

CSE 5311 - Design and Analysis of Algorithms

Maximum Flow

Alex Dillhoff

University of Texas at Arlington

Flow Networks

A **flow network** is a directed graph $G = (V, E)$

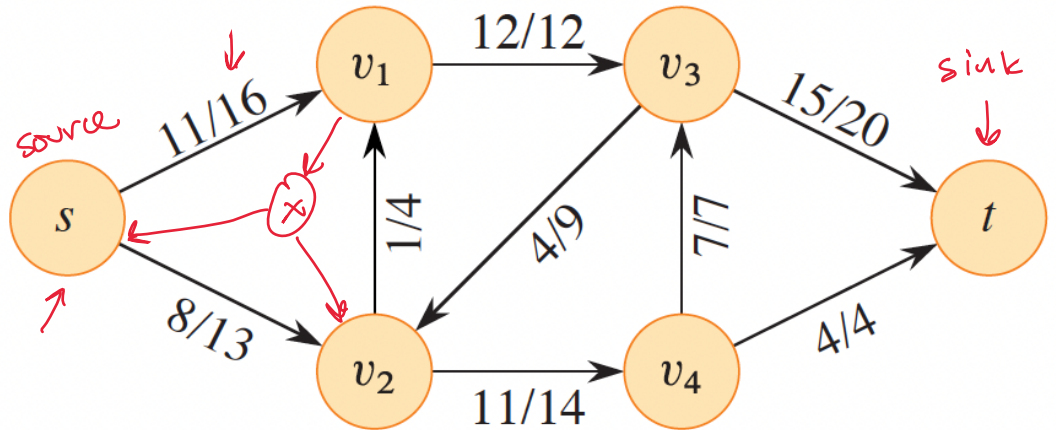
- Requires a source vertex $s \in V$ and a sink vertex $t \in V$
- Edges have a capacity $c(u, v) \geq 0$ for all $u, v \in V$

Flow Networks

The concept of *flow* can take on many meanings.

- The amount of water flowing through a pipe.
- Bandwidth of data through a network.
- Traffic flow through a road network.
- etc.

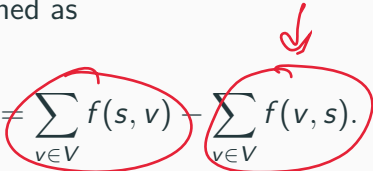
Flow Networks



An example flow network (Cormen et al.).

Flow Networks

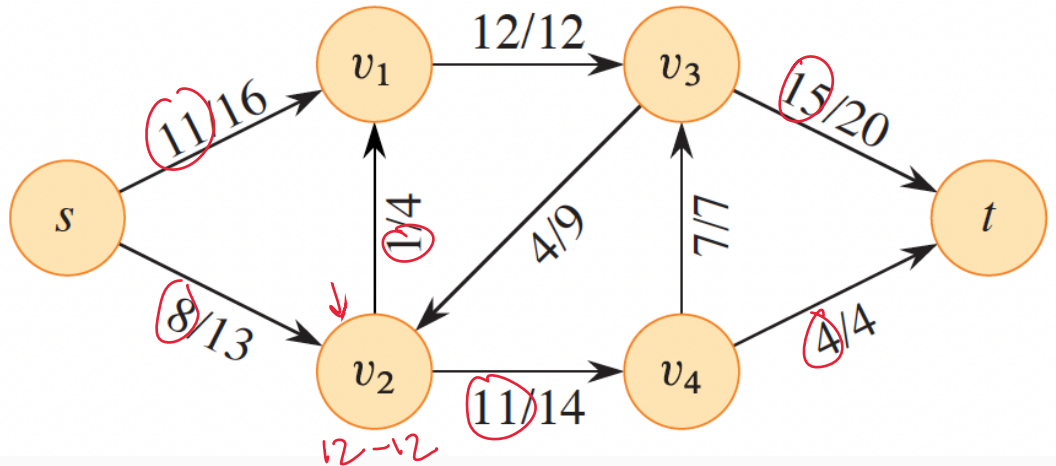
The value of a flow f is defined as

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


In words, this is the total flow out of the source minus the total flow into the source.

What is the flow in the example network?

Flow Networks



An example flow network (Cormen et al.).

Flow Networks

A **flow** in a graph G is a function $f : V \times V \rightarrow \mathbb{R}$ that satisfies two properties:

1. **Capacity constraint:** For all $u, v \in V$,

$$0 \leq f(u, v) \leq c(u, v).$$

2. **Flow conservation:** For all $u \in V - \{s, t\}$,

$$\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v).$$

Flow Networks

There are usually many different possibly paths of flow in a flow network.

The maximum flow problem asks: what is the path that yields the maximum flow?

Antiparallel Edges

The restriction that no two nodes may have more than one edge seems to be unrealistic.

Antiparallel Edges

The restriction that no two nodes may have more than one edge seems to be unrealistic.

For example, modeling the flow of a network graph with this restriction means that network traffic can only move in one direction between two datacenters.

Antiparallel Edges

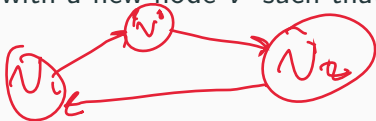
If there were two edges between adjacent nodes v_1 and v_2 such that $(v_1, v_2) \in E$ and $(v_2, v_1) \in E$, we would call these edges **antiparallel**.



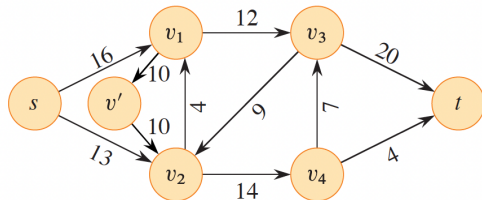
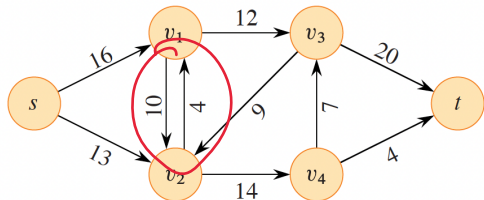
Antiparallel Edges

If there were two edges between adjacent nodes v_1 and v_2 such that $(v_1, v_2) \in E$ and $(v_2, v_1) \in E$, we would call these edges **antiparallel**.

In such cases, the graph is modified with a new node v' such that $(v_1, v') \in E$ and $(v', v_2) \in E$.



Flow Networks



An example flow network with antiparallel edges (Cormen et al.).

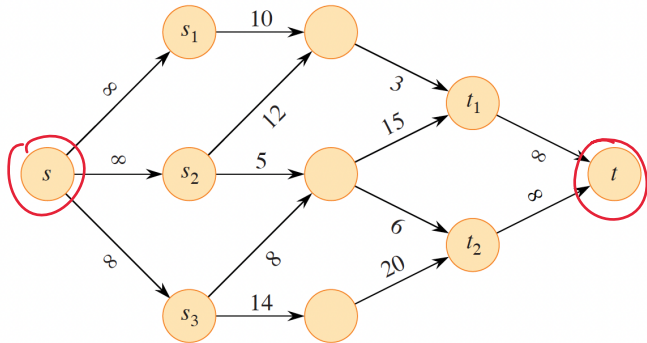
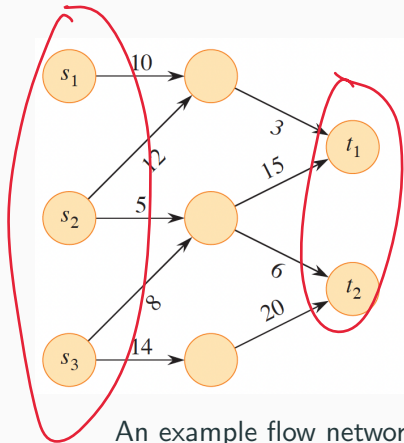
Multiple Sources and Sinks

Maximum flow graphs can only have a single source and sink.

It is easy to imagine a scenario where multiple sources and sinks within a network.

The graph can be modified to accommodate this scenario by defining a **supersource** and **supersink** whose outgoing and incoming flows are infinite.

Flow Networks



An example flow network with multiple sources and sinks (Cormen et al.).

Ford-Fulkerson Algorithm

Ford-Fulkerson Algorithm

A linear time solution was presented by Ford and Fulkerson in 1956.

The solution presented is merely a framework for solving the maximum flow problem.

The Edmonds-Karp algorithm is a specific implementation of the Ford-Fulkerson algorithm that uses BFS to find augmenting paths.

Ford-Fulkerson Algorithm

Ford-Fulkerson relies on three foundational concepts:

1. Residual networks
2. Augmenting paths
3. Cuts

Ford-Fulkerson Algorithm

The solution presented by Ford and Fulkerson is not a single algorithm but rather a set of general instructions.

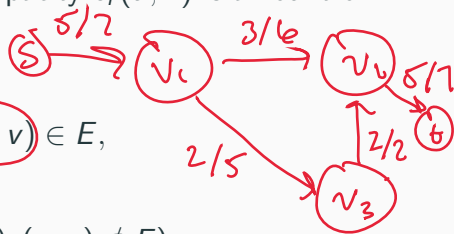
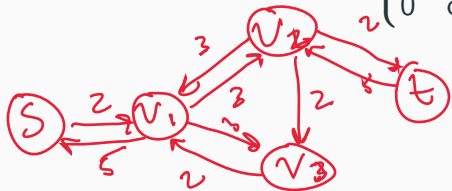
1. Initialize $f(u, v) = 0$ for all $u, v \in V$, giving an initial flow of 0.
2. Increase the flow by finding an augmenting path in a residual network.
3. The edges of the augmented path indicate where to increase the flow.

This is repeated until the residual network has no more augmenting paths.

Residual Networks

Consider a pair of vertices $u, v \in V$, the residual capacity $c_f(u, v)$ is amount of additional flow that can be pushed from u to v .

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } (u, v) \in E, \\ f(v, u) & \text{if } (v, u) \in E, \\ 0 & \text{otherwise (i.e., } (u, v), (v, u) \notin E). \end{cases}$$



$c_f(v_2, v_3) = 2$

$\{S, v_1, v_2, t\}$

Residual Networks

The **residual network** is $G_f = (\dot{V}, \dot{E}_f)$, where

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

Residual Networks

Definition

- The edges of G_f represent those edges in G with the capacity to change the flow.
- There is also no requirement for all edges in G to be present in G_f .
- As the algorithm works out the solution, we are only considered with edges that permit more flow.

Residual Networks

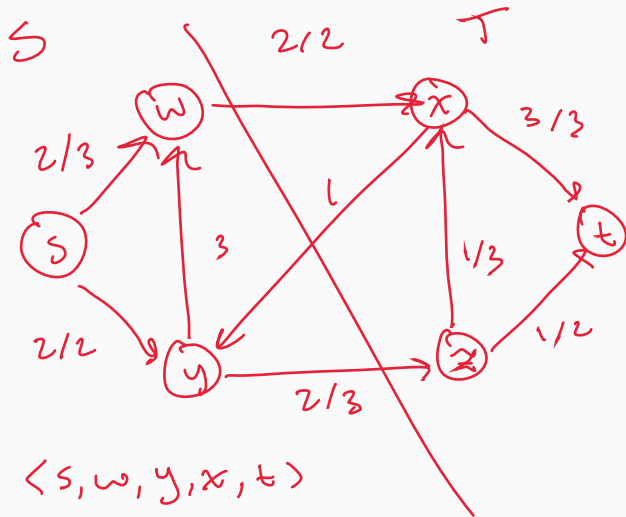
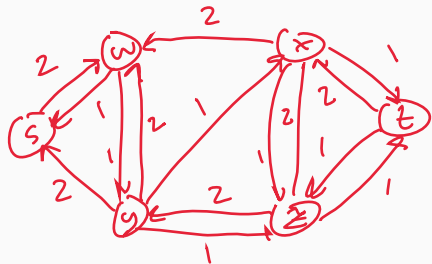
Definition

G_f

- An edge $(u, v) \in E$ means that the reverse edge $(v, u) \notin E$.
- The residual network can have edges that are not in G .
- These are used to represent paths in which flow is sent in the reverse direction.
- This can happen if reducing flow from one edge results in a net increase across some other.
- In G_f , the reverse edges (v, u) represent the flow on $(u, v) \in G$ that could be sent back.

Residual Networks

Example: Residual Network



Cuts

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

The **capacity** of the cut is the sum of the capacities of the edges from S to T .

Any cut is a valid cut as long as the source is in S and the sink is in T .

If f is a flow in G and (S, T) is a cut of G , then the **net flow** across the cut is

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

The **capacity** of the cut is

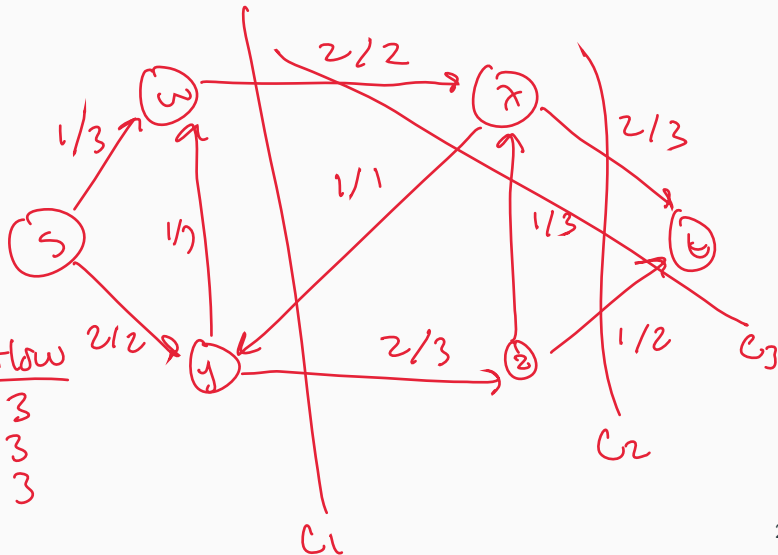
$$c(S, T) = \sum_{u \in S} \sum_{v \in T} c(u, v).$$

A **minimum cut** is a cut whose capacity is the smallest among all cuts.

Cuts

Example: Cuts

Cuts	Capacity	Flow
C_1	5	3
C_2	5	3
C_3	7	3



Lemma

For any flow f and any cut (S, T) of G , we have that $|f| = f(S, T)$.

This lemma states that the flow across a cut is equal to the value of the flow.

Key concept: We can determine the flow by making cuts in the graph.

The minimum cut leads to the maximum flow.

Corollary

The value of any flow f in a flow network G is bounded from above by the capacity of any cut of G .

Max-flow Min-cut Theorem

The following statements are logically equivalent.

1. The flow f is a maximum flow in G .
2. The residual network G_f contains no augmenting paths.
3. The value of the flow f is equal to the capacity of the cut (S, T) for some cut of G .

Augmentation Function

An **augmentation function** is a function that increases the flow of a network by a certain amount.

Not only does it allow for the increase of flow, but it also permits an augmented *reverse* flow.

Augmentation Function

Given flows f in G and f' in G_f , define the **augmentation** of f by f' , as a function $V \times V \rightarrow \mathbb{R}$:

$$(f \uparrow f')(u, v) = \begin{cases} f(u, v) + f'(u, v) - \cancel{f(v, u)} & \text{if } (u, v) \in E, \\ 0 & \text{otherwise} \end{cases}$$

Handwritten annotations: Red circles around f and f' in the left-hand side. A red arrow points from the circled $f(u, v)$ to the circled f' . Another red arrow points from the circled f' to the $f(v, u)$ term in the denominator, which is crossed out. Below the denominator, $f'(v, u)$ is written in red with an arrow pointing to the $f(v, u)$ term.

for all $(u, v) \in V$.

Augmentation Function

This augmentation function represents an increase of flow on (u, v) by $f'(u, v)$ with a decrease by $f'(v, u)$ since pushing flow on the reverse edge in G_f represents a decrease in G .

This is known as **cancellation**.

Augmentation Function

Lemma

Given a flow network G , a flow f in G , and the residual network G_f , let f' be a flow in G_f . Then $(f \uparrow f')$ is a flow in G with value $|f \uparrow f'| = |f| + |f'|$.

Augmentation Function

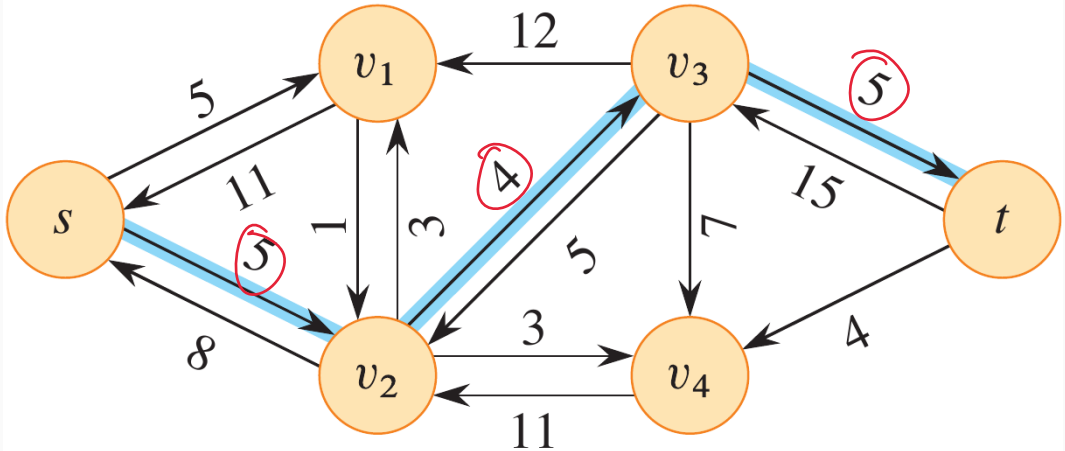
The definition of an augmentation function sets up a mechanism for increasing the flow in G by considering the flow in G_f .

Augmenting Paths

An **augmenting path** is a simple path from the source to the sink in the residual network.

- The purpose of an augmenting path is to increase the flow from the source to the sink.
- The flow is increased by the minimum capacity of the edges in the path.

Augmenting Paths



An example augmenting path (Cormen et al.).

Augmenting Paths

Lemma 24.2

Let $G = (V, E)$ be a flow network, let f be a flow in G , and let p be an augmenting path in G_f . Define $f_p : V \times V \rightarrow \mathbb{R}$ by

$$f_p(u, v) = \begin{cases} c_f(p) & \text{if } (u, v) \text{ is in } p, \\ 0 & \text{otherwise.} \end{cases}$$

Then f_p is a flow in G_f with value $|f_p| > 0$.

Augmenting Paths

The maximum amount that an augmenting path can be increased is the minimum capacity of the edges in the path.

This is known as the **residual capacity** of the path.

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

Augmenting Paths

Put simply, if there is a path in the residual network, the flow can be increased by the minimum capacity of the edges in the path.

If there is no path, the flow is at its maximum.

Ford-Fulkerson Algorithm

```
def ford_fulkerson(G, s, t):  
    f = {u: {v: 0 for v in G} for u in G}  
    while True:  
        # Find an augmenting path  
        path = bfs(G, s, t, f)  
        if not path:  
            break  
        cf = min(G[u][v] - f[u][v] for u, v in path)  
        for u, v in path:  
            f[u][v] += cf  
            f[v][u] -= cf  
    return f
```

Run Ford-Fulkerson on an example graph.