Analysis & Design of Algorithms (CS302)

Gomory-Hu Tree Algorithm

Problem Description

Given a graph G(V, E), a *k-cut* is a partition of the vertices V into k sets $S_1, S_2, \ldots, S_k$. Any edge (u,v) $\in$ E with u $\in S_i$ and v $\in S_j$ (i $\neq$ j) is said to be crossing the cut and is a cut edge. The *capacity* of a k-cut is the sum of weights of all the cut edges.

Minimum K-Cut Problem: Given a graph G(V, E), find the minimum weight set of edges E′ $\subseteq$ E such that removing E′ from G leaves k connected components i.e. find the k-cut with the minimum capacity.

Algorithm

MINIMUM K-CUT WITH GOMORY-HU TREE (G(V, E), w: E → R^+, k)

1. Compute the Gomory-Hu Tree of G, call it T
2. Pick the k − 1 lightest edges in the T, call them e_1, e_2, ..., e_{k-1}
3. Return the union of the corresponding cuts, C, in G,
   $C = C_{e1} \cup C_{e2} \cup \ldots \cup C_{ek-1}$

Gomory-Hu Tree

Definition: 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. Each Edge in T corresponds to min-cut between two adjacent vertices in a tree.
2. Equivalent Flow Tree: For any pair of vertices s, t $\in$ V, maximum s-t flow in G is equal to the maximum s-t flow in T.
3. Cut Property: A minimum s-t cut in T is also a minimum s-t cut in G.

Gomory-Hu Tree Construction 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 $S_1 = V$.

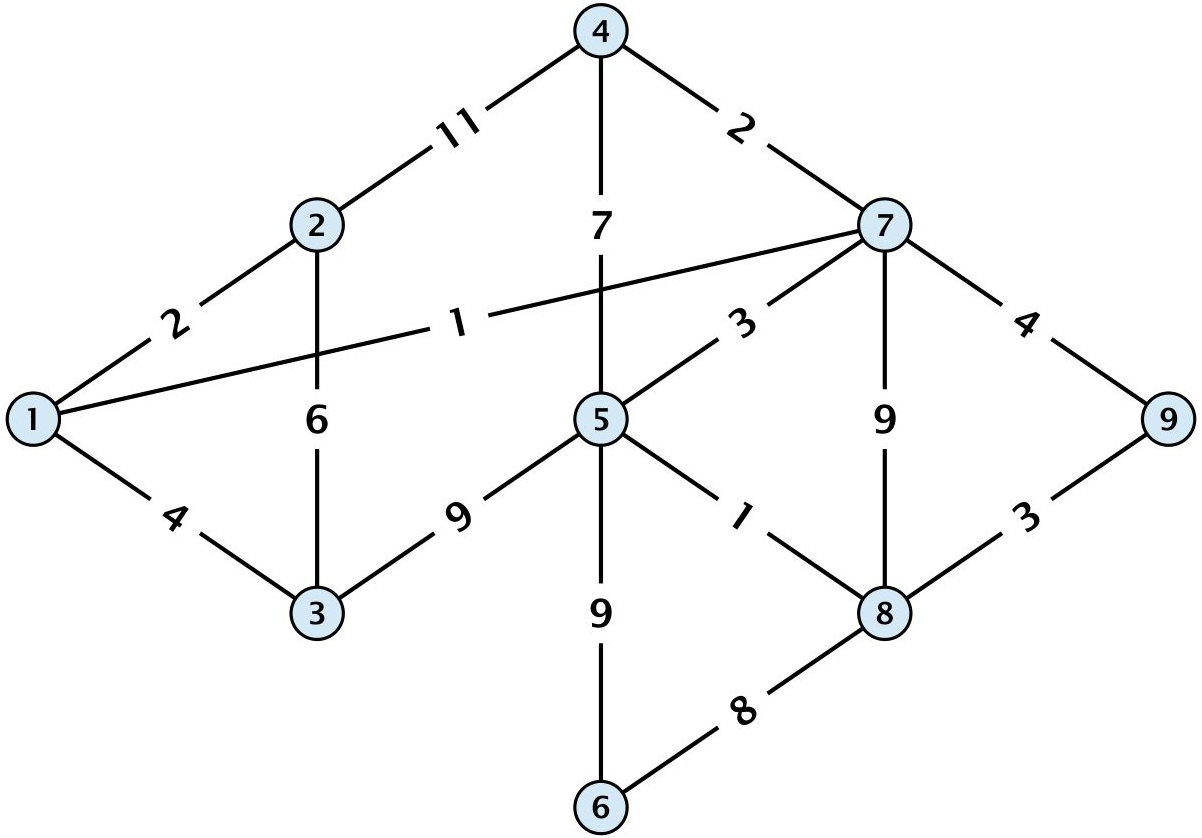
Then the algorithm performs $n - 1$ split-operations:

- 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.

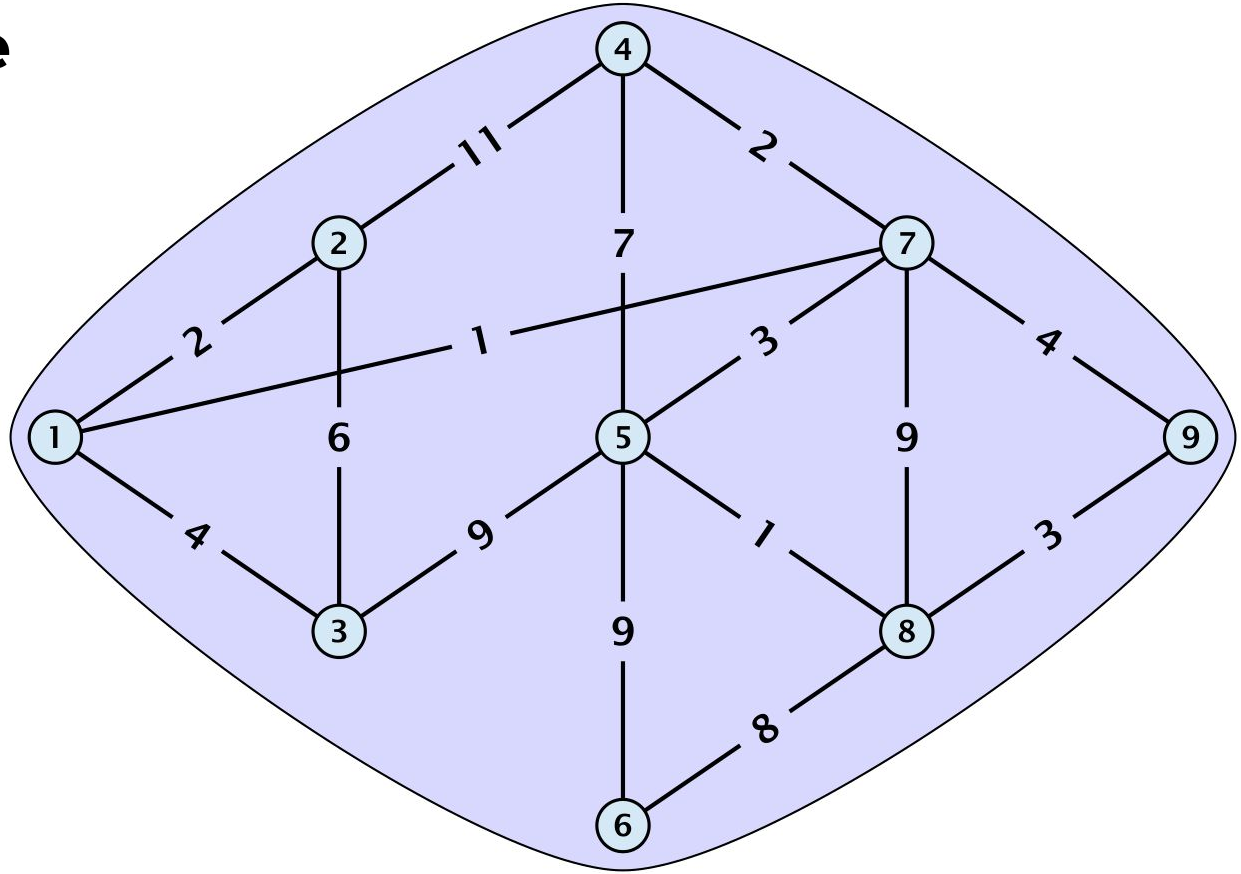
Details of the Split-operation

- 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 $X = S_i \cap A$ and $Y = S_i \cap B$ and add edge {X, Y} with capacity equal to that of the cut.
- Replace an edge $\{S_i, S_j\}$ with $\{X, S_j\}$ if $S_j \in A$ and by $\{Y, S_j\}$ if $S_j \in B$.

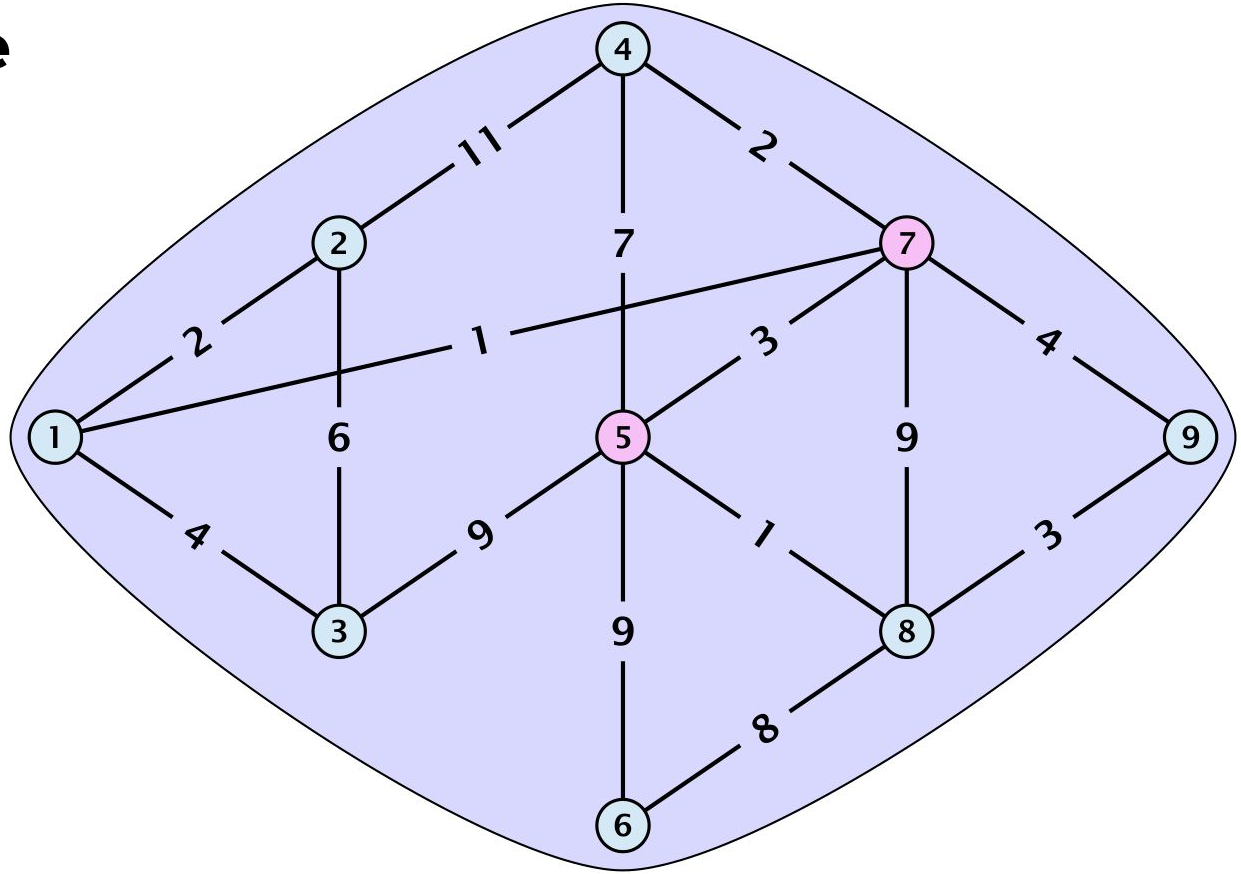
Example



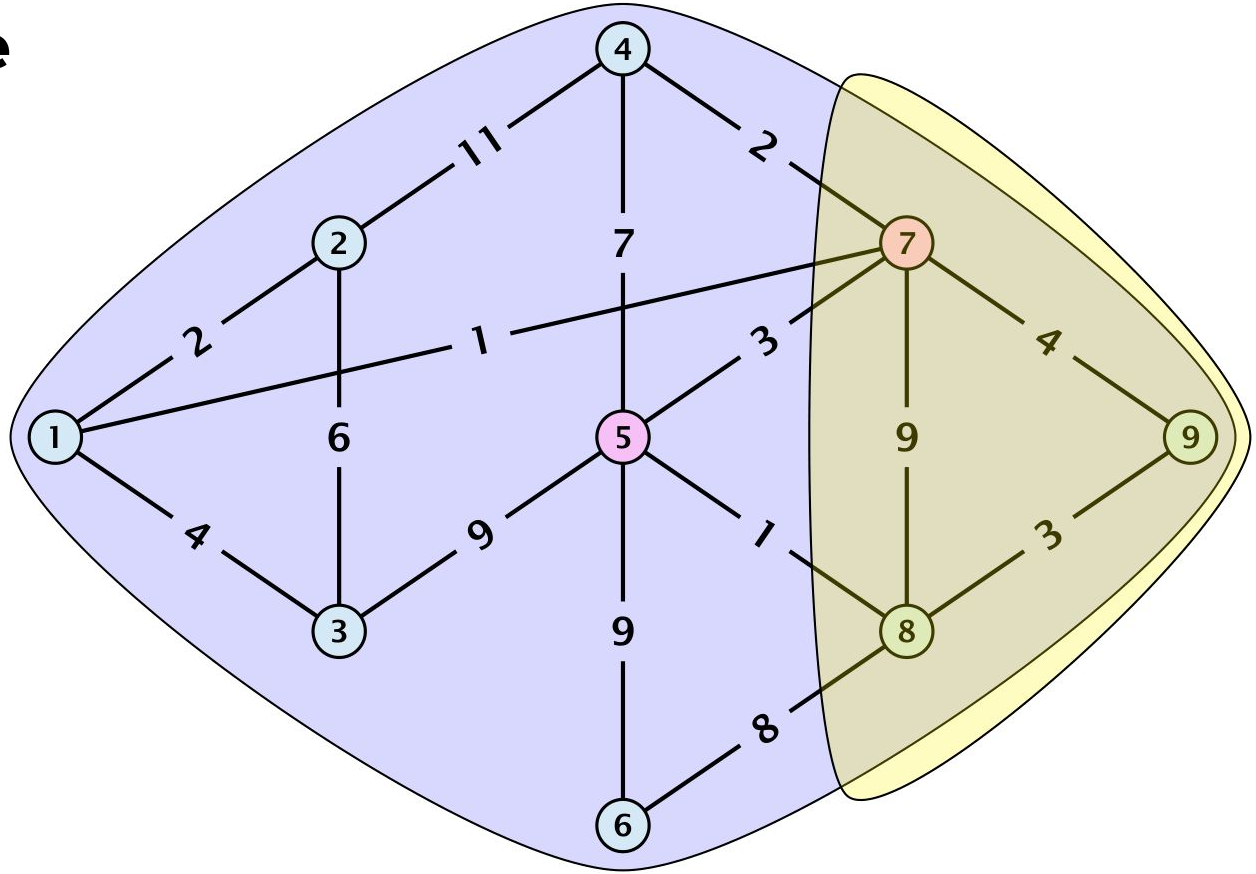
Example



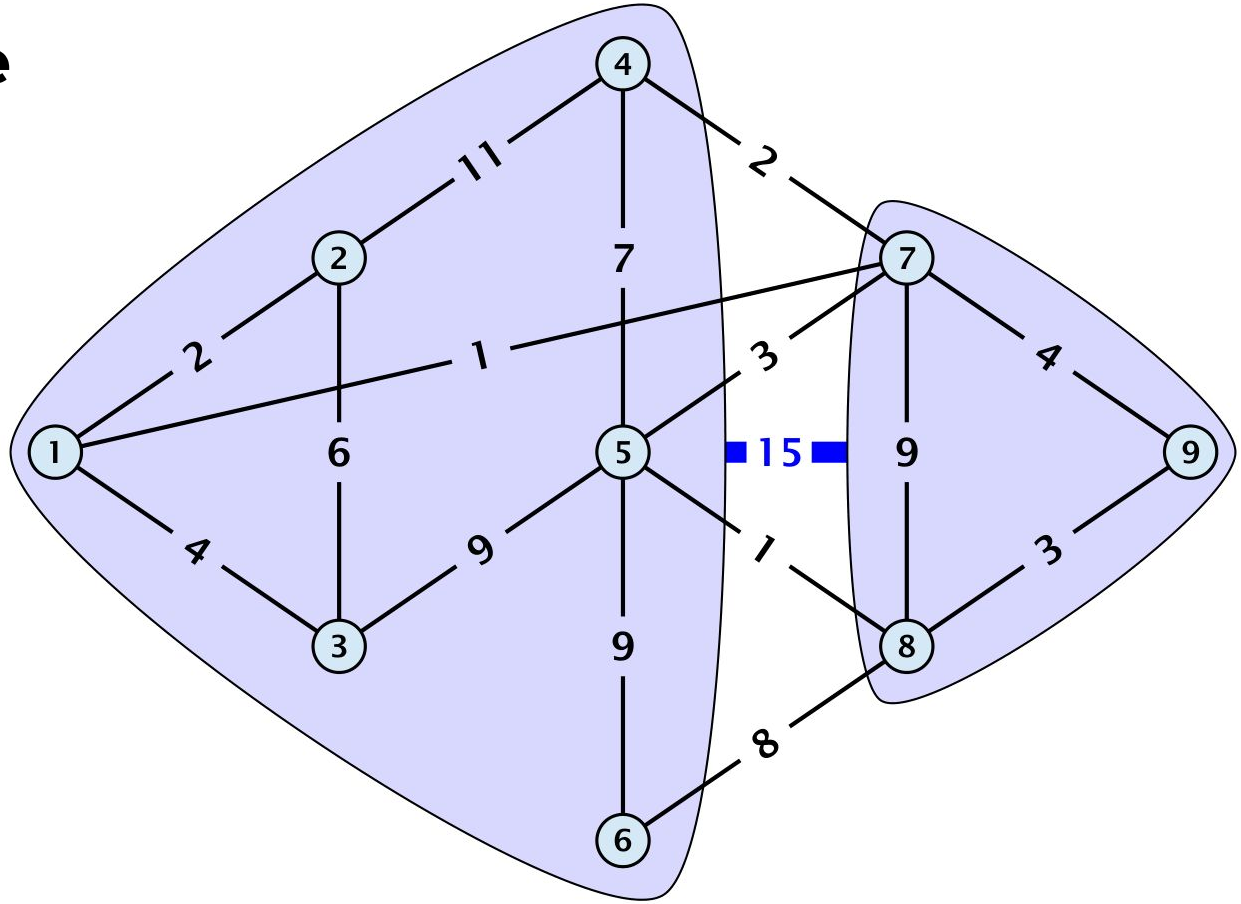
Example



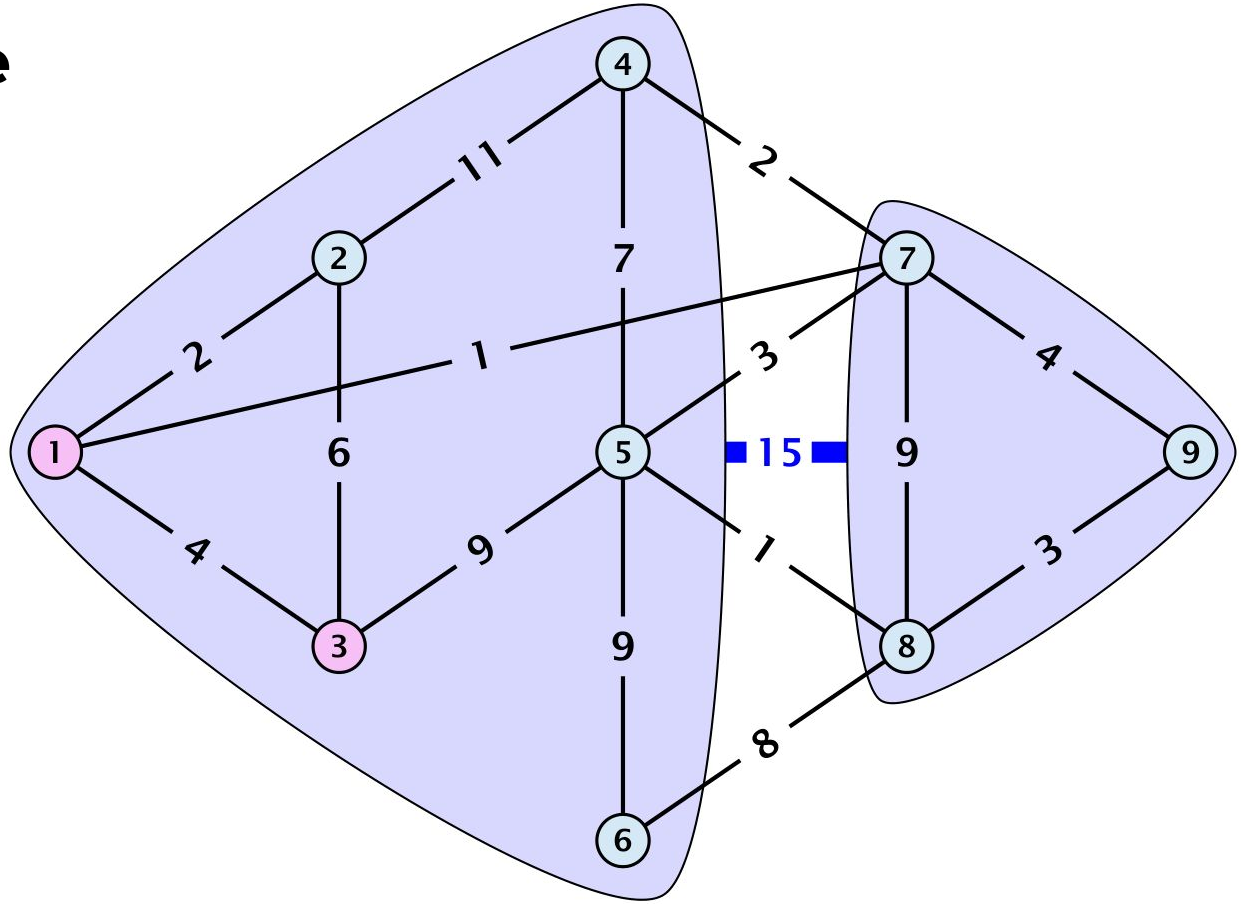
Example



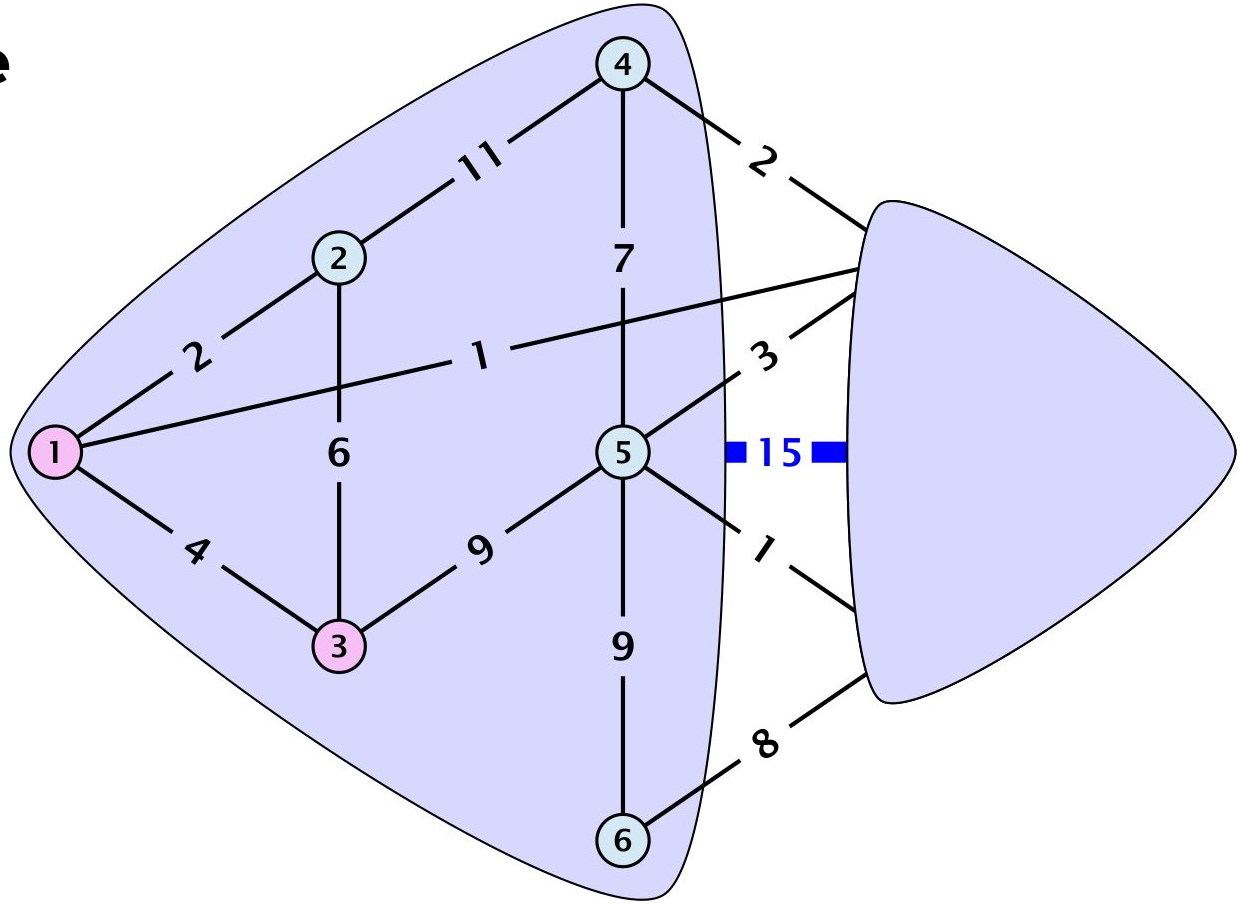
Example



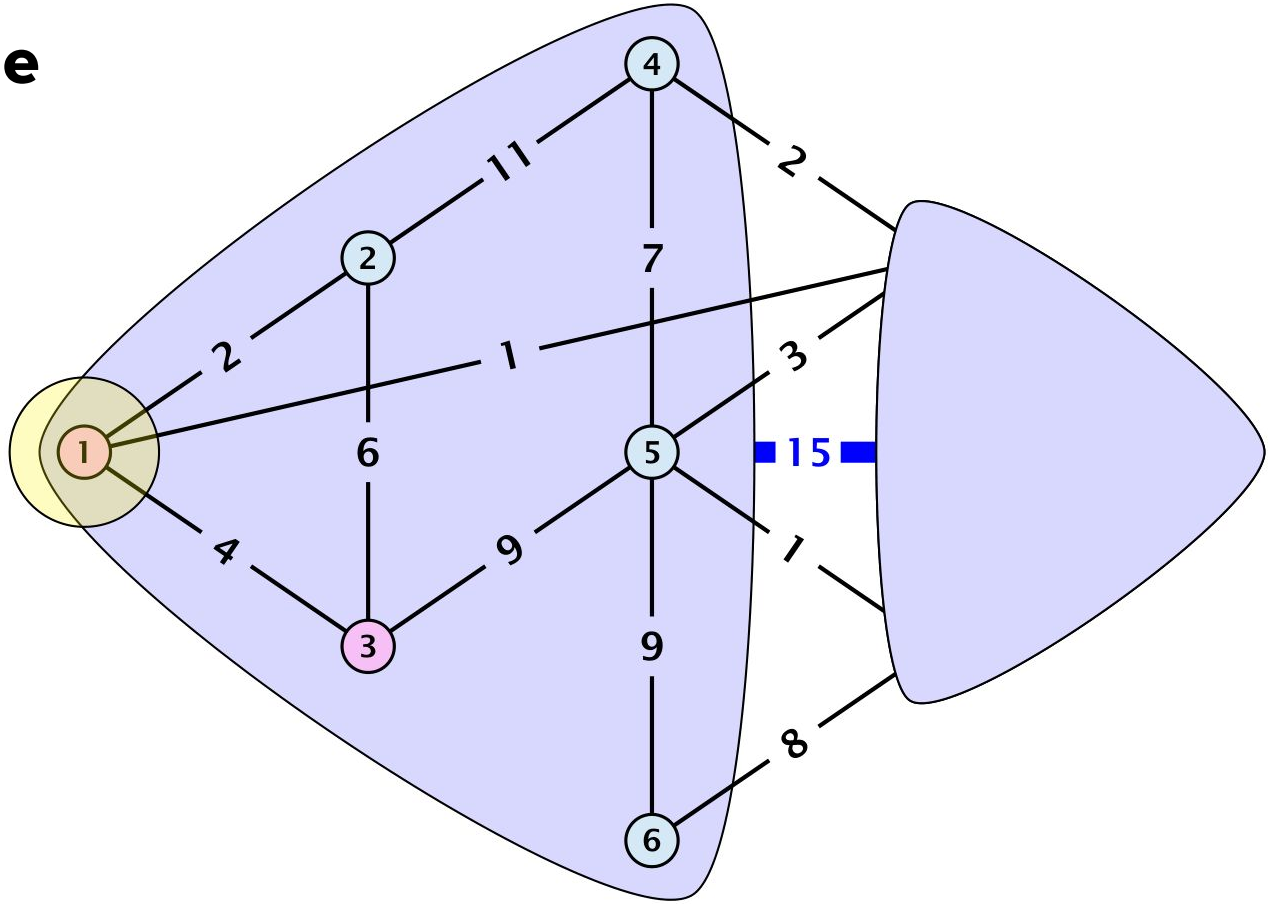
Example



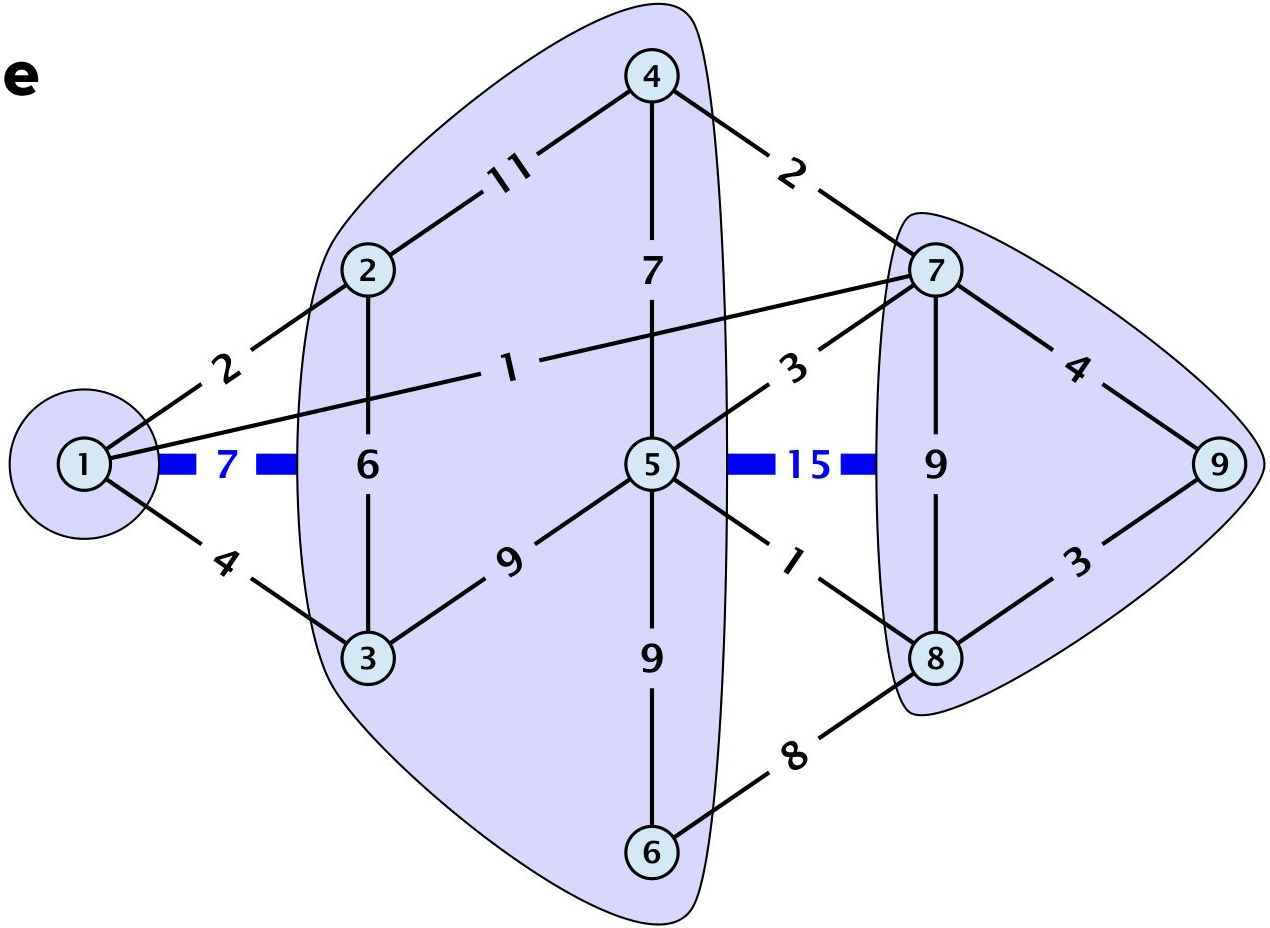
Example



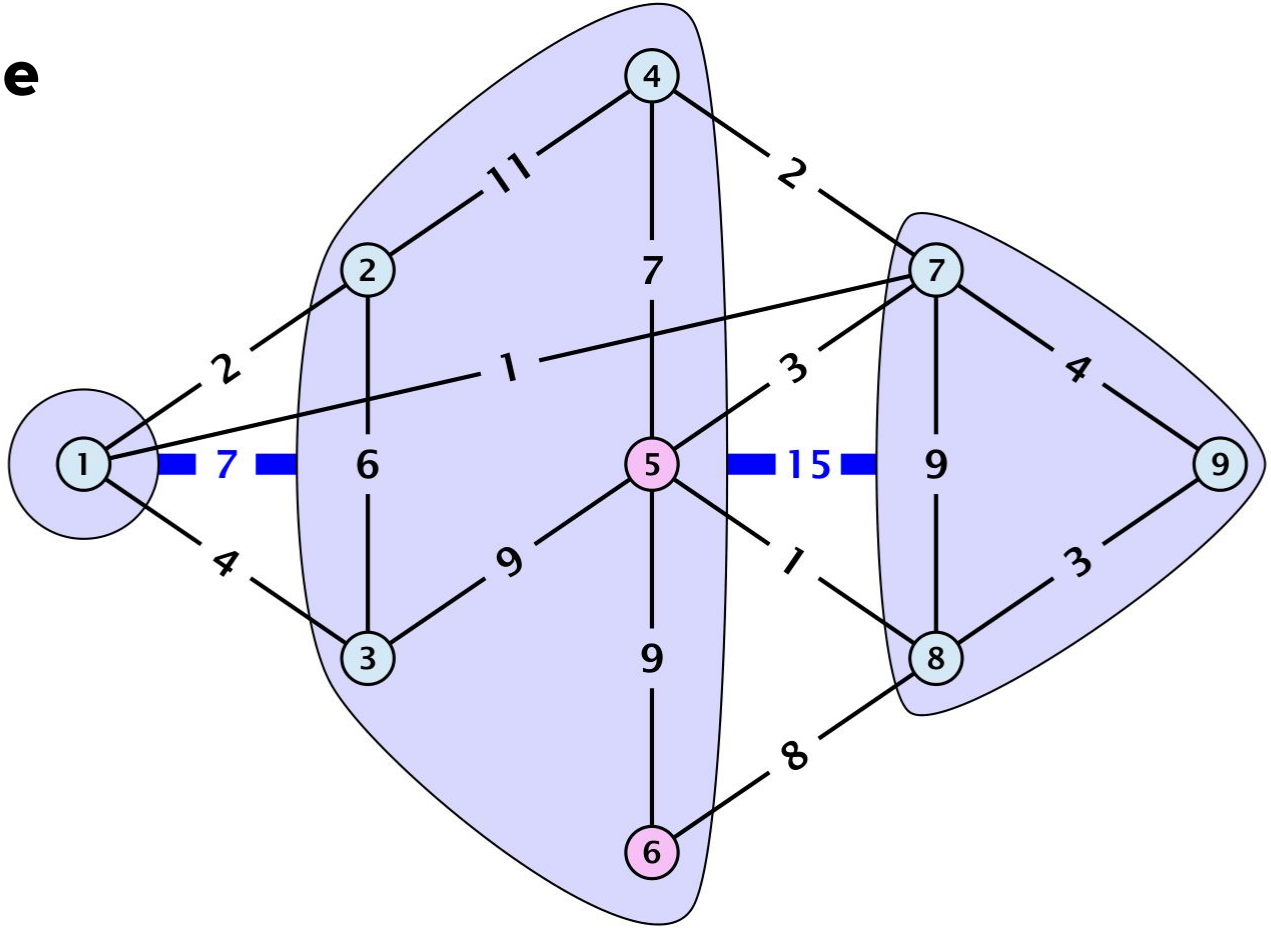
Example



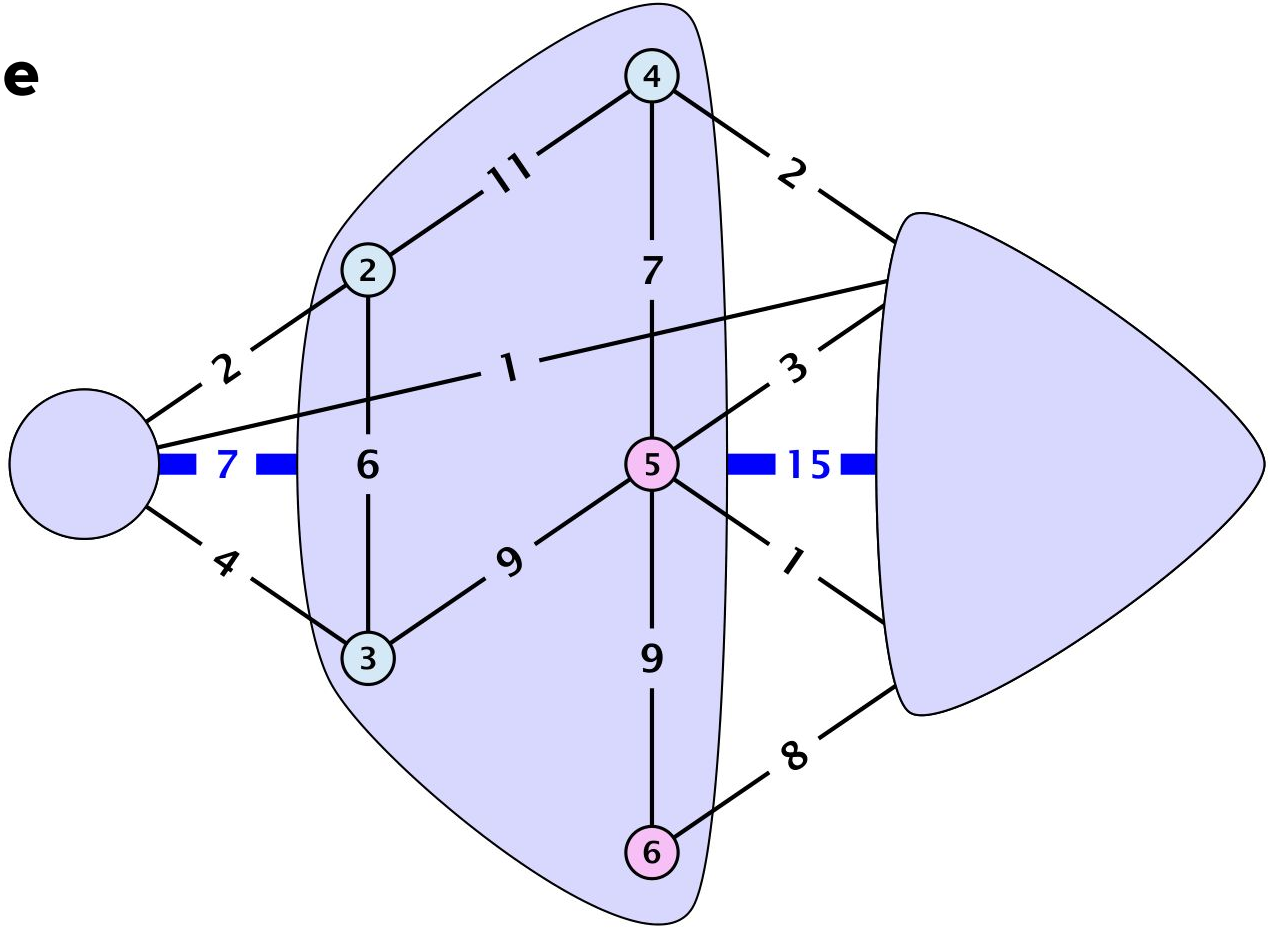
Example



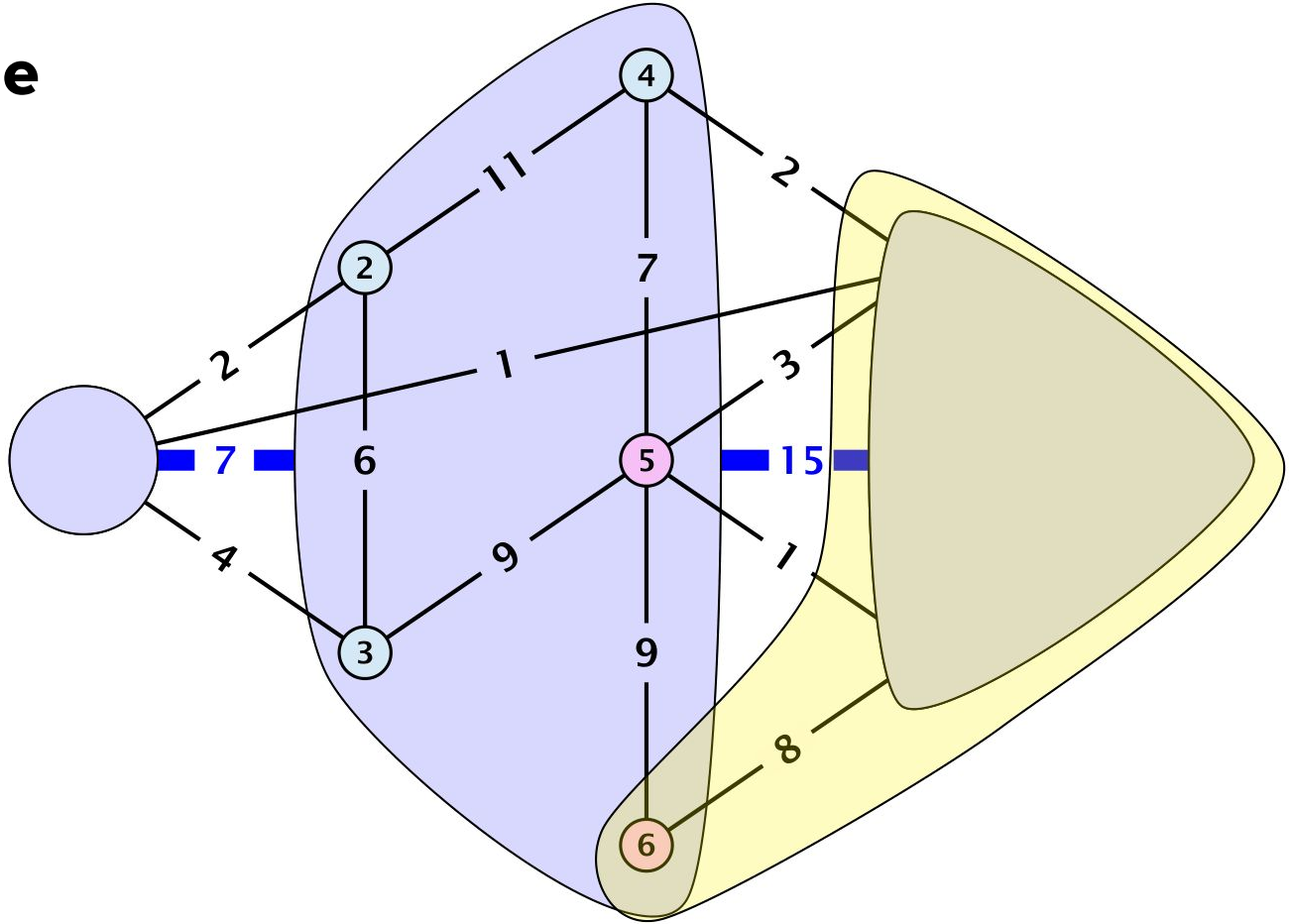
Example



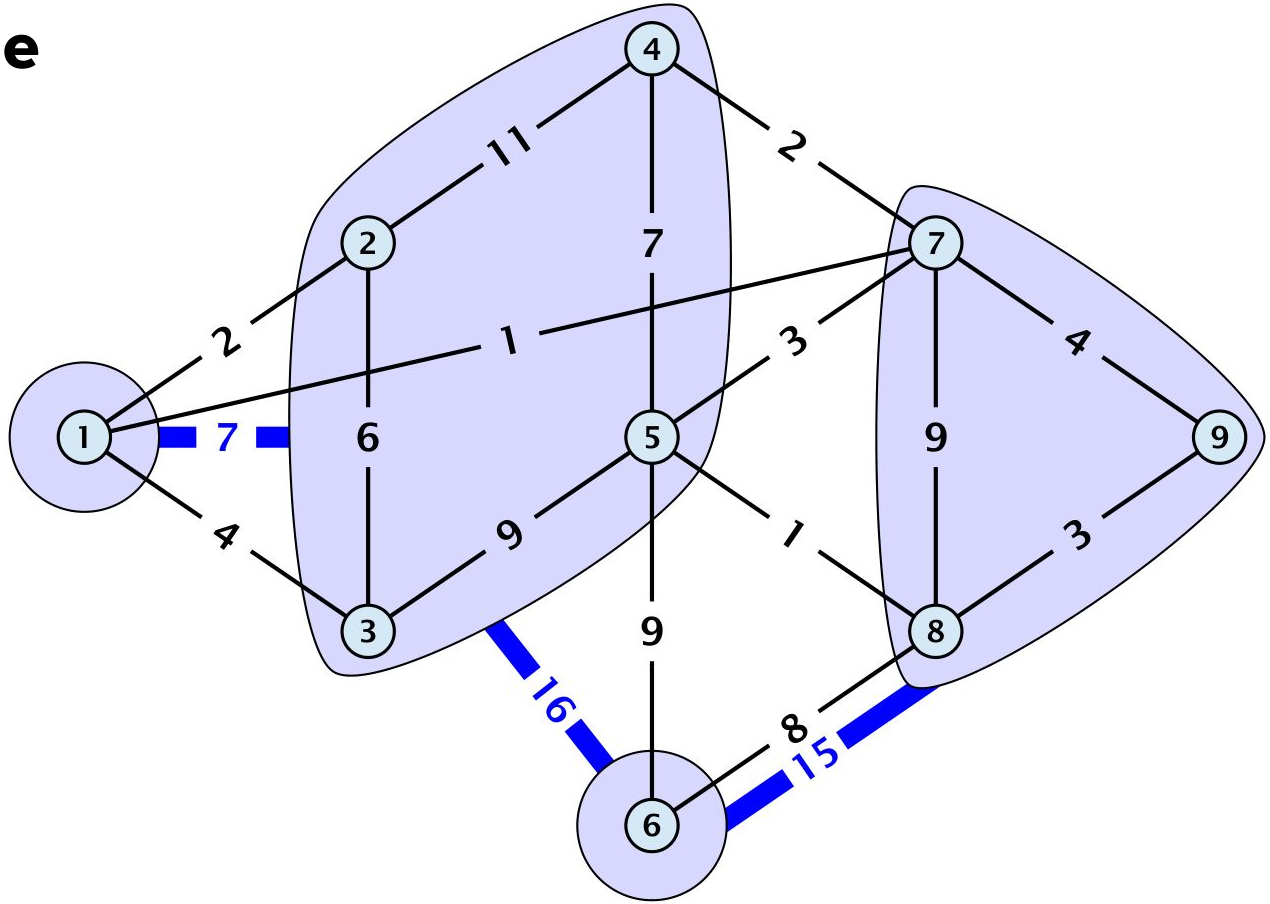
Example



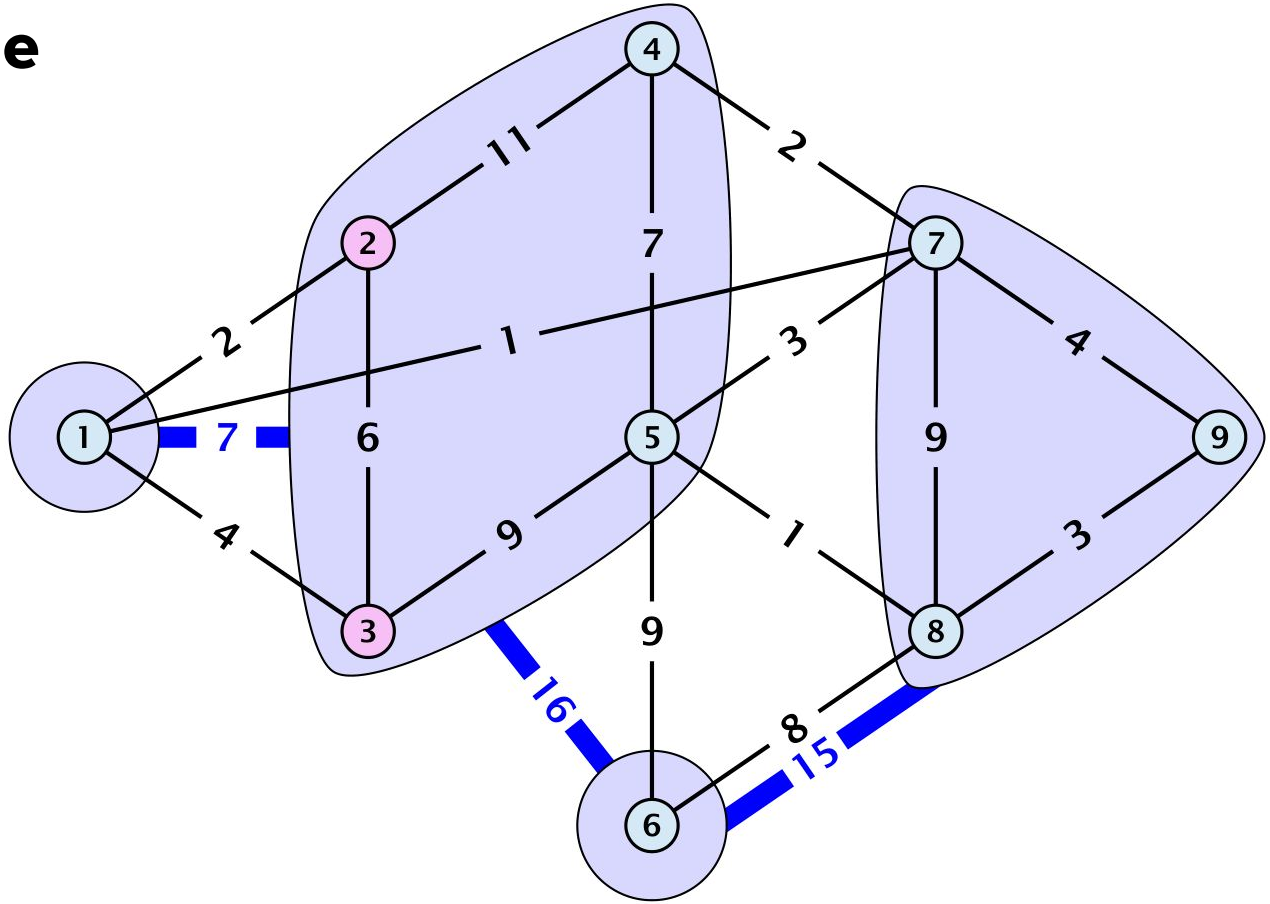
Example



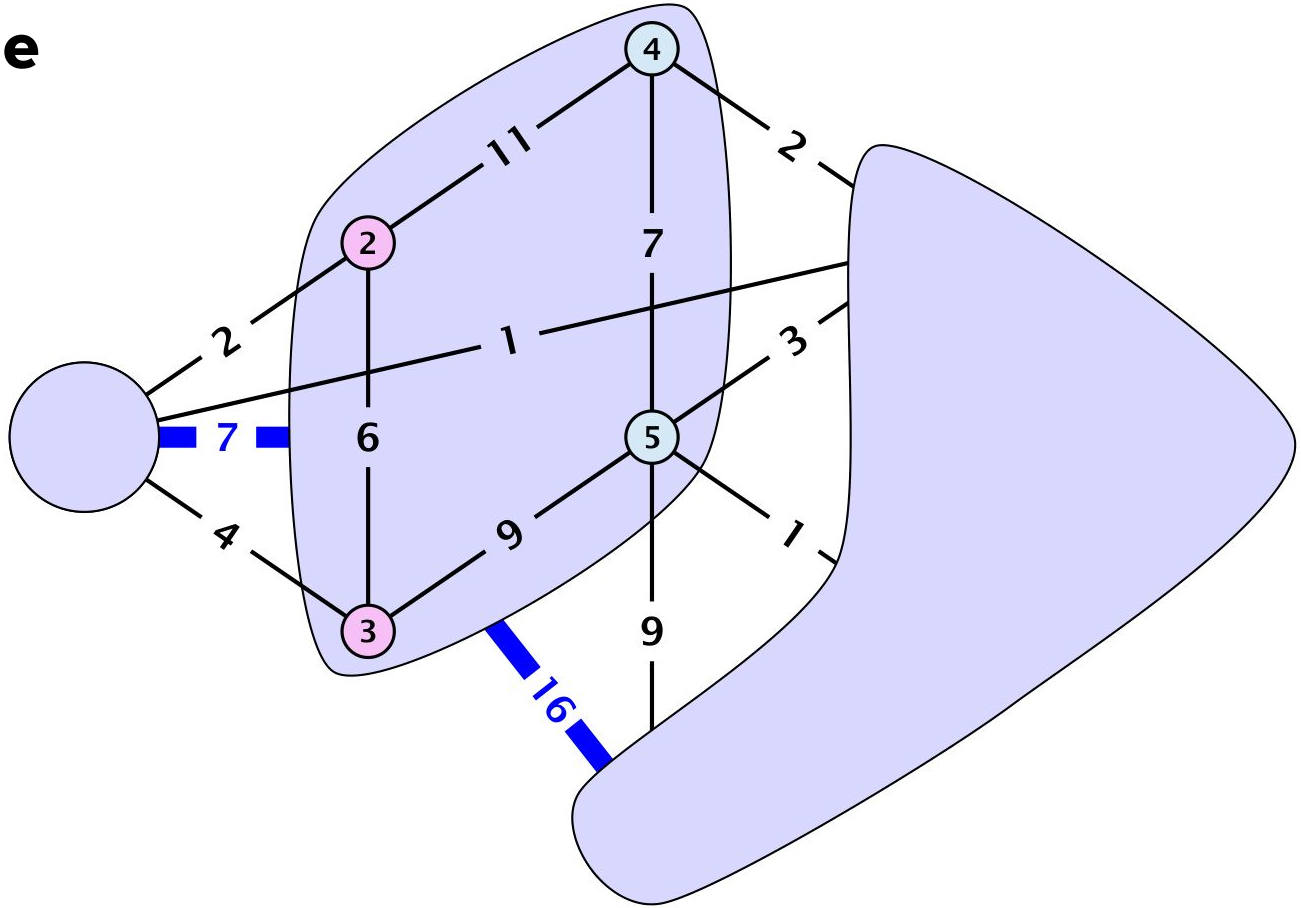
Example



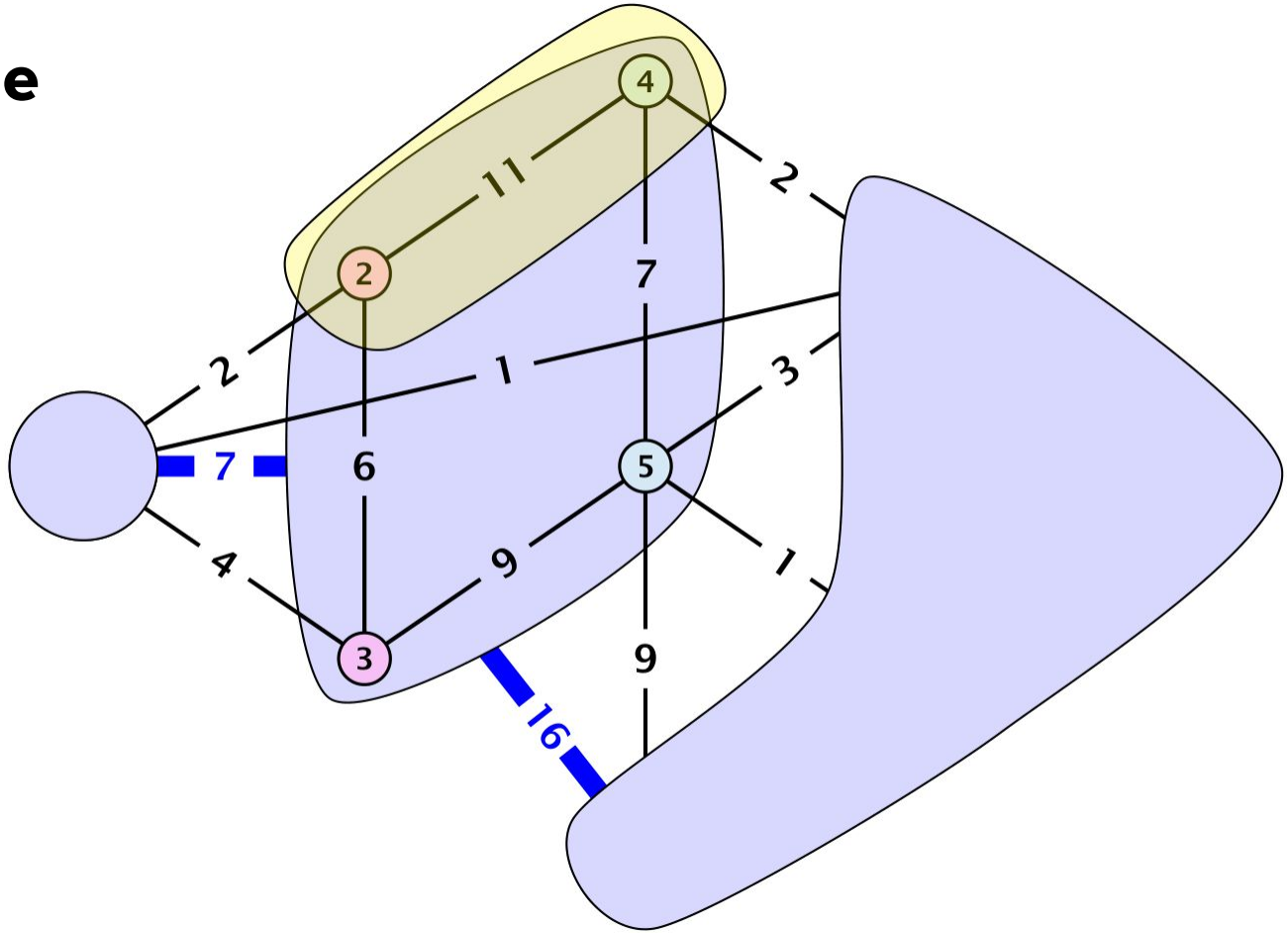
Example



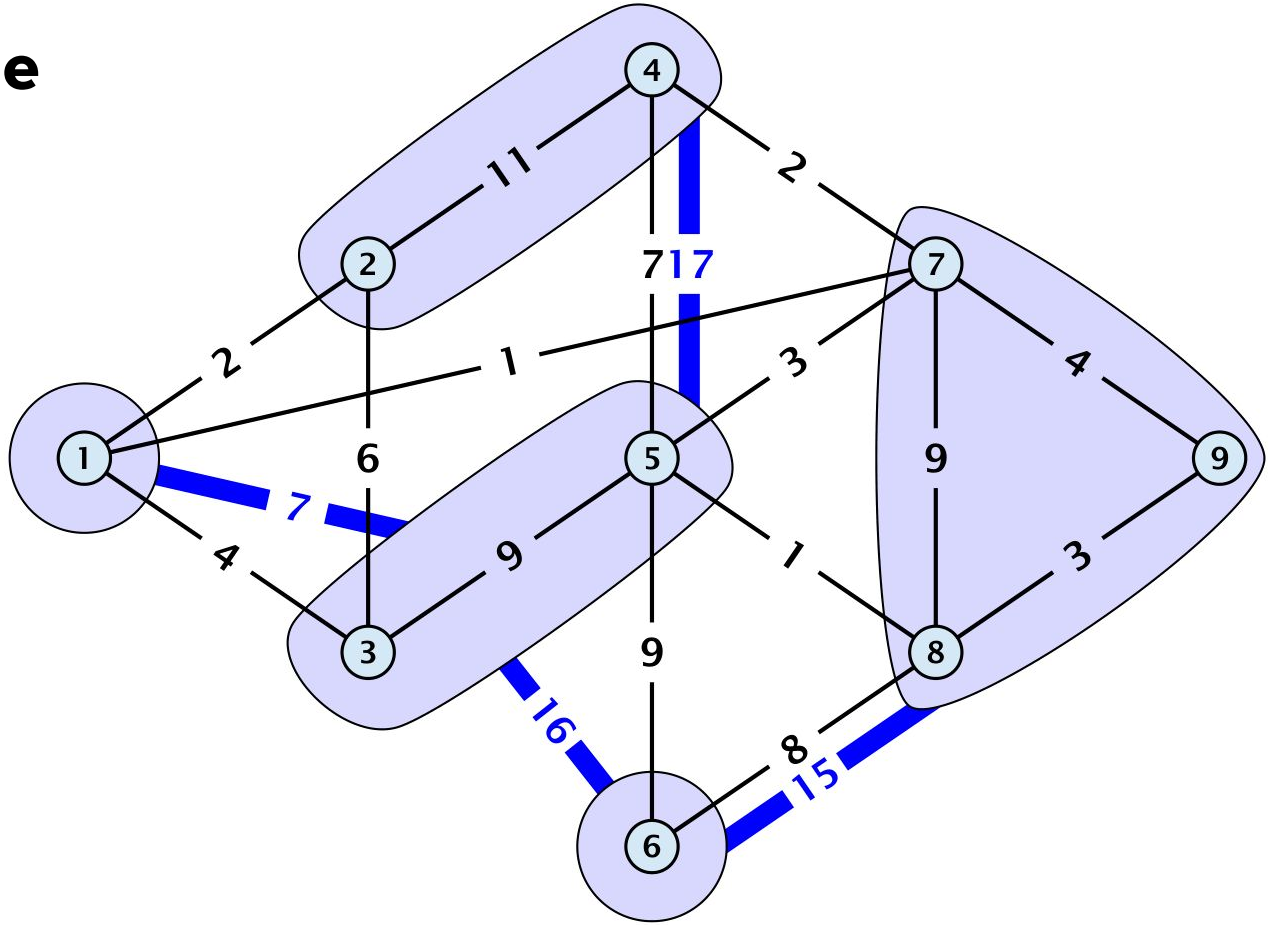
Example



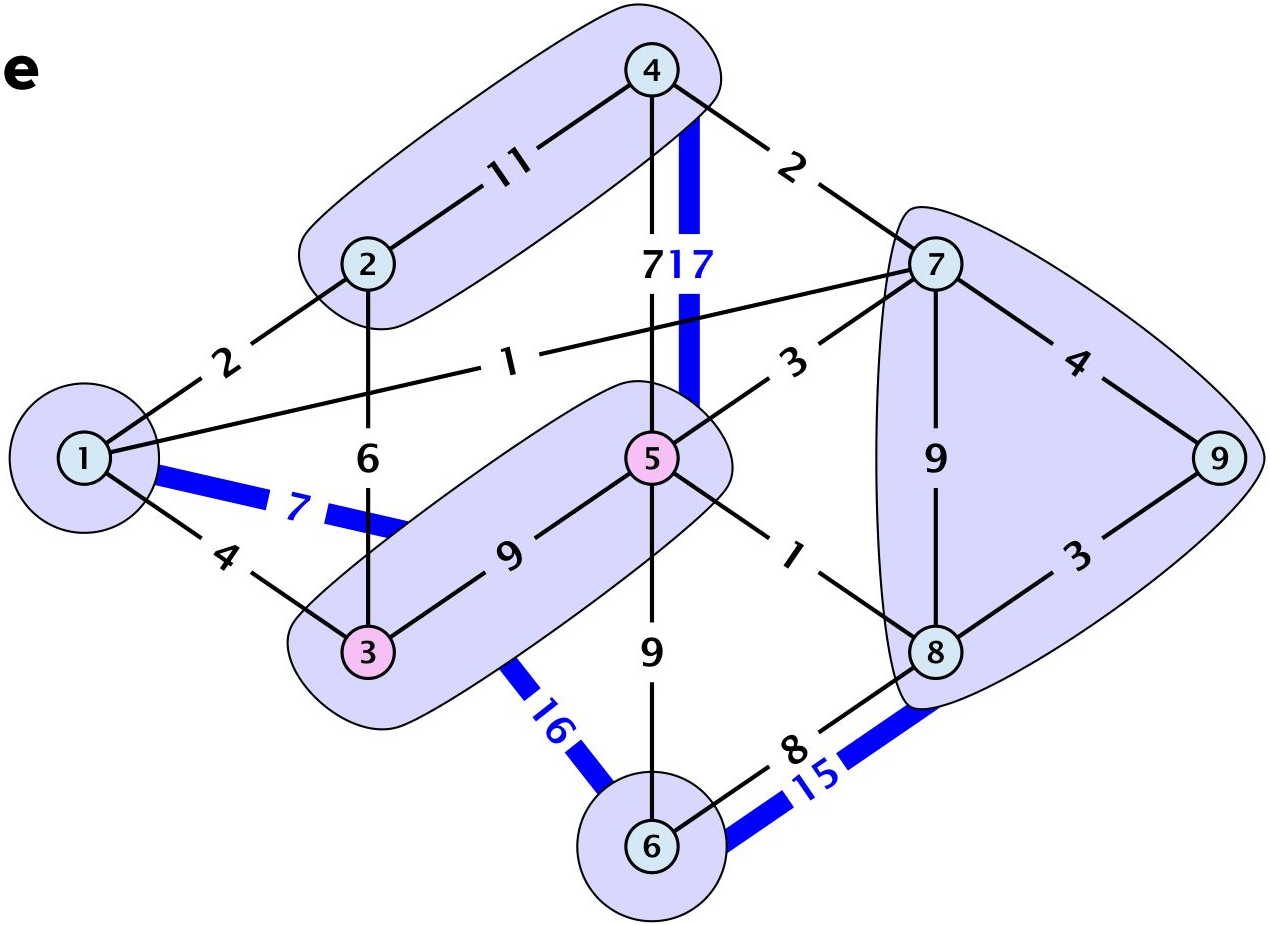
Example



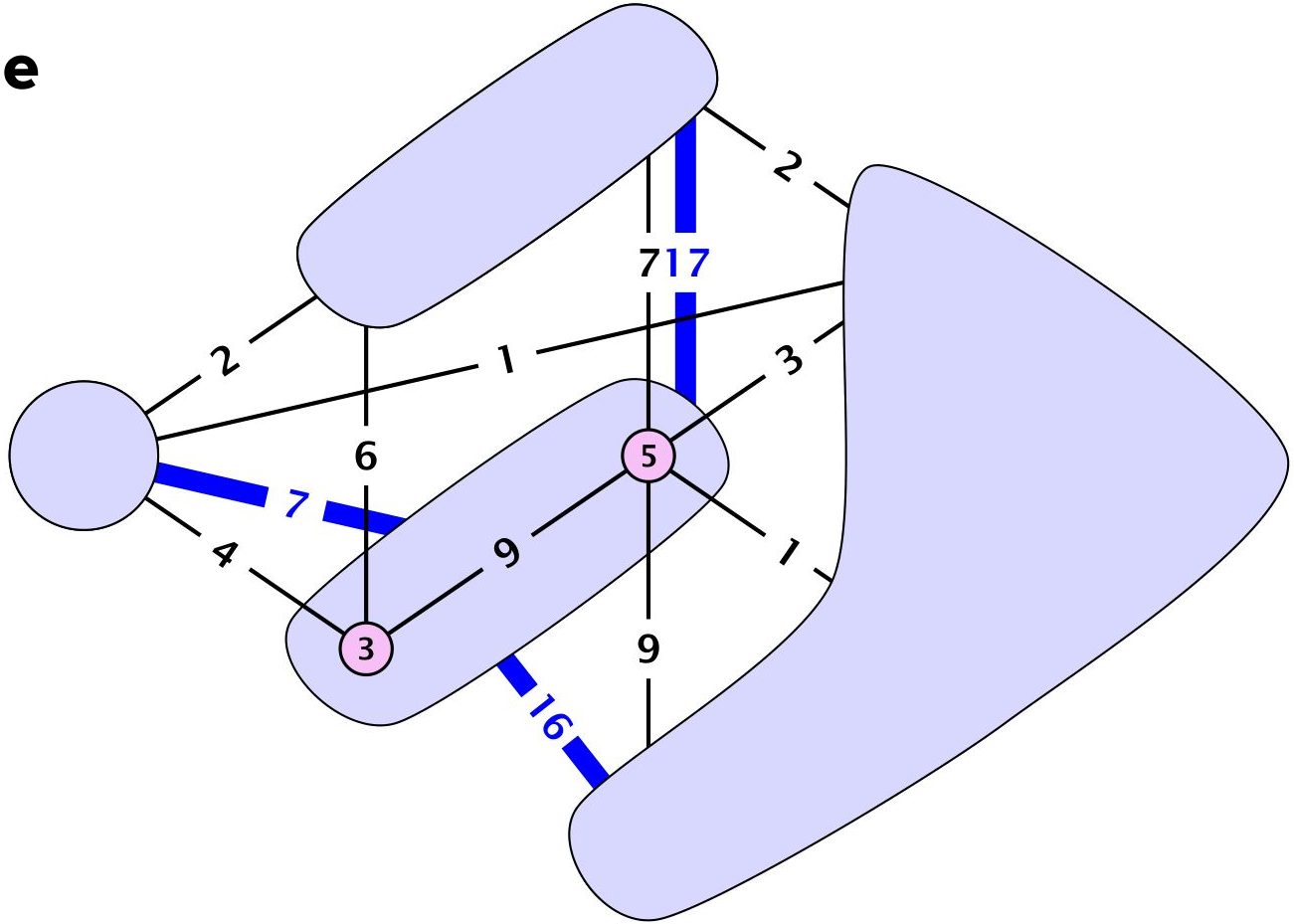
Example



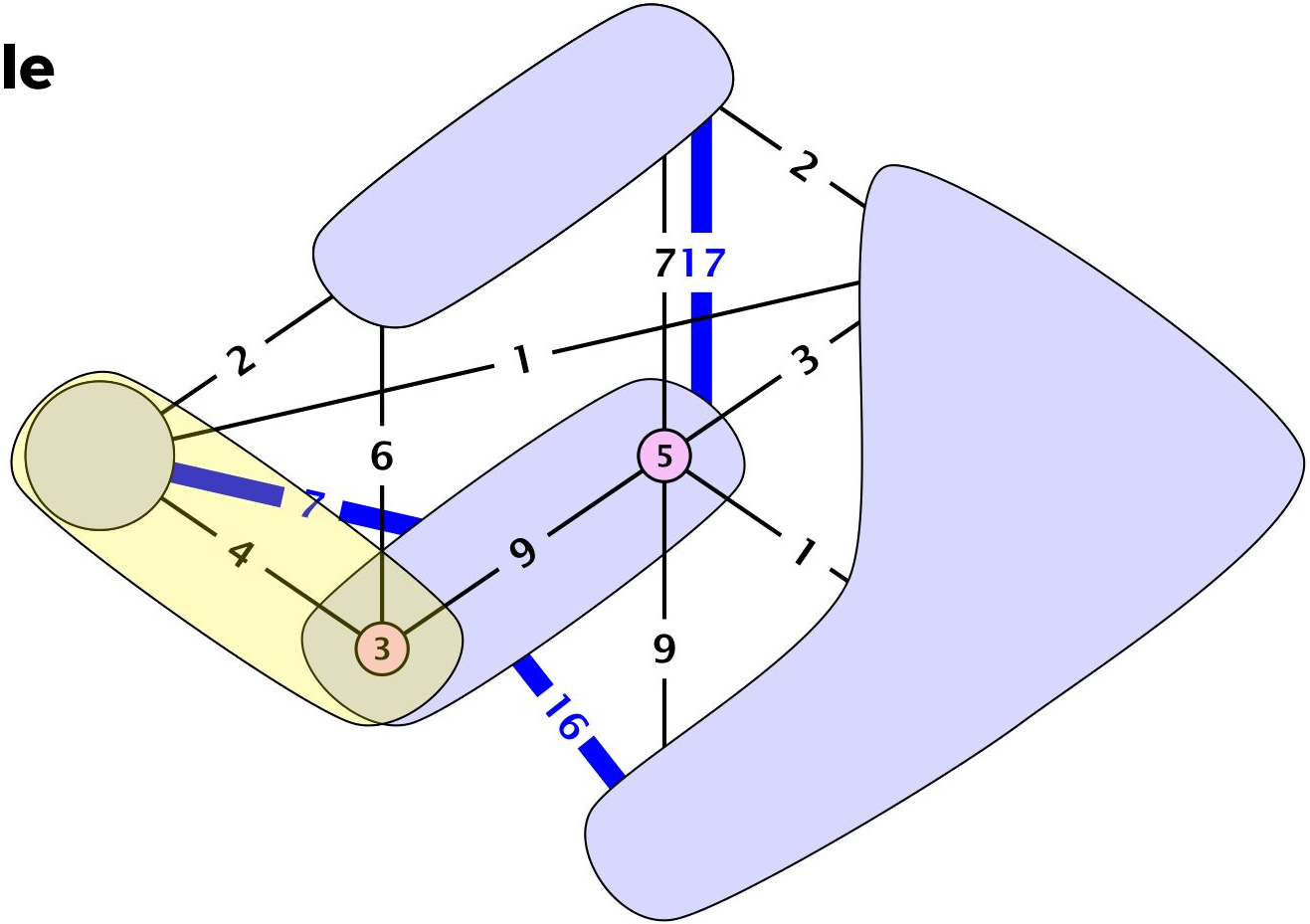
Example



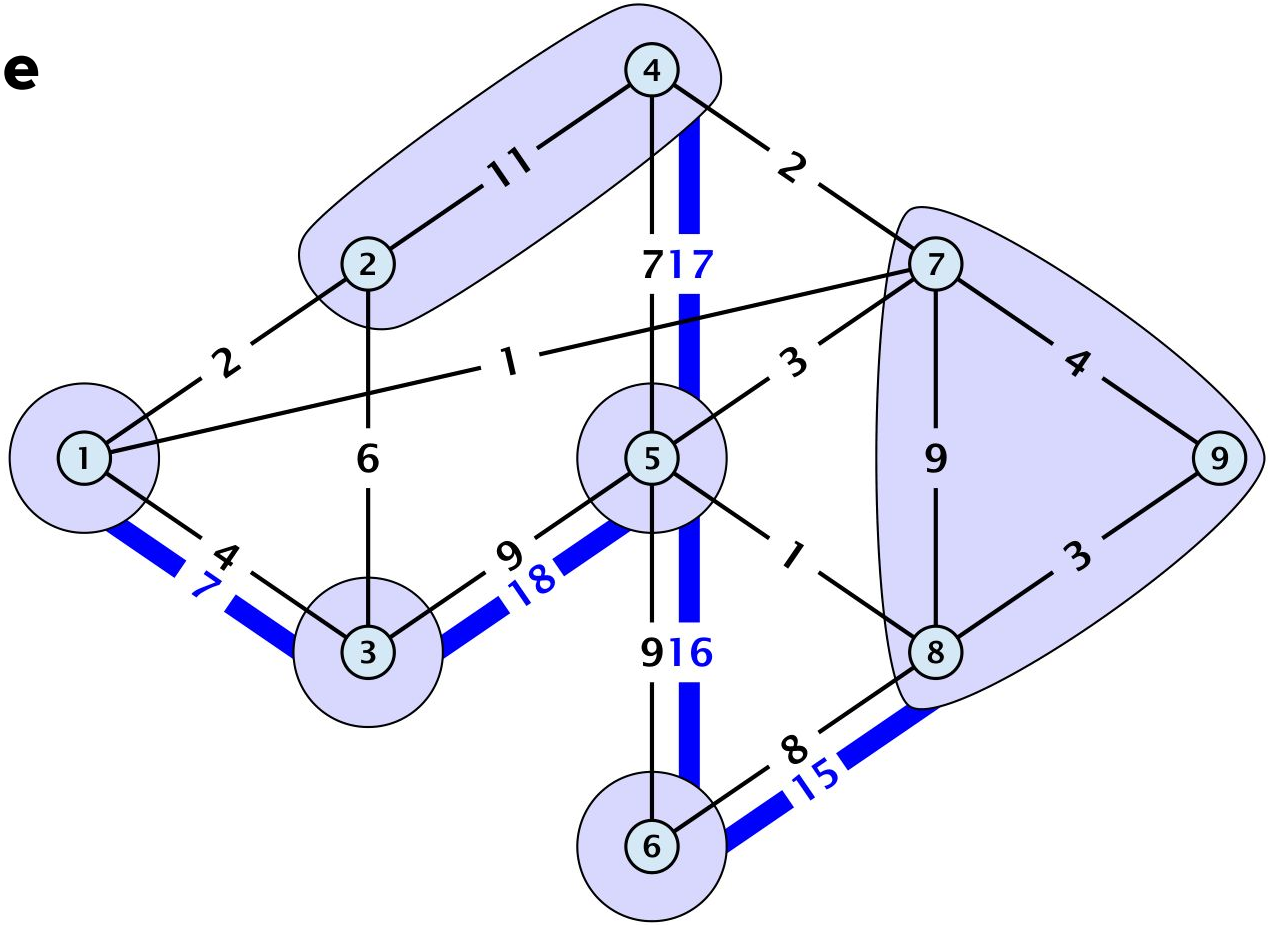
Example



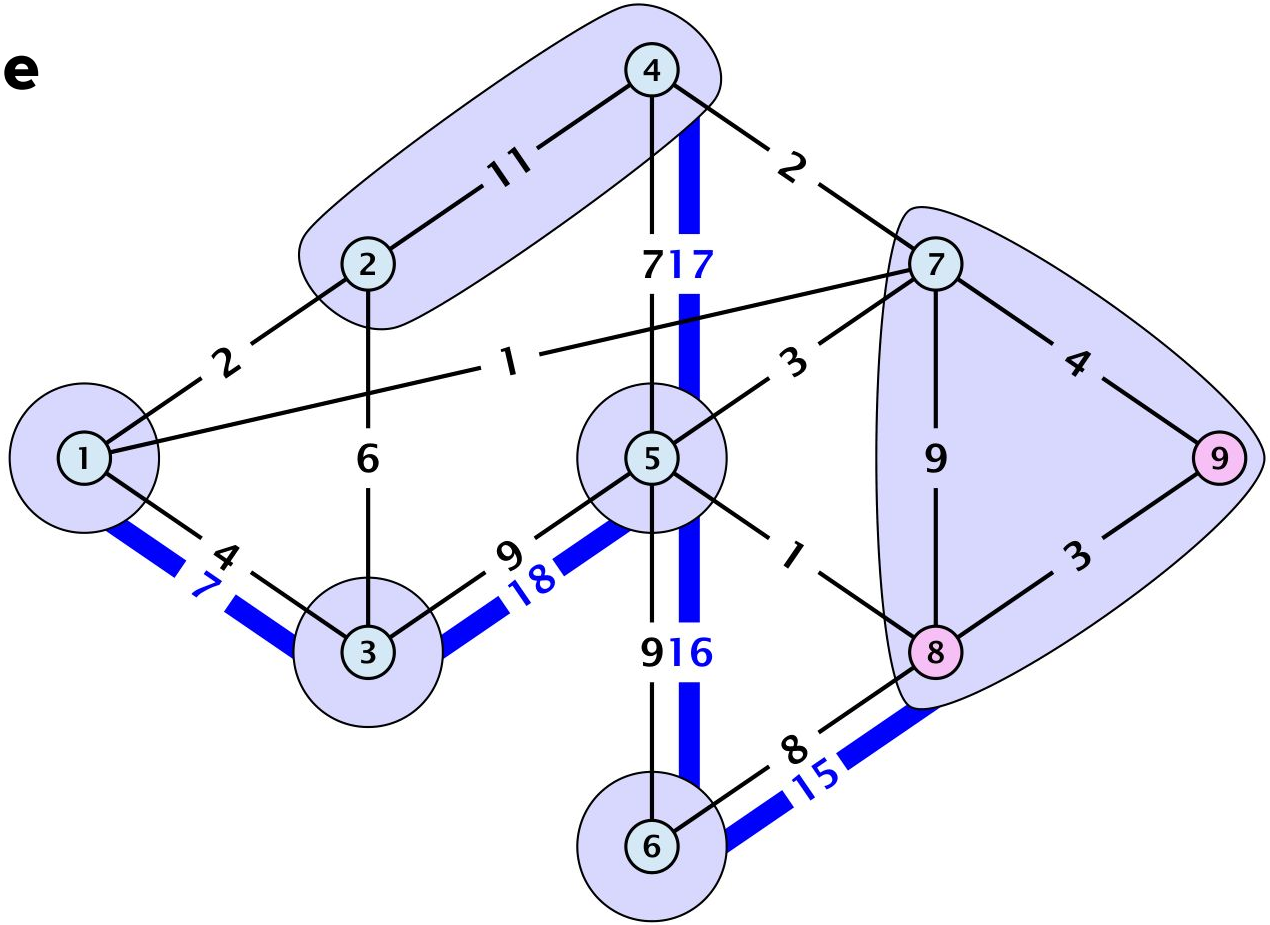
Example



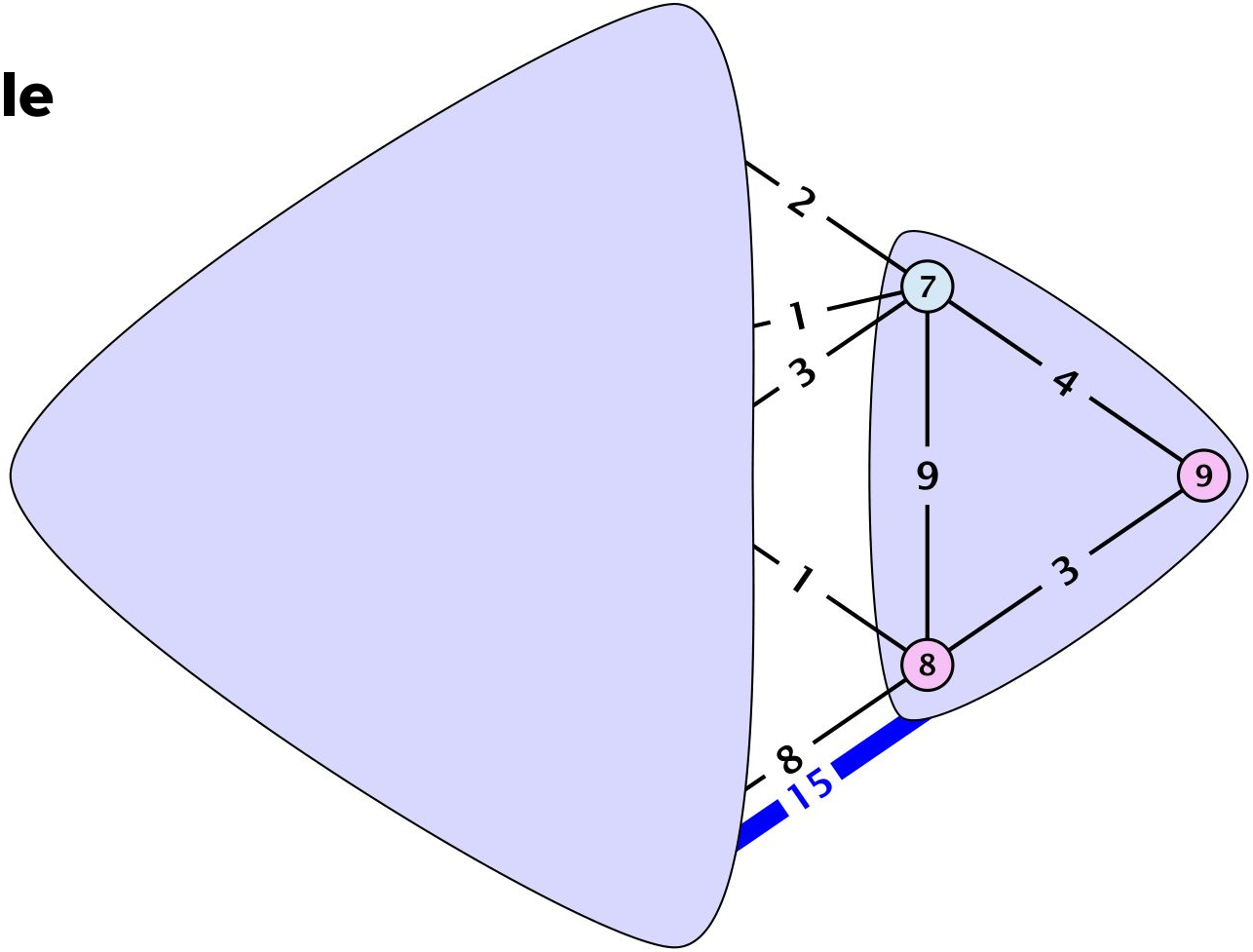
Example



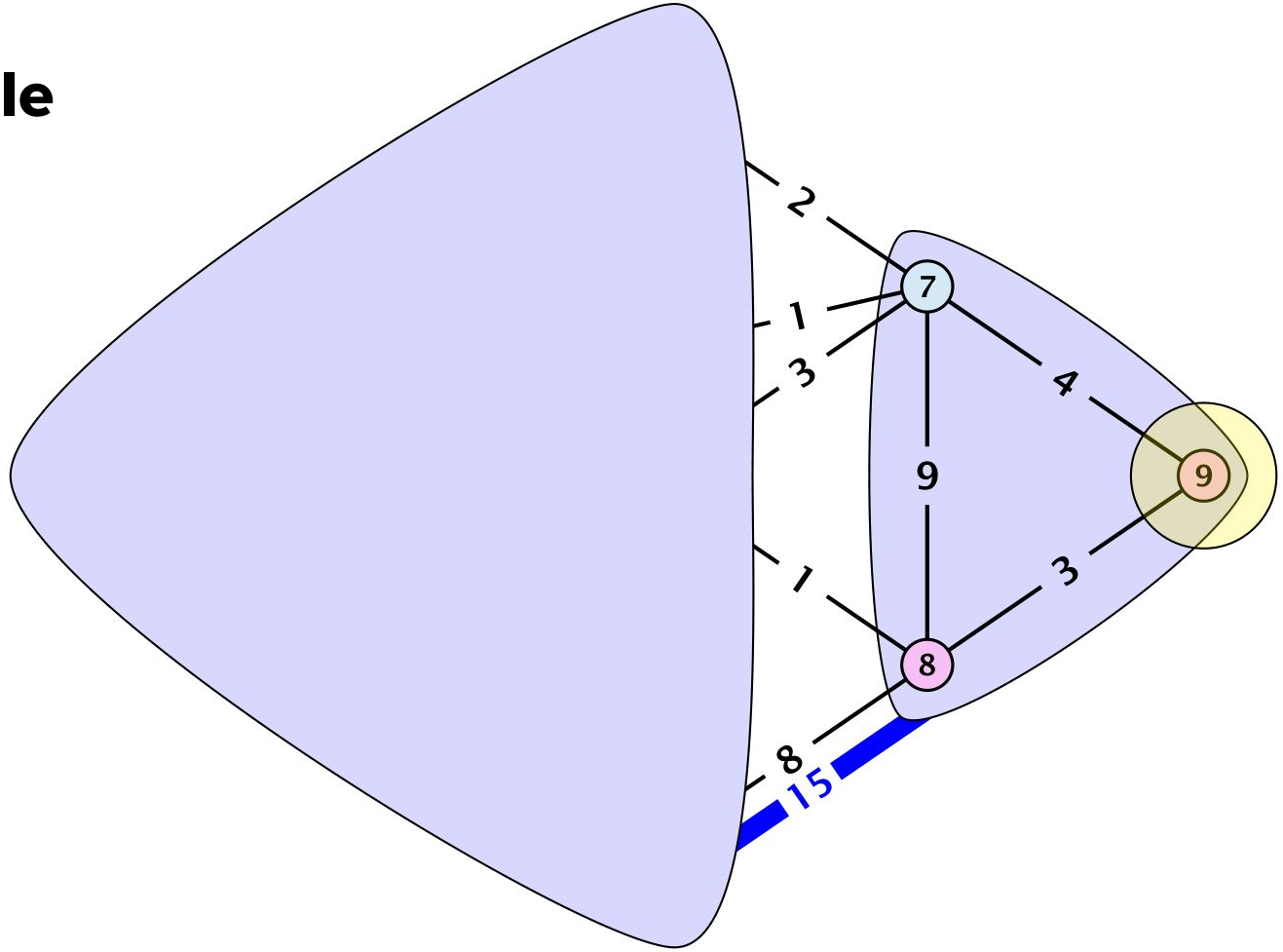
Example



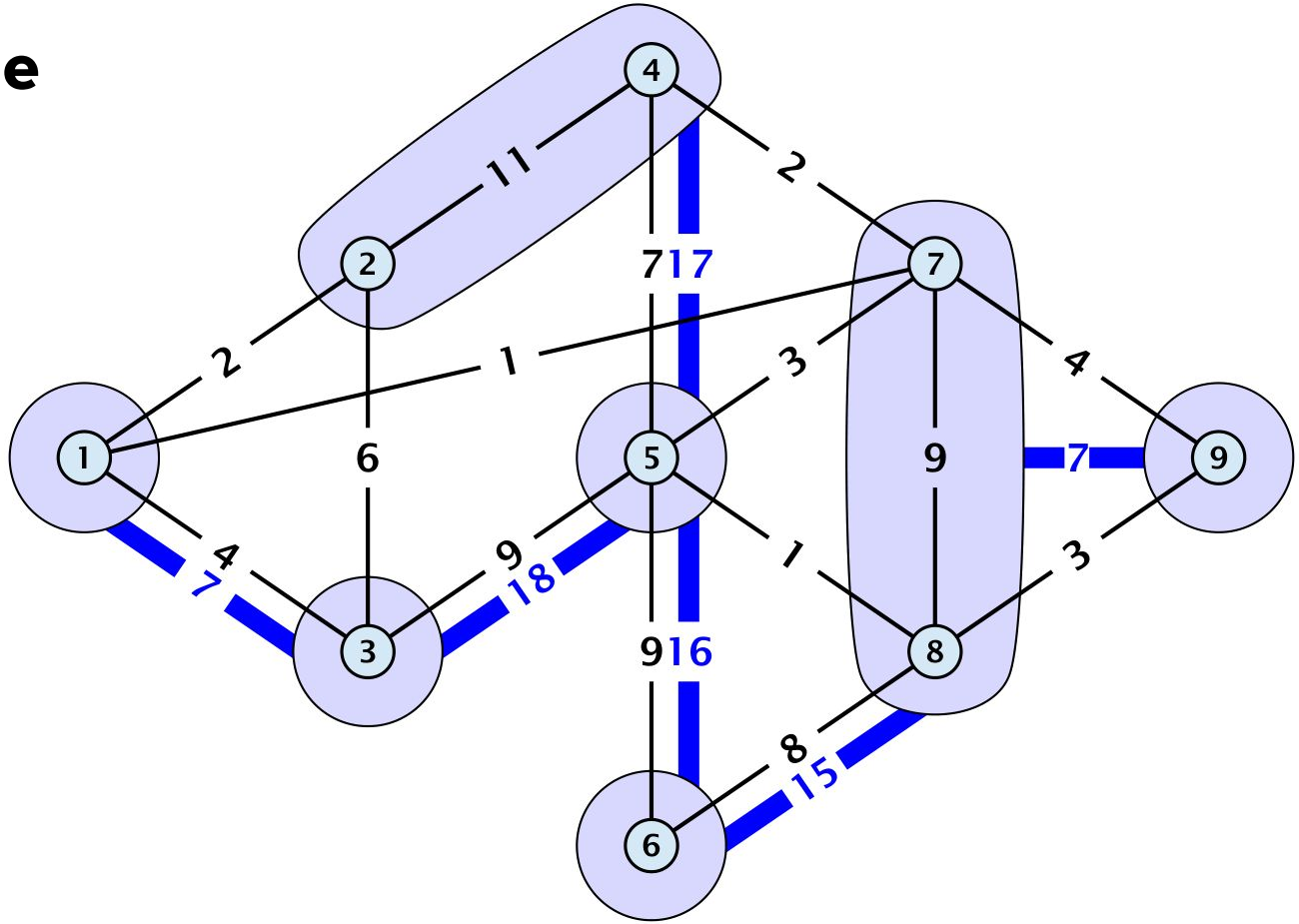
Example



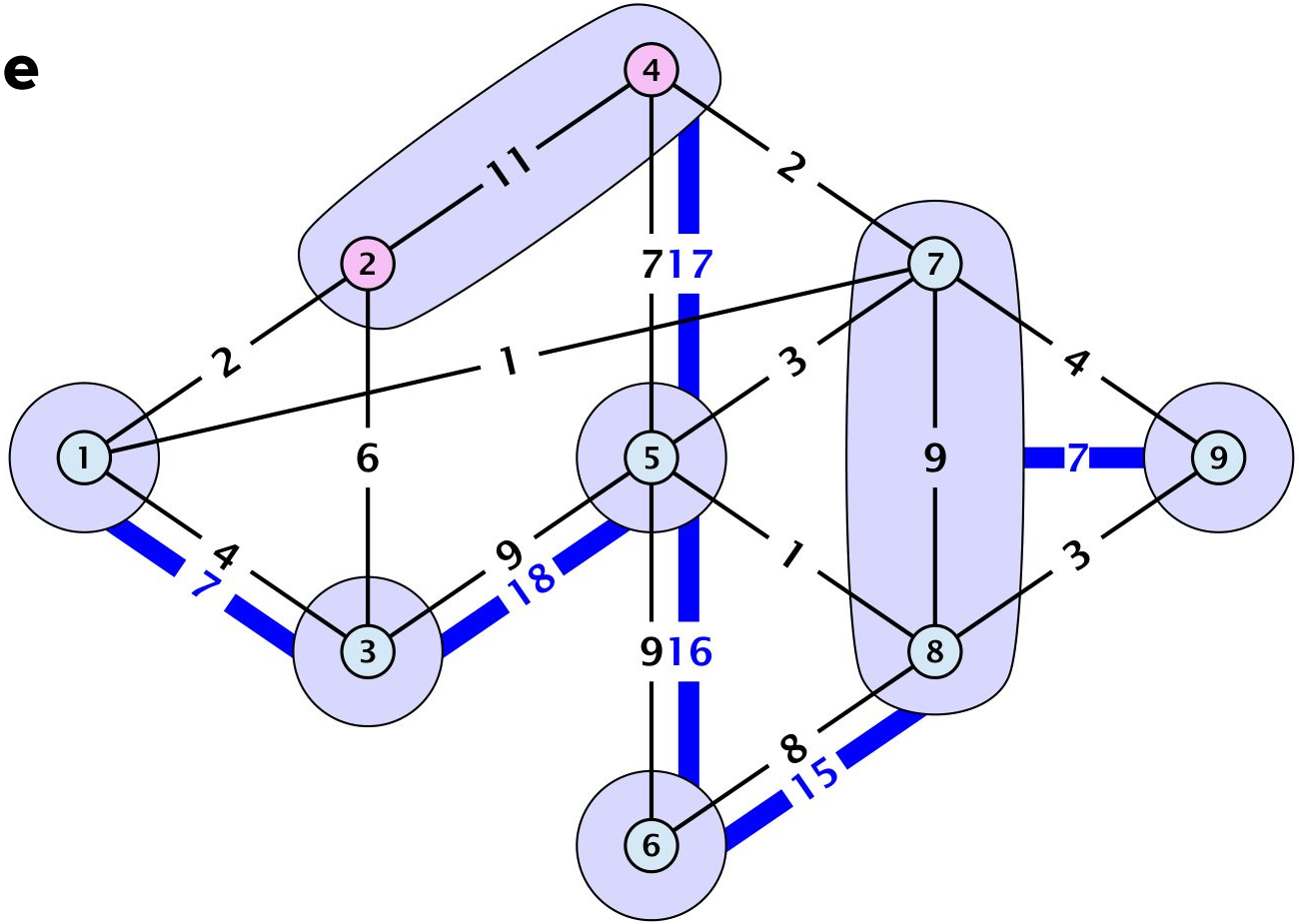
Example



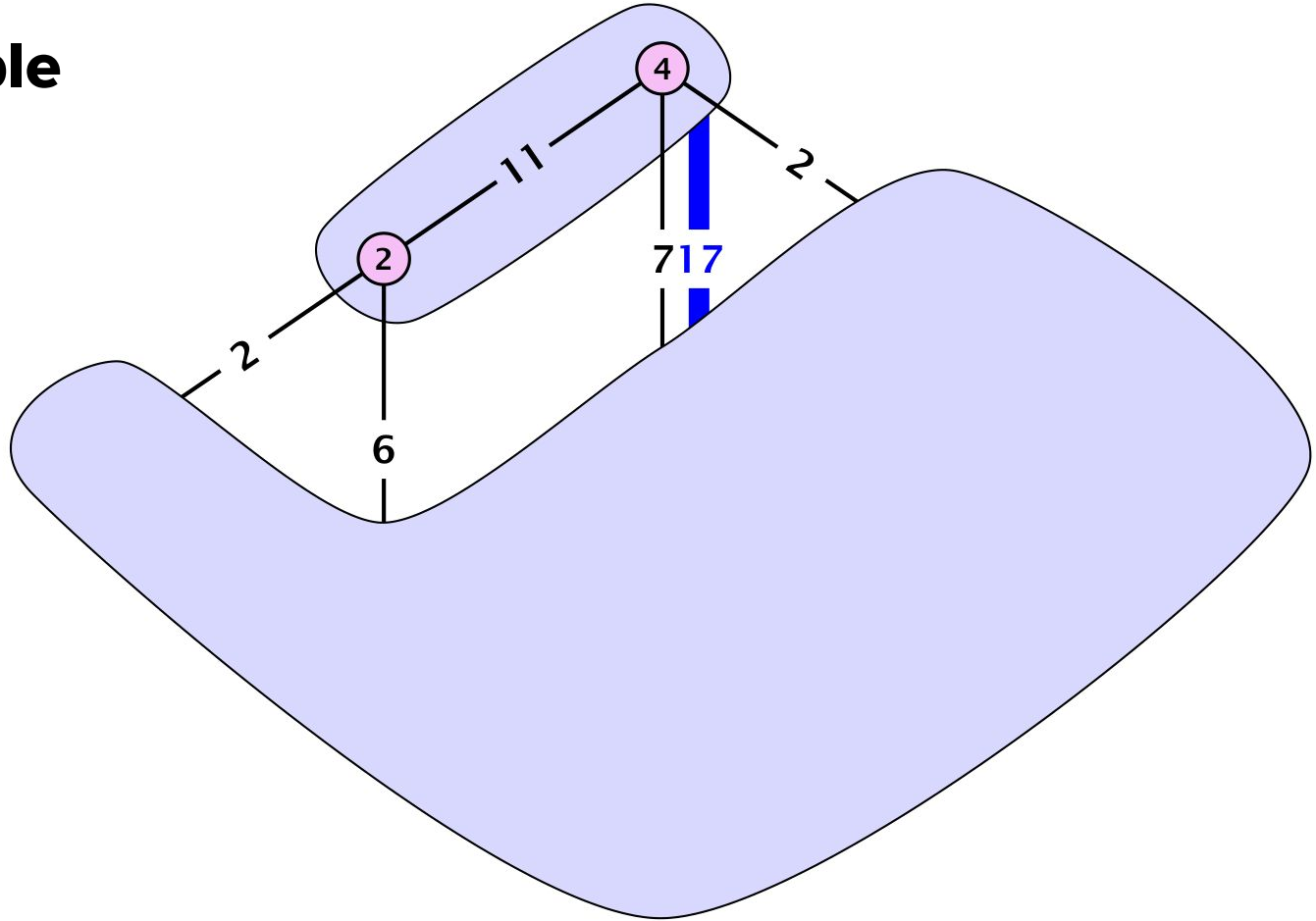
Example



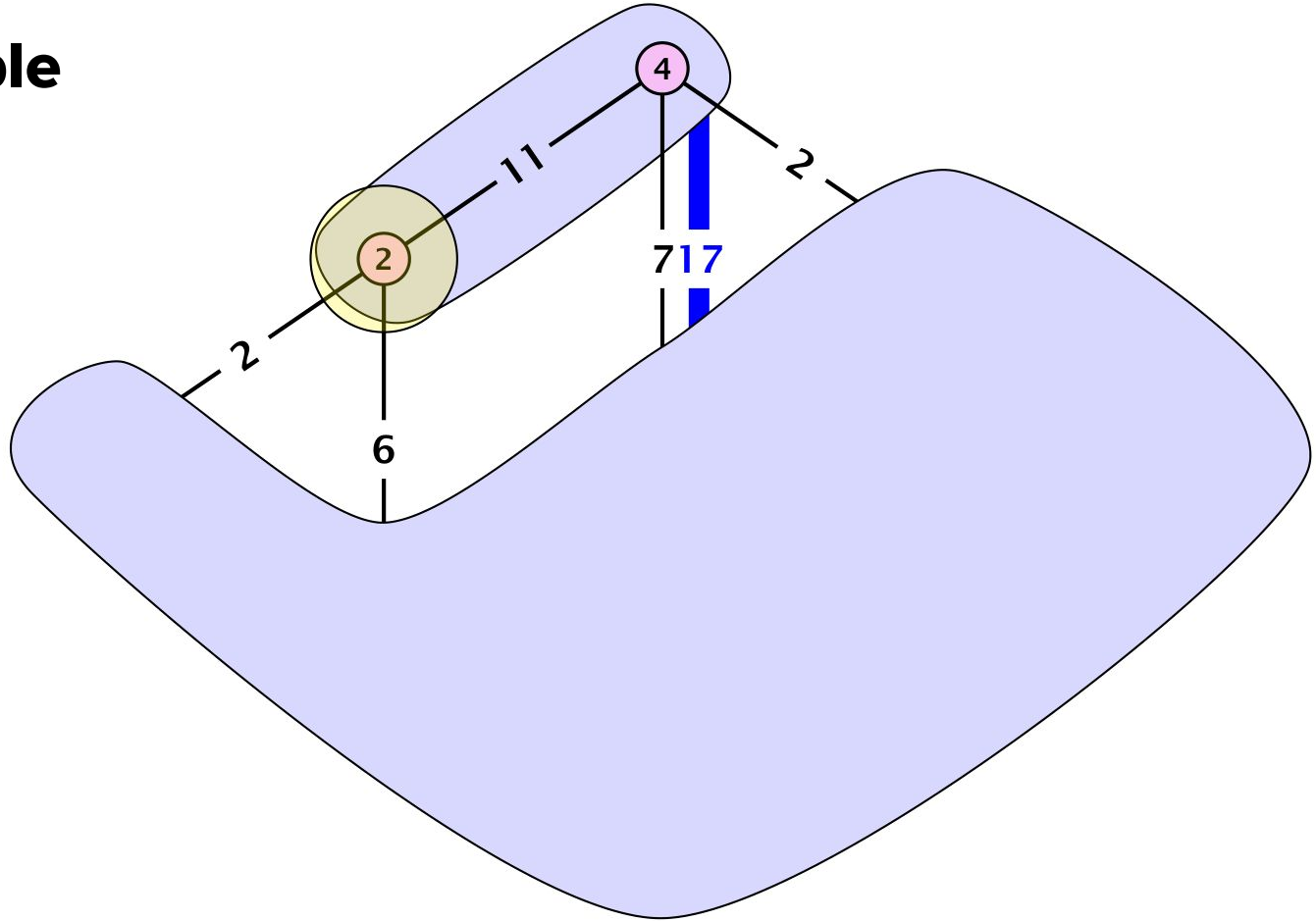
Example



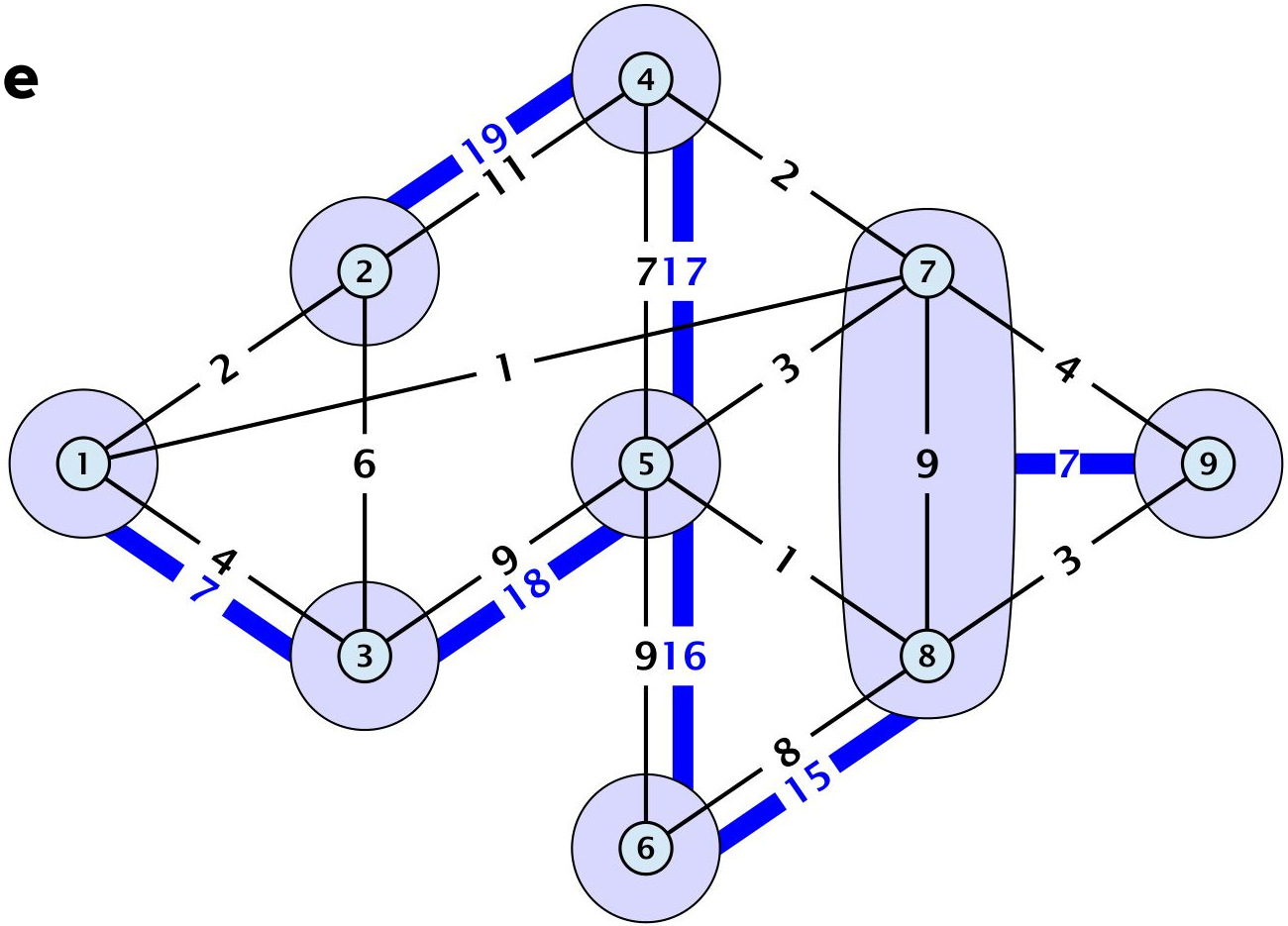
Example



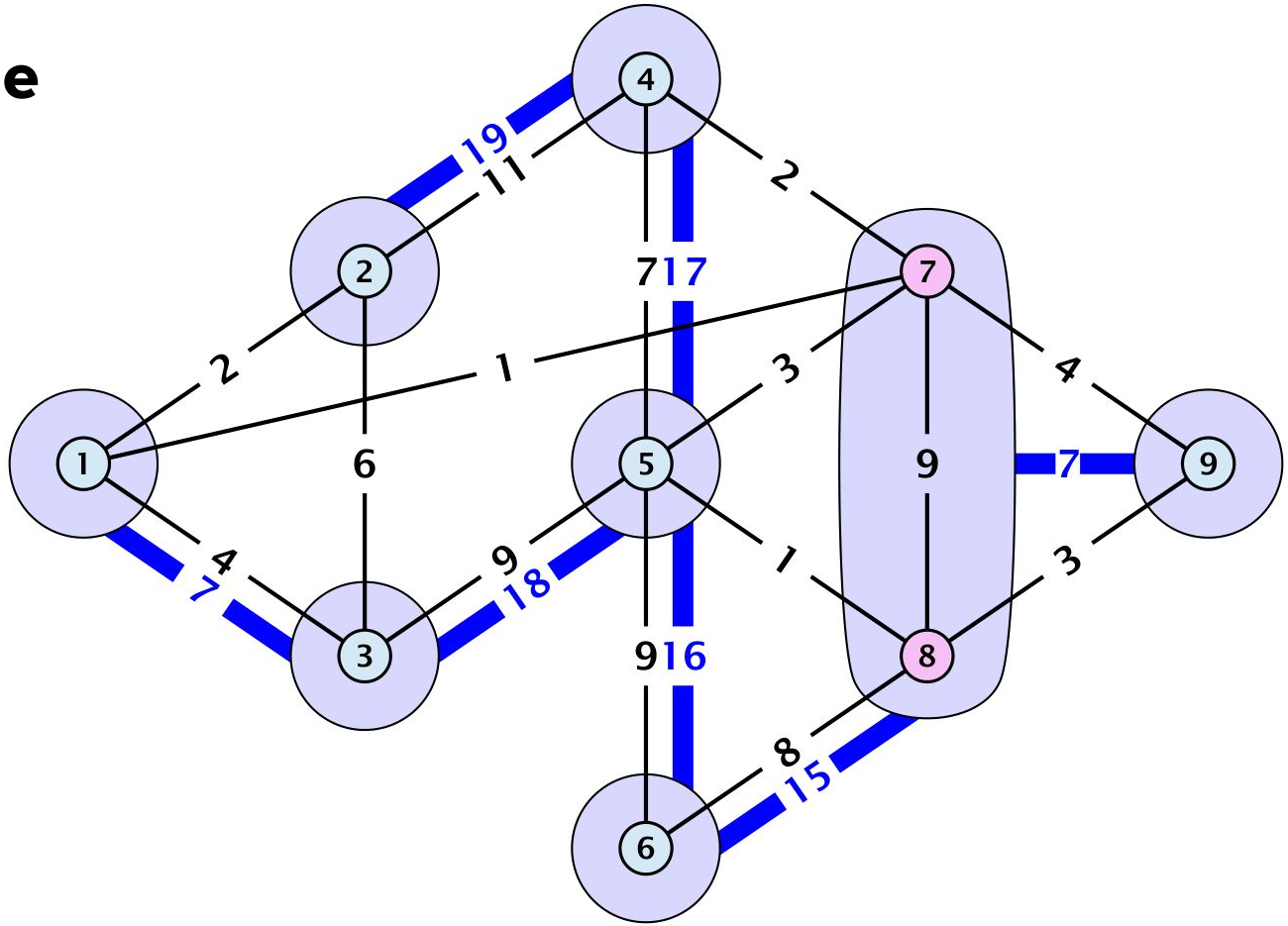
Example



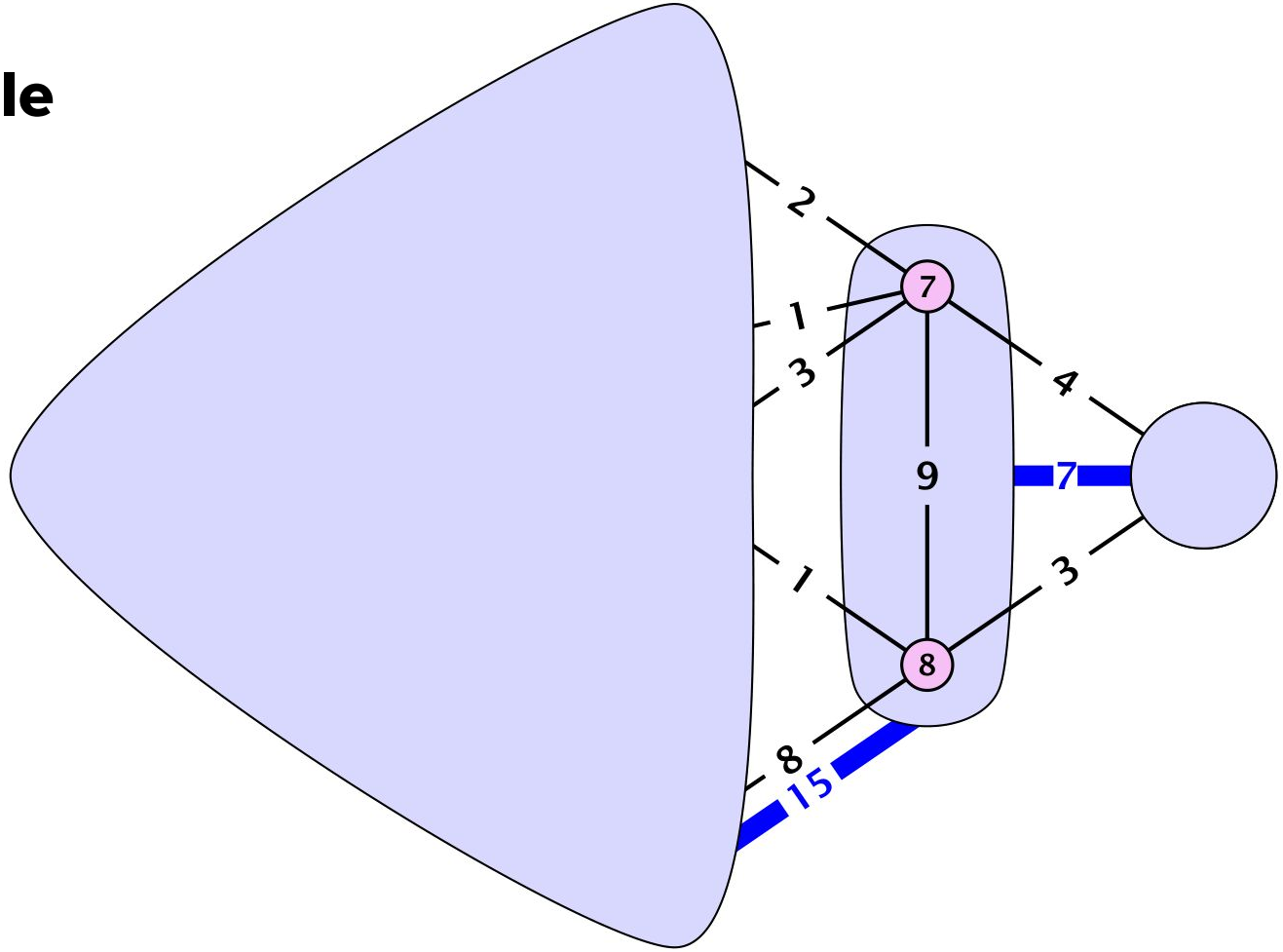
Example



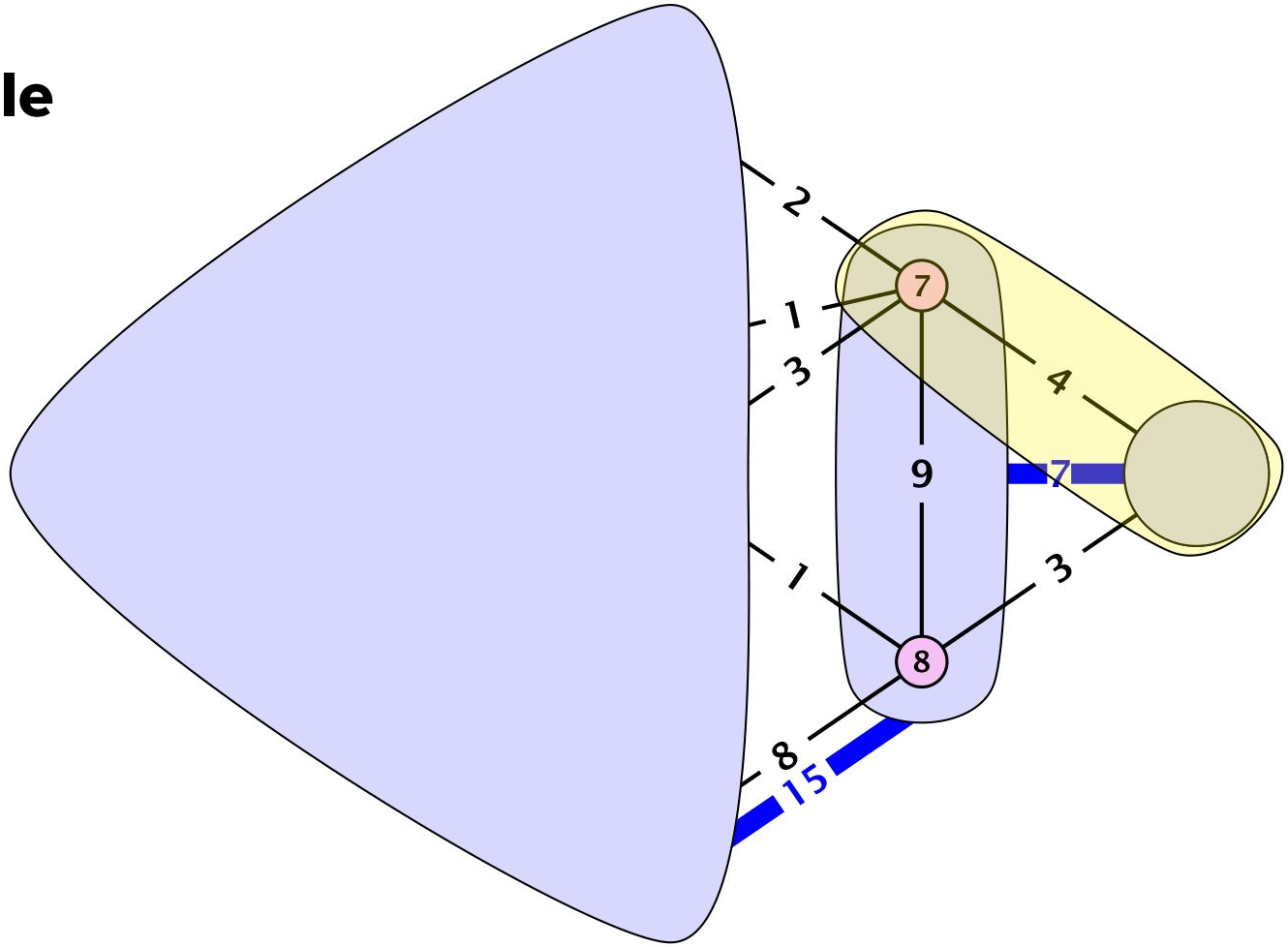
Example



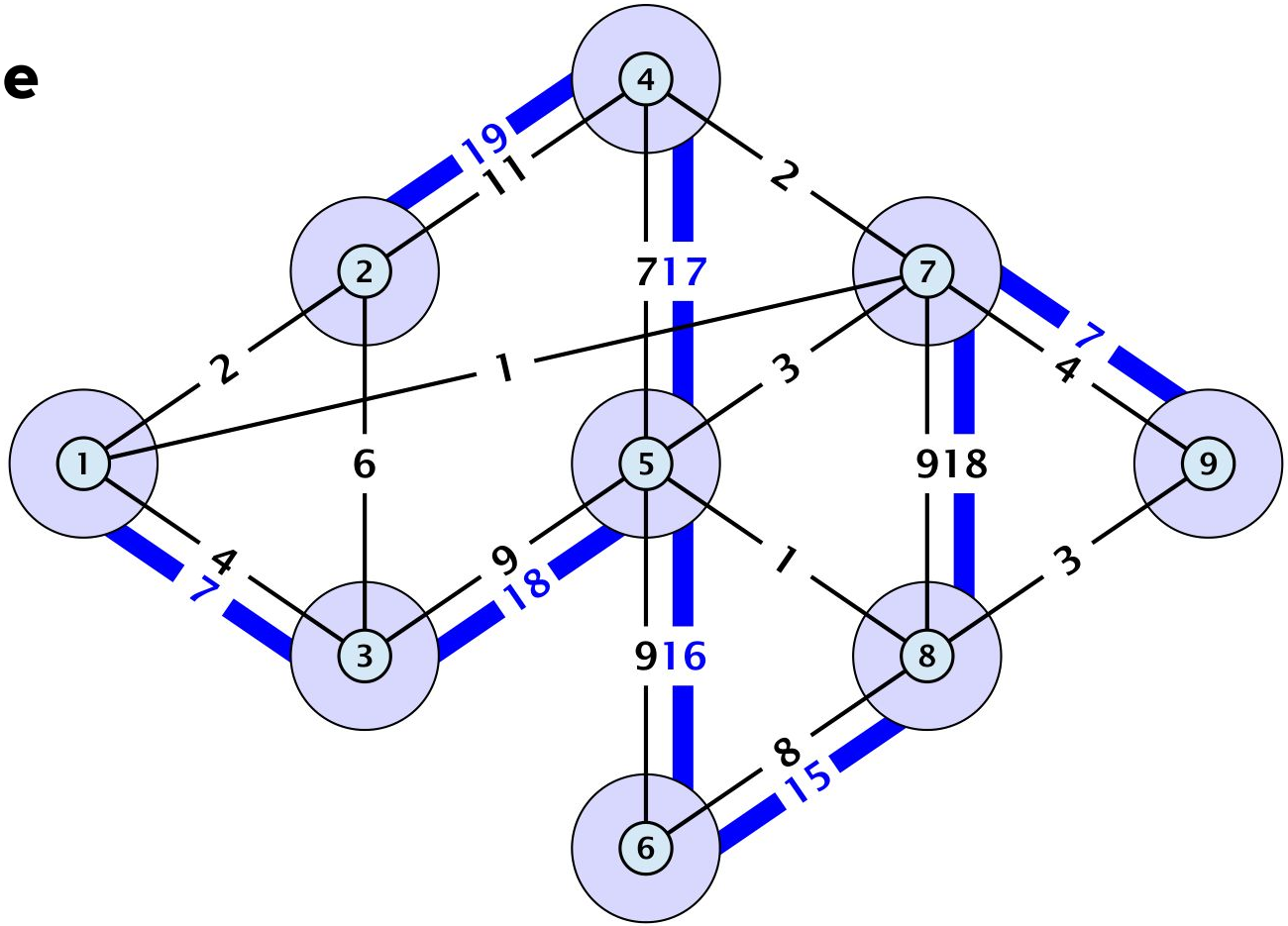
Example



Example



Example



Lemma 1 : For nodes s, t, x $\in$ V we have $f(s, t) \geq \min\{f(s, x), f(x, t)\}$.

Proof :- Suppose this was false and for some nodes s, t, x $\in$ V,

$$f(s, t) < \min\{f(s, x), f(x, t)\}$$

Consider the minimum s-t cut of the graph into node sets S and T. The capacity of the cut is equal to f(s, t). Now,

If x $\in$ S, this is also a x-t cut with capacity f = f(s, t) < f(x, t), which is a contradiction,

Or else x $\in$ T, this is a s-x cut with capacity f = f(s, t) < f(s, x), also a contradiction.

$\therefore$ $f(s, t) \geq \min\{f(s, x), f(x, t)\}$ for all nodes s, t, x $\in$ V.

Lemma 2 : For nodes s, t, $x_1, \ldots, x_k \in V$ we have, $f(s, t) \geq \min\{f(s, x_1), f(x_1, x_2), \ldots, f(x_{k-1}, x_k), f(x_k, t)\}$.

Proof :- Lemma 2, by induction, is an immediate consequence of Lemma 1.

P(m): For nodes s, t, $x_1, \ldots, x_m \in V$ we have, $f(s, t) \geq \min\{f(s, x_1), f(x_1, x_2), \ldots, f(x_{m-1}, x_m), f(x_m, t)\}$

Base case: P(m) is true for m = 1 (lemma 1)

Inductive Step: Let P(m) be true for m = k, then for m = k + 1, we have,

$\min\{f(s, x_1), f(x_1, x_2), \ldots, f(x_k, x_{k+1}), f(x_{k+1}, t)\} = \min\{\min\{f(s, x_1), f(x_1, x_2), \ldots, f(x_k, x_{k+1})\}, f(x_{k+1}, t)\} \leq \min\{f(s, x_{k+1}), f(x_{k+1}, t)\} \leq f(s, t)$

For some set of vertices U, we define δ(U) to be the set of edges with one endpoint in U. Then, for a minimum s-t cut that partitions the vertex set V into S and T, $f(s, t) = c(\delta(S)) = c(\delta(T))$ where c is the *cut function* on the set of vertices.

Definition 1 : Given a finite set E, $f : 2^E \to R$ is *submodular* if for all $A, B \in 2^E$, $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$.

Definition 2 : Given a finite set E, $f : 2^E \to R$ is *posi-modular* if for all $A, B \in 2^E$, $f(A) + f(B) \geq f(A - B) + f(B - A)$.

Cut function is submodular

Let’s show that the cut function on the vertices of a graph is submodular.

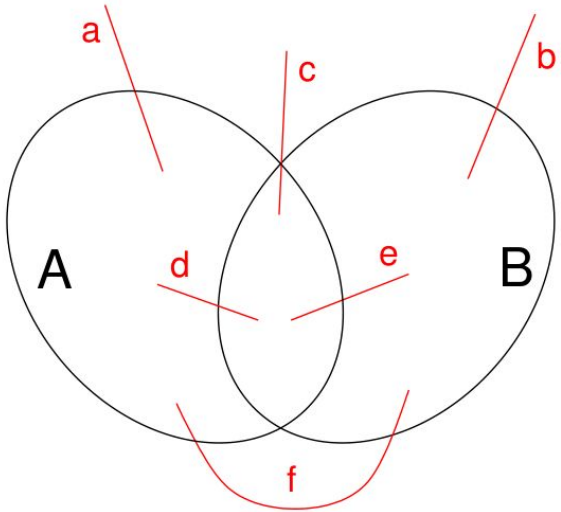
$c(\delta(A)) = a + c + e + f,\ c(\delta(B)) = b + c + d + f$

$c(\delta(A \cup B)) = a + b + c,\ c(\delta(A \cap B)) = c + d + e$

$c(\delta(A)) + c(\delta(B)) = a + b + 2c + d + e + 2f$

$= c(\delta(A \cup B)) + c(\delta(A \cap B)) + 2f$

$\geq c(\delta(A \cup B)) + c(\delta(A \cap B))$



Cut function is posi-modular

Let’s show that the cut function on the vertices of a graph is submodular.

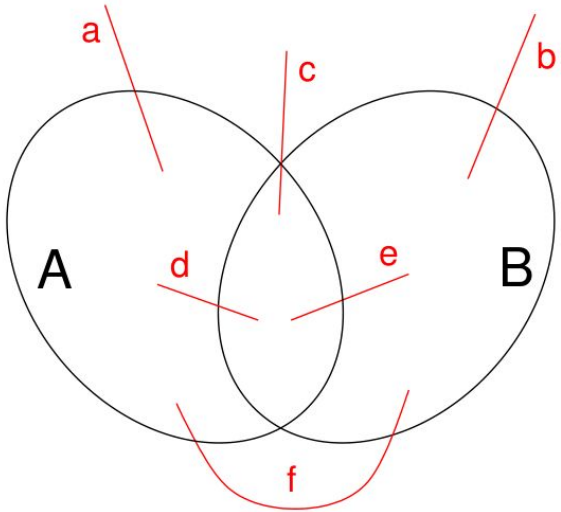
$c(\delta(A)) = a + c + e + f$, $c(\delta(B)) = b + c + d + f$

$c(\delta(A - B)) = a + d + f$, $c(\delta(B - A)) = b + e + f$

$$c(\delta(A)) + c(\delta(B)) = a + b + 2c + d + e + 2f$$

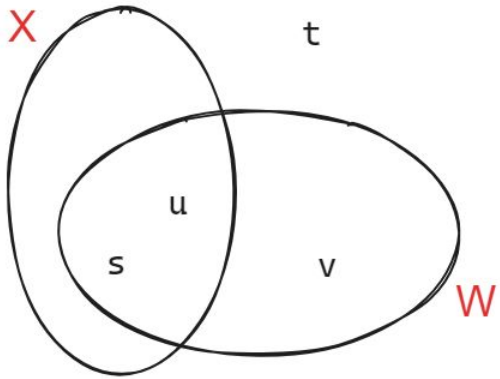
$$= c(\delta(A - B)) + c(\delta(B - A)) + 2c$$

$$\geq c(\delta(A - B)) + c(\delta(B - A))$$



Lemma 3 : Let $\delta(W)$ be an s-t minimum cut in a graph G with respect to a capacity function c. Then for any $u, v \in W$, $u \neq v$, there is a u-v minimum cut $\delta(X)$ where $X \subseteq W$.

Proof :- Let $\delta(X)$ be any u-v minimum cut that crosses W. Suppose without loss of generality that $s \in W$, $s \in X$, and $u \in X$. Then there are two cases to consider:



Case 1 : $t \notin X$, then, since c is submodular,

$\Rightarrow c(\delta(X)) + c(\delta(W)) \geq c(\delta(X \cup W)) + c(\delta(X \cap W))$

Since $X \cup W$ is an s-t cut and $X \cap W$ is an u-v cut,

$\Rightarrow c(\delta(X \cup W)) \geq f(s, t) = c(\delta(W))$ and $c(\delta(X \cap W)) \geq f(u, v) = c(\delta(X))$

$\Rightarrow c(\delta(X \cup W)) = f(s, t) = c(\delta(W))$ and $c(\delta(X \cap W)) = f(u, v) = c(\delta(X))$

$\therefore$ $X \cap W$ is a minimum u-v cut.

Case 2 : $t \in X$. Since c is posi-modular,

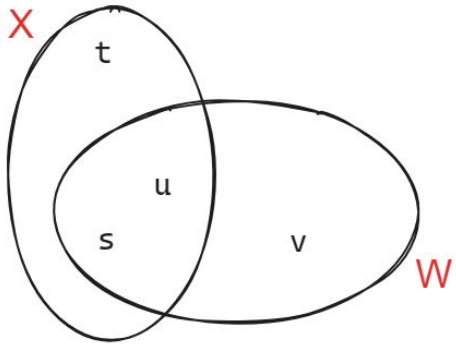
$\Rightarrow c(\delta(X)) + c(\delta(W)) \geq c(\delta(X - W)) + c(\delta(W - X))$

Since X - W is an s-t cut and W - X is an u-v cut,

$\Rightarrow c(\delta(X - W)) \geq f(s, t) = c(\delta(W))$ and $c(\delta(W - X)) \geq f(u, v) = c(\delta(X))$

$\Rightarrow c(\delta(X - W)) = f(s, t) = c(\delta(W))$ and $c(\delta(W - X)) = f(u, v) = c(\delta(X))$

$\therefore$ W - X is a minimum u-v cut.



So, lemma 3 shows that minimum cuts can be uncrossed, a technique that is useful in the construction of the Gomory-Hu Tree.

Theorem: Let T be a Gomory-Hu tree for a graph G = (V, E). Then, for all u, v $\in$ V , let st be the edge on the unique path in T from u to v such that f(s, t) is minimized. Then, f(u, v) = f(s, t) and the cut $\delta(W)$ induced by T − st is a u-v minimum cut in G.

Proof :- Consider the path from u to v in T. We note that if uv = st, then f(u, v) = f(s, t). Otherwise let u, x_1, x_2, ... , x_k, v be the path from u to v in T, then by lemma 2, $f(u, v) \geq \min\{f(u, x_1), f(x_1, x_2), \ldots , f(x_k, v)\} = f(s, t)$. However, we have that $f(u, v) \leq f(s, t)$ since the cut induced by T - st is a valid u-v cut. Thus, we have f(u, v) = f(s, t) and the cut induced by T - st is a minimum u-v cut.

Theorem: The Minimum K-Cut with Gomory-Hu Tree Algorithm has an approximation ratio of 2 − 2/k.

Proof :- Let $E^* \subseteq E$ be an optimal k-cut in G that defines a graph of k connected components, V_1, V_2, ..., V_k. Let E^*_i be the cut that separates the component V_i from the rest of the graph. So, $E^* = \cup E^*_i$.

We can assume without loss of generality that $c(\delta(V_1)) \leq c(\delta(V_2)) \leq \ldots \leq c(\delta(V_k))$. As before, every edge in E^* is incident to two of the connected components. Therefore, each edge will be in exactly two of the cuts $\delta(V_i)$, and therefore, $2c(E^*) = c(\delta(V_1)) + c(\delta(V_2)) + \ldots + c(\delta(V_k))$.

$$c(C) = \sum_{i=1}^{k-1} w(e_i) \leq \sum_{i=1}^{k-1} c(\delta(V_i)) \leq (1 - \tfrac{1}{k}) \sum_{i=1}^{k} c(\delta(V_i)) = 2(1 - \tfrac{1}{k}) c(E^*)$$

where w is the weight function in the Gomory-Hu Tree.

Running Time Analysis

1)**Construction of Gomory-Hu tree**:-

In the process of constructing the Gomory-Hu tree, the algorithm iteratively finds the minimum cut V-1 times, considering each possible pair of vertices. For each iteration, the Ford-Fulkerson algorithm is applied. The time complexity of the Ford-Fulkerson algorithm is O(V^2 * max_flow), where V is the number of vertices.

Therefore, the overall time complexity for constructing the Gomory-Hu tree is given by:

$$T_{\text{Gomory-Hu}} = (V - 1) \cdot O(V^2 \cdot \text{max\_flow})$$

Running Time Analysis

2) Finding k-Min Cut Solution-

The min-K-cut algorithm, following Gomory-Hu tree construction, exhibits a concise time complexity of O(N). **This efficiency arises from the algorithm's focus on finding the path between two vertices in the tree** and determining the minimum weight edges along that path (By Sorting) . Mathematically, the time complexity is succinctly expressed as:-

$$T_{\text{minKcut}} = O(N \log N)$$

N represents the number of vertices considered, aligning directly with the total vertices in the Gomory-Hu tree.

Team Members

- Sahil (2021CSB1128)
- Harsh Raj Srivastava (2021CSB1091)
- Gyanendra Mani (2021CSB1090)
- Gopal Bansal (2021CSB1089)
- Vikalp Dhalwal (2021CSB1140)

THANK YOU!