Maximum Flow

Professor Siu-Wing Cheng

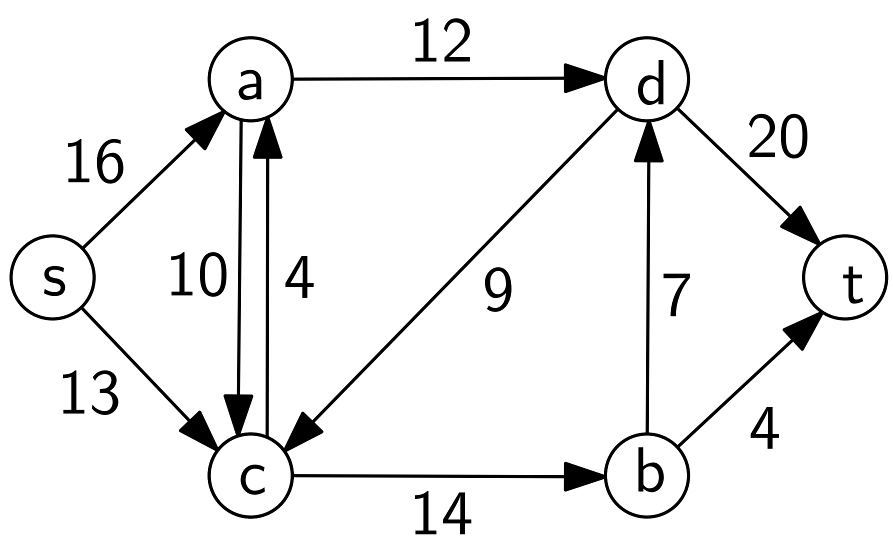
Suppose that you have some goods to transport by a road network from a warehouse to a port for shipping. The roads can handle different maximum loads per day. What is the maximum load that can be transported from the warehouse to the port per day?

Graph Model:

A directed graph $G = (V, E)$. There are two special vertices. One is the source s. The other is the sink t.

Each edge e of G models a transportation channel and has a capacity $c(e)$.

G is called a flow network.



Flow:

A flow f is a function that maps an ordered pair of vertices (u, v) to a real number $f(u, v)$ that models the net amount of goods to be transported along the edge (u, v). By default, $f(u, u) = 0$ for every vertex u.

Flow:

A flow f is a function that maps an ordered pair of vertices (u, v) to a real number $f(u, v)$ that models the net amount of goods to be transported along the edge (u, v). By default, $f(u, u) = 0$ for every vertex u.

Even if (u, v) is not an edge in G, $f(u, v)$ is defined and it may be non-zero (negative). This will be convenient for our purposes.

Flow:

A flow f is a function that maps an ordered pair of vertices (u, v) to a real number $f(u, v)$ that models the net amount of goods to be transported along the edge (u, v). By default, $f(u, u) = 0$ for every vertex u.

Even if (u, v) is not an edge in G, $f(u, v)$ is defined and it may be non-zero (negative). This will be convenient for our purposes.

We write $\sum_v f(u, v)$ to denote the summation of $f(u, v)$ over all vertices v in G (including u itself).

A flow f has to satisfy three conditions:

A flow f has to satisfy three conditions:

1. Capacity constraint: $f(u, v) \leq c(u, v)$. That is, the amount of goods transported on the edge (u, v) cannot exceed its capacity $c(u, v)$. We define $c(u, v) = 0$ when (u, v) is not an edge of G.

A flow f has to satisfy three conditions:

1. Capacity constraint: $f(u, v) \leq c(u, v)$. That is, the amount of goods transported on the edge (u, v) cannot exceed its capacity $c(u, v)$. We define $c(u, v) = 0$ when (u, v) is not an edge of G.

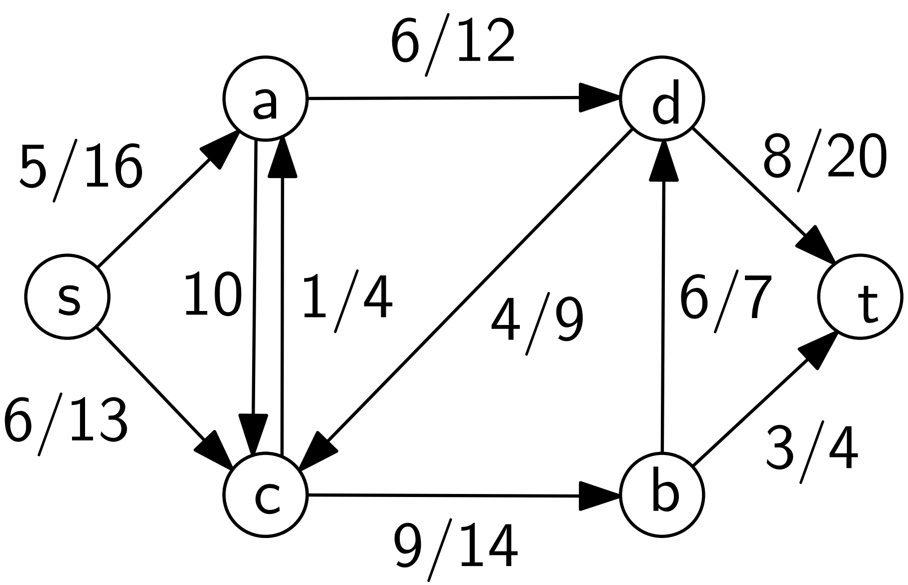
2. Skew symmetry: $f(u, v) = -f(v, u)$ for every pair of vertices u and v. A credit/deficit view. If v gets x units of goods from u, it is equivalent to u getting $-x$ units of good from v.

A flow f has to satisfy three conditions:

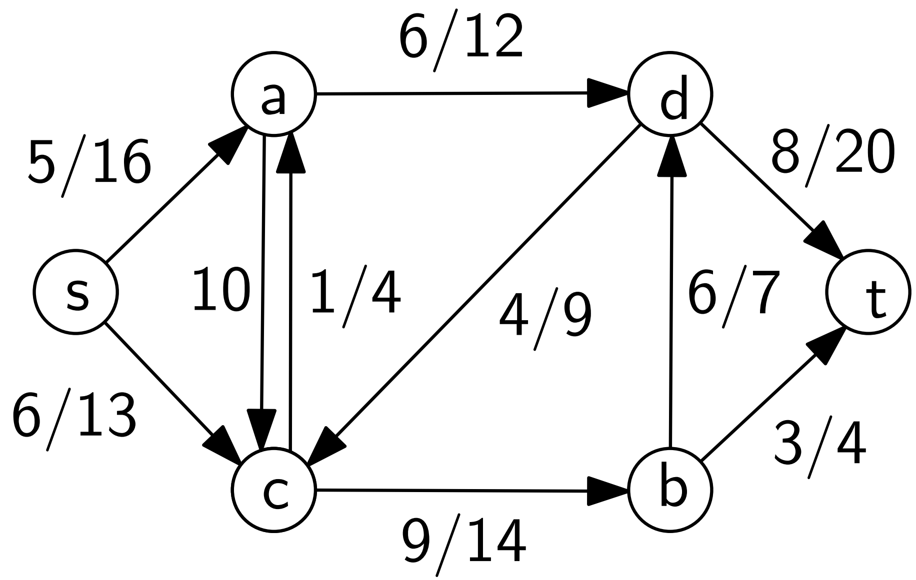
1. Capacity constraint: $f(u, v) \leq c(u, v)$. That is, the amount of goods transported on the edge (u, v) cannot exceed its capacity $c(u, v)$. We define $c(u, v) = 0$ when (u, v) is not an edge of G.

2. Skew symmetry: $f(u, v) = -f(v, u)$ for every pair of vertices u and v. A credit/deficit view. If v gets x units of goods from u, it is equivalent to u getting $-x$ units of good from v.

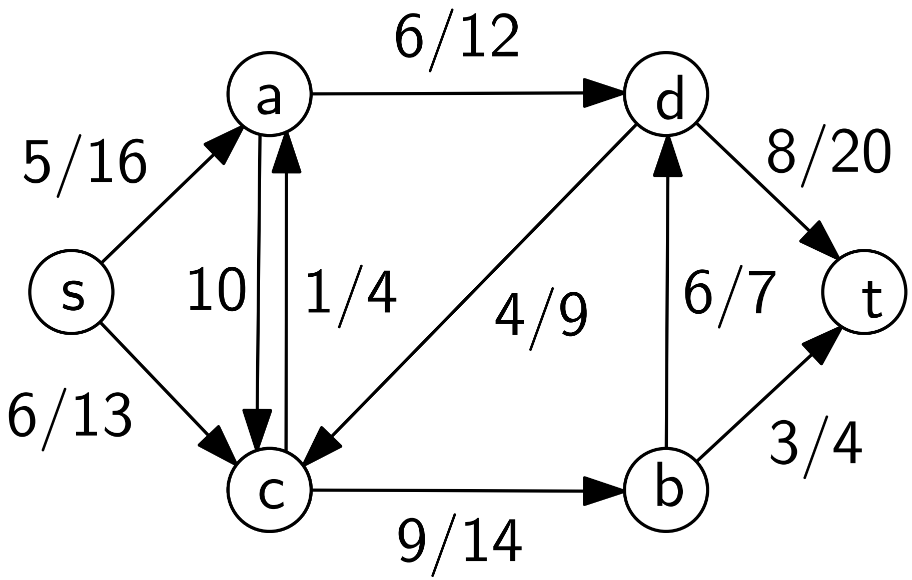
3. Flow conservation: for every vertex u other than s and t, $\sum_v f(u, v) = 0$. Model the fact that no good stops at any vertex other than s and t.



The left number is the flow value along an edge. The right number is the edge capacity. If the left number is missing, the flow value along that edge is zero.



By flow conservation, the net flow into any vertex other than s and t is zero.



By flow conservation, the net flow into any vertex other than s and t is zero.

There is a net flow out of s. There is a net flow into t. These two values are equal. This is the value of the flow f. We denote it by $|f|$, and $|f| = \sum_v f(s,v) = \sum_v f(v,t)$.

Ford-Fulkerson's algorithm:

There are three main steps.

1. Start with a zero flow. That is, $f(u, v) = 0$ for all ordered pairs (u, v) of vertices.
2. Find a path in G from s to t that allows us to increase the current flow value. This path is called an augmenting path.
3. Use the augmenting path to increase the flow value. Then repeat (2) and (3) until an augmenting path cannot be found.

The final flow is a flow that achieves the maximum flow value, i.e. maximum flow.

Residual Network:

The residual network is defined with respect to the flow network G and the current flow f in G. We denote it by G_f.

Residual Network:

The residual network is defined with respect to the flow network G and the current flow f in G. We denote it by G_f.

G_f has the same vertex set as G. For every ordered pair of vertices (u, v), define its residual capacity $c_f(u, v) = c(u, v) - f(u, v)$. G_f contains the edge (u, v) if and only if $c_f(u, v) > 0$.

Residual Network:

The residual network is defined with respect to the flow network G and the current flow f in G. We denote it by G_f.

G_f has the same vertex set as G. For every ordered pair of vertices (u, v), define its residual capacity $c_f(u, v) = c(u, v) - f(u, v)$. G_f contains the edge (u, v) if and only if $c_f(u, v) > 0$.

Physical meaning: If $f(u, v) \geq 0$, then $c_f(u, v)$ is the unused capacity of (u, v) that we can exploit.

Residual Network:

The residual network is defined with respect to the flow network G and the current flow f in G. We denote it by G_f.

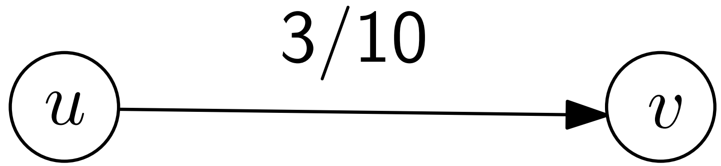
G_f has the same vertex set as G. For every ordered pair of vertices (u, v), define its residual capacity $c_f(u, v) = c(u, v) - f(u, v)$. G_f contains the edge (u, v) if and only if $c_f(u, v) > 0$.

Physical meaning: If $f(u, v) \geq 0$, then $c_f(u, v)$ is the unused capacity of (u, v) that we can exploit.
If $f(u, v) < 0$, then u is accepting $f(v, u)$ units of flow from v. Then, $c_f(u, v)$ models the possibility of pushing these units back from u to v as well as sending another $c(u, v)$ units from u to v.

Example of $c_f(u, v)$

$f(u, v) = 3$

$f(v, u) = -3$



flow on (u, v) in G

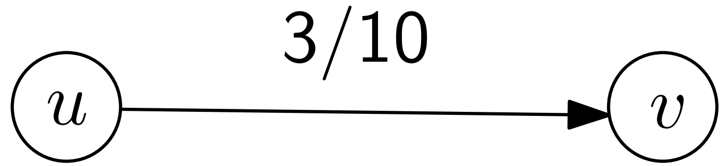
$c(u, v) = 10$

$c(v, u) = 0$

Example of $c_f(u,v)$

$f(u,v) = 3$

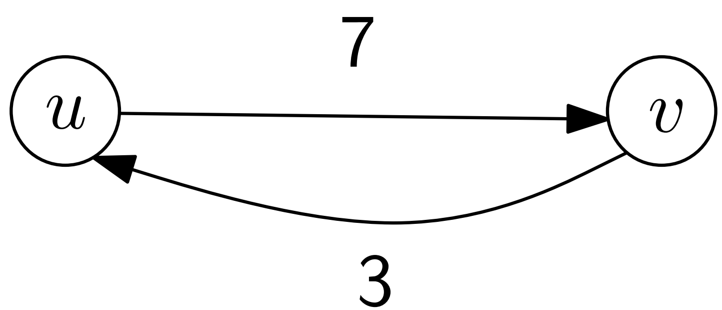
$f(v,u) = -3$



$c(u,v) = 10$

$c(v,u) = 0$

flow on (u,v) in G

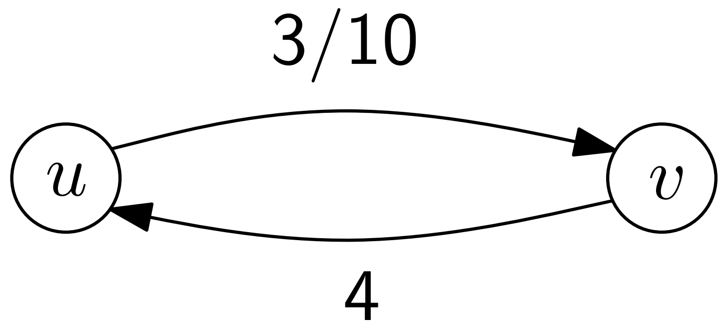


residual capacities of (u,v) and (v,u) in G_f

Example of $c_f(u, v)$

$$f(u, v) = 3$$

$$f(v, u) = -3$$



$$c(u, v) = 10$$

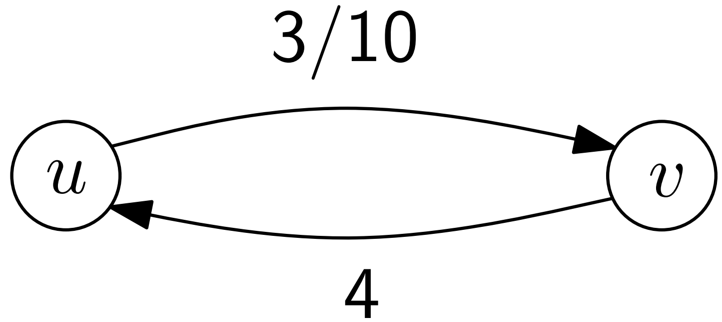
$$c(v, u) = 4$$

flow on (u, v) and (v, u) in G

Example of $c_f(u,v)$

$f(u,v) = 3$

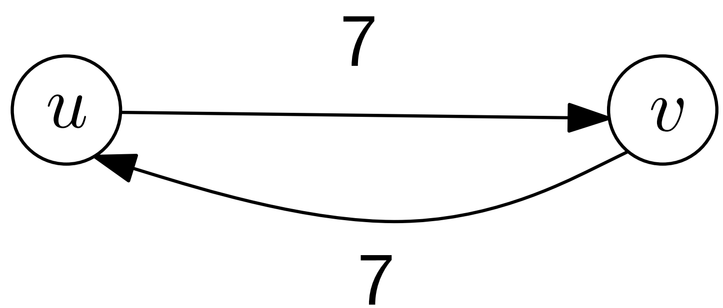
$f(v,u) = -3$



$c(u,v) = 10$

$c(v,u) = 4$

flow on (u,v) and (v,u) in G

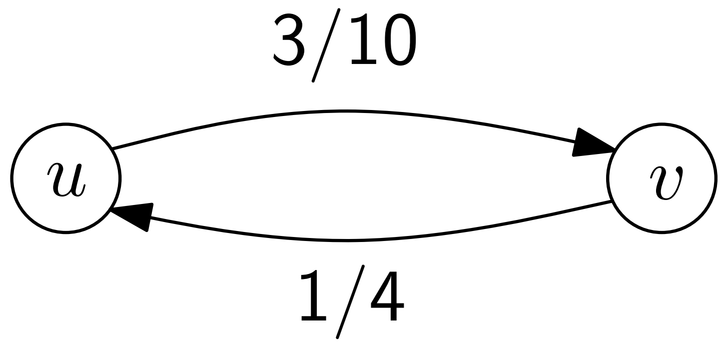


residual capacities of (u,v) and (v,u) in G_f

Example of $c_f(u,v)$

$f(u,v) = 2$

$f(v,u) = -2$



$c(u,v) = 10$

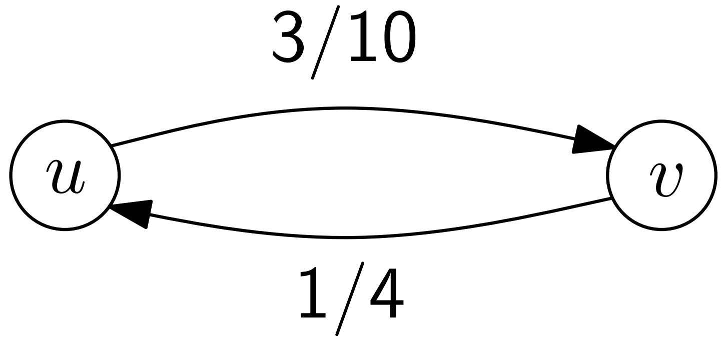
$c(v,u) = 4$

flow on (u,v) and (v,u) in G

Example of $c_f(u, v)$

$f(u, v) = 2$

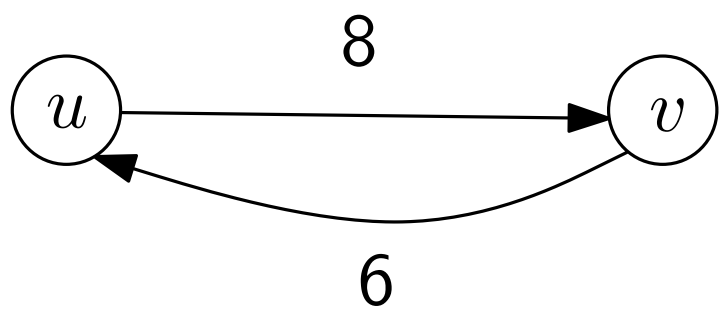
$f(v, u) = -2$



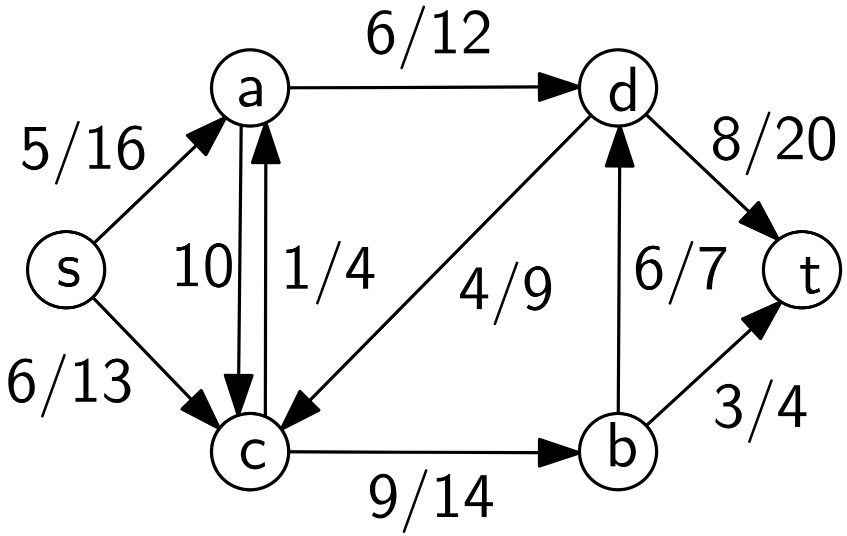
$c(u, v) = 10$

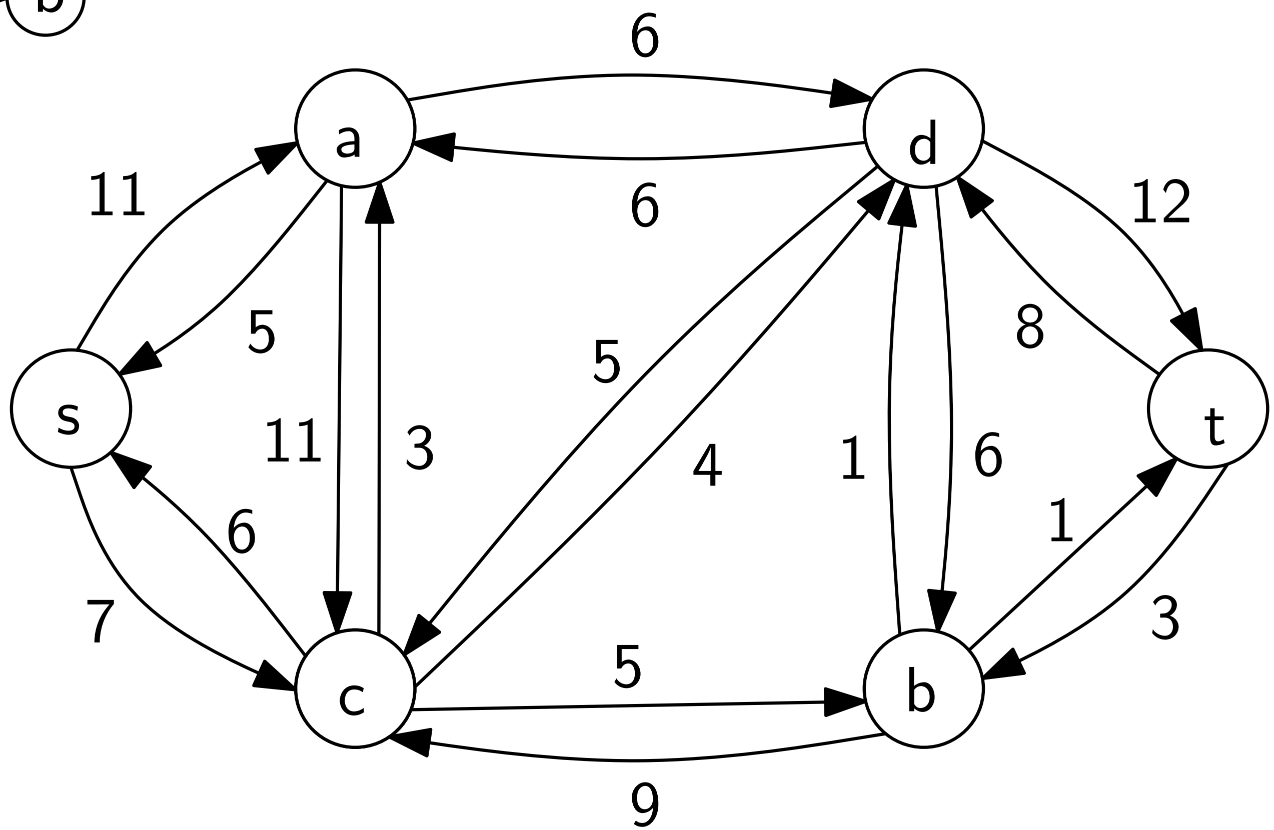
$c(v, u) = 4$

flow on (u, v) and (v, u) in G



residual capacities of (u, v) and (v, u) in G_f





residual network

Augmenting Path

Any path from s to t in G_f is an augmenting path. It can be found by depth-first search or breadth-first search.

Augmenting Path

Any path from s to t in G_f is an augmenting path. It can be found by depth-first search or breadth-first search.

How many edges are there in G_f?

Augmenting Path

Any path from s to t in G_f is an augmenting path. It can be found by depth-first search or breadth-first search.

How many edges are there in G_f?

Given two vertices u and v, if neither (u, v) nor (v, u) is an edge in G, then $c(u, v) = c(v, u) = 0 \Rightarrow f(u, v) = f(v, u) = 0 \Rightarrow c_f(u, v) = c_f(v, u) = 0$. So neither (u, v) nor (v, u) is an edge in G_f.

Augmenting Path

Any path from s to t in G_f is an augmenting path. It can be found by depth-first search or breadth-first search.

How many edges are there in G_f?

Given two vertices u and v, if neither (u, v) nor (v, u) is an edge in G, then $c(u, v) = c(v, u) = 0 \Rightarrow f(u, v) = f(v, u) = 0 \Rightarrow c_f(u, v) = c_f(v, u) = 0$. So neither (u, v) nor (v, u) is an edge in G_f.

If (u, v) is an edge in G, then both (u, v) and (v, u) may be edges in G_f. So there are at most $2m$ edges in G_f.

Augmenting Path

Any path from s to t in G_f is an augmenting path. It can be found by depth-first search or breadth-first search.

How many edges are there in G_f?

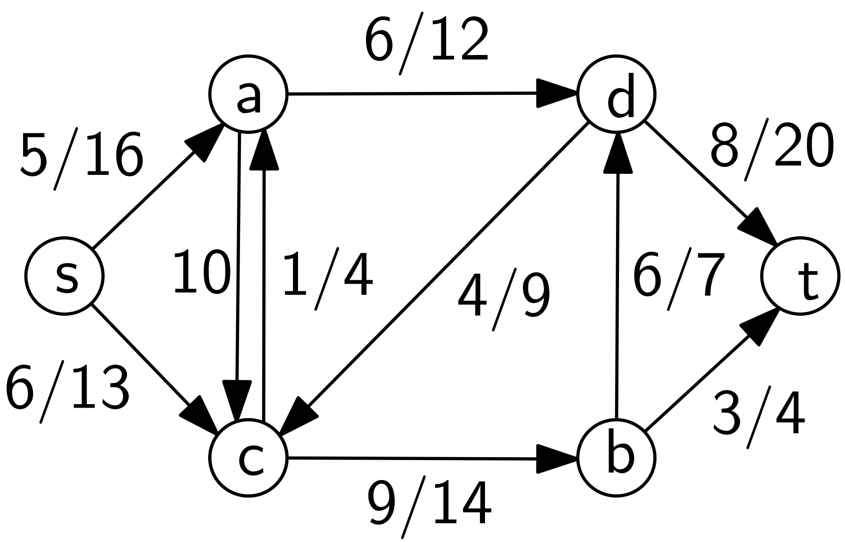
Given two vertices u and v, if neither (u, v) nor (v, u) is an edge in G, then $c(u, v) = c(v, u) = 0 \Rightarrow f(u, v) = f(v, u) = 0 \Rightarrow c_f(u, v) = c_f(v, u) = 0$. So neither (u, v) nor (v, u) is an edge in G_f.

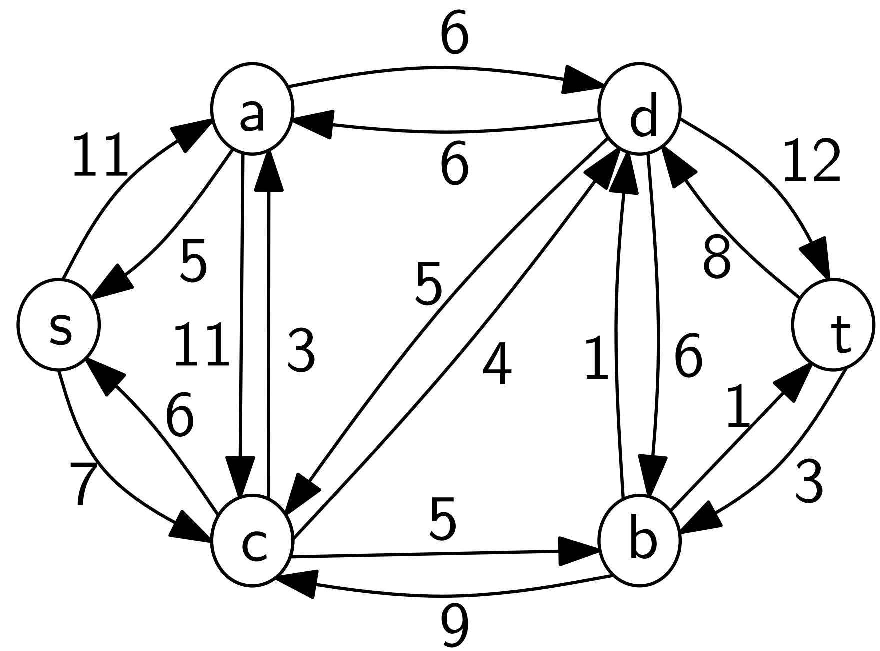
If (u, v) is an edge in G, then both (u, v) and (v, u) may be edges in G_f. So there are at most $2m$ edges in G_f.

Therefore, the graph search takes $O(n + m)$ time.

Let Δ be the minimum edge capacity on the augmenting path. We can send Δ units of flow along the augmenting path,

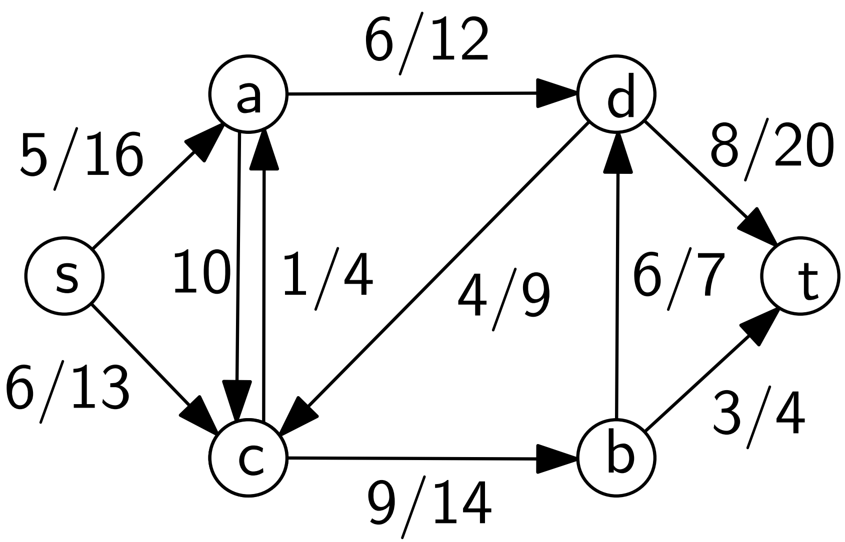
This corresponds to updating the current flow so that an additional Δ units are sent from s to t.

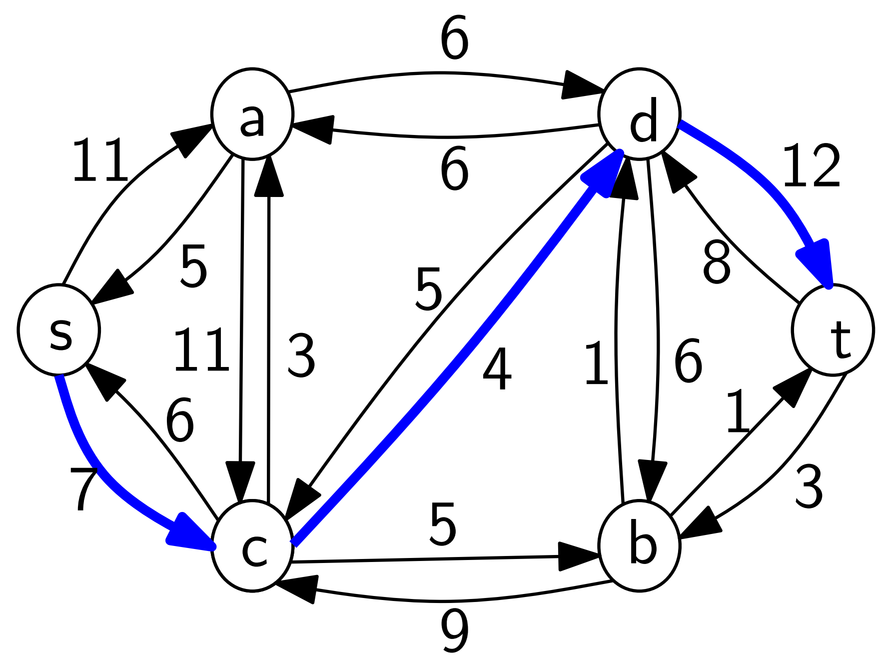




Let Δ be the minimum edge capacity on the augmenting path. We can send Δ units of flow along the augmenting path,

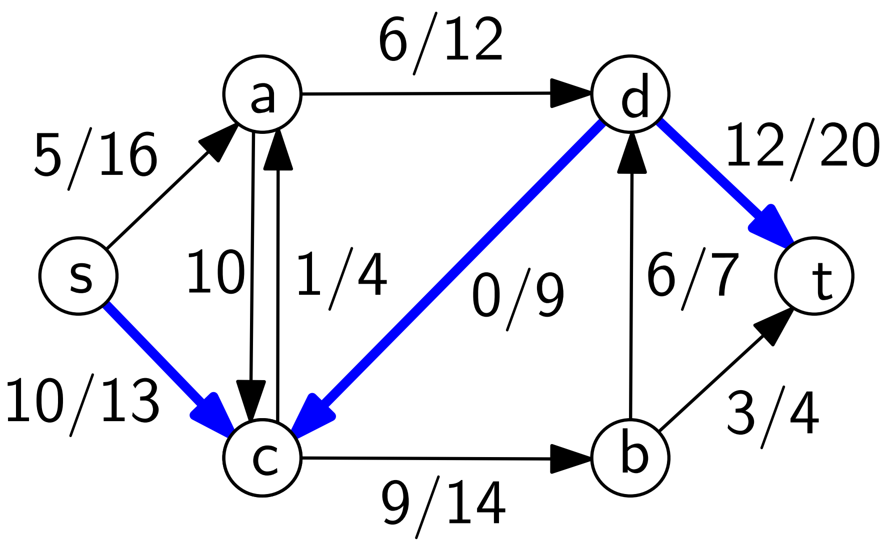
This corresponds to updating the current flow so that an additional Δ units are sent from s to t.

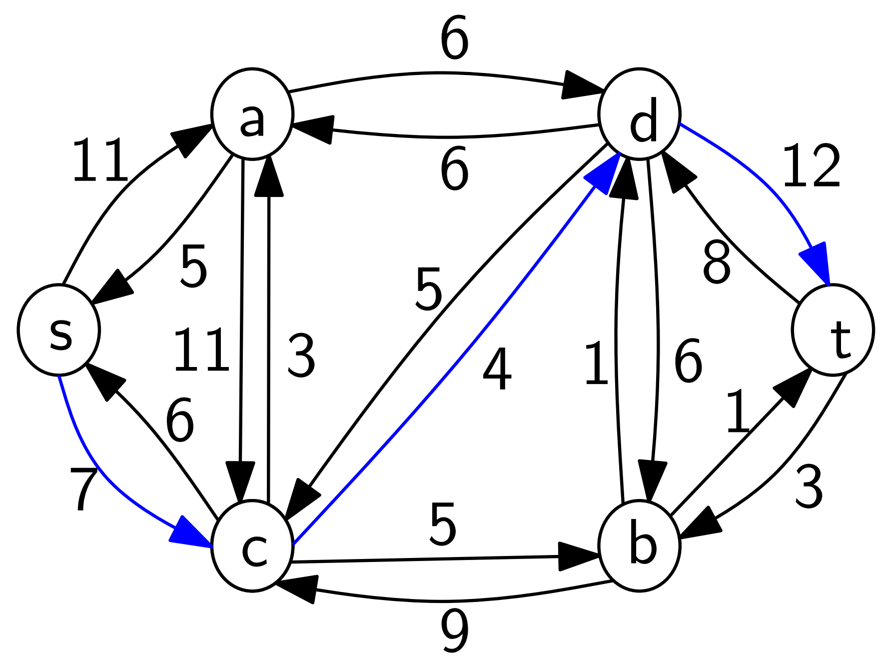




Let Δ be the minimum edge capacity on the augmenting path. We can send Δ units of flow along the augmenting path,

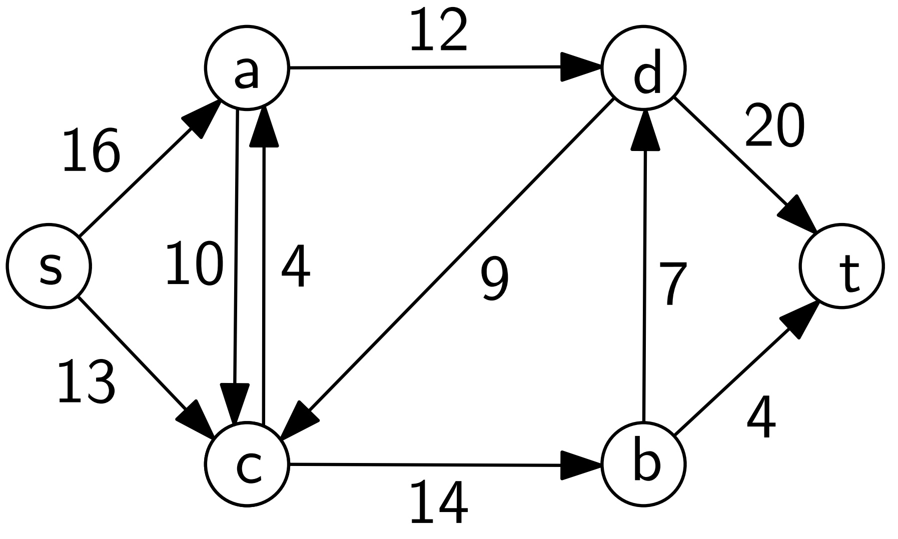
This corresponds to updating the current flow so that an additional Δ units are sent from s to t.



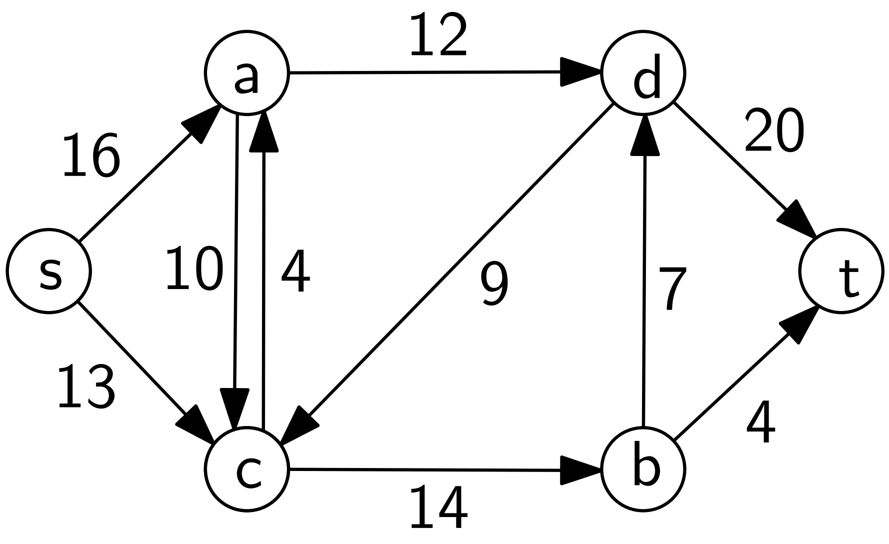


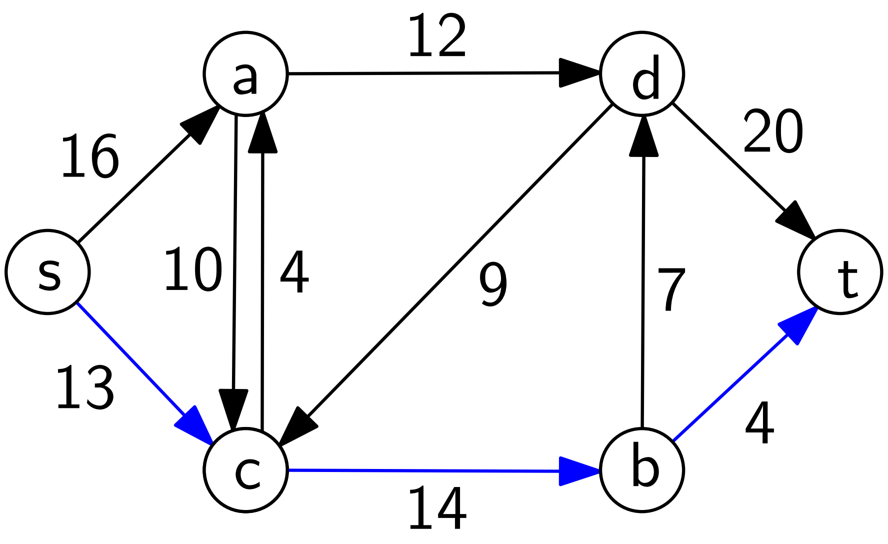
Flow value increases from 11 to 15

Example

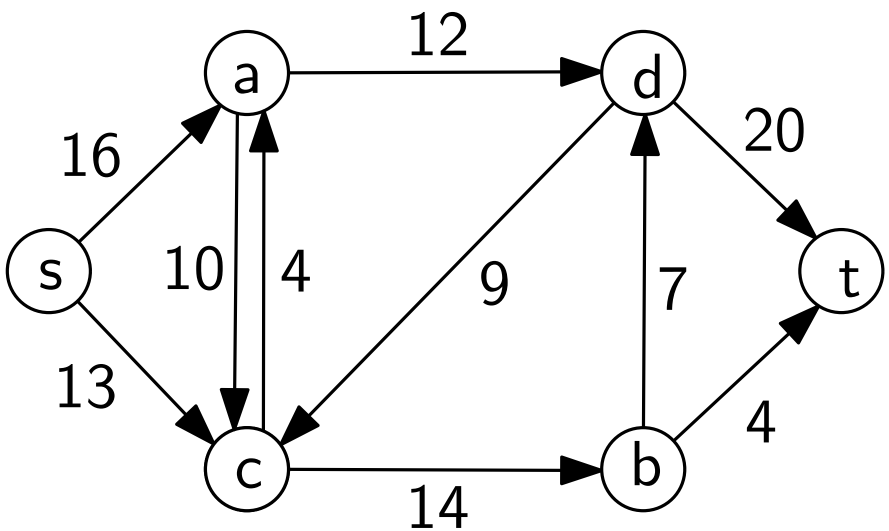


Example

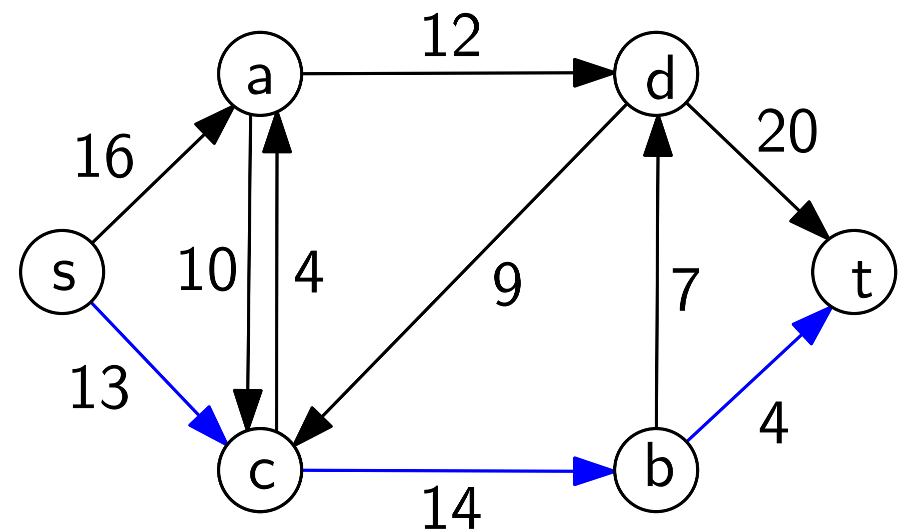




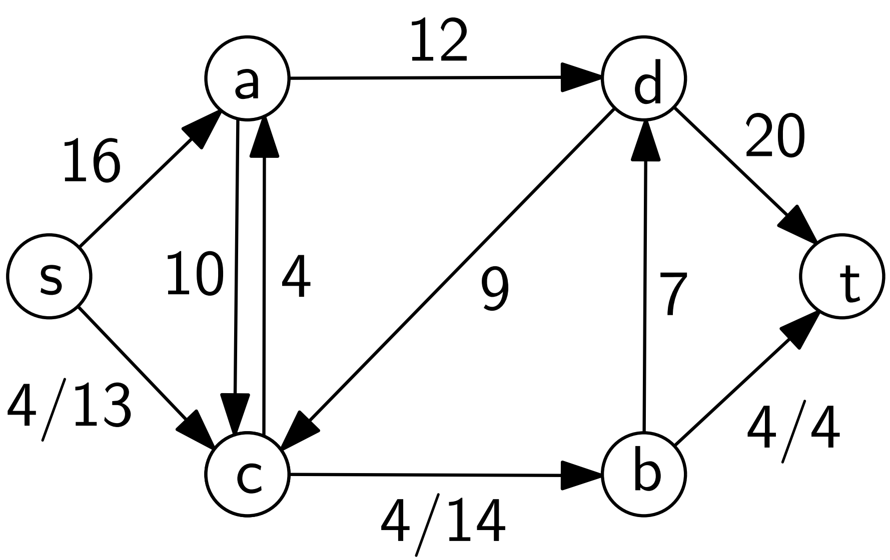
Example



a
d
s
t
c
b
12
16
20
10
4
9
7
13
4
14

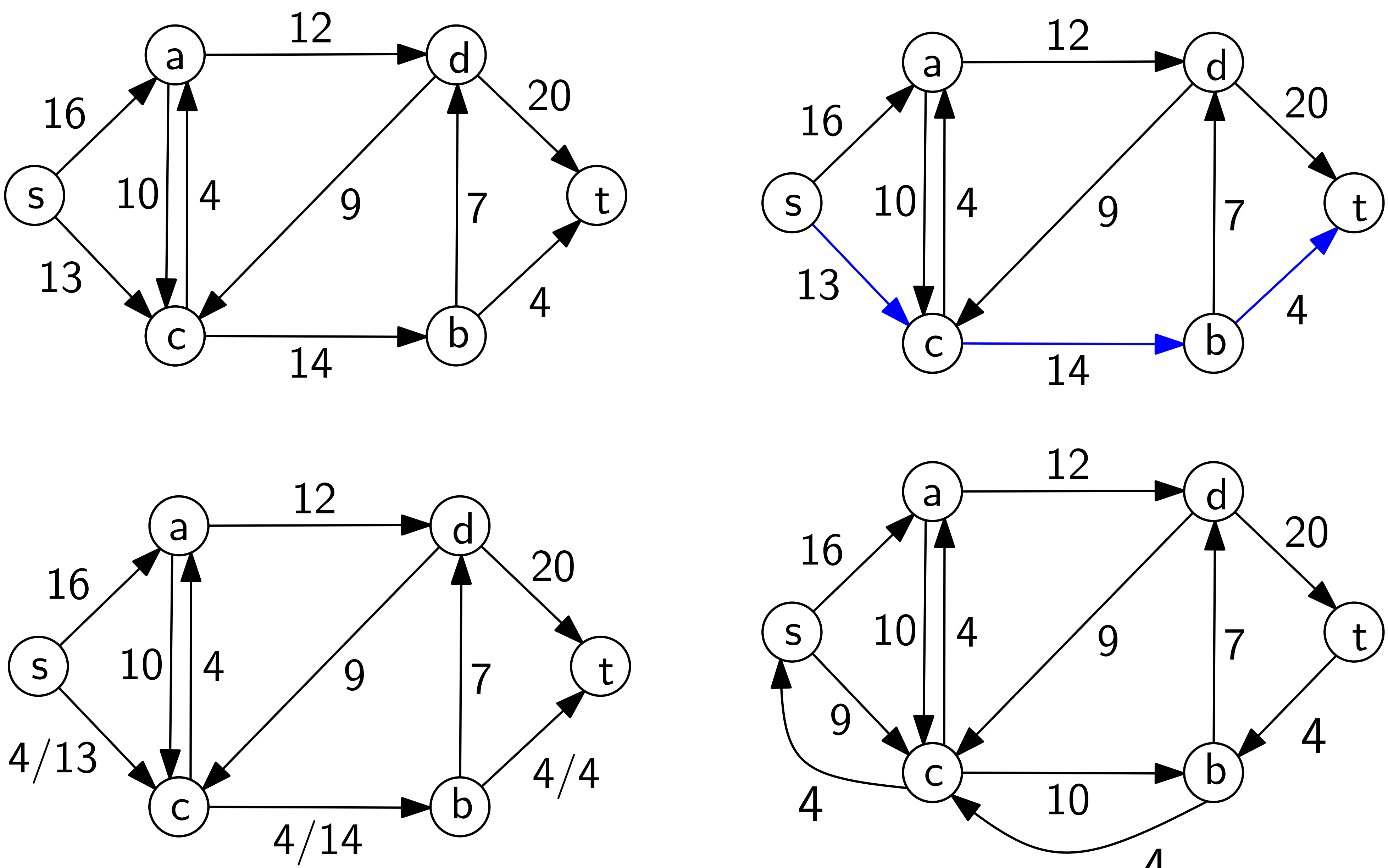


a
d
s
t
c
b
12
16
20
10
4
9
7
13
4
14

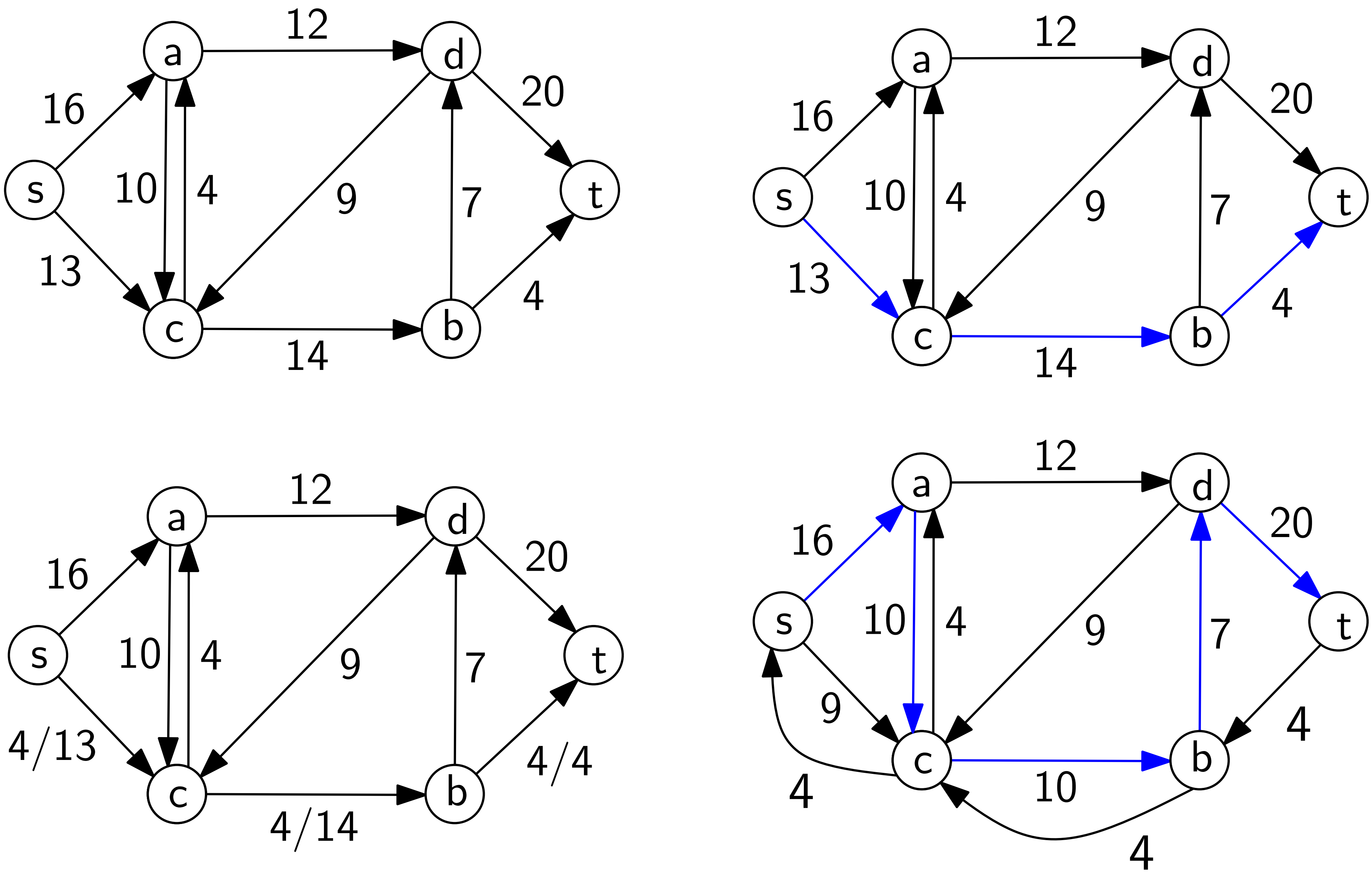


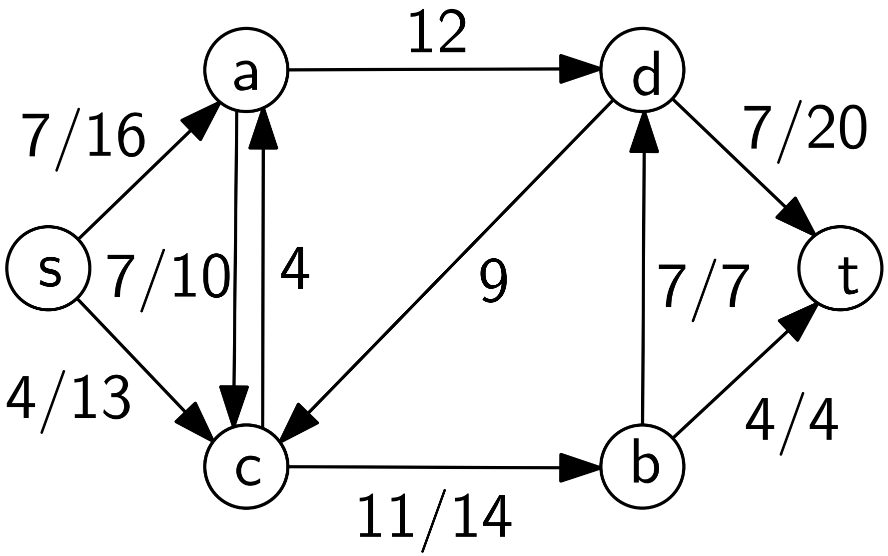
a
d
s
t
c
b
12
16
20
10
4
9
7
4/13
4/4
4/14

Example

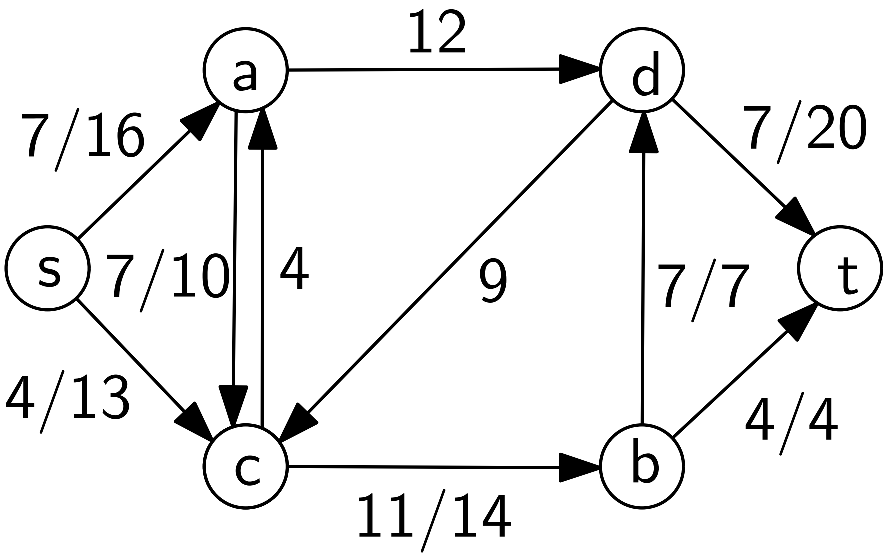


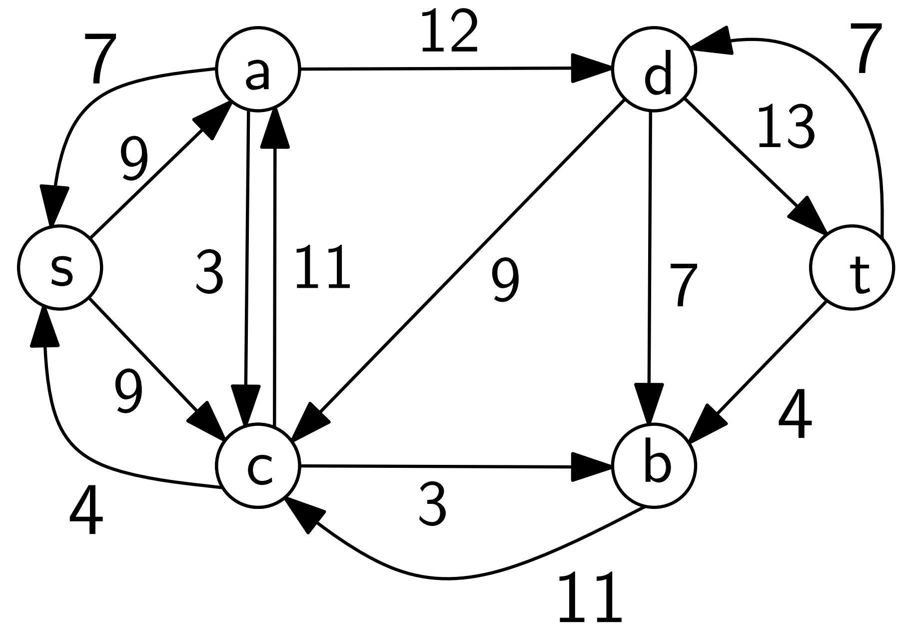
Example

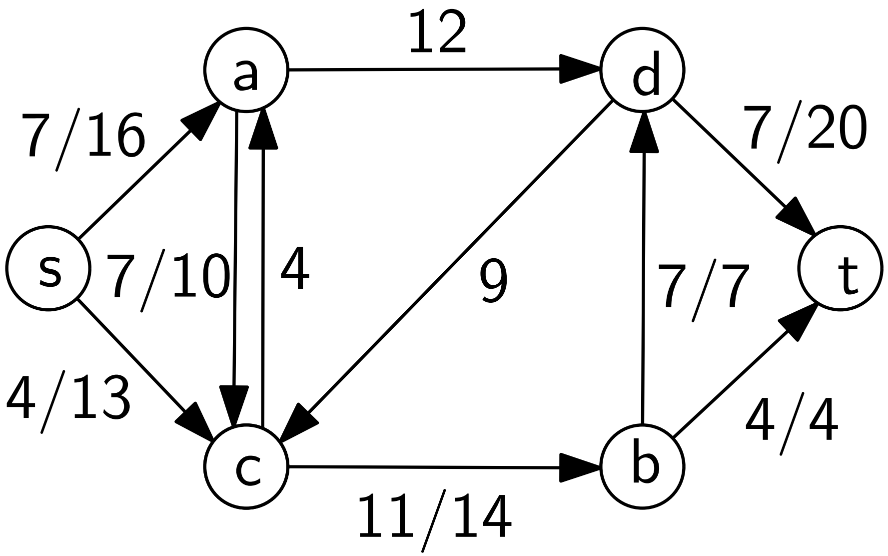

a
d
s
t
c
b
12
16
20
10
4
9
7
13
4
14
a
d
s
t
c
b
12
16
20
10
4
9
7
13
4
14
a
d
s
t
c
b
12
16
20
10
4
9
7
4/13
4/4
4/14
a
d
s
t
c
b
12
16
20
10
4
9
7
9
4
4
10
4



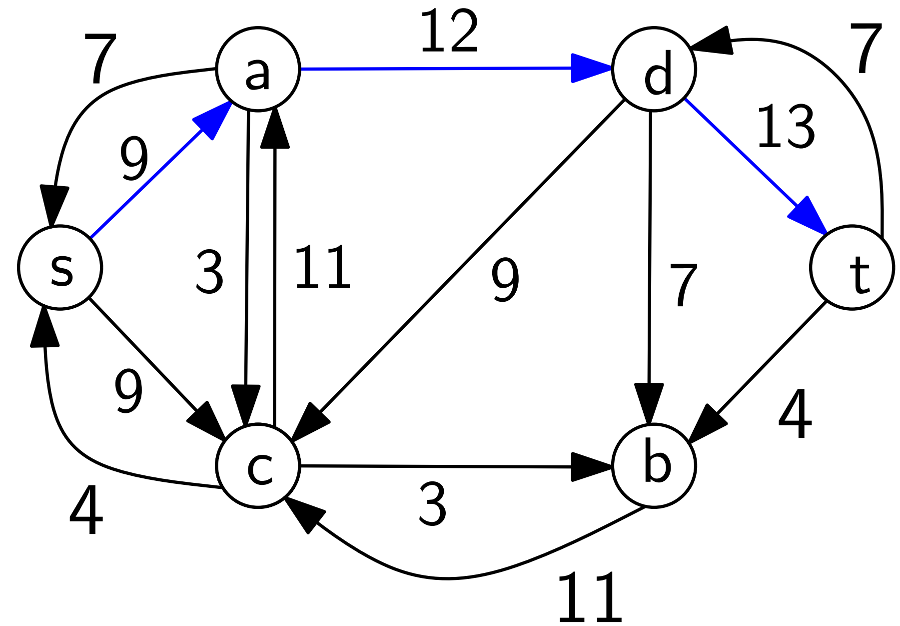
s
a
c
d
b
t
7/16
4/13
7/10
4
12
9
7/7
7/20
4/4
11/14


s
a
c
d
b
t
7/16
4/13
7/10
4
12
9
11/14
7/7
7/20
4/4

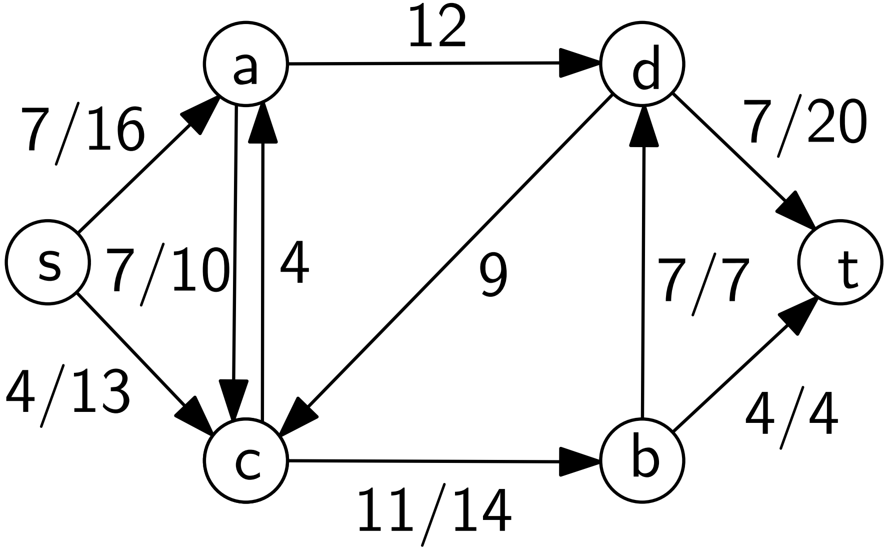

s
a
c
d
b
t
7
9
9
4
3
11
12
9
3
11
7
13
7
4

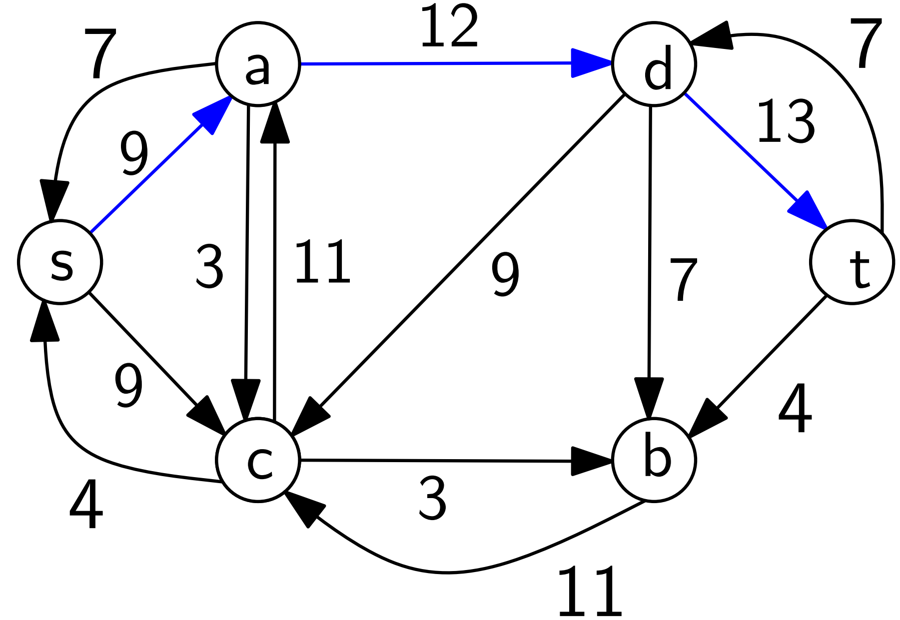


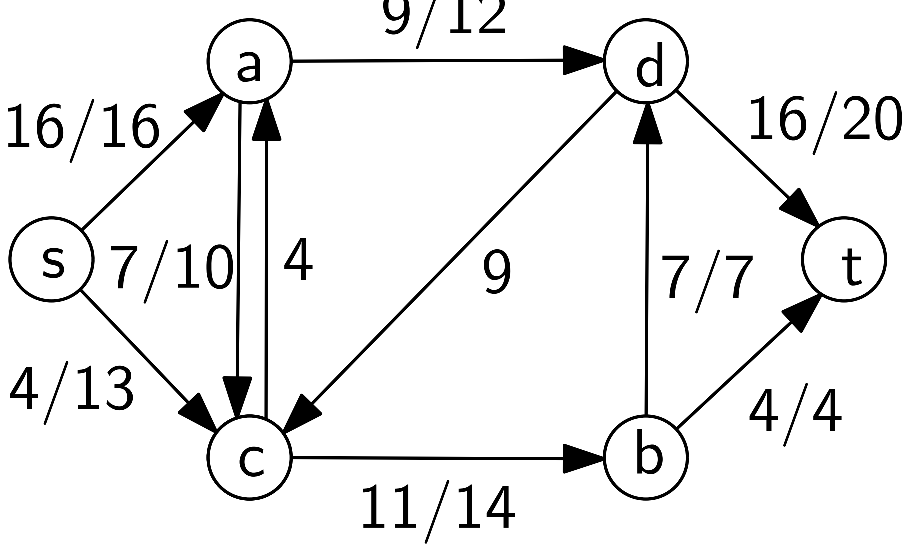
a
d
s
t
c
b
12
7/16
7/20
7/10
4
9
7/7
4/13
4/4
11/14

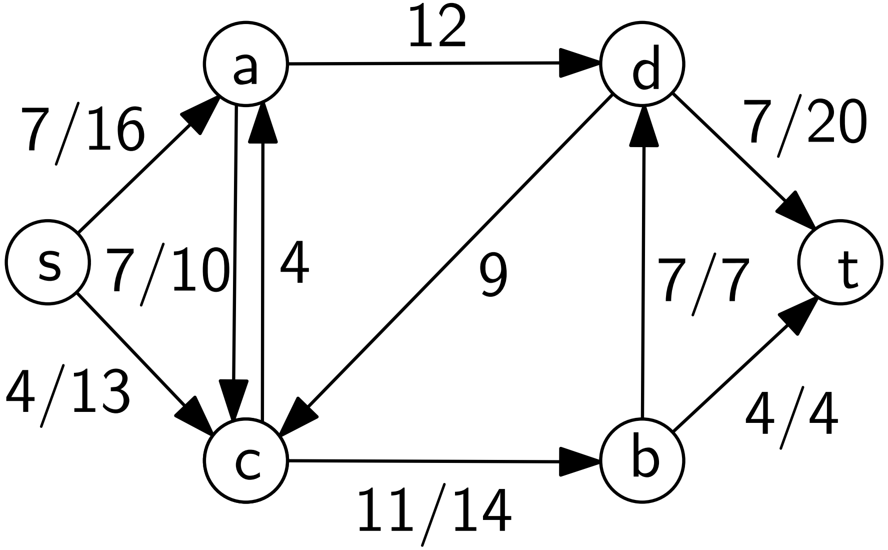


a
d
s
t
c
b
7
12
7
9
13
3
11
9
7
9
4
3
4
11

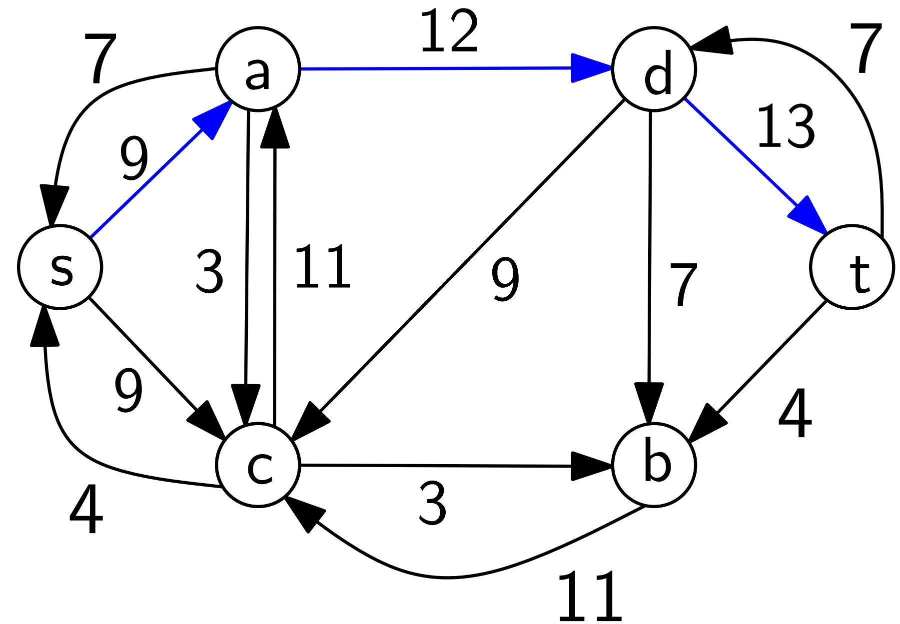

s
a
c
d
b
t
7/16
4/13
7/10
4
12
9
11/14
7/7
7/20
4/4


s
a
c
d
b
t
7
9
3
11
4
9
12
9
3
11
7
13
7
4

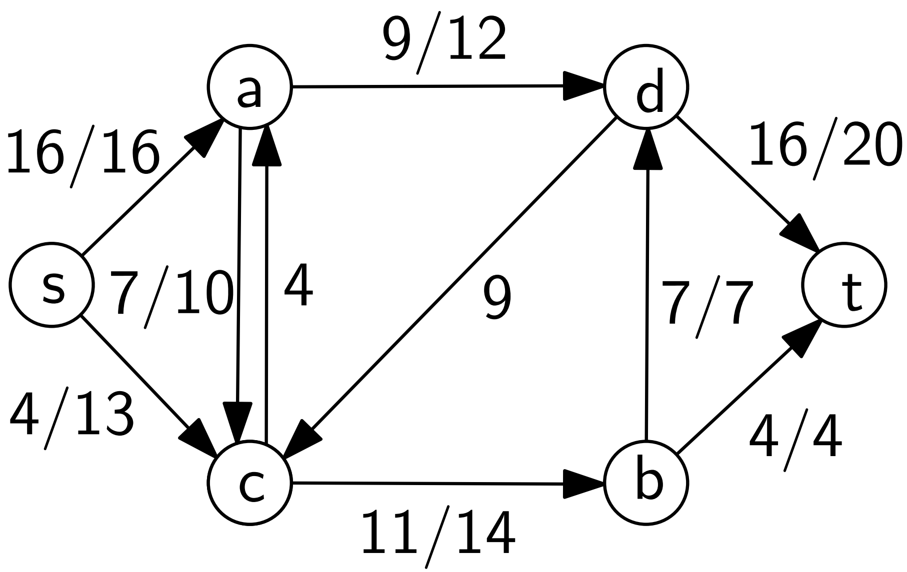

s
a
c
d
b
t
16/16
4/13
7/10
4
9/12
9
11/14
7/7
16/20
4/4



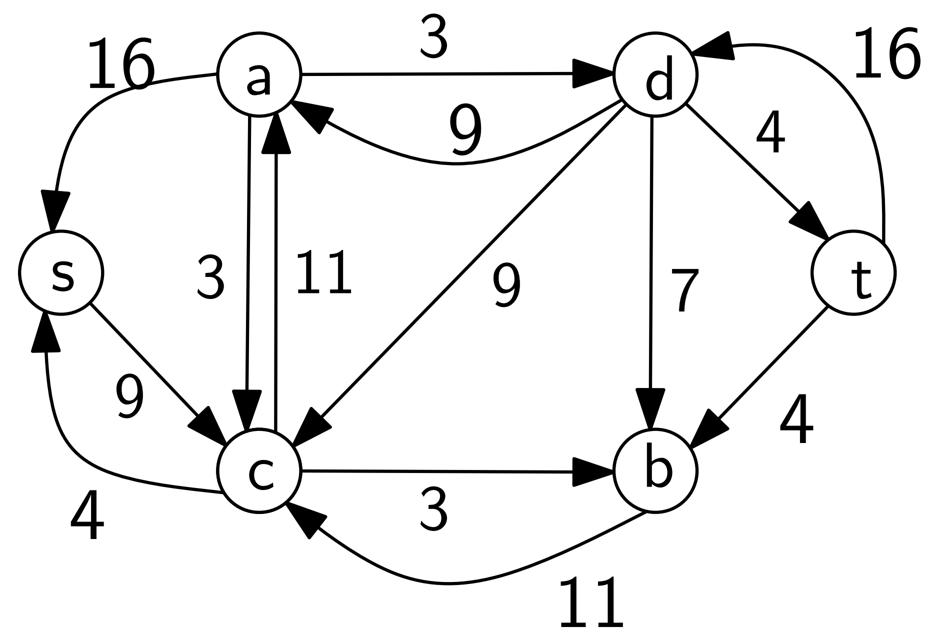
a
d
s
t
c
b
12
7/16
7/20
7/10
4
9
7/7
4/13
4/4
11/14



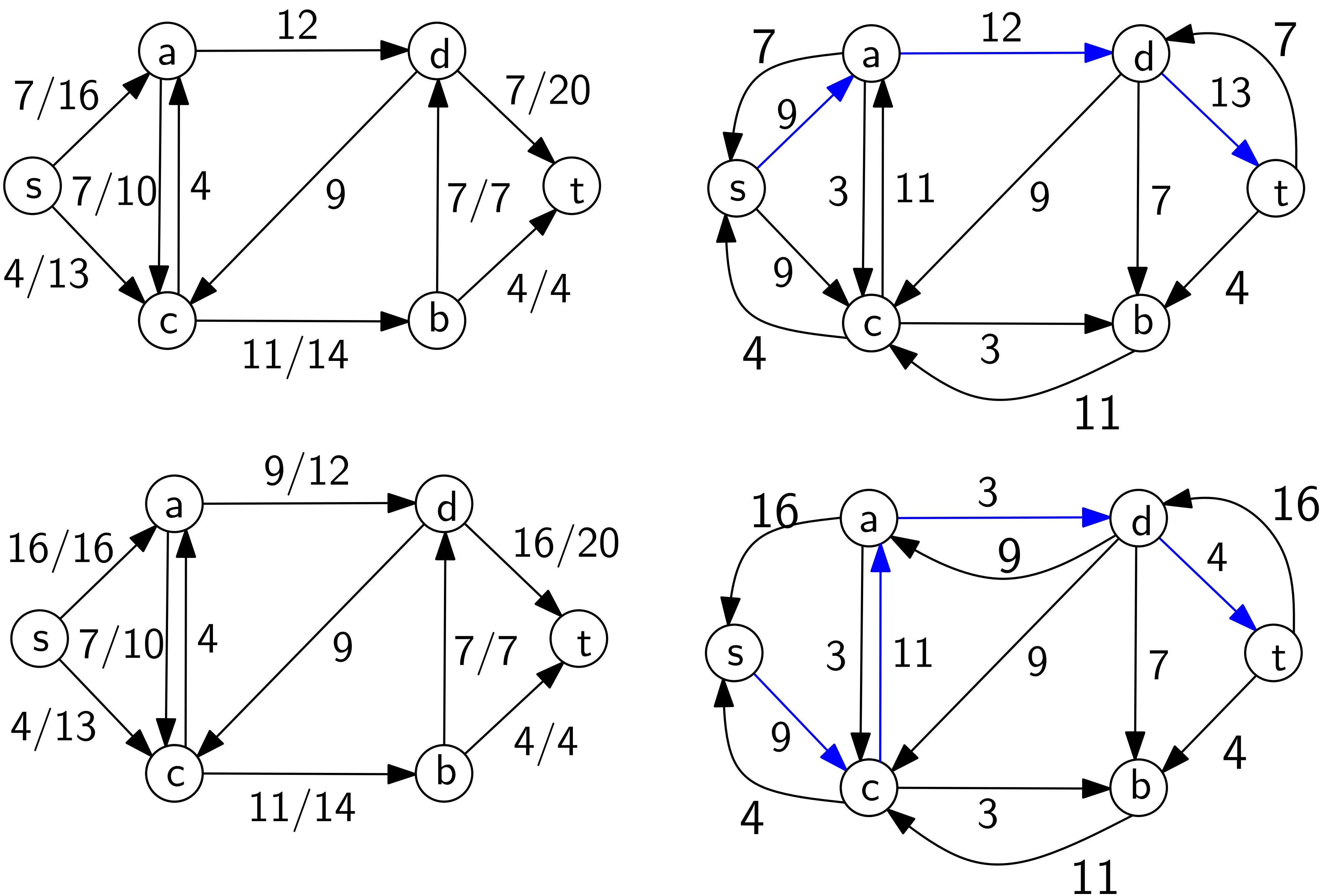
a
d
s
t
c
b
12
7
7
9
13
3
11
9
7
9
4
4
3
11



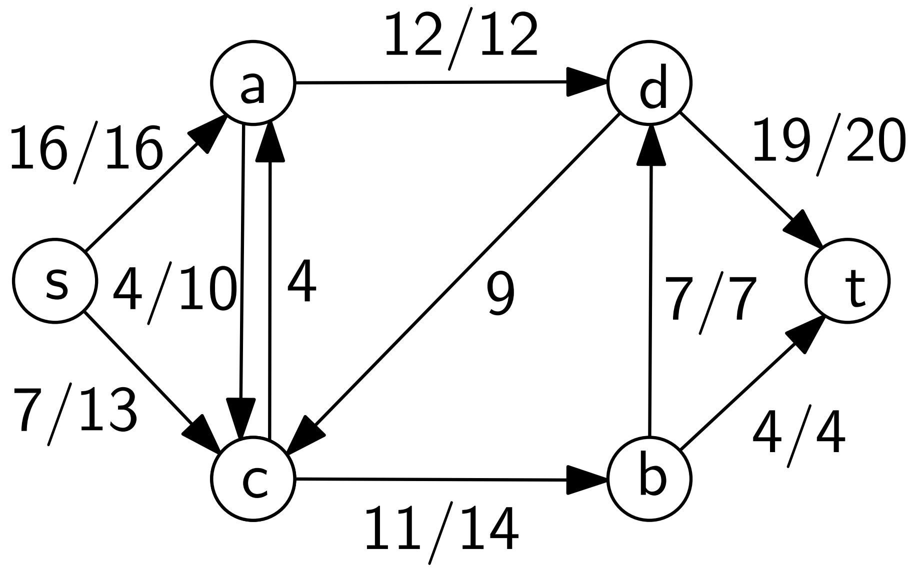
a
d
s
t
c
b
9/12
16/16
16/20
7/10
4
9
7/7
4/13
4/4
11/14

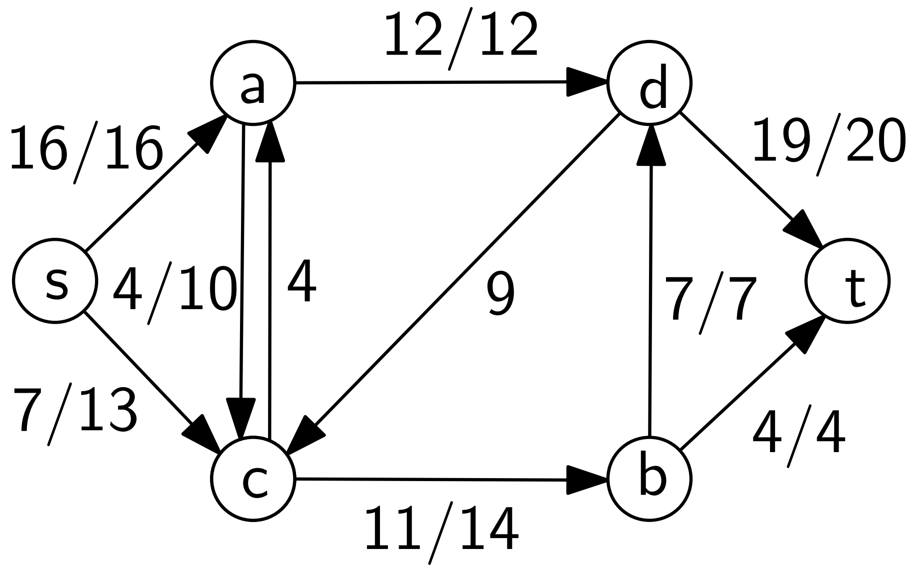


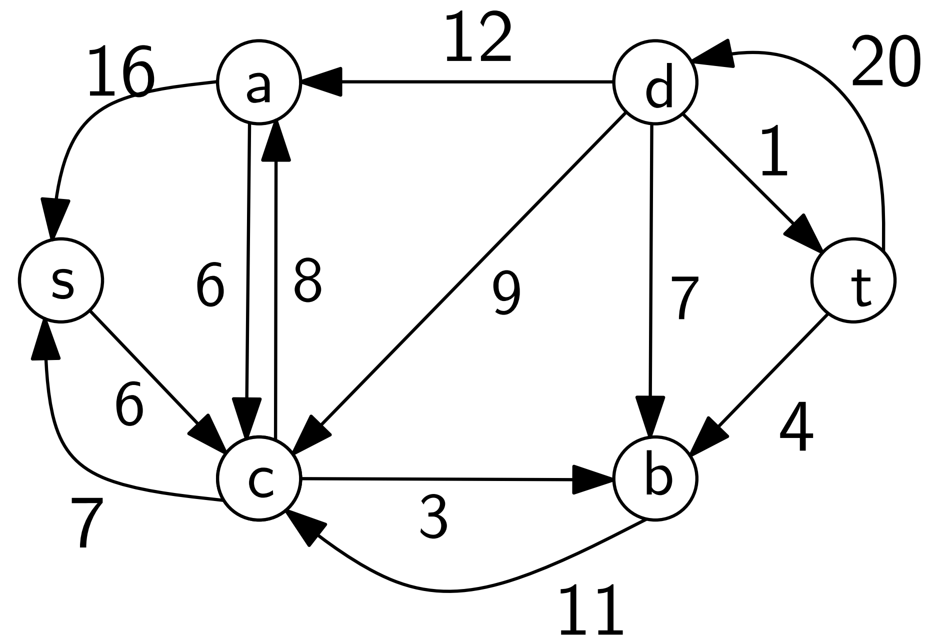
a
d
s
t
c
b
3
16
16
9
4
3
11
9
7
9
4
4
3
11

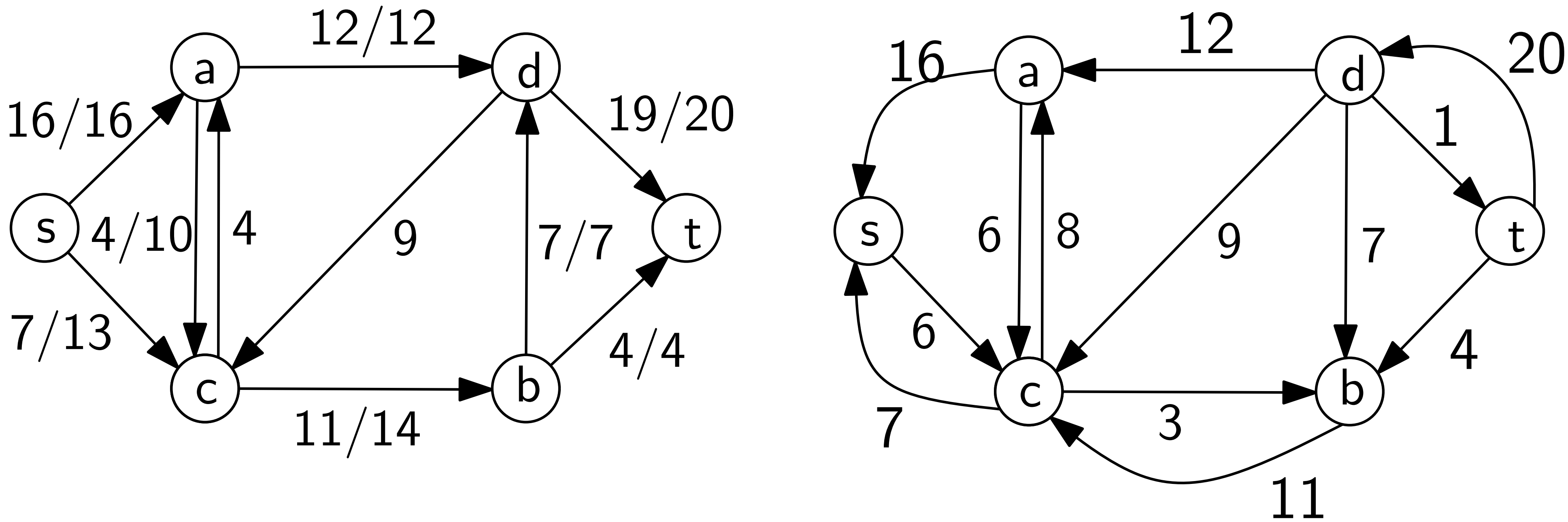


a
d
s
t
c
b
12
7/16
7/20
7/10
4
9
7/7
4/13
4/4
11/14
7
12
13
9
3
11
9
7
9
4
4
3
11
9/12
16/16
16/20
7/10
4
9
7/7
4/13
4/4
11/14
16
3
16
9
4
3
11
9
7
9
4
4
3
11

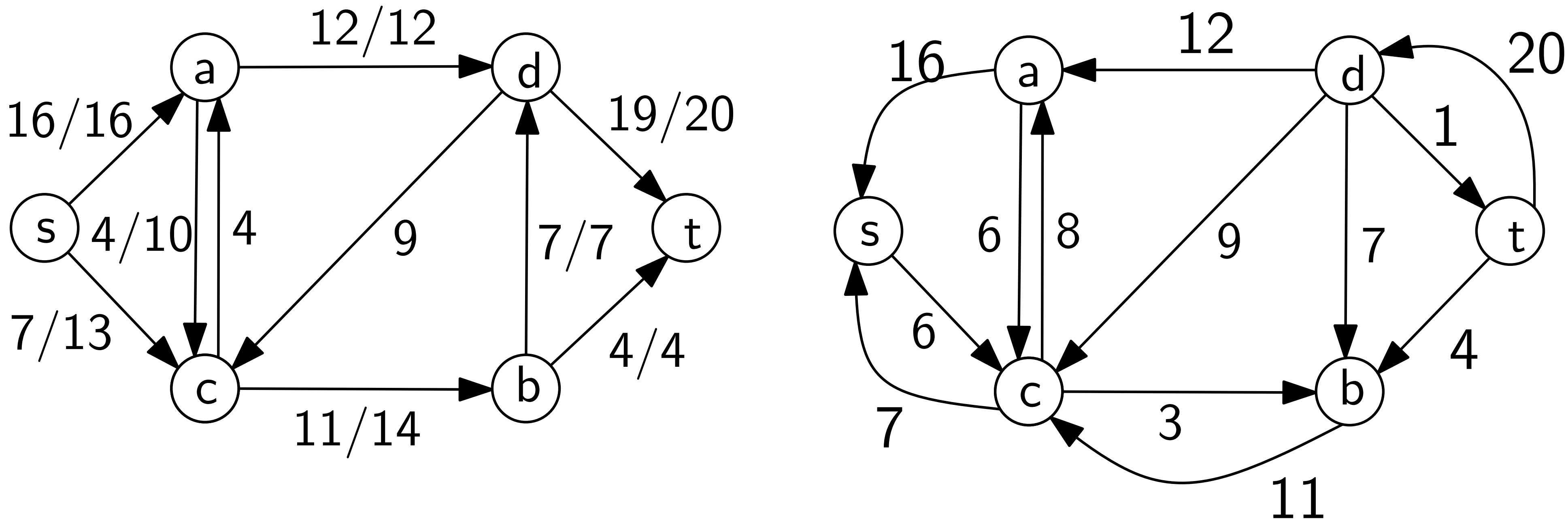

a
d
s
t
c
b
12/12
16/16
19/20
4/10
4
9
7/7
7/13
4/4
11/14


12/12
a
d
16/16
19/20
s
4/10
4
9
7/7
t
7/13
4/4
c
b
11/14


12
16
a
d
20
1
s
6
8
9
7
t
6
4
c
b
7
3
11



No more augmenting path



No more augmenting path

Maximum flow value = 23

Cut

A cut is a partition of the vertex set into two disjoint subsets S and T, written as (S, T), such that $s \in S$, $t \in T$, and $S \cup T$ is the entire vertex set.

Cut

A cut is a partition of the vertex set into two disjoint subsets S and T, written as (S, T), such that $s \in S$, $t \in T$, and $S \cup T$ is the entire vertex set.

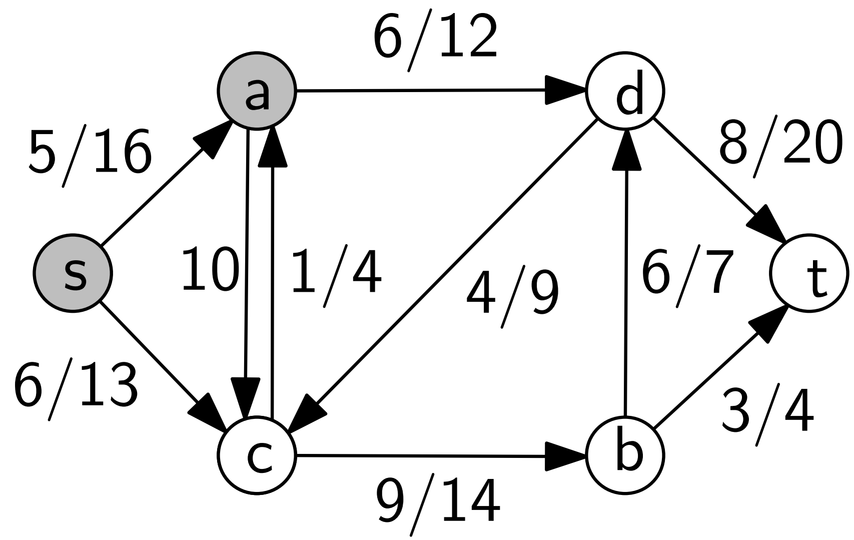
The capacity of a cut (S, T) is defined as $\sum_{u \in S,\, v \in T} c(u, v)$.

Cut

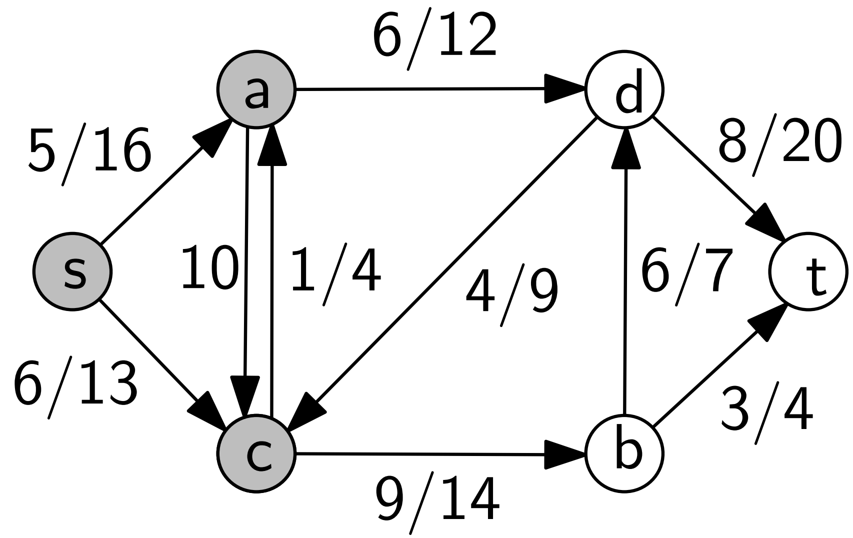
A cut is a partition of the vertex set into two disjoint subsets S and T, written as (S, T), such that $s \in S$, $t \in T$, and $S \cup T$ is the entire vertex set.

The capacity of a cut (S, T) is defined as $\sum_{u \in S,\, v \in T} c(u, v)$.

Given a flow f, the net flow across a cut (S, T) is defined as $\sum_{u \in S,\, v \in T} f(u, v)$. Notice that such a value $f(u, v)$ may be negative, but $f(u, v) \leq c(u, v)$ is always true.



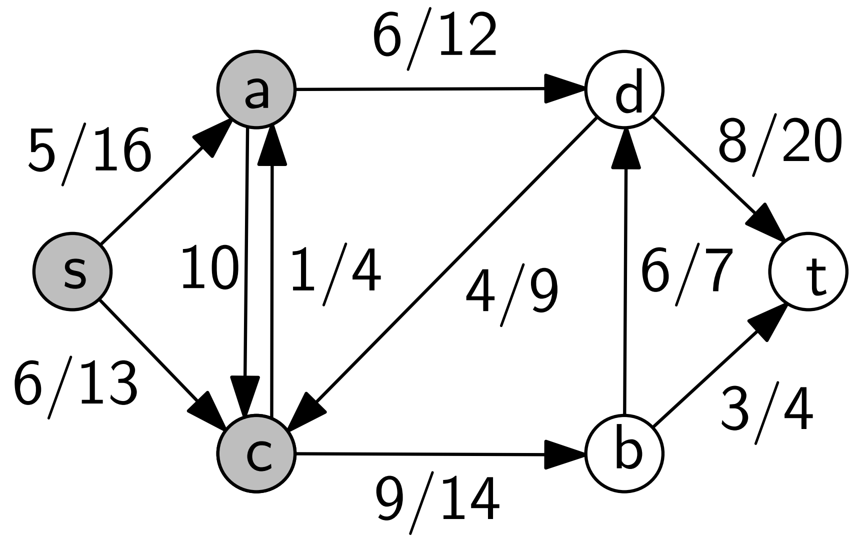
Cut capacity $= 13 + 10 + 12 = 35$.
Net flow across cut $= 6 - 1 + 6 = 11$.



Same flow but different cut.

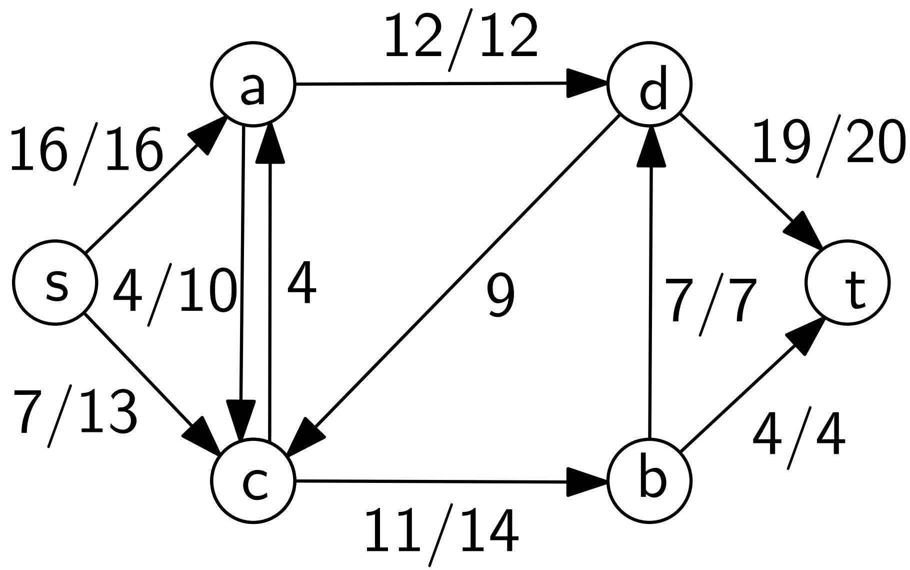
Cut capacity $= 12 + 14 = 26$.

Net flow across cut $= 6 - 4 + 9 = 11$.

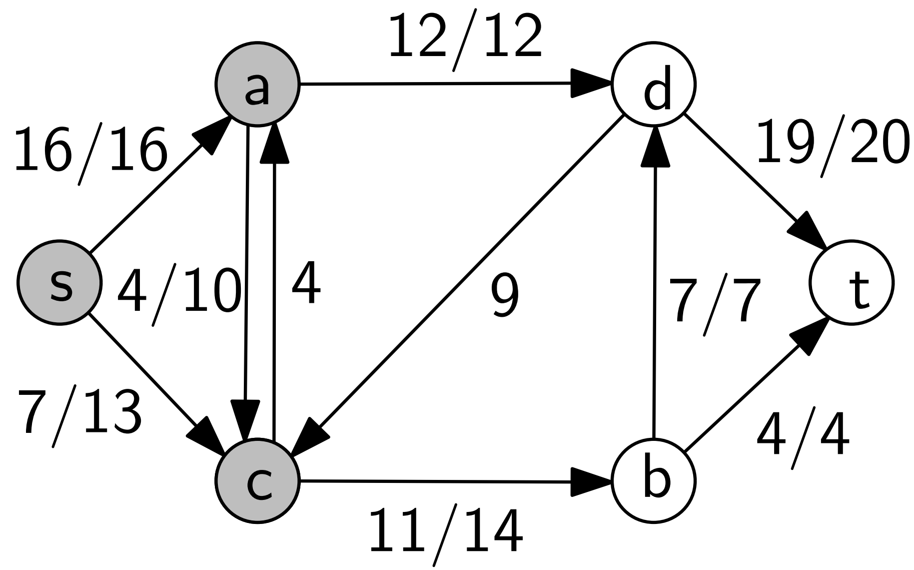


Same flow but different cut.
Cut capacity $= 12 + 14 = 26$.
Net flow across cut $= 6 - 4 + 9 = 11$.

Different cuts may have different capacities.
Given the same flow, the net flow across different cuts are always the same because it is the value of the flow.

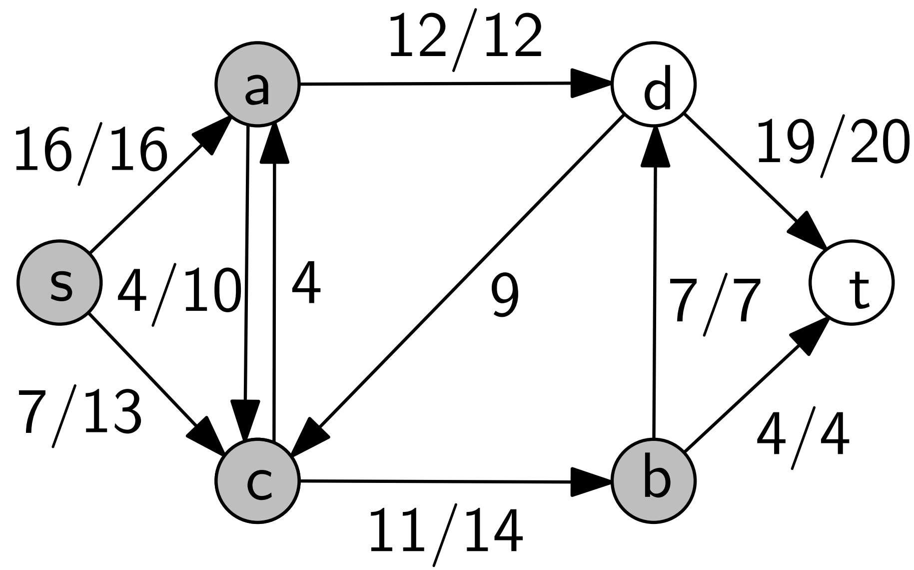


Recall that this is the maximum flow and its value is 23.

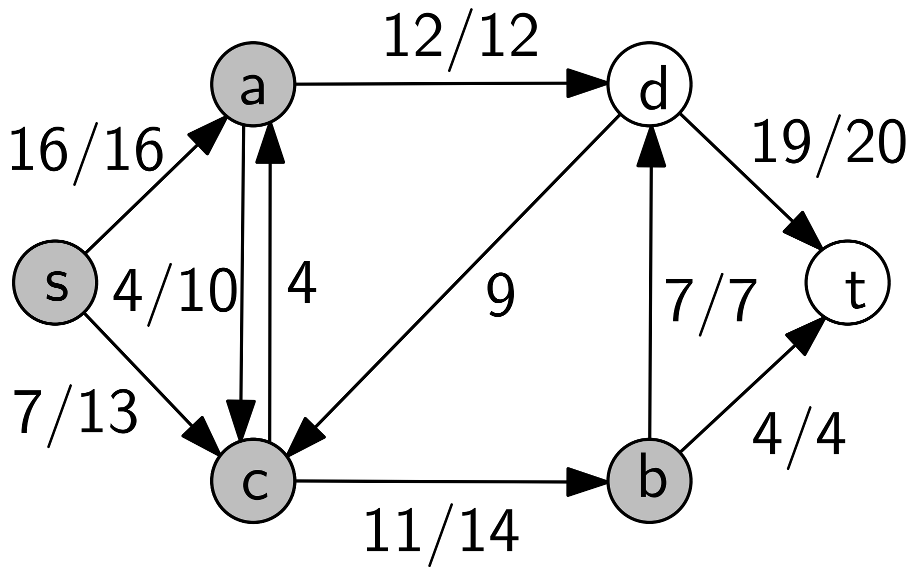


Recall that this is the maximum flow and its value is 23.

Cut capacity $= 12 + 14 = 26$.

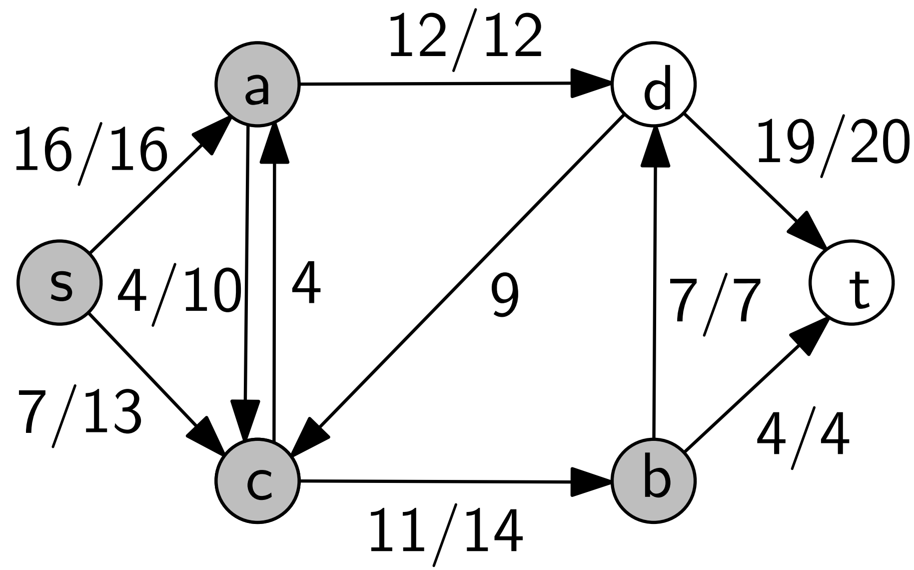


Recall that this is the maximum flow and its value is 23.



Recall that this is the maximum flow and its value is 23.

Cut capacity $= 12 + 7 + 4 = 23 =$ flow value.



Recall that this is the maximum flow and its value is 23.

Cut capacity $= 12 + 7 + 4 = 23 =$ flow value.

Since the cut capacity is an upper bound on the net flow across the cut, this flow value 23 must be the maximum possible.

Max Flow Min Cut Theorem

Let G be a flow network. Let f be a flow in G. The following statements are equivalent:

1. f is a maximum flow.
2. G_f has no augmenting path.
3. There exists a cut (S, T) with capacity equal to $|f|$.

Max Flow Min Cut Theorem

Let G be a flow network. Let f be a flow in G. The following statements are equivalent:

1. f is a maximum flow.
2. G_f has no augmenting path.
3. There exists a cut (S, T) with capacity equal to $|f|$.

Proof.
$(1) \Rightarrow (2)$: If G_f has an augumenting path, we can use it to increase the flow value. But then f is not a maximum flow.

$(3) \Rightarrow (1)$: Follows from the fact that the capacity of a cut is an upper bound on the value of the net flow across any cut.

$(3) \Rightarrow (1)$: Follows from the fact that the capacity of a cut is an upper bound on the value of the net flow across any cut.

$(2) \Rightarrow (3)$:
Let S be the set of vertices that are reachable from s in G_f. That is, there is a path from s to every vertex $u \in S$. Let T be the other vertices of G_f.

$(3) \Rightarrow (1)$: Follows from the fact that the capacity of a cut is an upper bound on the value of the net flow across any cut.

$(2) \Rightarrow (3)$:
Let S be the set of vertices that are reachable from s in G_f. That is, there is a path from s to every vertex $u \in S$. Let T be the other vertices of G_f.

Note that $s \in S$ and $t \in T$ because G_f has no augmenting path. So (S, T) is a cut.

$(3) \Rightarrow (1)$: Follows from the fact that the capacity of a cut is an upper bound on the value of the net flow across any cut.

$(2) \Rightarrow (3)$:
Let S be the set of vertices that are reachable from s in G_f. That is, there is a path from s to every vertex $u \in S$. Let T be the other vertices of G_f.

Note that $s \in S$ and $t \in T$ because G_f has no augmenting path. So (S, T) is a cut.

For every $u \in S$ and every $v \in T$, the edge (u, v) does not exists in G_f.

$(3) \Rightarrow (1)$: Follows from the fact that the capacity of a cut is an upper bound on the value of the net flow across any cut.

$(2) \Rightarrow (3)$:
Let S be the set of vertices that are reachable from s in G_f. That is, there is a path from s to every vertex $u \in S$. Let T be the other vertices of G_f.

Note that $s \in S$ and $t \in T$ because G_f has no augmenting path. So (S, T) is a cut.

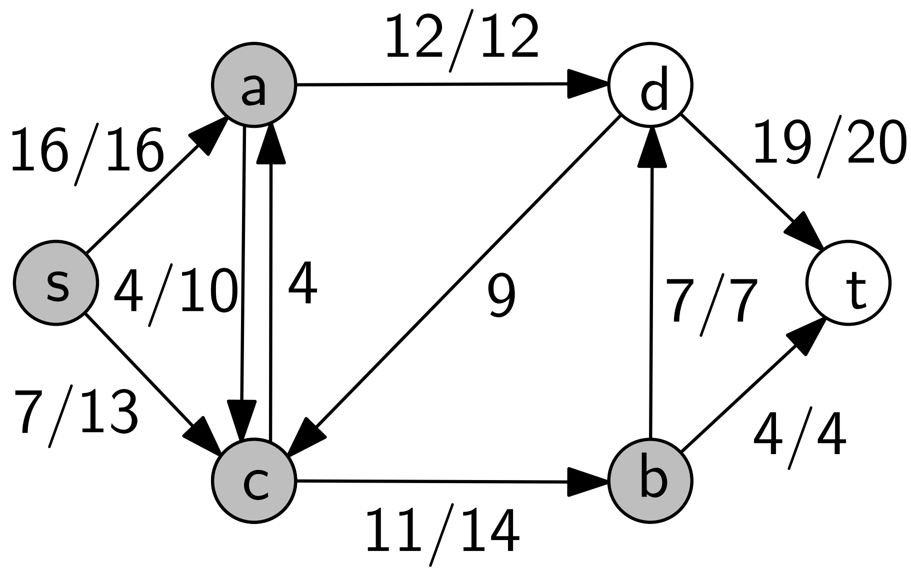
For every $u \in S$ and every $v \in T$, the edge (u, v) does not exists in G_f. Therefore, $c_f(u, v) = 0 \Rightarrow c(u, v) = f(u, v)$.

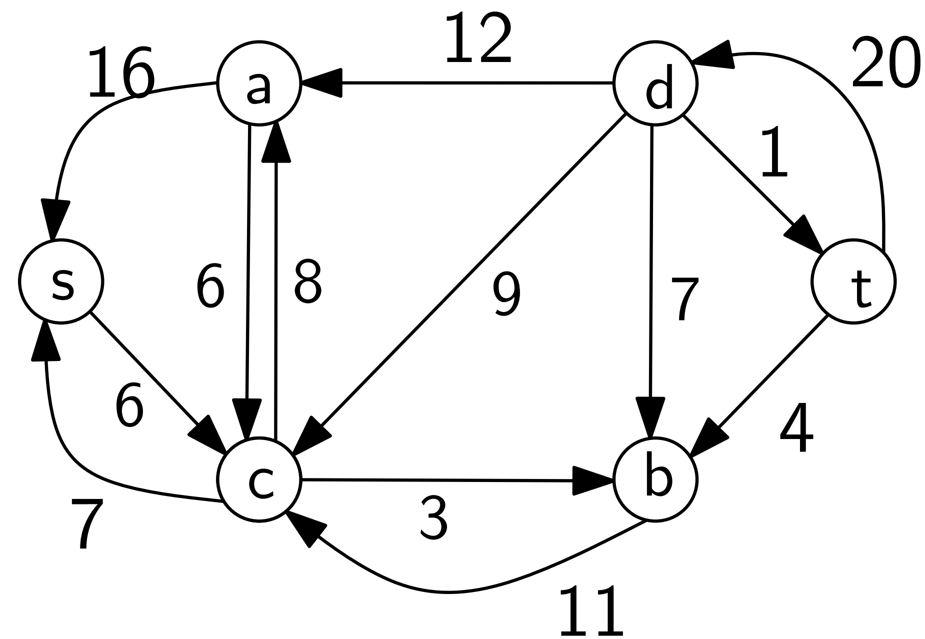
$(3) \Rightarrow (1)$: Follows from the fact that the capacity of a cut is an upper bound on the value of the net flow across any cut.

$(2) \Rightarrow (3)$:
Let S be the set of vertices that are reachable from s in G_f. That is, there is a path from s to every vertex $u \in S$. Let T be the other vertices of G_f.

Note that $s \in S$ and $t \in T$ because G_f has no augmenting path. So (S, T) is a cut.

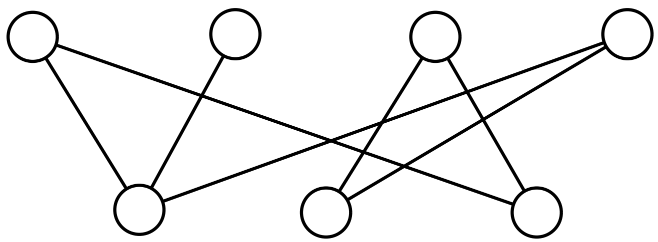
For every $u \in S$ and every $v \in T$, the edge (u, v) does not exists in G_f. Therefore, $c_f(u, v) = 0 \Rightarrow c(u, v) = f(u, v)$. Hence, net flow $\sum_{u \in S,\, v \in T} f(u, v)$ is equal to cut capacity $\sum_{u \in S, v \in T} c(u, v)$.





The vertices a, b and c are the vertices reachable from s in the residual network G_f.

Bipartitie Matching

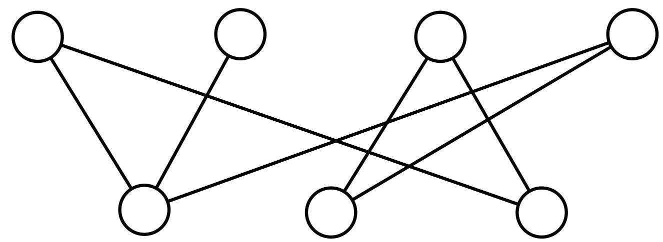


students

hall single rooms

edges indicate choices

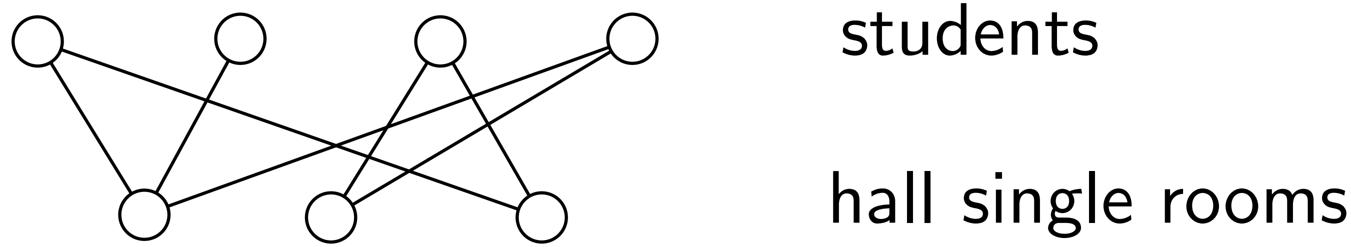
Bipartitie Matching



edges indicate choices

How to assign students to the rooms in order to maximize the number of assigned pairs? The constraint is that every student can be assigned to at most one room. Every room is assigned to at most one student.

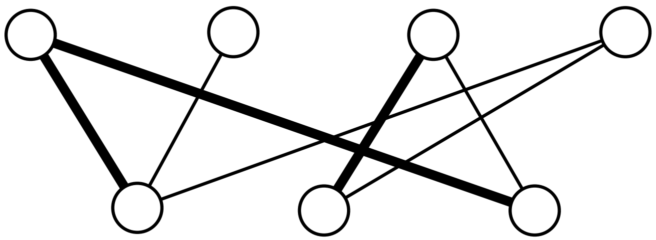
Bipartitie Matching



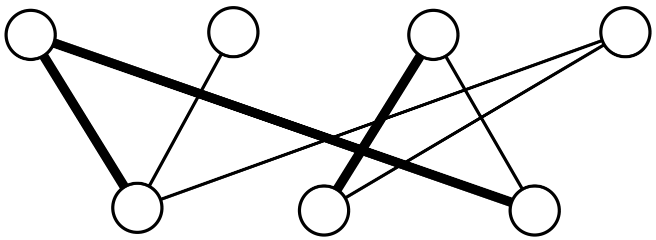
edges indicate choices

How to assign students to the rooms in order to maximize the number of assigned pairs? The constraint is that every student can be assigned to at most one room. Every room is assigned to at most one student.

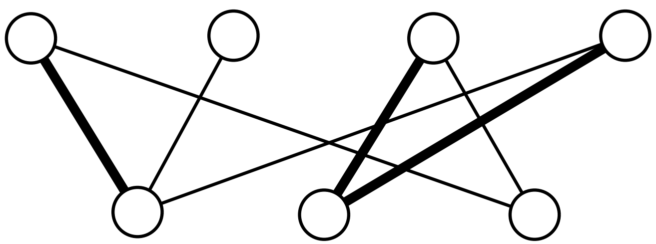
The input graph is called a bipartitie graph. The edges in the solution form a maximum bipartite matching. The edges are called matching edges.



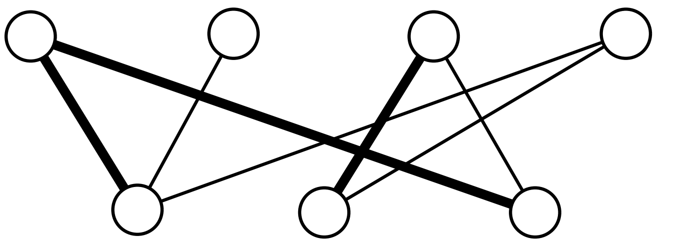
Not a bipartitie matching



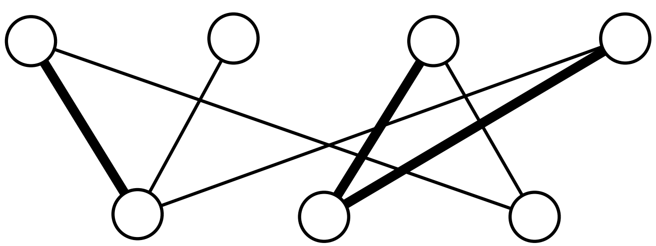
Not a bipartitie matching



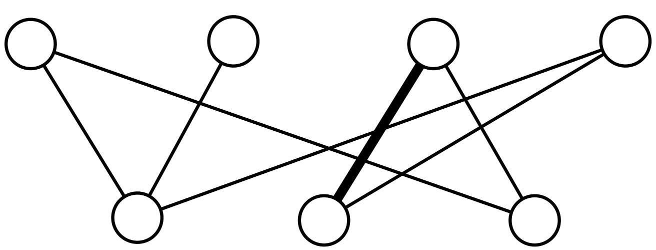
Not a bipartitie matching



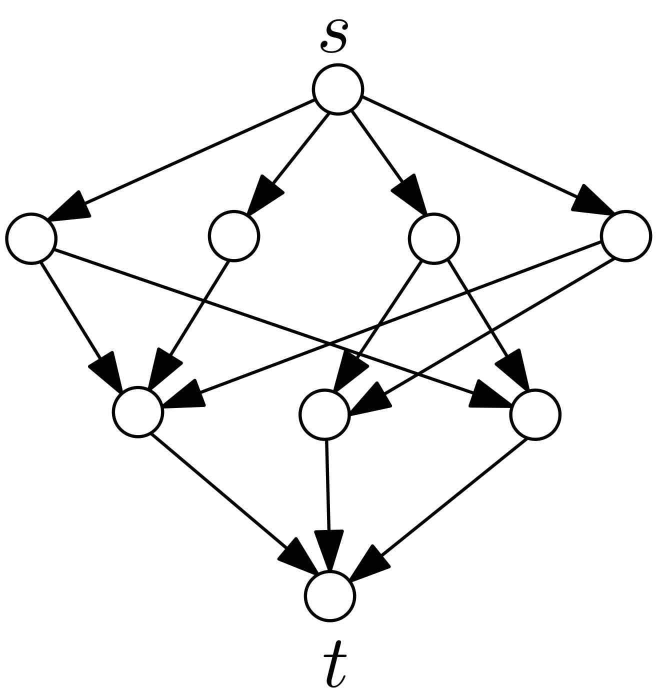
Not a bipartitie matching



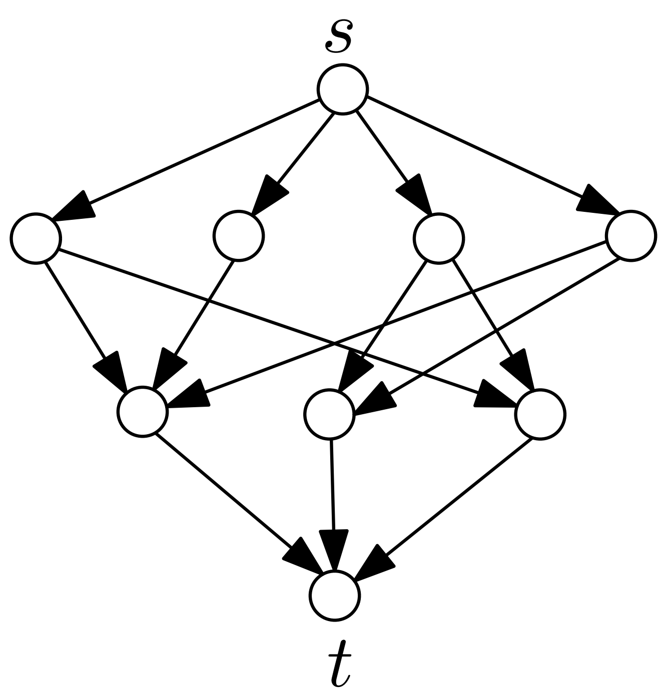
Not a bipartitie matching



A bipartitite matching, but not maximum.



Every edge has capacity 1.



Every edge has capacity 1.

Find the maximum flow.

Edges between the original vertices that have flow values 1 are the maximum bipartite matching edges.