16 Gomory Hu Trees

Given an undirected, weighted graph $G = (V, E, c)$ a cut-tree $T = (V, F, w)$ is a tree with edge-set F and capacities w that fulfills the following properties.

1. **Equivalent Flow Tree:** For any pair of vertices $s, t \in V$, $f(s, t)$ in G is equal to $f_T(s, t)$.
2. **Cut Property:** A minimum s-t cut in T is also a minimum cut in G.

Here, $f(s, t)$ is the value of a maximum s-t flow in G, and $f_T(s, t)$ is the corresponding value in T.

Overview of the Algorithm

The algorithm maintains a partition of V, (sets $S_1, \ldots, S_t$), and a spanning tree T on the vertex set $\{S_1, \ldots, S_t\}$.

Overview of the Algorithm

The algorithm maintains a partition of V, (sets $S_1, \ldots, S_t$), and a spanning tree T on the vertex set $\{S_1, \ldots, S_t\}$.

Initially, there exists only the set $S_1 = V$.

Overview of the Algorithm

The algorithm maintains a partition of V, (sets $S_1, \dots, S_t$), and a spanning tree T on the vertex set $\{S_1, \dots, S_t\}$.

Initially, there exists only the set $S_1 = V$.

Then the algorithm performs $n - 1$ split-operations:

Overview of the Algorithm

The algorithm maintains a partition of V, (sets $S_1, \dots, S_t$), and a spanning tree T on the vertex set $\{S_1, \dots, S_t\}$.

Initially, there exists only the set $S_1 = V$.

Then the algorithm performs $n - 1$ split-operations:

- In each such split-operation it chooses a set S_i with $|S_i| \geq 2$ and splits this set into two non-empty parts X and Y.

Overview of the Algorithm

The algorithm maintains a partition of V, (sets $S_1, \ldots, S_t$), and a spanning tree T on the vertex set $\{S_1, \ldots, S_t\}$.

Initially, there exists only the set $S_1 = V$.

Then the algorithm performs $n - 1$ split-operations:

- In each such split-operation it chooses a set S_i with $|S_i| \geq 2$ and splits this set into two non-empty parts X and Y.
- S_i is then removed from T and replaced by X and Y.

Overview of the Algorithm

The algorithm maintains a partition of V, (sets $S_1, \dots, S_t$), and a spanning tree T on the vertex set $\{S_1, \dots, S_t\}$.

Initially, there exists only the set $S_1 = V$.

Then the algorithm performs $n - 1$ split-operations:

- In each such split-operation it chooses a set S_i with $|S_i| \geq 2$ and splits this set into two non-empty parts X and Y.
- S_i is then removed from T and replaced by X and Y.
- X and Y are connected by an edge, and the edges that before the split were incident to S_i are attached to either X or Y.

Overview of the Algorithm

The algorithm maintains a partition of V, (sets $S_1, \ldots, S_t$), and a spanning tree T on the vertex set $\{S_1, \ldots, S_t\}$.

Initially, there exists only the set $S_1 = V$.

Then the algorithm performs $n - 1$ split-operations:

- In each such split-operation it chooses a set S_i with $|S_i| \geq 2$ and splits this set into two non-empty parts X and Y.
- S_i is then removed from T and replaced by X and Y.
- X and Y are connected by an edge, and the edges that before the split were incident to S_i are attached to either X or Y.

In the end this gives a tree on the vertex set V.

Details of the Split-operation

- Select S_i that contains at least two nodes a and b.

Details of the Split-operation

- Select S_i that contains at least two nodes a and b.
- Compute the connected components of the forest obtained from the current tree T after deleting S_i. Each of these components corresponds to a set of vertices from V.

Details of the Split-operation

- Select S_i that contains at least two nodes a and b.
- Compute the connected components of the forest obtained from the current tree T after deleting S_i. Each of these components corresponds to a set of vertices from V.
- Consider the graph H obtained from G by contracting these connected components into single nodes.

Details of the Split-operation

- Select S_i that contains at least two nodes a and b.
- Compute the connected components of the forest obtained from the current tree T after deleting S_i. Each of these components corresponds to a set of vertices from V.
- Consider the graph H obtained from G by contracting these connected components into single nodes.
- Compute a minimum a-b cut in H. Let A, and B denote the two sides of this cut.

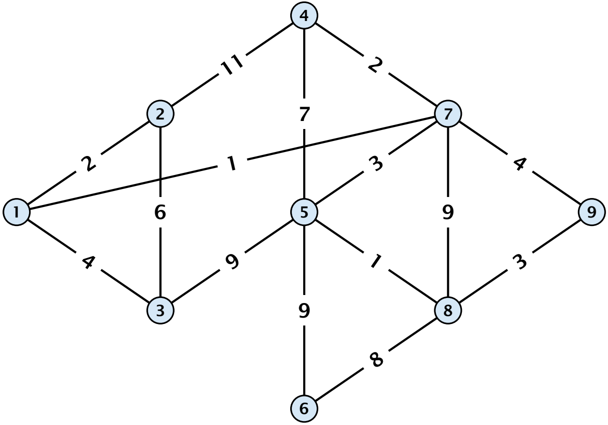
Details of the Split-operation

- Select S_i that contains at least two nodes a and b.
- Compute the connected components of the forest obtained from the current tree T after deleting S_i. Each of these components corresponds to a set of vertices from V.
- Consider the graph H obtained from G by contracting these connected components into single nodes.
- Compute a minimum a-b cut in H. Let A, and B denote the two sides of this cut.
- Split S_i in T into two sets/nodes $S_i^a \coloneqq S_i \cap A$ and $S_i^b \coloneqq S_i \cap B$ and add edge $\{S_i^a, S_i^b\}$ with capacity $f_H(a, b)$.

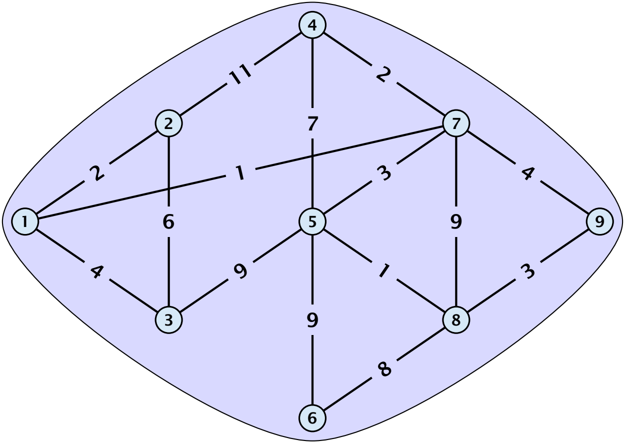
Details of the Split-operation

- Select S_i that contains at least two nodes a and b.
- Compute the connected components of the forest obtained from the current tree T after deleting S_i. Each of these components corresponds to a set of vertices from V.
- Consider the graph H obtained from G by contracting these connected components into single nodes.
- Compute a minimum a-b cut in H. Let A, and B denote the two sides of this cut.
- Split S_i in T into two sets/nodes $S_i^a \coloneqq S_i \cap A$ and $S_i^b \coloneqq S_i \cap B$ and add edge $\{S_i^a, S_i^b\}$ with capacity $f_H(a, b)$.
- Replace an edge $\{S_i, S_x\}$ by $\{S_i^a, S_x\}$ if $S_x \subset A$ and by $\{S_i^b, S_x\}$ if $S_x \subset B$.

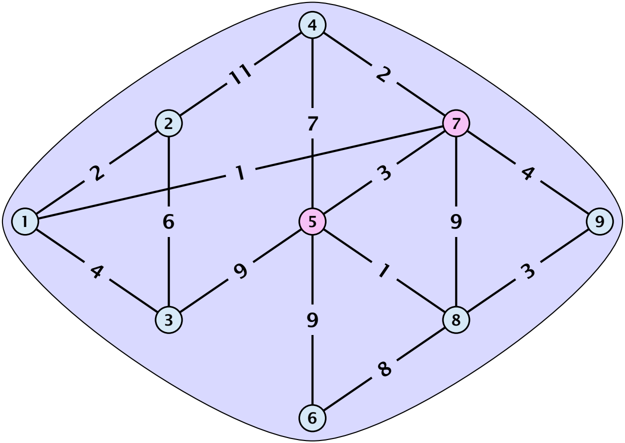
Example: Gomory-Hu Construction



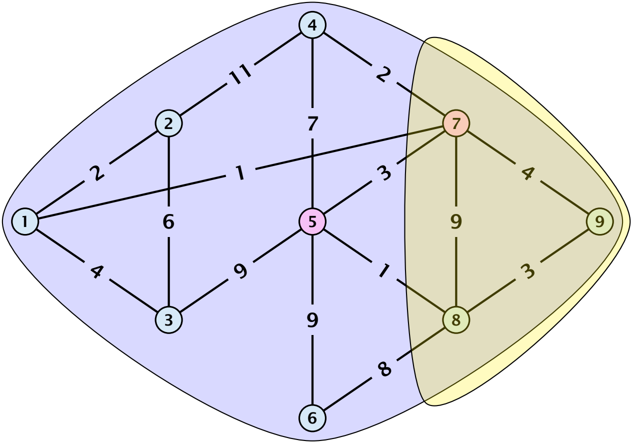
Example: Gomory-Hu Construction



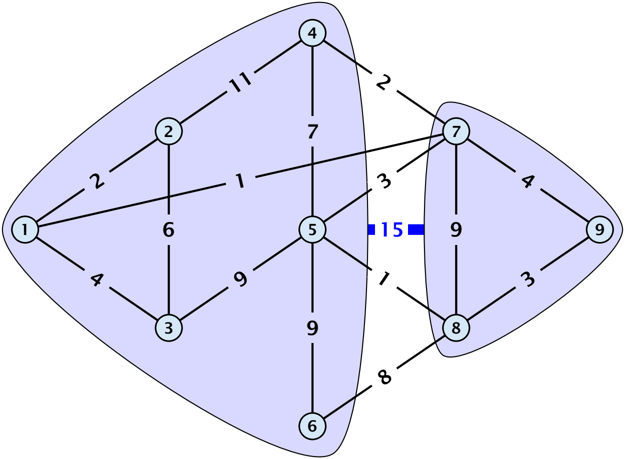
Example: Gomory-Hu Construction



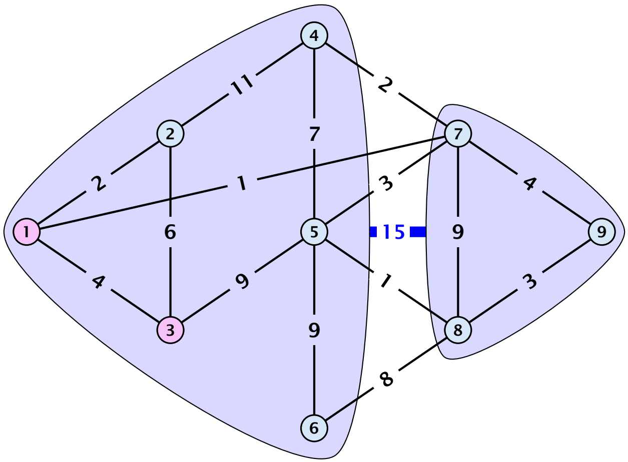
Example: Gomory-Hu Construction



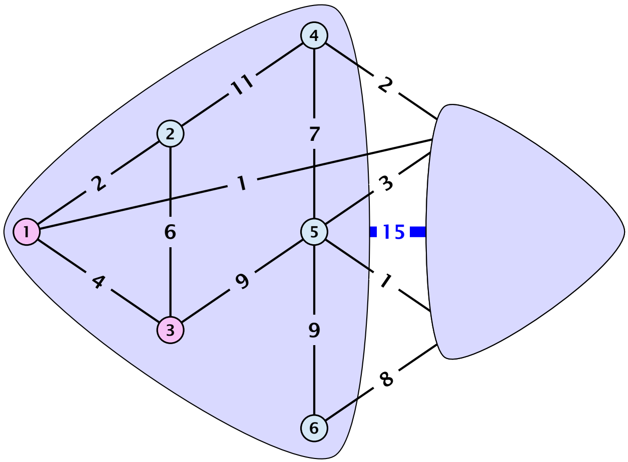
Example: Gomory-Hu Construction



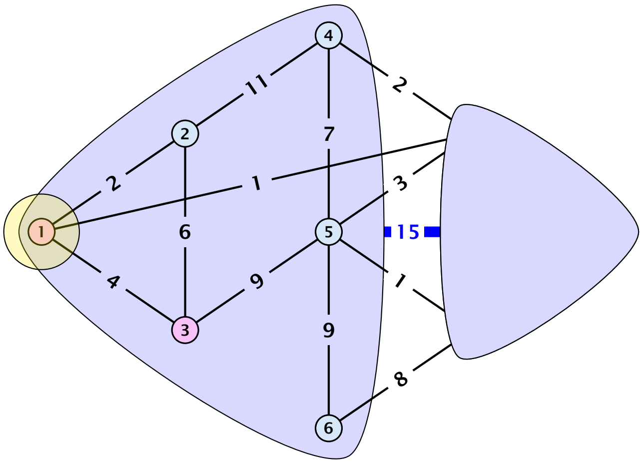
Example: Gomory-Hu Construction



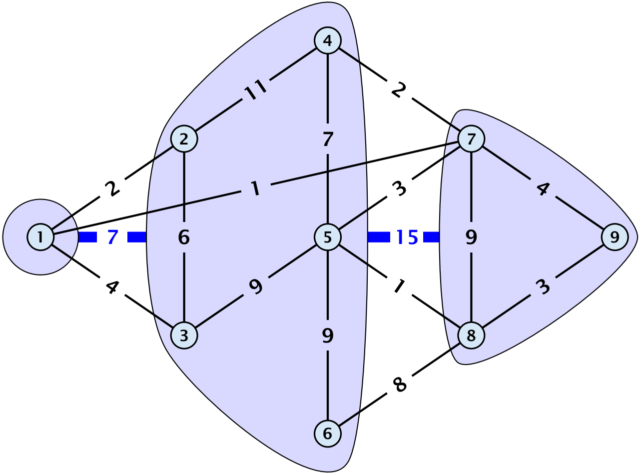
Example: Gomory-Hu Construction



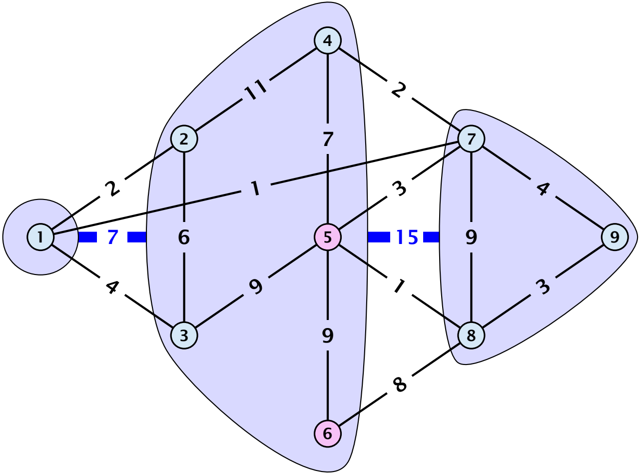
Example: Gomory-Hu Construction



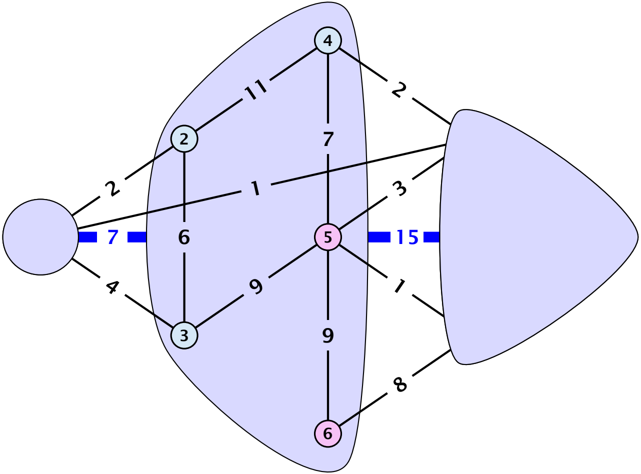
Example: Gomory-Hu Construction



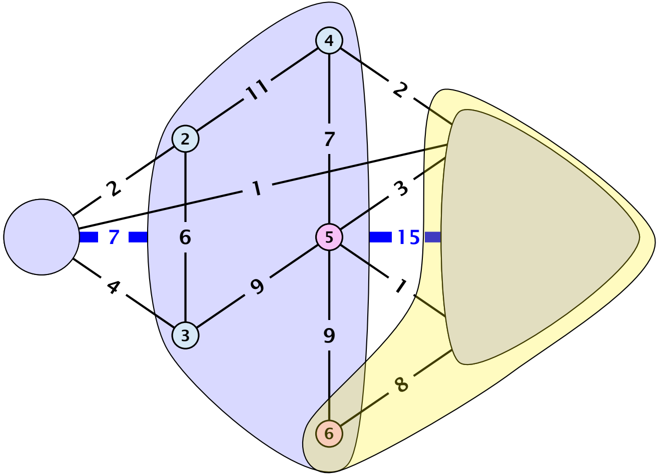
Example: Gomory-Hu Construction



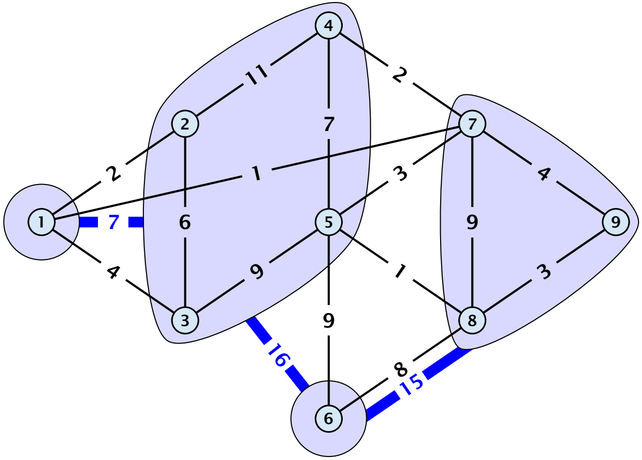
Example: Gomory-Hu Construction



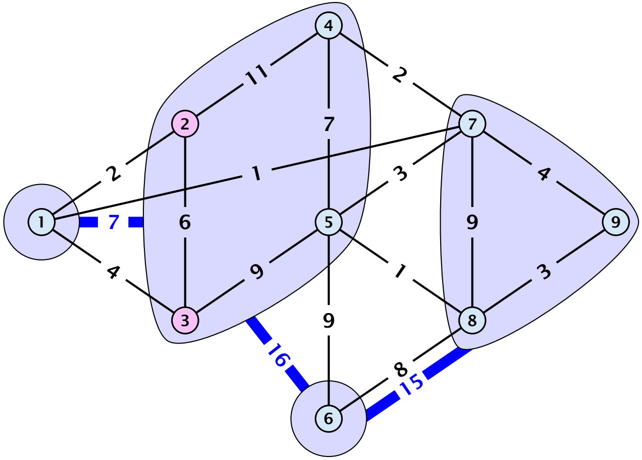
Example: Gomory-Hu Construction



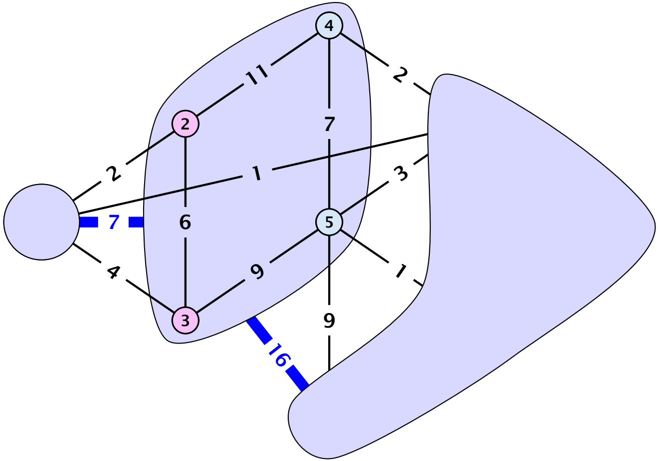
Example: Gomory-Hu Construction



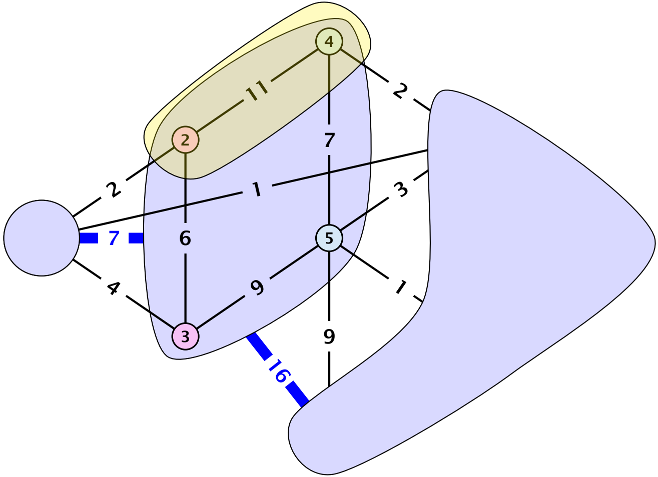
Example: Gomory-Hu Construction



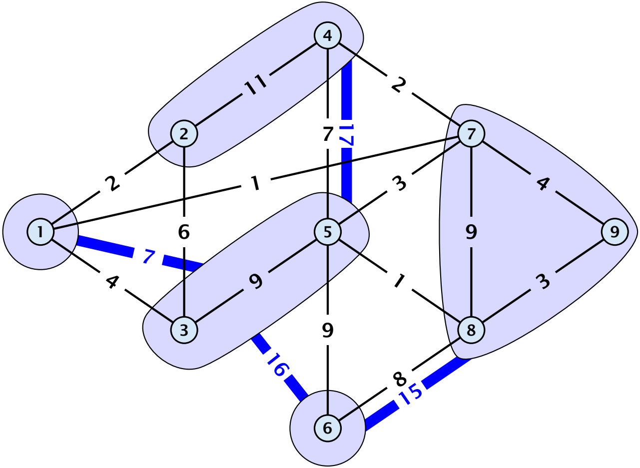
Example: Gomory-Hu Construction



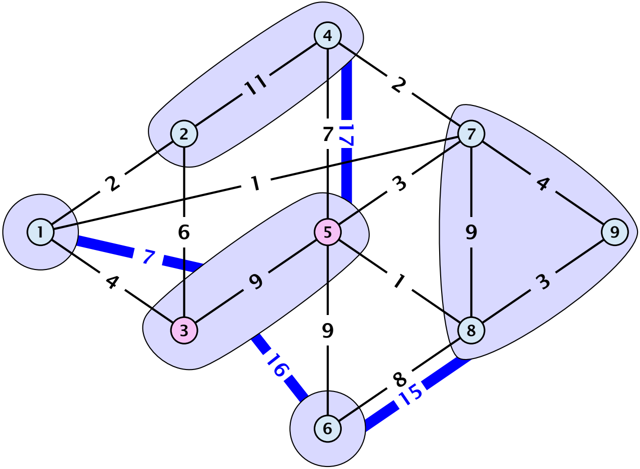
Example: Gomory-Hu Construction



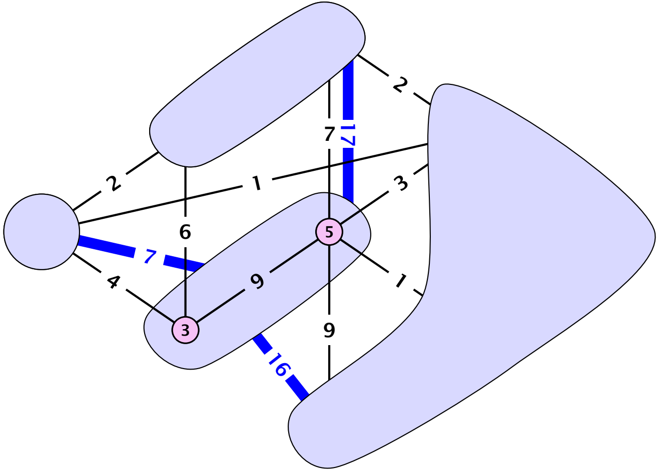
Example: Gomory-Hu Construction



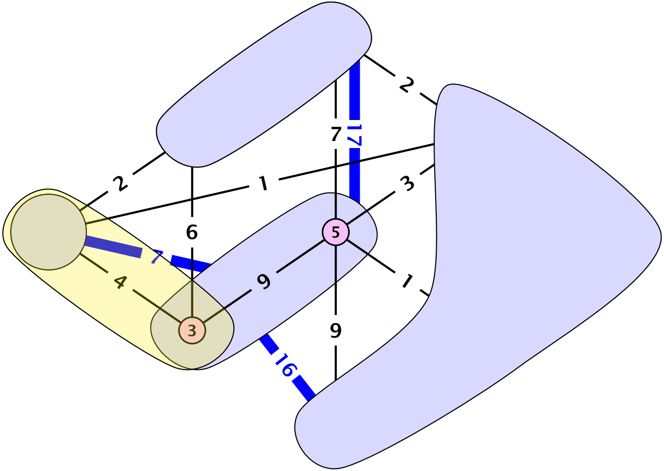
Example: Gomory-Hu Construction



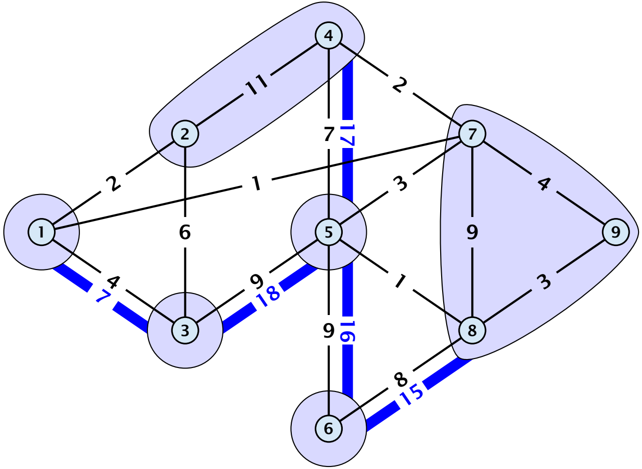
Example: Gomory-Hu Construction



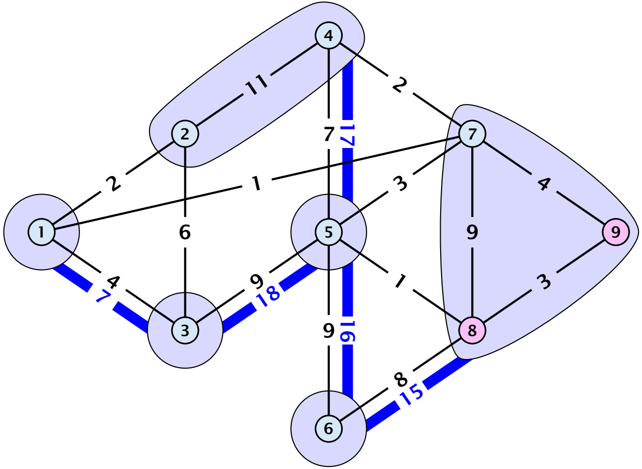
Example: Gomory-Hu Construction



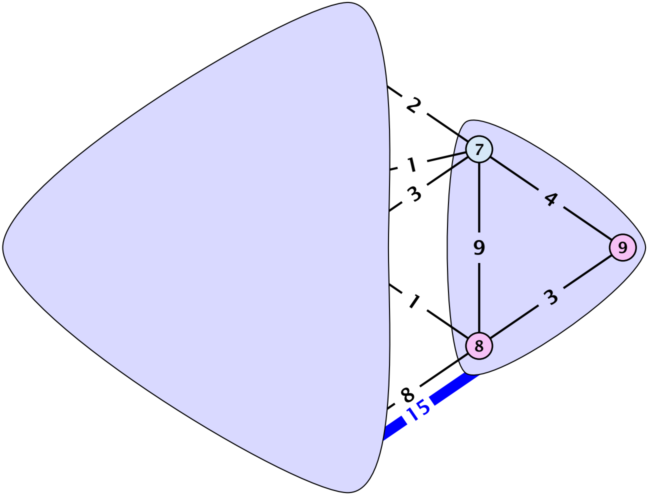
Example: Gomory-Hu Construction



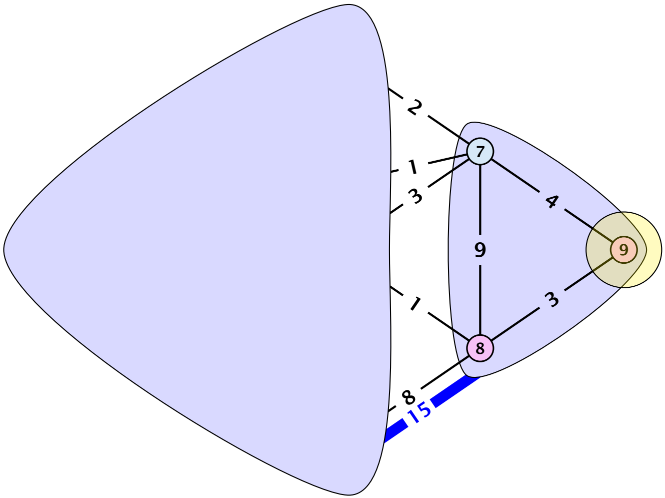
Example: Gomory-Hu Construction



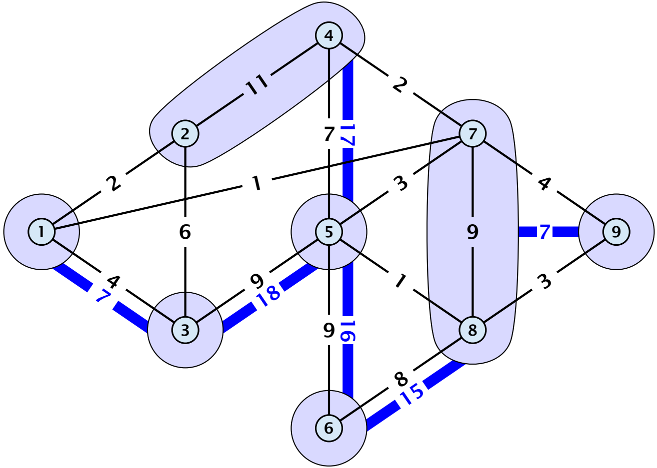
Example: Gomory-Hu Construction



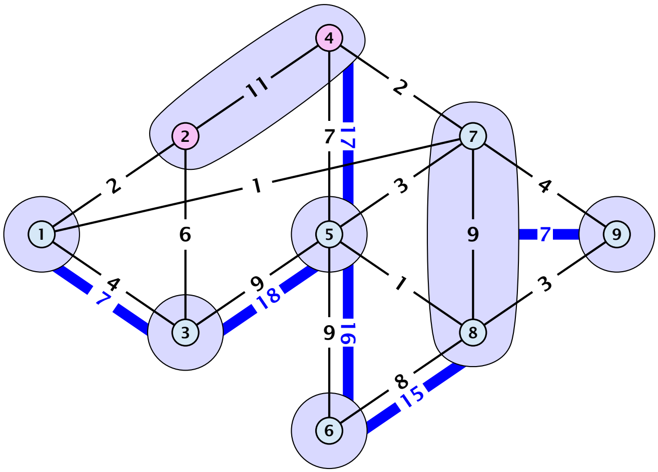
Example: Gomory-Hu Construction



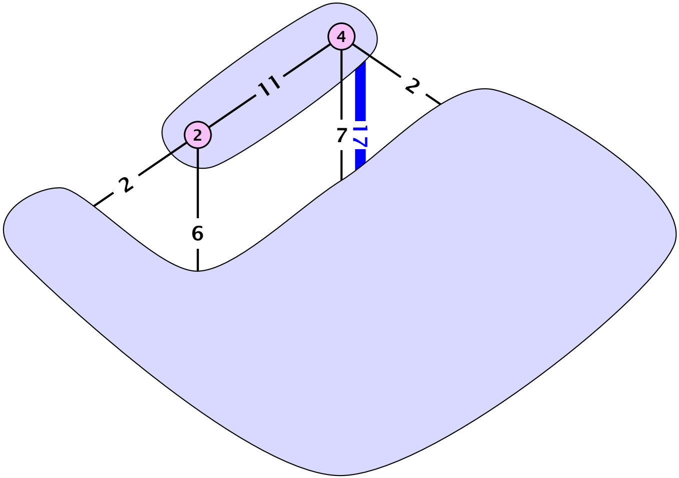
Example: Gomory-Hu Construction



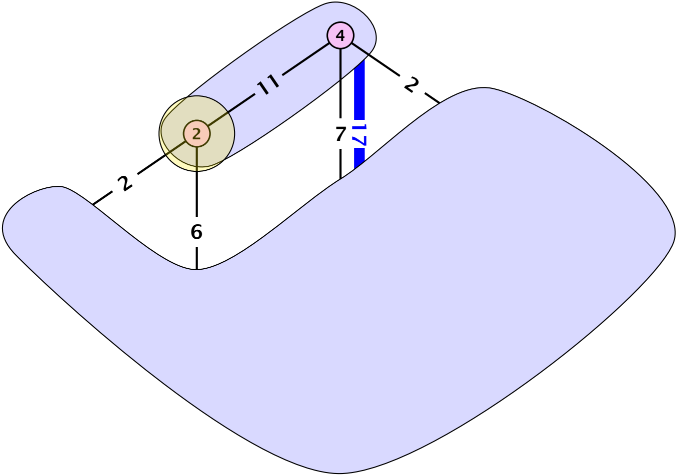
Example: Gomory-Hu Construction



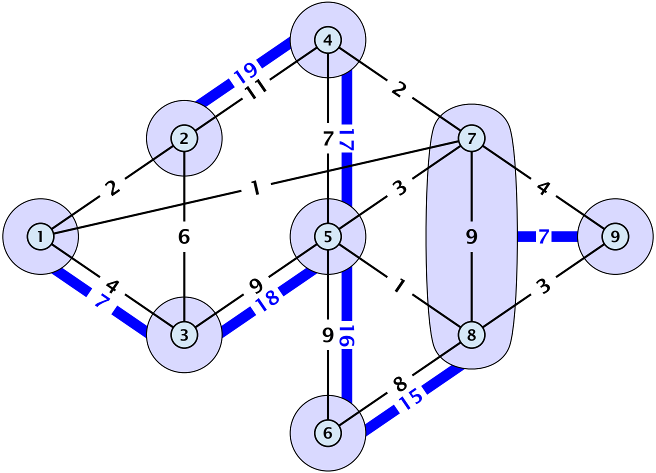
Example: Gomory-Hu Construction



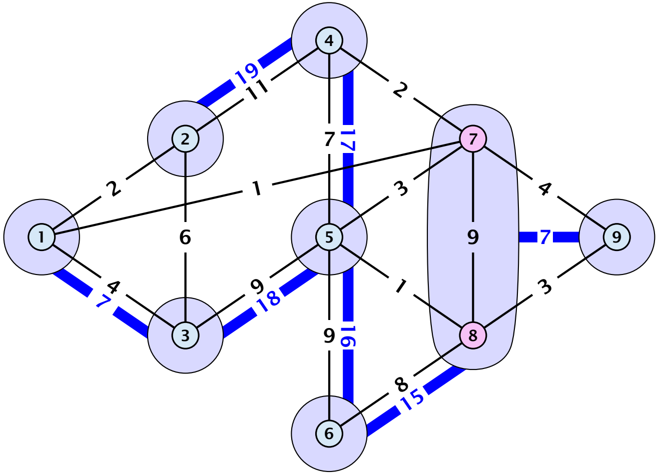
Example: Gomory-Hu Construction



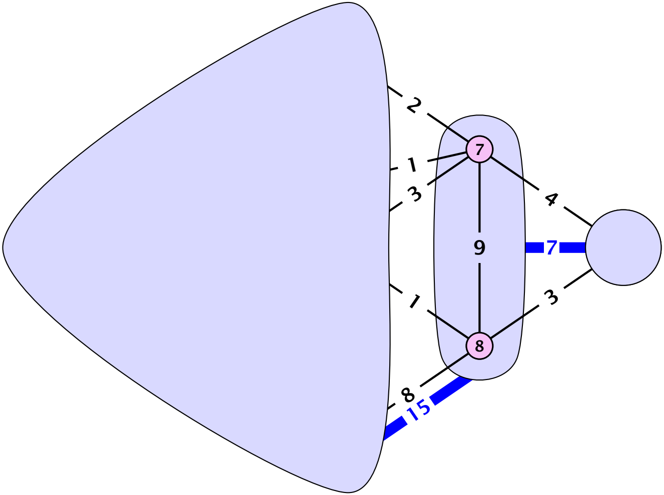
Example: Gomory-Hu Construction



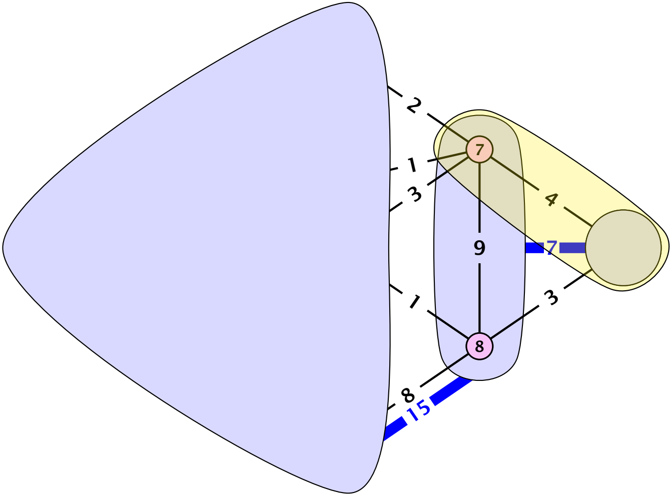
Example: Gomory-Hu Construction



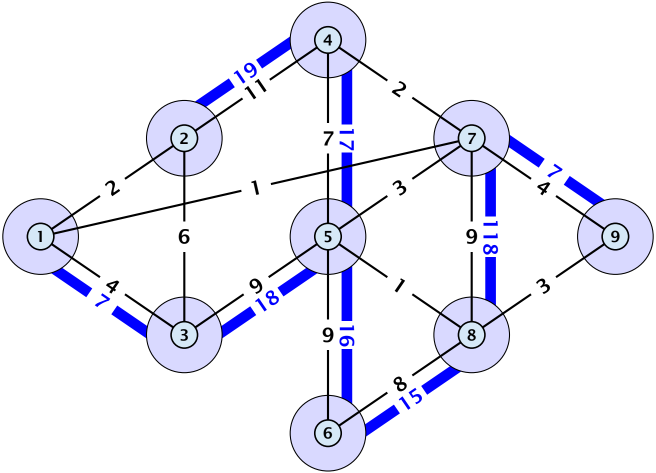
Example: Gomory-Hu Construction



Example: Gomory-Hu Construction



Example: Gomory-Hu Construction



Analysis

Lemma 7
For nodes $s, t, x \in V$ *we have* $f(s,t) \geq \min\{f(s,x), f(x,t)\}$

Analysis

Lemma 7
For nodes $s, t, x \in V$ *we have* $f(s,t) \geq \min\{f(s,x), f(x,t)\}$

Lemma 8
For nodes $s, t, x_1, \ldots, x_k \in V$ *we have*
$f(s,t) \geq \min\{f(s,x_1), f(x_1,x_2), \ldots, f(x_{k-1},x_k), f(x_k,t)\}$

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Proof: Let X be a minimum v-w cut with $X \cap S \neq \emptyset$ and $X \cap (V \setminus S) \neq \emptyset$.

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Proof: Let X be a minimum v-w cut with $X \cap S \neq \emptyset$ and $X \cap (V \setminus S) \neq \emptyset$. Note that $S \setminus X$ and $S \cap X$ are v-w cuts inside S.

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Proof: Let X be a minimum v-w cut with $X \cap S \neq \emptyset$ and $X \cap (V \setminus S) \neq \emptyset$. Note that $S \setminus X$ and $S \cap X$ are v-w cuts inside S.

We may assume w.l.o.g. $s \in X$.

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Proof: Let X be a minimum v-w cut with $X \cap S \neq \emptyset$ and $X \cap (V \setminus S) \neq \emptyset$. Note that $S \setminus X$ and $S \cap X$ are v-w cuts inside S.

We may assume w.l.o.g. $s \in X$.

First case $r \in X$.

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Proof: Let X be a minimum v-w cut with $X \cap S \neq \emptyset$ and $X \cap (V \setminus S) \neq \emptyset$. Note that $S \setminus X$ and $S \cap X$ are v-w cuts inside S.

We may assume w.l.o.g. $s \in X$.

First case $r \in X$.

- $\mathrm{cap}(X \setminus S) + \mathrm{cap}(S \setminus X) \leq \mathrm{cap}(S) + \mathrm{cap}(X)$.

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Proof: Let X be a minimum v-w cut with $X \cap S \neq \emptyset$ and $X \cap (V \setminus S) \neq \emptyset$. Note that $S \setminus X$ and $S \cap X$ are v-w cuts inside S.

We may assume w.l.o.g. $s \in X$.

First case $r \in X$.

- $\mathrm{cap}(X \setminus S) + \mathrm{cap}(S \setminus X) \leq \mathrm{cap}(S) + \mathrm{cap}(X)$.
- $\mathrm{cap}(X \setminus S) \geq \mathrm{cap}(S)$ because $X \setminus S$ is an r-s cut.

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Proof: Let X be a minimum v-w cut with $X \cap S \neq \emptyset$ and $X \cap (V \setminus S) \neq \emptyset$. Note that $S \setminus X$ and $S \cap X$ are v-w cuts inside S.

We may assume w.l.o.g. $s \in X$.

First case $r \in X$.

- $\text{cap}(X \setminus S) + \text{cap}(S \setminus X) \leq \text{cap}(S) + \text{cap}(X)$.
- $\text{cap}(X \setminus S) \geq \text{cap}(S)$ because $X \setminus S$ is an r-s cut.
- This gives $\text{cap}(S \setminus X) \leq \text{cap}(X)$.

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Proof: Let X be a minimum v-w cut with $X \cap S \neq \emptyset$ and $X \cap (V \setminus S) \neq \emptyset$. Note that $S \setminus X$ and $S \cap X$ are v-w cuts inside S.

We may assume w.l.o.g. $s \in X$.

First case $r \in X$.

- $\text{cap}(X \setminus S) + \text{cap}(S \setminus X) \leq \text{cap}(S) + \text{cap}(X)$.
- $\text{cap}(X \setminus S) \geq \text{cap}(S)$ because $X \setminus S$ is an r-s cut.
- This gives $\text{cap}(S \setminus X) \leq \text{cap}(X)$.

Second case $r \notin X$.

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Proof: Let X be a minimum v-w cut with $X \cap S \neq \emptyset$ and $X \cap (V \setminus S) \neq \emptyset$. Note that $S \setminus X$ and $S \cap X$ are v-w cuts inside S.

We may assume w.l.o.g. $s \in X$.

First case $r \in X$.

- $\mathrm{cap}(X \setminus S) + \mathrm{cap}(S \setminus X) \leq \mathrm{cap}(S) + \mathrm{cap}(X)$.
- $\mathrm{cap}(X \setminus S) \geq \mathrm{cap}(S)$ because $X \setminus S$ is an r-s cut.
- This gives $\mathrm{cap}(S \setminus X) \leq \mathrm{cap}(X)$.

Second case $r \notin X$.

- $\mathrm{cap}(X \cup S) + \mathrm{cap}(S \cap X) \leq \mathrm{cap}(S) + \mathrm{cap}(X)$.

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Proof: Let X be a minimum v-w cut with $X \cap S \neq \emptyset$ and $X \cap (V \setminus S) \neq \emptyset$. Note that $S \setminus X$ and $S \cap X$ are v-w cuts inside S.

We may assume w.l.o.g. $s \in X$.

First case $r \in X$.

- $\mathrm{cap}(X \setminus S) + \mathrm{cap}(S \setminus X) \leq \mathrm{cap}(S) + \mathrm{cap}(X)$.
- $\mathrm{cap}(X \setminus S) \geq \mathrm{cap}(S)$ because $X \setminus S$ is an r-s cut.
- This gives $\mathrm{cap}(S \setminus X) \leq \mathrm{cap}(X)$.

Second case $r \notin X$.

- $\mathrm{cap}(X \cup S) + \mathrm{cap}(S \cap X) \leq \mathrm{cap}(S) + \mathrm{cap}(X)$.
- $\mathrm{cap}(X \cup S) \geq \mathrm{cap}(S)$ because $X \cup S$ is an r-s cut.

Lemma 9

Let S be some minimum r-s cut for some nodes $r, s \in V$ ($s \in S$), and let $v, w \in S$. Then there is a minimum v-w-cut T with $T \subset S$.

Proof: Let X be a minimum v-w cut with $X \cap S \neq \emptyset$ and $X \cap (V \setminus S) \neq \emptyset$. Note that $S \setminus X$ and $S \cap X$ are v-w cuts inside S.

We may assume w.l.o.g. $s \in X$.

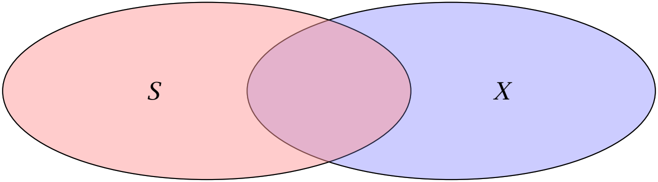
First case $r \in X$.

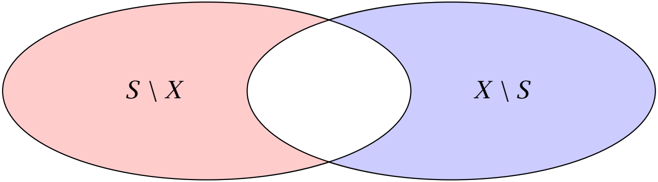
- $\mathrm{cap}(X \setminus S) + \mathrm{cap}(S \setminus X) \leq \mathrm{cap}(S) + \mathrm{cap}(X)$.
- $\mathrm{cap}(X \setminus S) \geq \mathrm{cap}(S)$ because $X \setminus S$ is an r-s cut.
- This gives $\mathrm{cap}(S \setminus X) \leq \mathrm{cap}(X)$.

Second case $r \notin X$.

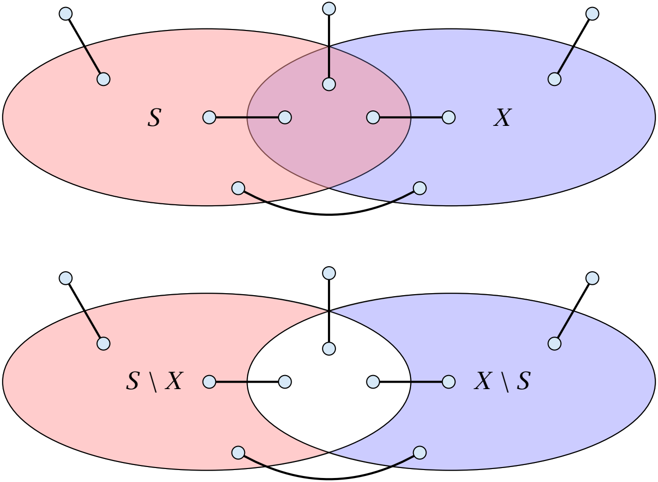
- $\mathrm{cap}(X \cup S) + \mathrm{cap}(S \cap X) \leq \mathrm{cap}(S) + \mathrm{cap}(X)$.
- $\mathrm{cap}(X \cup S) \geq \mathrm{cap}(S)$ because $X \cup S$ is an r-s cut.
- This gives $\mathrm{cap}(S \cap X) \leq \mathrm{cap}(X)$.

$$\mathrm{cap}(S \setminus X) + \mathrm{cap}(X \setminus S) \leq \mathrm{cap}(S) + \mathrm{cap}(X)$$

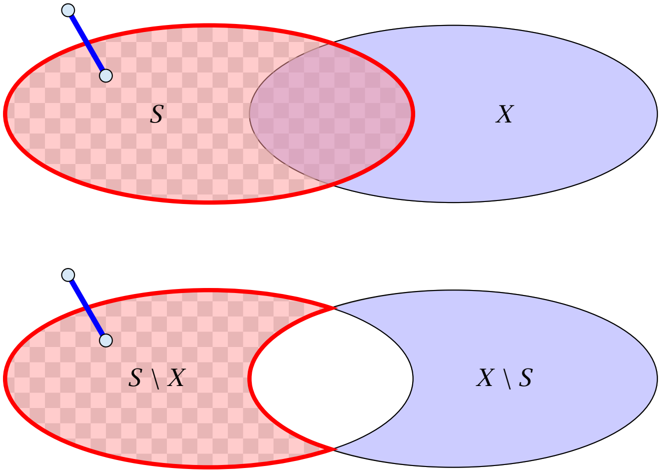




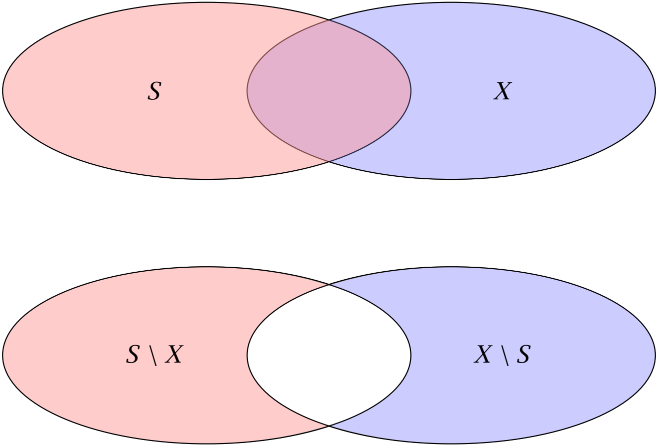
$$\mathbf{cap}(S \setminus X) + \mathbf{cap}(X \setminus S) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



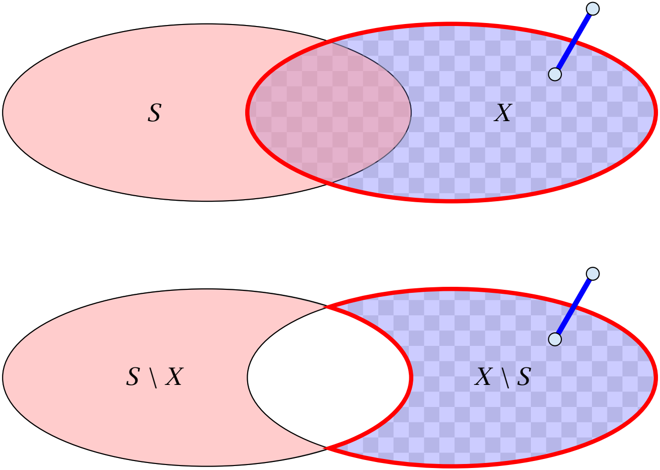
$$\mathbf{cap}(S \setminus X) + \mathbf{cap}(X \setminus S) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



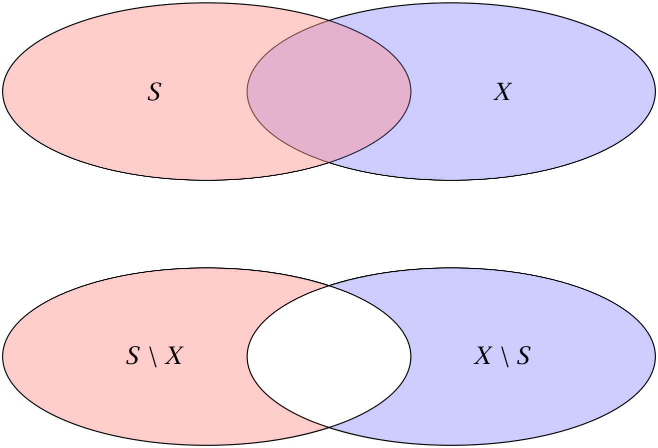
$\mathbf{cap}(S \setminus X) + \mathbf{cap}(X \setminus S) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$



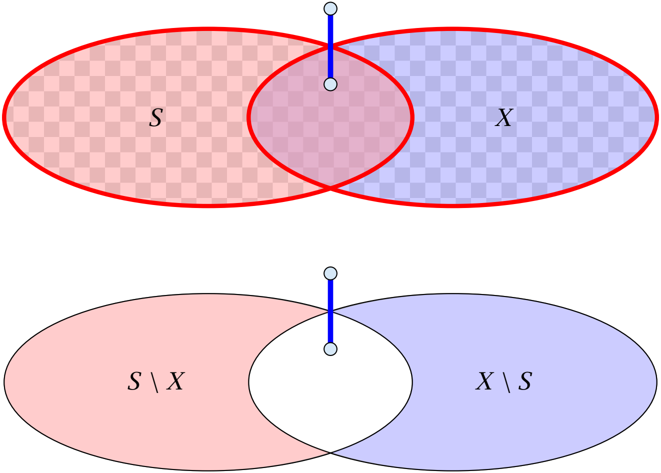
$\mathbf{cap}(S \setminus X) + \mathbf{cap}(X \setminus S) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$



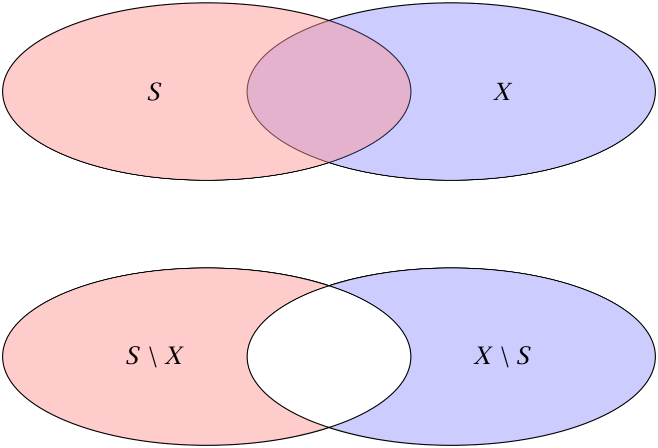
$$\mathrm{cap}(S \setminus X) + \mathrm{cap}(X \setminus S) \leq \mathrm{cap}(S) + \mathrm{cap}(X)$$



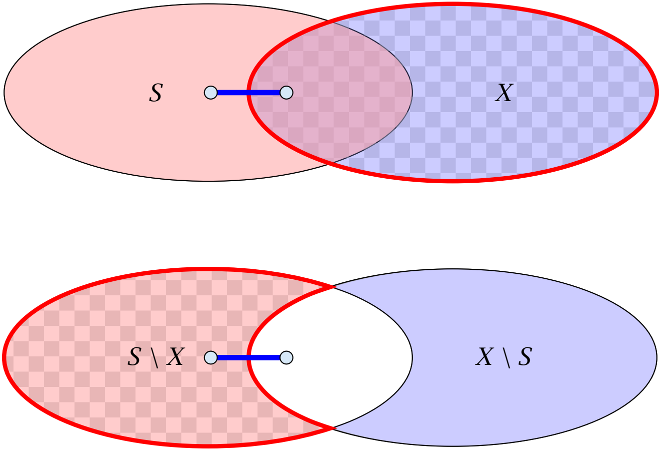
$\text{cap}(S \setminus X) + \text{cap}(X \setminus S) \leq \text{cap}(S) + \text{cap}(X)$



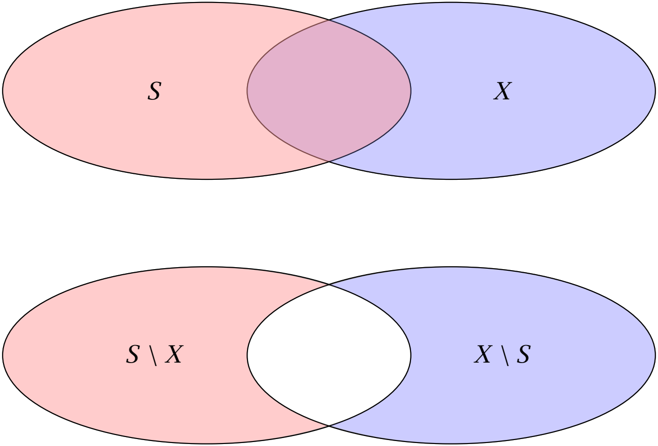
$$\mathbf{cap}(S \setminus X) + \mathbf{cap}(X \setminus S) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



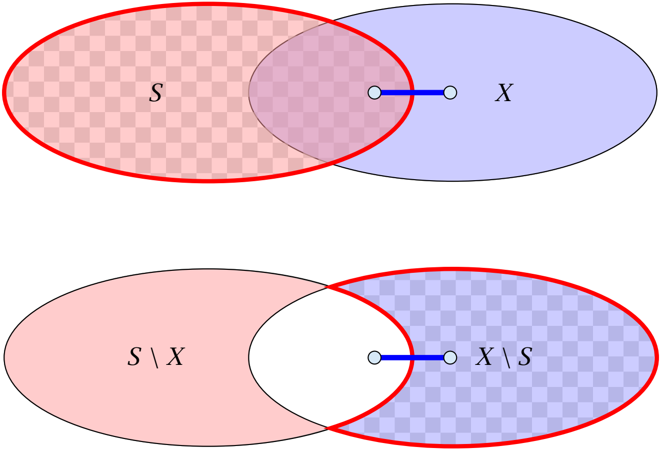
$$\mathbf{cap}(S \setminus X) + \mathbf{cap}(X \setminus S) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



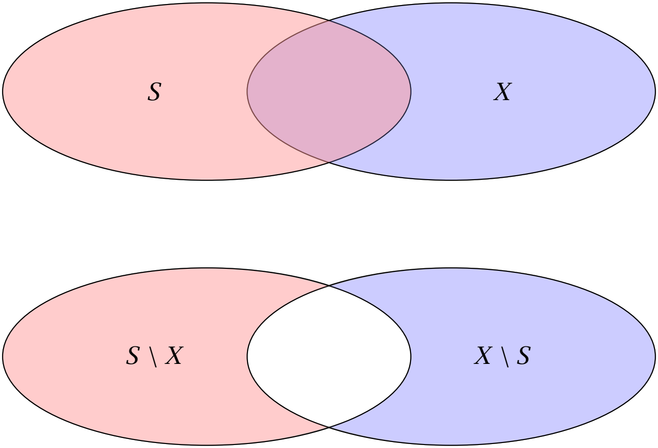
$\mathbf{cap}(S \setminus X) + \mathbf{cap}(X \setminus S) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$



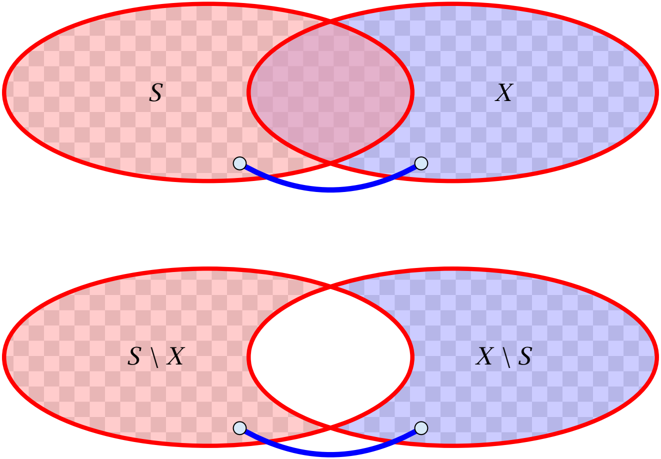
$$\mathrm{cap}(S \setminus X) + \mathrm{cap}(X \setminus S) \le \mathrm{cap}(S) + \mathrm{cap}(X)$$



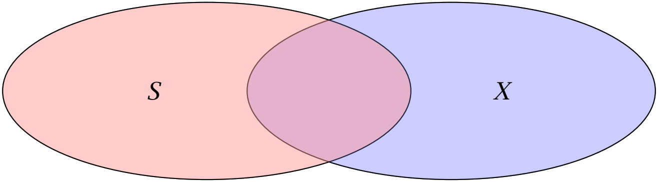
$$\mathbf{cap}(S \setminus X) + \mathbf{cap}(X \setminus S) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$

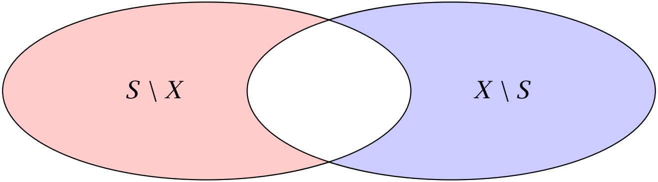


$$\mathbf{cap}(S \setminus X) + \mathbf{cap}(X \setminus S) \le \mathbf{cap}(S) + \mathbf{cap}(X)$$

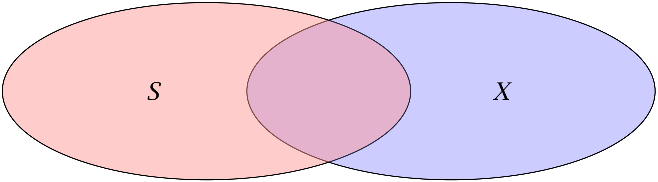


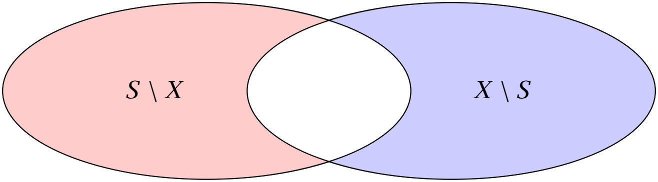
$$\mathbf{cap}(S \setminus X) + \mathbf{cap}(X \setminus S) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



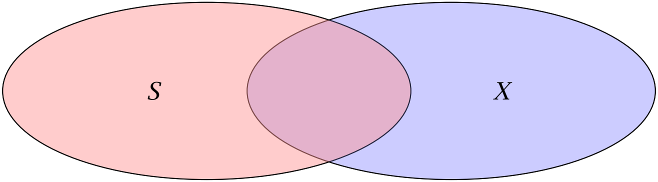


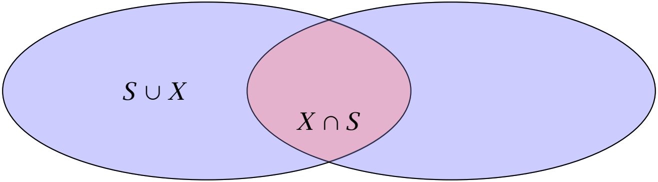
$$\mathbf{cap}(S \setminus X) + \mathbf{cap}(X \setminus S) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



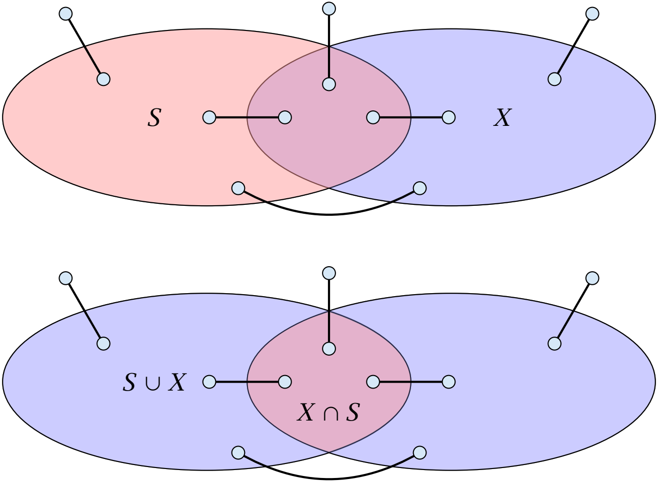


$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$

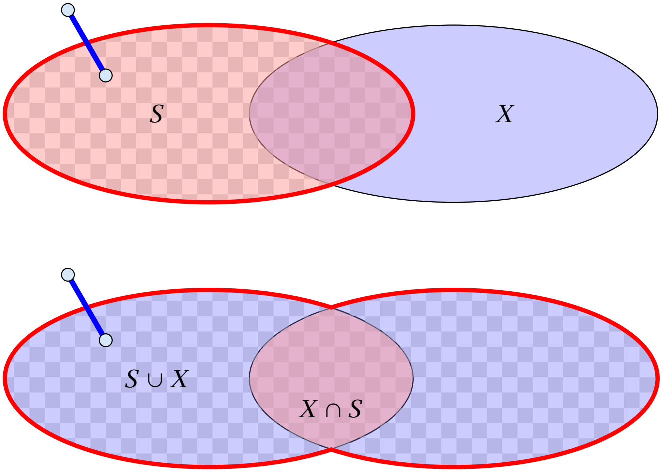




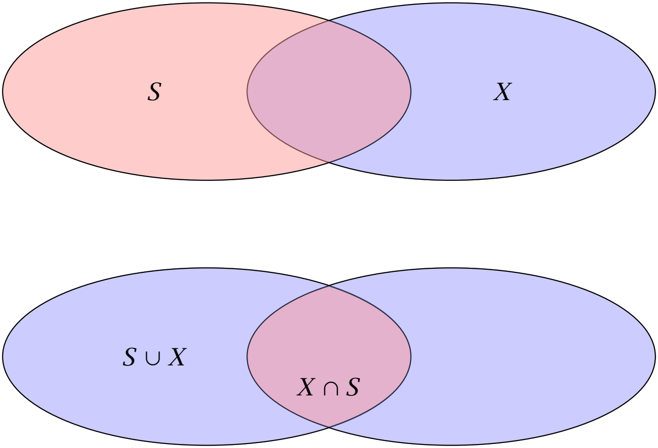
$$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



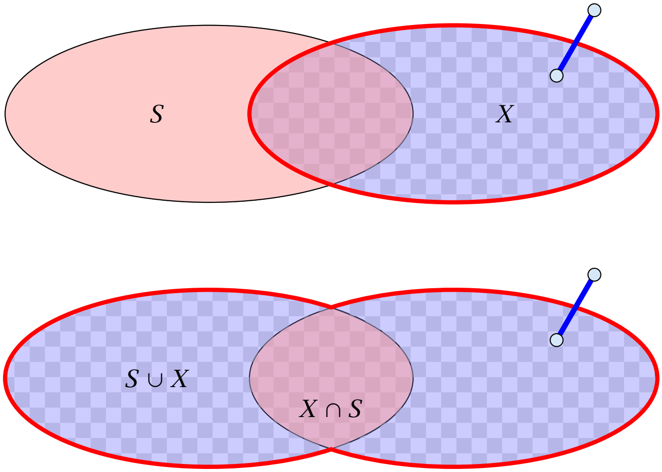
$$\mathrm{cap}(X \cup S) + \mathrm{cap}(S \cap X) \le \mathrm{cap}(S) + \mathrm{cap}(X)$$



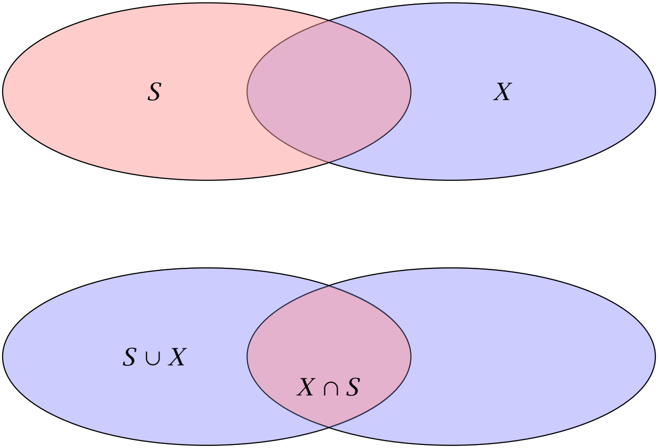
$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \le \mathbf{cap}(S) + \mathbf{cap}(X)$



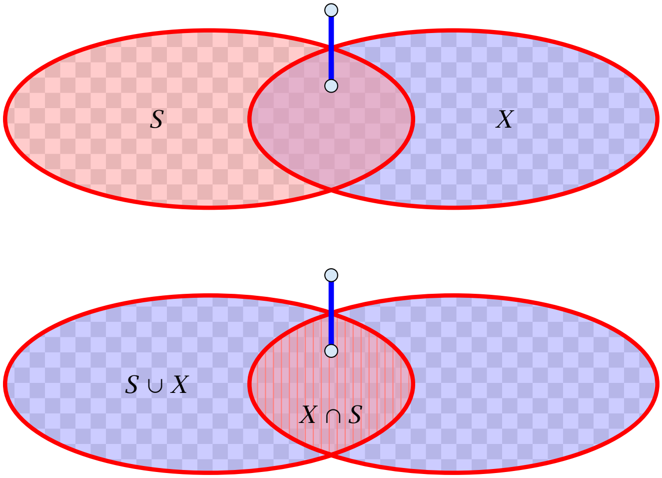
$$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



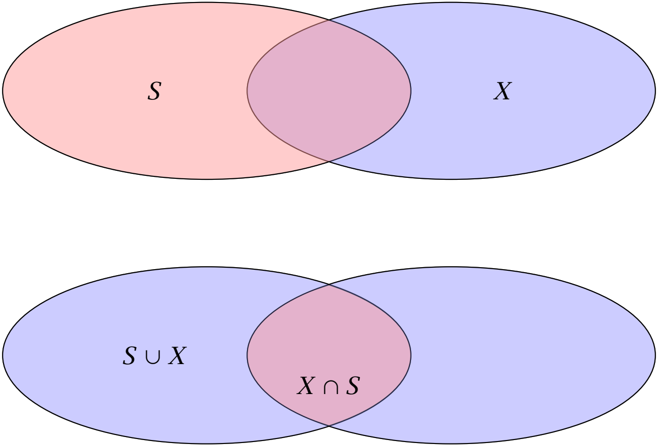
$$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



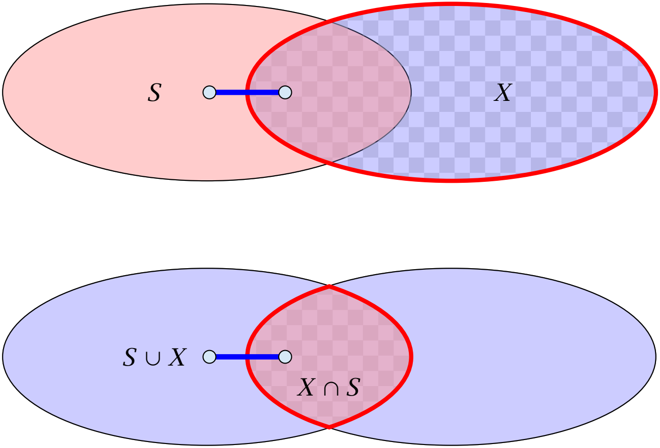
$$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



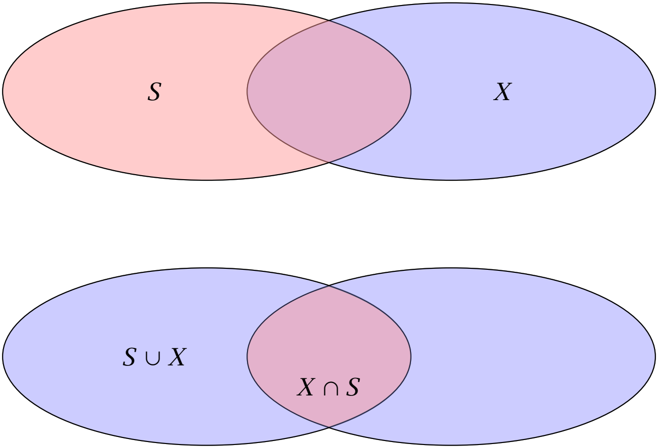
$$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



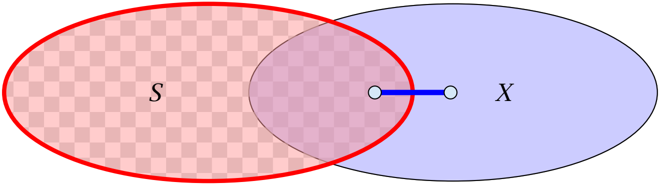
$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$

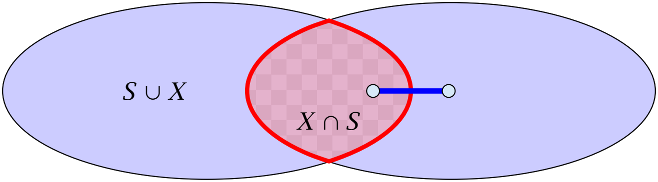


$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$

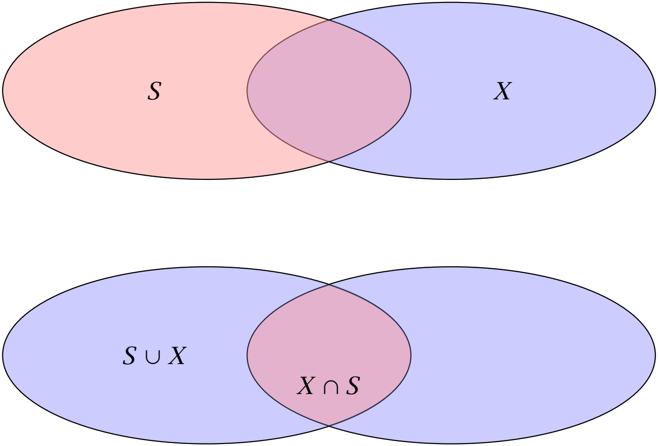


$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \le \mathbf{cap}(S) + \mathbf{cap}(X)$

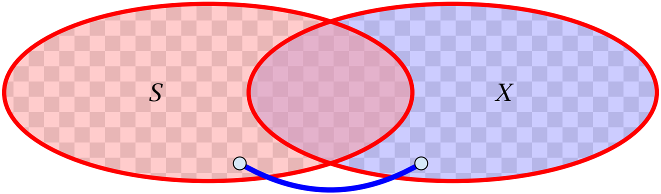


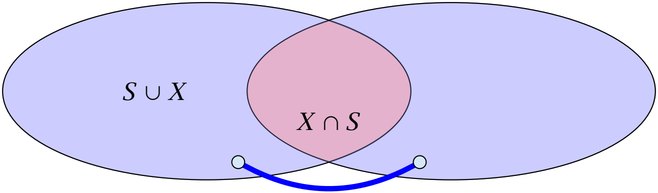


$$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$

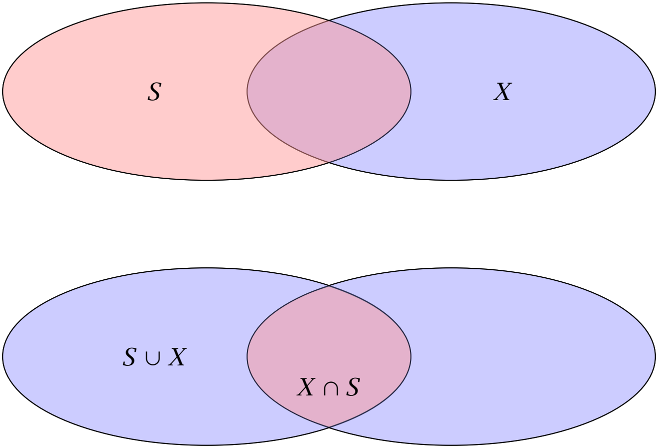


$$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \le \mathbf{cap}(S) + \mathbf{cap}(X)$$

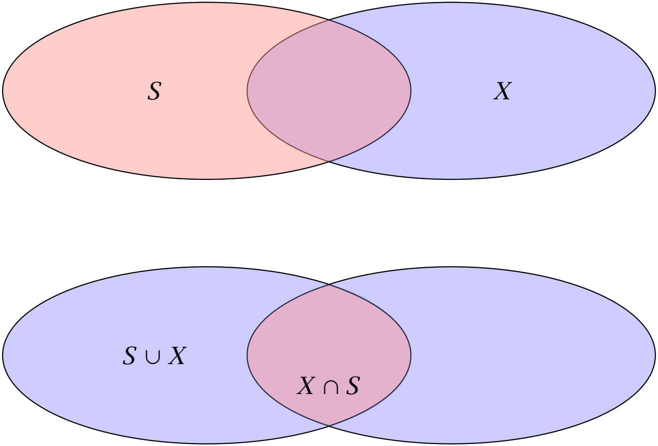




$$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$$



$\mathbf{cap}(X \cup S) + \mathbf{cap}(S \cap X) \leq \mathbf{cap}(S) + \mathbf{cap}(X)$



Analysis

Lemma 9 tells us that if we have a graph $G = (V, E)$ and we contract a subset $X \subset V$ that corresponds to some mincut, then the value of $f(s,t)$ does not change for two nodes $s, t \notin X$.

We will show (later) that the connected components that we contract during a split-operation each correspond to some mincut and, hence, $f_H(s,t) = f(s,t)$, where $f_H(s,t)$ is the value of a minimum s-t mincut in graph H.

Analysis

Invariant [existence of representatives]:
For any edge $\{S_i, S_j\}$ in T, there are vertices $a \in S_i$ and $b \in S_j$ such that $w(S_i, S_j) = f(a, b)$ and the cut defined by edge $\{S_i, S_j\}$ is a minimum a-b cut in G.

Analysis

We first show that the invariant implies that at the end of the algorithm T is indeed a cut-tree.

Analysis

We first show that the invariant implies that at the end of the algorithm T is indeed a cut-tree.

- Let $s = x_0, x_1, \ldots, x_{k-1}, x_k = t$ be the unique simple path from s to t in the final tree T. From the invariant we get that $f(x_i, x_{i+1}) = w(x_i, x_{i+1})$ for all j.

Analysis

We first show that the invariant implies that at the end of the algorithm T is indeed a cut-tree.

- Let $s = x_0, x_1, \ldots, x_{k-1}, x_k = t$ be the unique simple path from s to t in the final tree T. From the invariant we get that $f(x_i, x_{i+1}) = w(x_i, x_{i+1})$ for all j.
- Then

$$f_T(s,t)$$

Analysis

We first show that the invariant implies that at the end of the algorithm T is indeed a cut-tree.

- Let $s = x_0, x_1, \dots, x_{k-1}, x_k = t$ be the unique simple path from s to t in the final tree T. From the invariant we get that $f(x_i, x_{i+1}) = w(x_i, x_{i+1})$ for all j.
- Then

$$f_T(s,t) = \min_{i\in\{0,\dots,k-1\}} \{w(x_i, x_{i+1})\}$$

Analysis

We first show that the invariant implies that at the end of the algorithm T is indeed a cut-tree.

- Let $s = x_0, x_1, \dots, x_{k-1}, x_k = t$ be the unique simple path from s to t in the final tree T. From the invariant we get that $f(x_i, x_{i+1}) = w(x_i, x_{i+1})$ for all j.
- Then

$$\begin{aligned} f_T(s,t) &= \min_{i \in \{0,\dots,k-1\}} \{w(x_i, x_{i+1})\} \\ &= \min_{i \in \{0,\dots,k-1\}} \{f(x_i, x_{i+1})\} \end{aligned}$$

Analysis

We first show that the invariant implies that at the end of the algorithm T is indeed a cut-tree.

- Let $s = x_0, x_1, \dots, x_{k-1}, x_k = t$ be the unique simple path from s to t in the final tree T. From the invariant we get that $f(x_i, x_{i+1}) = w(x_i, x_{i+1})$ for all j.
- Then

$$
\begin{aligned}
f_T(s,t) &= \min_{i\in\{0,\dots,k-1\}} \{w(x_i, x_{i+1})\} \\
&= \min_{i\in\{0,\dots,k-1\}} \{f(x_i, x_{i+1})\} \le f(s,t) \enspace .
\end{aligned}
$$

Analysis

We first show that the invariant implies that at the end of the algorithm T is indeed a cut-tree.

- Let $s = x_0, x_1, \ldots, x_{k-1}, x_k = t$ be the unique simple path from s to t in the final tree T. From the invariant we get that $f(x_i, x_{i+1}) = w(x_i, x_{i+1})$ for all j.
- Then

$$f_T(s,t) = \min_{i \in \{0,\ldots,k-1\}} \{w(x_i, x_{i+1})\}$$
$$= \min_{i \in \{0,\ldots,k-1\}} \{f(x_i, x_{i+1})\} \le f(s,t) \ .$$

- Let $\{x_j, x_{j+1}\}$ be the edge with minimum weight on the path.

Analysis

We first show that the invariant implies that at the end of the algorithm T is indeed a cut-tree.

- Let $s = x_0, x_1, \ldots, x_{k-1}, x_k = t$ be the unique simple path from s to t in the final tree T. From the invariant we get that $f(x_i, x_{i+1}) = w(x_i, x_{i+1})$ for all j.
- Then

$$\begin{aligned} f_T(s,t) &= \min_{i \in \{0,\ldots,k-1\}} \{w(x_i, x_{i+1})\} \\ &= \min_{i \in \{0,\ldots,k-1\}} \{f(x_i, x_{i+1})\} \le f(s,t) \ . \end{aligned}$$

- Let $\{x_j, x_{j+1}\}$ be the edge with minimum weight on the path.
- Since by the invariant this edge induces an s-t cut with capacity $f(x_j, x_{j+1})$ we get $f(s,t) \le f(x_j, x_{j+1}) = f_T(s,t)$.

Analysis

- Hence, $f_T(s,t) = f(s,t)$ (flow equivalence).

Analysis

- Hence, $f_T(s,t) = f(s,t)$ (flow equivalence).
- The edge $\{x_j, x_{j+1}\}$ is a mincut between s and t in T.

Analysis

- Hence, $f_T(s,t) = f(s,t)$ (flow equivalence).
- The edge $\{x_j, x_{j+1}\}$ is a mincut between s and t in T.
- By invariant, it forms a cut with capacity $f(x_j, x_{j+1})$ in G (which separates s and t).

Analysis

- Hence, $f_T(s,t) = f(s,t)$ (flow equivalence).
- The edge $\{x_j, x_{j+1}\}$ is a mincut between s and t in T.
- By invariant, it forms a cut with capacity $f(x_j, x_{j+1})$ in G (which separates s and t).
- Since, we can send a flow of value $f(x_j, x_{j+1})$ btw. s and t, this is an s-t mincut (cut property).

Proof of Invariant

Proof of Invariant

The invariant obviously holds at the beginning of the algorithm.

Proof of Invariant

The invariant obviously holds at the beginning of the algorithm.

Now, we show that it holds after a split-operation provided that it was true before the operation.

Proof of Invariant

The invariant obviously holds at the beginning of the algorithm.

Now, we show that it holds after a split-operation provided that it was true before the operation.

Let S_i denote our selected cluster with nodes a and b. Because of the invariant all edges leaving $\{S_i\}$ in T correspond to some mincuts.

Proof of Invariant

The invariant obviously holds at the beginning of the algorithm.

Now, we show that it holds after a split-operation provided that it was true before the operation.

Let S_i denote our selected cluster with nodes a and b. Because of the invariant all edges leaving $\{S_i\}$ in T correspond to some mincuts.

Therefore, contracting the connected components does not change the mincut btw. a and b due to Lemma 9.

Proof of Invariant

The invariant obviously holds at the beginning of the algorithm.

Now, we show that it holds after a split-operation provided that it was true before the operation.

Let S_i denote our selected cluster with nodes a and b. Because of the invariant all edges leaving $\{S_i\}$ in T correspond to some mincuts.

Therefore, contracting the connected components does not change the mincut btw. a and b due to Lemma 9.

After the split we have to choose representatives for all edges. For the new edge $\{S_i^a, S_i^b\}$ with capacity $w(S_i^a, S_i^b) = f_H(a, b)$ we can simply choose a and b as representatives.

Proof of Invariant

Proof of Invariant

For edges that are not incident to S_i we do not need to change representatives as the neighbouring sets do not change.

Proof of Invariant

For edges that are not incident to S_i we do not need to change representatives as the neighbouring sets do not change.

Consider an edge $\{X, S_i\}$, and suppose that before the split it used representatives $x \in X$, and $s \in S_i$. Assume that this edge is replaced by $\{X, S_i^a\}$ in the new tree (the case when it is replaced by $\{X, S_i^b\}$ is analogous).

Proof of Invariant

For edges that are not incident to S_i we do not need to change representatives as the neighbouring sets do not change.

Consider an edge $\{X, S_i\}$, and suppose that before the split it used representatives $x \in X$, and $s \in S_i$. Assume that this edge is replaced by $\{X, S_i^a\}$ in the new tree (the case when it is replaced by $\{X, S_i^b\}$ is analogous).

If $s \in S_i^a$ we can keep x and s as representatives.

Proof of Invariant

For edges that are not incident to S_i we do not need to change representatives as the neighbouring sets do not change.

Consider an edge $\{X, S_i\}$, and suppose that before the split it used representatives $x \in X$, and $s \in S_i$. Assume that this edge is replaced by $\{X, S_i^a\}$ in the new tree (the case when it is replaced by $\{X, S_i^b\}$ is analogous).

If $s \in S_i^a$ we can keep x and s as representatives.

Otherwise, we choose x and a as representatives. We need to show that $f(x, a) = f(x, s)$.

Proof of Invariant

Proof of Invariant

Because the invariant was true before the split we know that the edge $\{X, S_i\}$ induces a cut in G of capacity $f(x,s)$. Since, x and a are on opposite sides of this cut, we know that $f(x,a) \leq f(x,s)$.

Proof of Invariant

Because the invariant was true before the split we know that the edge $\{X, S_i\}$ induces a cut in G of capacity $f(x, s)$. Since, x and a are on opposite sides of this cut, we know that $f(x, a) \leq f(x, s)$.

The set B forms a mincut separating a from b. Contracting all nodes in this set gives a new graph G' where the set B is represented by node v_B. Because of Lemma 9 we know that $f'(x, a) = f(x, a)$ as $x, a \notin B$.

Proof of Invariant

Because the invariant was true before the split we know that the edge $\{X, S_i\}$ induces a cut in G of capacity $f(x,s)$. Since, x and a are on opposite sides of this cut, we know that $f(x,a) \le f(x,s)$.

The set B forms a mincut separating a from b. Contracting all nodes in this set gives a new graph G' where the set B is represented by node v_B. Because of Lemma 9 we know that $f'(x,a) = f(x,a)$ as $x, a \notin B$.

We further have $f'(x,a) \ge \min\{f'(x,v_B), f'(v_B,a)\}$.

Proof of Invariant

Because the invariant was true before the split we know that the edge $\{X, S_i\}$ induces a cut in G of capacity $f(x,s)$. Since, x and a are on opposite sides of this cut, we know that $f(x,a) \le f(x,s)$.

The set B forms a mincut separating a from b. Contracting all nodes in this set gives a new graph G' where the set B is represented by node v_B. Because of Lemma 9 we know that $f'(x,a) = f(x,a)$ as $x, a \notin B$.

We further have $f'(x,a) \ge \min\{f'(x,v_B), f'(v_B,a)\}$.

Since $s \in B$ we have $f'(v_B,x) \ge f(s,x)$.

Proof of Invariant

Because the invariant was true before the split we know that the edge $\{X, S_i\}$ induces a cut in G of capacity $f(x,s)$. Since, x and a are on opposite sides of this cut, we know that $f(x,a) \le f(x,s)$.

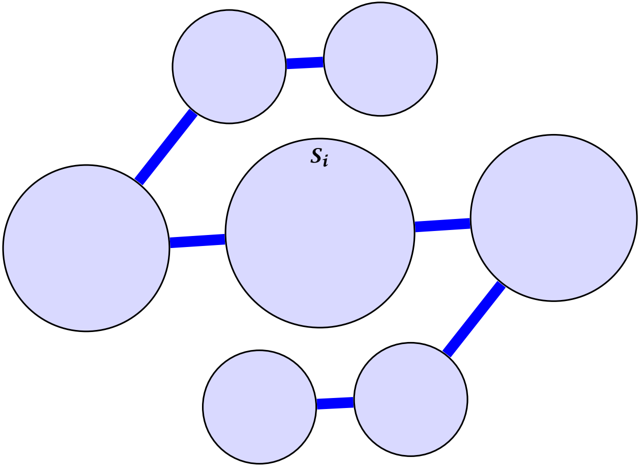
The set B forms a mincut separating a from b. Contracting all nodes in this set gives a new graph G' where the set B is represented by node v_B. Because of Lemma 9 we know that $f'(x,a) = f(x,a)$ as $x, a \notin B$.

We further have $f'(x,a) \ge \min\{f'(x,v_B), f'(v_B,a)\}$.

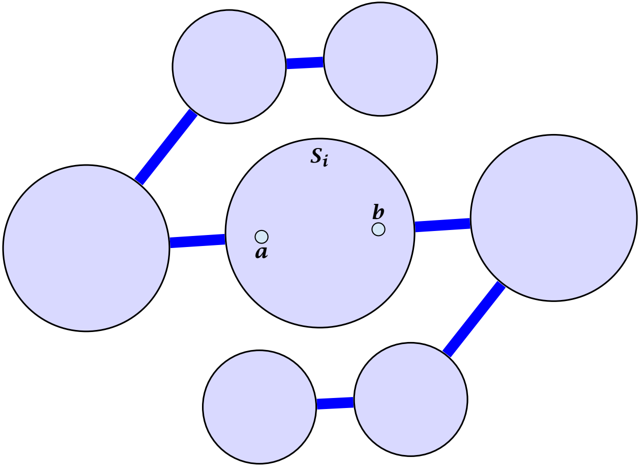
Since $s \in B$ we have $f'(v_B,x) \ge f(s,x)$.

Also, $f'(a,v_B) \ge f(a,b) \ge f(x,s)$ since the a-b cut that splits S_i into S_i^a and S_i^b also separates s and x.

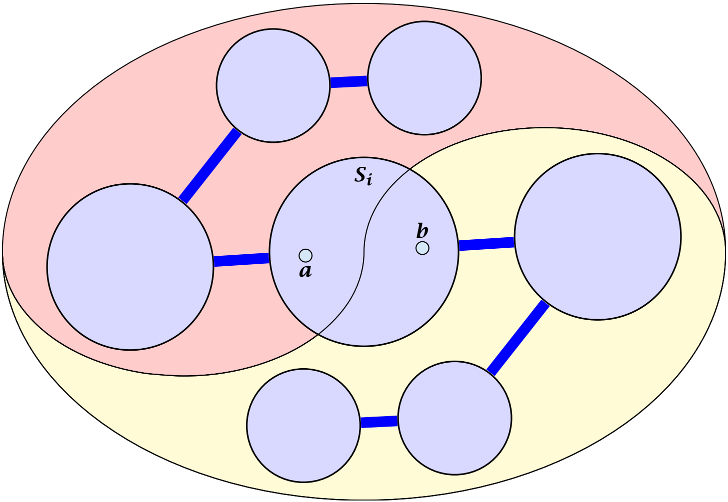
Analysis



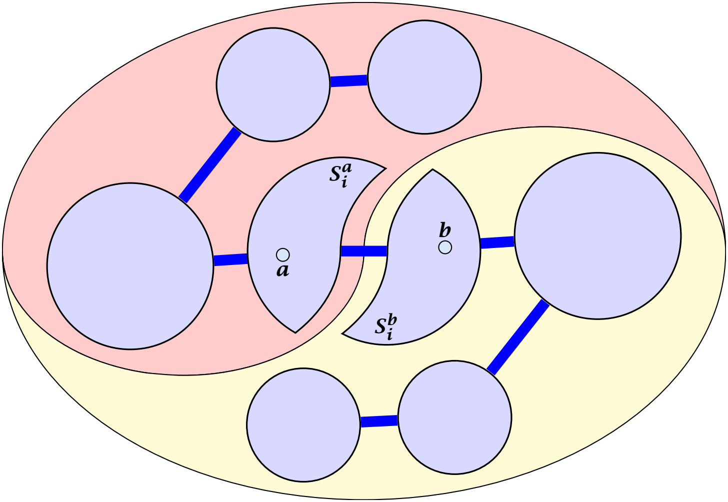
Analysis



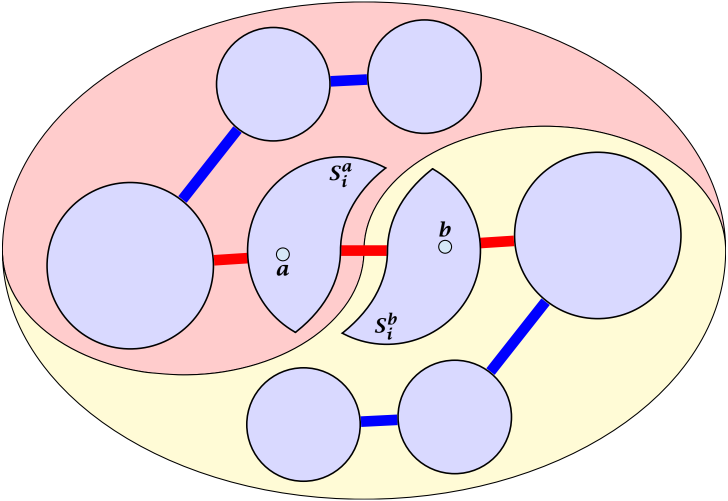
Analysis



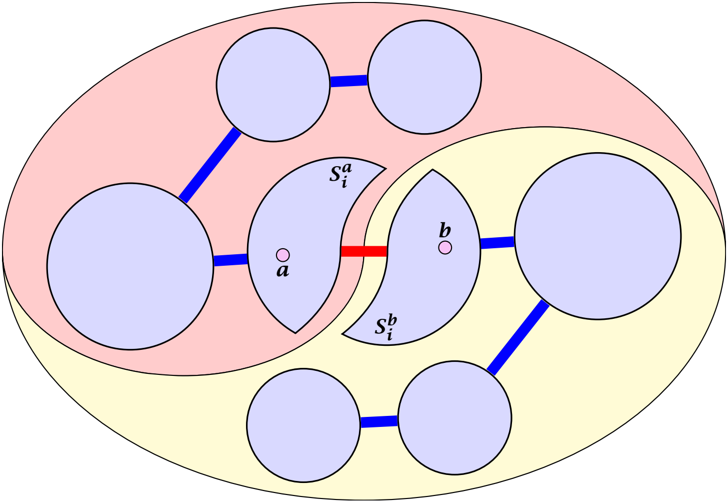
Analysis



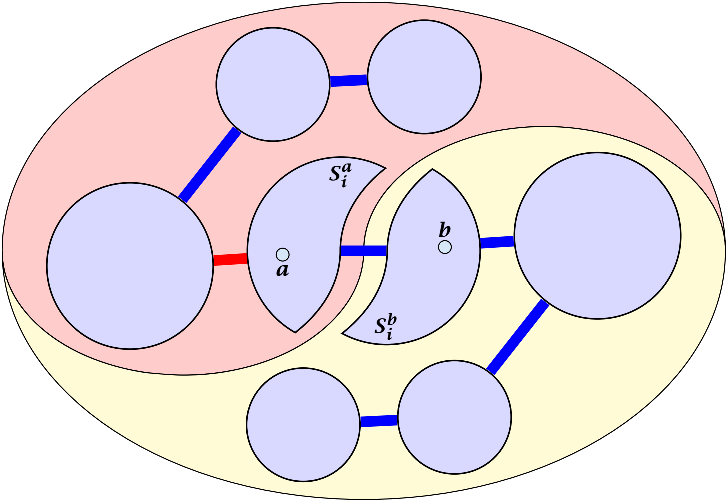
Analysis



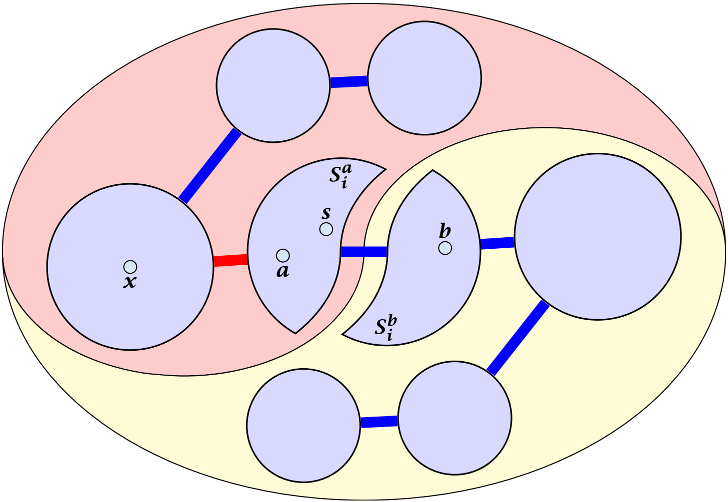
Analysis



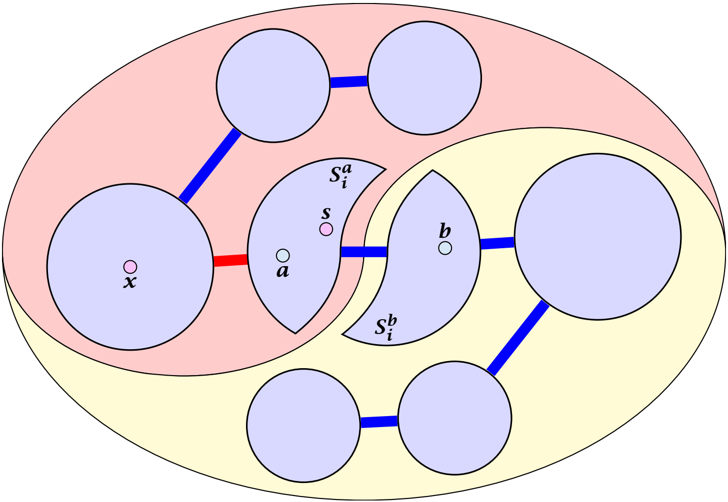
Analysis



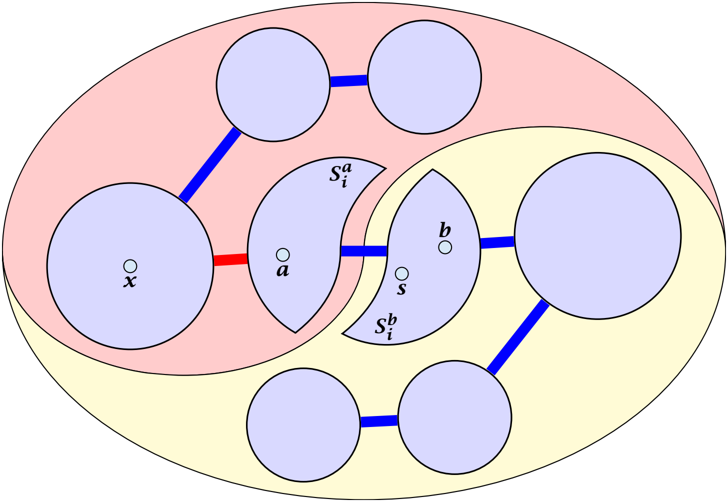
Analysis



Analysis



Analysis



Analysis