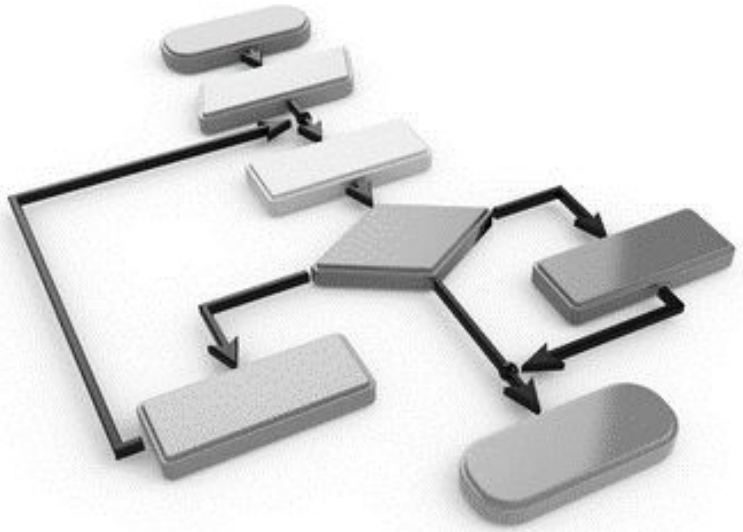


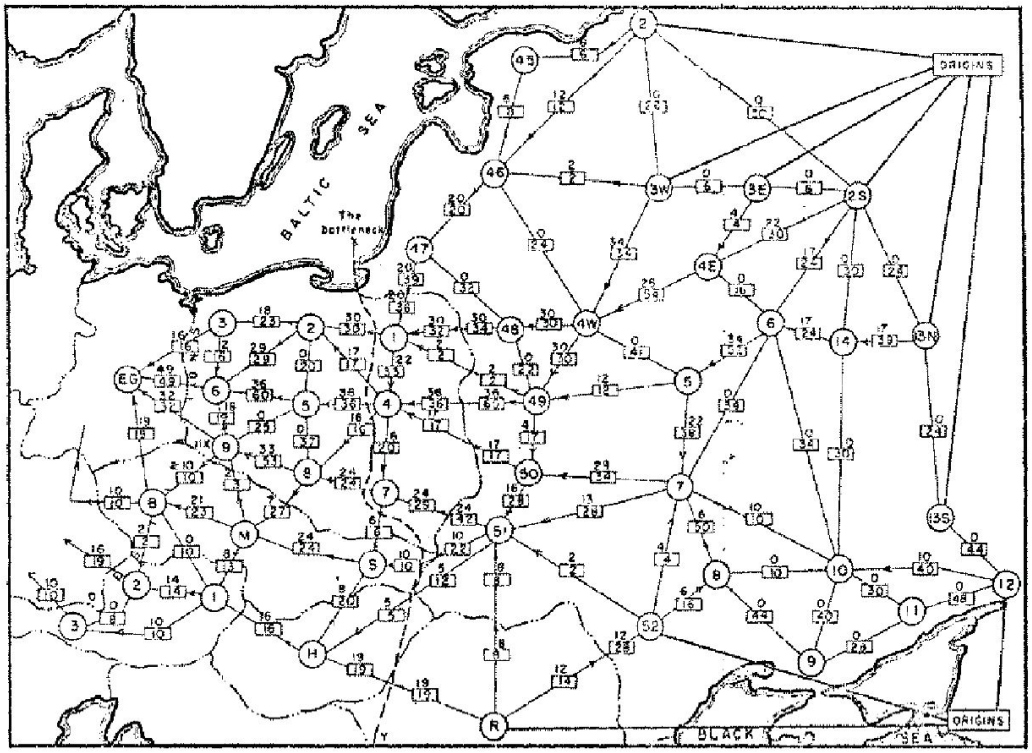
Lecture 17

Graph-Based Algorithms

CSE373: Design and Analysis of Algorithms

Background

Soviet Rail Network, 1955



Reference: *On the history of the transportation and maximum flow problems.* Alexander Schrijver in Math Programming, 91: 3, 2002.

- material coursing through a system from a source to a sink

Background

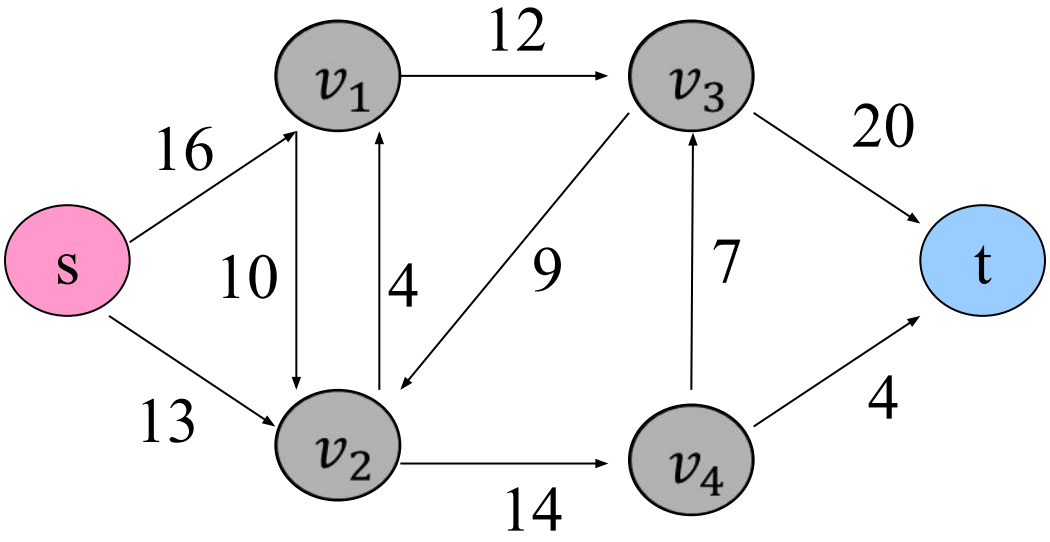
How can we maximize the flow in a network from a source or set of sources to a destination or set of destinations?

The problem reportedly rose to prominence in relation to the rail networks of the Soviet Union, during the 1950's. The US wanted to know how quickly the Soviet Union could get supplies through its rail network to its satellite states in Eastern Europe.

In addition, the US wanted to know which rails it could destroy most easily to cut off the satellite states from the rest of the Soviet Union. It turned out that these two problems were closely related, and that solving the **max flow problem** also solves the **min cut problem** of figuring out the cheapest way to cut off the Soviet Union from its satellites.

Flow Networks

- A flow network G=(V,E): a directed graph, where each edge (u,v)∈E has a nonnegative capacity c(u,v)>=0. If (u,v)∉E, we assume that c(u,v)=0.
  - Edges represent pipes that carry flow
- Two distinct vertices: a source s and a sink t.

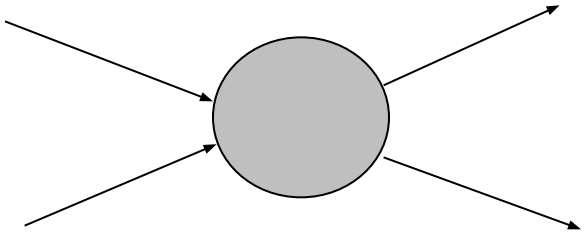


Flow

Given $G = (V, E)$, a flow network with capacity function c, a source s and a sink t, a flow in G is a real-valued function $f: V * V \rightarrow R$ satisfying the following two properties:

Capacity constraint: $\forall u,v \in V,\ f(u,v) \leq c(u,v)$

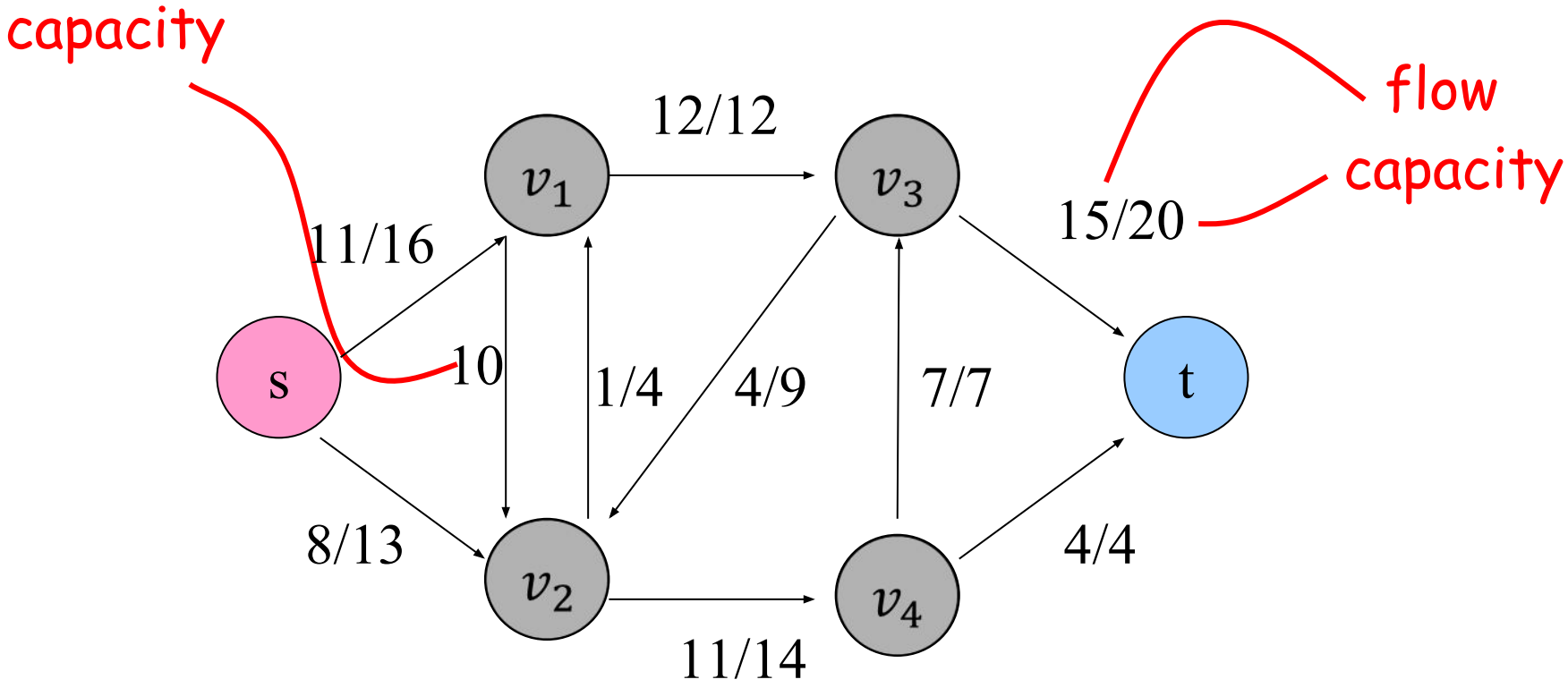
Flow conservation: $\forall u \in V - \{s,t\},\ \sum_{v \in V} f(u,v) = 0$



- Notes:
  - Function f also satisfies skew symmetry property: f(u,v)=-f(v,u) for any u,v in V
    - It implies that f(u,u)=0.
  - We show only the *positive* flows in the flow network.

Flow: Example

$$f(v_2, v_1) = 1, \qquad c(v_2, v_1) = 4.$$
$$f(v_1, v_2) = -1, \qquad c(v_1, v_2) = 10.$$
$$f(v_3, s) + f(v_3, v_1) + f(v_3, v_2) + f(v_3, v_4) + f(v_3, t) = 0 + (-12) + 4 + (-7) + 15 = 0$$



Value of a Flow

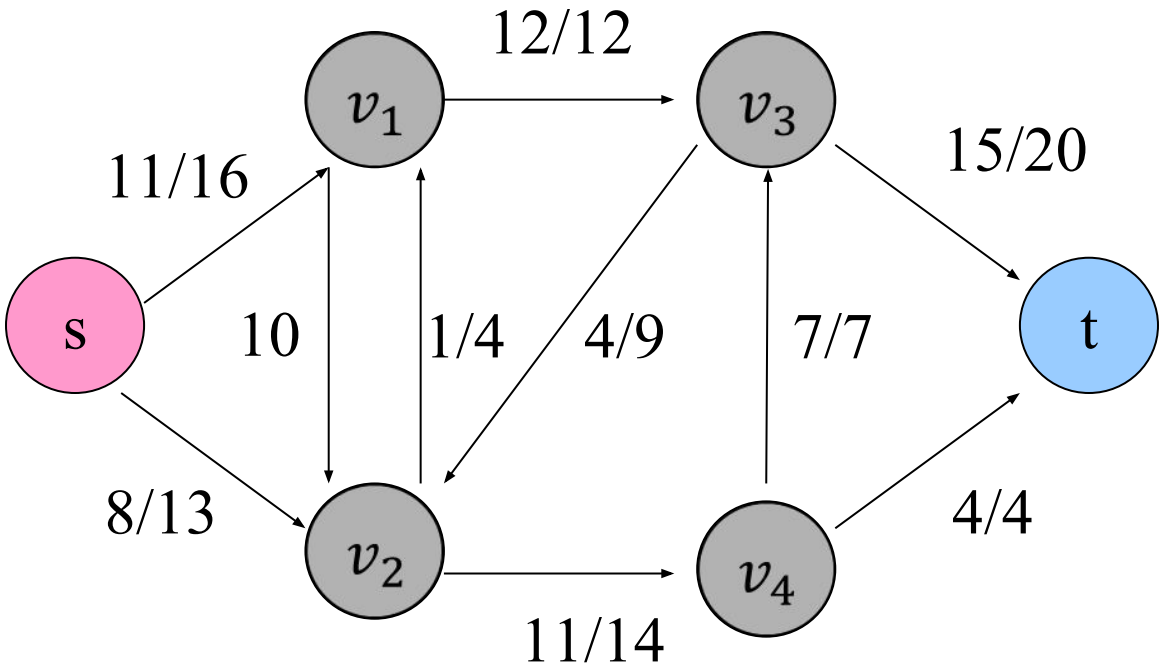
- The total flow from source to any other vertices.

$$|f| = \sum_{v \in V} f(s, v)$$

- The same as the total flow from any vertices to the sink.
  - The total flow leaving s = the total flow arriving at t.

$$| f | = \sum_{v \in V} f(s, v) = \sum_{v \in V} f(v, t)$$

Value of a Flow



A flow f in G with value $|f| = 19$.

$$|f| = f(s,v_1) + f(s,v_2) + f(s,v_3) + f(s,v_4) + f(s,t)$$
$$= 11 + 8 + 0 + 0 + 0 = 19$$
$$|f| = f(s,t) + f(v_1,t) + f(v_2,t) + f(v_3,t) + f(v_4,t)$$
$$= 0 + 0 + 0 + 15 + 4 = 19$$

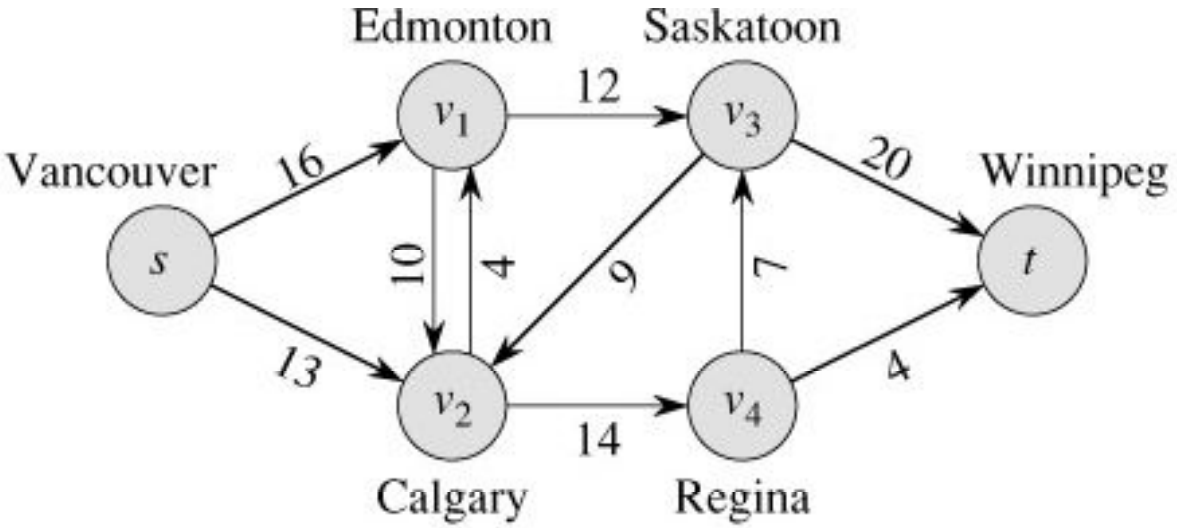
The Problem

Use a graph to model material that flows through conduits.

Each edge represents one conduit, and has a capacity, which is an upper bound on the flow rate, in units/time.

Can think of edges as pipes (or, roads) of different sizes (or, widths).

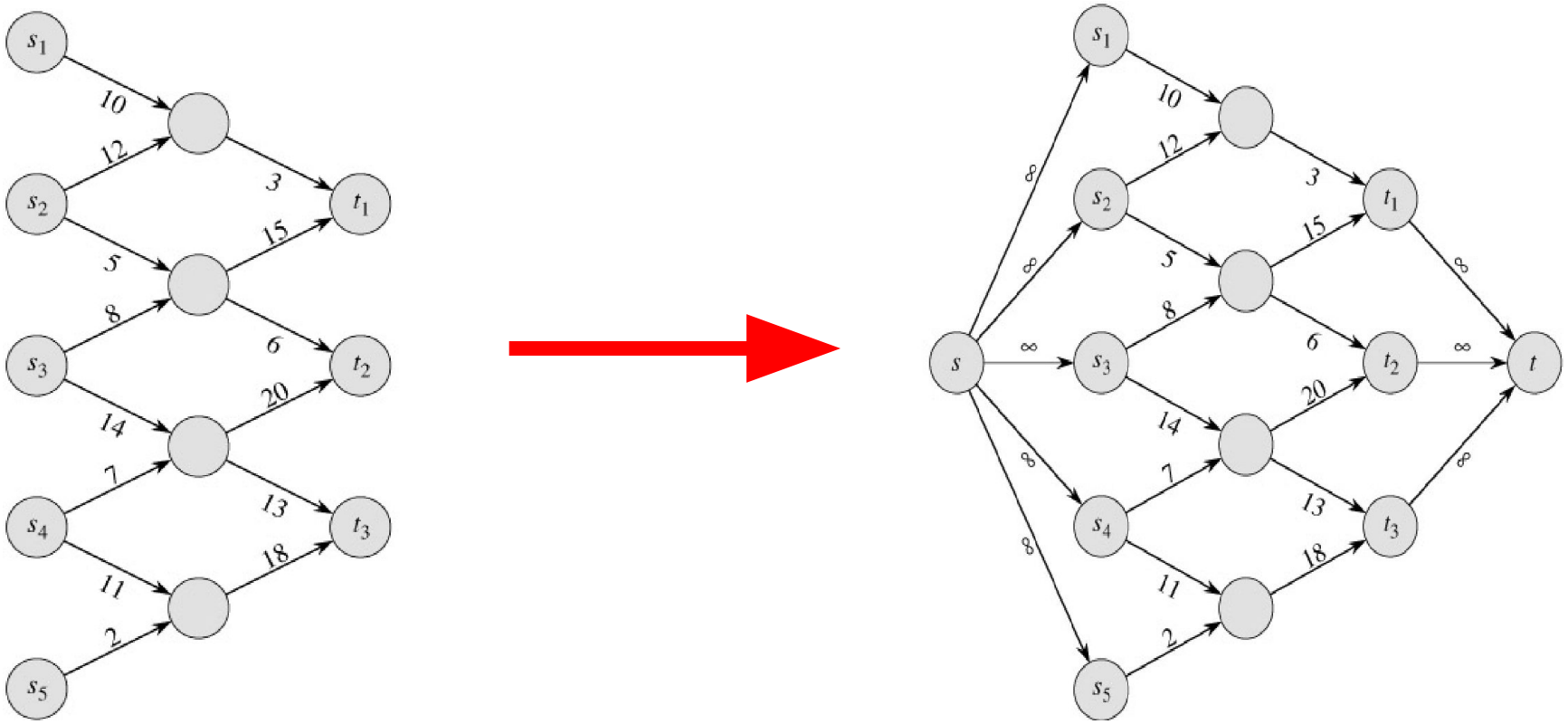
Calculate the max rate that we can ship material from a designated source to a designated sink. That is, calculate the maximum flow.



Multiple Sources Network

We have several sources and several targets. We want to maximize the total flow from all sources to all targets.

We can reduce this problem to ordinary max-flow problem by attaching a supersource to each source and a supersink to each sink as below:



Residual Networks

The residual capacity of an edge *(u,v)* in a network with a flow f is given by:

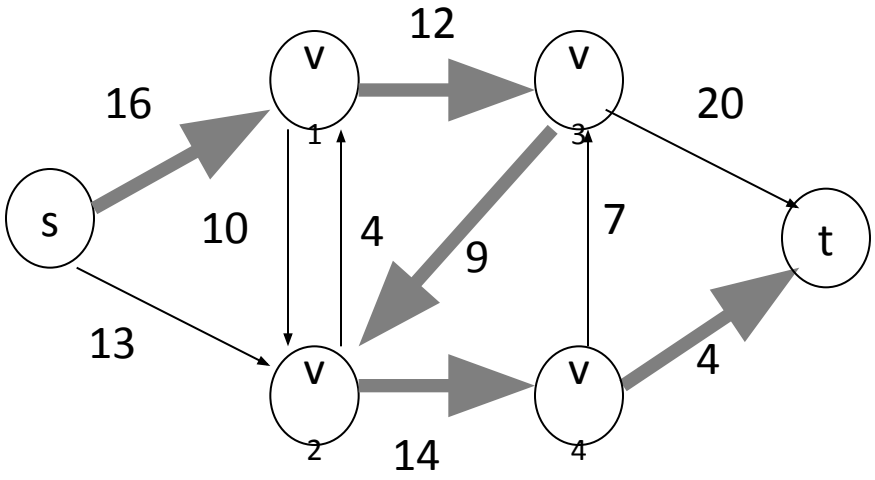
$$c_f(u,v) = c(u,v) - f(u,v)$$

- The residual network of a graph *G* induced by a flow *f* is the graph including only the edges with positive residual capacity, i.e.,

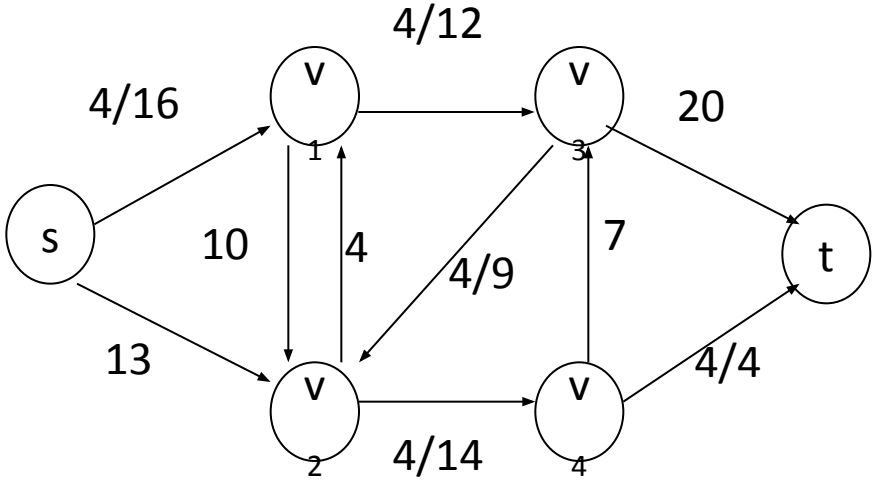
$$G_f = (V, E_f), \text{ where } E_f = \{(u,v) \in V \times V : c_f(u,v) > 0\}$$

- Intuitively a residual network consists of edges that can admit more flow.

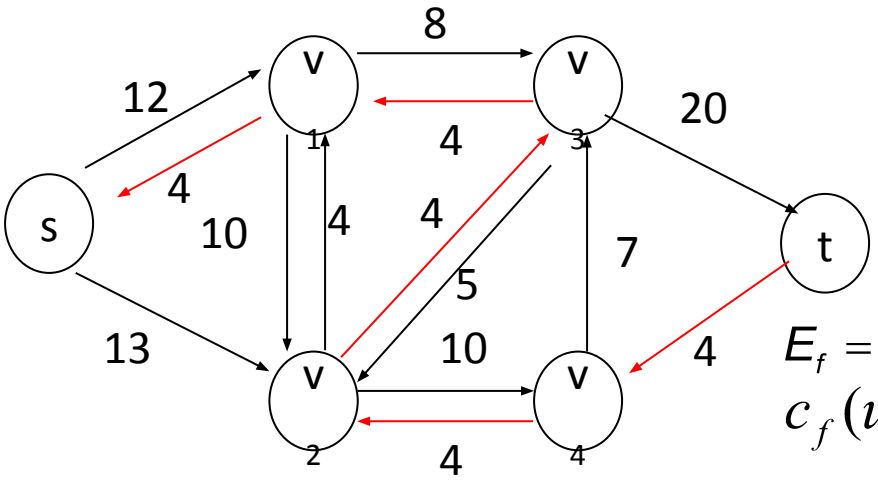
Residual Networks



A graph and a flow along the shaded path



Same graph and flow: flow value is shown as flow/cap on each edge of the flow-path



Flow network representation

$$E_f = \{(u,v) \in V \times V : c_f(u,v) > 0\}$$

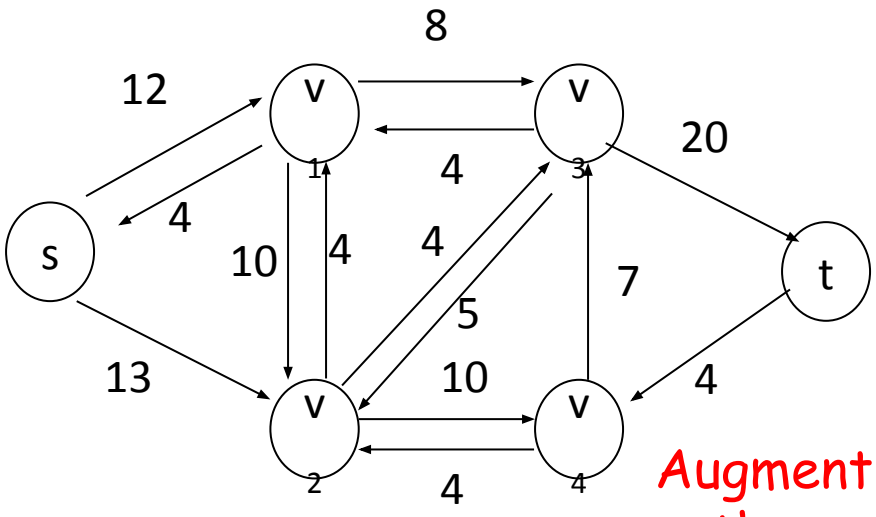
$$c_f(u,v) = c(u,v) - f(u,v)$$

Augmenting Paths

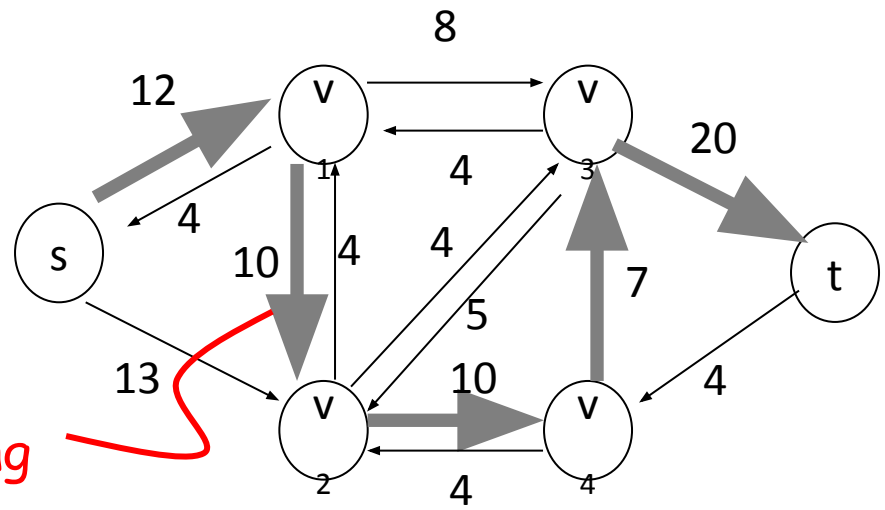
An augmenting path *p* is a simple path from s to t on the residual network.

We call the maximum capacity by which we can increase the flow on *p* the residual capacity of path *p*.

$$c_f(p) = \min\{c_f(u,v) : (u,v) \text{ is on } p\}$$



Network:

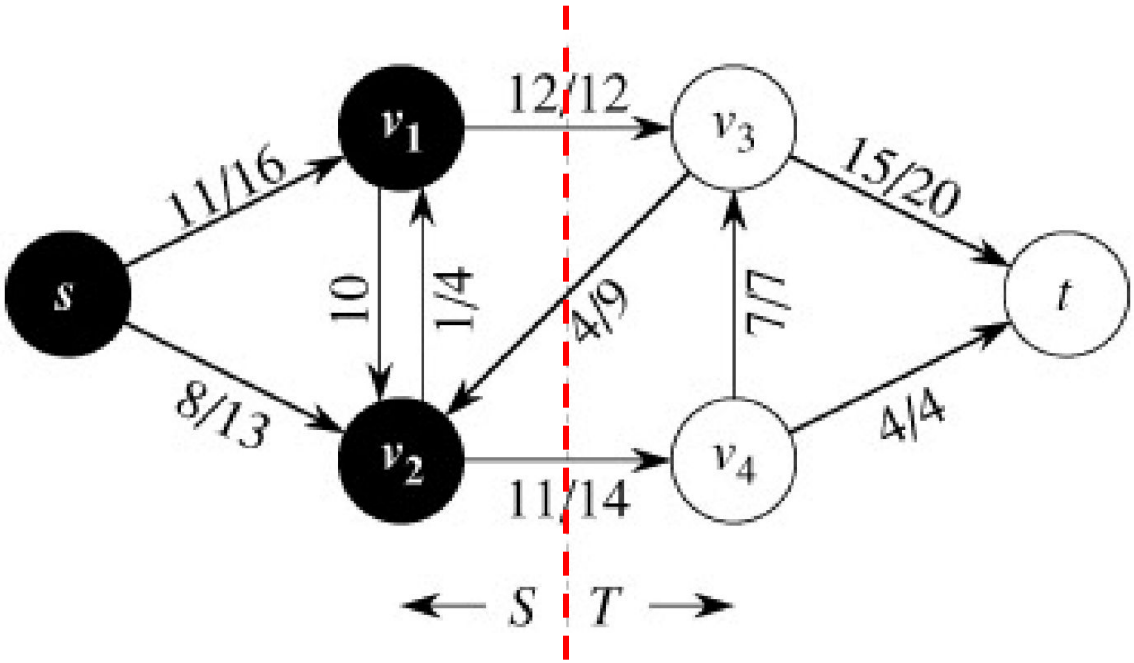


Augmenting path

Residual network:

Cuts of Flow Networks

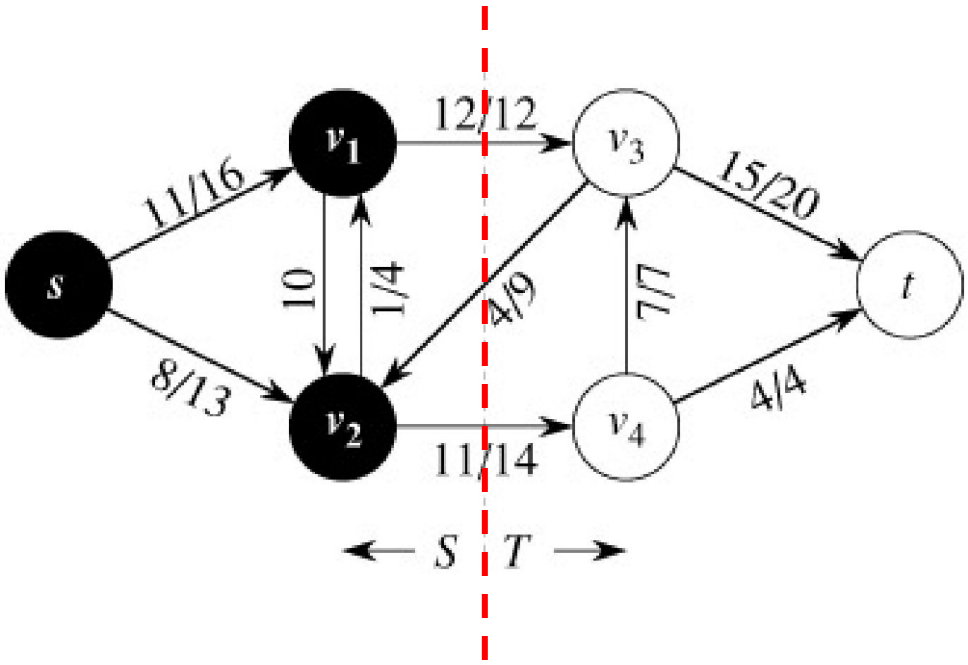
A cut (S,T) of a flow network is a partition of V into S and $T = V - S$ such that $s \in S$ and $t \in T$.



The Net Flow Across a Cut

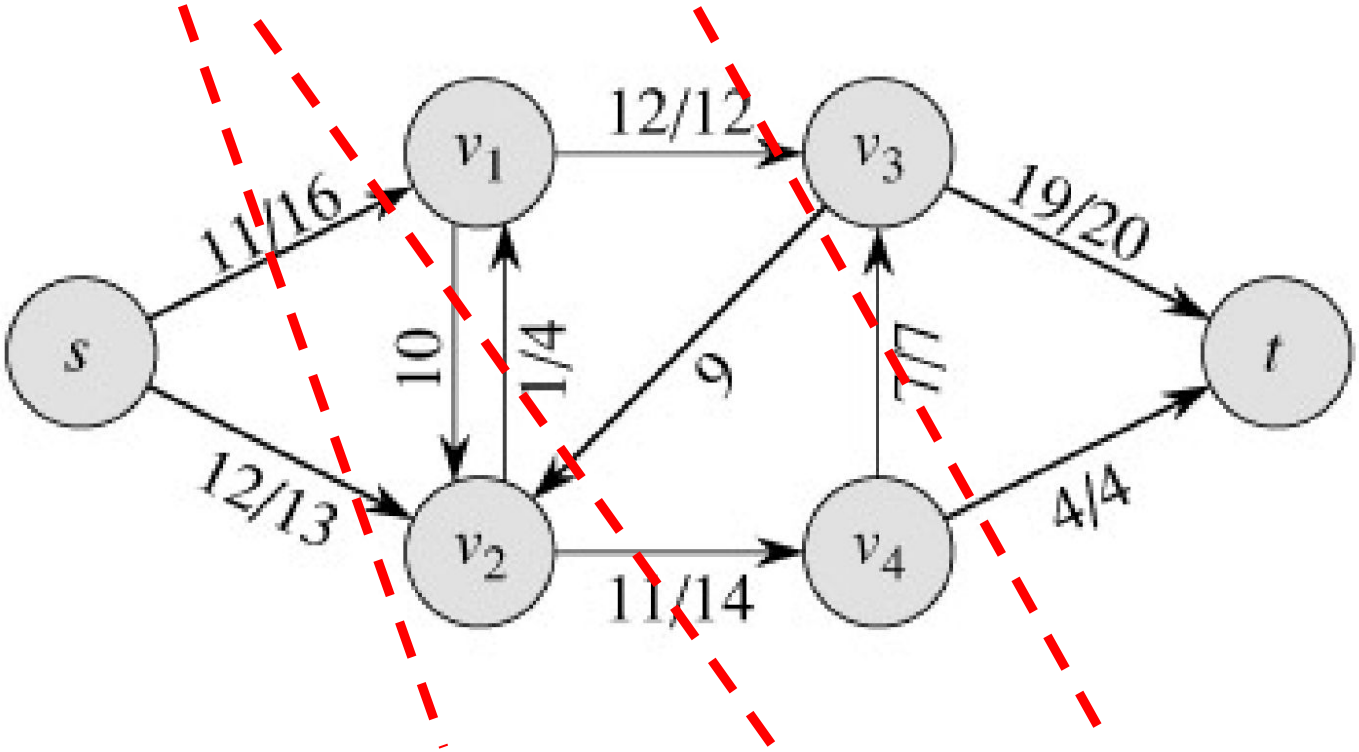
$$f(S,T) = \sum_{u \in S, v \in T} f(u,v) - \sum_{u \in S, v \in T} f(v,u)$$

$$f(S,T) = 12 - 4 + 11 = 19$$



The Net Flow Across a Cut

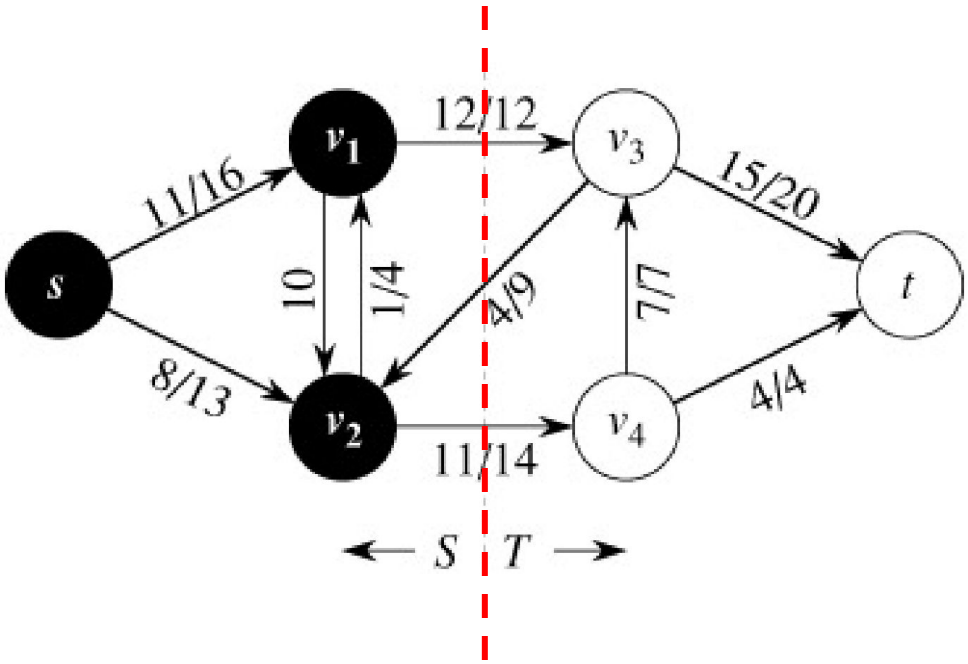
The net flow across any cut is the same and equal to the flow of the network |f|.



The Capacity of a Cut

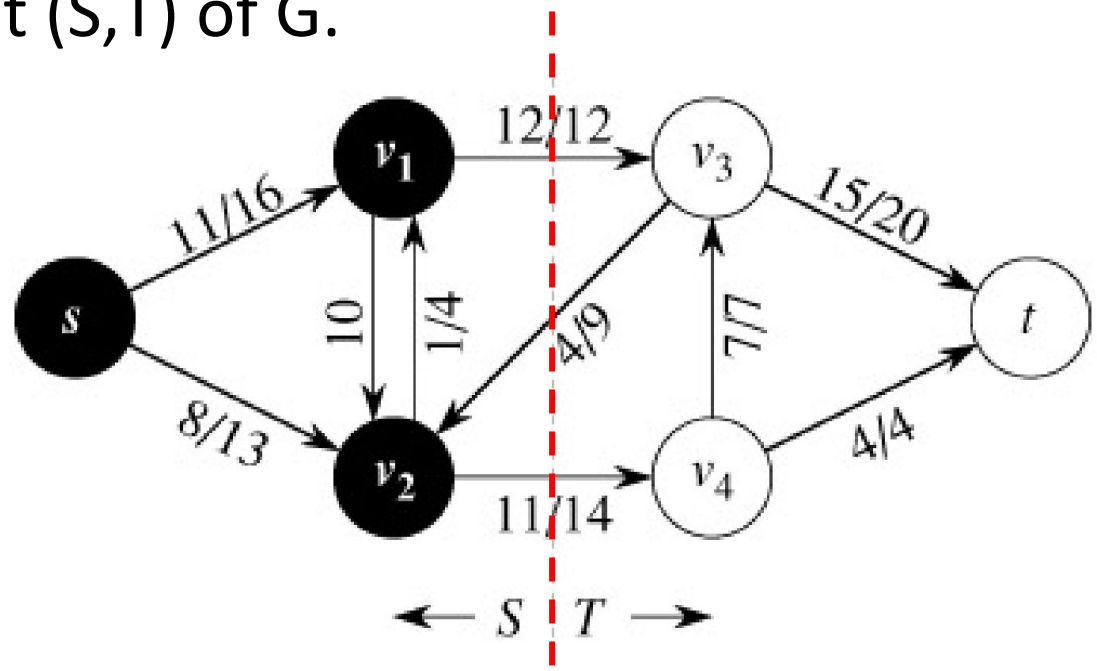
$$c(S,T) = \sum_{u \in S, v \in T} c(u,v)$$

$$c(S,T) = 12 + 0 + 14 = 26$$



Bounding the Network Flow

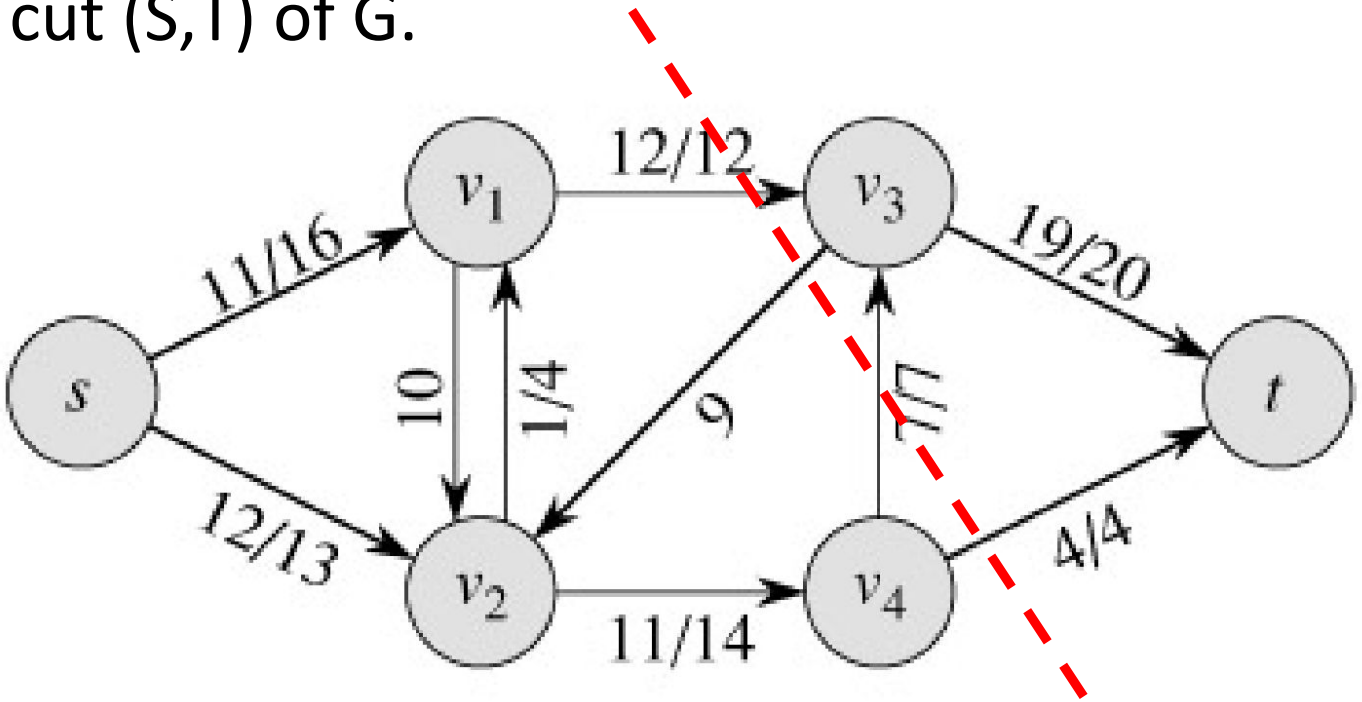
The value of any flow f in a flow network G is bounded from above by the capacity of any cut of G, *i.e.,* $|f| \leq c(S,T)$ for any cut (S,T) of G.



- The capacity of the cut is 26 and flow in the network is 19.

Bounding the Network Flow

The value of any flow f in a flow network G is bounded from above by the capacity of any cut of G, *i.e.,* $|f| \leq c(S,T)$ for any cut (S,T) of G.



- The capacity of the cut is 23.
- In this case, the network does have a maximum flow of 23 and this is a minimum cut: set S of edges going from S to T s.t. sum of capacities/weights of edges in S is minimum

Max-Flow Min-Cut Theorem

If *f* is a flow in a flow network *G=(V,E)*, with source *s* and sink *t*, then the following conditions are equivalent:

1. *f* is a maximum flow in *G*.
2. The residual network G_f contains no augmented paths.
3. *|f| = c(S,T)* for some cut *(S,T)* (*i.e.,* max-flow = min-cut).

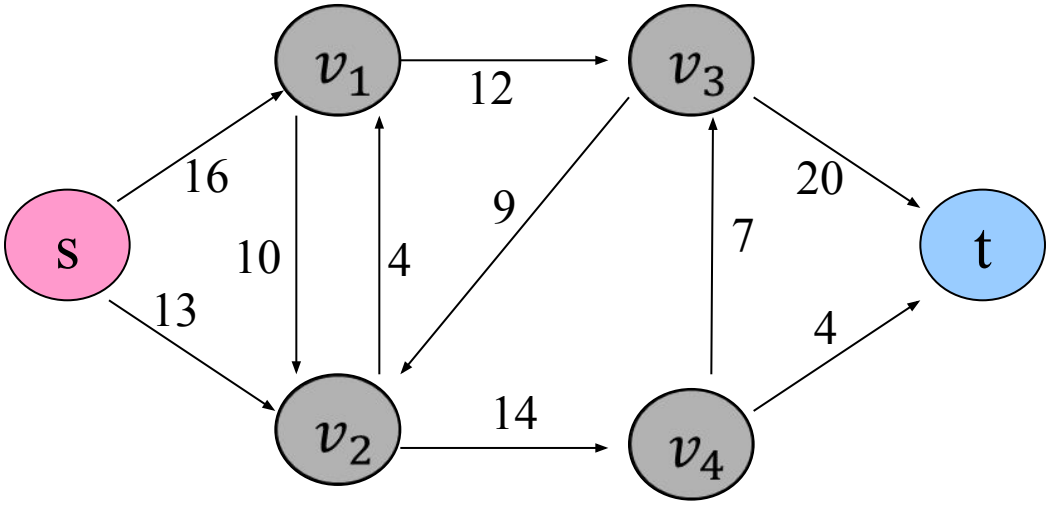
The Basic Ford-Fulkerson Algorithm

FORD-FULKERSON(G,s,t)

1. **for** each edge $(u, v) \in G.E$
2. $\quad (u, v).f = 0$
3. **while** there exists a path p from s to t in the residual network G_f
4. $\quad c_f(p) = \min\{c_f(u, v)\text{: } (u, v) \text{ is in } p\}$
5. $\quad$ **for** each edge (u, v) in p
6. $\quad\quad$ **if** $(u, v) \in E$
7. $\quad\quad\quad (u, v).f = (u, v).f + c_f(p)$
8. $\quad\quad$ **else** $(v, u).f = (v, u).f - c_f(p)$

Example

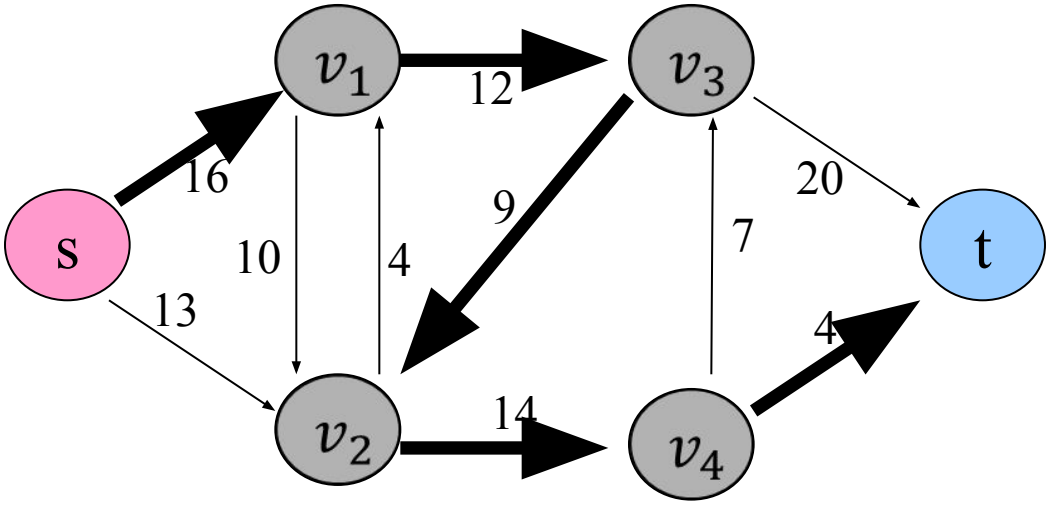
Execution of Ford-Fulkerson Method



Flow: 0

Example

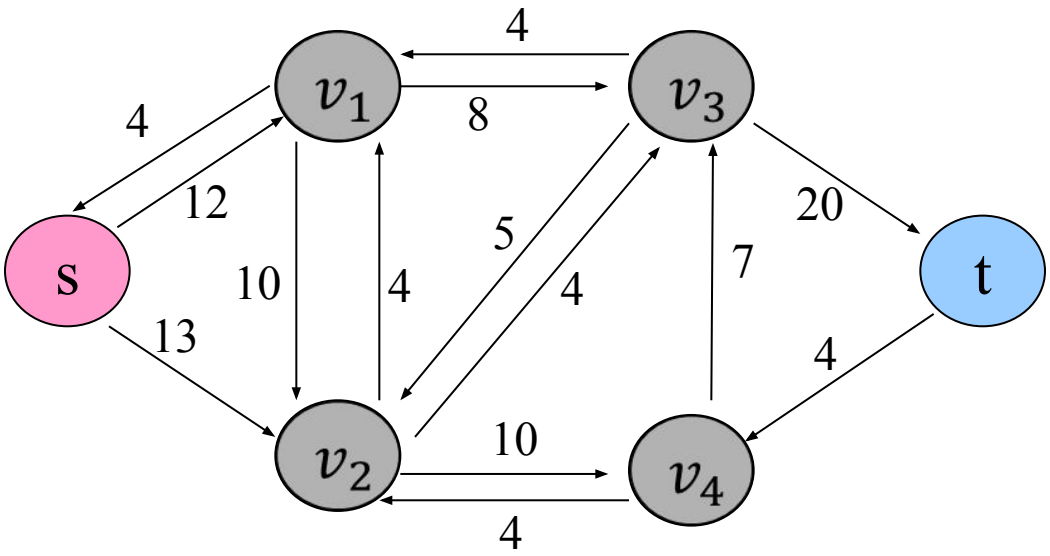
Execution of Ford-Fulkerson Method



Flow: 0

Example

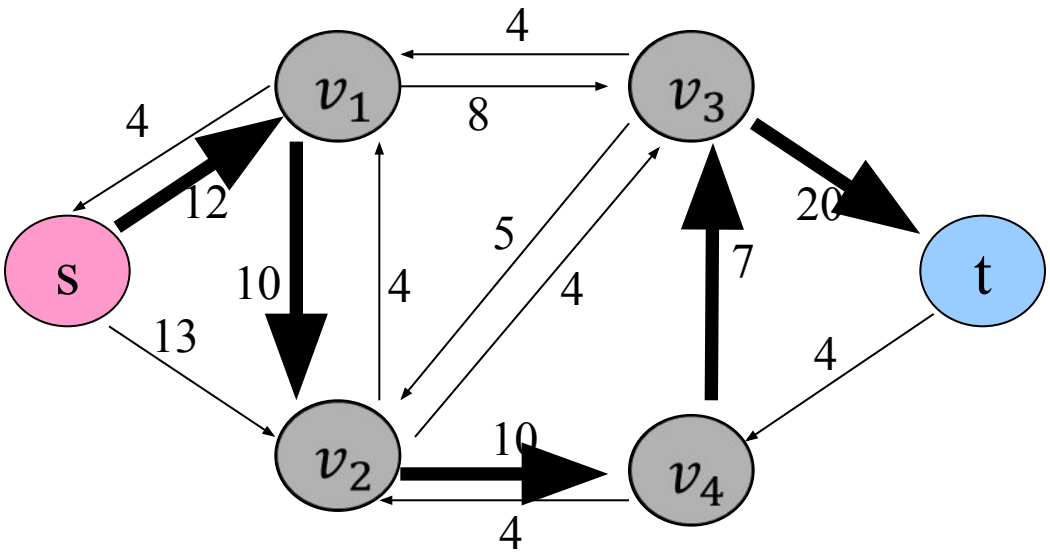
Execution of Ford-Fulkerson Method



Flow: 0+4

Example

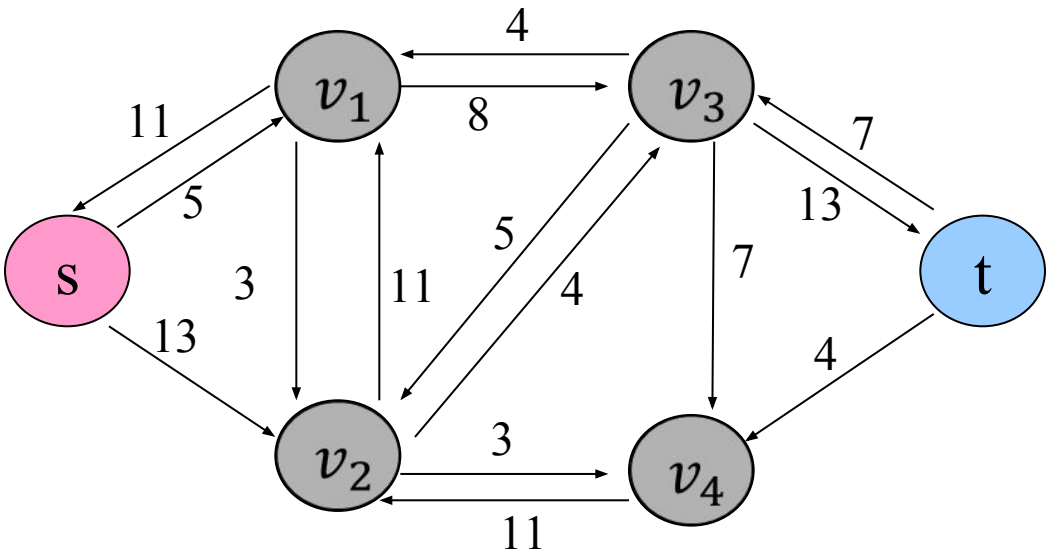
Execution of Ford-Fulkerson Method



Flow: 0+4

Example

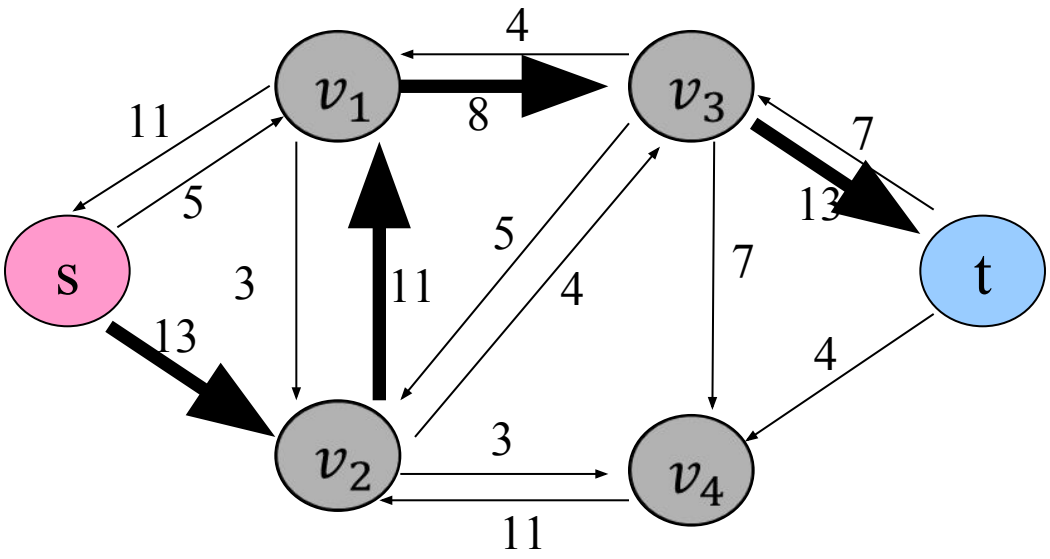
Execution of Ford-Fulkerson Method



Flow: 0+4+7

Example

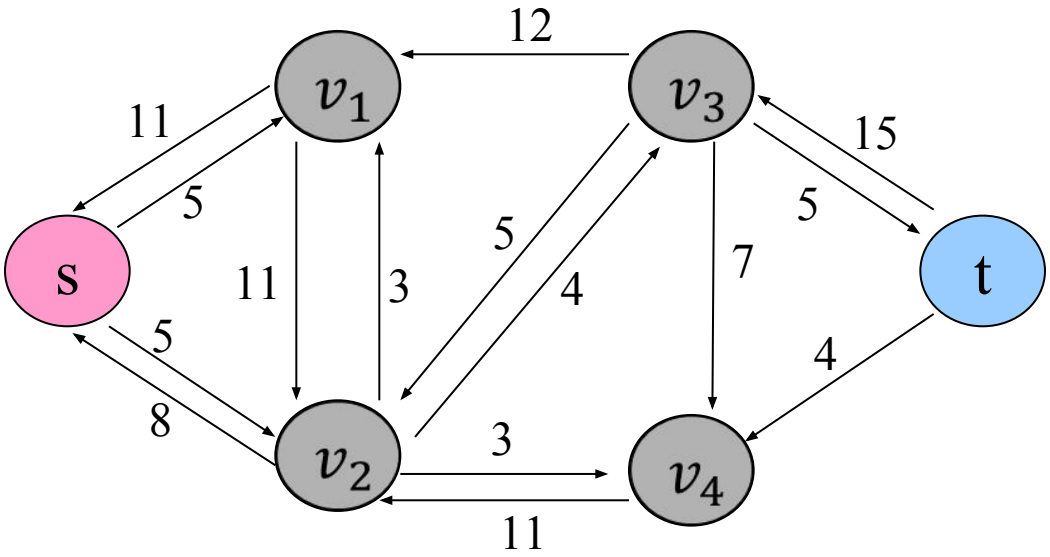
Execution of Ford-Fulkerson Method



Flow: 0+4+7

Example

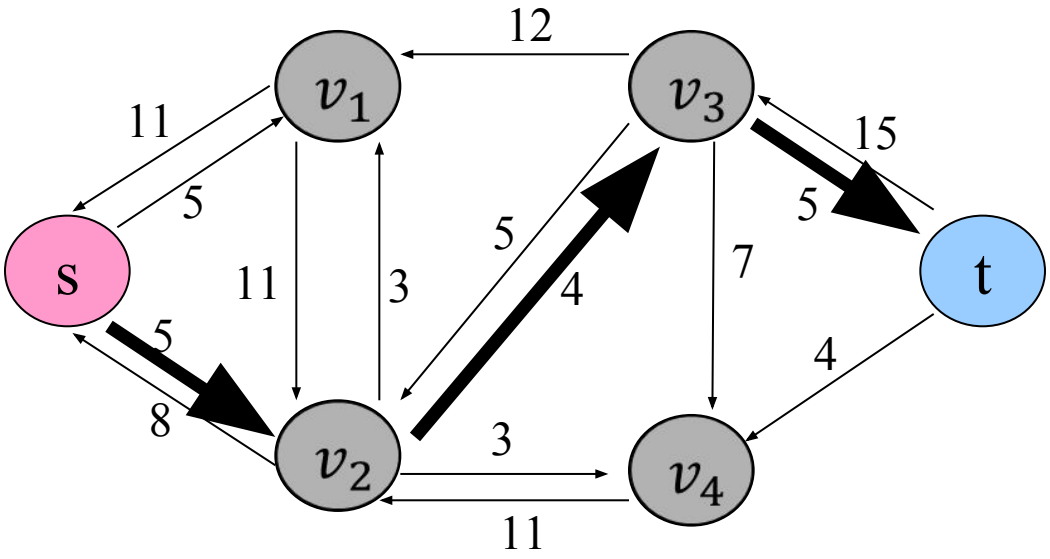
Execution of Ford-Fulkerson Method



Flow: 0+4+7+8

Example

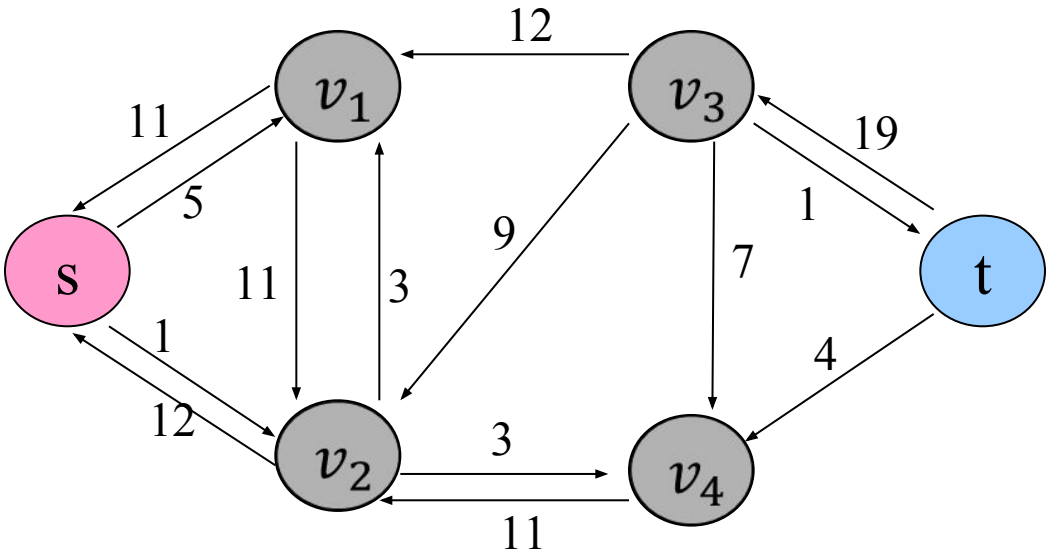
Execution of Ford-Fulkerson Method



Flow: 0+4+7+8

Example

Execution of Ford-Fulkerson Method



Flow: 0+4+7+8+4=23 (no more augmenting path)

Analysis

FORD-FULKERSON(G,s,t)

1. **for** each edge $(u,v) \in G.E$
2. $\quad (u,v).f = 0$
3. **while** there exists a path p from s to t in the residual network G_f
4. $\quad c_f(p) = \min\{c_f(u,v)\text{: } (u,v) \text{ is in } p\}$
5. $\quad$ **for** each edge (u,v) in p
6. $\quad\quad$ **if** $(u,v) \in E$
7. $\quad\quad\quad (u,v).f = (u,v).f + c_f(p)$
8. $\quad\quad$ **else** $(v,u).f = (v,u).f - c_f(p)$

O(E)

?

O(E)

Analysis

If capacities are all integer, then each augmenting path raises $|f|$ by ≥ 1.

If max flow is f^*, then need $\leq |f^*|$ iterations ⇒ time is $O(E|f^*|)$.

Note that this running time is **not polynomial** in input size. It depends on $|f^*|$, which is not a function of $|V|$ or $|E|$.

If capacities are rational, can scale them to integers.

If irrational, FORD-FULKERSON might never terminate!