FIT2004 - Algorithms and Data Structures

Seminar 8 - Network Flow

Rafael Dowsley

28 April 2025

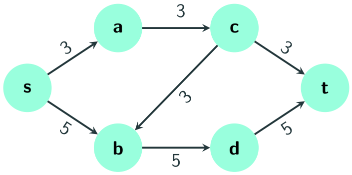
Agenda

Divide-and-Conquer (W1-3)	Greedy Algorithms (W4-5)	Dynamic Programming (W6-7)	Network Flow (W8-9)	Data Structures (W10-11)

1. Maximum Flow Problem
2. Ford-Fulkerson method
3. Min-cut Max-flow Theorem

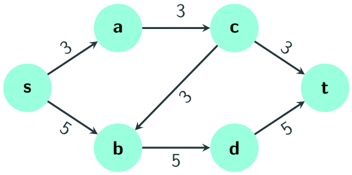
Maximum Flow Problem

Flow networks



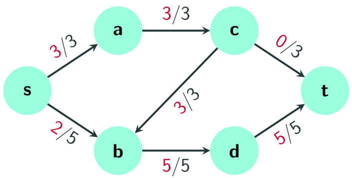
- A flow network is a connected directed graph where:
  - There is a single source vertex, often denoted by s, which only has outgoing edges.
  - There is a single sink/target vertex, often denoted by t, which only has incoming edges.
  - Each edge has a given non-negative capacity (usually integers) giving the maximum amount/rate of flow that the edge can carry.

What are flow networks used for?



- Flow networks can model many real-world problems, such as:
  - Water flowing through an assembly of pipes.
  - Electric current flowing through electrical circuits.
  - Information flowing through communication networks.
- **Can be applied to solve a large range of other combinatorial problems unrelated to physical flows.**

Flow



- For an edge e, its flow $f(e)$ is an assignment of how much material is flowing through it in the flow network given its capacity $c(e)$.
- All vertices (except source and sink) conserve their flow:
  - Let $E_{in}(v)$ denote the set of all incoming edges to a vertex v, and similarly $E_{out}(v)$ denote its set of outgoing edges.
  - The total amount flowing into any vertex (through incoming edges) is equal to the total amount flowing out of that vertex (through outgoing edges).

Quiz time!

Properties of a flow network

A flow network must satisfy the following properties:

Capacity constraint

For every edge e, its flow is bounded by its capacity: $0 \leq f(e) \leq c(e)$.

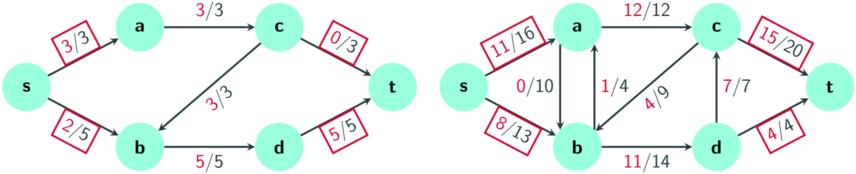
Flow conservation

For every vertex $v \in V \setminus \{s, t\}$, it holds that

$$\sum_{e_{in} \in E_{in}(v)} f(e_{in}) = \sum_{e_{out} \in E_{out}(v)} f(e_{out})$$

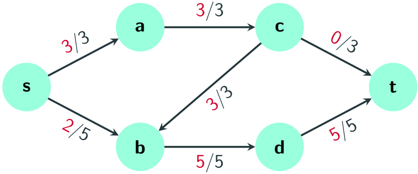
We only consider integer capacities.

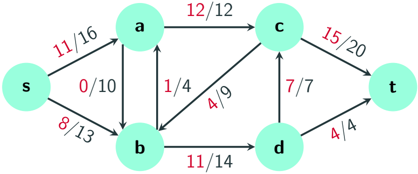
Flow of the network



- Given that the flow network satisfies the capacity constraint and flow conservation properties, the flow of the network is the total flow out of the source vertex.
  - Equivalently, this is the same as the total flow into sink vertex.
  - What is the flow value in the left flow network? 5
  - What is the flow value in the right flow network? 19

Maximum-flow problem





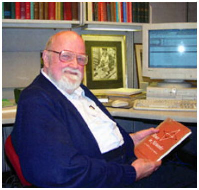
Maximum-flow problem

Given a flow network, determine the maximum value of the flow that can be sent from source s to sink t without violating the capacity constraint and flow conservation properties.

Ford-Fulkerson method

Ford-Fulkerson method

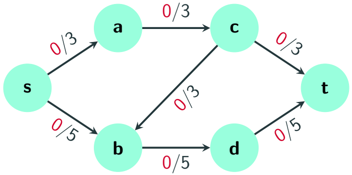
Ford-Fulkerson is a method for solving max-flow problems.



Lester Ford Jr.

Delbert Fulkerson

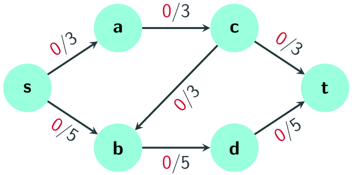
Ford-Fulkerson intuition



How can we increase the flow in the above network?

Quiz time!

Ford-Fulkerson intuition



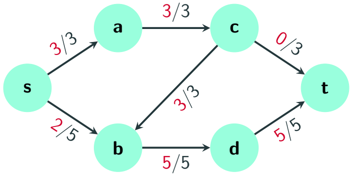
How can we increase the flow in the above network?

1. Choose a path from source to sink.

2. Increase flow along it as much as possible.

Seems easy enough!

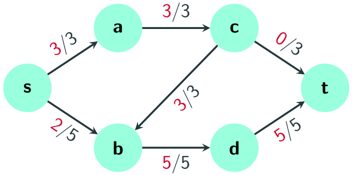
Ford-Fulkerson intuition



Can we increase the flow in the above network?

Quiz time!

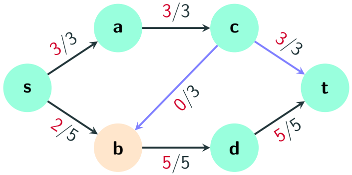
Ford-Fulkerson intuition



Can we increase the flow in the above network?

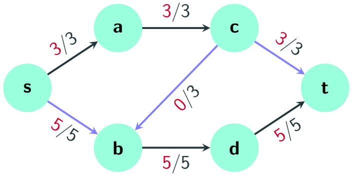
We can! But there is no path from source to sink with spare capacity. . .

Ford-Fulkerson intuition



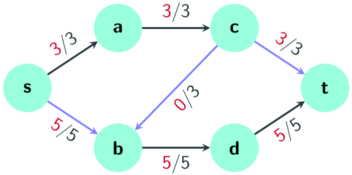
- There is no path from s to t with spare capacity.
- Redirect the 3 units on the edge $c \to b$ to go to edge $c \to t$.
- **Problem:** The flow through b is not conserved anymore.

Ford-Fulkerson intuition

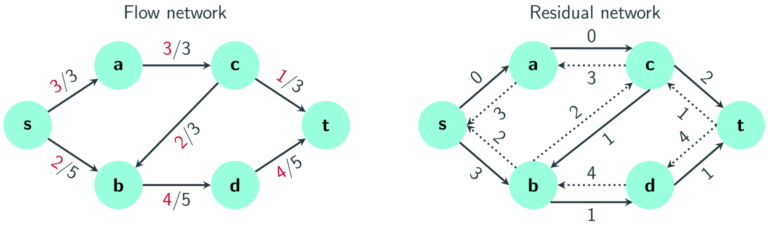


- There is no path from s to t with spare capacity.
- Redirect the 3 units on the edge $c \to b$ to go to edge $c \to t$.
- **Problem:** The flow through b is not conserved anymore.
- **Solution:** Send 3 more units along $s \to b$.

What actually happened here?



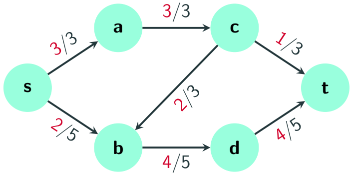
- We increased the total flow by 3.
- We sent 3 additional units along $s \to b$, $c \to t$.
- We removed 3 units of flow from $c \to b$.
- Our path was $s \to b \to c \to t$, but we had a backwards edge...



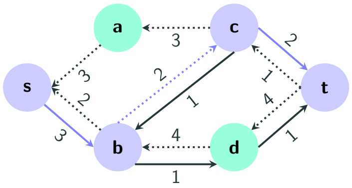
- Residual network has the same vertices as the original network.
- For every directed edge $u \to v$ in flow network, we add two edges in the residual network:
  - **Forward edge/residual edge:** an edge in the same direction as $u \to v$ with the residual/remaining capacity in the flow network.
  - **Backward edge/reversible flow edge:** an edge in the direction $v \to u$ with weight equal to the current flow of $u \to v$ in the flow network.

Augmenting path

Flow network



Residual network



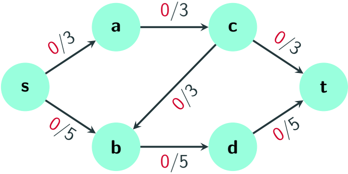
- Augmenting path is any simple path (a path without repeating vertices) from source s to target t along edges with positive weight in the residual network.
  - We can omit edges with weight 0 from the residual network.
- Residual capacity is the minimum edge weight in the residual along this augmenting path (e.g., 2 in the example).

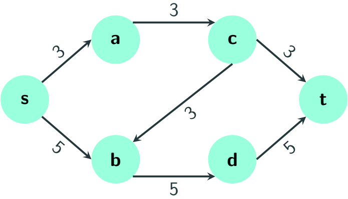
Ford-Fulkerson method

Ford-Fulkerson method

function MAX_FLOW($G = (V, E), s, t$)
  set initial flow f to 0 on all edges
  while there exists an augmenting path p in the residual network G_f **do**
    augment the flow f along the augmenting path p as much as possible
  return f

Flow of the network $= 0$



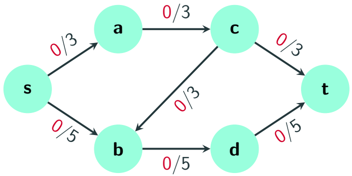


Ford-Fulkerson method

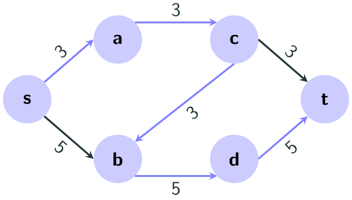
Ford-Fulkerson method

function $\textsc{max\_flow}(G = (V, E), s, t)$
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f **do**
        augment the flow f along the augmenting path p as much as possible
    return f

Flow of the network $= 0$



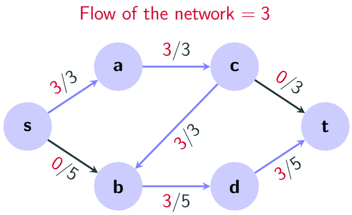
Path capacity $= 3$

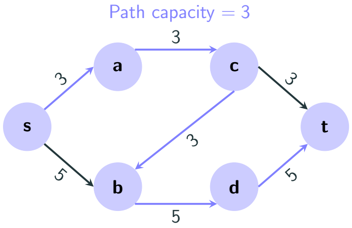


Ford-Fulkerson method

Ford-Fulkerson method

function $\text{MAX\_FLOW}(G = (V, E), s, t)$
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f **do**
        augment the flow f along the augmenting path p as much as possible
    return f



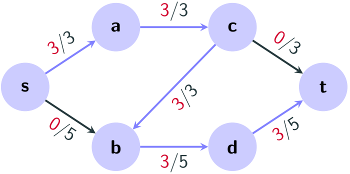


Ford-Fulkerson method

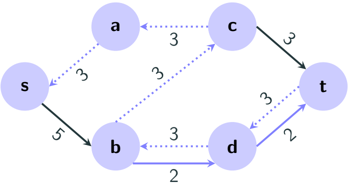
Ford-Fulkerson method

function MAX_FLOW$(G = (V, E), s, t)$
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f **do**
        augment the flow f along the augmenting path p as much as possible
    return f

Flow of the network $= 3$



Residual needs to be updated!

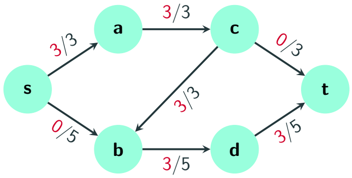


Ford-Fulkerson method

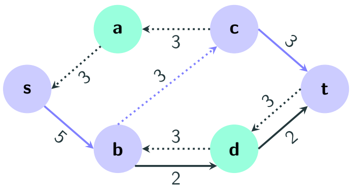
Ford-Fulkerson method

function MAX_FLOW($G = (V, E), s, t$)
  set initial flow f to 0 on all edges
  while there exists an augmenting path p in the residual network G_f **do**
    augment the flow f along the augmenting path p as much as possible
  return f

Flow of the network = 3



Path capacity = 3

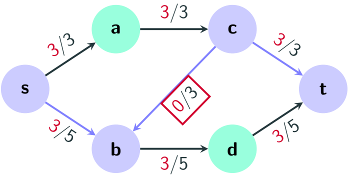


Ford-Fulkerson method

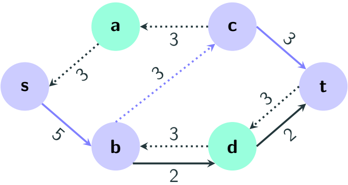
Ford-Fulkerson method

function $\textsc{max\_flow}(G = (V, E), s, t)$
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f **do**
        augment the flow f along the augmenting path p as much as possible
    return f

Flow of the network $= 6$



Path capacity $= 3$

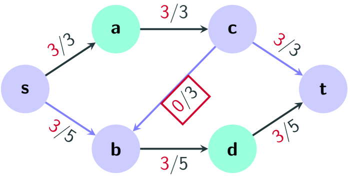


Ford-Fulkerson method

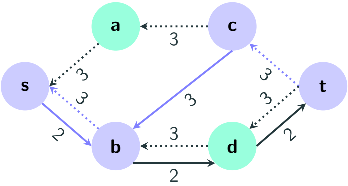
Ford-Fulkerson method

function MAX_FLOW$(G = (V, E), s, t)$
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f **do**
        augment the flow f along the augmenting path p as much as possible
    return f

Flow of the network $= 6$



Residual needs to be updated!

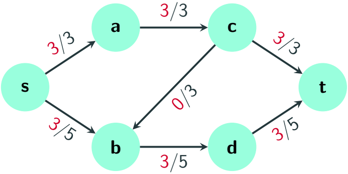


Ford-Fulkerson method

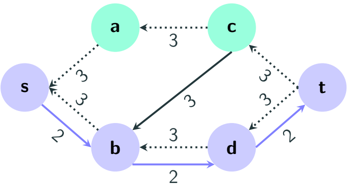
Ford-Fulkerson method

function MAX_FLOW($G = (V, E), s, t$)
  set initial flow f to 0 on all edges
  while there exists an augmenting path p in the residual network G_f **do**
    augment the flow f along the augmenting path p as much as possible
  return f

Flow of the network $= 6$



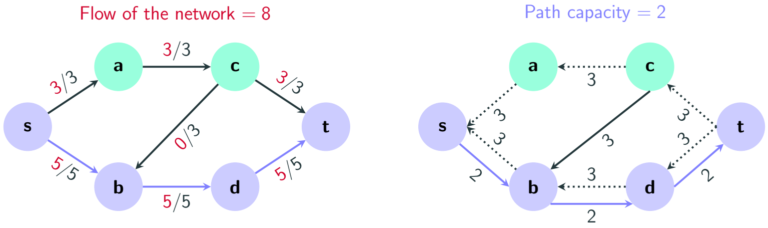
Path capacity $= 2$



Ford-Fulkerson method

Ford-Fulkerson method

function MAX_FLOW($G = (V, E), s, t$)
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f **do**
        augment the flow f along the augmenting path p as much as possible
    return f

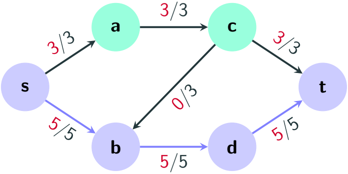


Ford-Fulkerson method

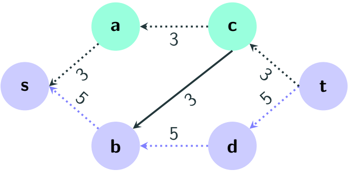
Ford-Fulkerson method

function $\textsc{max\_flow}(G = (V, E), s, t)$
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f **do**
        augment the flow f along the augmenting path p as much as possible
    return f

Flow of the network $= 8$



Residual needs to be updated!

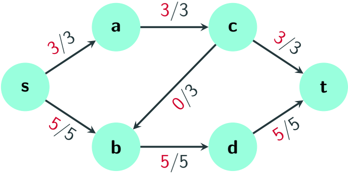


Ford-Fulkerson method

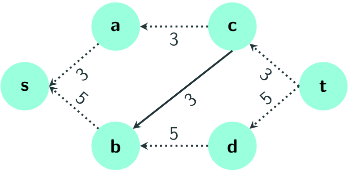
Ford-Fulkerson method

function MAX_FLOW($G = (V, E), s, t$)
  set initial flow f to 0 on all edges
  while there exists an augmenting path p in the residual network G_f **do**
    augment the flow f along the augmenting path p as much as possible
  return f

Flow of the network $= 8$



No further augmenting paths!

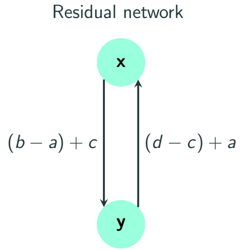


Implementation details

- Edge "type" in the residual does not matter as we can send flow along any of them.
  - Forward edges can have flow sent along them because they have spare capacity.
  - Backward edges can have flow sent along them because reducing a flow in one direction is the same as increasing the flow in the opposite direction.
- If a pair of vertices in the flow network have edges in both directions, then in the residual network there are only 2 edges (not 4).

Quiz time!

Implementation details



- The amount of flow we can send from x to y consists of:
  - The spare capacity $(b - a)$.
  - The existing flow from y to x which can be reversed (c units).

Time complexity

Ford-Fulkerson method

function MAX_FLOW($G = (V, E), s, t$)
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f **do**
        augment the flow f along the augmenting path p as much as possible
    return f

- Cost of finding an augmenting path: $O(|V| + |E|)$ using BFS/DFS.
- Augmenting flow along a path: length of path $\leq |V| - 1$, so $O(|V|)$.
- Updating the residual: two edges per edge of the augmenting path, so $O(|V|)$.
- **Total work in one iteration of the loop:** $O(|V| + |E|) = O(|E|)$ as the graph is connected.

Time complexity

Ford-Fulkerson method

```
function MAX_FLOW(G = (V, E), s, t)
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f do
        augment the flow f along the augmenting path p as much as possible
    return f
```

- **Total work in one iteration of the loop:** $O(|E|)$.
- How many iterations?

Quiz time!

Time complexity

Ford-Fulkerson method

```
function MAX_FLOW(G = (V, E), s, t)
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f do
        augment the flow f along the augmenting path p as much as possible
    return f
```

- **Total work in one iteration of the loop:** $O(|E|)$.
- How many iterations?
  - Assuming integer capacities, Ford-Fulkerson flows are always integer-valued, so flow grows by at least 1 in each iteration.
  - If maximum flow in the network is F, then at most F iterations.
- **Total work:** $O(|E| \cdot F)$.

Time complexity

Ford-Fulkerson method

```
function MAX_FLOW(G = (V, E), s, t)
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f do
        augment the flow f along the augmenting path p as much as possible
    return f
```

- **Total work:** $O(|E| \cdot F)$.
  - This looks polynomial.
  - But it isn't because F is a number, so its value is exponential in the space required to store it.
- It can be proven that the complexity is $O(|V| \cdot |E|^2)$ when using BFS to find augmenting paths, which is polynomial.
  - These two bounds are incomparable.

Correctness

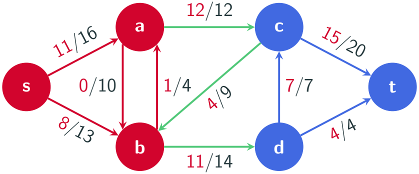
Ford-Fulkerson method

```
function MAX_FLOW(G = (V, E), s, t)
    set initial flow f to 0 on all edges
    while there exists an augmenting path p in the residual network G_f do
        augment the flow f along the augmenting path p as much as possible
    return f
```

- **Does the algorithm terminate?**
  - Yes, assuming all capacities are integers.
  - The flow always increases by at least 1 per iteration and there cannot be any augmenting path if all source's outgoing edges (similarly, if all sink's incoming edges) are saturated.
- In order to show that the algorithm terminates exactly once it finds a flow whose value is the maximum possible value among all feasible flows, we will need to study the Min-cut Max-flow Theorem.

Min-cut Max-flow Theorem

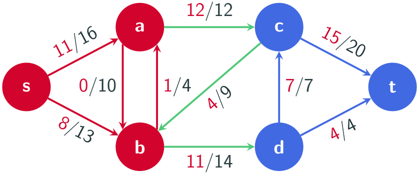
Cut



- A cut (S, T) of a flow network partitions the vertices into two disjoint sets S and T such that $s \in S$ and $t \in T$.

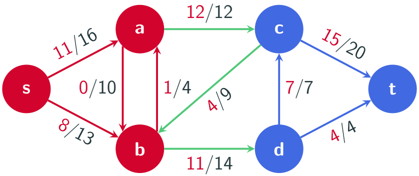
- The cut-set of a cut (S, T) is the set of edges that "cross" the cut, i.e., each edge connects one vertex in S with another in T.
  - Outgoing edges of the cut: from a vertex in S to a vertex in T.
  - Incoming edges of the cut: from a vertex in T to a vertex in S.

Flow and capacity of a cut



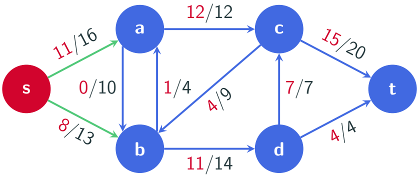
- Capacity of a cut (S, T) is the total capacity of its outgoing edges.
- Flow of a cut (S, T) is equal to total flow of outgoing edges − total flow of incoming edges.
- Note that the flow of a cut is always $\leq$ to the capacity of the cut.
  - Flow of an edge $\leq$ capacity of an edge.
  - Capacity of a cut does not subtract capacities for incoming edges.

Flow and capacity of a cut



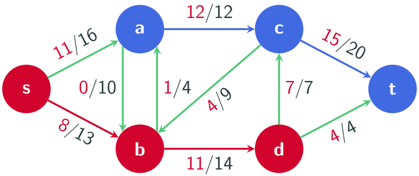
- Capacity of a cut (S, T) is the total capacity of its outgoing edges.
  - What is the capacity of this cut?
  - 26
- Flow of a cut (S, T) is equal to total flow of outgoing edges $-$ total flow of incoming edges.
  - What is the flow of this cut?
  - 19

Another cut



- Capacity of a cut (S, T) is the total capacity of its outgoing edges.
  - What is the capacity of this cut?
  - 29
- Flow of a cut (S, T) is equal to total flow of outgoing edges $-$ total flow of incoming edges.
  - What is the flow of this cut?
  - 19

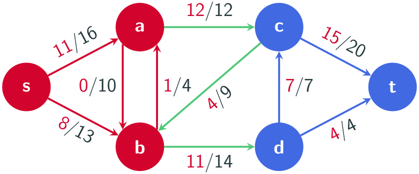
Yet another cut



- Capacity of a cut (S, T) is the total capacity of its outgoing edges.
  - What is the capacity of this cut?
  - 31

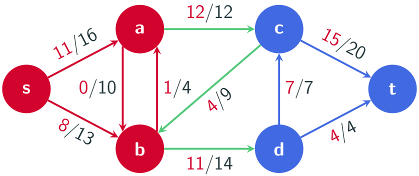
- Flow of a cut (S, T) is equal to total flow of outgoing edges $-$ total flow of incoming edges.
  - What is the flow of this cut?
  - 19

Are the flows of every cut the same?



- It seems that the flow of every cut is the same, let's prove this!
- Let $F_{out}(v)$ denote the total flow going out of vertex v and $F_{in}(v)$ denote the total flow coming into vertex v.
- Flow conservation property: $F_{out}(v) - F_{in}(v) = 0$ for every vertex $v \in V \setminus \{s, t\}$.
- Flow of the network $= F_{out}(s)$.

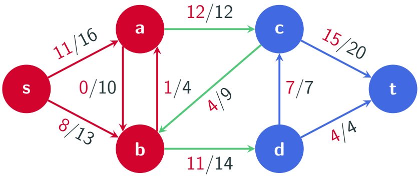
Flow of every cut is the same



$$
\begin{aligned}
F_{out}(s) &= F_{out}(s) + \sum_{v \in S \setminus s} (F_{out}(v) - F_{in}(v)) \\
&= \sum_{v \in S} (F_{out}(v) - F_{in}(v)) \\
&= \sum_{v \in S} (F_{out}(v) + F_{out}(v) - F_{in}(v) - F_{in}(v))
\end{aligned}
$$

1st equation: by flow conservation, 2nd equation: as source has no incoming edges and so $F_{in}(s) = 0$, 3rd equation: just separates the flow through red and green arrows.

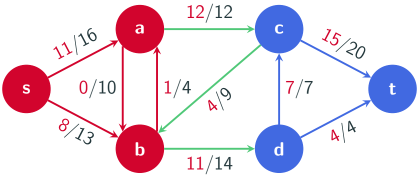
Flow of every cut is the same



$$
\begin{aligned}
F_{out}(s) &= \sum_{v \in S} \left(F_{out}(v) + F_{out}(v) - F_{in}(v) - F_{in}(v)\right) \\
&= \sum_{v \in S} \left(F_{out}(v) - F_{in}(v)\right) + \sum_{v \in S} \left(F_{out}(v) - F_{in}(v)\right) \\
&= \sum_{v \in S} \left(F_{out}(v) - F_{in}(v)\right)
\end{aligned}
$$

The last equation follows from the fact that each red edge appears once as an incoming edge and once as an outgoing edge.

Flow of every cut is the same



$$F_{out}(s) \quad = \quad \sum_{v \in S} (F_{out}(v) - F_{in}(v))$$

We conclude that the flow of any cut is equal to the flow of the network.

Min-cut

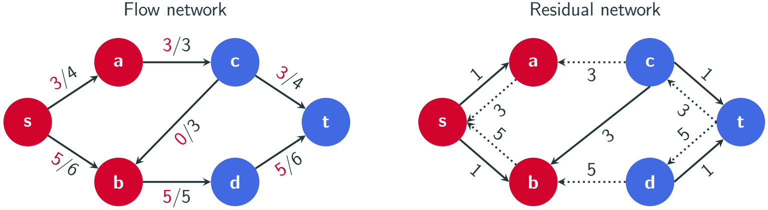
- **Min-cut of a flow network is the cut with the minimum capacity.**
- Flow of the network $=$ flow of any cut $\leq$ capacity of that cut.
- Maximum possible flow of the network $\leq$ capacity of min-cut.
- What if we can find a pair of flow and cut such that the flow of the network $=$ capacity of the cut?
  - The flow value cannot be increased any further as it would violate the capacity of that cut, so it is the maximum flow.
  - The cut is a min-cut of the flow network.

Min-cut Max-flow Theorem

Maximum possible flow of a network = capacity of the min-cut.

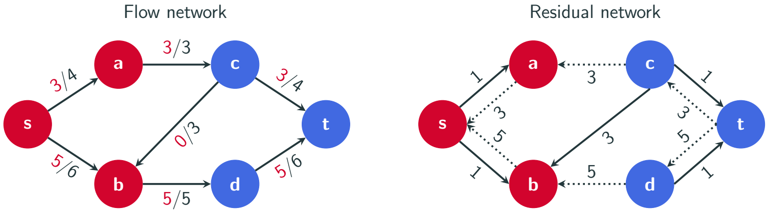
We will show that the Ford-Fulkerson method always terminates and outputs a flow with value equal to the capacity of the min-cut.

Proof of correctness



- Suppose the Ford-Fulkerson method has terminated (i.e., there does not exist any augmenting path in the residual network).
- We define a cut (S, T) such that:
  - S contains every vertex v that is reachable from s in the residual network.
  - T contains every other vertex. Note t cannot be in S because it is not reachable from s (there is no further augmenting paths).

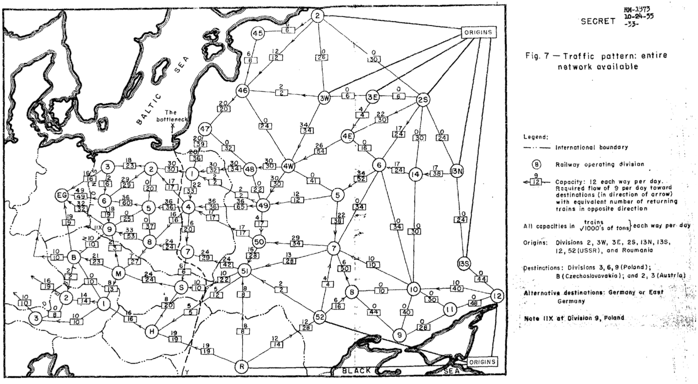
Proof of correctness



- The flow of this cut equals to the capacity of this cut:
  - For each outgoing edge, e.g., $a \to c$, its flow is equal to the capacity of the edge. Otherwise, edge $a \to c$ would be in the residual network and c would be reachable from s (but we know this is not the case as $c \notin S$).
  - For each incoming edge, e.g., $c \to b$, its flow is zero. Otherwise, there would be an edge $b \to c$ in the residual network implying c is reachable from s (but we know this is not the case as $c \notin S$).

Min-cut max-flow connection used in practice

Original application: US Air Force wanted to identify targets for air strikes to effectively cut off Soviet supply chains in Eastern Europe in case of a conflict during the Cold War.



Min-cut max-flow connection used in practice

- Manage traffic flow and preventing bottlenecks on roads.
- Identifying parts that can lead to catastrophic failures in the power grid.
- Businesses use it to analyse critical parts of their supply chain and operational networks.
- Social networks use it to analyse and optimise flow of information.
- It can help identifying potential vulnerabilities in complex systems and planning to increase their resilience.

Reading

- Course Notes: Chapter 9
- You can also check algorithms' textbooks for contents related to this lecture, e.g.:
    - CLRS: Sections 26.1 and 26.2
    - KT: Sections 7.1, 7.2 and 7.3

Concluding remarks

- Take home message: maximum flow of a network is equal to the capacity of its min-cut and both can be found using Ford-Fulkerson.
- Things to do: **make sure you understand Ford-Fulkerson algorithm, why it is correct, and how to use it to find max-flows and min-cuts.**
- Coming up next: Circulation with demands, applications of network flow to combinatorial problems.