let A be a factor-α-approx. alg. for Π_2.

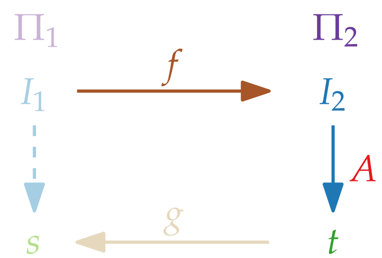
Let I_1 be an instance of Π_1.

Set $I_2 := f(I_1)$, $t := A(I_2)$ and $s :=$.

Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor-α-approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.

Proof.



Let A be a factor-α-approx. alg. for Π_2.

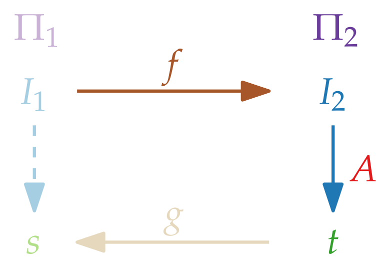
Let I_1 be an instance of Π_1.

Set $I_2 := f(I_1)$, $t := A(I_2)$ and $s := g(I_1, t)$.

Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor-α-approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.

Proof.



Let A be a factor-α-approx. alg. for Π_2.

Let I_1 be an instance of Π_1.

Set $I_2 := f(I_1)$, $t := A(I_2)$ and $s := g(I_1, t)$.

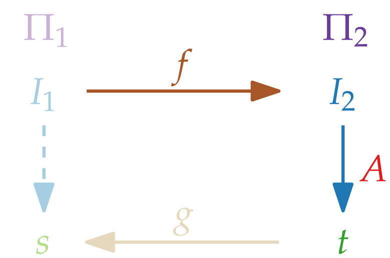
Then:

$\text{obj}_{\Pi_1}(I_1, s)$

Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor-α-approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.

Proof.



Let A be a factor-α-approx. alg. for Π_2.

Let I_1 be an instance of Π_1.

Set $I_2 := f(I_1)$, $t := A(I_2)$ and $s := g(I_1, t)$.

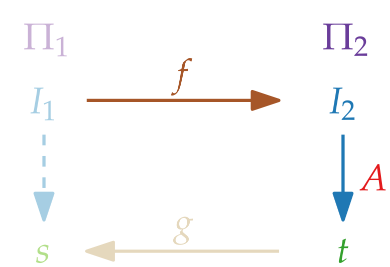
Then:

$\text{obj}_{\Pi_1}(I_1, s) \leq \text{obj}_{\Pi_2}(I_2, t)$

Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor-α-approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.

Proof.



Let A be a factor-α-approx. alg. for Π_2.

Let I_1 be an instance of Π_1.

Set $I_2 := f(I_1)$, $t := A(I_2)$ and $s := g(I_1, t)$.

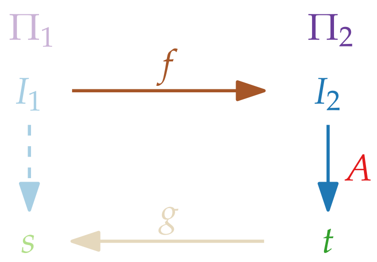
Then:

$$\text{obj}_{\Pi_1}(I_1, s) \leq \text{obj}_{\Pi_2}(I_2, t) \leq \alpha \cdot \text{OPT}_{\Pi_2}(I_2)$$

Approximation Preserving Reduction

Theorem. Let Π_1, Π_2 be minimization problems where there is an approximation preserving reduction (f, g) from Π_1 to Π_2. Then there is a factor-α-approximation algorithm of Π_1 for each factor-α-approximation algorithm of Π_2.

Proof.



Let A be a factor-α-approx. alg. for Π_2.

Let I_1 be an instance of Π_1.

Set $I_2 := f(I_1)$, $t := A(I_2)$ and $s := g(I_1, t)$.

Then:

$$\text{obj}_{\Pi_1}(I_1, s) \leq \text{obj}_{\Pi_2}(I_2, t) \leq \alpha \cdot \text{OPT}_{\Pi_2}(I_2) \leq \alpha \cdot \text{OPT}_{\Pi_1}(I_1)$$

Approximation Algorithms

Lecture 3:
SteinerTree and MultiwayCut

Part III:
Reduction to MetricSteinerTree

Joachim Spoerhase

Winter 2021/22

MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f $\quad I_1 \xrightarrow{f} I_2$

MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f $\quad I_1 \xrightarrow{f} I_2$

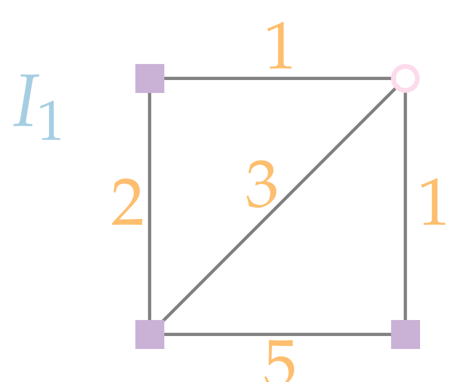
Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \dot\cup S$

MetricSteinerTree

> **Theorem.** There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f $\quad I_1 \xrightarrow{f} I_2$

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \dot\cup S$



MetricSteinerTree

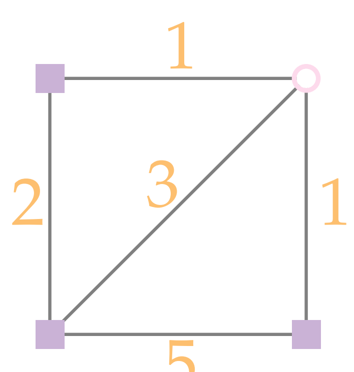
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f $I_1 \xrightarrow{f} I_2$

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \mathbin{\dot\cup} S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

I_1



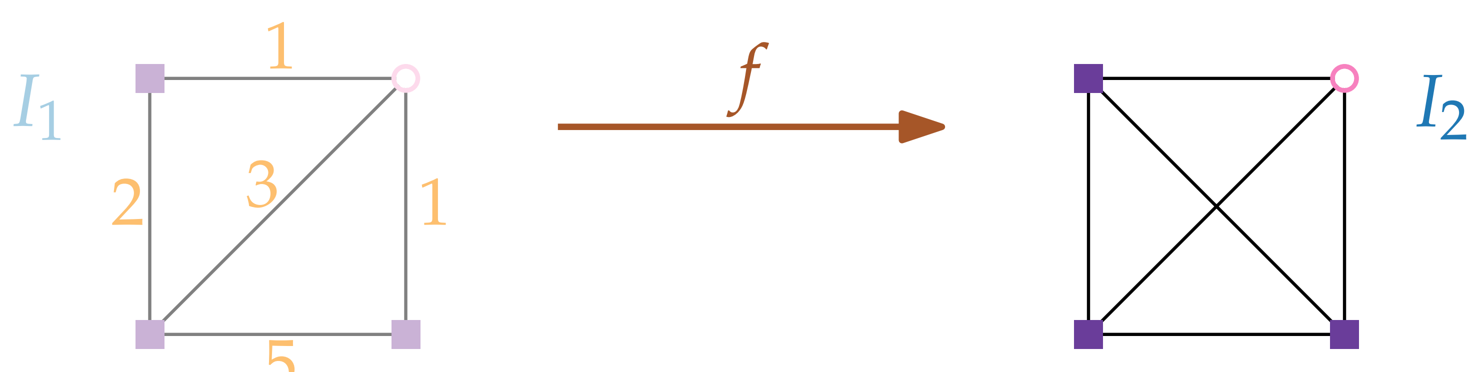
MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f $I_1 \xrightarrow{f} I_2$

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \dot\cup S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1



MetricSteinerTree

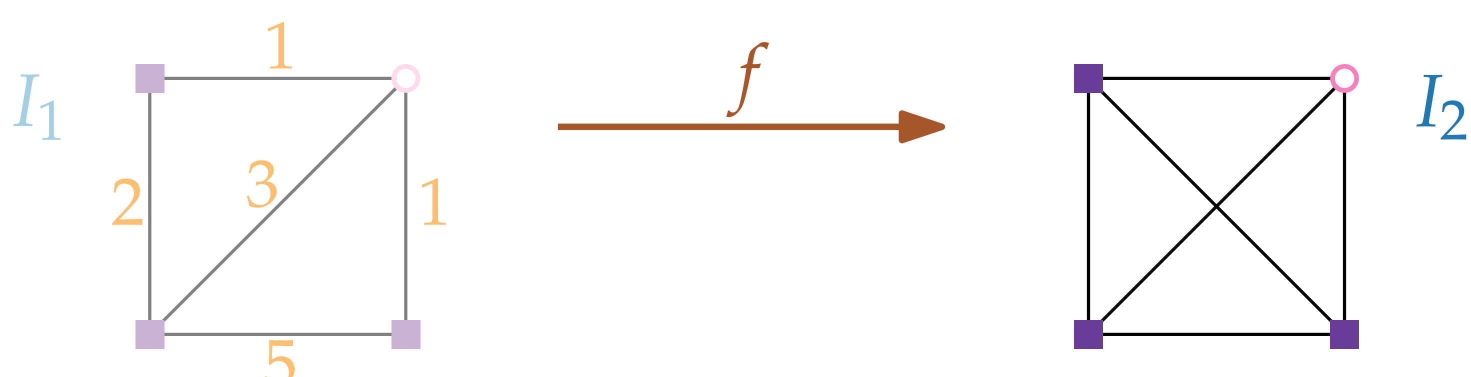
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f $\quad I_1 \xrightarrow{f} I_2$

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \dot\cup S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v) :=$ Length of shortest u–v-path in G_1



MetricSteinerTree

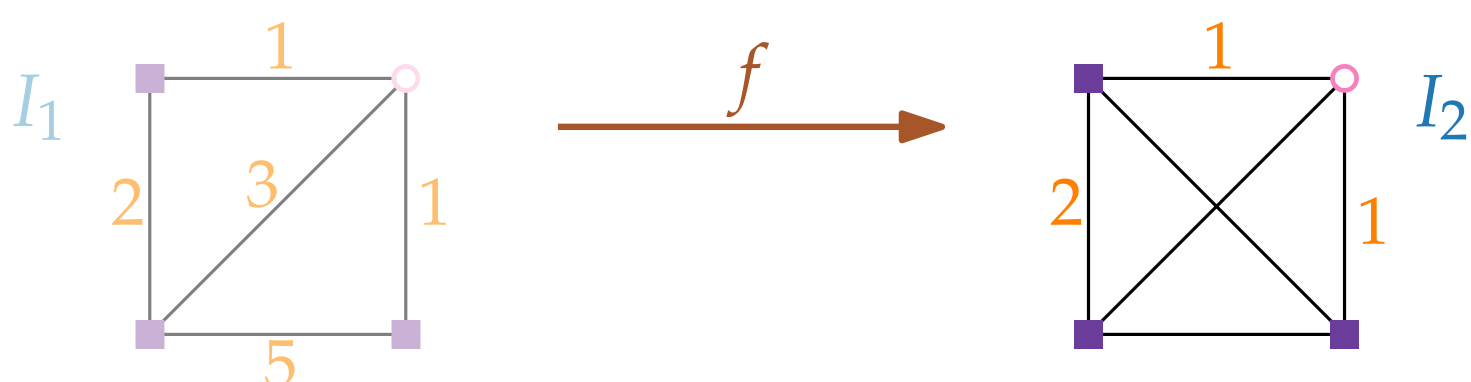
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f $I_1 \xrightarrow{f} I_2$

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \,\dot{\cup}\, S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v) :=$ Length of shortest u–v-path in G_1



METRICSTEINERTREE

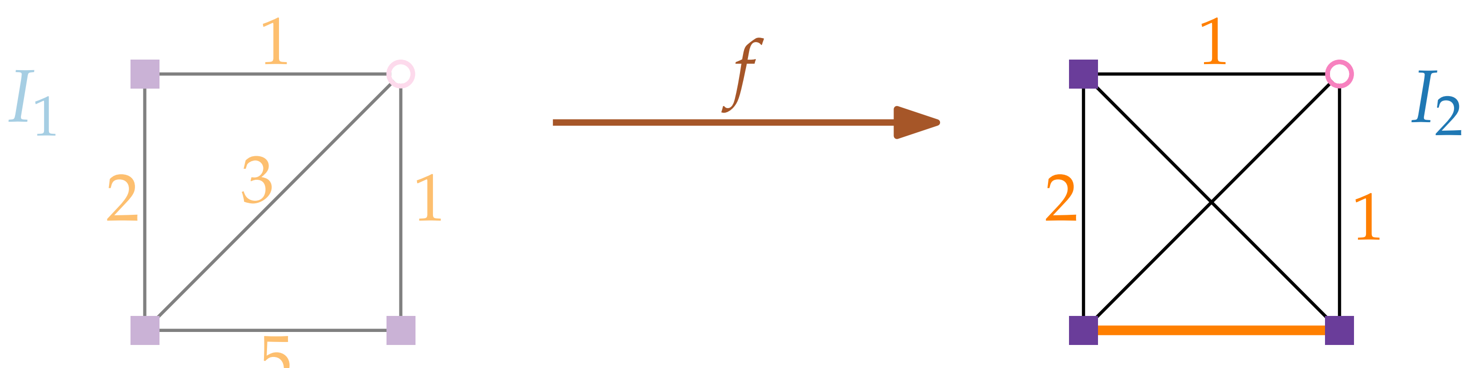
Theorem. There is an approximation preserving reduction from STEINERTREE to METRICSTEINERTREE.

Proof. (1) Mapping f

Instance I_1 of STEINERTREE: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \cup S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v) :=$ Length of shortest u–v-path in G_1



MetricSteinerTree

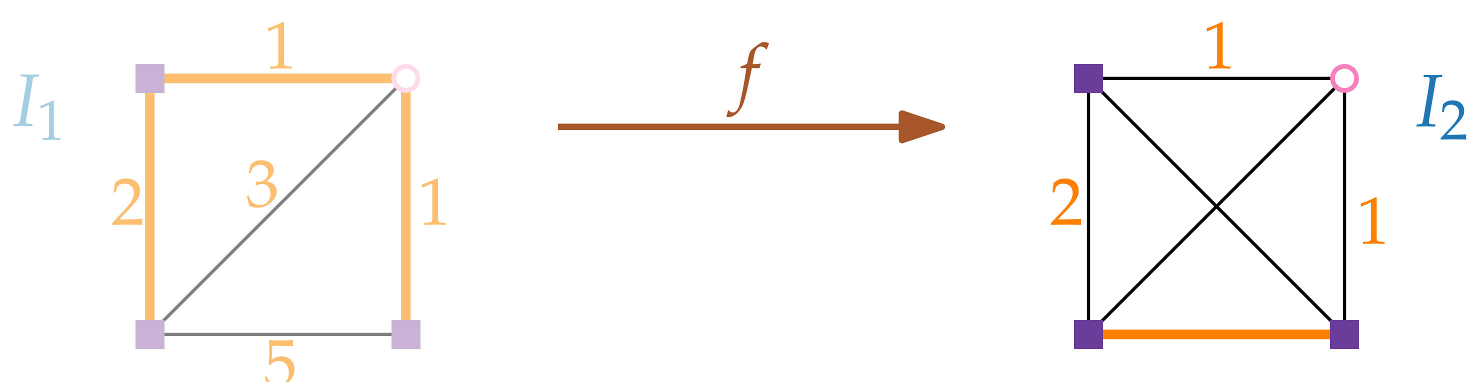
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f $\quad I_1 \xrightarrow{f} I_2$

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \,\dot\cup\, S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v) :=$ Length of shortest u–v-path in G_1



MetricSteinerTree

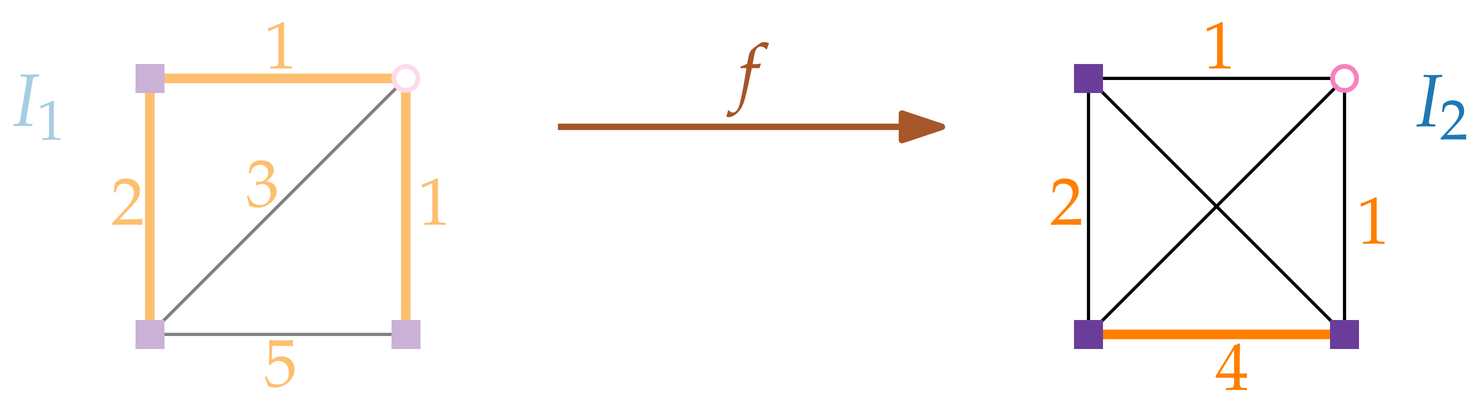
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \dot\cup S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v)$:= Length of shortest u–v-path in G_1



MetricSteinerTree

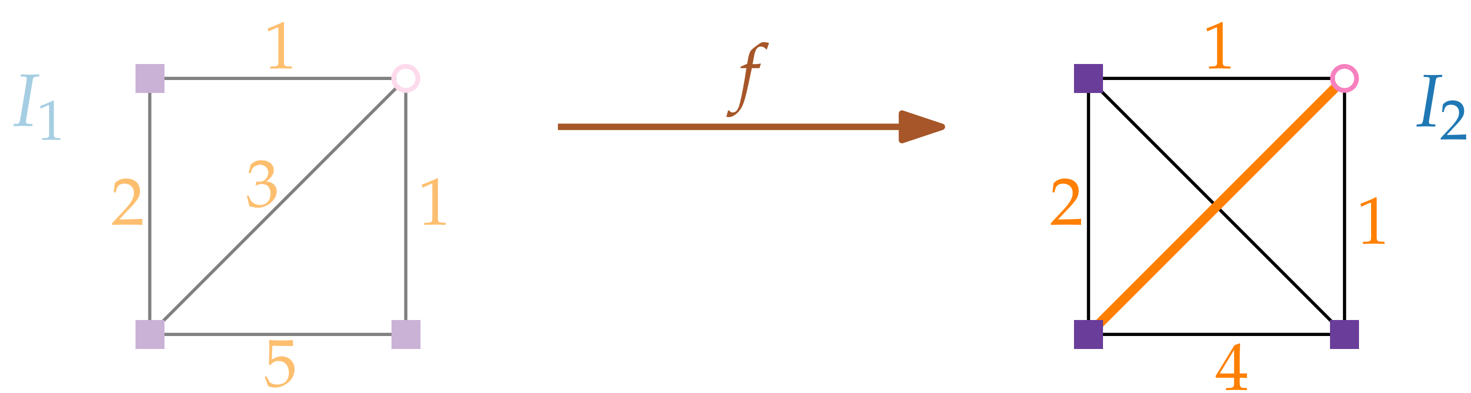
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f $I_1 \xrightarrow{f} I_2$

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \mathbin{\dot\cup} S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v)$:= Length of shortest u–v-path in G_1



MetricSteinerTree

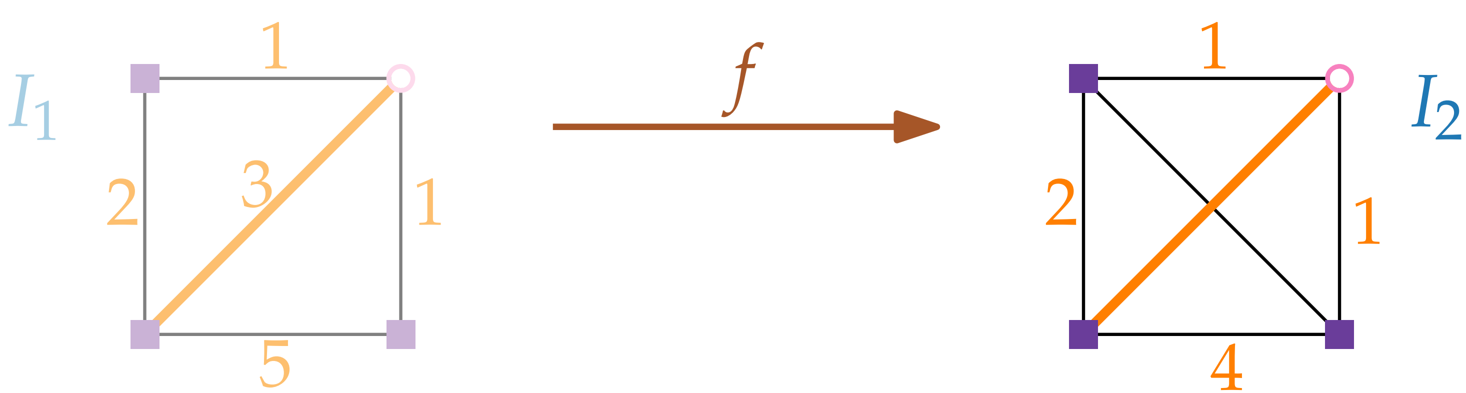
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f $\quad I_1 \xrightarrow{f} I_2$

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \,\dot\cup\, S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v)$:= Length of shortest u–v-path in G_1



MetricSteinerTree

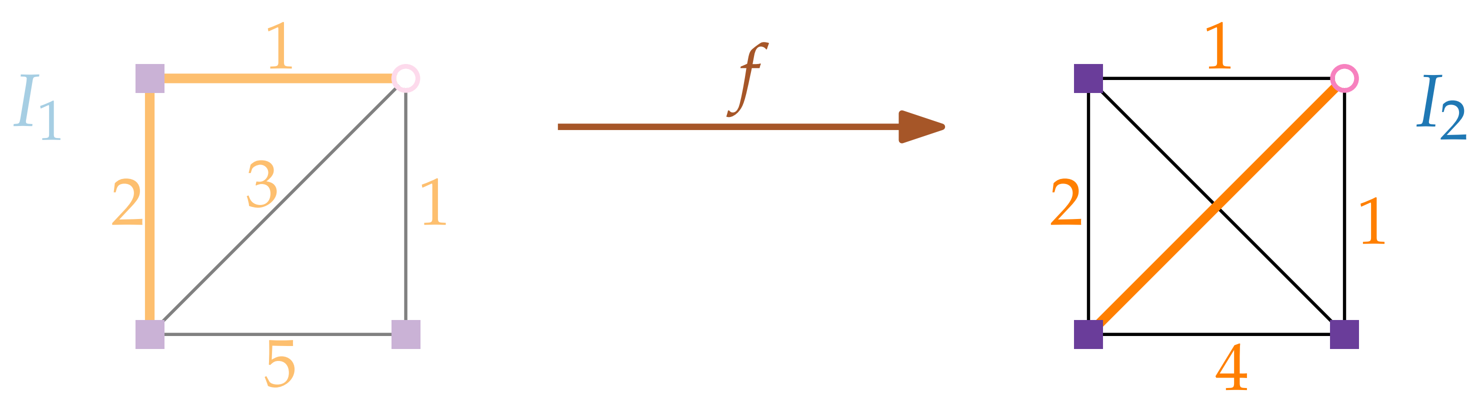
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f $\quad I_1 \xrightarrow{f} I_2$

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \,\dot\cup\, S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v) :=$ Length of shortest u–v-path in G_1



MetricSteinerTree

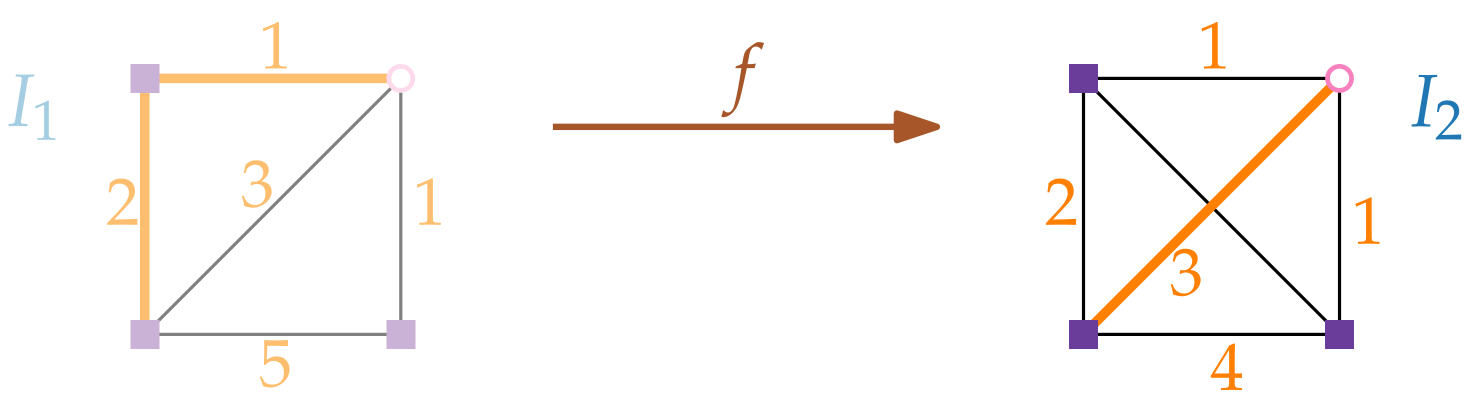
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \,\dot\cup\, S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v) :=$ Length of shortest u–v-path in G_1



METRICSTEINERTREE

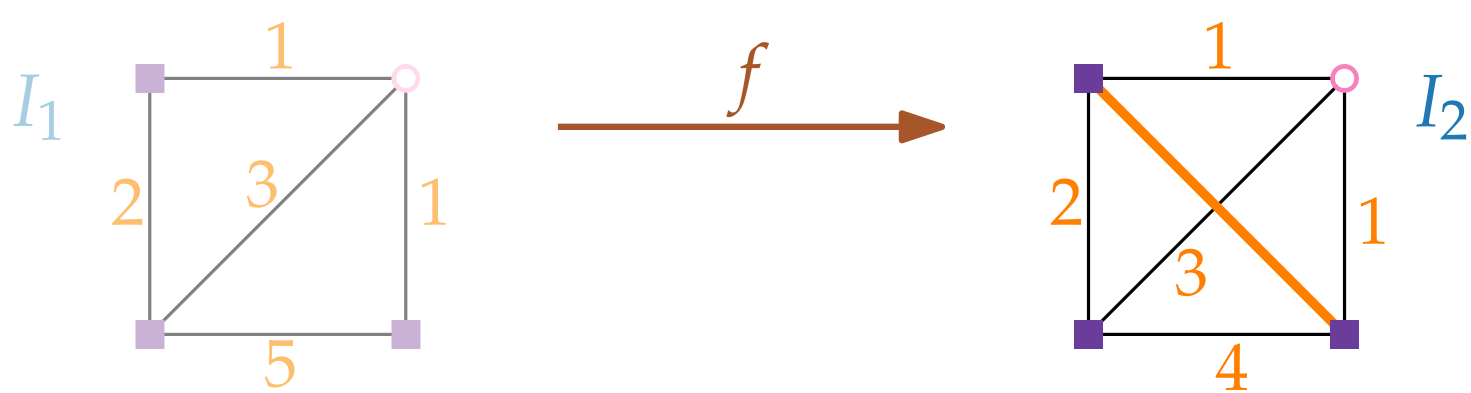
Theorem. There is an approximation preserving reduction from STEINERTREE to METRICSTEINERTREE.

Proof. (1) Mapping f $\quad I_1 \xrightarrow{f} I_2$

Instance I_1 of STEINERTREE: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \,\dot\cup\, S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v) :=$ Length of shortest u–v-path in G_1



MetricSteinerTree

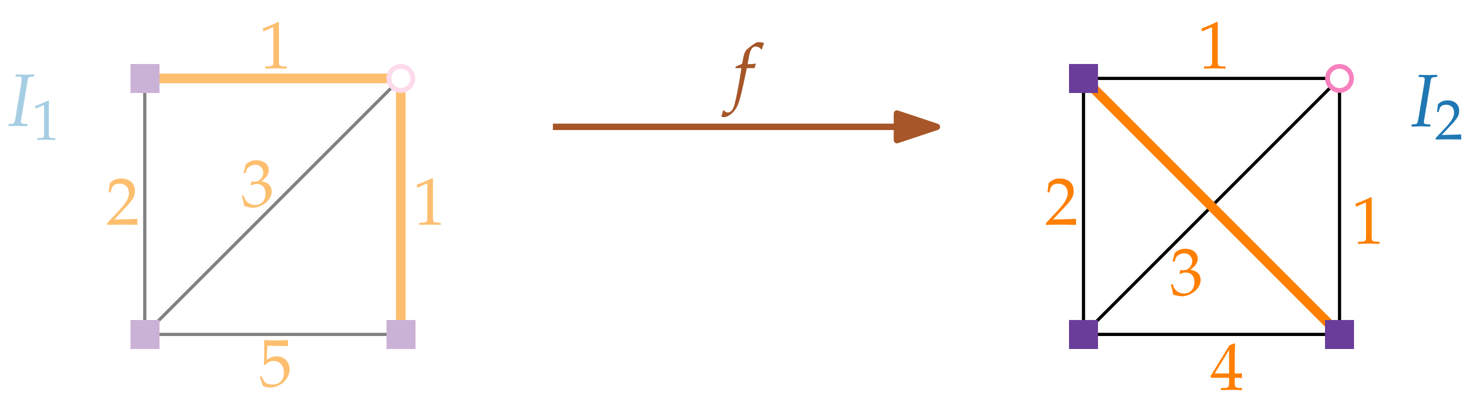
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \,\dot\cup\, S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v)$:= Length of shortest u–v-path in G_1



METRICSTEINERTREE

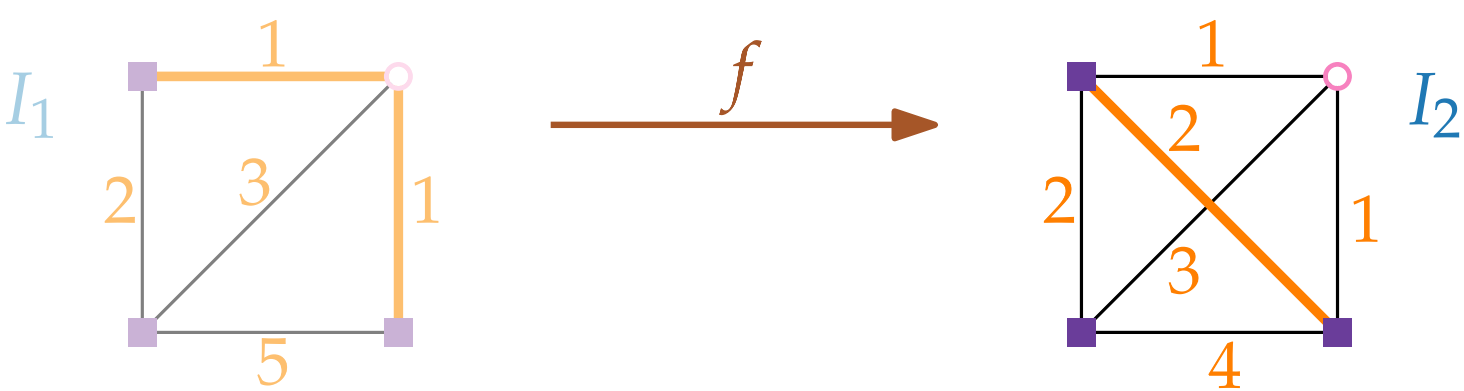
Theorem. There is an approximation preserving reduction from STEINERTREE to METRICSTEINERTREE.

Proof. (1) Mapping f

Instance I_1 of STEINERTREE: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \dot\cup S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v) :=$ Length of shortest u–v-path in G_1



MetricSteinerTree

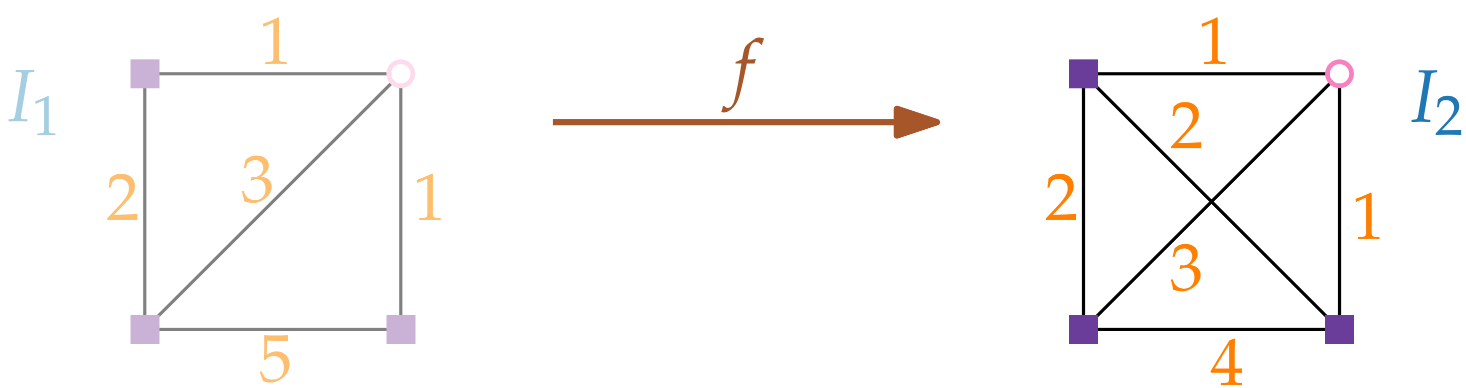
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (1) Mapping f

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \mathbin{\dot\cup} S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v)$:= Length of shortest u–v-path in G_1



MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

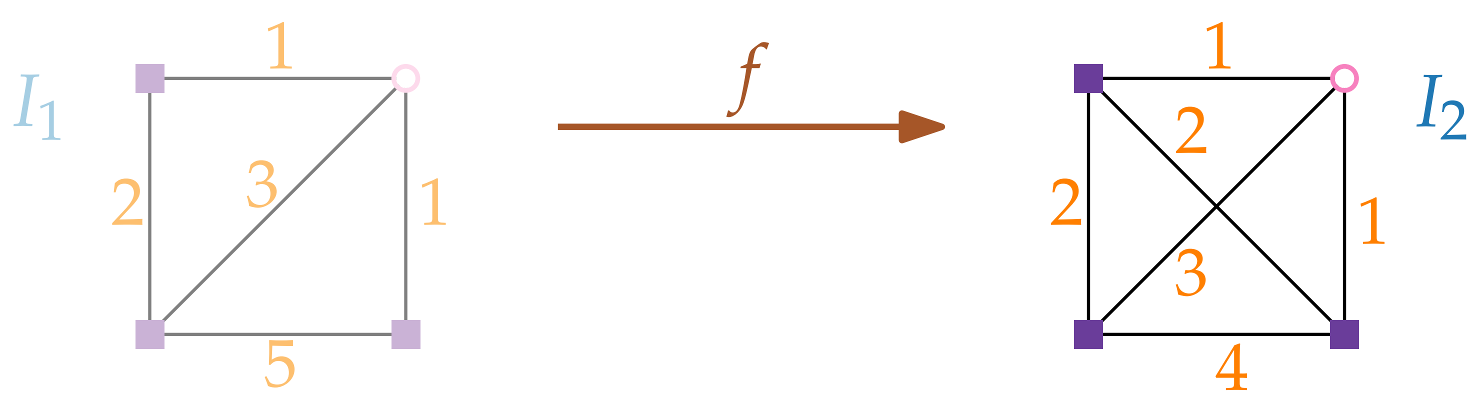
Proof. (1) Mapping f $\quad I_1 \xrightarrow{f} I_2$

Instance I_1 of SteinerTree: Graph $G_1 = (V, E_1)$, edge weights c_1, partition $V = T \,\dot\cup\, S$

Metric instance $I_2 := f(I_1)$: Complete graph $G_2 = (V, E_2)$, partition T, S as in I_1

$c_2(u, v)$:= Length of shortest u–v-path in G_1

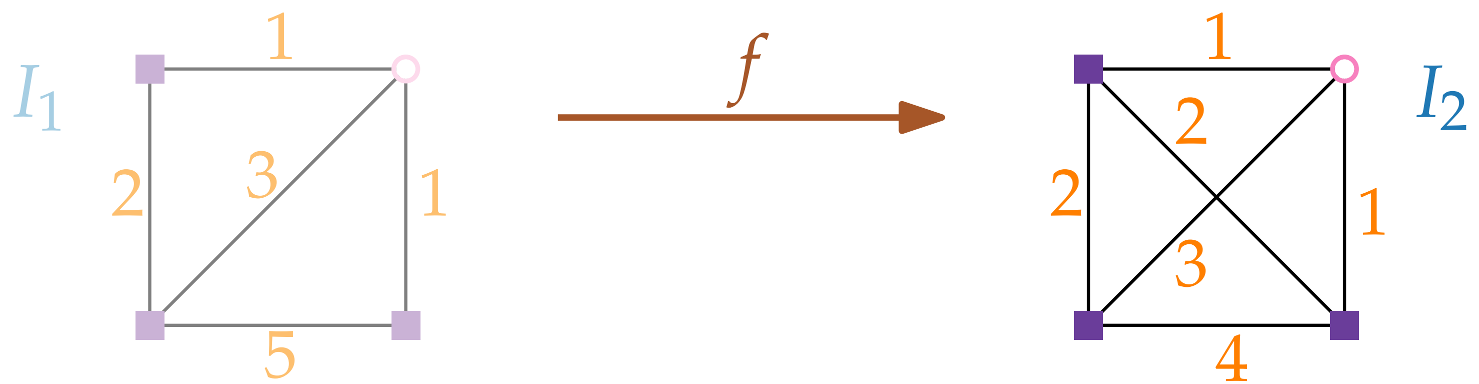
$c_2(u, v) \leq c_1(u, v)$ for all $(u, v) \in E$



MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (2) $\mathrm{OPT}(I_2) \leq \mathrm{OPT}(I_1)$

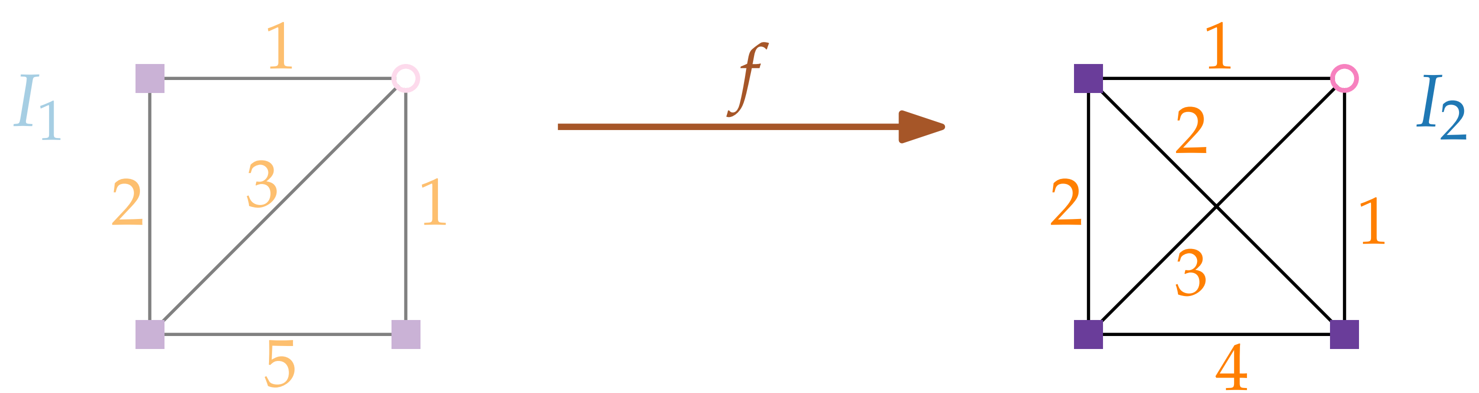


MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (2) $\mathrm{OPT}(I_2) \leq \mathrm{OPT}(I_1)$

Let B^* be optimal Steiner tree for I_1

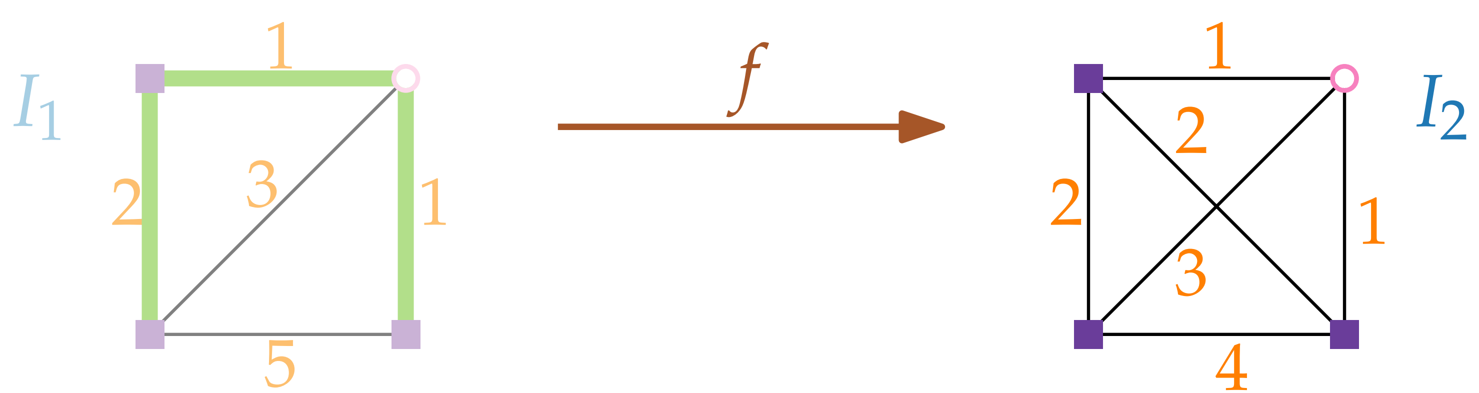


MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (2) $\mathrm{OPT}(I_2) \leq \mathrm{OPT}(I_1)$

Let B^* be optimal Steiner tree for I_1



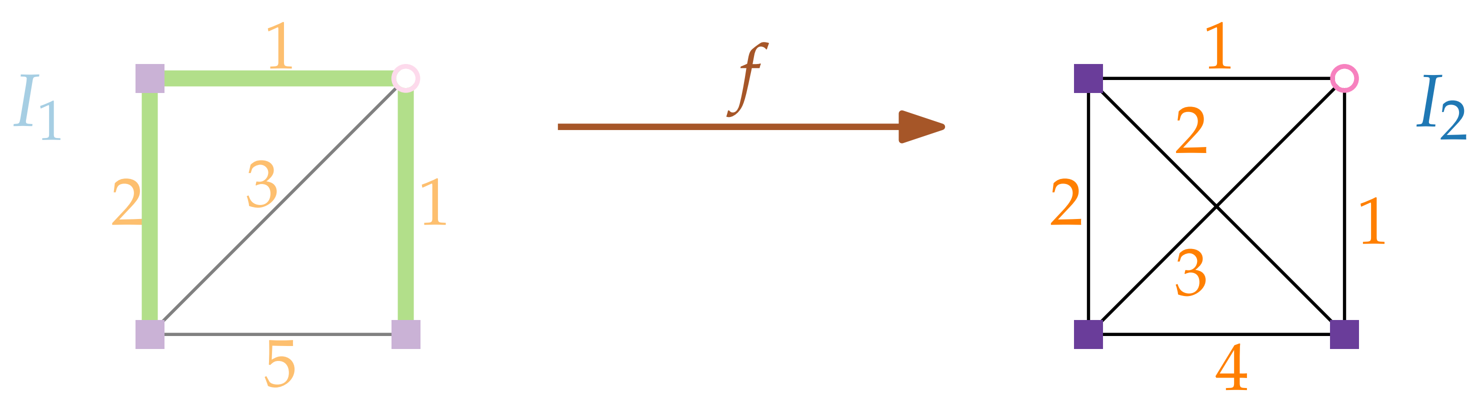
MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (2) $\mathrm{OPT}(I_2) \leq \mathrm{OPT}(I_1)$

Let B^* be optimal Steiner tree for I_1

B^* is also a feasible solution for I_2, since $E_1 \subseteq E_2$ and the vertex sets V, T, S are the same



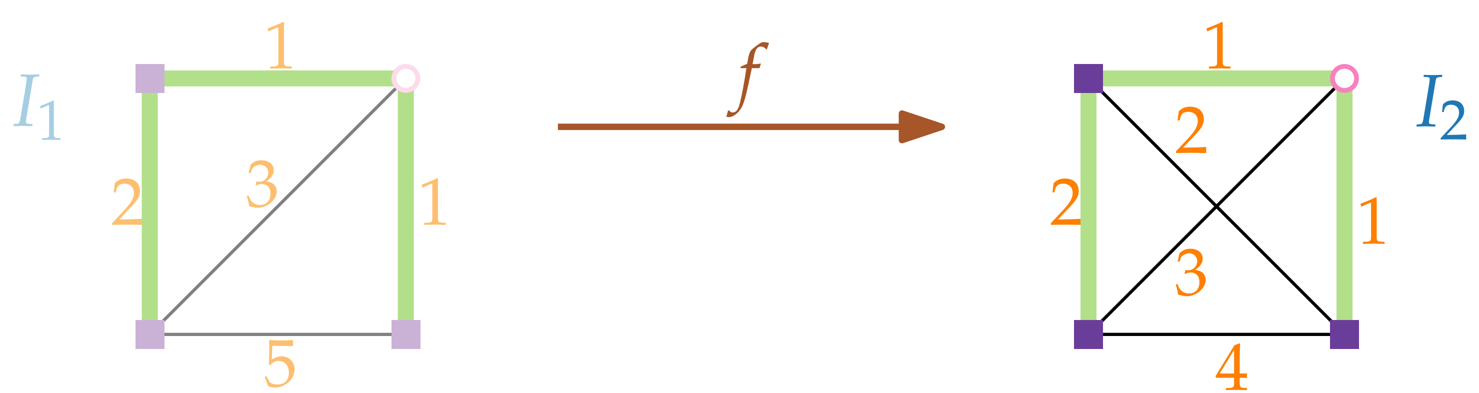
MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (2) $\mathrm{OPT}(I_2) \leq \mathrm{OPT}(I_1)$

Let B^* be optimal Steiner tree for I_1

B^* is also a feasible solution for I_2, since $E_1 \subseteq E_2$ and the vertex sets V, T, S are the same



MetricSteinerTree

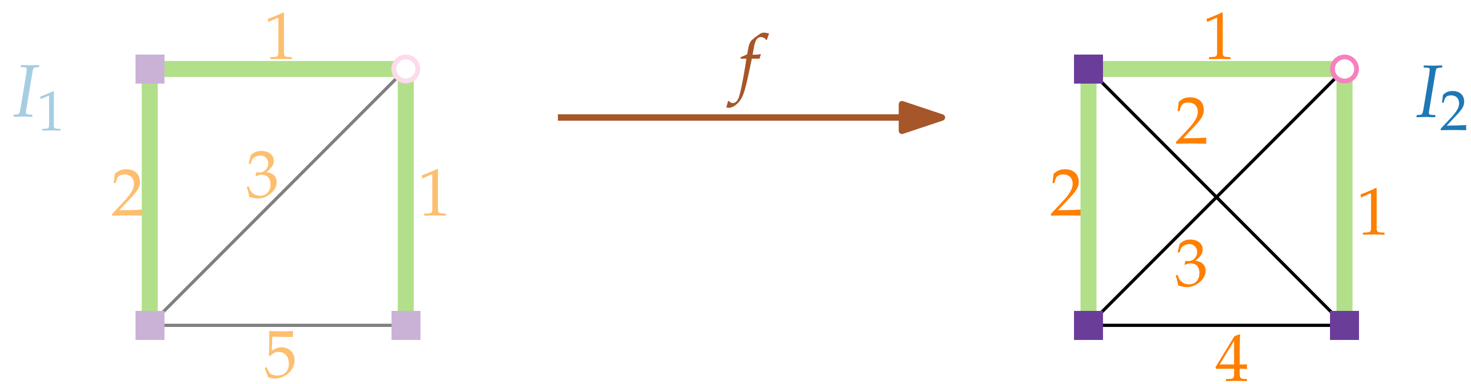
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (2) $\mathrm{OPT}(I_2) \leq \mathrm{OPT}(I_1)$

Let B^* be optimal Steiner tree for I_1

B^* is also a feasible solution for I_2, since $E_1 \subseteq E_2$ and the vertex sets V, T, S are the same

$\mathrm{OPT}(I_2)$



MetricSteinerTree

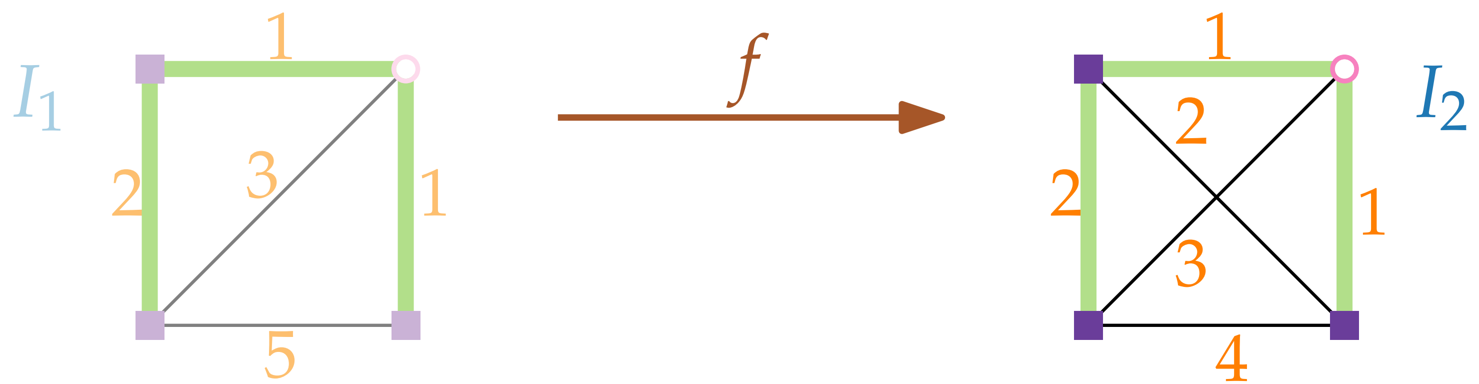
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (2) $\text{OPT}(I_2) \leq \text{OPT}(I_1)$

Let B^* be optimal Steiner tree for I_1

B^* is also a feasible solution for I_2, since $E_1 \subseteq E_2$ and the vertex sets V, T, S are the same

$\text{OPT}(I_2) \leq c_2(B^*)$



MetricSteinerTree

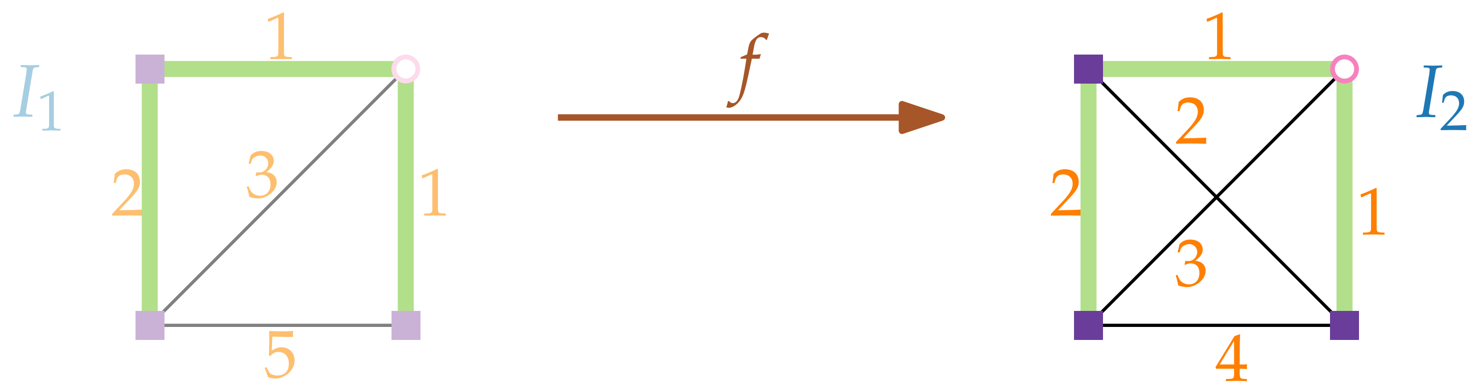
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (2) $\mathrm{OPT}(I_2) \leq \mathrm{OPT}(I_1)$

Let B^* be optimal Steiner tree for I_1

B^* is also a feasible solution for I_2, since $E_1 \subseteq E_2$ and the vertex sets V, T, S are the same

$\mathrm{OPT}(I_2) \leq c_2(B^*) \leq c_1(B^*)$



MetricSteinerTree

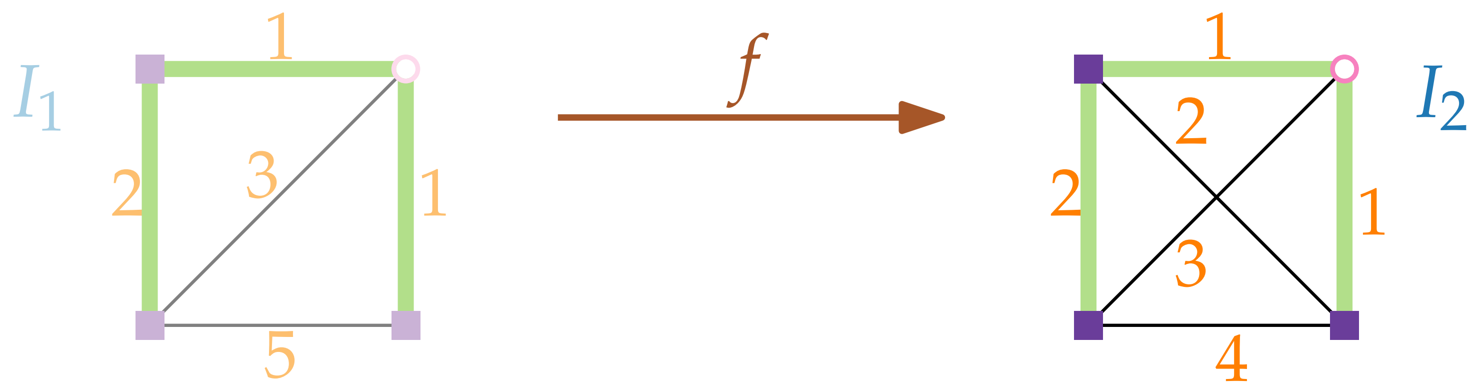
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (2) $\mathrm{OPT}(I_2) \leq \mathrm{OPT}(I_1)$

Let B^* be optimal Steiner tree for I_1

B^* is also a feasible solution for I_2, since $E_1 \subseteq E_2$ and the vertex sets V, T, S are the same

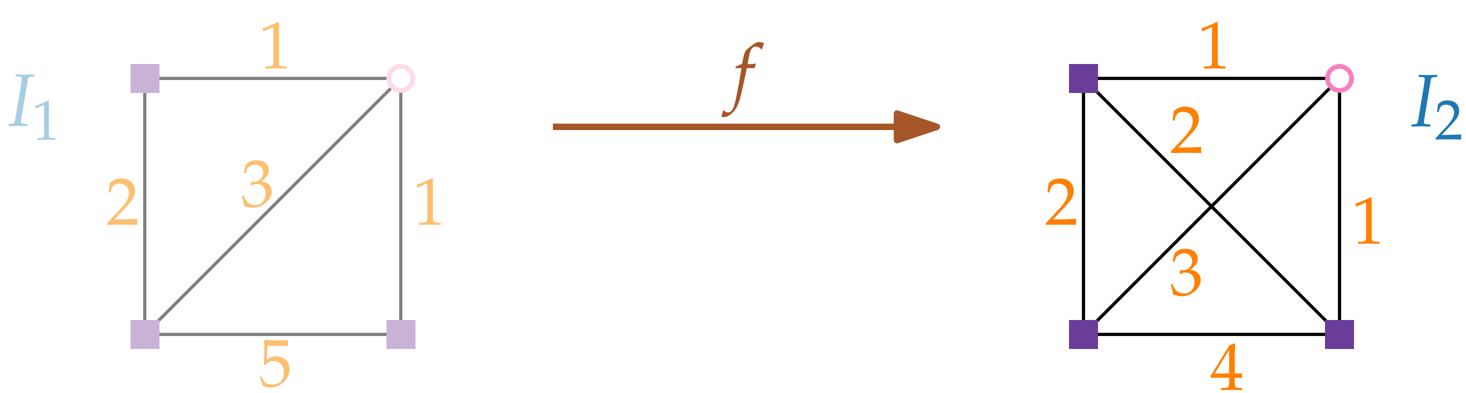
$\mathrm{OPT}(I_2) \leq c_2(B^*) \leq c_1(B^*) = \mathrm{OPT}(I_1)$



MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g

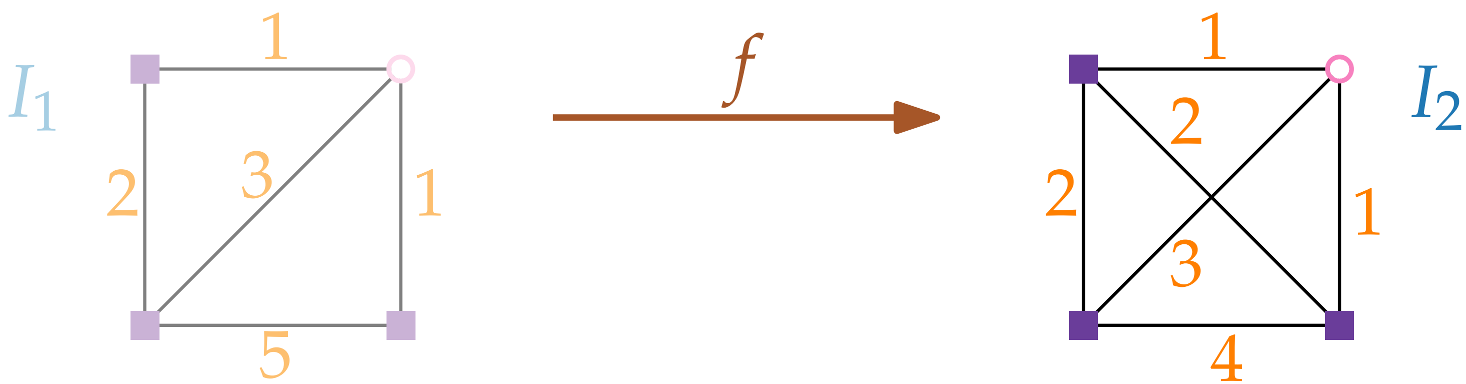


MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g

Let B_2 be Steiner tree of G_2

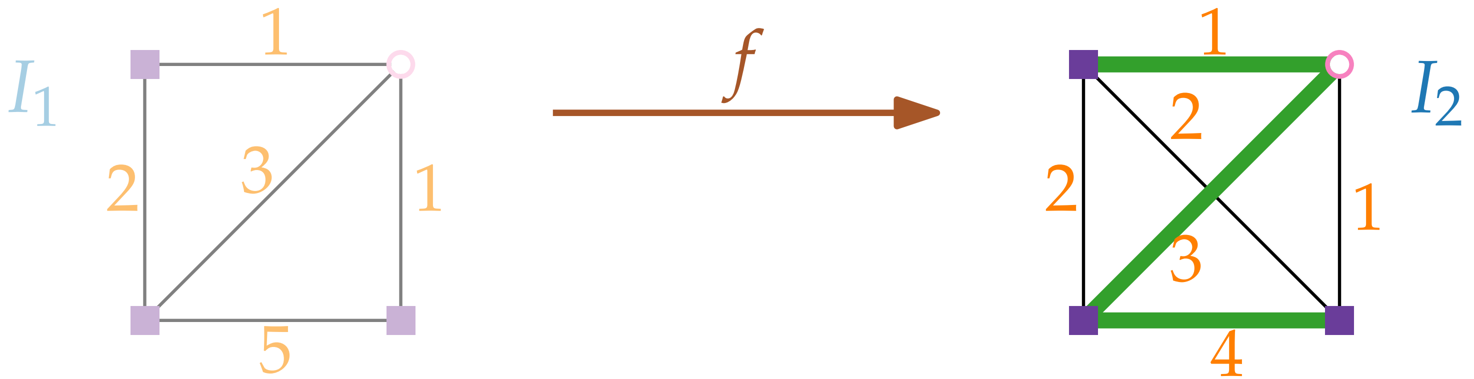


MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g

Let B_2 be Steiner tree of G_2



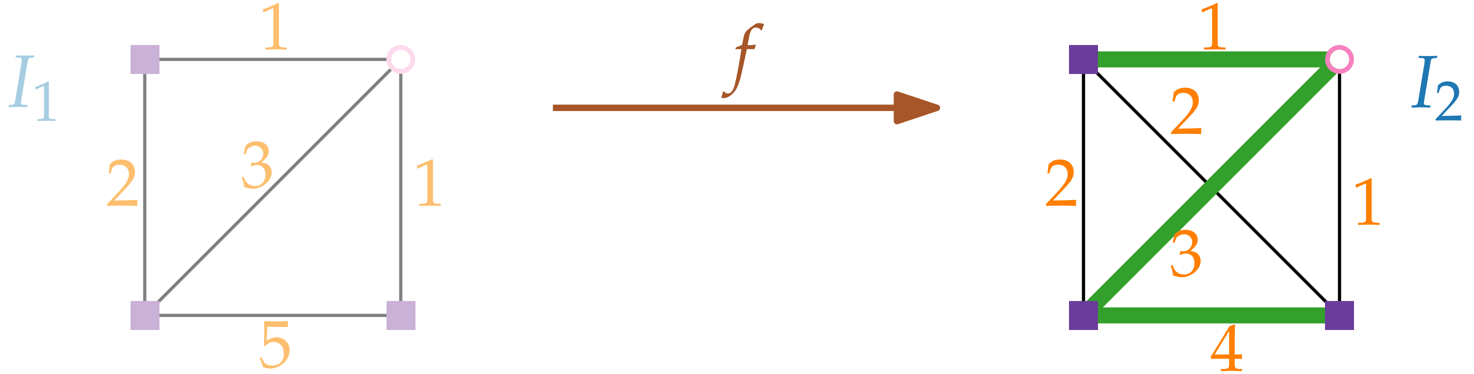
MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g

Let B_2 be Steiner tree of G_2

Construct $G_1' \subseteq G_1$ from B_2 by replacing each edge (u, v) of B_2 by a shortest u–v-path in G_1.



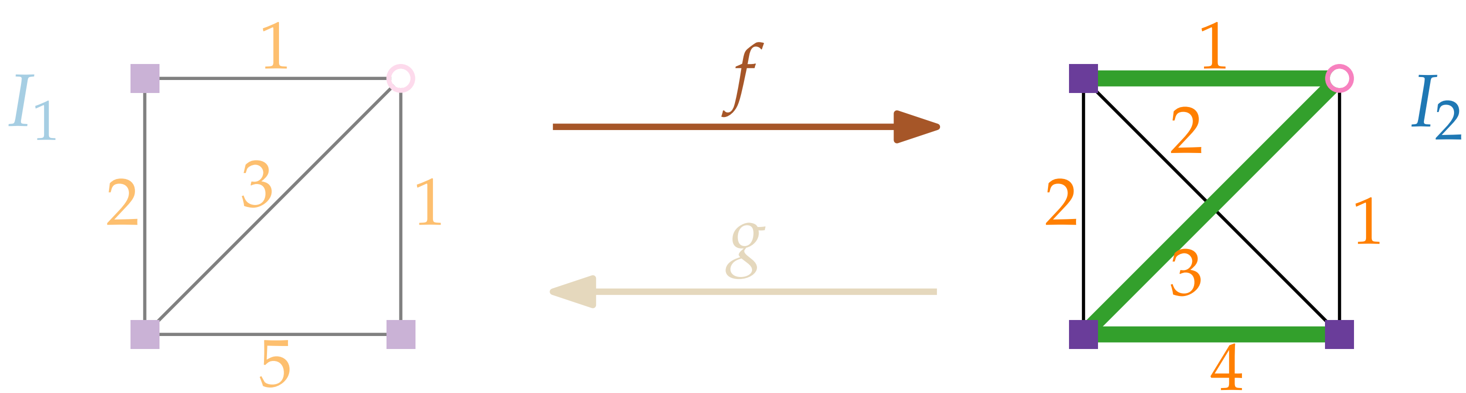
MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g

Let B_2 be Steiner tree of G_2

Construct $G_1' \subseteq G_1$ from B_2 by replacing each edge (u,v) of B_2 by a shortest u–v-path in G_1.



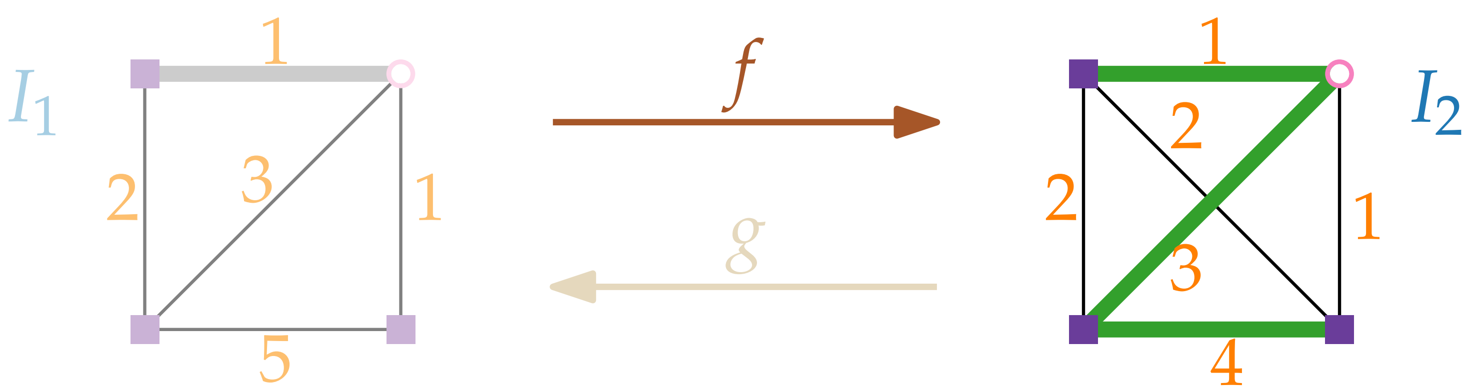
MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g

Let B_2 be Steiner tree of G_2

Construct $G'_1 \subseteq G_1$ from B_2 by replacing each edge (u, v) of B_2 by a shortest u–v-path in G_1.



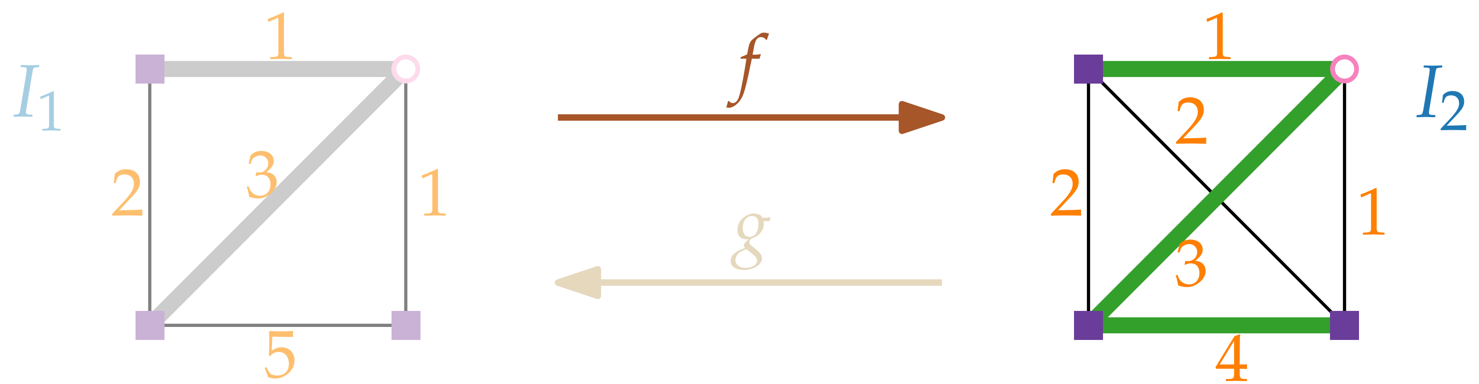
MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g

Let B_2 be Steiner tree of G_2

Construct $G_1' \subseteq G_1$ from B_2 by replacing each edge (u,v) of B_2 by a shortest u–v-path in G_1.



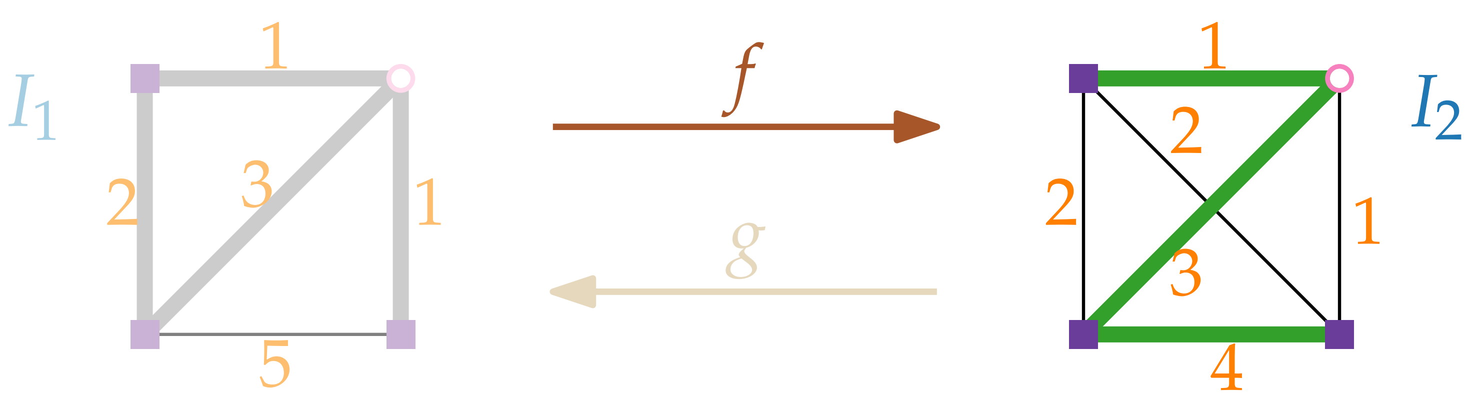
MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g

Let B_2 be Steiner tree of G_2

Construct $G_1' \subseteq G_1$ from B_2 by replacing each edge (u, v) of B_2 by a shortest u–v-path in G_1.



MetricSteinerTree

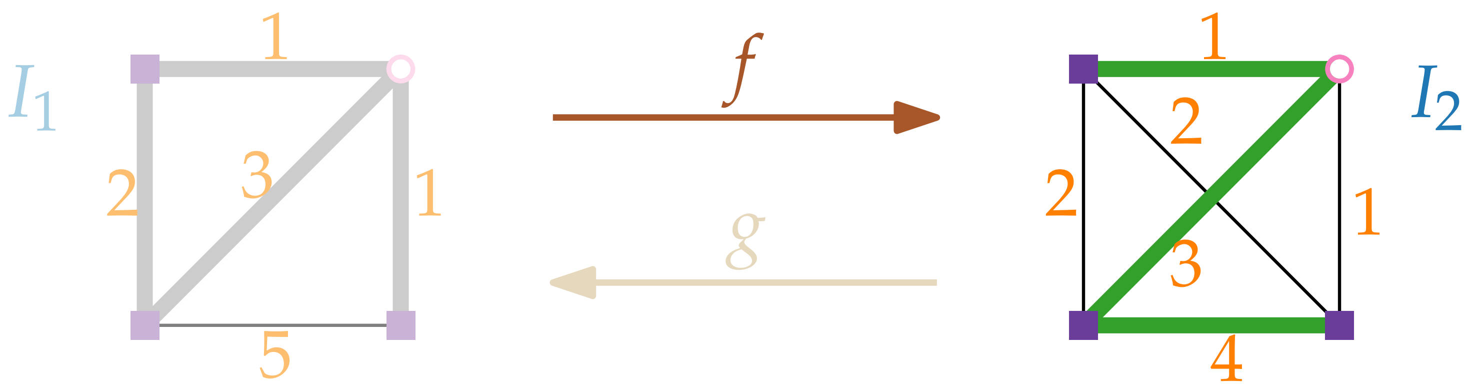
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g

Let B_2 be Steiner tree of G_2

Construct $G_1' \subseteq G_1$ from B_2 by replacing each edge (u, v) of B_2 by a shortest u–v-path in G_1.

$c_1(G_1') \leq c_2(B_2)$



MetricSteinerTree

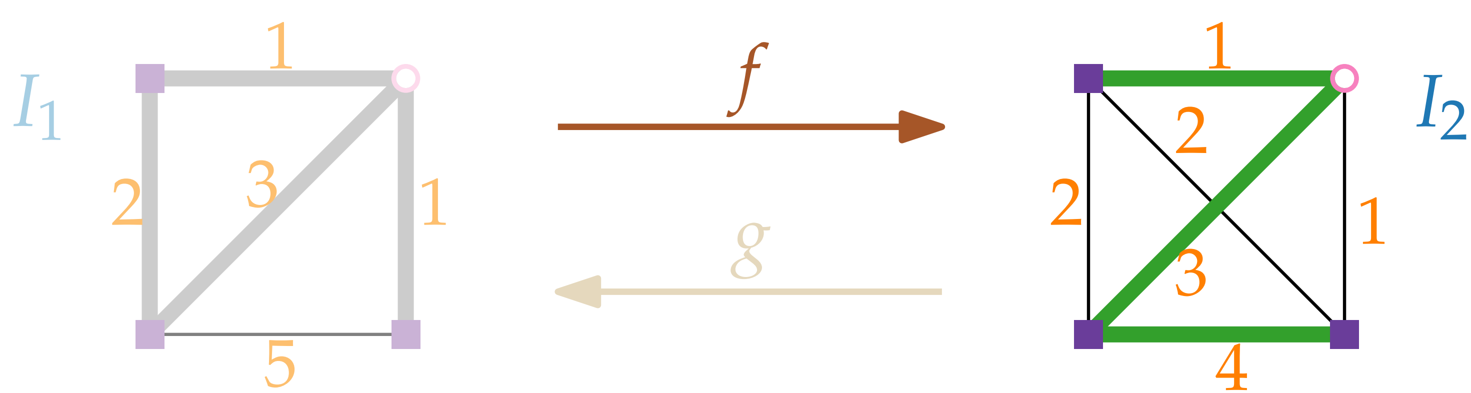
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g $s \xleftarrow{g} t$

Let B_2 be Steiner tree of G_2

Construct $G_1' \subseteq G_1$ from B_2 by replacing each edge (u, v) of B_2 by a shortest u–v-path in G_1.

$c_1(G_1') \leq c_2(B_2)$; G_1' connects all terminals



MetricSteinerTree

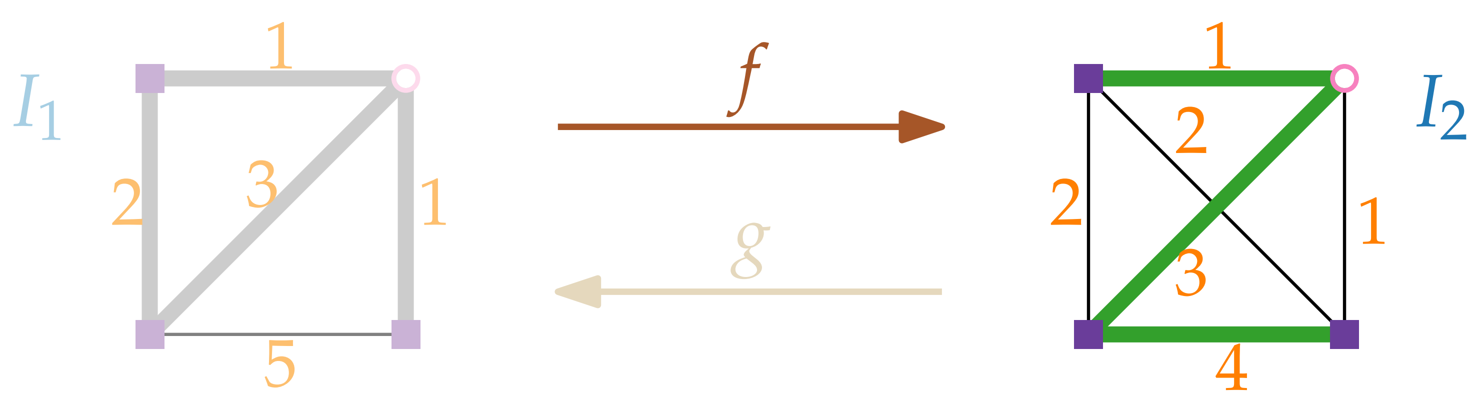
Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g $s \xleftarrow{g} t$

Let B_2 be Steiner tree of G_2

Construct $G_1' \subseteq G_1$ from B_2 by replacing each edge (u, v) of B_2 by a shortest u–v-path in G_1.

$c_1(G_1') \leq c_2(B_2)$; G_1' connects all terminals; not nec. a tree



MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

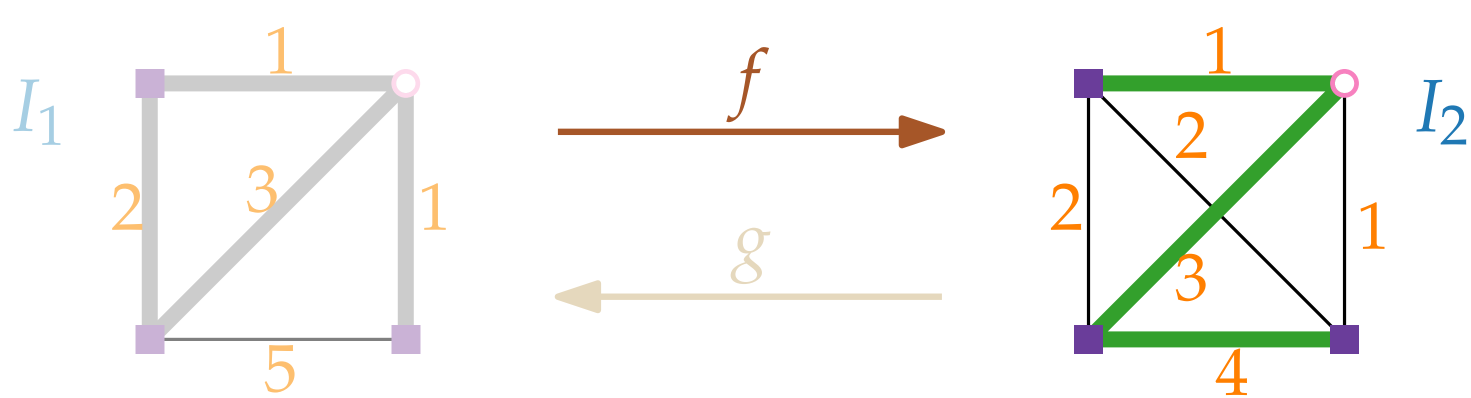
Proof. (3) Mapping g $s \xleftarrow{g} t$

Let B_2 be Steiner tree of G_2

Construct $G_1' \subseteq G_1$ from B_2 by replacing each edge (u, v) of B_2 by a shortest u–v-path in G_1.

$c_1(G_1') \leq c_2(B_2)$; G_1' connects all terminals; not nec. a tree

Consider spanning tree B_1 of G_1'



MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

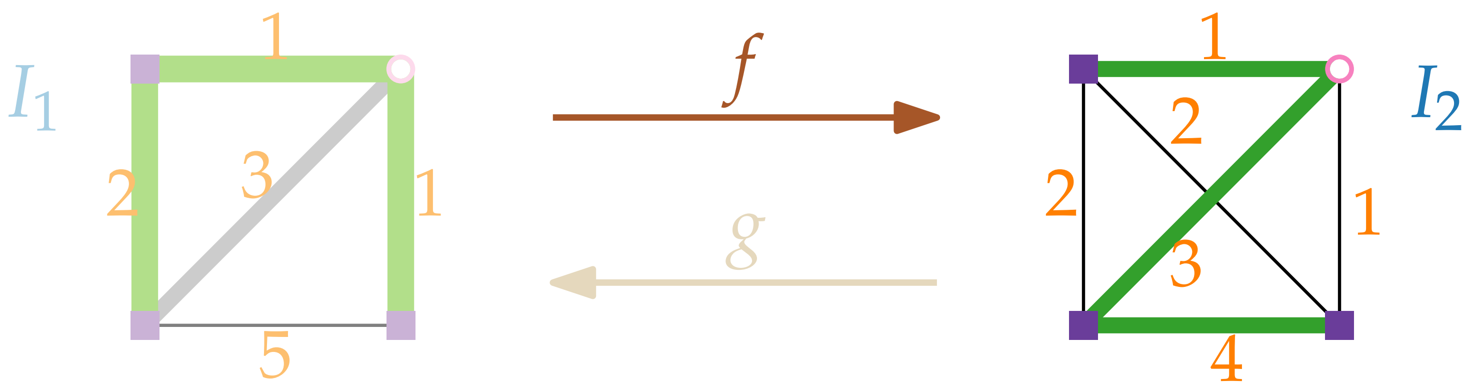
Proof. (3) Mapping g

Let B_2 be Steiner tree of G_2

Construct $G_1' \subseteq G_1$ from B_2 by replacing each edge (u, v) of B_2 by a shortest u–v-path in G_1.

$c_1(G_1') \leq c_2(B_2)$; G_1' connects all terminals; not nec. a tree

Consider spanning tree B_1 of G_1'



MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

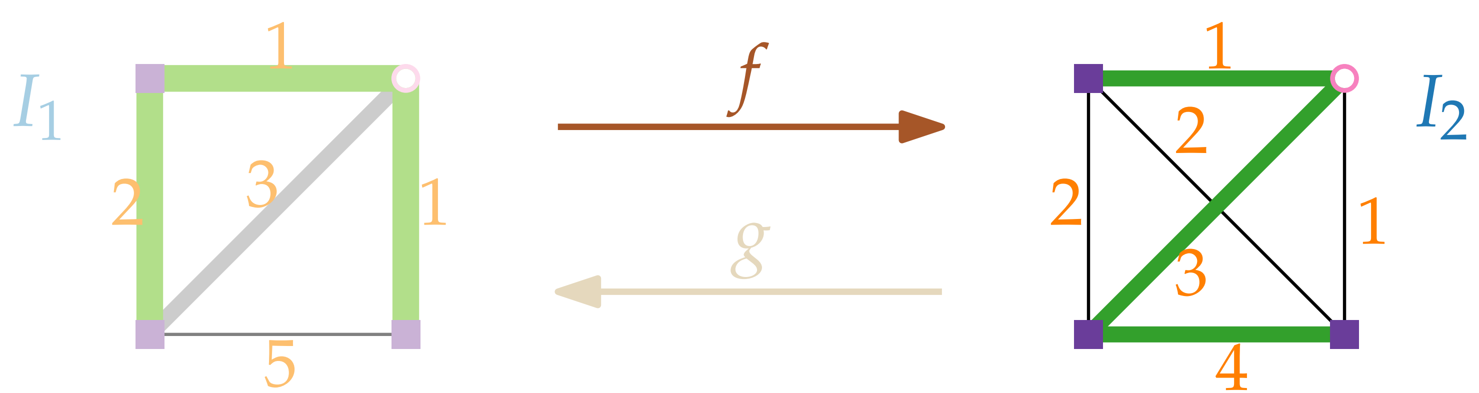
Proof. (3) Mapping g $s \xleftarrow{g} t$

Let B_2 be Steiner tree of G_2

Construct $G_1' \subseteq G_1$ from B_2 by replacing each edge (u, v) of B_2 by a shortest u–v-path in G_1.

$c_1(G_1') \leq c_2(B_2)$; G_1' connects all terminals; not nec. a tree

Consider spanning tree B_1 of G_1' $\rightsquigarrow$ Steiner tree B_1 of G_1



MetricSteinerTree

Theorem. There is an approximation preserving reduction from SteinerTree to MetricSteinerTree.

Proof. (3) Mapping g

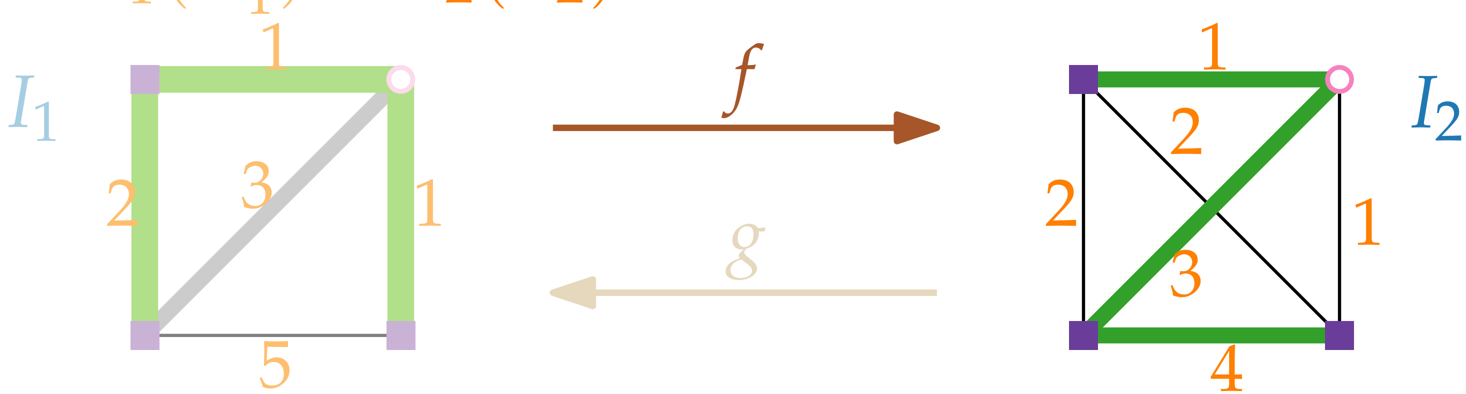
Let B_2 be Steiner tree of G_2

Construct $G_1' \subseteq G_1$ from B_2 by replacing each edge (u, v) of B_2 by a shortest u–v-path in G_1.

$c_1(G_1') \leq c_2(B_2)$; G_1' connects all terminals; not nec. a tree

Consider spanning tree B_1 of G_1' $\rightsquigarrow$ Steiner tree B_1 of G_1

$c_1(B_1) \leq c_1(G_1') \leq c_2(B_2)$



Approximation Algorithms

Lecture 3:
SteinerTree and MultiwayCut

Part IV:
2-Approximation for SteinerTree

Joachim Spoerhase

Winter 2021/22

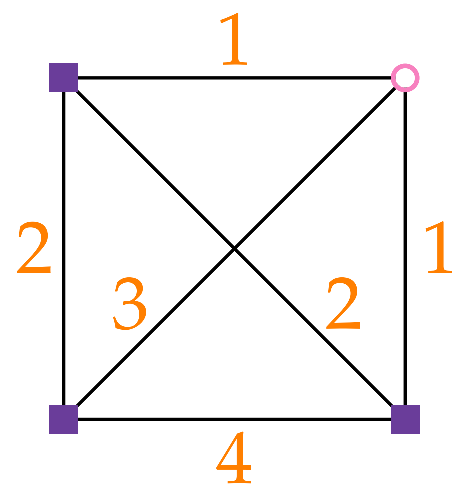
2-Approximation for SteinerTree

2-Approximation for SteinerTree

Theorem. For an instance of MetricSteinerTree, let B be a minimum spanning tree (MST) of the subgraph $G[T]$ induced by the terminal set T. Then $c(B) \leq 2 \cdot \mathrm{OPT}$.

2-Approximation for SteinerTree

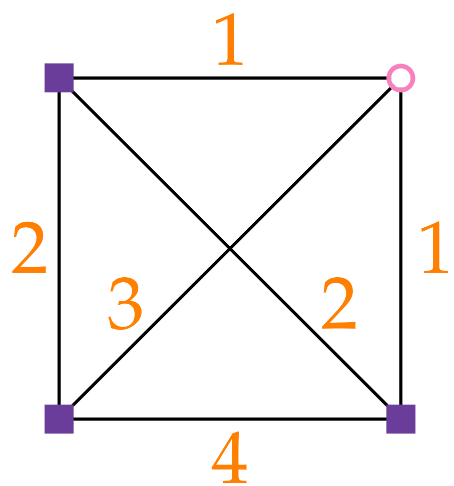
Theorem. For an instance of MetricSteinerTree, let B be a minimum spanning tree (MST) of the subgraph $G[T]$ induced by the terminal set T. Then $c(B) \leq 2 \cdot \mathrm{OPT}$.



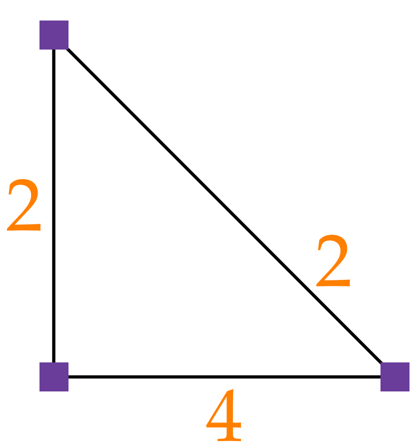
G

2-Approximation for SteinerTree

Theorem. For an instance of MetricSteinerTree, let B be a minimum spanning tree (MST) of the subgraph $G[T]$ induced by the terminal set T. Then $c(B) \leq 2 \cdot \mathrm{OPT}$.



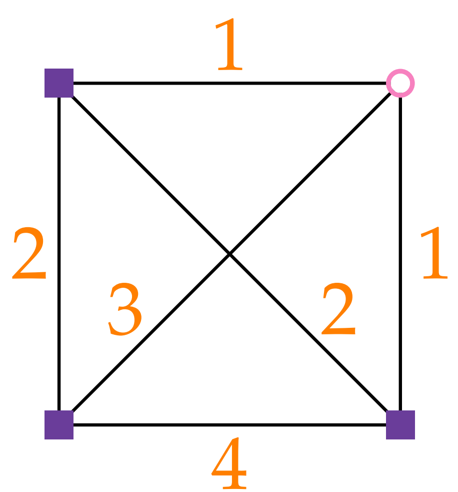
G



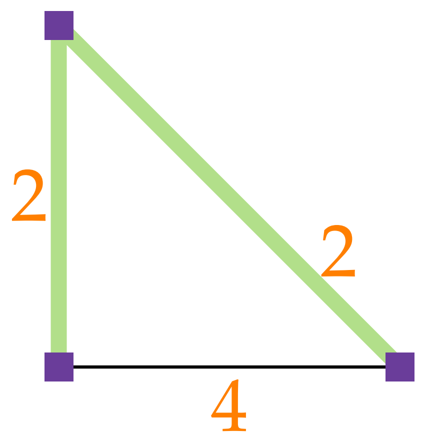
$G[T]$

2-Approximation for SteinerTree

Theorem. For an instance of MetricSteinerTree, let B be a minimum spanning tree (MST) of the subgraph $G[T]$ induced by the terminal set T. Then $c(B) \leq 2 \cdot \mathrm{OPT}$.



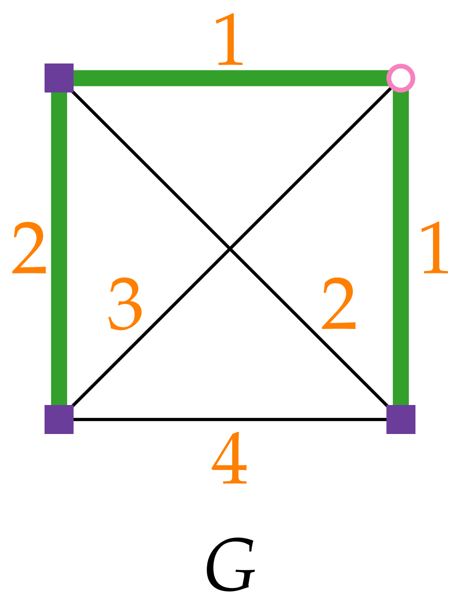
G

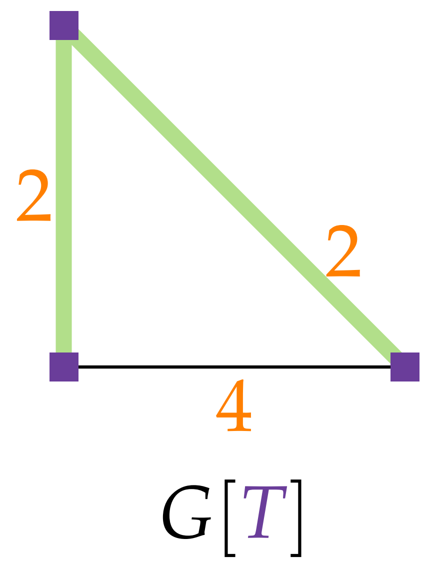


$G[T]$

2-Approximation for SteinerTree

Theorem. For an instance of MetricSteinerTree, let B be a minimum spanning tree (MST) of the subgraph $G[T]$ induced by the terminal set T. Then $c(B) \leq 2 \cdot \mathrm{OPT}$.



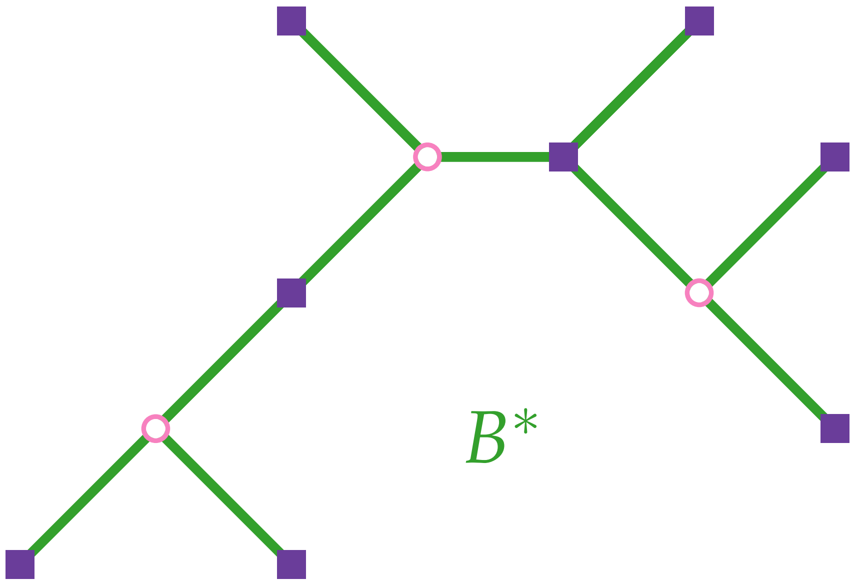


Proof of Approximation Factor

Consider optimal Steiner tree B^*

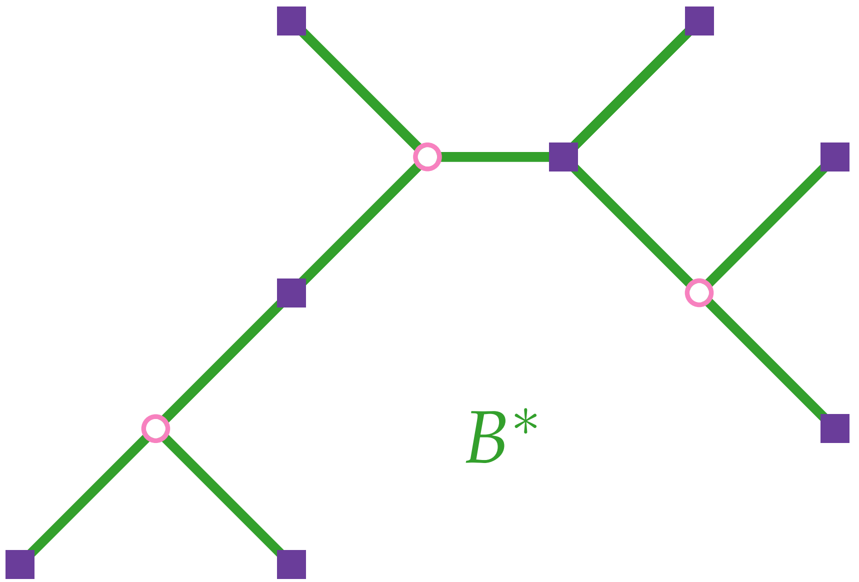
Proof of Approximation Factor

Consider optimal Steiner tree B^*



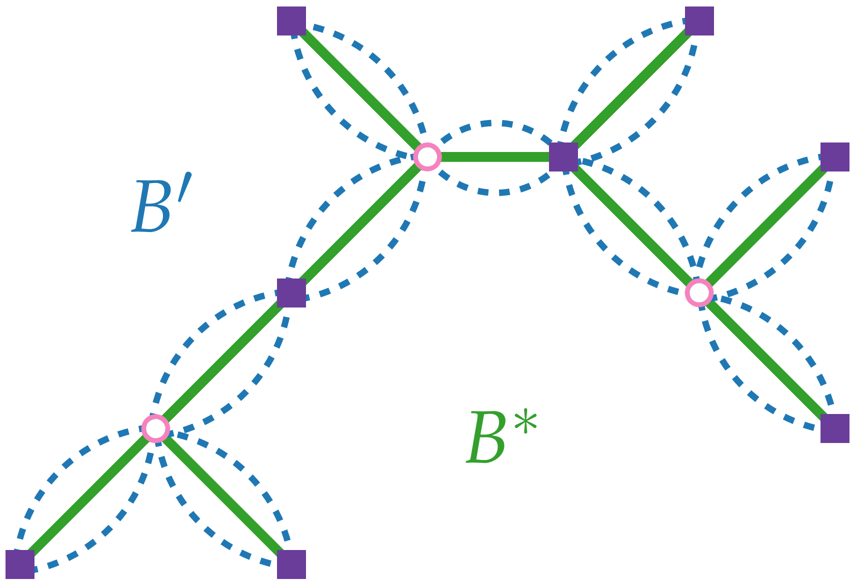
Proof of Approximation Factor

Consider optimal Steiner tree B^*
Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$



Proof of Approximation Factor

Consider optimal Steiner tree B^*
Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$

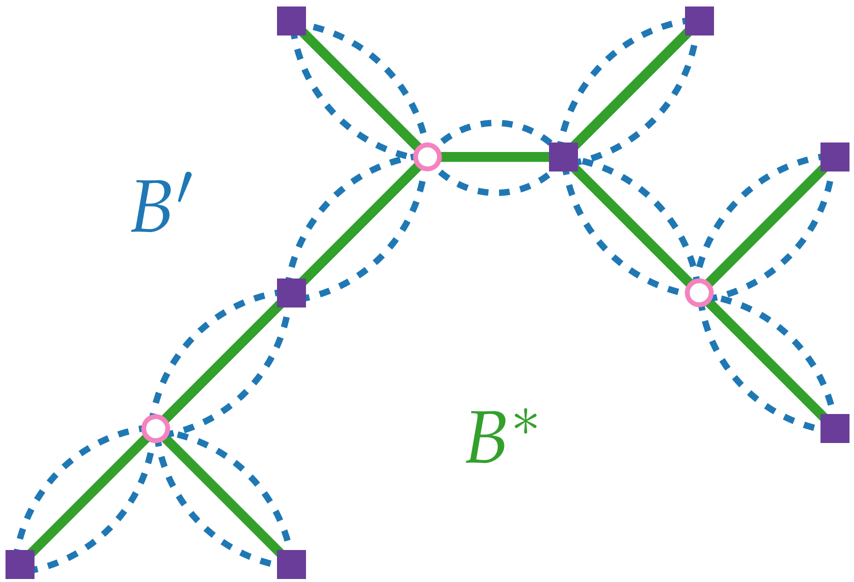


Proof of Approximation Factor

Consider optimal Steiner tree B^*

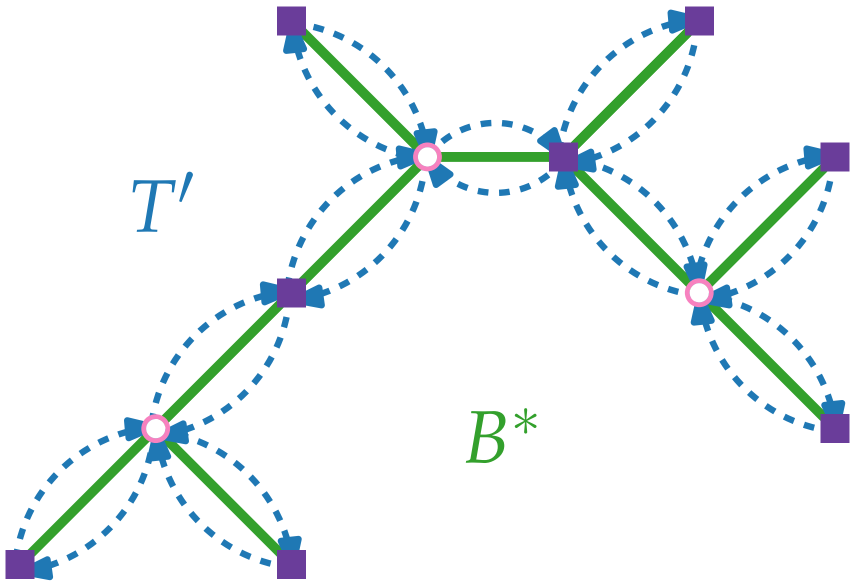
Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$

Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$



Proof of Approximation Factor

Consider optimal Steiner tree B^*
Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$
Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$



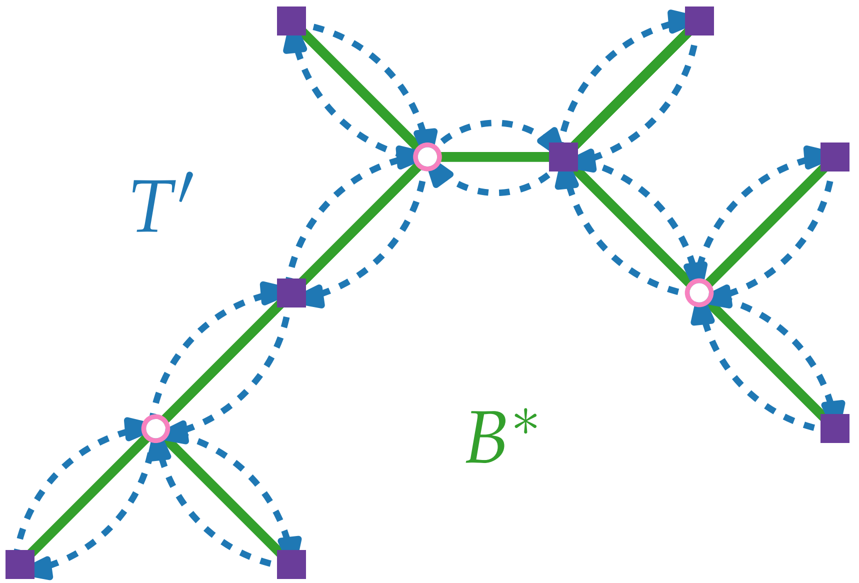
Proof of Approximation Factor

Consider optimal Steiner tree B^*

Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$

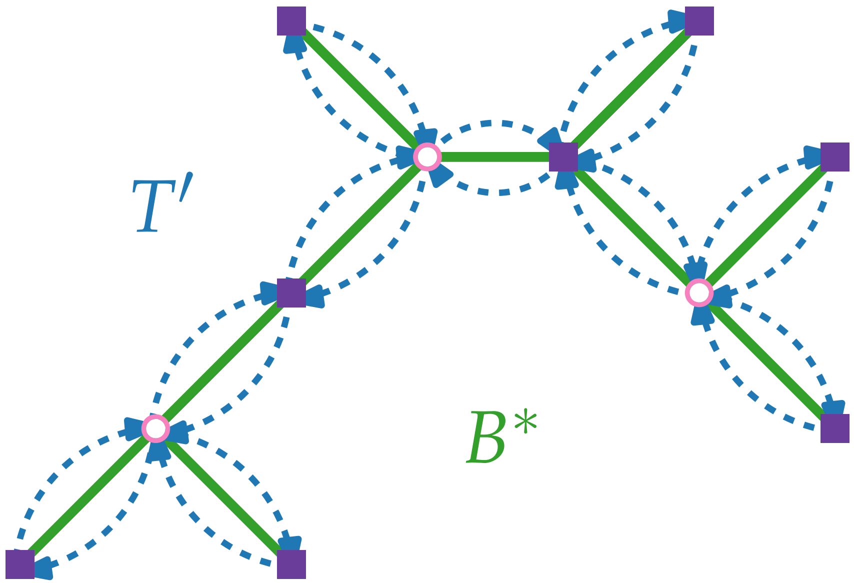
Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$

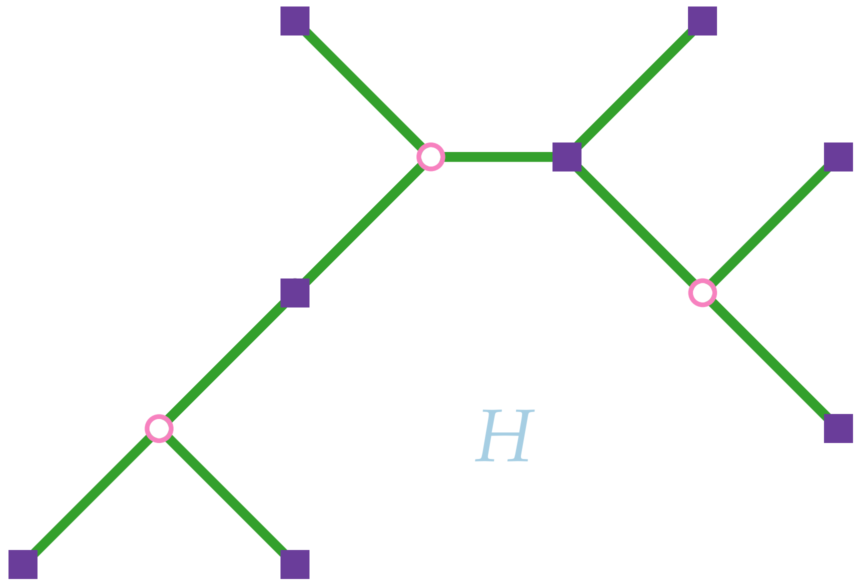
Find a Hamiltonian path H in $G[T]$ by "short-cutting" Steiner vertices and previously visited terminals



Proof of Approximation Factor

Consider optimal Steiner tree B^*
Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$
Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$
Find a Hamiltonian path H in $G[T]$ by "short-cutting" Steiner vertices and previously visited terminals





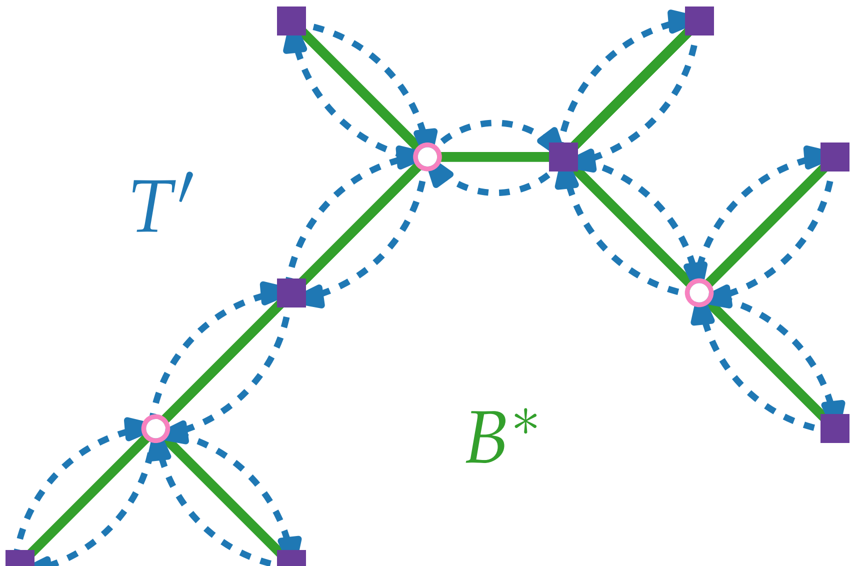
Proof of Approximation Factor

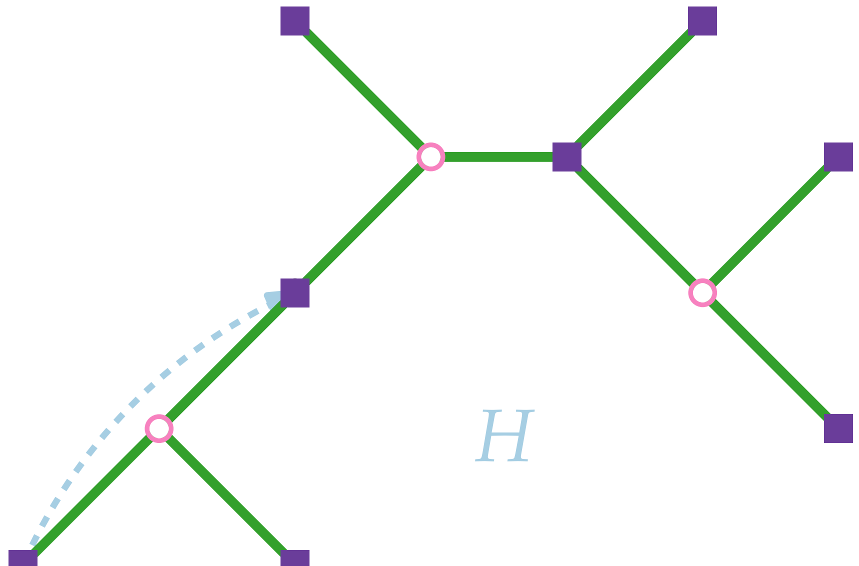
Consider optimal Steiner tree B^*

Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$

Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$

Find a Hamiltonian path H in $G[T]$ by "short-cutting" Steiner vertices and previously visited terminals





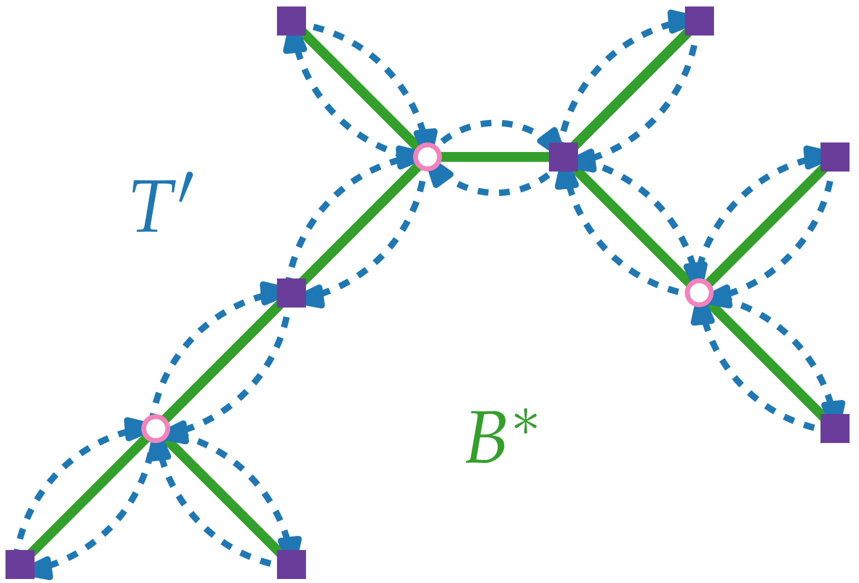
Proof of Approximation Factor

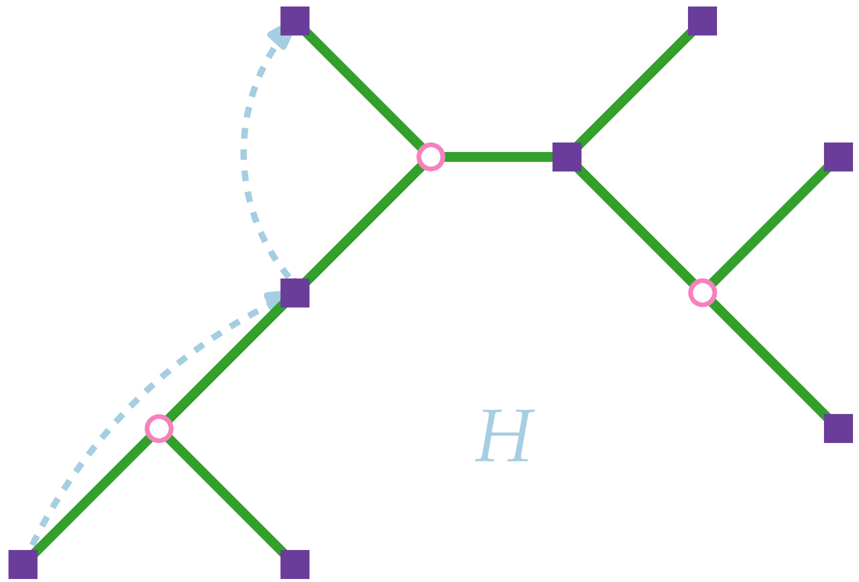
Consider optimal Steiner tree B^*

Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$

Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$

Find a Hamiltonian path H in $G[T]$ by "short-cutting" Steiner vertices and previously visited terminals





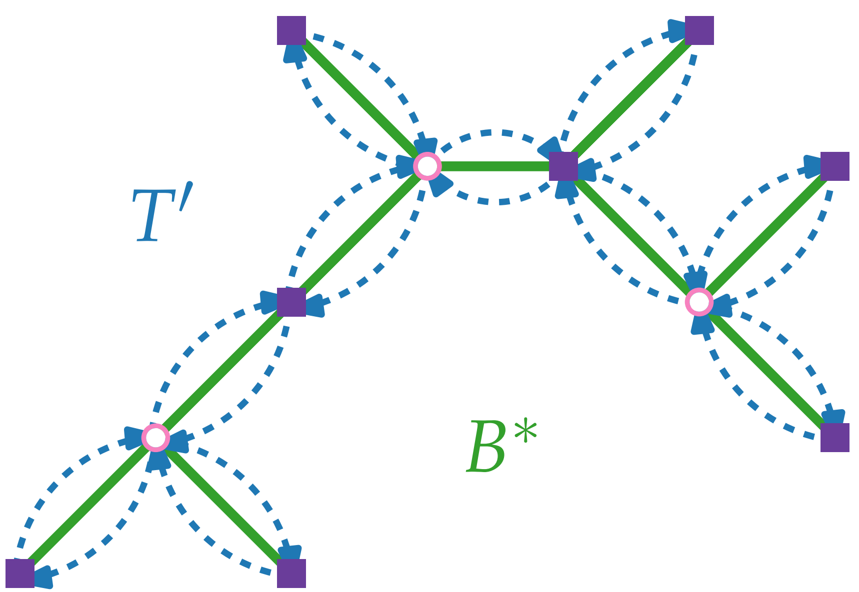
Proof of Approximation Factor

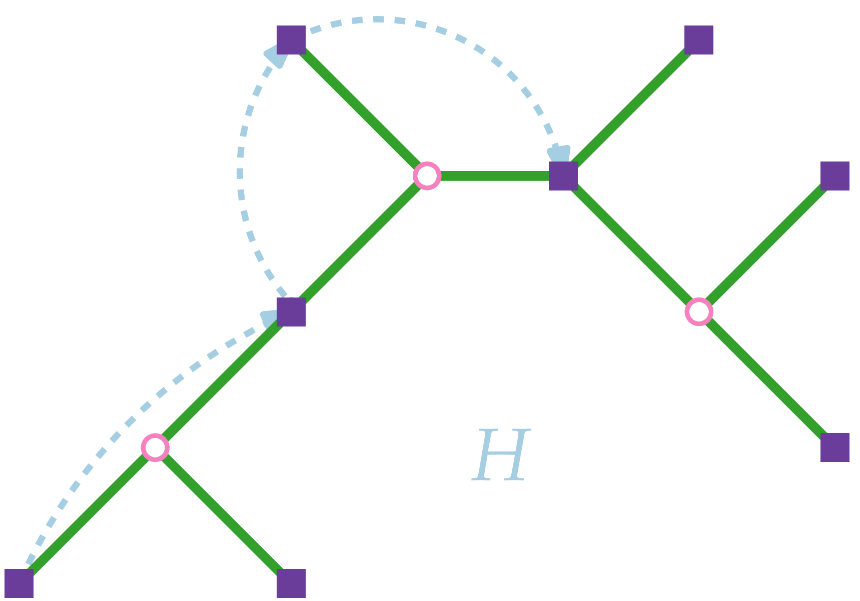
Consider optimal Steiner tree B^*

Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$

Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$

Find a Hamiltonian path H in $G[T]$ by "short-cutting" Steiner vertices and previously visited terminals





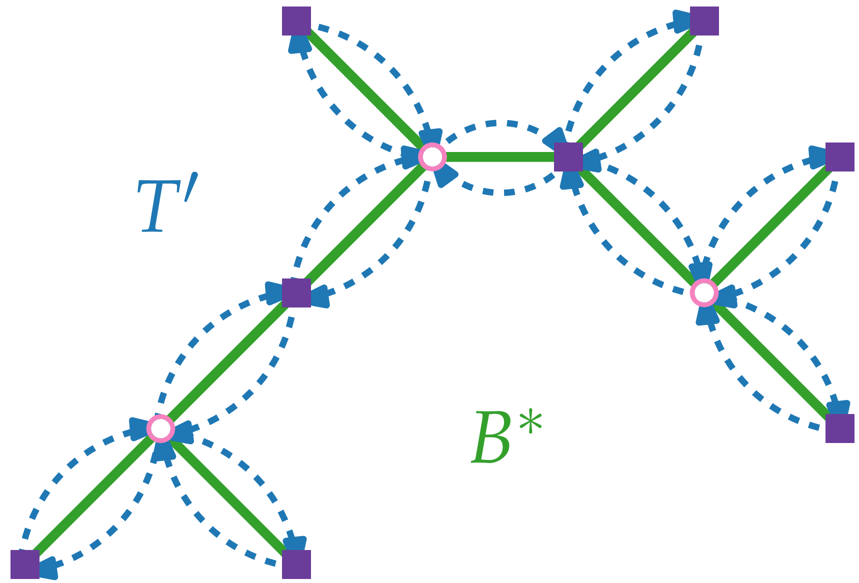
Proof of Approximation Factor

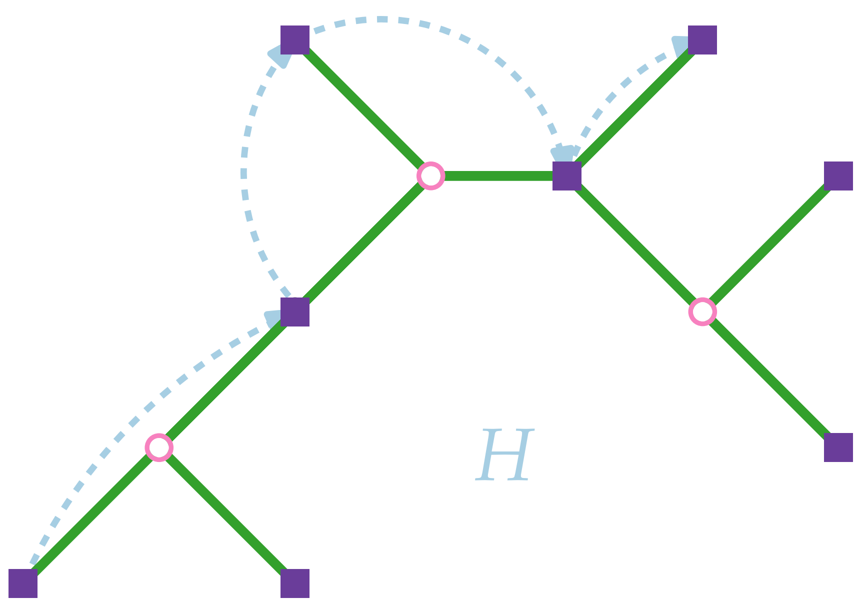
Consider optimal Steiner tree B^*

Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$

Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$

Find a Hamiltonian path H in $G[T]$ by "short-cutting" Steiner vertices and previously visited terminals





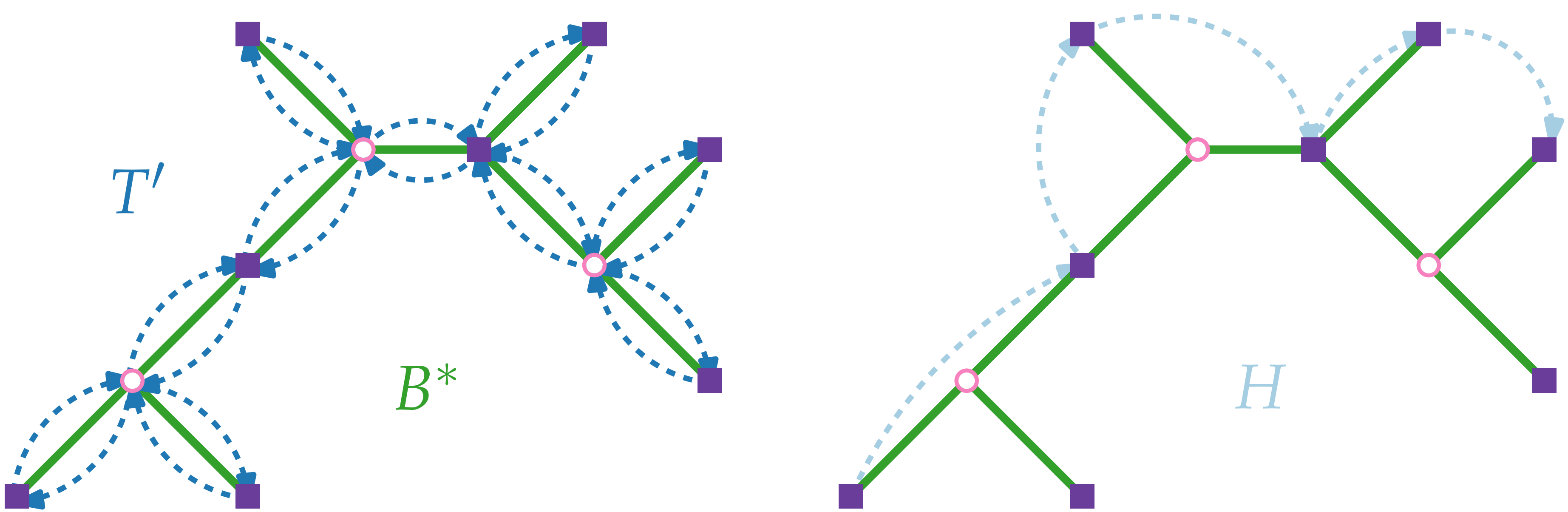
Proof of Approximation Factor

Consider optimal Steiner tree B^*

Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$

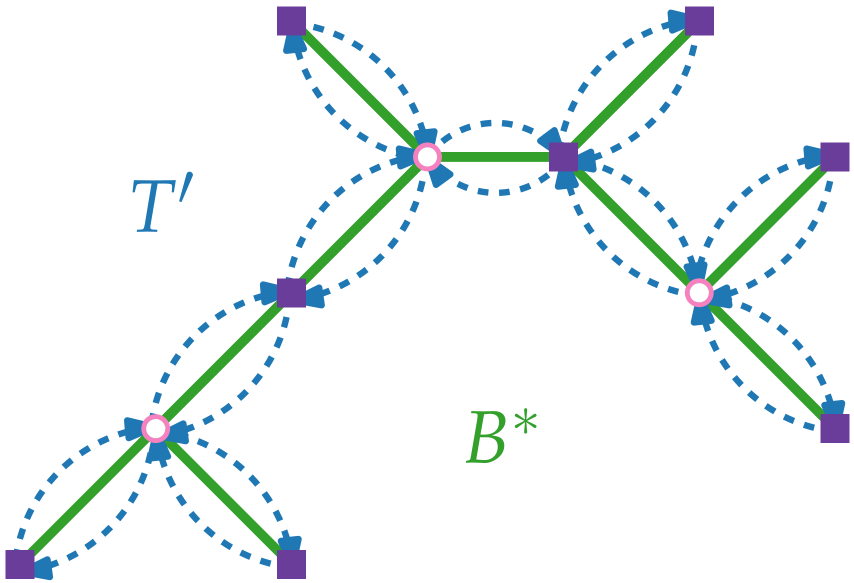
Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$

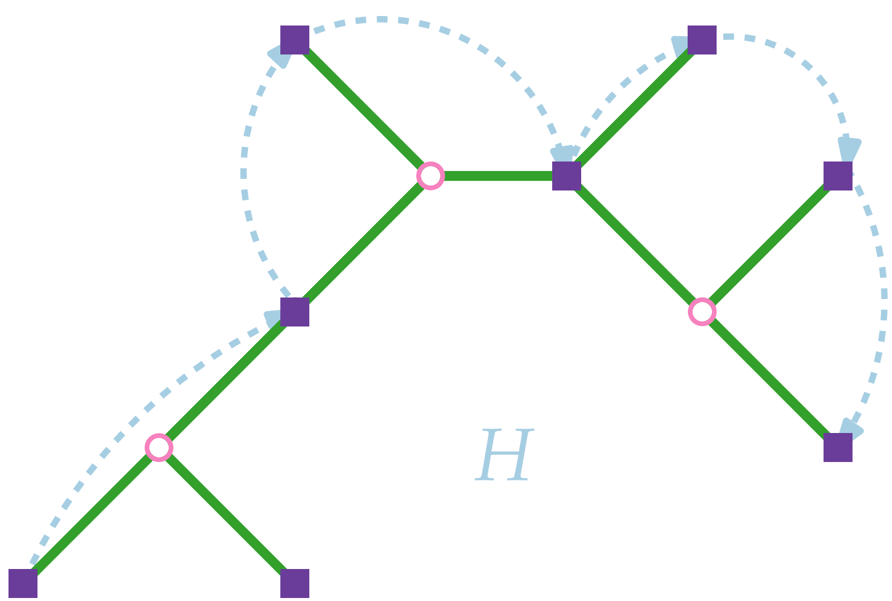
Find a Hamiltonian path H in $G[T]$ by "short-cutting" Steiner vertices and previously visited terminals



Proof of Approximation Factor

Consider optimal Steiner tree B^*
Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$
Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$
Find a Hamiltonian path H in $G[T]$ by "short-cutting" Steiner vertices and previously visited terminals





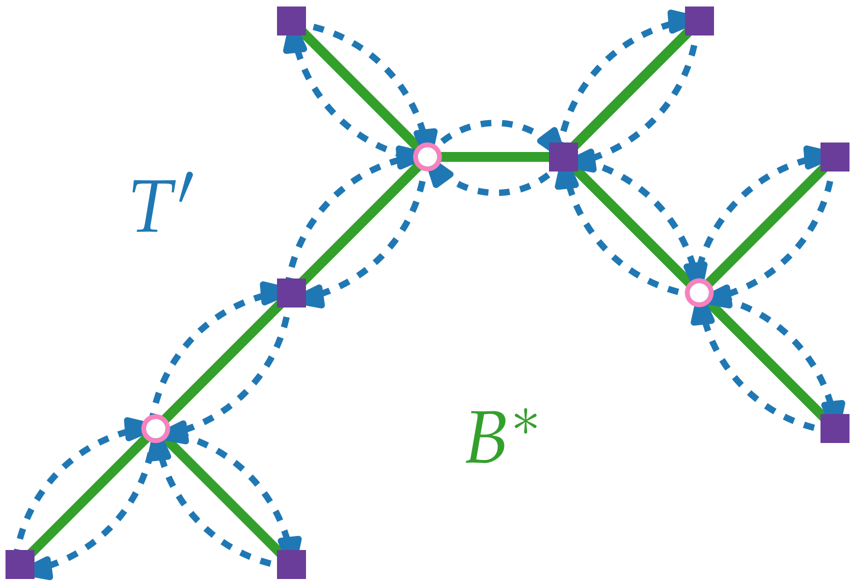
Proof of Approximation Factor

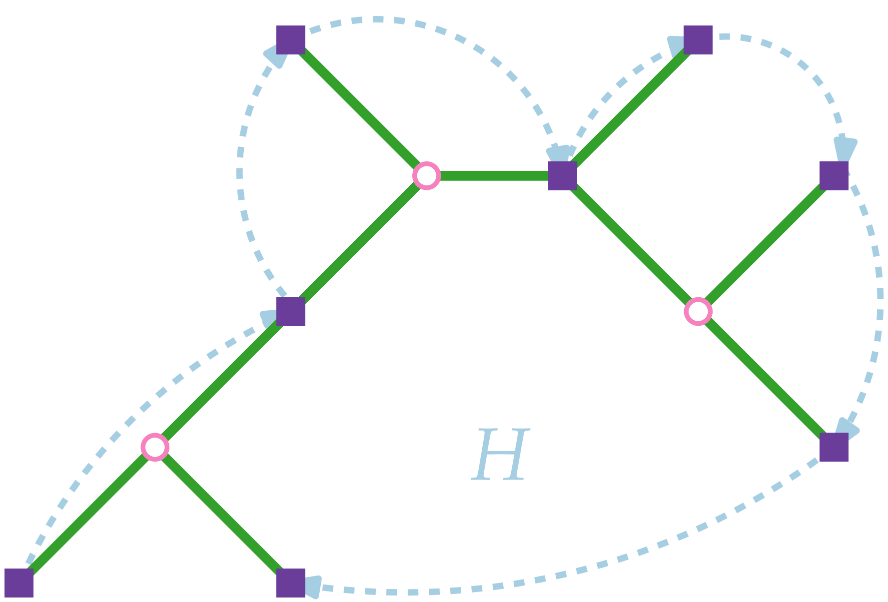
Consider optimal Steiner tree B^*

Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$

Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$

Find a Hamiltonian path H in $G[T]$ by "short-cutting" Steiner vertices and previously visited terminals





Proof of Approximation Factor

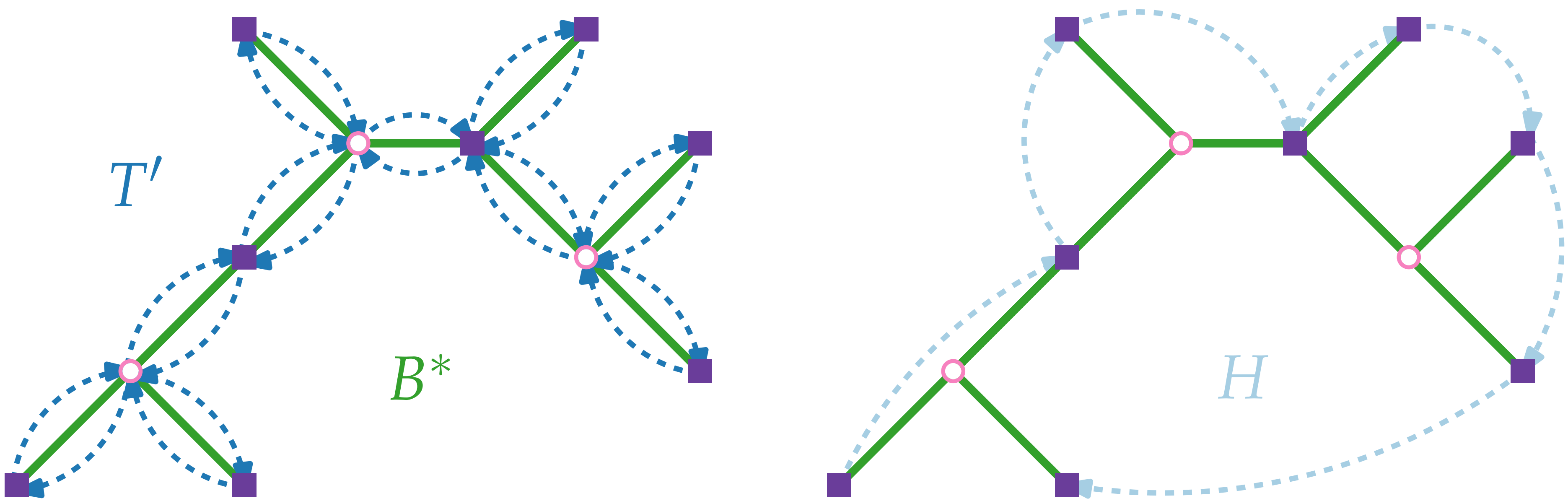
Consider optimal Steiner tree B^*

Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$

Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$

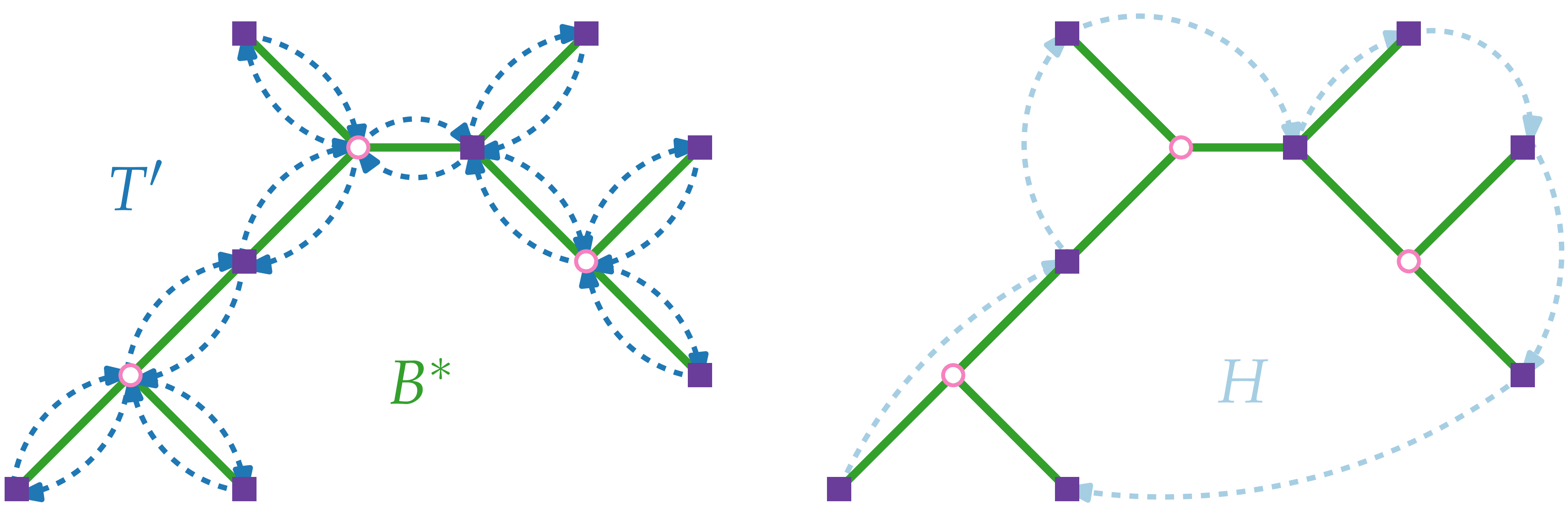
Find a Hamiltonian path H in $G[T]$ by “short-cutting” Steiner vertices and previously visited terminals

$\rightsquigarrow$ $c(H) \leq c(T') = 2 \cdot \mathrm{OPT}$, since G is metric



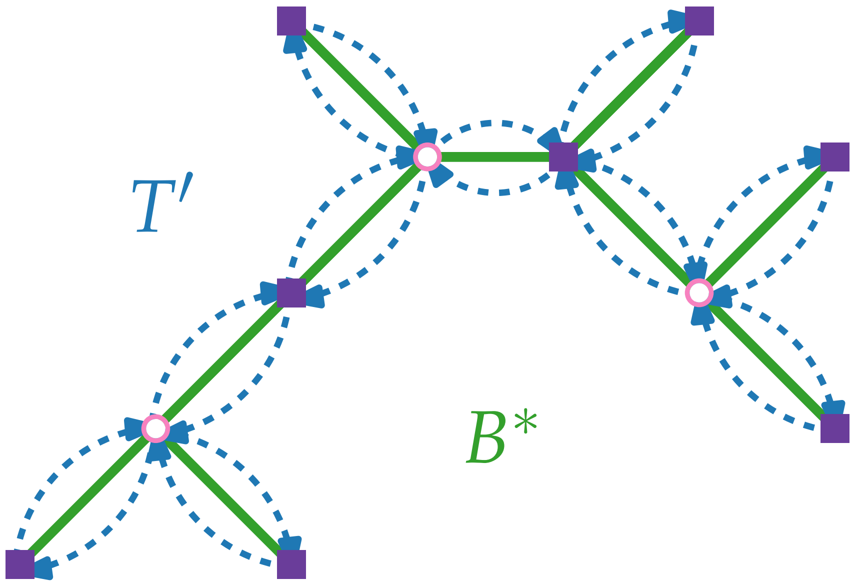
Proof of Approximation Factor

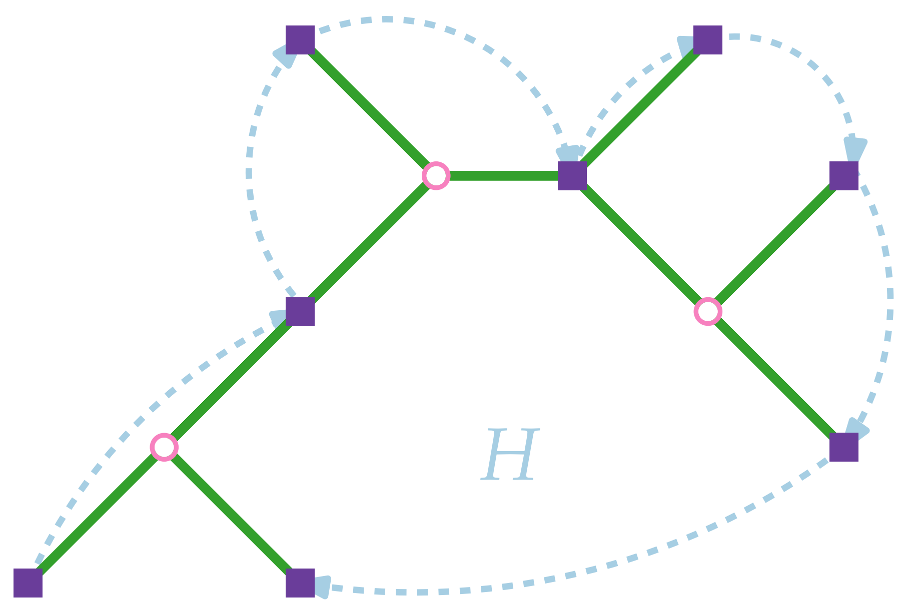
Consider optimal Steiner tree B^*
Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$
Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$
Find a Hamiltonian path H in $G[T]$ by "short-cutting" Steiner vertices and previously visited terminals
$\rightsquigarrow$ $c(H) \leq c(T') = 2 \cdot \mathrm{OPT}$, since G is metric
MST B of $G[T]$ has $c(B) \leq c(H) \leq 2 \cdot \mathrm{OPT}$,



Proof of Approximation Factor

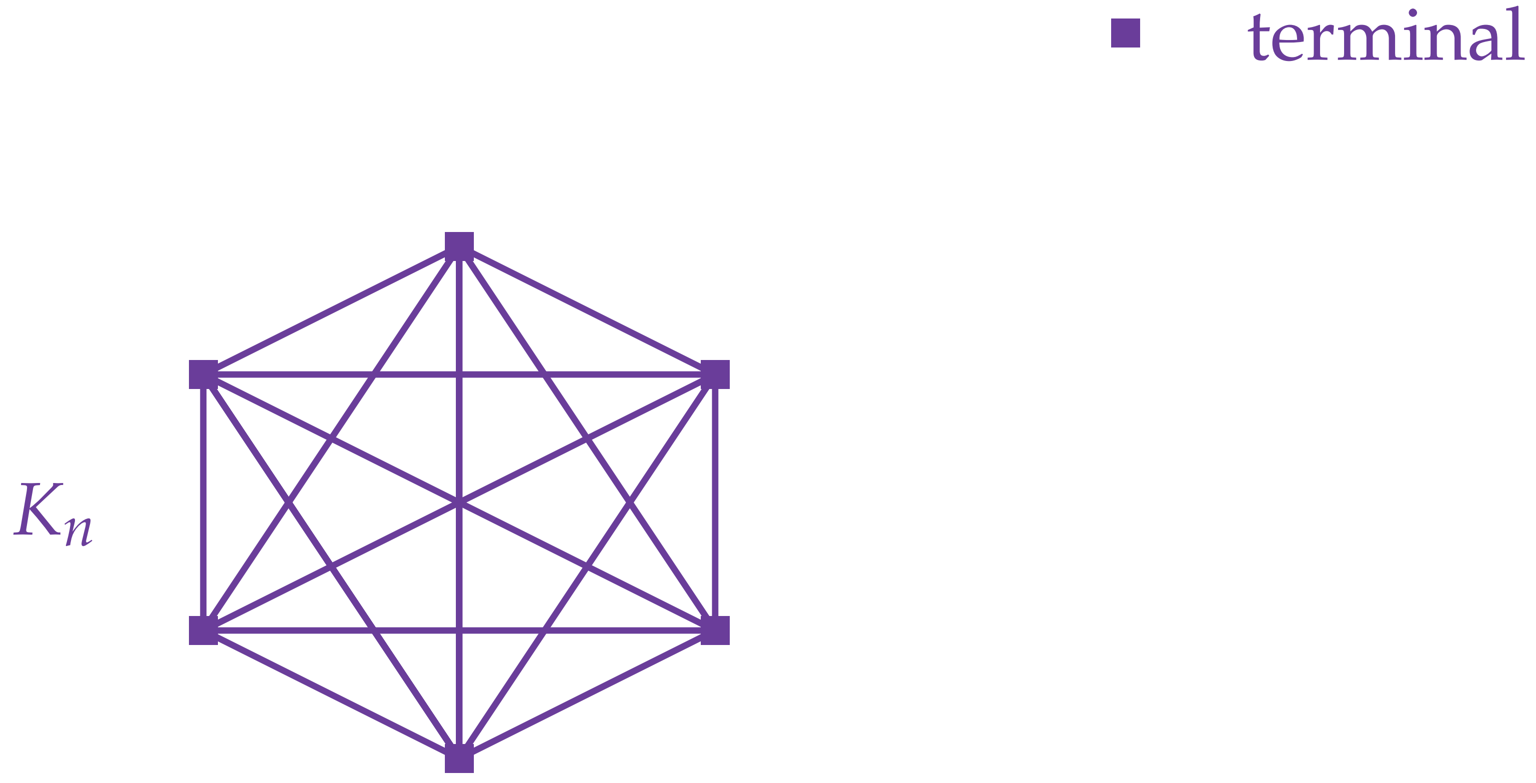
Consider optimal Steiner tree B^*
Duplicate all edges in B^* $\rightsquigarrow$ Eulerian (multi-)graph B' with cost $c(B') = 2 \cdot \mathrm{OPT}$
Find an Eulerian tour T' in B' $\rightsquigarrow$ $c(T') = c(B') = 2 \cdot \mathrm{OPT}$
Find a Hamiltonian path H in $G[T]$ by "short-cutting" Steiner vertices and previously visited terminals
$\rightsquigarrow$ $c(H) \leq c(T') = 2 \cdot \mathrm{OPT}$, since G is metric
MST B of $G[T]$ has $c(B) \leq c(H) \leq 2 \cdot \mathrm{OPT}$,
since H is a spanning tree of $G[T]$



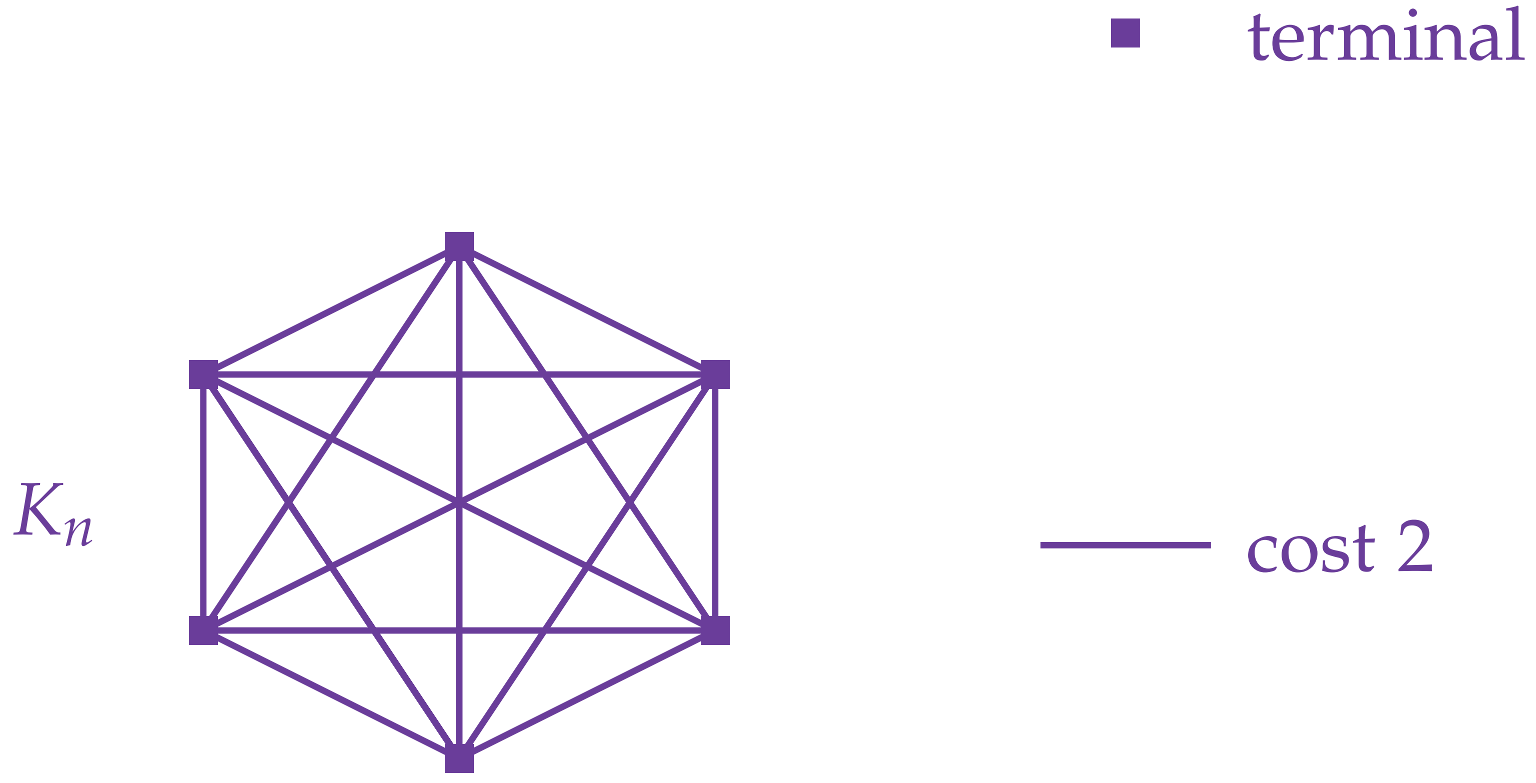


Analysis Sharp?

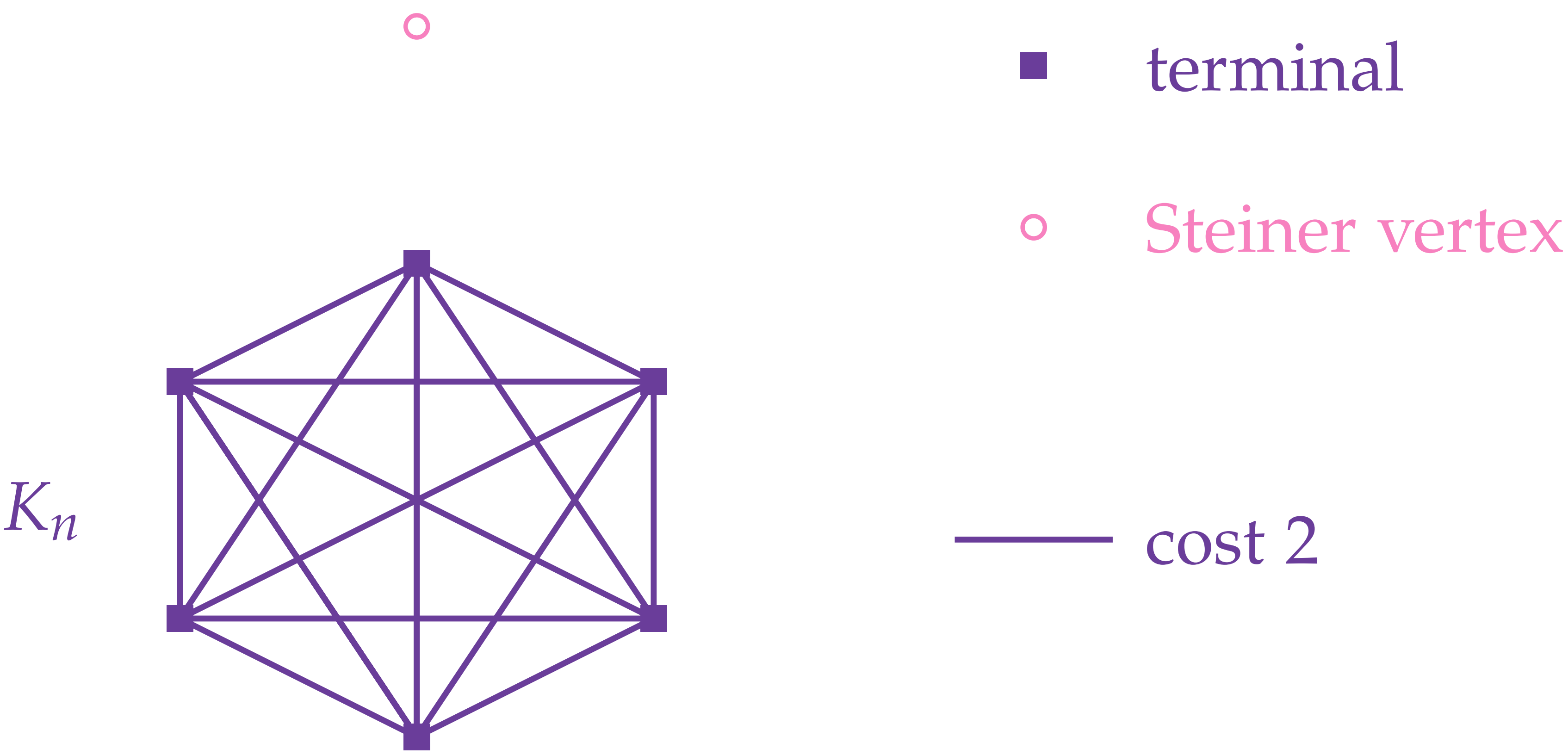
Analysis Sharp?



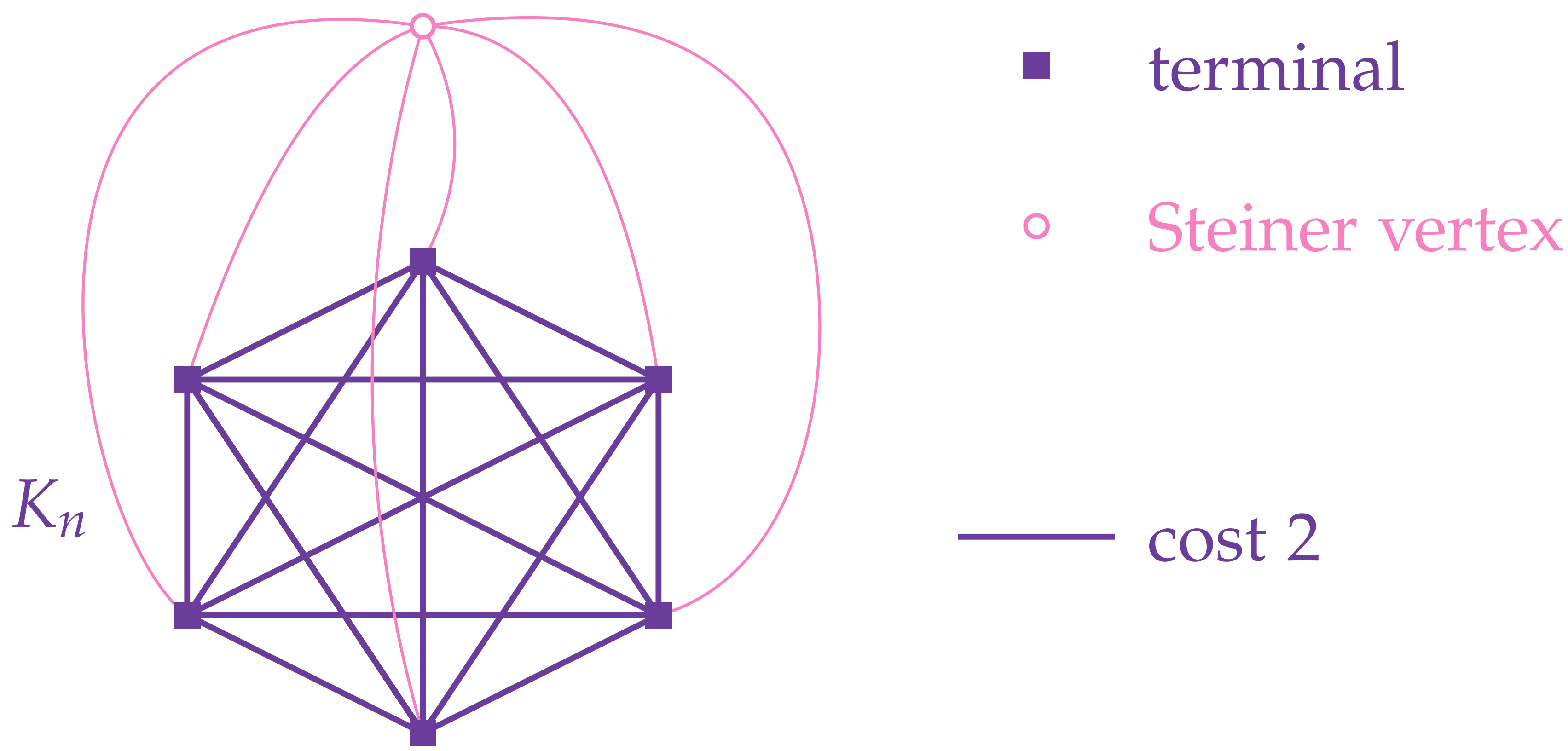
Analysis Sharp?



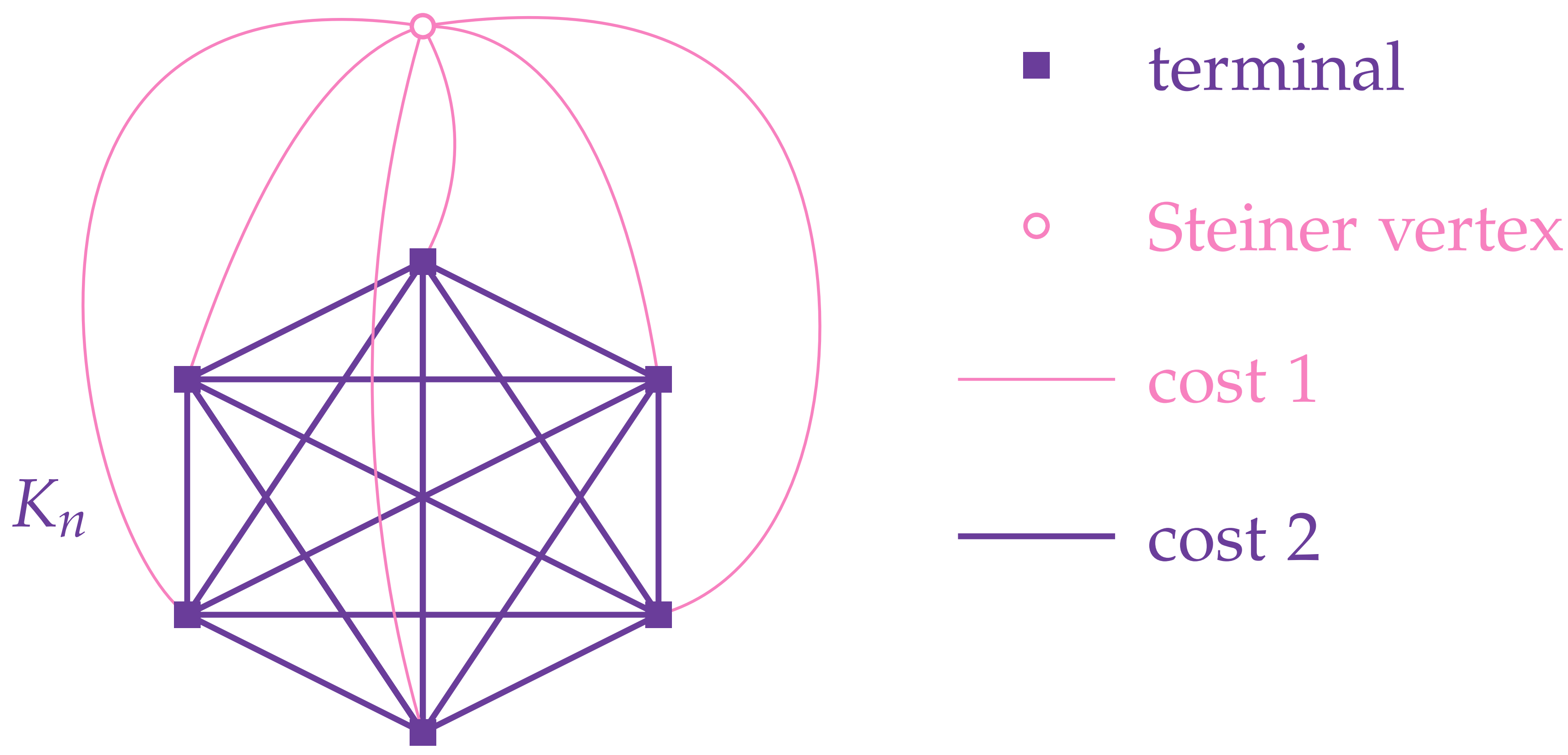
Analysis Sharp?



Analysis Sharp?

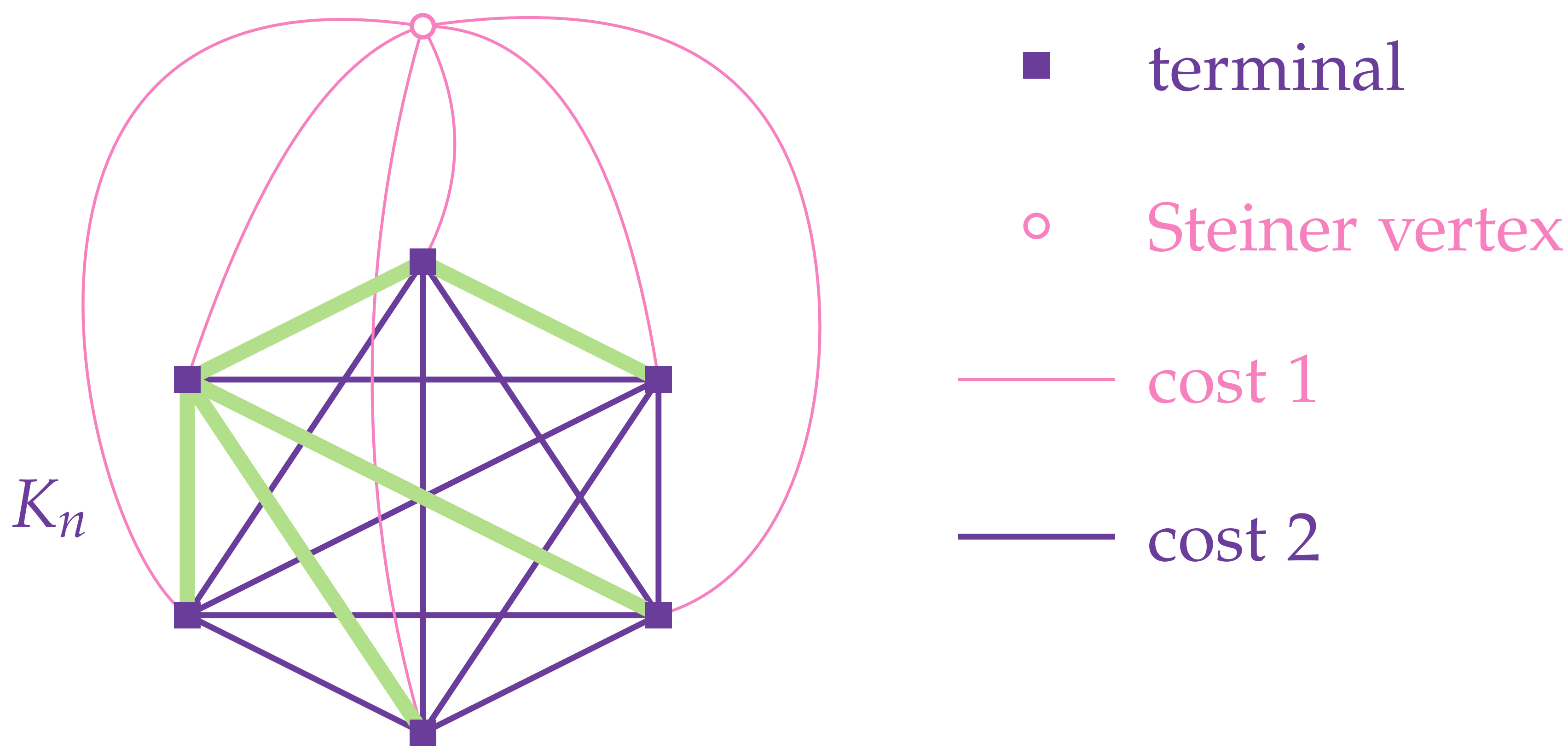


Analysis Sharp?



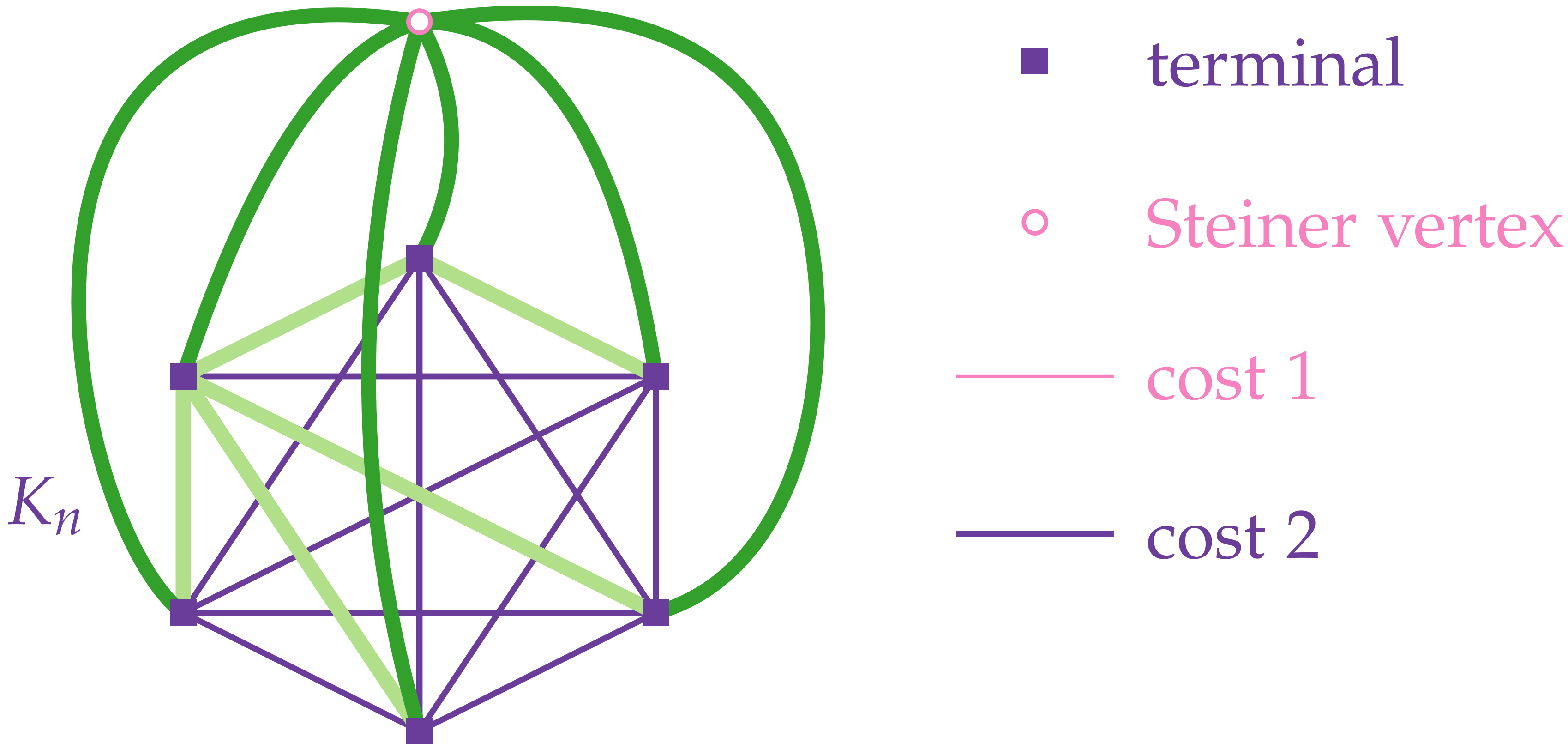
Analysis Sharp?

MST of $G[T]$ with cost $2(n-1)$



Analysis Sharp?

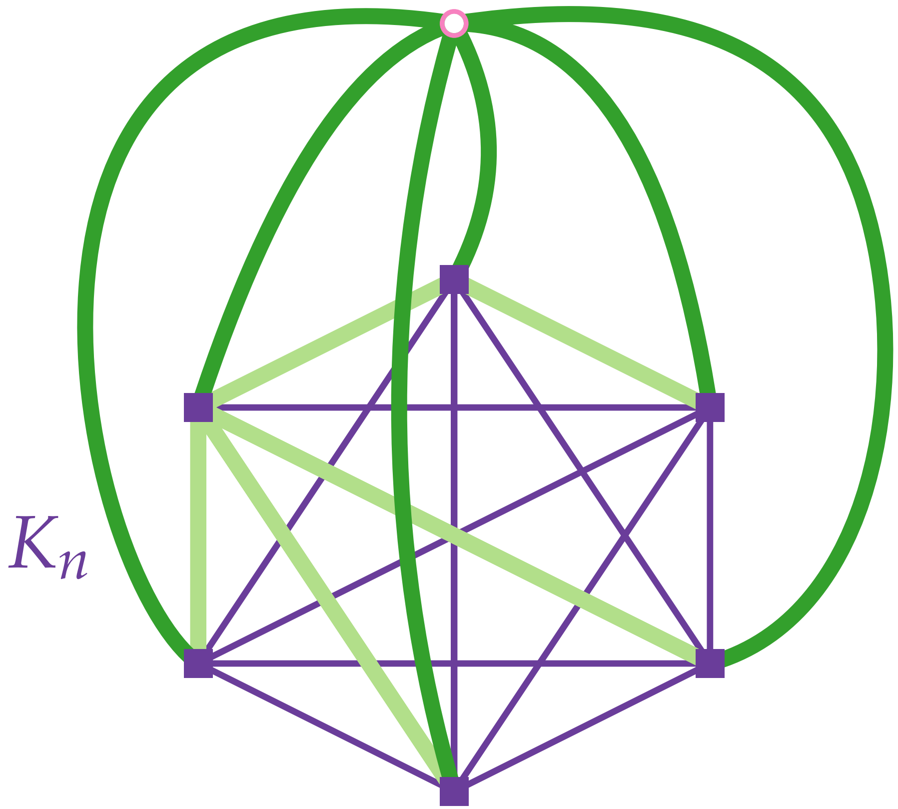
MST of $G[T]$ with cost $2(n-1)$
Optimal solution with cost n

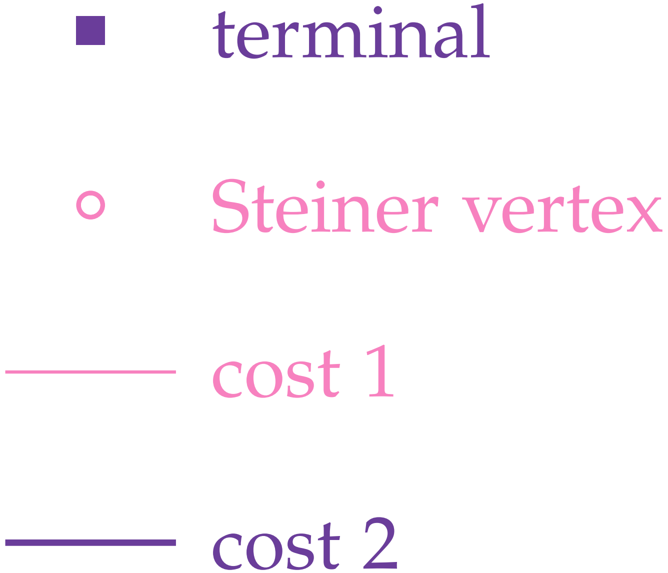


Analysis Sharp?

MST of $G[T]$ with cost $2(n-1)$
Optimal solution with cost n

$$\frac{2(n-1)}{n} \to 2$$



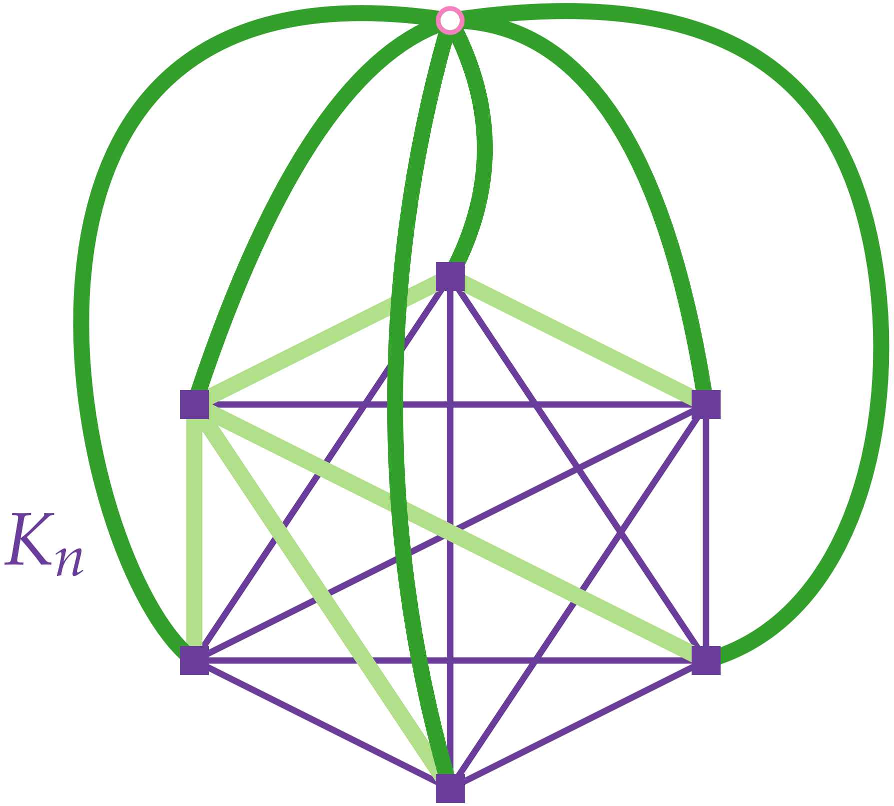


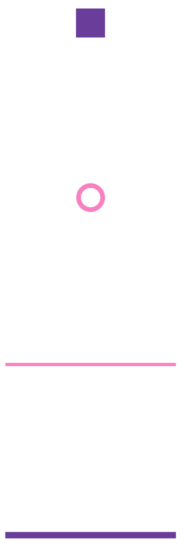
Analysis Sharp?

MST of $G[T]$ with cost $2(n-1)$

Optimal solution with cost n

$$\frac{2(n-1)}{n} \to 2$$



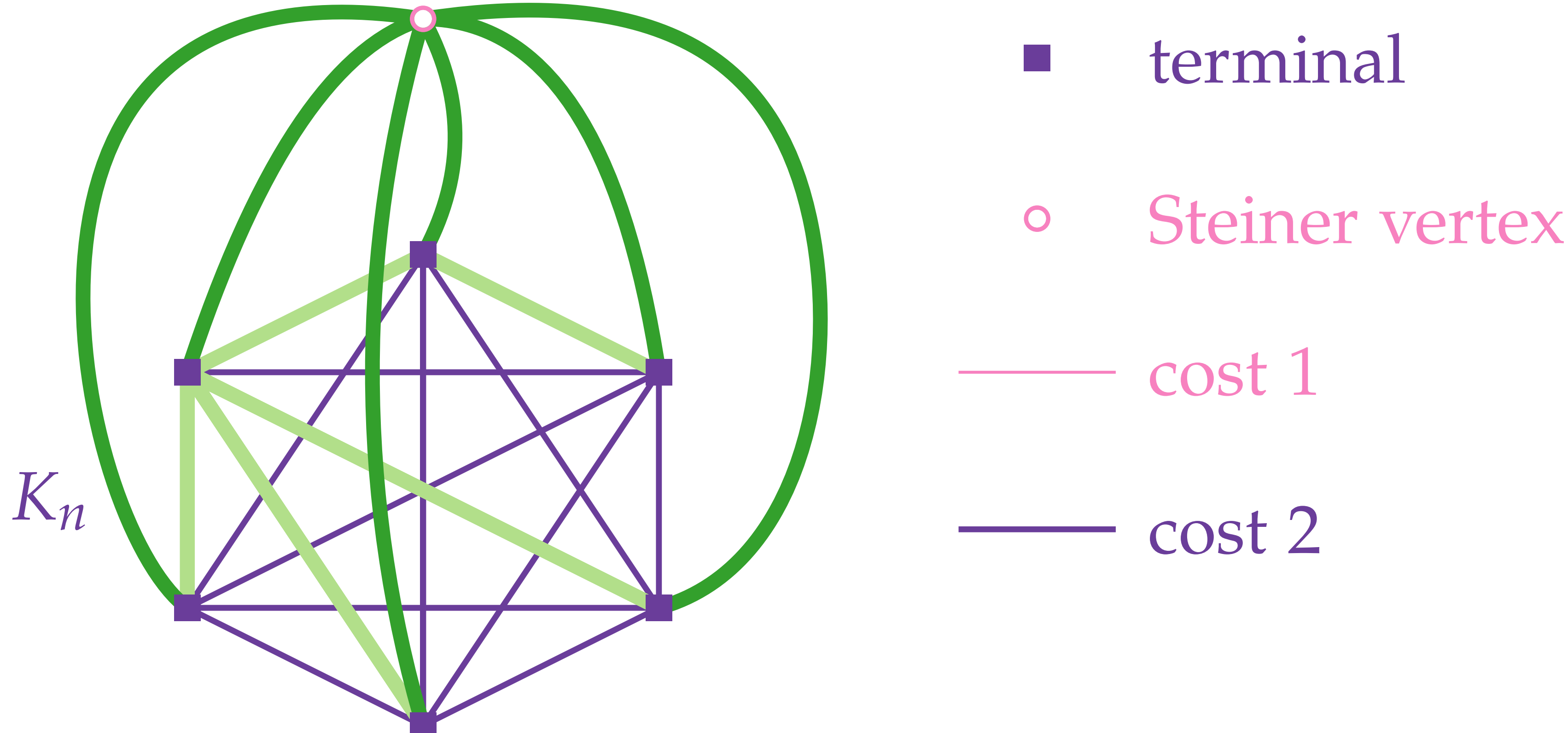


better?

Analysis Sharp?

MST of $G[T]$ with cost $2(n-1)$
Optimal solution with cost n

$$\frac{2(n-1)}{n} \to 2$$



better?
The best-known approximation factor for SteinerTree is $\ln(4) + \varepsilon \approx 1.39$

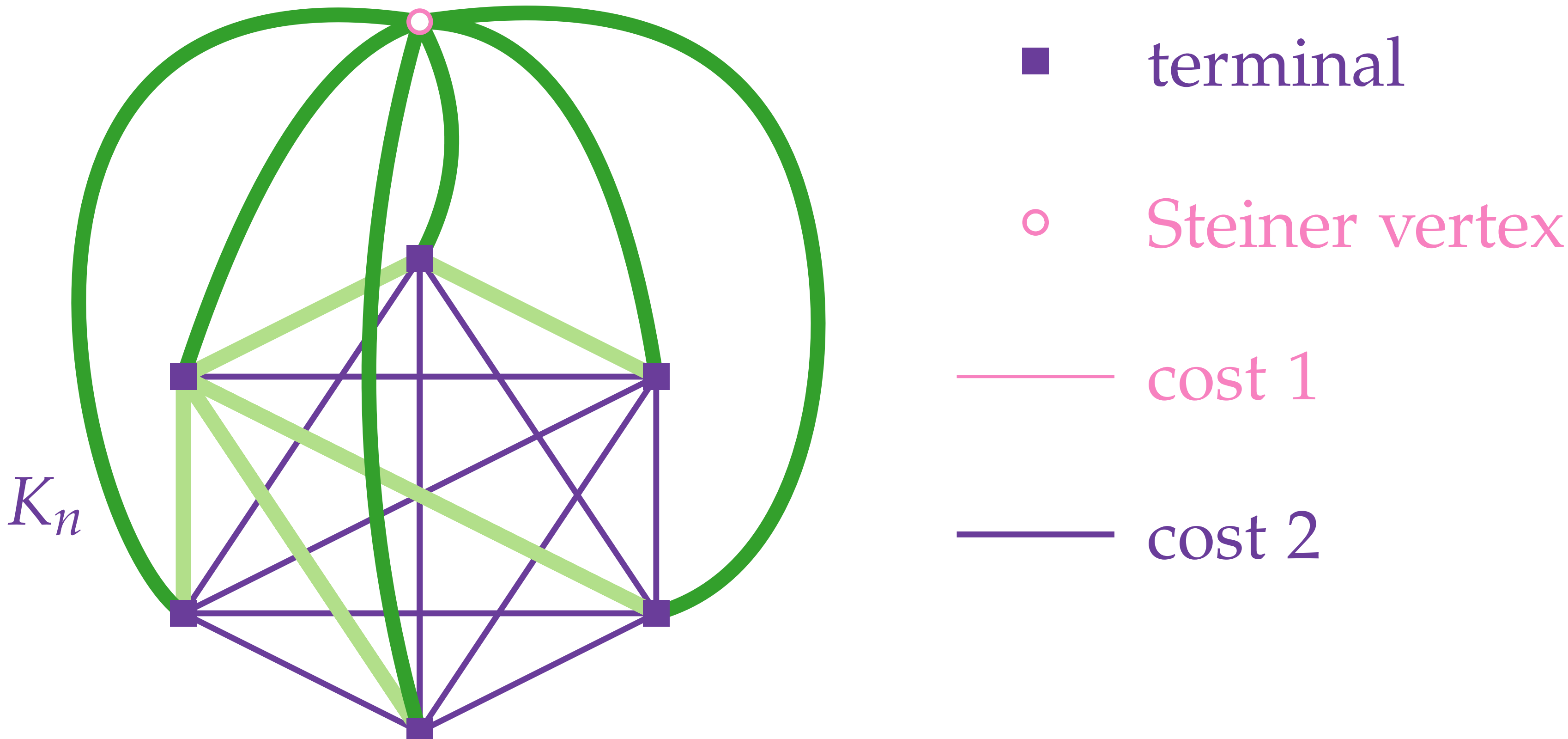
[Byrka, Grandoni, Rothvoß & Sanita '10]

Analysis Sharp?

MST of $G[T]$ with cost $2(n-1)$

Optimal solution with cost n

$$\frac{2(n-1)}{n} \to 2$$



better?

The best-known approximation factor for SteinerTree is $\ln(4) + \varepsilon \approx 1.39$ [Byrka, Grandoni, Rothvoß & Sanita '10]

SteinerTree cannot be approximated within factor $\frac{96}{95} \approx 1.0105$ (unless P=NP) [Chlebik & Chlebikova '08]

Approximation Algorithms

Lecture 3:
SteinerTree and MultiwayCut

Part V:
MultiwayCut

Joachim Spoerhase

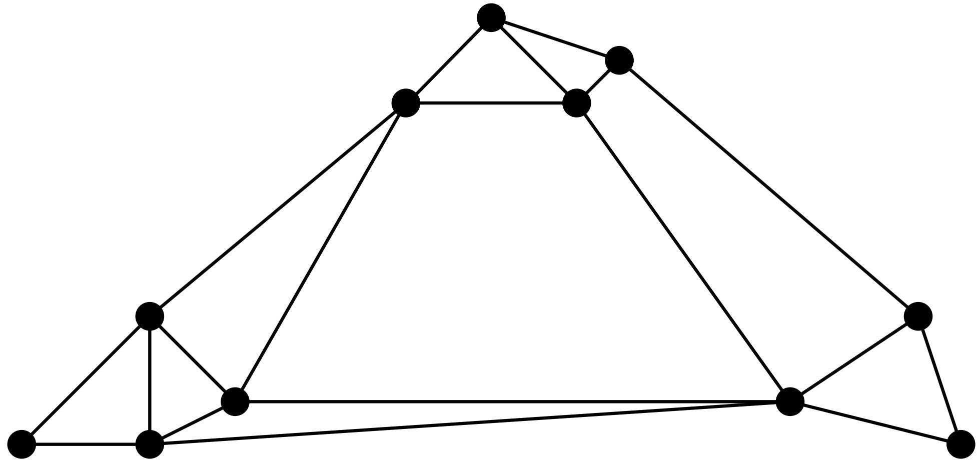
Winter 2021/22

MultiwayCut

Given: A connected graph $G = (V, E)$

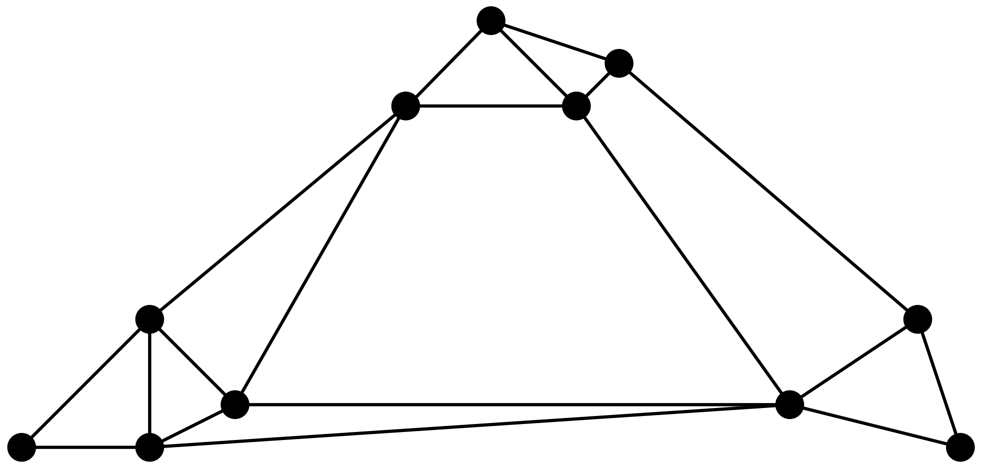
MultiwayCut

Given: A connected graph $G = (V, E)$



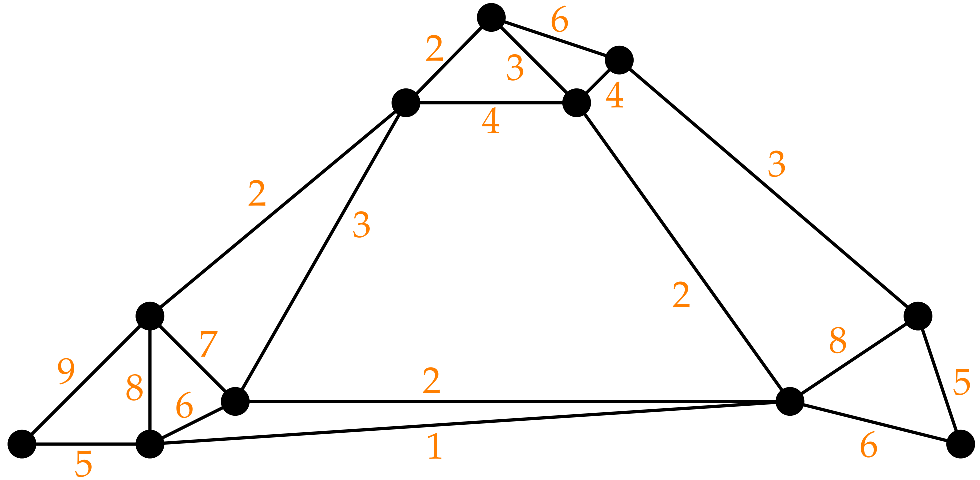
MultiwayCut

Given: A connected graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{Q}^+$



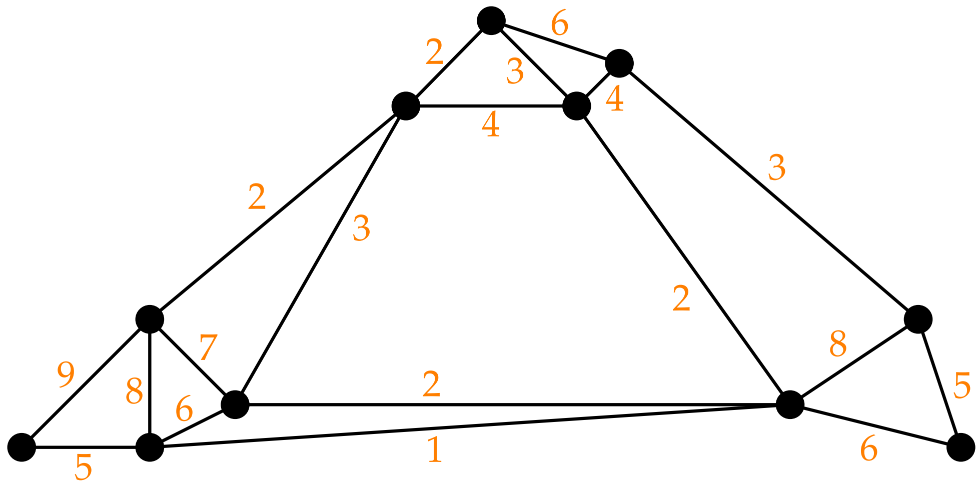
MultiwayCut

Given: A connected graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{Q}^+$



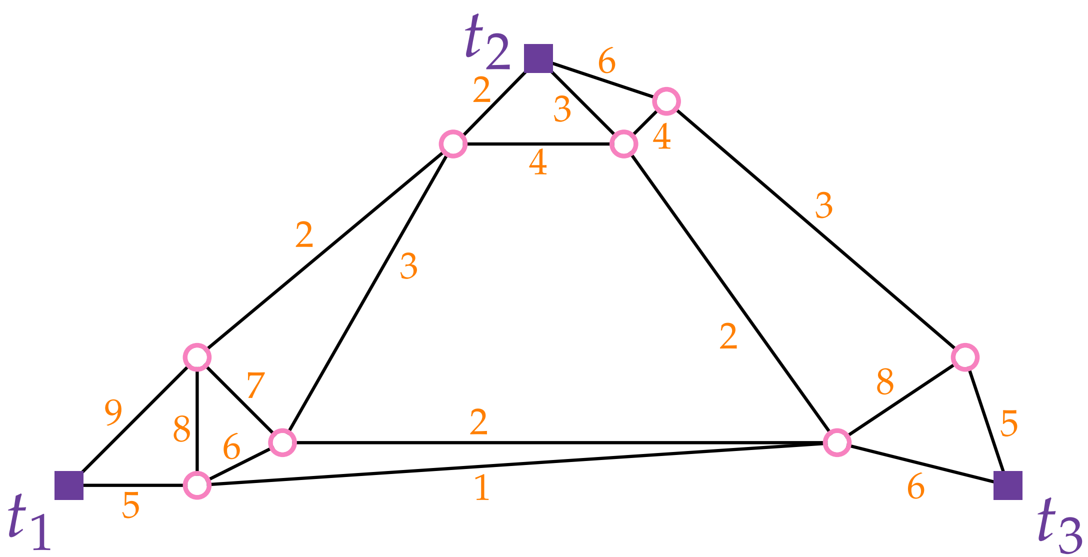
MultiwayCut

Given: A connected graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{Q}^+$ and a set $T = \{t_1, \ldots, t_k\} \subseteq V$ of **terminals**.



MultiwayCut

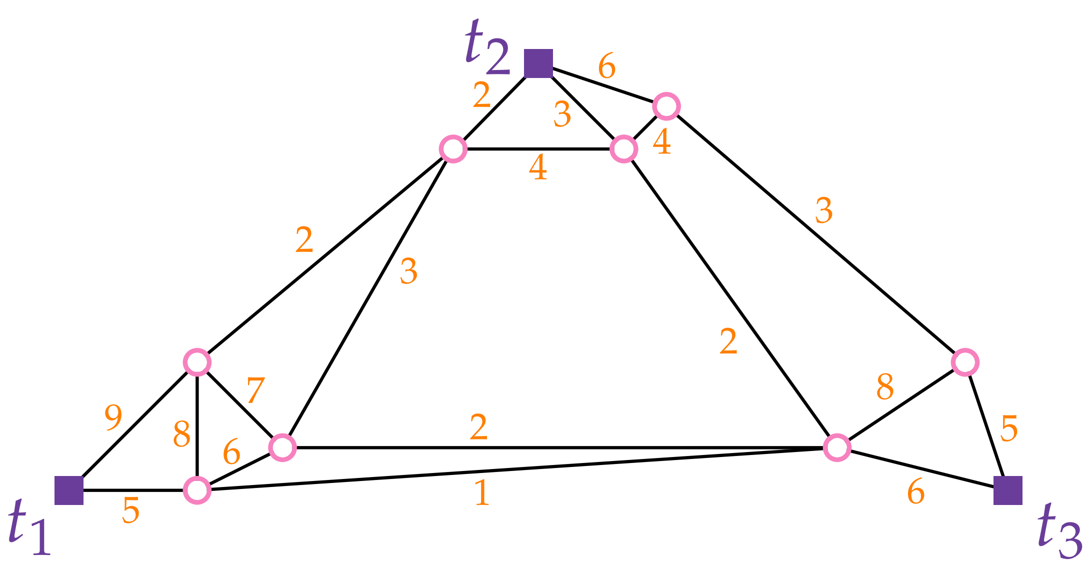
Given: A connected graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{Q}^+$ and a set $T = \{t_1, \ldots, t_k\} \subseteq V$ of **terminals**.



MultiwayCut

Given: A connected graph $G = (V, E)$ with edge costs $c \colon E \to \mathbb{Q}^+$ and a set $T = \{t_1, \dots, t_k\} \subseteq V$ of **terminals**.

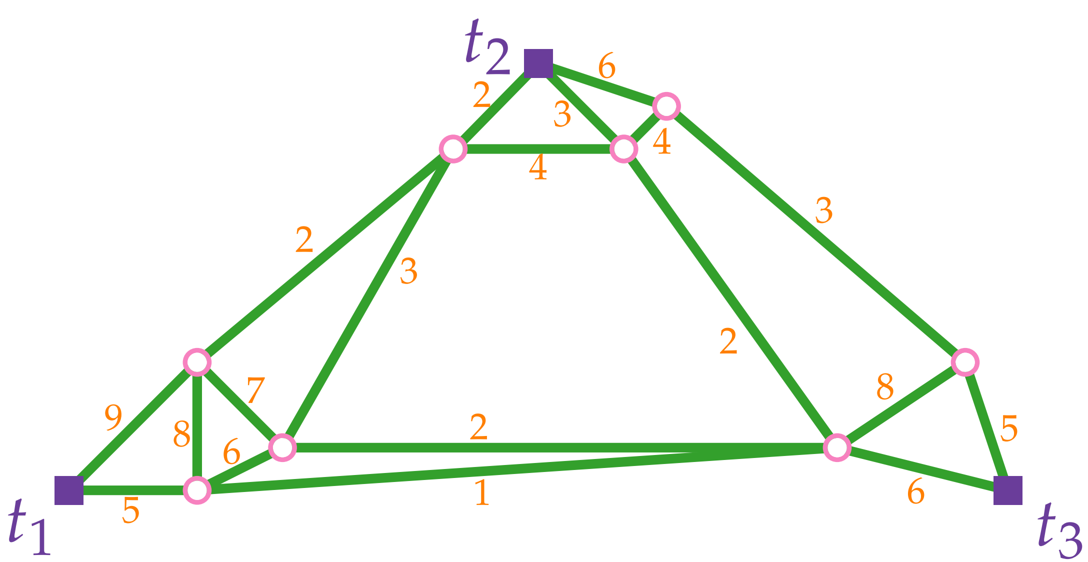
A **multiway cut** of T is a subset E' of edges such that no two terminals in the graph $(V, E - E')$ are connected.



MultiwayCut

Given: A connected graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{Q}^+$ and a set $T = \{t_1, \ldots, t_k\} \subseteq V$ of **terminals**.

A **multiway cut** of T is a subset E' of edges such that no two terminals in the graph $(V, E - E')$ are connected.

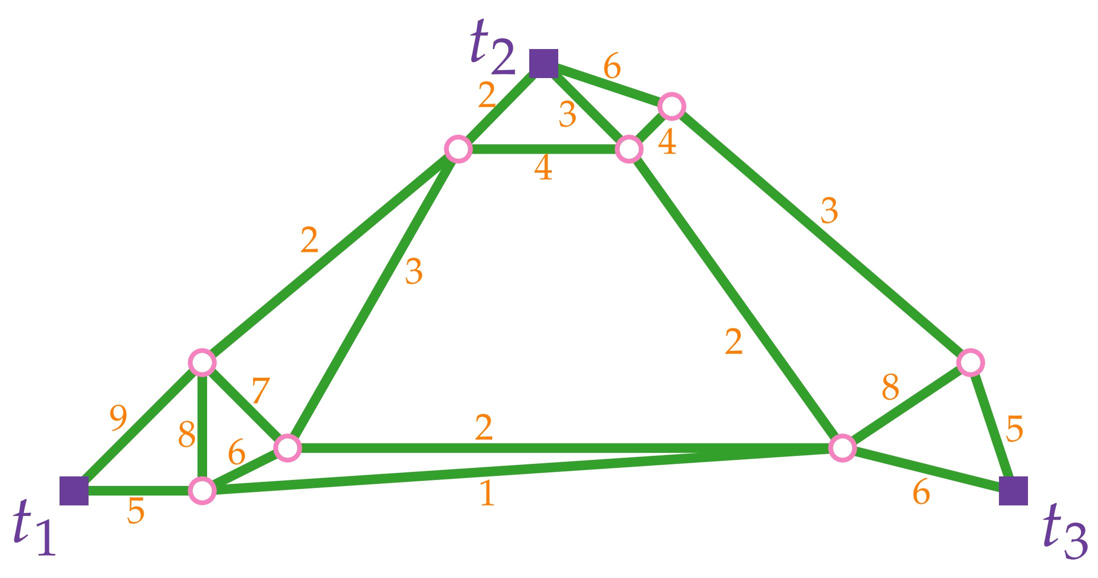


MultiwayCut

Given: A connected graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{Q}^+$ and a set $T = \{t_1, \dots, t_k\} \subseteq V$ of **terminals**.

A **multiway cut** of T is a subset E' of edges such that no two terminals in the graph $(V, E - E')$ are connected.

Find: A minimum cost multiway cut of T.

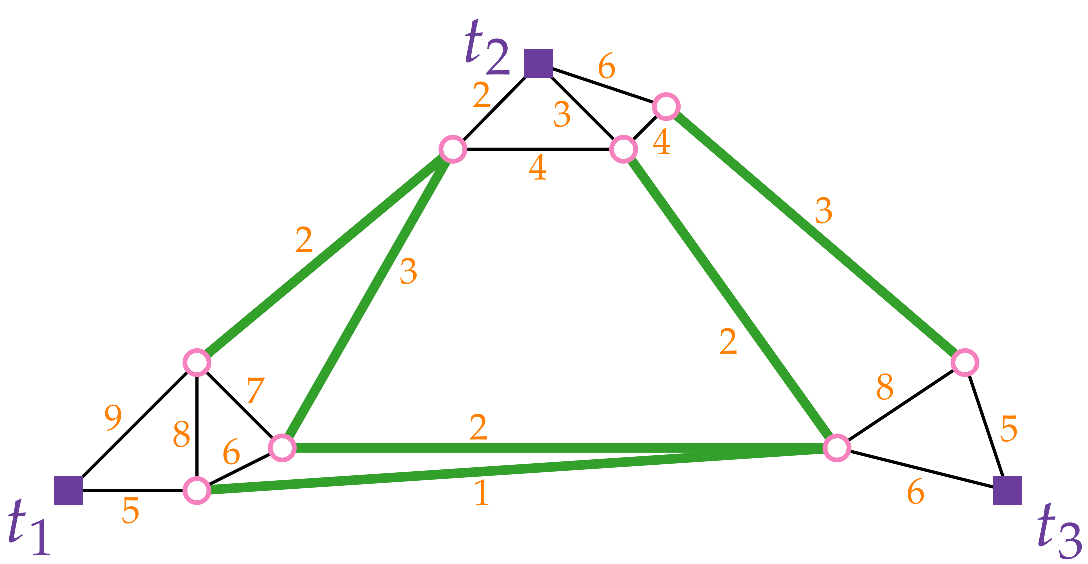


MultiwayCut

Given: A connected graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{Q}^+$ and a set $T = \{t_1, \ldots, t_k\} \subseteq V$ of **terminals**.

A **multiway cut** of T is a subset E' of edges such that no two terminals in the graph $(V, E - E')$ are connected.

Find: A minimum cost multiway cut of T.

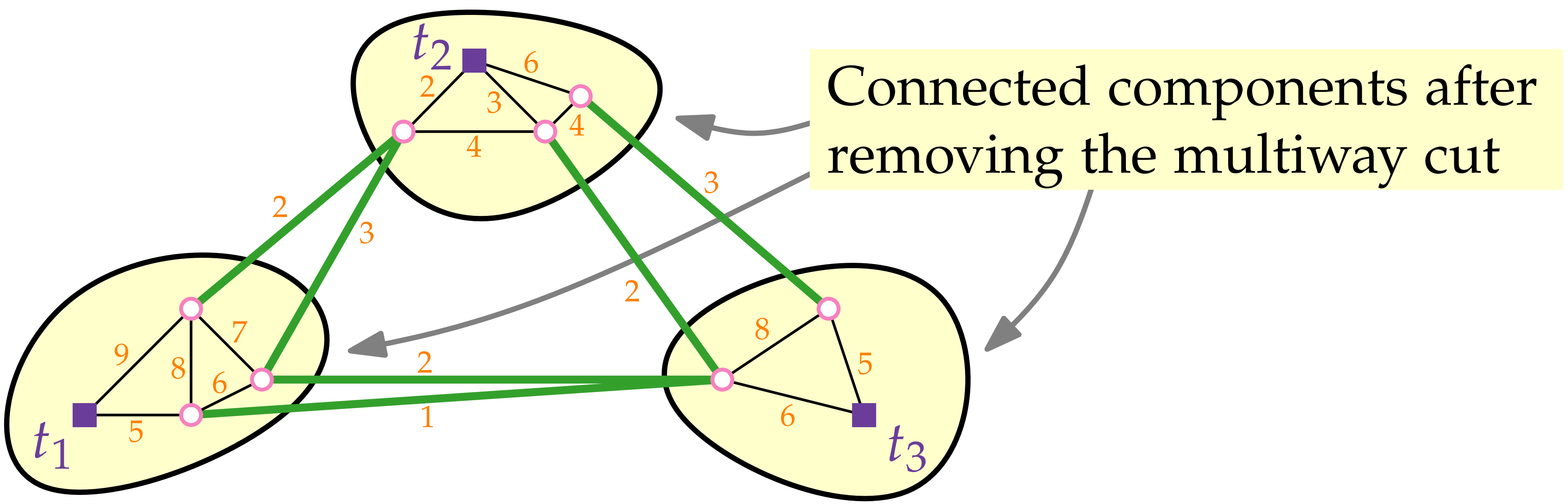


MultiwayCut

Given: A connected graph $G = (V, E)$ with edge costs $c \colon E \to \mathbb{Q}^+$ and a set $T = \{t_1, \ldots, t_k\} \subseteq V$ of **terminals**.

A **multiway cut** of T is a subset E' of edges such that no two terminals in the graph $(V, E - E')$ are connected.

Find: A minimum cost multiway cut of T.

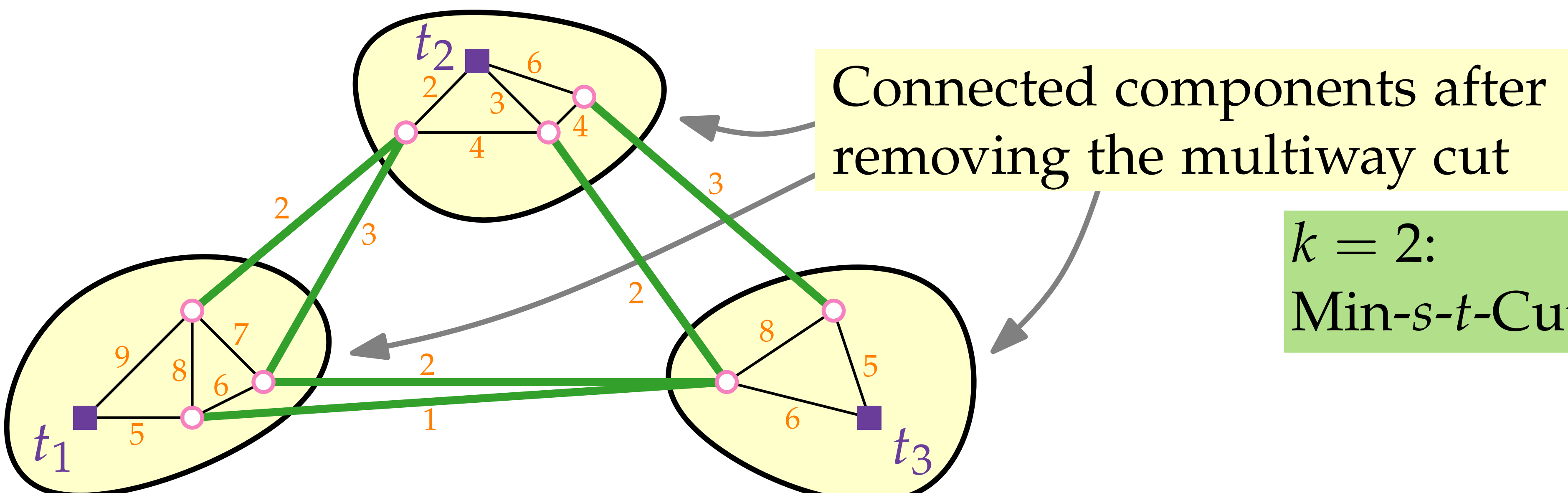


MultiwayCut

Given: A connected graph $G = (V, E)$ with edge costs $c \colon E \to \mathbb{Q}^+$ and a set $T = \{t_1, \dots, t_k\} \subseteq V$ of **terminals**.

A **multiway cut** of T is a subset E' of edges such that no two terminals in the graph $(V, E - E')$ are connected.

Find: A minimum cost multiway cut of T.



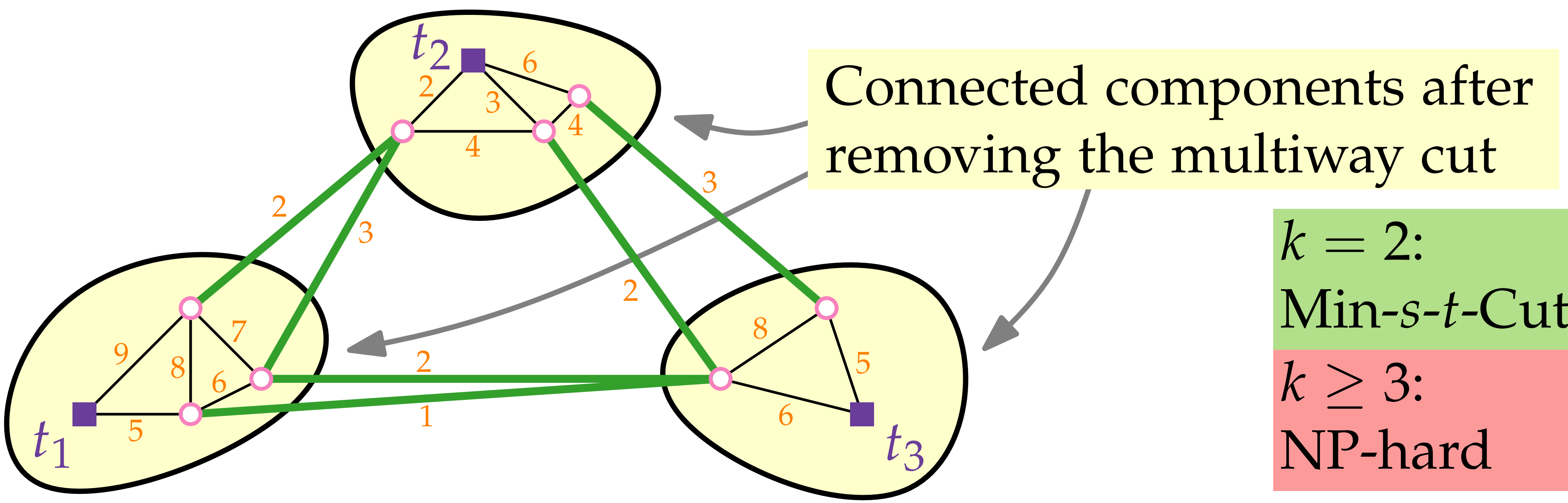
$k = 2$:
Min-s-t-Cut

MultiwayCut

Given: A connected graph $G = (V, E)$ with edge costs $c\colon E \to \mathbb{Q}^+$ and a set $T = \{t_1, \ldots, t_k\} \subseteq V$ of **terminals**.

A **multiway cut** of T is a subset E' of edges such that no two terminals in the graph $(V, E - E')$ are connected.

Find: A minimum cost multiway cut of T.



$k = 2$: Min-s-t-Cut

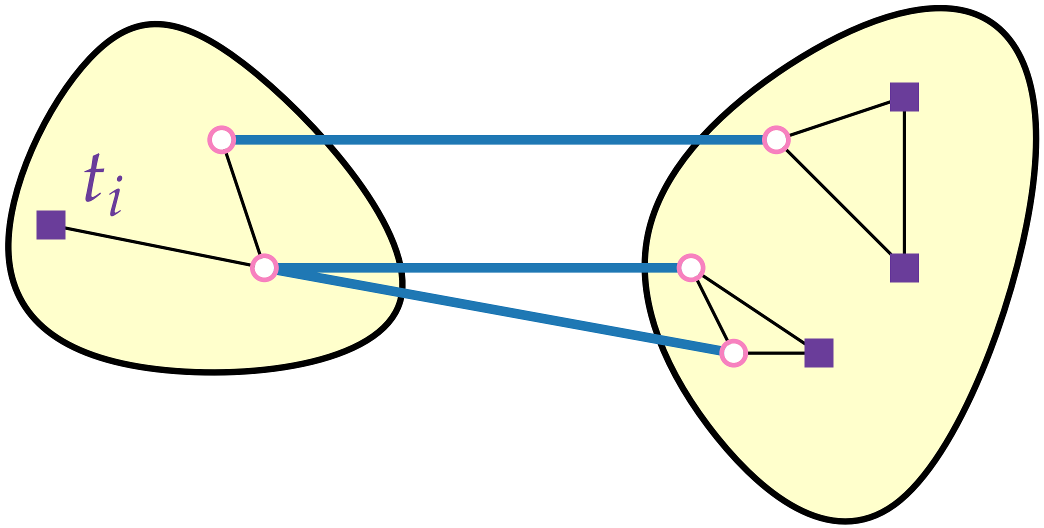
$k \geq 3$: NP-hard

Isolating Cuts

An **isolating cut** for a terminal t_i is a set of edges separating t_i from all other terminals.

Isolating Cuts

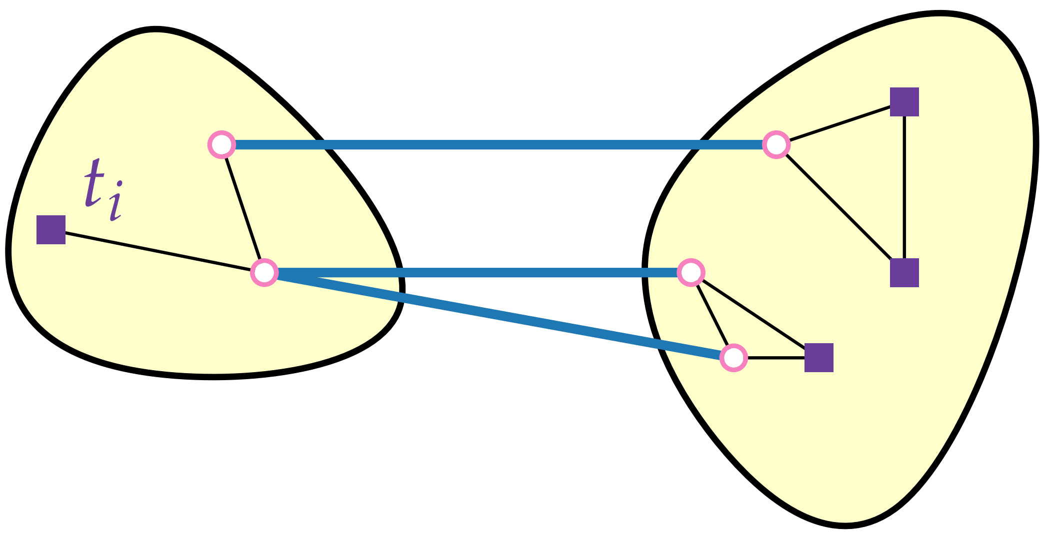
An **isolating cut** for a terminal t_i is a set of edges separating t_i from all other terminals.



Isolating Cuts

An **isolating cut** for a terminal t_i is a set of edges separating t_i from all other terminals.

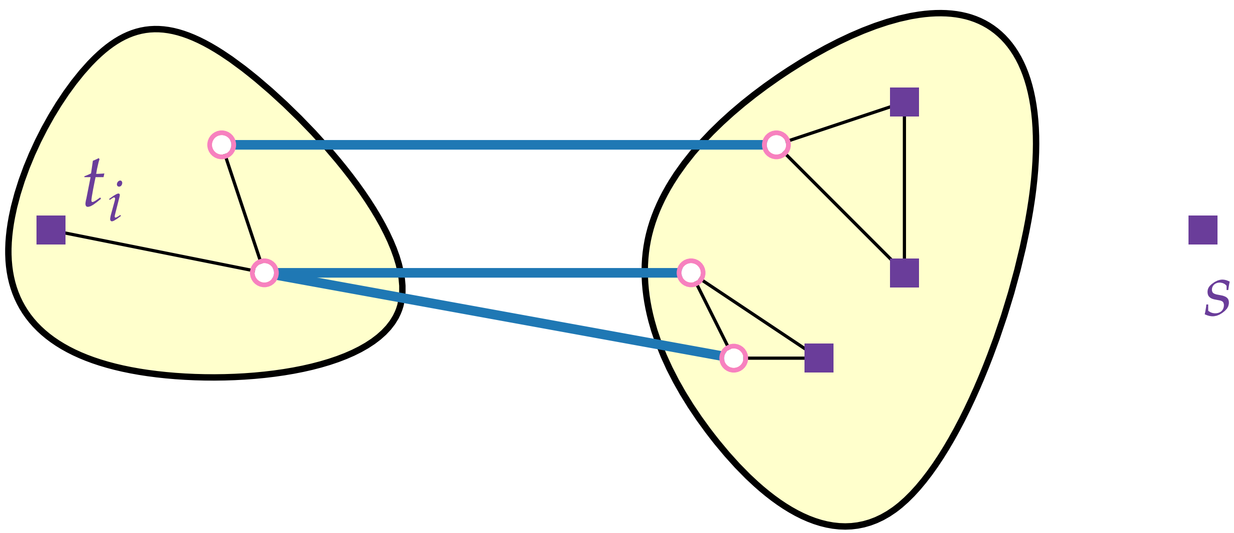
Minimum cost isolating cut can be computed efficiently!



Isolating Cuts

An **isolating cut** for a terminal t_i is a set of edges separating t_i from all other terminals.

Minimum cost isolating cut can be computed efficiently!

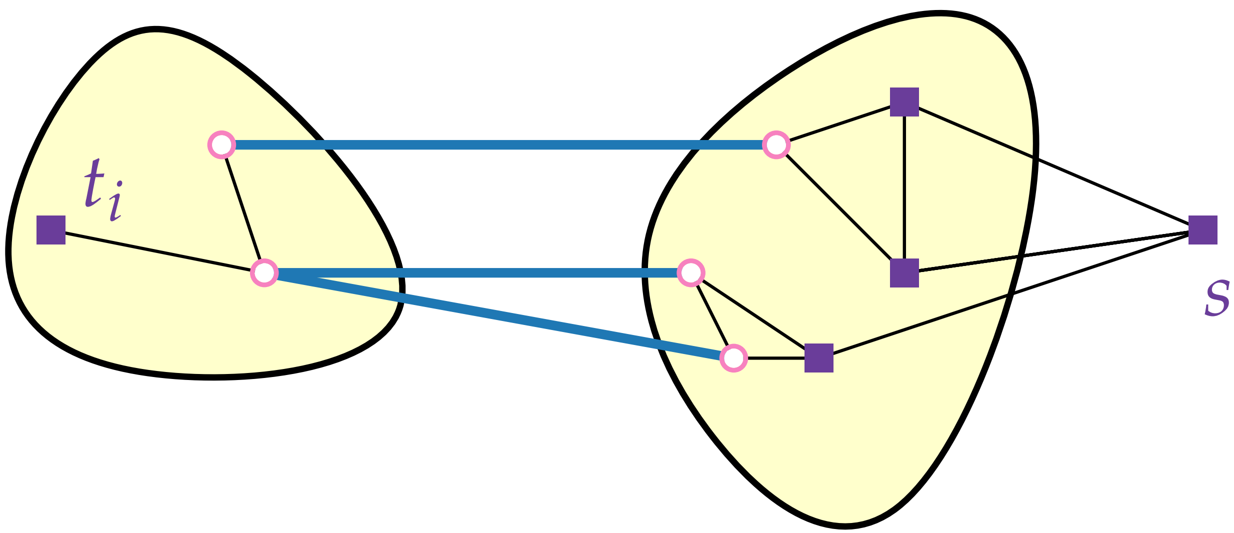


Add dummy terminal s

Isolating Cuts

An **isolating cut** for a terminal t_i is a set of edges separating t_i from all other terminals.

Minimum cost isolating cut can be computed efficiently!

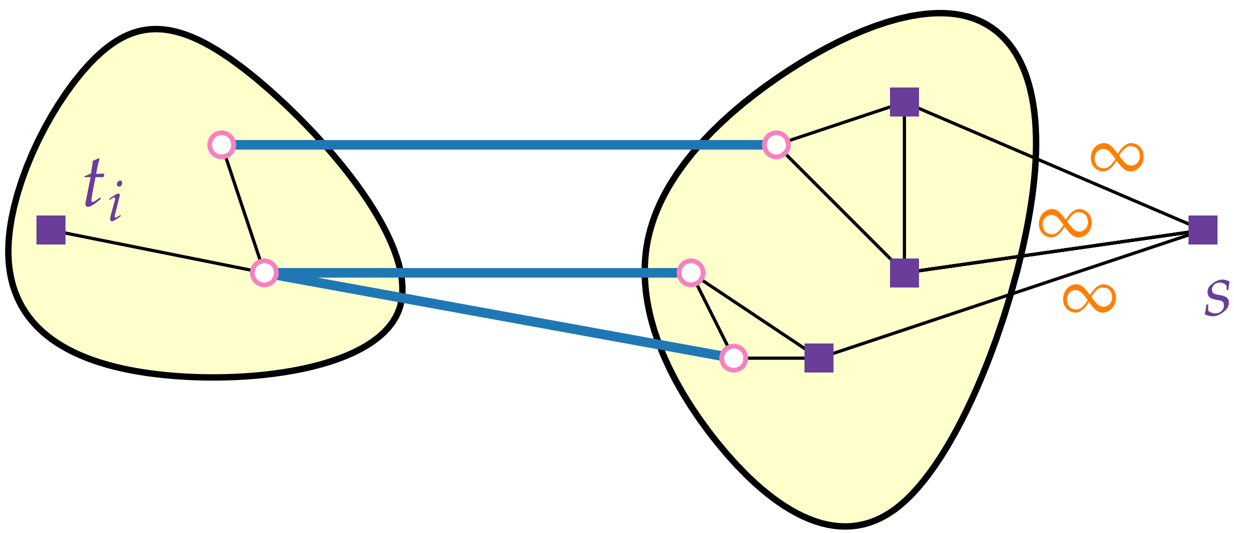


Add dummy terminal s

Isolating Cuts

An **isolating cut** for a terminal t_i is a set of edges separating t_i from all other terminals.

Minimum cost isolating cut can be computed efficiently!

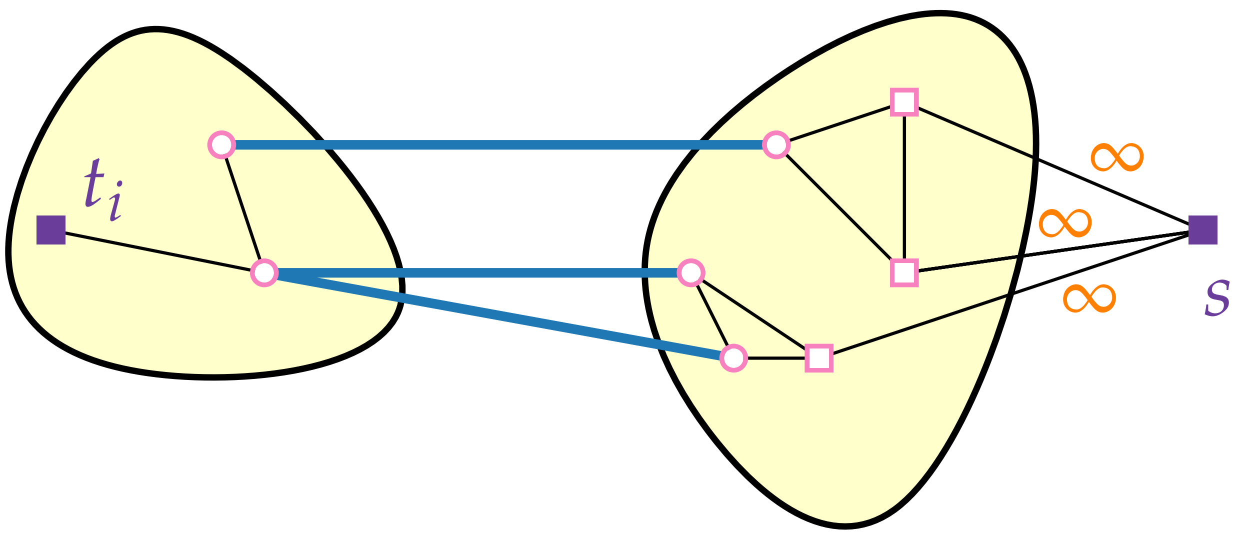


Add dummy terminal s

Isolating Cuts

An **isolating cut** for a terminal t_i is a set of edges separating t_i from all other terminals.

Minimum cost isolating cut can be computed efficiently!

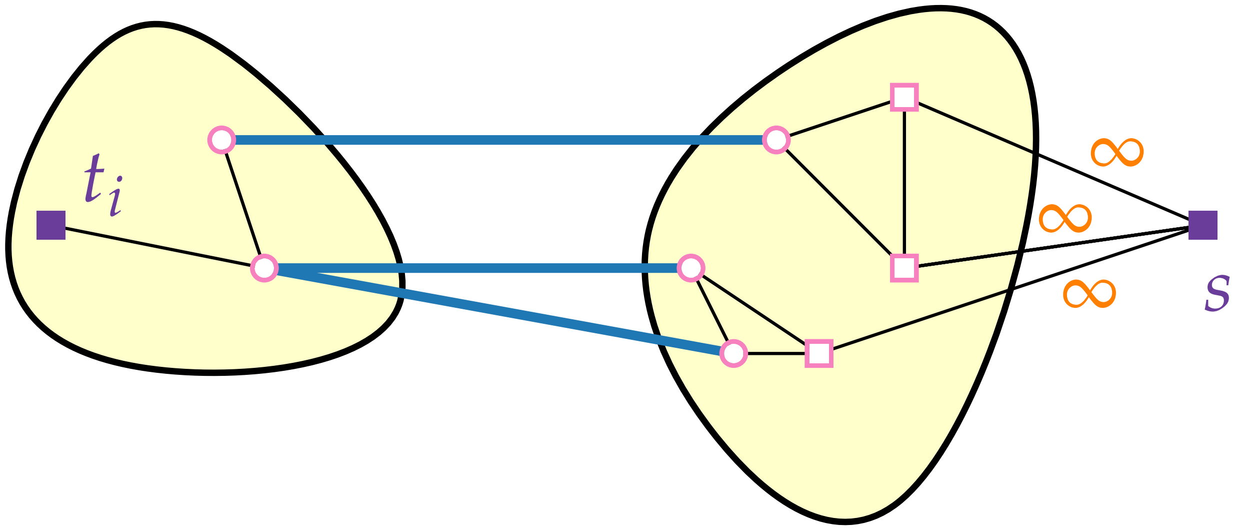


Add dummy terminal s

Isolating Cuts

An **isolating cut** for a terminal t_i is a set of edges separating t_i from all other terminals.

Minimum cost isolating cut can be computed efficiently!



Add dummy terminal s and find minimum cost s-t_i-cut.

Approximation Algorithms

Lecture 3:
SteinerTree and MultiwayCut

Part VI:
Algorithm for MultiwayCut

Joachim Spoerhase

Winter 2021/22

Algorithm MultiwayCut

For $i = 1, \ldots, k$:
Compute a minimum cost isolating cut C_i for t_i.

Algorithm MultiwayCut

For $i = 1, \ldots, k$:
Compute a minimum cost isolating cut C_i for t_i.

Return the union of $\mathcal{C}$ of the $k-1$ cheapest such isolating cuts.

Algorithm MultiwayCut

For $i = 1, \ldots, k$:
Compute a minimum cost isolating cut C_i for t_i.

Return the union of $\mathcal{C}$ of the $k - 1$ cheapest such isolating cuts.

In other words:
Ignore the most expensive of the isolating cuts $C_1, \ldots, C_k$.

Algorithm MultiwayCut

For $i = 1, \ldots, k$:
Compute a minimum cost isolating cut C_i for t_i.

Return the union of $\mathcal{C}$ of the $k-1$ cheapest such isolating cuts.

In other words:
Ignore the most expensive of the isolating cuts $C_1, \ldots, C_k$.

$\Rightarrow c(\mathcal{C})$ **?** $\sum_{i=1}^{k} c(C_i)$

Algorithm MultiwayCut

For $i = 1, \ldots, k$:
Compute a minimum cost isolating cut C_i for t_i.

Return the union of $\mathcal{C}$ of the $k-1$ cheapest such isolating cuts.

In other words:
Ignore the most expensive of the isolating cuts $C_1, \ldots, C_k$.

$$\Rightarrow c(\mathcal{C}) \leq \sum_{i=1}^{k} c(C_i)$$

Algorithm MultiwayCut

For $i = 1, \ldots, k$:
Compute a minimum cost isolating cut C_i for t_i.

Return the union of $\mathcal{C}$ of the $k-1$ cheapest such isolating cuts.

In other words:
Ignore the most expensive of the isolating cuts $C_1, \ldots, C_k$.

$$\Rightarrow c(\mathcal{C}) \leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} c(C_i)$$ because:

Algorithm MultiwayCut

For $i = 1, \ldots, k$:
Compute a minimum cost isolating cut C_i for t_i.

Return the union of $\mathcal{C}$ of the $k-1$ cheapest such isolating cuts.

In other words:
Ignore the most expensive of the isolating cuts $C_1, \ldots, C_k$.

$\Rightarrow c(\mathcal{C}) \leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} c(C_i)$ because:

for the most expensive cut of $C_1, \ldots, C_k$, say C_1, we have

$$c(C_1) \geq$$

Algorithm MultiwayCut

For $i = 1, \dots, k$:
Compute a minimum cost isolating cut C_i for t_i.

Return the union of $\mathcal{C}$ of the $k-1$ cheapest such isolating cuts.

In other words:
Ignore the most expensive of the isolating cuts $C_1, \dots, C_k$.

$$\Rightarrow c(\mathcal{C}) \leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} c(C_i) \quad \text{because:}$$

for the most expensive cut of $C_1, \dots, C_k$, say C_1, we have

$$c(C_1) \geq \frac{1}{k} \sum_{i=1}^{k} c(C_i).$$

Approximation Factor

Theorem. This algorithm is a factor-( )-approximation algorithm for MultiwayCut.

Approximation Factor

Theorem. This algorithm is a factor-$(2 - 2/k)$-approximation algorithm for MultiwayCut.

Approximation Factor

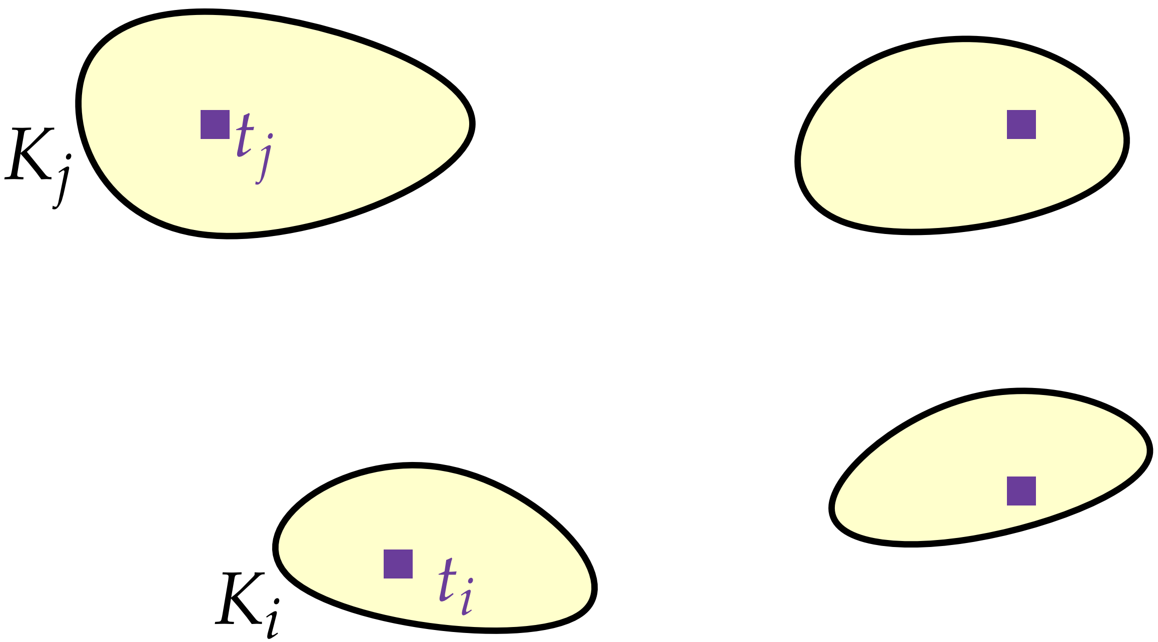
Theorem. This algorithm is a factor-($2 - 2/k$)-approximation algorithm for MultiwayCut.

Proof. Consider optimal multiway cut A:

Approximation Factor

Theorem. This algorithm is a factor-($2 - 2/k$)-approximation algorithm for MultiwayCut.

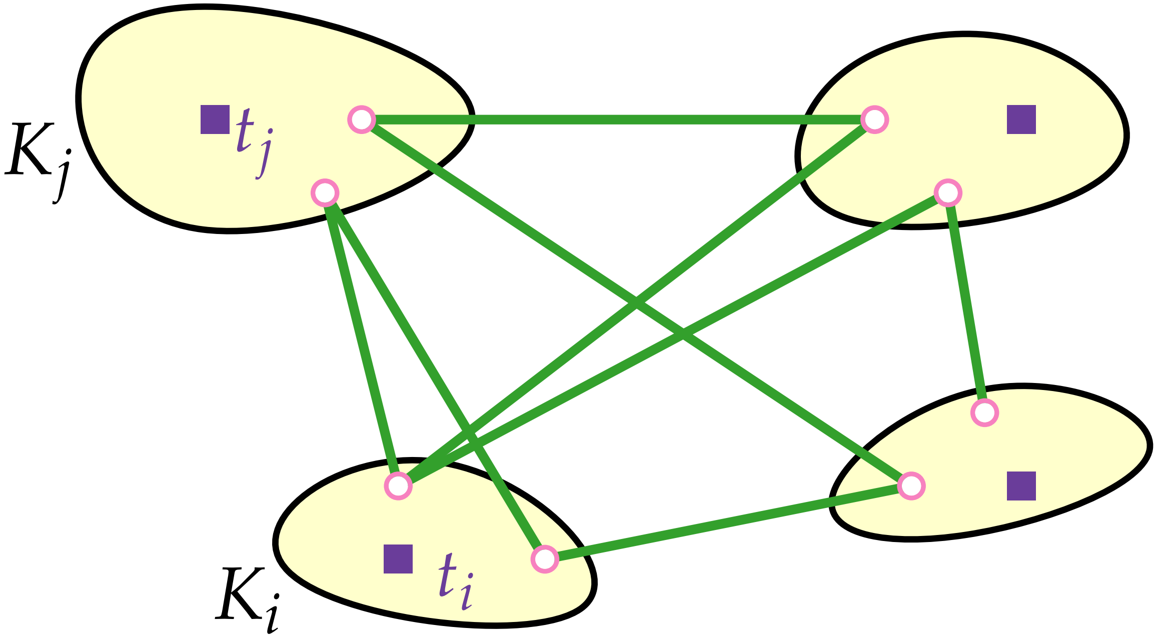
Proof. Consider optimal multiway cut A:



Approximation Factor

Theorem. This algorithm is a factor-$(2 - 2/k)$-approximation algorithm for MULTIWAYCUT.

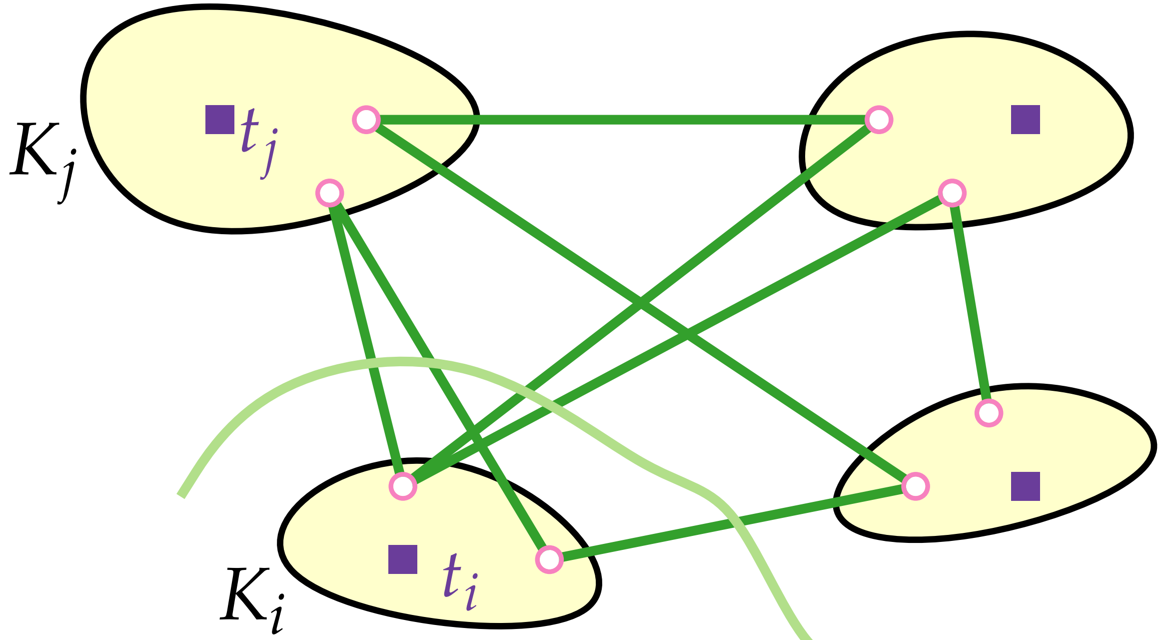
Proof. Consider optimal multiway cut A:



Approximation Factor

Theorem. This algorithm is a factor-($2 - 2/k$)-approximation algorithm for MULTIWAYCUT.

Proof. Consider optimal multiway cut A:

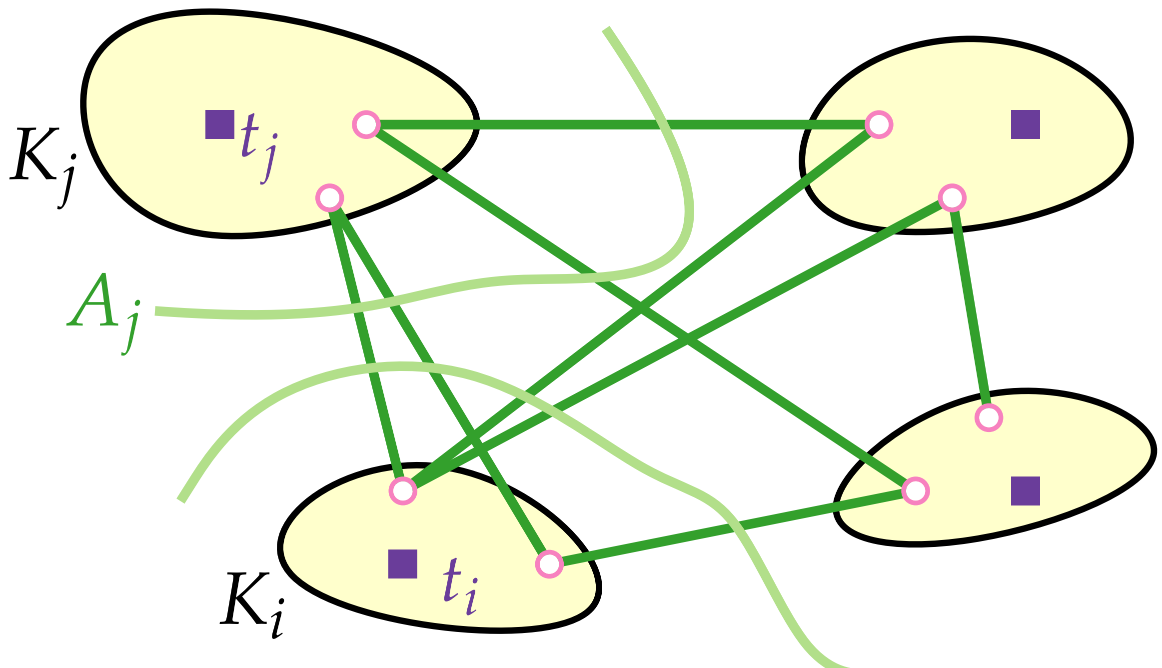


$A_i = \{uv \in A : u \in K_i, v \notin K_i\}$

Approximation Factor

Theorem. This algorithm is a factor-($2 - 2/k$)-approximation algorithm for MULTIWAYCUT.

Proof. Consider optimal multiway cut A:

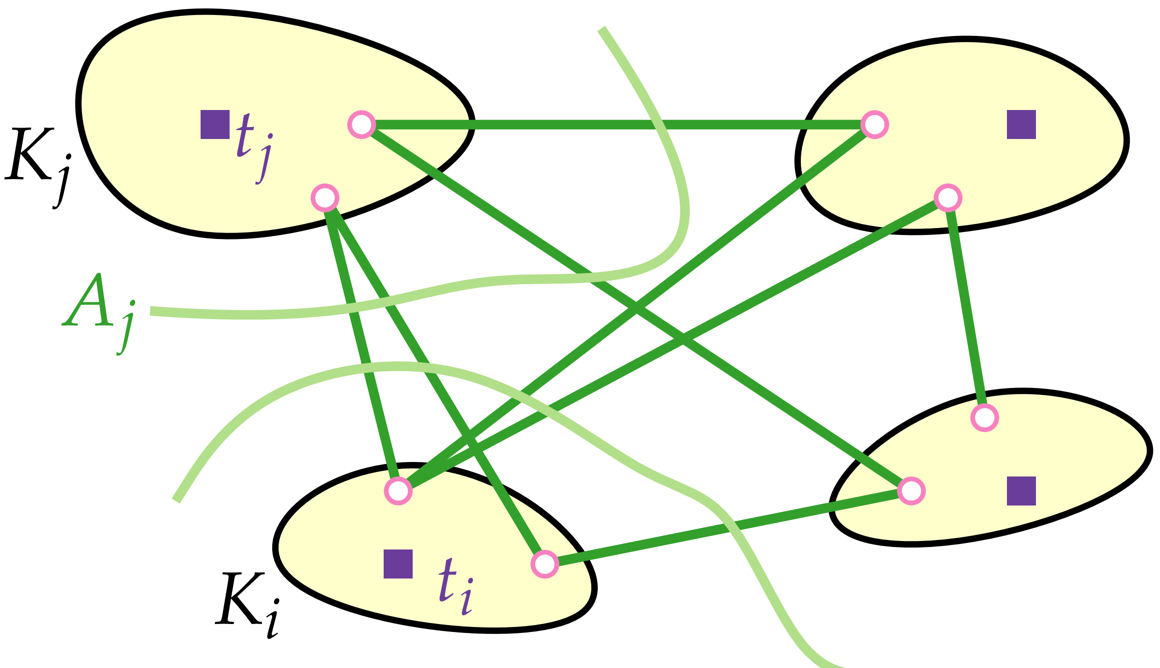


$A_i = \{uv \in A : u \in K_i, v \notin K_i\}$

Approximation Factor

Theorem. This algorithm is a factor-($2 - 2/k$)-approximation algorithm for MultiwayCut.

Proof. Consider optimal multiway cut A:



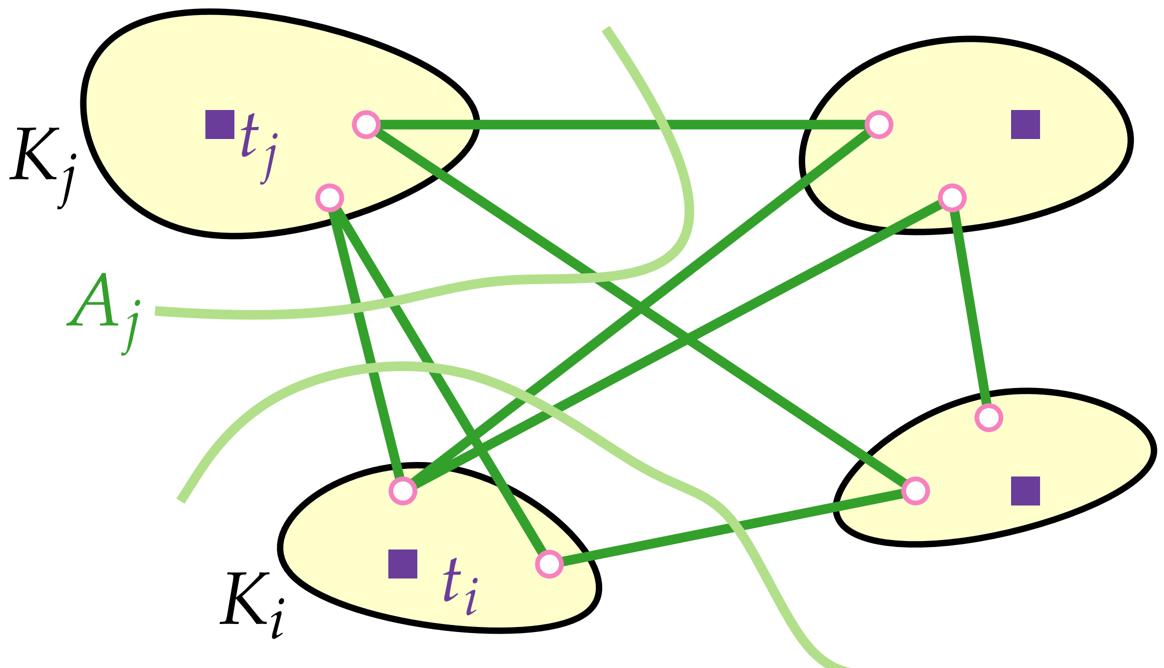
$A_i = \{uv \in A \colon u \in K_i, v \notin K_i\}$

Observation. $A =$

Approximation Factor

Theorem. This algorithm is a factor-($2 - 2/k$)-approximation algorithm for MultiwayCut.

Proof. Consider optimal multiway cut A:



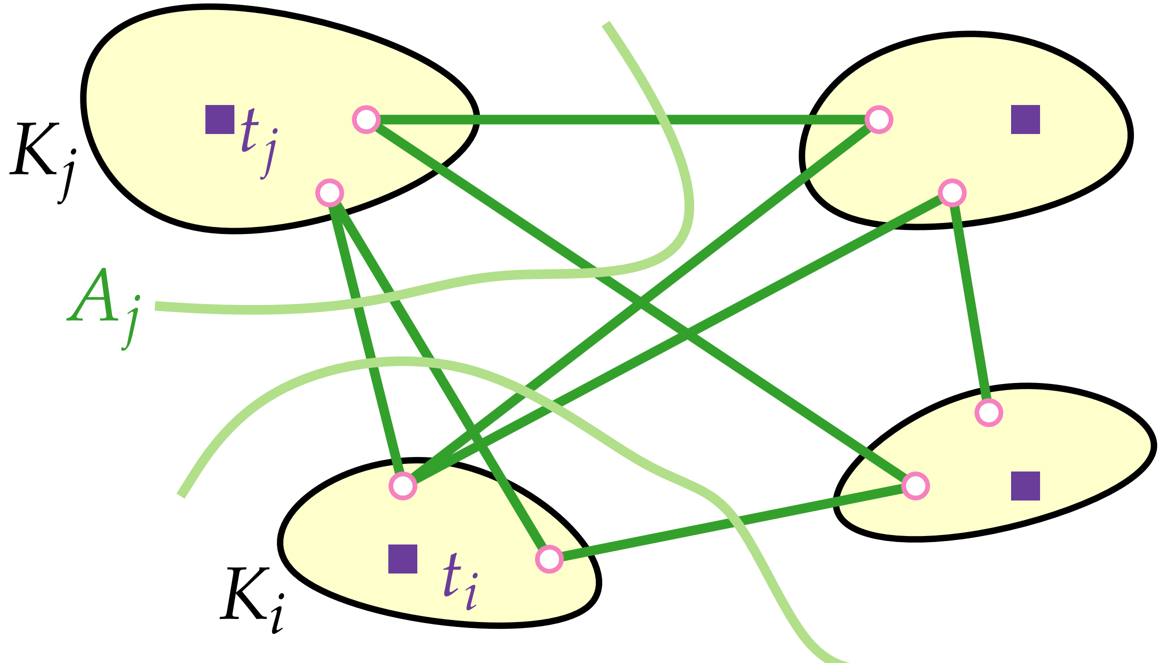
$A_i = \{uv \in A : u \in K_i, v \notin K_i\}$

Observation. $A = \bigcup_{i=1}^{k} A_i$

Approximation Factor

Theorem. This algorithm is a factor-($2 - 2/k$)-approximation algorithm for MultiwayCut.

Proof. Consider optimal multiway cut A:



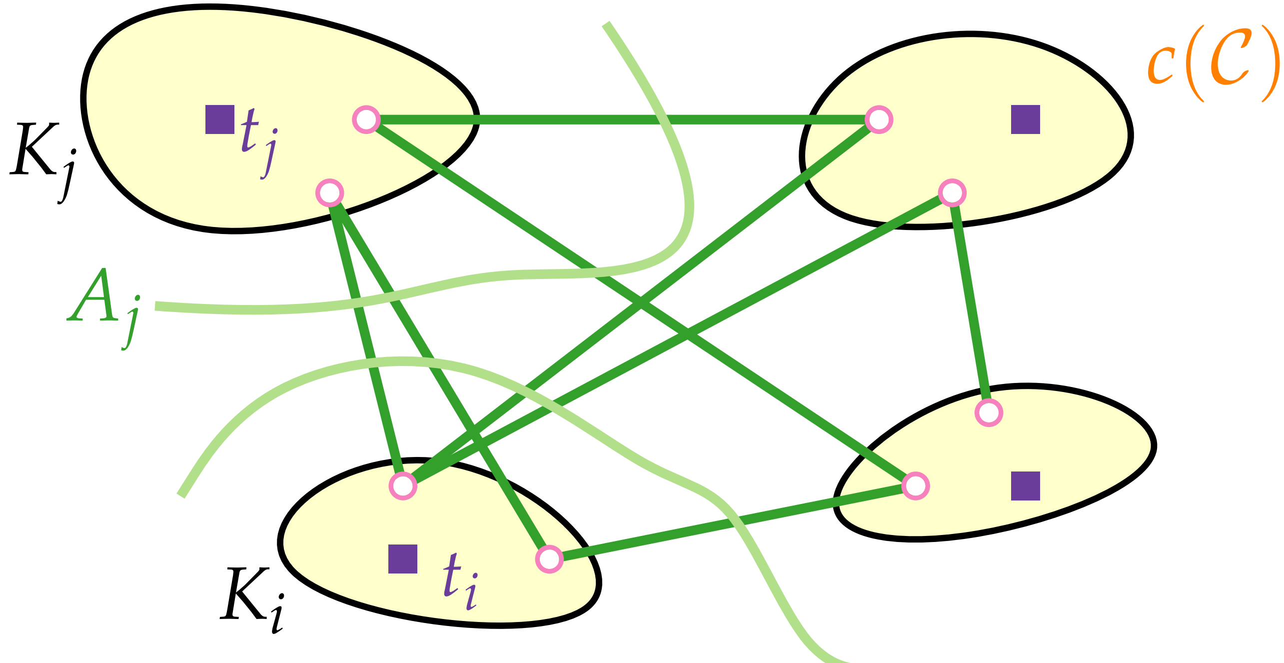
$A_i = \{uv \in A : u \in K_i, v \notin K_i\}$

Observation. $A = \bigcup_{i=1}^{k} A_i$ and $\sum_{i=1}^{k} c(A_i) \leq 2 \cdot c(A) = 2 \cdot \mathrm{OPT}$

Approximation Factor

Theorem. This algorithm is a factor-($2 - 2/k$)-approximation algorithm for MultiwayCut.

Proof. Consider optimal multiway cut A:



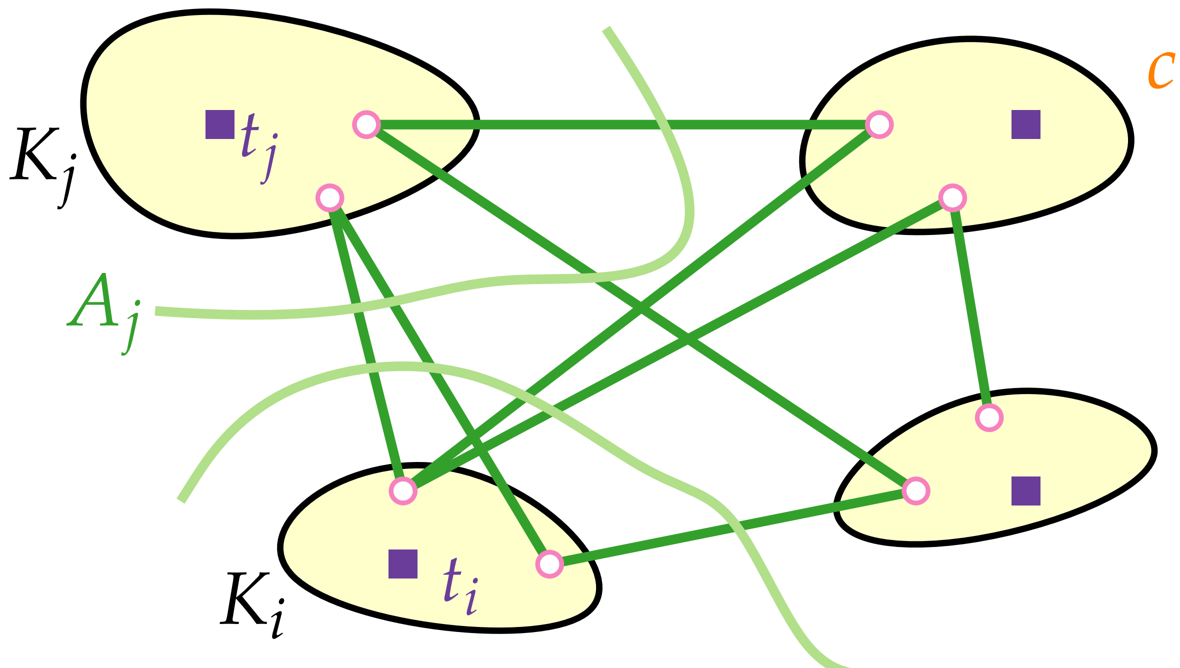
$A_i = \{uv \in A \colon u \in K_i, v \notin K_i\}$

Observation. $A = \bigcup_{i=1}^{k} A_i$ and $\sum_{i=1}^{k} c(A_i) \leq 2 \cdot c(A) = 2 \cdot \mathrm{OPT}$

Approximation Factor

Theorem. This algorithm is a factor-$(2 - 2/k)$-approximation algorithm for MultiwayCut.

Proof. Consider optimal multiway cut A:



$c(\mathcal{C}) \leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} c(C_i)$

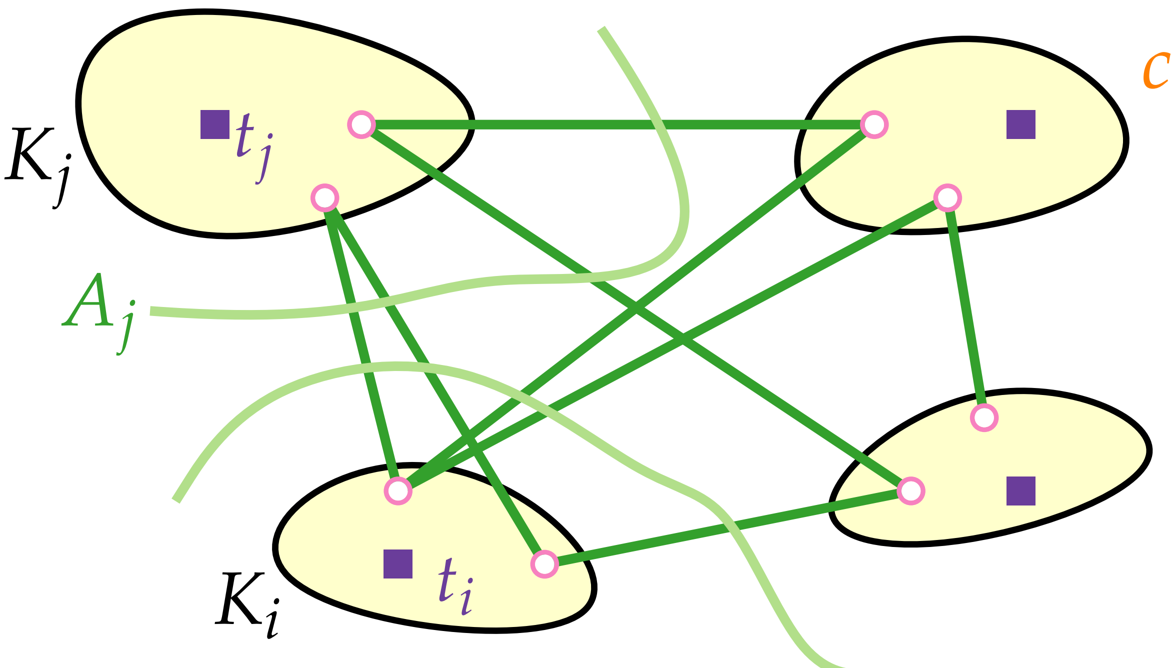
$A_i = \{uv \in A : u \in K_i, v \notin K_i\}$

Observation. $A = \bigcup_{i=1}^{k} A_i$ and $\sum_{i=1}^{k} c(A_i) \leq 2 \cdot c(A) = 2 \cdot \mathrm{OPT}$

Approximation Factor

Theorem. This algorithm is a factor-($2 - 2/k$)-approximation algorithm for MultiwayCut.

Proof. Consider optimal multiway cut A:



$$c(\mathcal{C}) \leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} c(C_i)$$

$$\leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} c(A_i)$$

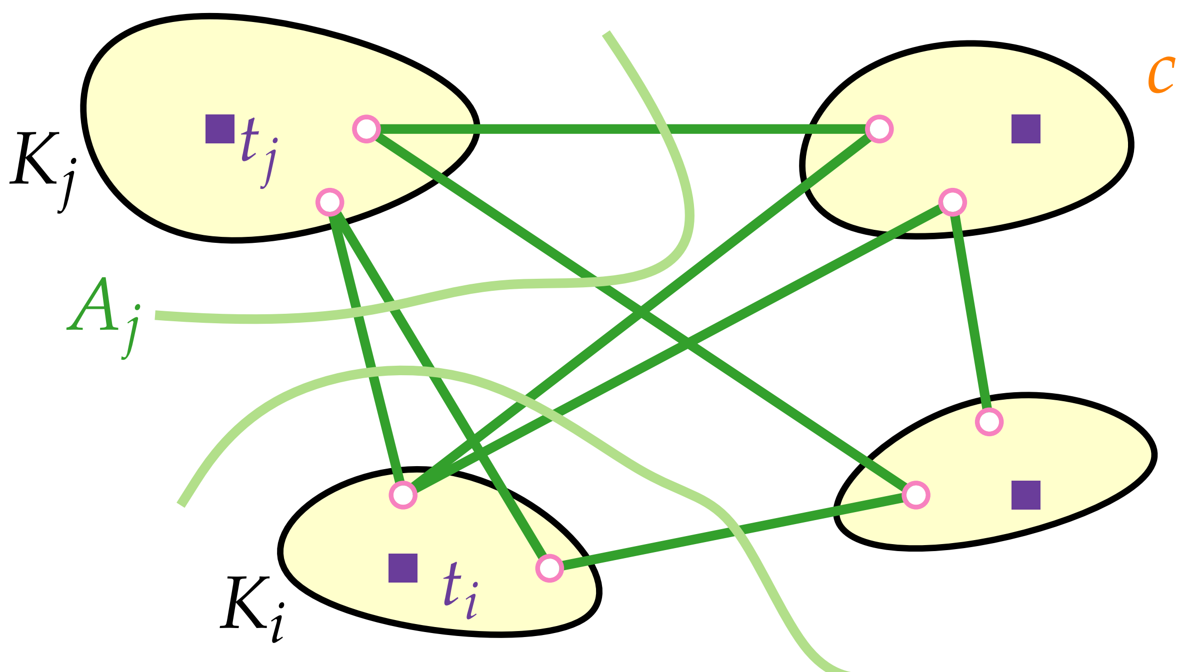
$$A_i = \{uv \in A \colon u \in K_i, v \notin K_i\}$$

Observation. $A = \bigcup_{i=1}^{k} A_i$ and $\sum_{i=1}^{k} c(A_i) \leq 2 \cdot c(A) = 2 \cdot \mathrm{OPT}$

Approximation Factor

Theorem. This algorithm is a factor-($2 - 2/k$)-approximation algorithm for MultiwayCut.

Proof. Consider optimal multiway cut A:



$$c(\mathcal{C}) \leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} c(C_i)$$
$$\leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} c(A_i)$$
$$\leq 2 \cdot \left(1 - \frac{1}{k}\right) c(A)$$

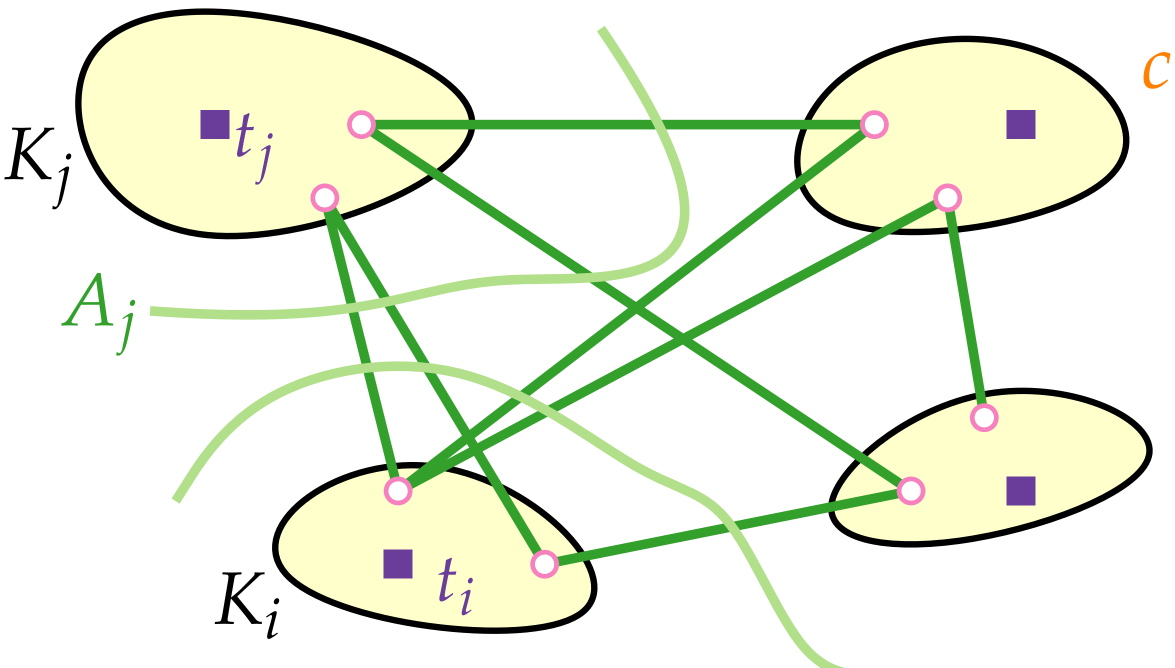
$$A_i = \{uv \in A : u \in K_i, v \notin K_i\}$$

Observation. $A = \bigcup_{i=1}^{k} A_i$ and $\sum_{i=1}^{k} c(A_i) \leq 2 \cdot c(A) = 2 \cdot \mathrm{OPT}$

Approximation Factor

Theorem. This algorithm is a factor-$(2 - 2/k)$-approximation algorithm for MultiwayCut.

Proof. Consider optimal multiway cut A:



$$c(\mathcal{C}) \leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} c(C_i)$$
$$\leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} c(A_i)$$
$$\leq 2 \cdot \left(1 - \frac{1}{k}\right) c(A)$$
$$\leq \left(2 - \frac{2}{k}\right) \mathrm{OPT}$$

$$A_i = \{uv \in A : u \in K_i, v \notin K_i\}$$

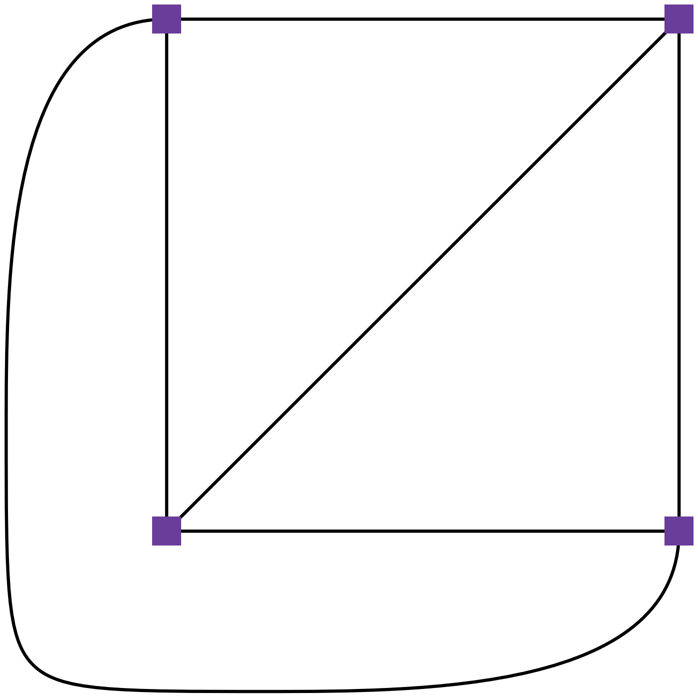
Observation. $A = \bigcup_{i=1}^{k} A_i$ and $\sum_{i=1}^{k} c(A_i) \leq 2 \cdot c(A) = 2 \cdot \mathrm{OPT}$

Analysis Sharp?

K_k

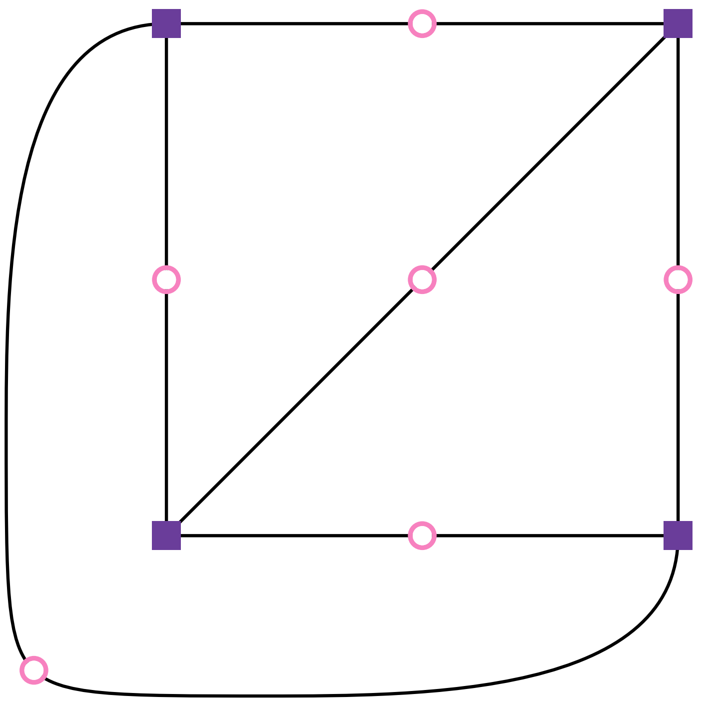
Analysis Sharp?

K_k



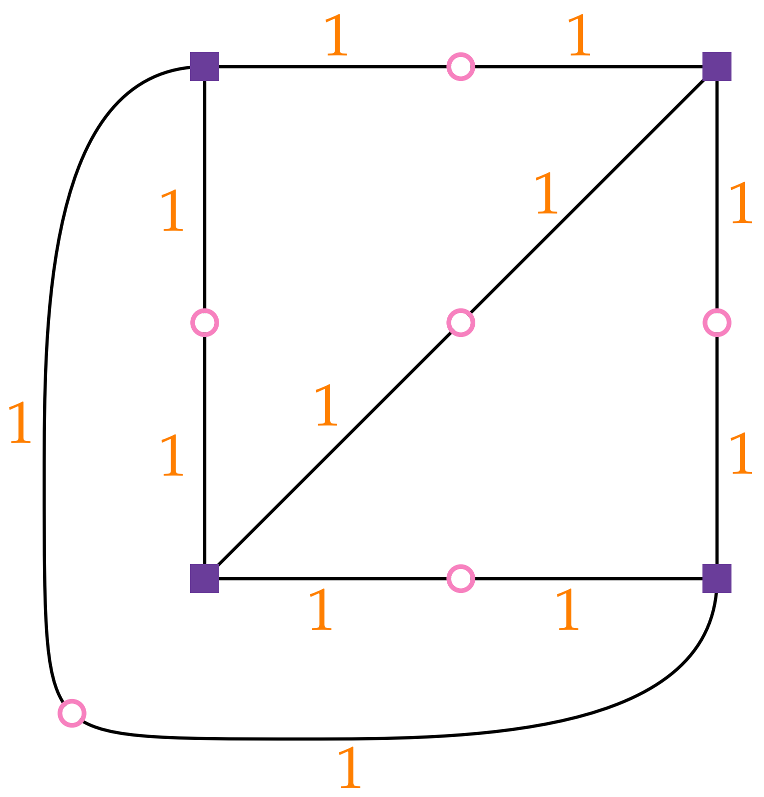
Analysis Sharp?

K_k



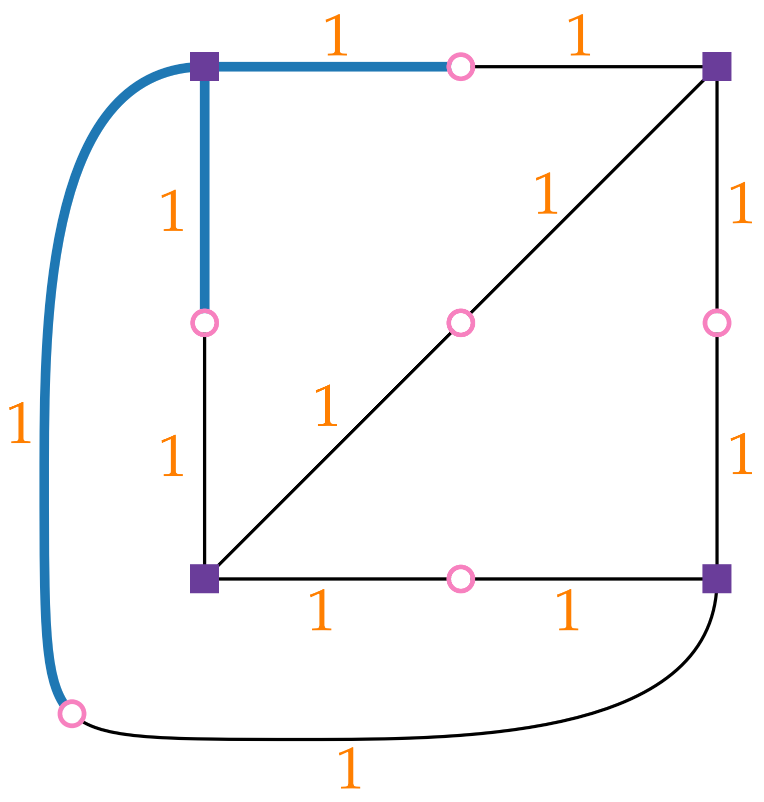
Analysis Sharp?

K_k

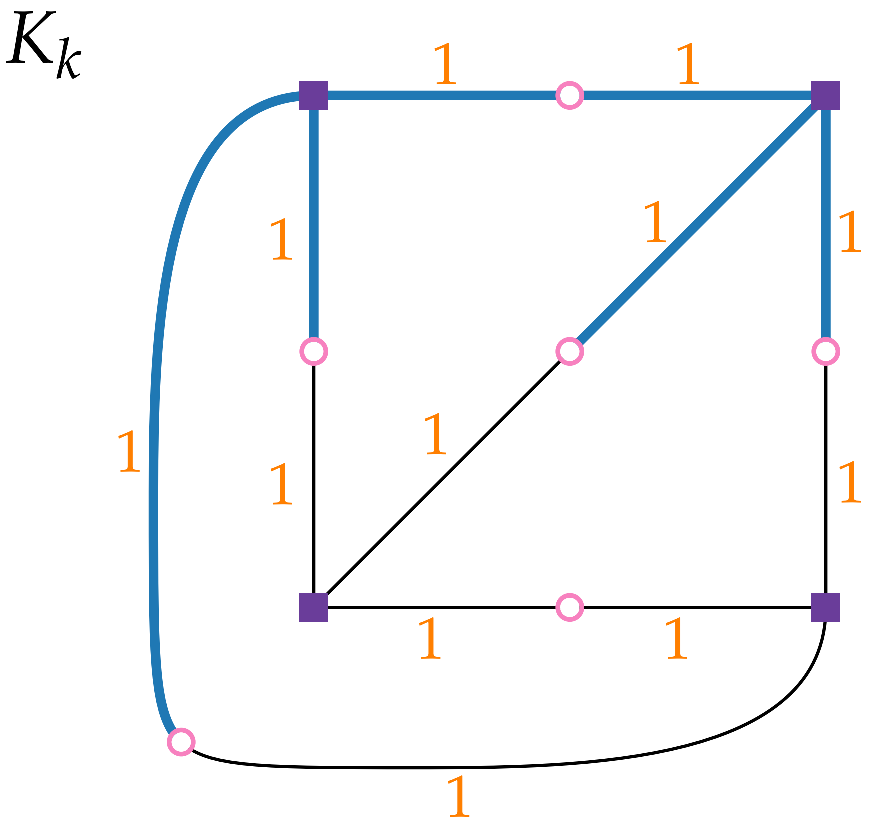


Analysis Sharp?

K_k

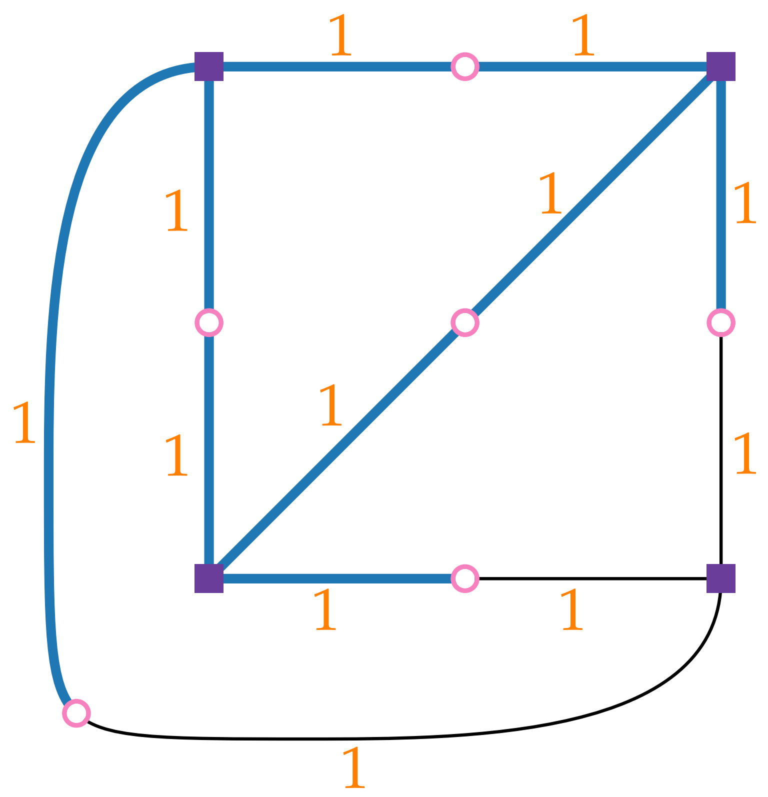


Analysis Sharp?

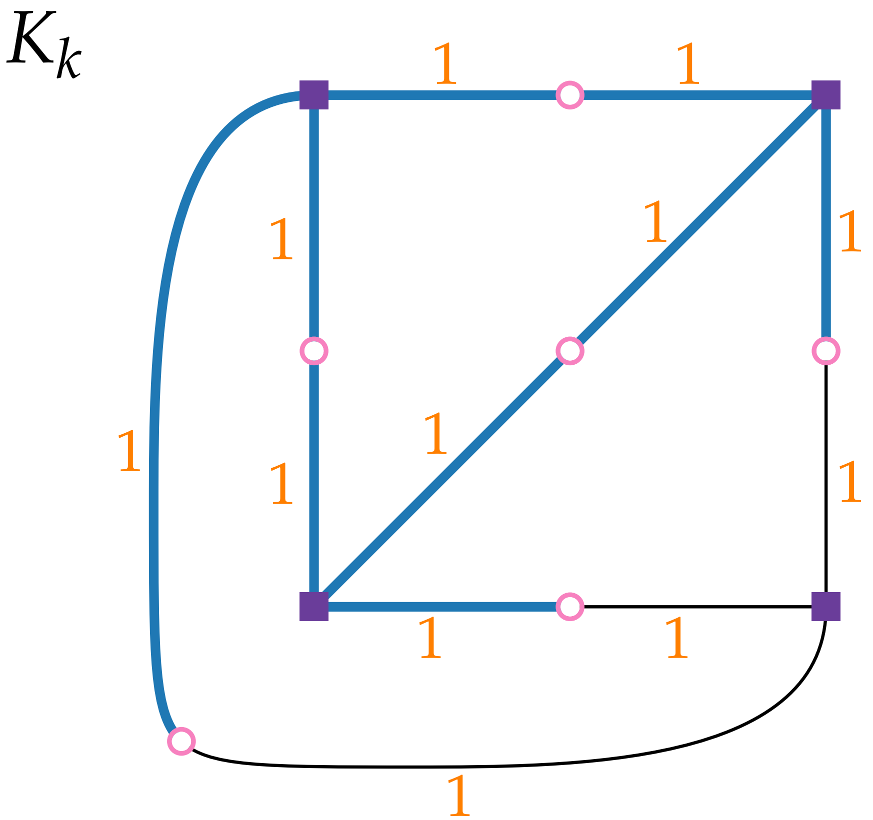


Analysis Sharp?

K_k

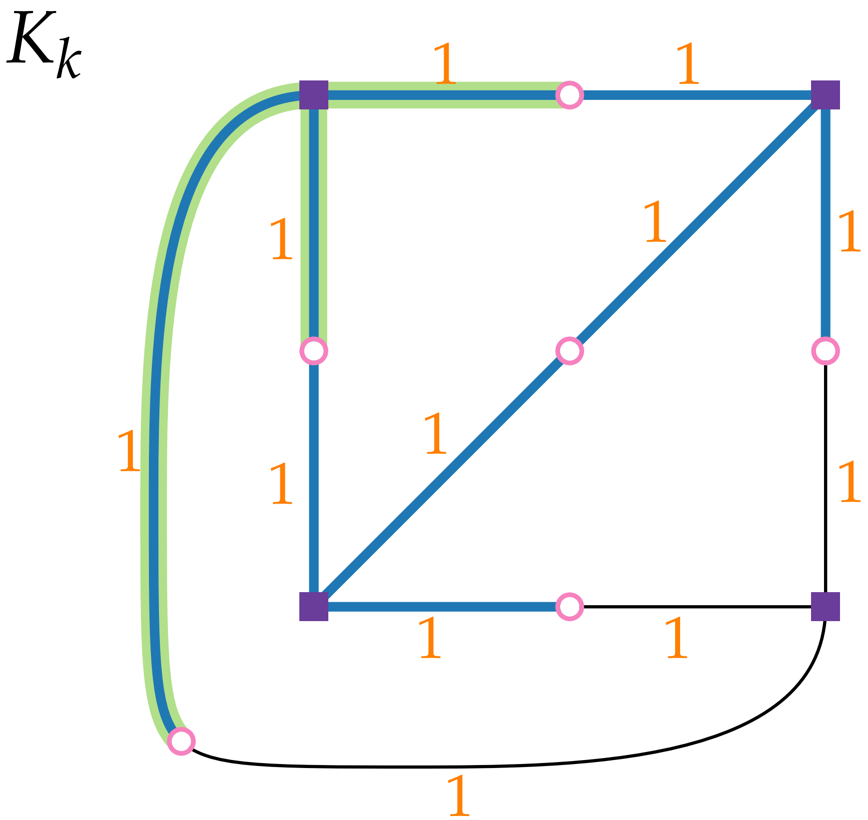


Analysis Sharp?



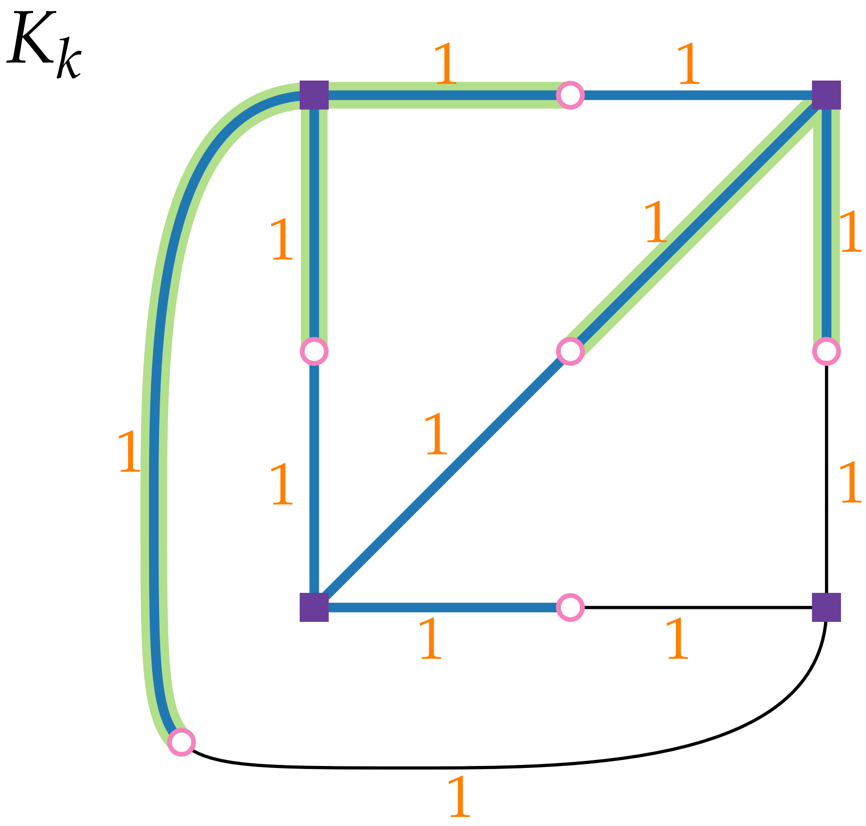
$\text{ALG} = (k-1)(k-1)$

Analysis Sharp?



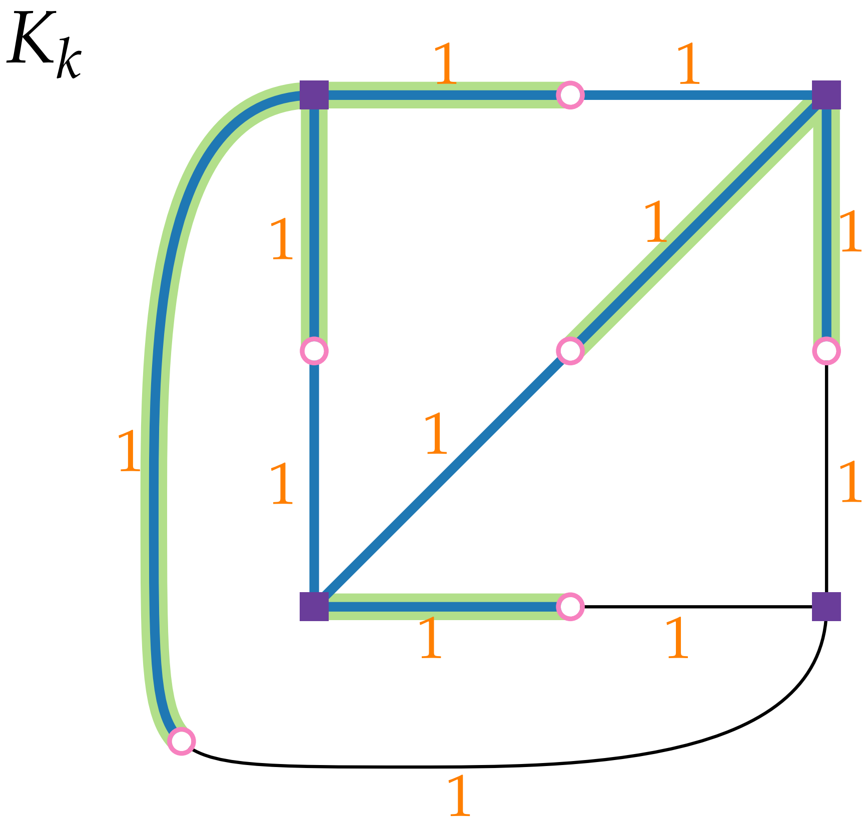
$\text{ALG} = (k-1)(k-1)$

Analysis Sharp?



$\mathrm{ALG} = (k-1)(k-1)$

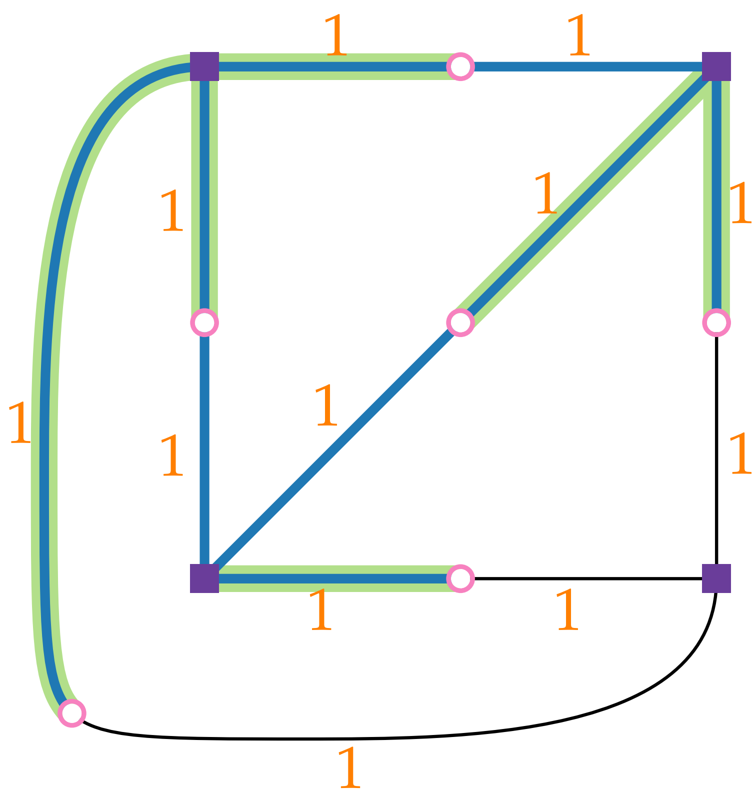
Analysis Sharp?



$$\text{ALG} = (k-1)(k-1)$$

Analysis Sharp?

K_k

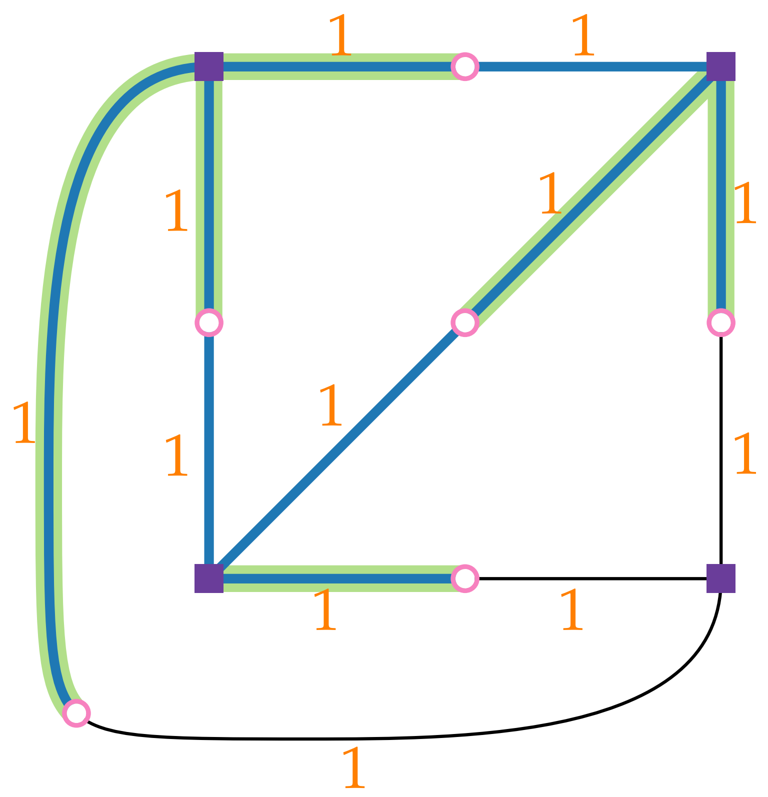


$\text{ALG} = (k-1)(k-1)$

$\text{OPT} = \sum_{i=1}^{k-1} i =$

Analysis Sharp?

K_k

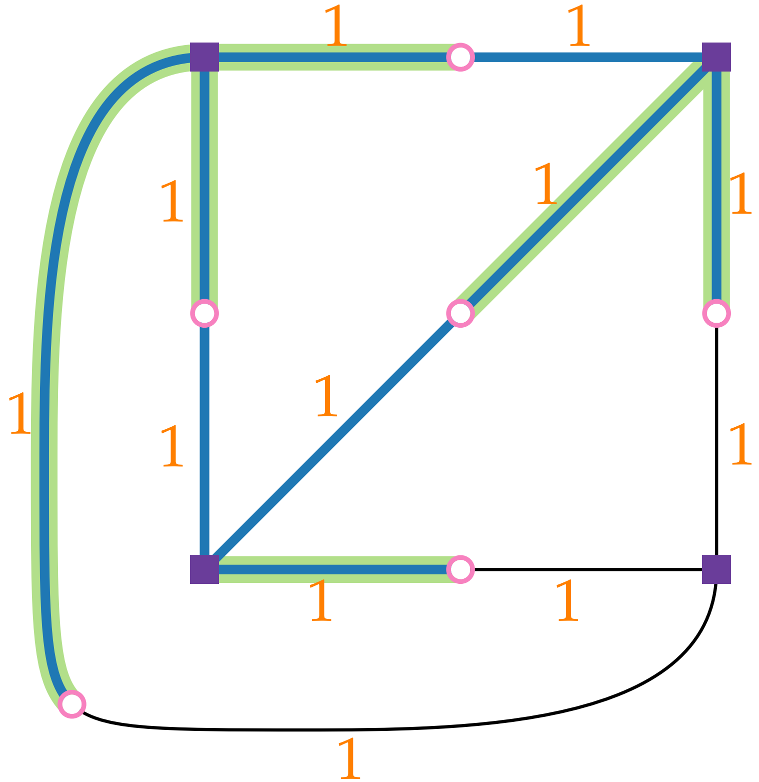


$\text{ALG} = (k-1)(k-1)$

$\text{OPT} = \sum_{i=1}^{k-1} i = \frac{k \cdot (k-1)}{2}$

Analysis Sharp?

K_k



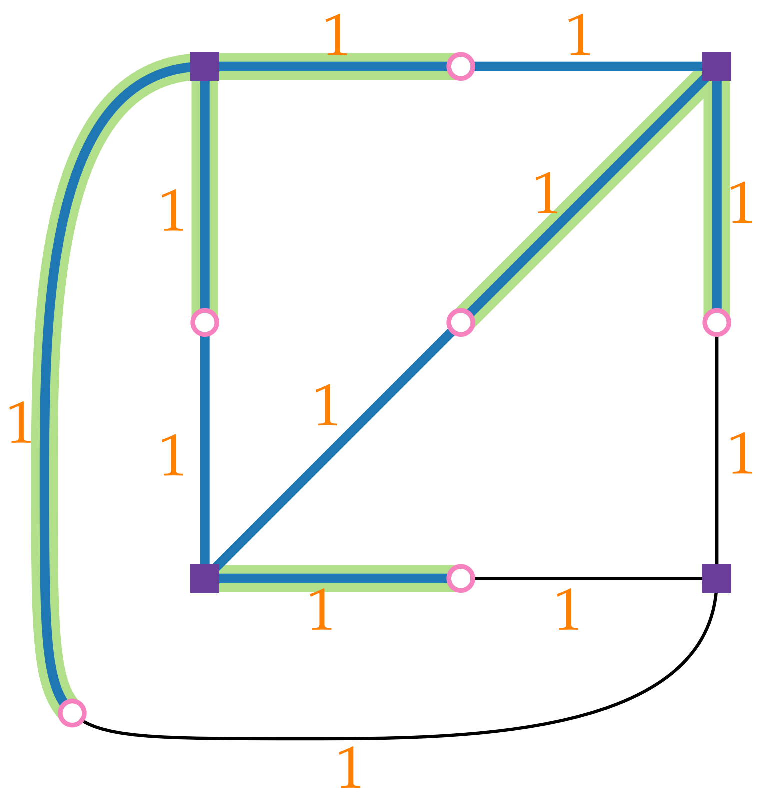
$\text{ALG} = (k-1)(k-1)$

$\text{OPT} = \sum_{i=1}^{k-1} i = \frac{k \cdot (k-1)}{2}$

$\text{ALG}/\text{OPT} =$

Analysis Sharp?

K_k

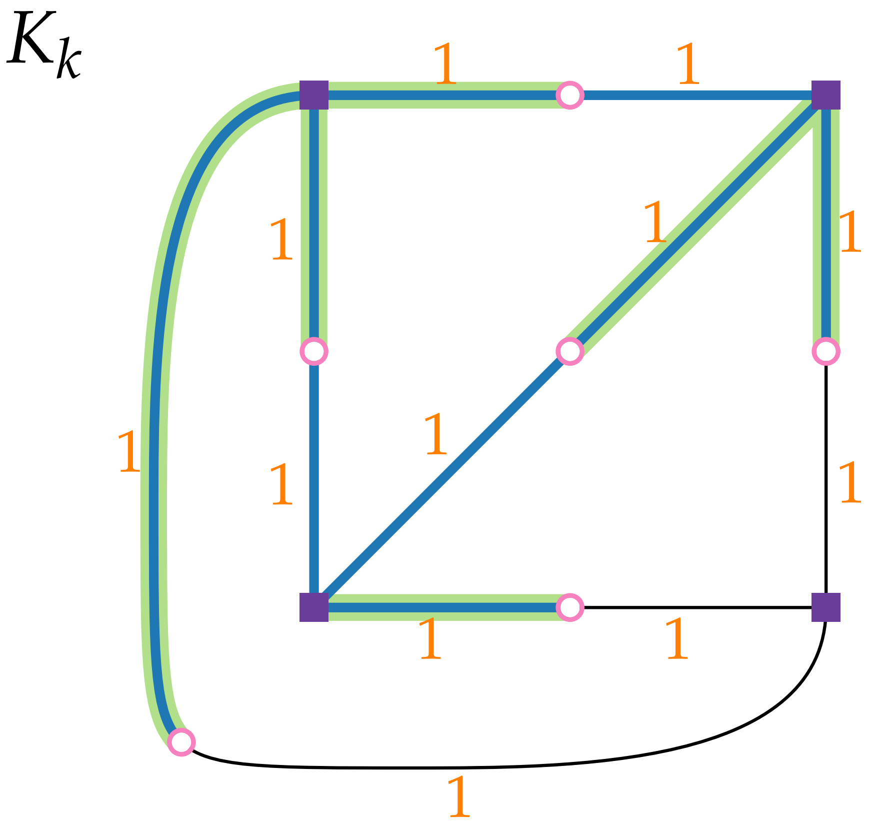


$\text{ALG} = (k-1)(k-1)$

$\text{OPT} = \sum_{i=1}^{k-1} i = \frac{k\cdot(k-1)}{2}$

$\text{ALG}/\text{OPT} = \frac{2k-2}{k} =$

Analysis Sharp?



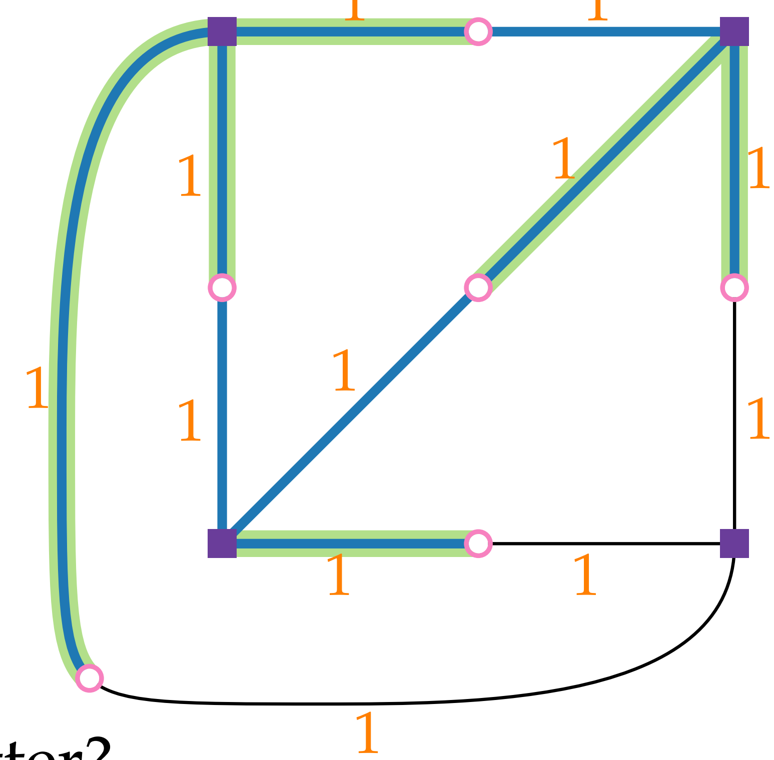
$\text{ALG} = (k-1)(k-1)$

$\text{OPT} = \sum_{i=1}^{k-1} i = \frac{k \cdot (k-1)}{2}$

$\text{ALG}/\text{OPT} = \frac{2k-2}{k} = 2 - \frac{2}{k}$

Analysis Sharp?

K_k



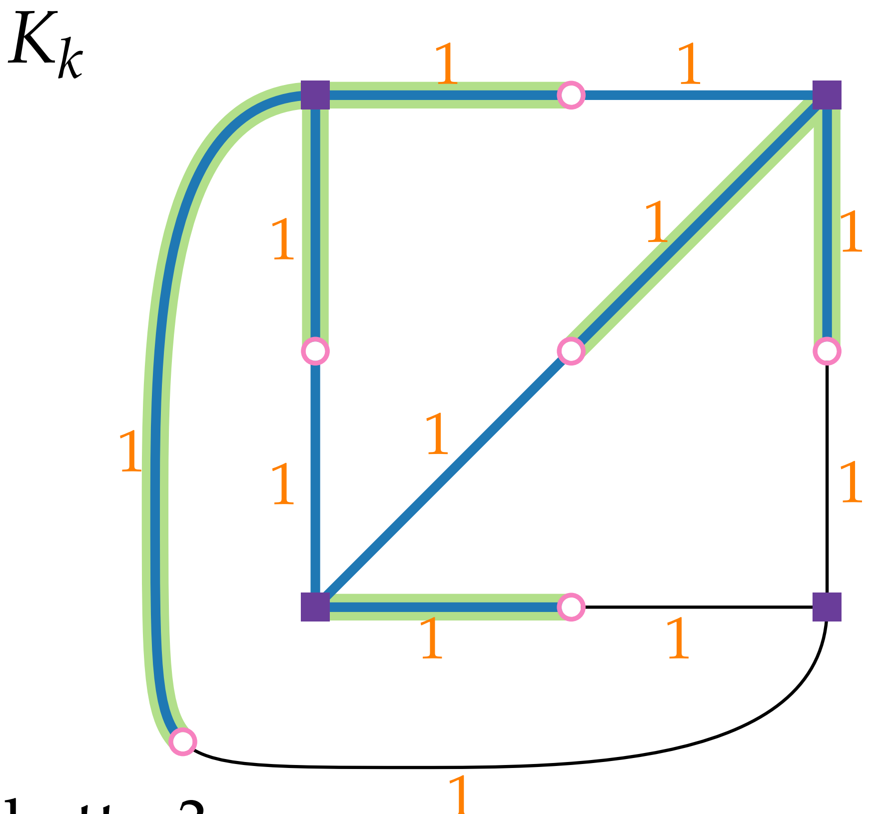
$\text{ALG} = (k-1)(k-1)$

$\text{OPT} = \sum_{i=1}^{k-1} i = \frac{k \cdot (k-1)}{2}$

$\text{ALG}/\text{OPT} = \frac{2k-2}{k} = 2 - \frac{2}{k}$

better?

Analysis Sharp?



$$\text{ALG} = (k-1)(k-1)$$

$$\text{OPT} = \sum_{i=1}^{k-1} i = \frac{k\cdot(k-1)}{2}$$

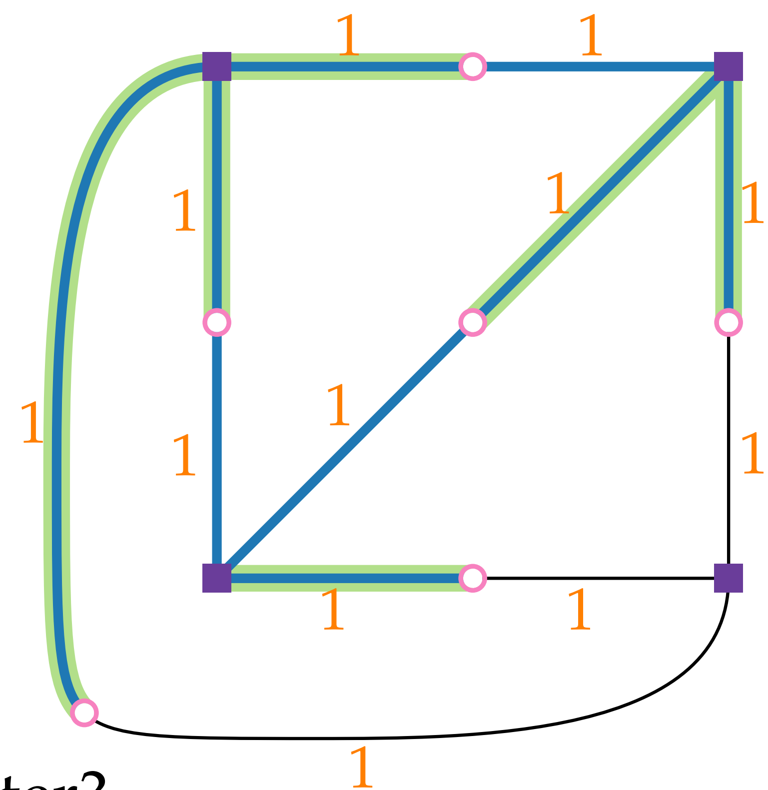
$$\text{ALG}/\text{OPT} = \frac{2k-2}{k} = 2 - \frac{2}{k}$$

better?

The best known approximation factor for MultiwayCut is $1.2965 - \frac{1}{k}$. [Sharma & Vondrák '14]

Analysis Sharp?

K_k



$\text{ALG} = (k-1)(k-1)$

$\text{OPT} = \sum_{i=1}^{k-1} i = \frac{k\cdot(k-1)}{2}$

$\text{ALG}/\text{OPT} = \frac{2k-2}{k} = 2 - \frac{2}{k}$

better?

The best known approximation factor for MultiwayCut is $1.2965 - \frac{1}{k}$. [Sharma & Vondrák '14]

MultiwayCut cannot be approximated within factor $1.20016 - O(1/k)$ (unless P=NP). [Bérczi, Chandrasekaran, Király & Madan '18]