Oblivious Routing and Minimum Bisection

Seminar: Approximation Algorithms

Markus Kaiser

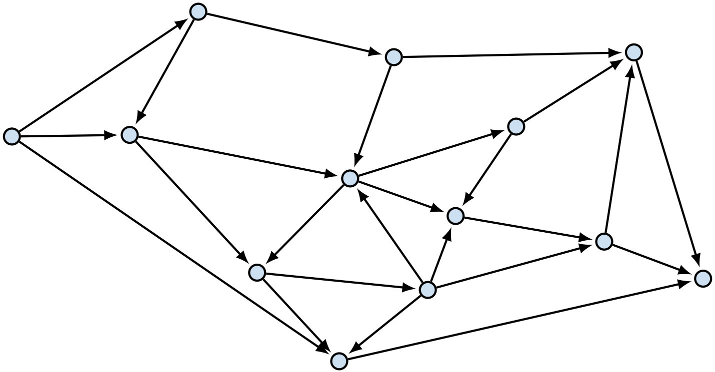
June 3, 2014

Problem (Single Commodity Flow)

Given

- *An (un)directed Graph* $G = (V, E)$
- *A capacity function* $c : E \to \mathbb{R}^+$
- *A source s and a target t*

Calculate a maximum possible flow $f : E \to \mathbb{R}^+$ *through G.*

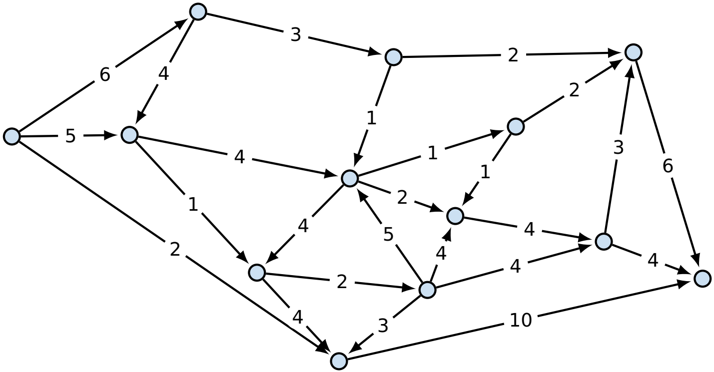


Problem (Single Commodity Flow)

Given

- *An (un)directed Graph $G = (V, E)$*
- *A capacity function $c : E \rightarrow \mathbb{R}^+$*
- *A source s and a target t*

Calculate a maximum possible flow $f : E \rightarrow \mathbb{R}^+$ through G.

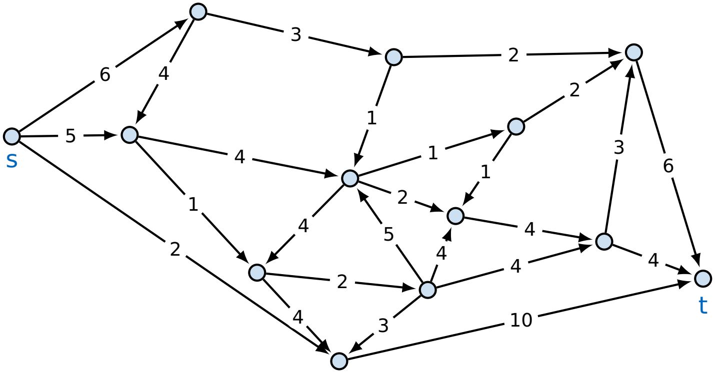


Problem (Single Commodity Flow)

Given

- *An (un)directed Graph $G = (V, E)$*
- *A capacity function $c : E \to \mathbb{R}^+$*
- *A source s and a target t*

Calculate a maximum possible flow $f : E \to \mathbb{R}^+$ through G.

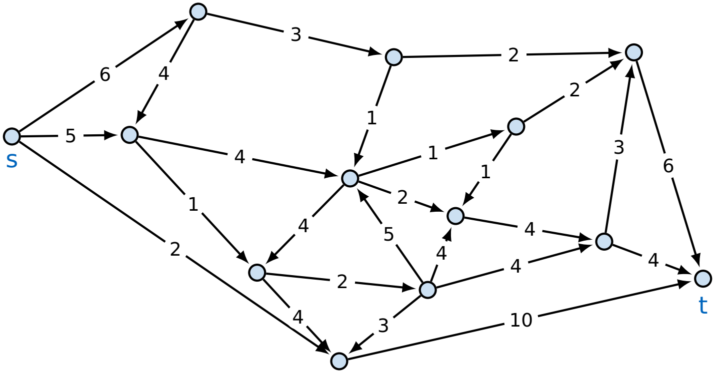


Problem (Single Commodity Flow)

Given

- *An (un)directed Graph $G = (V, E)$*
- *A capacity function $c : E \to \mathbb{R}^+$*
- *A source s and a target t*

Calculate a maximum possible flow $f : E \to \mathbb{R}^+$ through G.

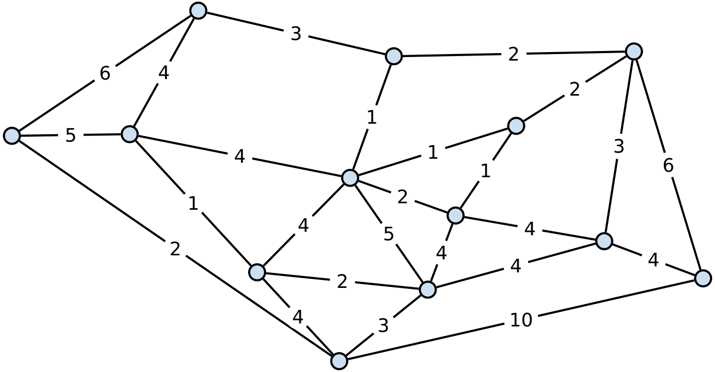


Problem (Multi Commodity Flow)

Given

- *An undirected Graph $G = (V, E)$*
- *A capacity function $c : E \to \mathbb{R}^+$*
- *A demand function $d : V^2 \to \mathbb{R}^+$*

Calculate a flow f with least congestion $\rho = \max_{e \in E} \frac{f_e}{c_e}$.

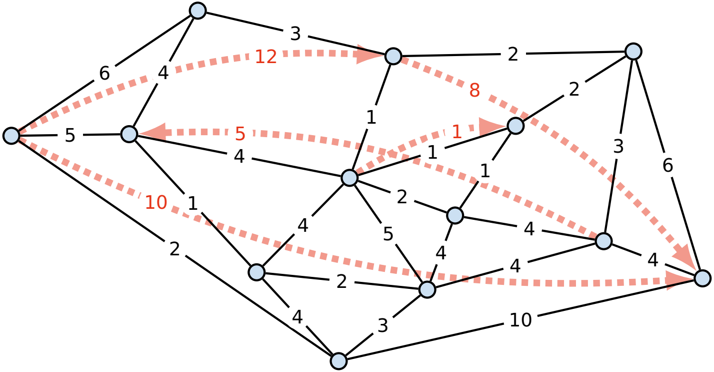


Problem (Multi Commodity Flow)

Given

- *An undirected Graph* $G = (V, E)$
- *A capacity function* $c : E \rightarrow \mathbb{R}^+$
- *A demand function* $d : V^2 \rightarrow \mathbb{R}^+$

Calculate a flow f *with least congestion* $\rho = \max_{e \in E} \frac{f_e}{c_e}$.

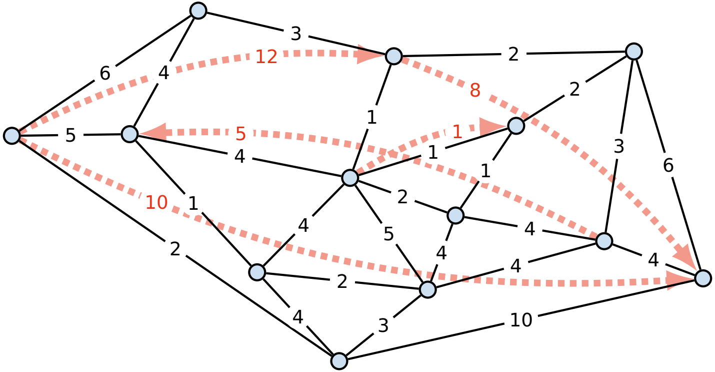


Problem (Multi Commodity Flow)

Given

- *An undirected Graph* $G = (V, E)$
- *A capacity function* $c : E \to \mathbb{R}^+$
- *A demand function* $d : V^2 \to \mathbb{R}^+$

Calculate a flow f with least congestion $\rho = \max_{e \in E} \frac{f_e}{c_e}$.

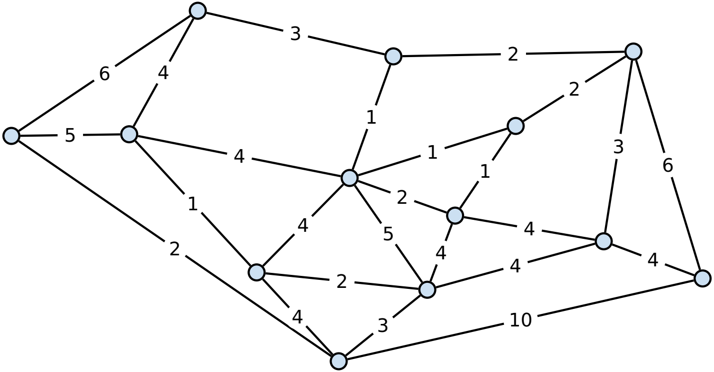


Problem (Oblivious Routing)

Given

- *An undirected Graph $G = (V, E)$*
- *A capacity function $c : E \to \mathbb{R}^+$*

Calculate a combination of paths for each $(u, v) \in V^2$ such that for any demand function the congestion will be as small as possible.

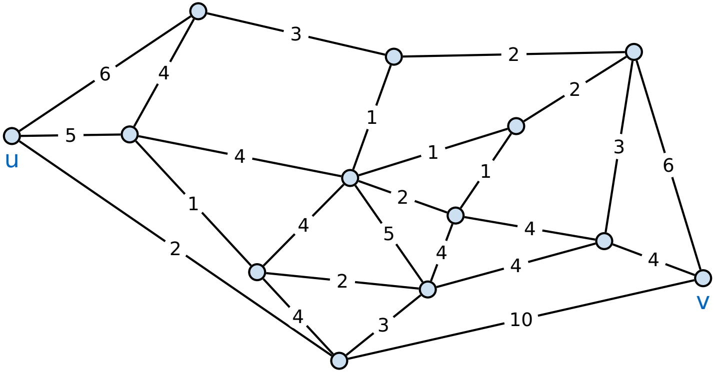


Problem (Oblivious Routing)

Given

- *An undirected Graph* $G = (V, E)$
- *A capacity function* $c : E \to \mathbb{R}^+$

Calculate a combination of paths for each $(u, v) \in V^2$ *such that for any demand function the congestion will be as small as possible.*

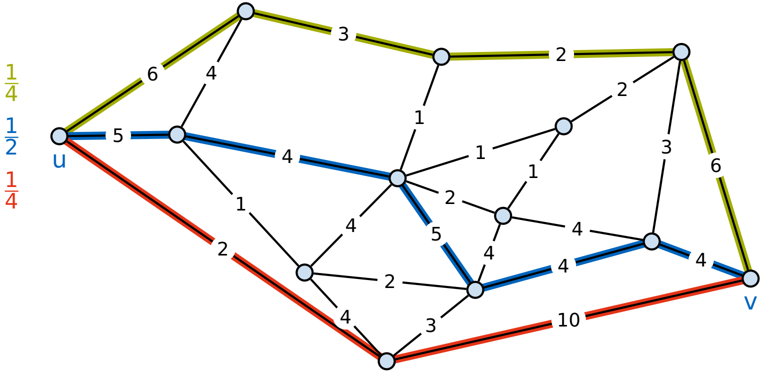


Problem (Oblivious Routing)

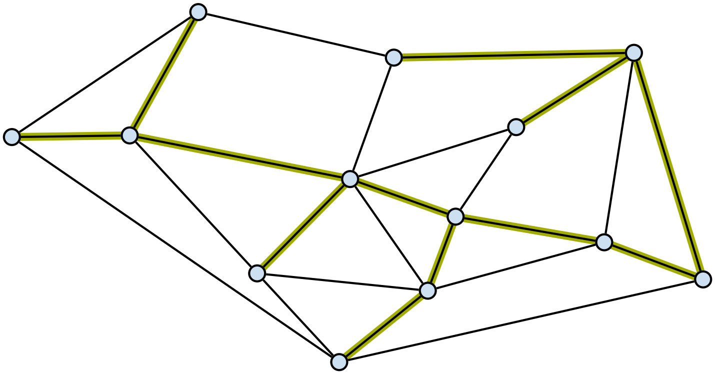
Given

- *An undirected Graph* $G = (V, E)$
- *A capacity function* $c : E \to \mathbb{R}^+$

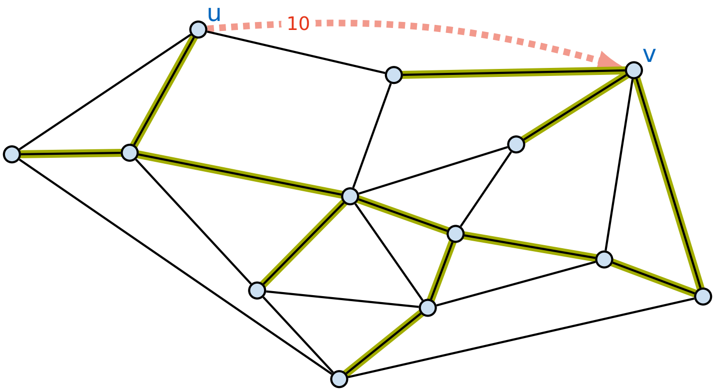
Calculate a combination of paths for each $(u, v) \in V^2$ *such that for any demand function the congestion will be as small as possible.*



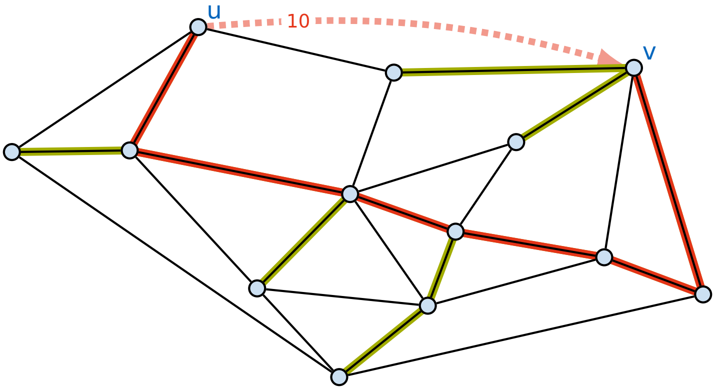
- Choose any spanning tree T of G
- Routing along its unique paths is a feasible solution



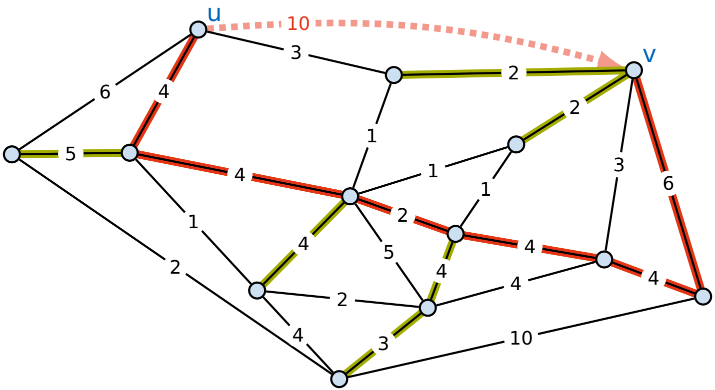
- Choose any spanning tree *T* of *G*
- Routing along its unique paths is a feasible solution



- Choose any spanning tree T of G
- Routing along its unique paths is a feasible solution



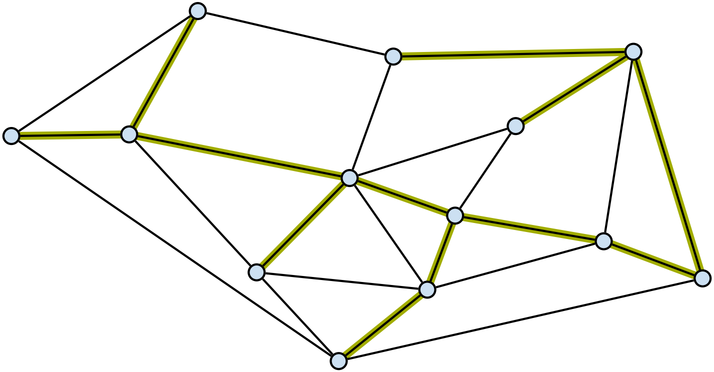
- Choose any spanning tree T of G
- Routing along its unique paths is a feasible solution



$$\rho = \max_{e \in E} \frac{f_e}{c_e} = \frac{10}{2} = 5$$

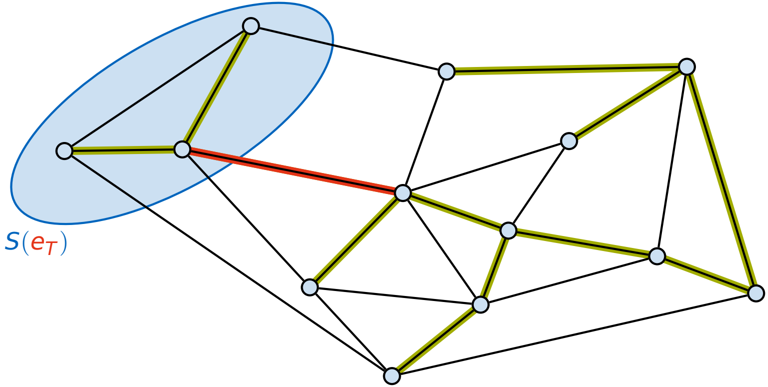
- Removing one edge e_T from a ST creates a node partition $S(e_T)$
- Every such partition has a capacity $C(e_T)$
- And a demand $D(e_T)$

$$C(e_T) = \sum_{\substack{u \in S(e_T),\\ v \notin S(e_T)}} c_{uv} \qquad D(e_T) = \sum_{\substack{u \in S(e_T),\\ v \notin S(e_T)}} c_{uv}$$



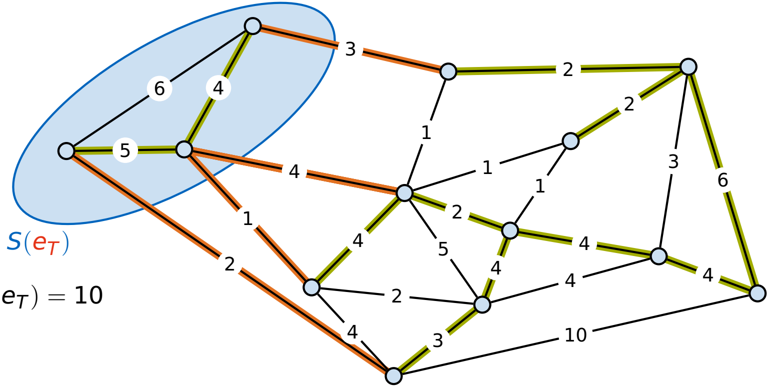
- Removing one edge e_T from a ST creates a node partition $S(e_T)$
- Every such partition has a capacity $C(e_T)$
- And a demand $D(e_T)$

$$C(e_T) = \sum_{\substack{u \in S(e_T), \\ v \notin S(e_T)}} c_{uv} \qquad D(e_T) = \sum_{\substack{u \in S(e_T), \\ v \notin S(e_T)}} c_{uv}$$



- Removing one edge e_T from a ST creates a node partition $S(e_T)$
- Every such partition has a capacity $C(e_T)$
- And a demand $D(e_T)$

$$C(e_T) = \sum_{\substack{u \in S(e_T), \\ v \notin S(e_T)}} c_{uv} \qquad D(e_T) = \sum_{\substack{u \in S(e_T), \\ v \notin S(e_T)}} c_{uv}$$

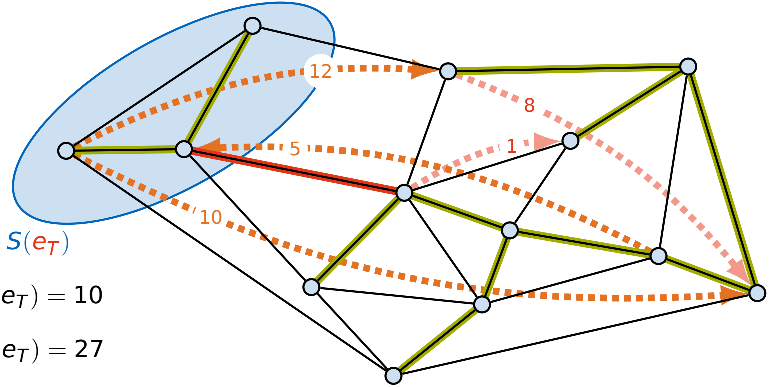


$C(e_T) = 10$

Tree Splits

- Removing one edge e_T from a ST creates a node partition $S(e_T)$
- Every such partition has a capacity $C(e_T)$
- And a demand $D(e_T)$

$$C(e_T) = \sum_{\substack{u \in S(e_T), \\ v \notin S(e_T)}} c_{uv} \qquad D(e_T) = \sum_{\substack{u \in S(e_T), \\ v \notin S(e_T)}} c_{uv}$$



$C(e_T) = 10$

$D(e_T) = 27$

Lemma

For any tree T and any tree edge e_T, we know that for any routing in G there must be an edge with congestion

$$\rho_e \geq \frac{D(e_T)}{C(e_T)}$$

And therefore the optimal solution ρ^* can be no better.

- Suppose we find a tree such that for some α

$$\forall e_T \in E_T. \quad c_{e_T} \geq \frac{1}{\alpha} C(e_T)$$

- Then we have

$$\rho_T = \max_{e_T} \frac{D(e_T)}{c_{e_T}} \leq \alpha \max_{e_T} \frac{D(e_T)}{C(e_T)} \leq \alpha\rho^*$$

Lemma

For any tree T and any tree edge e_T, we know that for any routing in G there must be an edge with congestion

$$\rho_e \geq \frac{D(e_T)}{C(e_T)}$$

And therefore the optimal solution ρ^* can be no better.

- Suppose we find a tree such that for some α

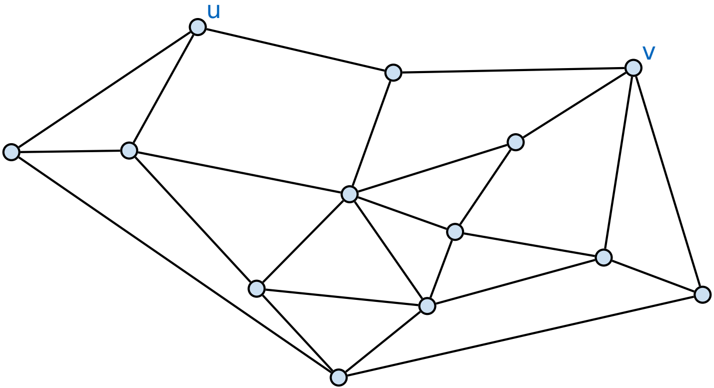
$$\forall e_T \in E_T. \quad c_{e_T} \geq \frac{1}{\alpha} C(e_T)$$

- Then we have

$$\rho_T = \max_{e_T} \frac{D(e_T)}{c_{e_T}} \leq \alpha \max_{e_T} \frac{D(e_T)}{C(e_T)} \leq \alpha \rho^*$$

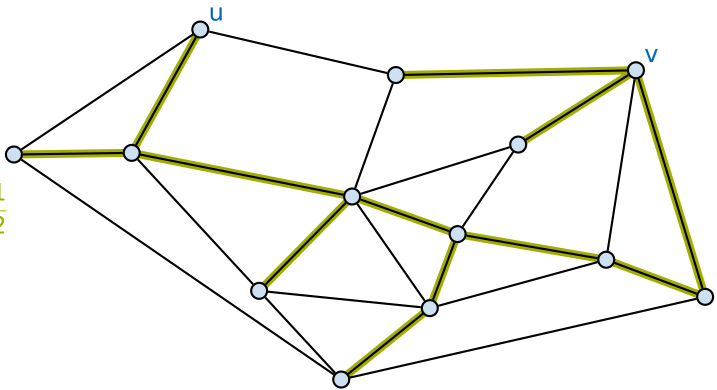
- Choose a set of spanning trees $\{T_i\}$ of G
- And a convex combination λ with $\sum_i \lambda_i = 1$, $\lambda \geq 0$
- Routing is now split according to this combination. For $e \in E$

$$f(e) = \sum_{\substack{i: \\ e \in T_i}} \lambda_i D_i(e)$$



- Choose a set of spanning trees $\{T_i\}$ of G
- And a convex combination λ with $\sum_i \lambda_i = 1$, $\lambda \geq 0$
- Routing is now split according to this combination. For $e \in E$

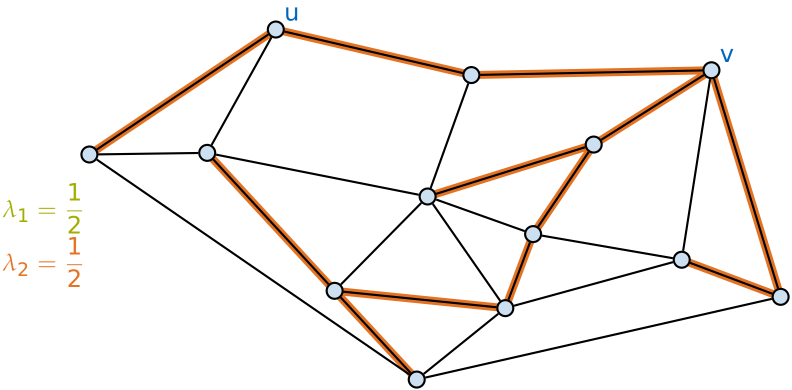
$$f(e) = \sum_{\substack{i:\\ e \in T_i}} \lambda_i D_i(e)$$



$\lambda_1 = \frac{1}{2}$

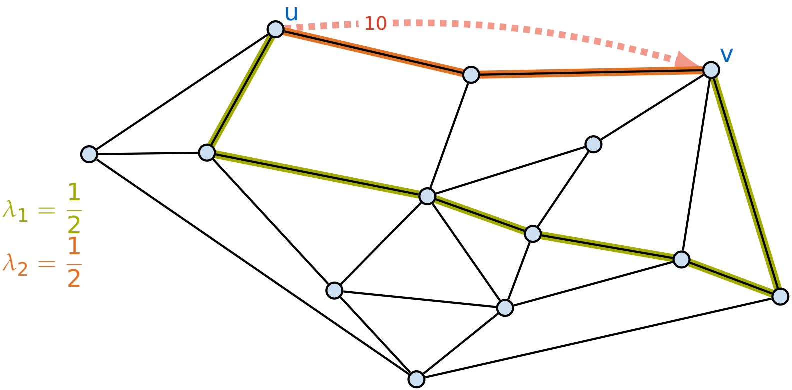
- Choose a set of spanning trees $\{T_i\}$ of G
- And a convex combination λ with $\sum_i \lambda_i = 1$, $\lambda \geq 0$
- Routing is now split according to this combination. For $e \in E$

$$f(e) = \sum_{\substack{i: \\ e \in T_i}} \lambda_i D_i(e)$$



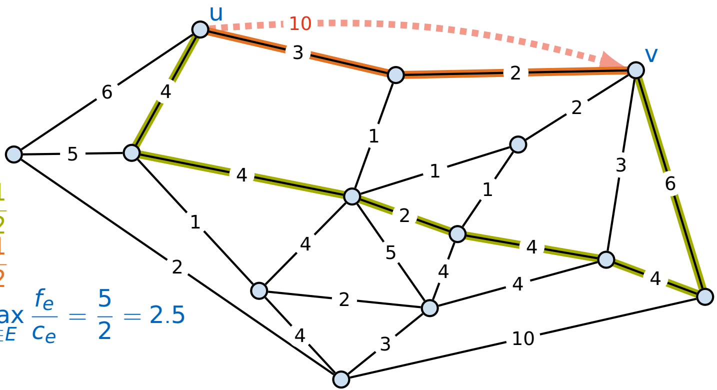
- Choose a set of spanning trees $\{T_i\}$ of G
- And a convex combination λ with $\sum_i \lambda_i = 1$, $\lambda \geq 0$
- Routing is now split according to this combination. For $e \in E$

$$f(e) = \sum_{\substack{i: \\ e \in T_i}} \lambda_i D_i(e)$$



- Choose a set of spanning trees $\{T_i\}$ of G
- And a convex combination λ with $\sum_i \lambda_i = 1$, $\lambda \geq 0$
- Routing is now split according to this combination. For $e \in E$

$$f(e) = \sum_{\substack{i: \\ e \in T_i}} \lambda_i D_i(e)$$



$\lambda_1 = \frac{1}{2}$

$\lambda_2 = \frac{1}{2}$

$\rho = \max_{e \in E} \frac{f_e}{c_e} = \frac{5}{2} = 2.5$

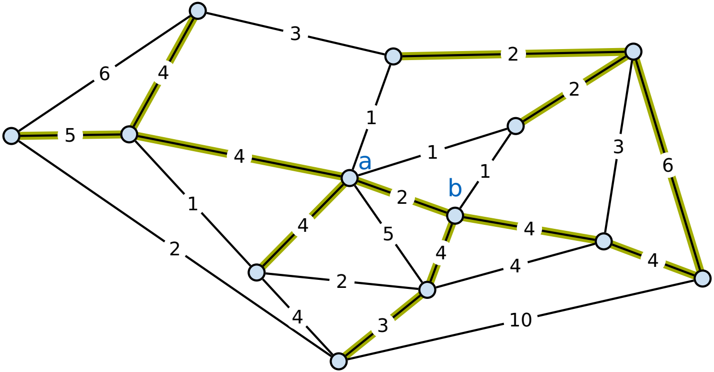
- Suppose we now find a set of trees such that for some α

$$\forall e \in E. \quad c_e \geq \frac{1}{\alpha} \sum_{\substack{i: \\ e \in T_i}} \lambda_i C_i(e)$$

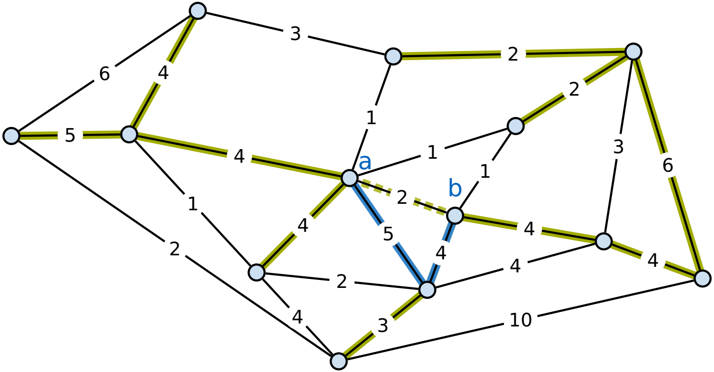
- Then we have

$$\begin{aligned} \rho = & \max_e \frac{f(e)}{c_e} \\ = & \max_e \frac{\sum_{\substack{i: \\ e \in T_i}} \lambda_i D_i(e)}{c_e} \\ \leq & \alpha \max_e \frac{\sum_{\substack{i: \\ e \in T_i}} \lambda_i D_i(e)}{\sum_{\substack{i: \\ e \in T_i}} \lambda_i C_i(e)} \\ \leq & \alpha \max_e \max_i \frac{D_i(e)}{C_i(e)} \leq \alpha \rho^* \end{aligned}$$

- Identify every edge in a tree with a path in G
- These paths can overlap
- For tree T we get a mapping $P_T : E_T \to E^+$

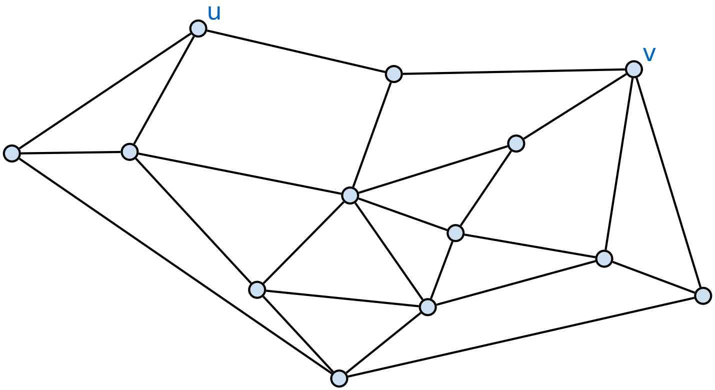


- Identify every edge in a tree with a path in G
- These paths can overlap
- For tree T we get a mapping $P_T : E_T \rightarrow E^+$



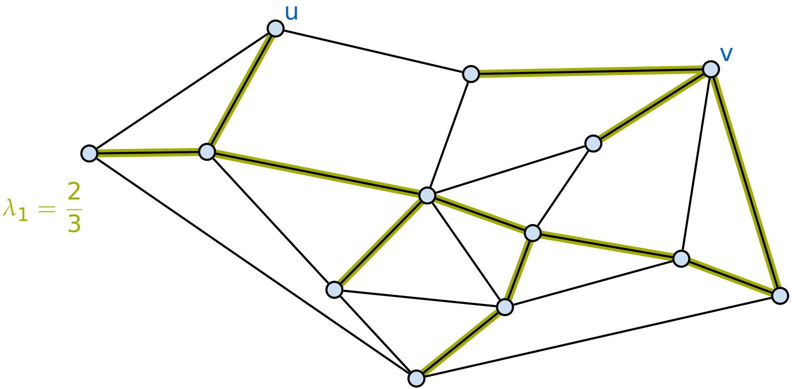
- Choose a set of pathtrees $\{T_i\}$ of G with combination λ
- Now route along the paths identified with edges. For $e \in E$

$$f(e) = \sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ e \in P_i(e_T)}} D_i(e_T)$$



- Choose a set of pathtrees $\{T_i\}$ of G with combination λ
- Now route along the paths identified with edges. For $e \in E$

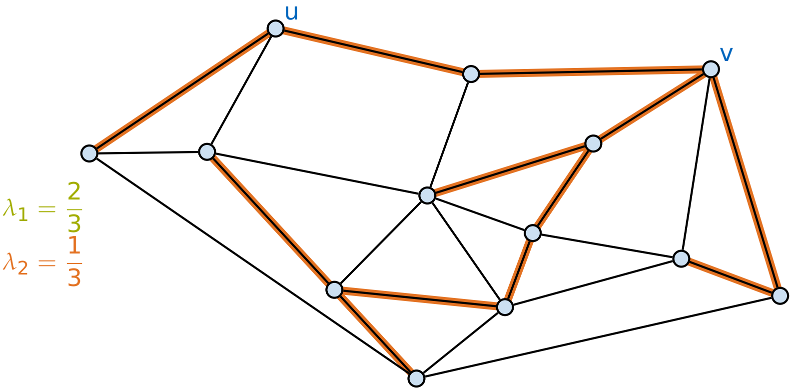
$$f(e) = \sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ e \in P_i(e_T)}} D_i(e_T)$$



Routing with multiple Pathtrees

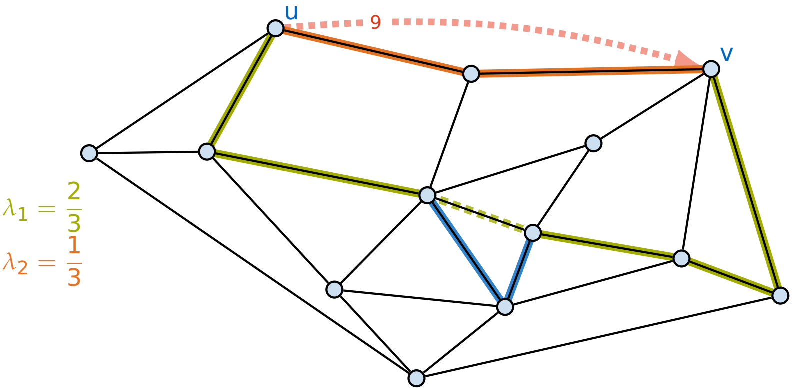
- Choose a set of pathtrees $\{T_i\}$ of G with combination λ
- Now route along the paths identified with edges. For $e \in E$

$$f(e) = \sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ e \in P_i(e_T)}} D_i(e_T)$$



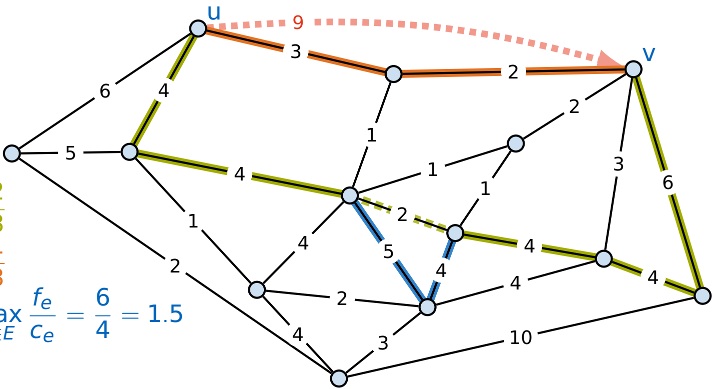
- Choose a set of pathtrees $\{T_i\}$ of G with combination λ
- Now route along the paths identified with edges. For $e \in E$

$$f(e) = \sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ e \in P_i(e_T)}} D_i(e_T)$$



- Choose a set of pathtrees $\{T_i\}$ of G with combination λ
- Now route along the paths identified with edges. For $e \in E$

$$f(e) = \sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ e \in P_i(e_T)}} D_i(e_T)$$



$\lambda_1 = \frac{2}{3}$

$\lambda_2 = \frac{1}{3}$

$\rho = \max_{e \in E} \frac{f_e}{c_e} = \frac{6}{4} = 1.5$

- Again suppose we now find a set of trees such that for some α

$$\forall e \in E. \quad c_e \geq \frac{1}{\alpha} \sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ e \in P_i(e_T)}} C_i(e_T)$$

- Then we have

$$\begin{aligned}
\rho = \quad & \max_e \frac{f(e)}{c_e} \\
\leq \alpha & \max_e \frac{\sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ e \in P_i(e_T)}} D_i(e_T)}{\sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ e \in P_i(e_T)}} C_i(e_T)} \\
\leq \alpha & \max_e \max_i \frac{D_i(e)}{C_i(e)} \leq \alpha \rho^*
\end{aligned}$$

- Again suppose we now find a set of trees such that for some α

$$\forall e \in E. \quad c_e \geq \frac{1}{\alpha} \sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ e \in P_i(e_T)}} C_i(e_T)$$

- Then we have

$$\begin{aligned} \rho = & \max_e \frac{f(e)}{c_e} \\ & \leq \alpha \max_e \frac{\sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ e \in P_i(e_T)}} D_i(e_T)}{\sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ e \in P_i(e_T)}} C_i(e_T)} \\ & \leq \alpha \max_e \max_i \frac{D_i(e)}{C_i(e)} \leq \alpha \rho^* \end{aligned}$$

How do we find such a set of trees? How large is α?

Primal Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.
We want to find the best trees with smallest α.

$$\begin{aligned}
\min_{\alpha,\lambda} \quad & \alpha \\
\text{s.t.} \quad & \sum_{i\in\mathcal{I}} \lambda_i \sum_{\substack{e_T\in T_i:\\ (u,v)\in P_i(e_T)}} C_i(e_T) \le \alpha c_{uv} \qquad \forall u,v\in V \\
& \sum_{i\in\mathcal{I}} \lambda_i = 1 \\
& \lambda \ge 0
\end{aligned}$$

We want to show that $\alpha \in \mathcal{O}(\log n)$

Dual Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.

$$\begin{aligned}
\max_{z,\mathcal{L}} \quad & z \\
\text{s.t.} \quad & \sum_{u,v \in V} c_{uv} \ell_{uv} = 1 \\
& z \leq \sum_{e_T \in T_i} C_i(e_T) \sum_{(u,v) \in P_i(e_T)} \ell_{uv} \qquad \forall i \in \mathcal{I} \\
& \mathcal{L} \geq 0
\end{aligned}$$

If $z \in \mathcal{O}(\log n)$ then $\alpha \in \mathcal{O}(\log n)$ by strong duality

Dual Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.

$$
\begin{aligned}
\max_{z,\mathcal{L}} \quad & z \\
\text{s.t.} \quad & \sum_{u,v\in V} c_{uv}\ell_{uv} = 1 \\
& z \le \sum_{e_T\in T_i} C_i(e_T) \sum_{(u,v)\in P_i(e_T)} \ell_{uv} \qquad \forall i \in \mathcal{I} \\
& \mathcal{L} \ge 0
\end{aligned}
$$

- We interpret the ℓ_{uv} as edge lengths in G
- They define a shortest path metric $d_\ell(u,v)$
- For an edge $e = (x,y)$ we write $d_\ell(e) := d_\ell(x,y)$

Dual Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.

$$
\begin{aligned}
\max_{z,\mathcal{L}} \quad & z \\
\text{s.t.} \quad & \sum_{u,v \in V} c_{uv} \ell_{uv} = 1 \\
& z \leq \sum_{e_T \in T_i} C_i(e_T) \underbrace{\sum_{(u,v) \in P_i(e_T)} \ell_{uv}}_{\geq d_\ell(e_T)} \qquad \forall i \in \mathcal{I} \\
& \mathcal{L} \geq 0
\end{aligned}
$$

- We interpret the ℓ_{uv} as edge lengths in G
- They define a shortest path metric $d_\ell(u, v)$
- For an edge $e = (x, y)$ we write $d_\ell(e) := d_\ell(x, y)$

Dual Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.

$$
\begin{aligned}
\max_{z,\mathcal{L}} \quad & z \\
\text{s.t.} \quad & \sum_{u,v\in V} c_{uv}\ell_{uv} = 1 \\
& z \le \sum_{e_T\in T_i} C_i(e_T) d_\ell(e_T) \qquad \forall i \in \mathcal{I} \\
& \mathcal{L} \ge 0
\end{aligned}
$$

- We interpret the ℓ_{uv} as edge lengths in G
- They define a shortest path metric $d_\ell(u,v)$
- For an edge $e = (x,y)$ we write $d_\ell(e) := d_\ell(x,y)$

Dual Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.

$$\begin{aligned} \max_{z,\mathcal{L}} \quad & z \\ \text{s.t.} \quad & \sum_{u,v \in V} c_{uv}\ell_{uv} = 1 \\ & z \leq \sum_{e_T \in T_i} C_i(e_T) d_\ell(e_T) \qquad \forall i \in \mathcal{I} \\ & \mathcal{L} \geq 0 \end{aligned}$$

$\geq \min_{i \in \mathcal{I}} \cdots$

- We interpret the ℓ_{uv} as edge lengths in G
- They define a shortest path metric $d_\ell(u, v)$
- For an edge $e = (x, y)$ we write $d_\ell(e) := d_\ell(x, y)$

Dual Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.

$$
\begin{aligned}
\max_{z,\mathcal{L}} \quad & z \\
\text{s.t.} \quad & \sum_{u,v\in V} c_{uv}\ell_{uv} = 1 \\
& z \le \min_{i\in\mathcal{I}} \sum_{e_T\in T_i} C_i(e_T)\,d_\ell(e_T) \\
& \mathcal{L} \ge 0
\end{aligned}
$$

- We interpret the ℓ_{uv} as edge lengths in G
- They define a shortest path metric $d_\ell(u, v)$
- For an edge $e = (x, y)$ we write $d_\ell(e) := d_\ell(x, y)$

Dual Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.

$$\begin{aligned}\max_{\mathcal{L}} \quad & \min_{i \in \mathcal{I}} \sum_{e_T \in T_i} C_i(e_T) d_\ell(e_T) \\ \text{s.t.} \quad & \sum_{u,v \in V} c_{uv} \ell_{uv} = 1 \\ & \mathcal{L} \geq 0\end{aligned}$$

- Now suppose

$$\sum_{u,v \in V} c_{uv} \ell_{uv} = \beta > 0$$

- If we scale every length by $\frac{1}{\beta}$ our solution will change by $\frac{1}{\beta}$

Dual Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.

$$\max_{\mathcal{L}} \quad \min_{i \in \mathcal{I}} \sum_{e_T \in T_i} C_i(e_T) d_\ell(e_T)$$

$$\text{s.t.} \quad \sum_{u,v \in V} c_{uv} \ell_{uv} = 1$$

$$\mathcal{L} \geq 0$$

- Now suppose

$$\sum_{u,v \in V} c_{uv} \ell_{uv} = \beta > 0$$

- If we scale every length by $\frac{1}{\beta}$ our solution will change by $\frac{1}{\beta}$

Dual Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.

$$\begin{array}{ll} \max\limits_{\mathcal{L}} & \min\limits_{i \in \mathcal{I}} \dfrac{\sum_{e_T \in T_i} C_i(e_T) d_\ell(e_T)}{\sum_{u,v \in V} c_{uv} \ell_{uv}} \\ \text{s.t.} & \mathcal{L} \geq 0 \end{array}$$

- Now suppose

$$\sum_{u,v \in V} c_{uv} \ell_{uv} = \beta > 0$$

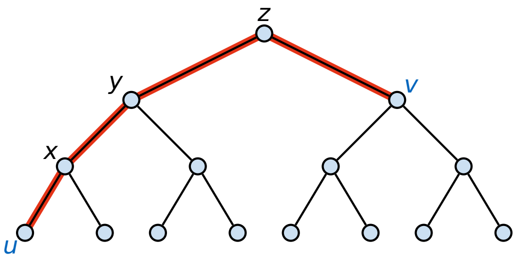
- If we scale every length by $\frac{1}{\beta}$ our solution will change by $\frac{1}{\beta}$

Theorem (Tree Metric)

For our metric d_ℓ there exists a tree metric (V, M) with

$$d_\ell(u,v) \le M_{uv} \qquad \forall u,v \in V$$

$$\sum_{u,v\in V} c_{uv} M_{uv} \le \mathcal{O}(\log n) \sum_{u,v\in V} c_{uv} d_\ell(u,v)$$

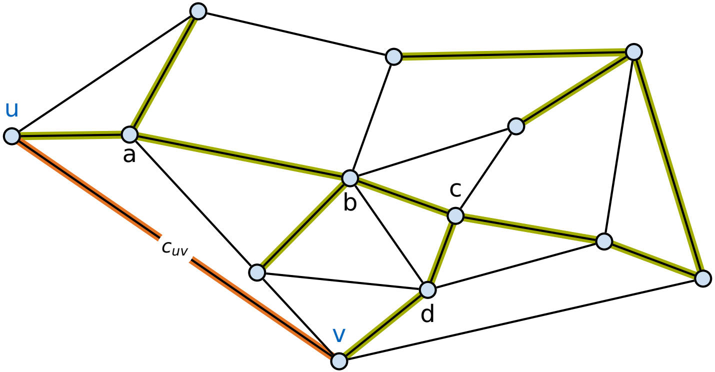


$$M_{uv} = M_{ux} + M_{xy} + M_{yz} + M_{zv}$$

Lemma

Let T be a spanning tree and (V, M) a tree metric of $G = (V, E)$. Then

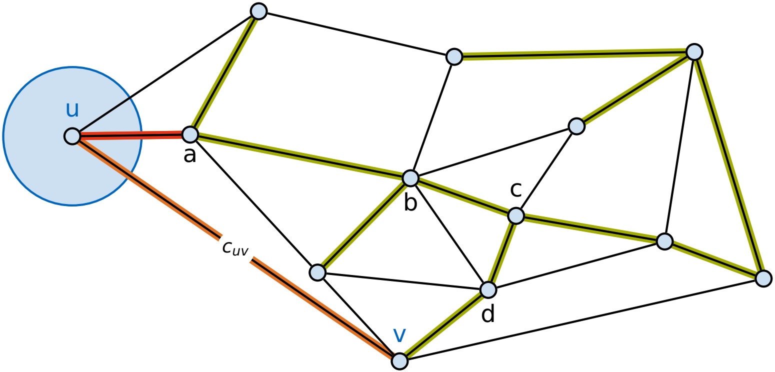
$$\sum_{(x,y)\in E_T} C(x,y) M_{xy} = \sum_{(u,v)\in E} c_{uv} M_{uv}$$



Lemma

Let T be a spanning tree and (V, M) a tree metric of $G = (V, E)$. Then

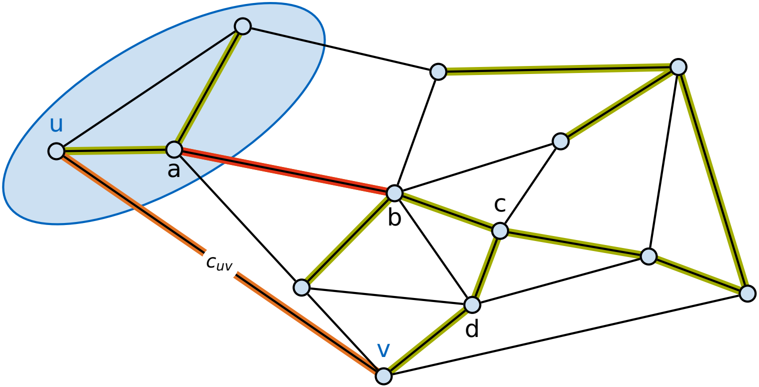
$$\sum_{(x,y)\in E_T} C(x,y) M_{xy} = \sum_{(u,v)\in E} c_{uv} M_{uv}$$



Lemma

Let T be a spanning tree and (V, M) a tree metric of $G = (V, E)$. Then

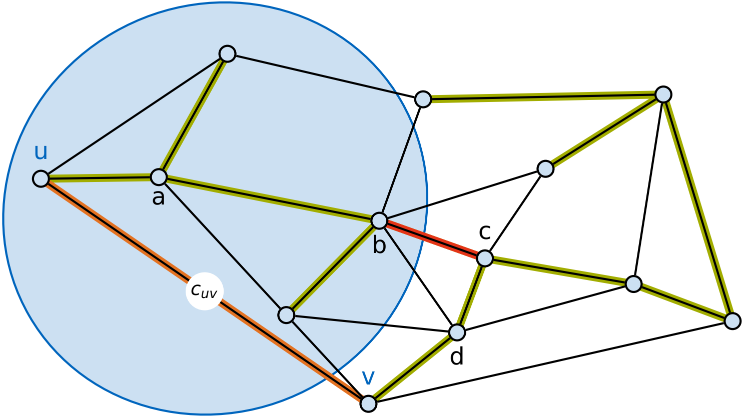
$$\sum_{(x,y)\in E_T} C(x,y) M_{xy} = \sum_{(u,v)\in E} c_{uv} M_{uv}$$



Lemma

Let T be a spanning tree and (V, M) a tree metric of $G = (V, E)$. Then

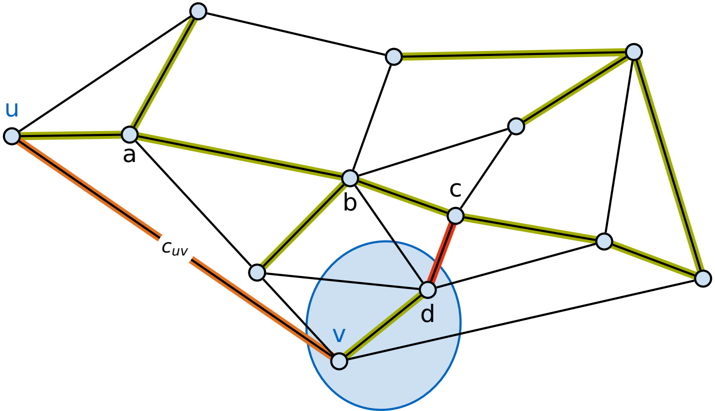
$$\sum_{(x,y)\in E_T} C(x,y) M_{xy} = \sum_{(u,v)\in E} c_{uv} M_{uv}$$



Lemma

Let T be a spanning tree and (V, M) a tree metric of $G = (V, E)$. Then

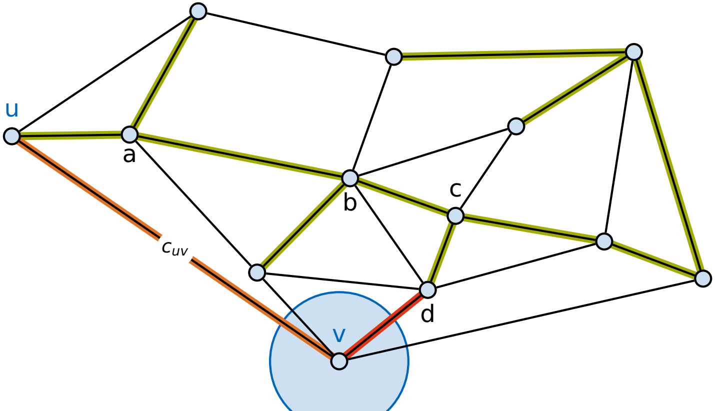
$$\sum_{(x,y)\in E_T} C(x,y) M_{xy} = \sum_{(u,v)\in E} c_{uv} M_{uv}$$



Lemma

Let T be a spanning tree and (V, M) a tree metric of $G = (V, E)$. Then

$$\sum_{(x,y)\in E_T} C(x,y)M_{xy} = \sum_{(u,v)\in E} c_{uv}M_{uv}$$



Value of the Dual Problem

Dual Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.

$$\begin{aligned} \max_{\mathcal{L}} \quad & \min_{i \in \mathcal{I}} \frac{\sum_{e_T \in T_i} C_i(e_T) d_\ell(e_T)}{\sum_{u,v \in V} c_{uv} \ell_{uv}} \\ \text{s.t.} \quad & \mathcal{L} \geq 0 \end{aligned}$$

For any $\mathcal{L}$ we know that for the minimizing tree T_i holds

$$\begin{aligned} \sum_{e_T \in T_i} C_i(e_T) d_\ell(e_T) &\leq \sum_{e_T \in T_i} C_i(e_T) M_{e_T} \\ &= \sum_{u,v \in V} c_{uv} M_{uv} \\ &\leq \mathcal{O}(\log n) \sum_{u,v \in V} c_{uv} d_\ell(u,v) \\ &\leq \mathcal{O}(\log n) \sum_{u,v \in V} c_{uv} \ell_{uv} \end{aligned}$$

$$\frac{\sum_{e_T \in T_i} C_i(e_T) d_\ell(e_T)}{\sum_{u,v \in V} c_{uv} \ell_{uv}} \leq \mathcal{O}(\log n)$$

Primal Program

Let $\mathcal{I}$ be the exponentially large set of all pathtrees.
We want to find the best trees with smallest α.

$$\begin{aligned} \min_{\alpha,\lambda} \quad & \alpha \\ \text{s.t.} \quad & \sum_{i \in \mathcal{I}} \lambda_i \sum_{\substack{e_T \in T_i: \\ (u,v) \in P_i(e_T)}} C_i(e_T) \leq \alpha c_{uv} \qquad \forall u,v \in V \\ & \sum_{i \in \mathcal{I}} \lambda_i = 1 \\ & \lambda \geq 0 \end{aligned}$$

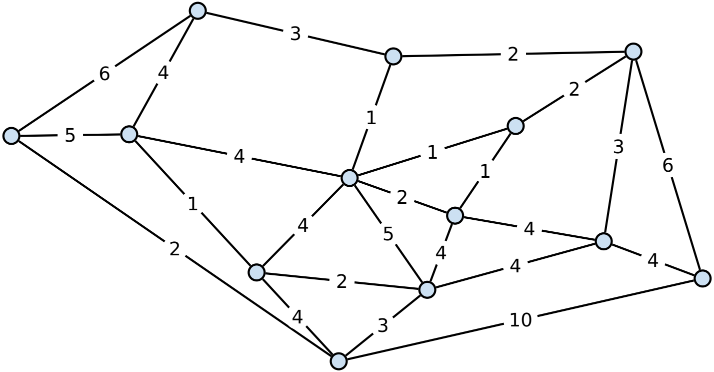
- There is a λ such that $\alpha \in \mathcal{O}(\log n)$
- But why are polynomially many trees enough?
- This gives an $\mathcal{O}(\log n)$-approximation

Problem (Minimum Bisection)

Given

- *An undirected Graph* $G = (V, E)$
- *A cost function* $c : E \to \mathbb{R}^+$

Find a set $S \subset V$ *containing half the vertices with minimal split cost.*

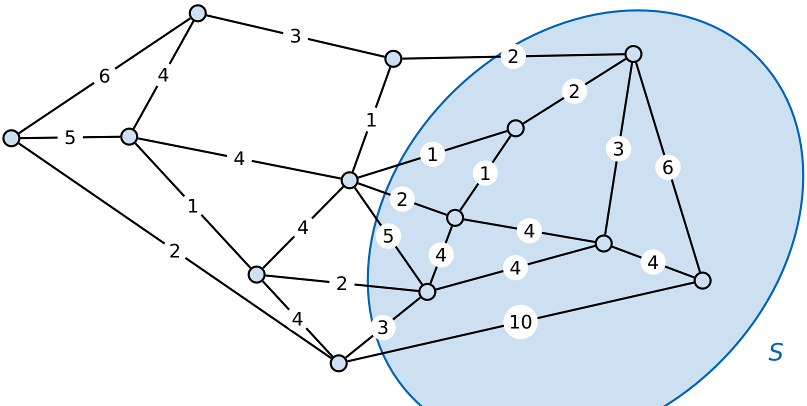


Problem (Minimum Bisection)

Given

- *An undirected Graph* $G = (V, E)$
- *A cost function* $c : E \to \mathbb{R}^+$

Find a set $S \subset V$ *containing half the vertices with minimal split cost.*

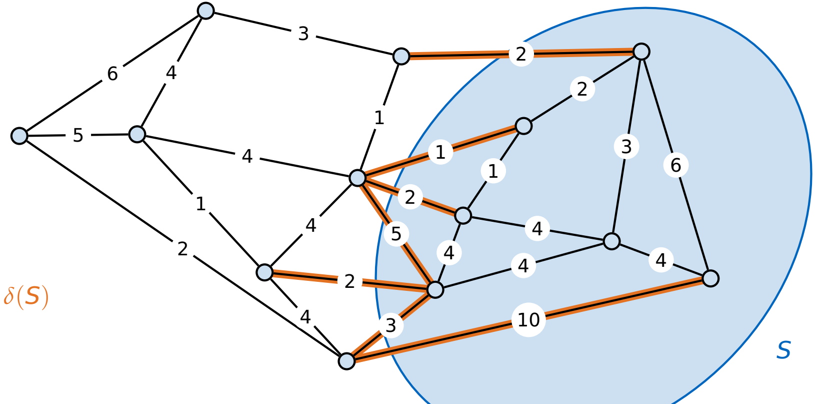


Problem (Minimum Bisection)

Given

- *An undirected Graph* $G = (V, E)$
- *A cost function* $c : E \to \mathbb{R}^+$

Find a set $S \subset V$ *containing half the vertices with minimal split cost.*

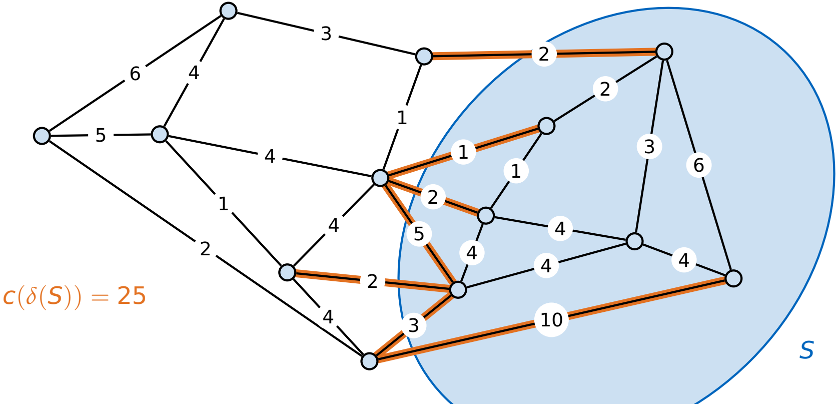


Problem (Minimum Bisection)

Given

- *An undirected Graph* $G = (V, E)$
- *A cost function* $c : E \to \mathbb{R}^+$

Find a set $S \subset V$ *containing half the vertices with minimal split cost.*



Minimum Bisection Approximation

Given graph $G = (V, E)$ and cost function $c : E \to \mathbb{R}^+$.

1. Interpret costs $c(e)$ as capacities
2. Solve oblivious routing on G, obtaining trees T_i
3. Find minimum tree bisections X_i for all trees T_i
4. Choose the X_i with lowest $c(\delta(X_i))$

We have to show

- What the X_i actually are
- An $\mathcal{O}(\log n)$-approximation guarantee
- That we can find the X_i in polynomial time

Minimum Bisection Approximation

Given graph $G = (V, E)$ and cost function $c : E \to \mathbb{R}^+$.

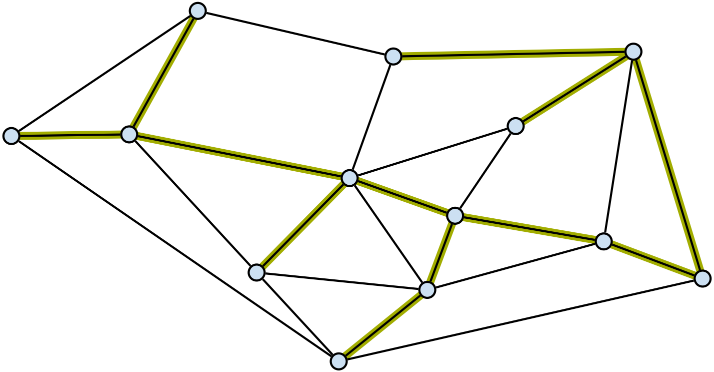
1. Interpret costs $c(e)$ as capacities
2. Solve oblivious routing on G, obtaining trees T_i
3. Find minimum tree bisections X_i for all trees T_i
4. Choose the X_i with lowest $c(\delta(X_i))$

We have to show

- What the X_i actually are
- An $\mathcal{O}(\log n)$-approximation guarantee
- That we can find the X_i in polynomial time

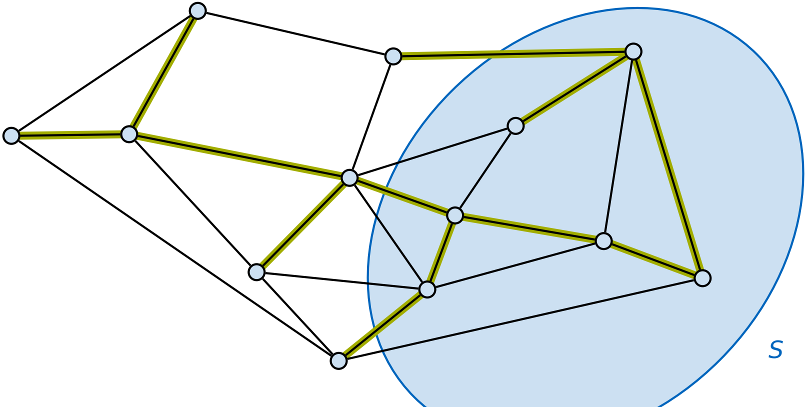
- Given a spanning tree T of G with an edge $e_T \in E_T$
- Define a new cost function c_T using tree splits

$$c_T(e_T) = C(e_T) \qquad\qquad c_T(\delta(S)) = \sum_{\substack{e_T \in E_T: \\ e_T \in \delta(S)}} C(e_T)$$



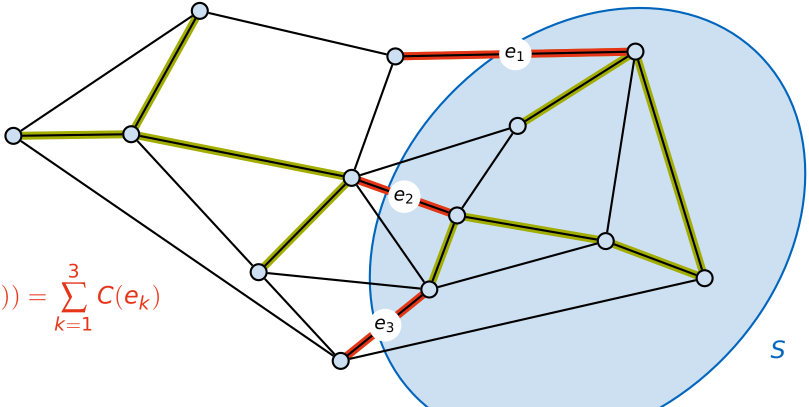
- Given a spanning tree T of G with an edge $e_T \in E_T$
- Define a new cost function c_T using tree splits

$$c_T(e_T) = C(e_T) \qquad c_T(\delta(S)) = \sum_{\substack{e_T \in E_T: \\ e_T \in \delta(S)}} C(e_T)$$



- Given a spanning tree T of G with an edge $e_T \in E_T$
- Define a new cost function c_T using tree splits

$$c_T(e_T) = C(e_T) \qquad c_T(\delta(S)) = \sum_{\substack{e_T \in E_T: \\ e_T \in \delta(S)}} C(e_T)$$

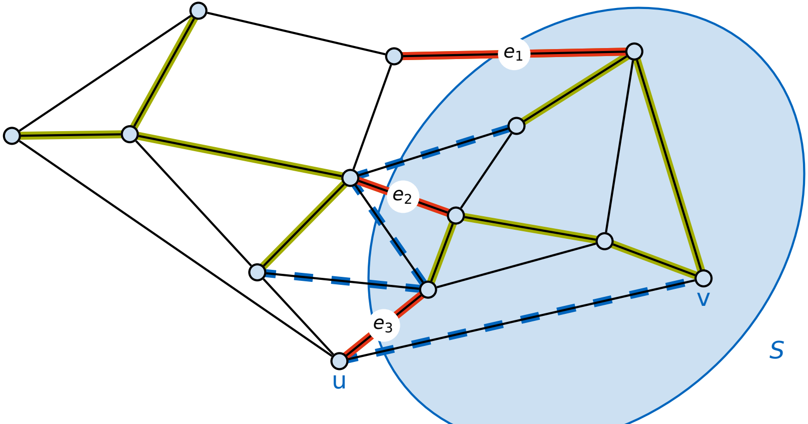


$$c_T(\delta(S)) = \sum_{k=1}^{3} C(e_k)$$

Lemma

For any spanning tree T and any $S \subseteq V$ we have

$$c(\delta(S)) \leq c_T(\delta(S))$$



Lemma

Let $\{T_i\}$ be a solution to the oblivious flow problem on G. Then for any $S \subseteq V$ we have

$$\sum_i \lambda_i c_{T_i}(\delta(S)) \leq \mathcal{O}(\log n) c(\delta(S))$$

- Remember from the primal program that for all $u, v \in V$

$$\sum_i \lambda_i \sum_{\substack{e_T \in T_i: \\ (u,v) \in P_i(e_T)}} C_i(e_T) \leq \mathcal{O}(\log n) c_{uv}$$

- We sum up the inequalities for all $(u, v) \in \delta(S)$

Lemma

Let $\{T_i\}$ be a solution to the oblivious flow problem on G. Then for any $S \subseteq V$ we have

$$\sum_i \lambda_i c_{T_i}(\delta(S)) \leq \mathcal{O}(\log n) c(\delta(S))$$

- We sum up the inequalities for all $(u,v) \in \delta(S)$
- This gives us

$$\sum_i \lambda_i \sum_{(u,v)\in\delta(S)} \sum_{\substack{e_T \in T_i:\\ (u,v)\in P_i(e_T)}} C_i(e_T) \leq \mathcal{O}(\log n) c(\delta(S))$$

- We are done with the observation that

$$c_{T_i}(\delta(S)) = \sum_{\substack{e_T \in E_{T_i}:\\ e_T \in \delta(S)}} C_i(e_T) \leq \sum_{(u,v)\in\delta(S)} \sum_{\substack{e_T \in T_i:\\ (u,v)\in P_i(e_T)}} C_i(e_T)$$

Minimum Bisection Approximation

Given graph $G = (V, E)$ and cost function $c : E \to \mathbb{R}^+$.

1. Interpret costs $c(e)$ as capacities
2. Solve oblivious routing on G, obtaining trees T_i
3. Find minimum tree bisections X_i for all trees T_i
4. Choose the X_i with lowest $c(\delta(X_i))$

- Let now X^*, X_i be the optimal solutions on G and the T_i. Then

$$\begin{aligned}\sum_i \lambda_i c(\delta(X_i)) &\leq \sum_i \lambda_i c_{T_i}(\delta(X_i)) \\ &\leq \sum_i \lambda_i c_{T_i}(\delta(X^*)) \\ &\leq \mathcal{O}(\log n) c(\delta(X^*))\end{aligned}$$

- This also holds for the best X_i, giving an $\mathcal{O}(\log n)$-approximation
- How to find the X_i?