

Harvard University SIMR (Summer Introduction to Mathematical Research) Network Flow

Yongwhan Lim
Research Mentor in SIMR at Harvard
Lecturer in EECS at MIT
Associate in Computer Science at Columbia
7:45pm ET, Tuesday, July 5, 2022



Yongwhan Lim



- **Currently:**
 - (July 11~) Senior Quantitative Software Engineer at Two Sigma
 - Research Mentor in SIMR at Harvard University
 - Lecturer in EECS at MIT
 - Associate in Computer Science at Columbia University
 - ICPC Head Coach at Columbia University
 - ICPC Judge for Mid-central and Greater New York regions in N.A.
- **Previously:**
 - Research Software Engineer at Google Research
- **Education:**
 - Stanford: Math & CS (BS '11) and CS (MS '13)
 - MIT: Operations Research (PhD, started in 2013 but on an extended leave-of-absence since 2016)



Network Flow: Real-life Example

- We are given a directed graph, where each vertex represents a city and each directed edge represents a one-way road from one city to another.
- Suppose we want to send trucks from city s to city t and the **capacity** of each directed edge represents the number of trucks that can go on that road every hour.



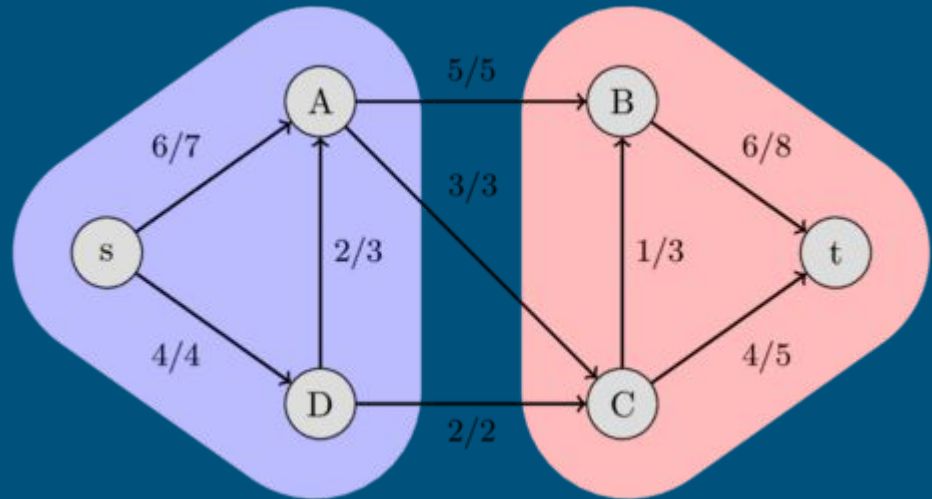
[Photo Credit](#)

Network Flow: Real-life Example (con't)

- We are given a directed graph, where each vertex represents a city and each directed edge represents a one-way road from one city to another.
- Suppose we want to send trucks from city s to city t and the **capacity** of each directed edge represents the number of trucks that can go on that road every hour.
- What is the **maximum number** of trucks that we can send from s to t every hour?

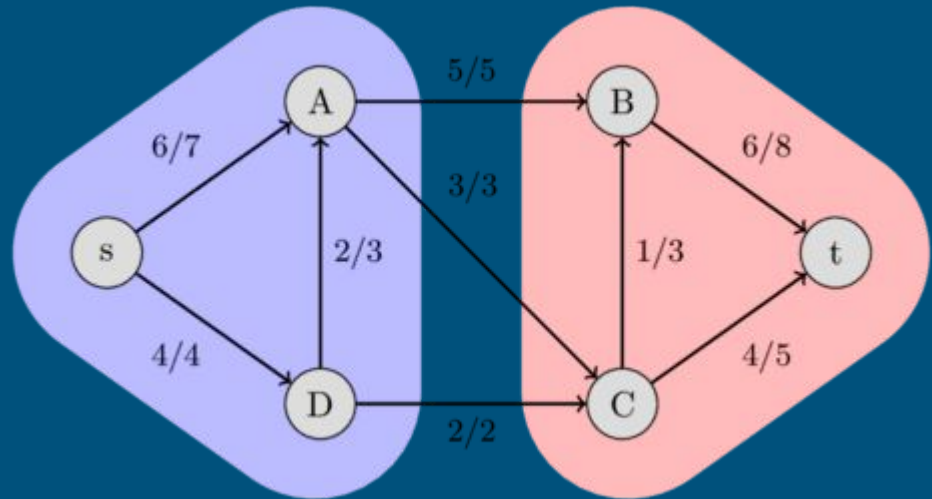
Lecture: Overview

- Flow Network
- Maximum Flow Problem
- Ford-Fulkerson Method
- Edmonds-Karp Algorithm
- Integral Flow Theorem
- s-t cut and its capacity
- Flow Value Lemma
- Flow Weak Duality
- Flow Certificate of Optimality
- Max-flow Min-cut Theorem
- Menger's Theorem
- Further Topics in Flows
- References
- Contact Information



Lecture: Overview

- Flow Network
- **Maximum Flow Problem**
- **Ford-Fulkerson Method**
- Edmonds-Karp Algorithm
- **Integral Flow Theorem**
- s-t cut and its capacity
- Flow Value Lemma
- Flow Weak Duality
- Flow Certificate of Optimality
- **Max-flow Min-cut Theorem**
- **Menger's Theorem**
- Further Topics in Flows
- References
- Contact Information



Flow Network

- A **network** is a directed graph G with vertices V and edges E combined with a function c , which assigns each edge e a non-negative integer value, the **capacity** of e .
- Such a network is called a **flow network**, if we additionally label two vertices, one as **source** and the other as **sink**.
- **Assumption:** We disallow a reverse edge $(v, u) \in E$ for each edge $(u, v) \in E$.

Flow Network (con't)

- A **flow** in a flow network is function f , that again assigns each edge e a non-negative integer value, namely the flow. The function has to fulfill the following conditions:
 - The flow of an edge cannot exceed the capacity: $f(e) \leq c(e)$.
 - The sum of the incoming flow of a vertex u has to be equal to the sum of the outgoing flow of u (except in the source and sink vertices): $\sum_v f((v, u)) = \sum_w f((u, w))$.
- The source vertex s only has outgoing flows.
- The sink vertex t only has incoming flows.
- $\sum_v f((v, t)) = \sum_w f((s, w))$.

Maximum Flow Problem

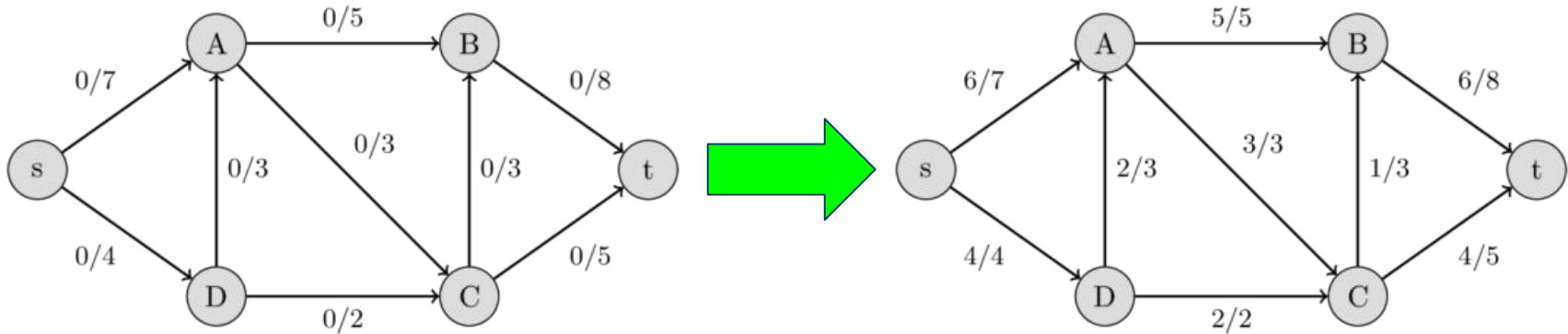
- The value of the flow of a network is the sum of all the flows that get produced in the source s , or equivalently to the sum of all the flows that are consumed by the sink t .
- Formally, $\text{val}(f)$, the **value** of a flow f , is defined as:

$$\sum_{e \text{ out of } s} f(e) - \sum_{e \text{ in to } s} f(e).$$

- A **maximum flow** is a flow with the maximum possible value of $\text{val}(f)$.
- **Problem: Find this maximum flow of a flow network!**

Ford-Fulkerson Method: Example

- The first value of each edge represents the flow, which is initially 0, and the second value represents the capacity.



Ford-Fulkerson Method (1956)

- A **residual capacity** of an directed edge is the capacity minus the flow.
 - For each edge (u, v) , we can create a reverse edge (v, u) with capacity 0 such that $f((v, u)) = -f((u, v))$.
 - This also defines the residual capacity for all the reversed edges.
- We can create a **residual network** from all these edges, which is just a network with the same vertices and edges, but we use the residual capacities as capacities.

Ford-Fulkerson Method (con't)

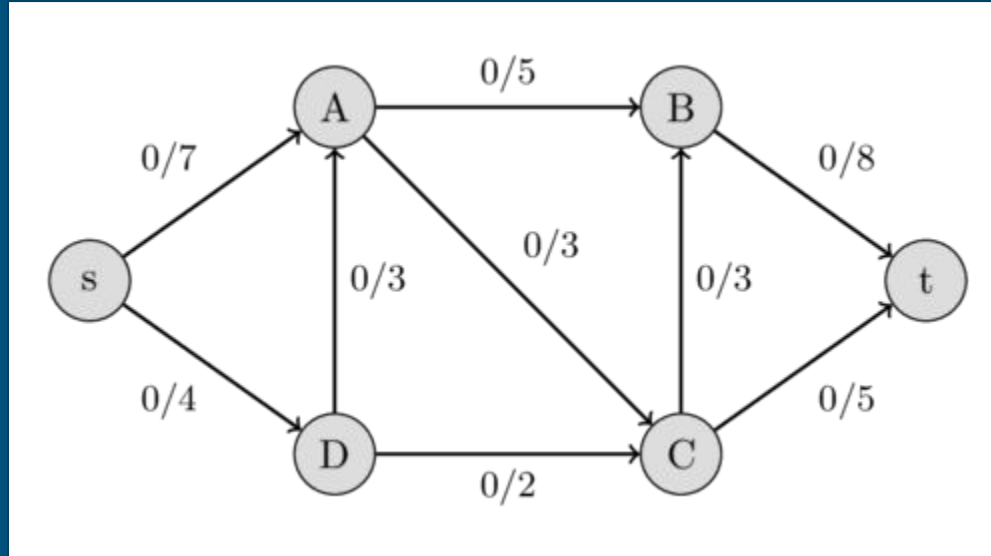
1. We set the flow of each edge to 0.
2. We look for an **augmenting path** from s to t .
 - An augmenting path is a simple path in the residual graph by always taking the edges whose residual capacity is **positive**.
3. If such a path is found then we can increase the flow along these edges.
 - Let C be the smallest residual capacity of the edges in the path. Then we increase the flow by updating $f((u, v)) += C$ and $f((v, u)) -= C$ for every edge (u, v) in the path.
4. Else, terminate! (A flow found is maximum).

Ford-Fulkerson Method: Augmenting Path?

- Ford-Fulkerson method **does not specify** how to find an augmenting path.
- Possible approaches are using *Depth-First Search* (DFS) or *Breadth-First Search* (BFS), both of which work in $O(E)$.
- **Integral** capacities
 - For each augmenting path, the flow of the network increases by at least 1. So, the complexity of Ford-Fulkerson is $O(EF)$ where F is the maximum flow of the network (but, usually *much faster* in practice)!
- **Rational** capacities
 - The algorithm will terminate but the complexity is not bounded.
- **Irrational** capacities
 - The algorithm might never terminate and might not even converge to the maximum flow.

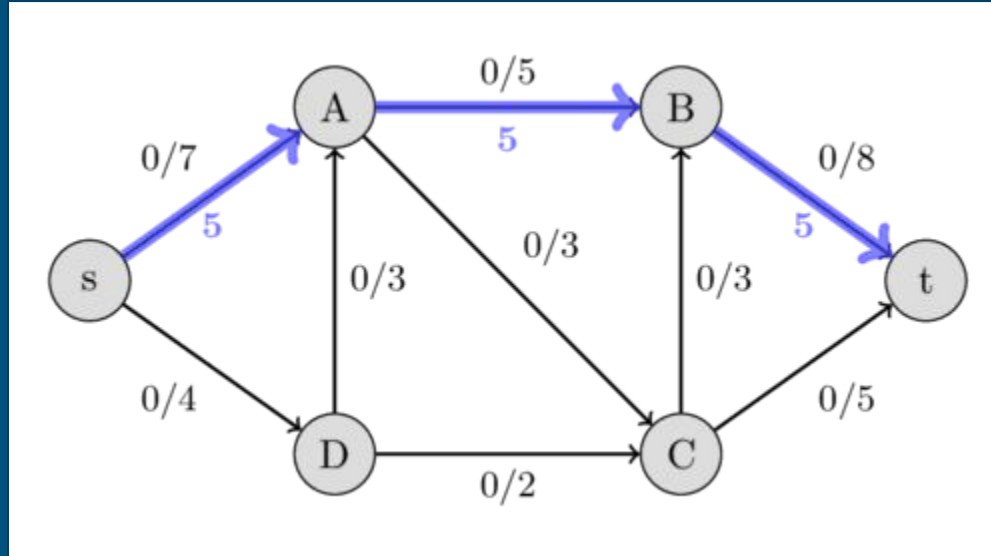
Ford-Fulkerson Method: Example

flow = 0



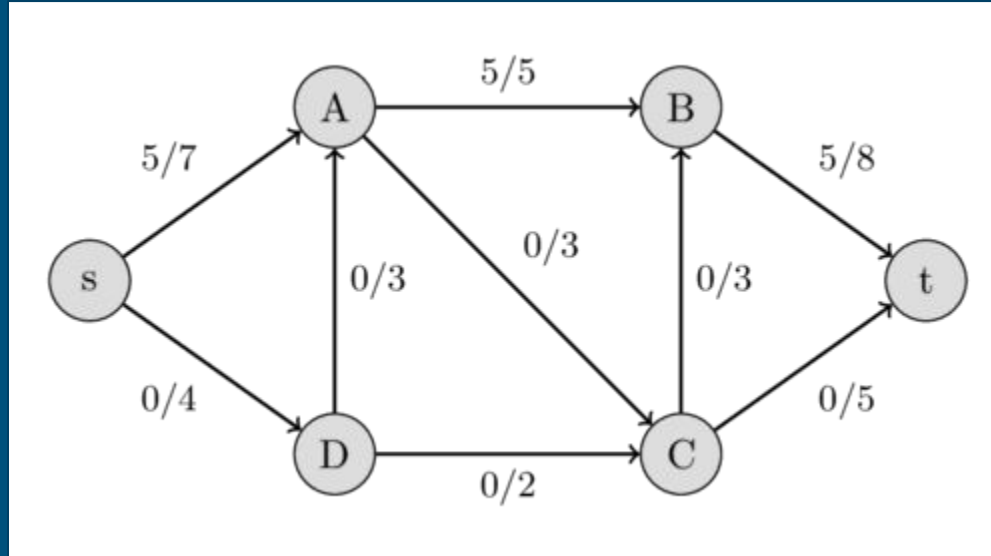
Ford-Fulkerson Method: Example

flow = 0



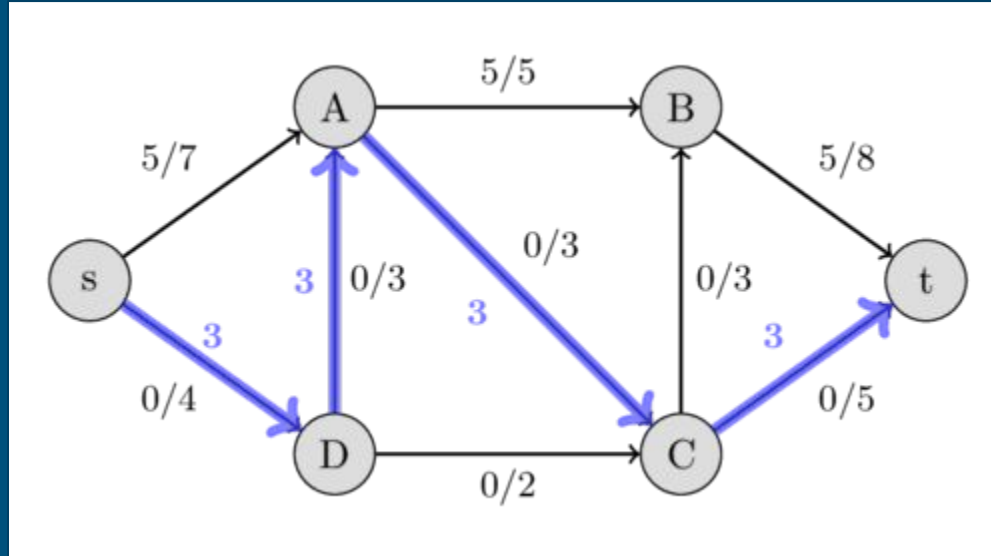
Ford-Fulkerson Method: Example

flow = 5



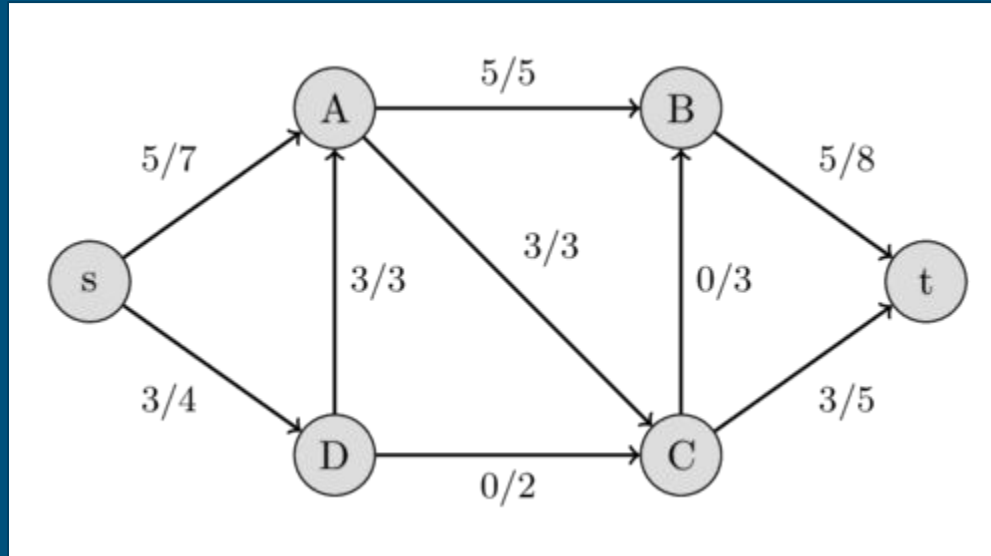
Ford-Fulkerson Method: Example

flow = 5



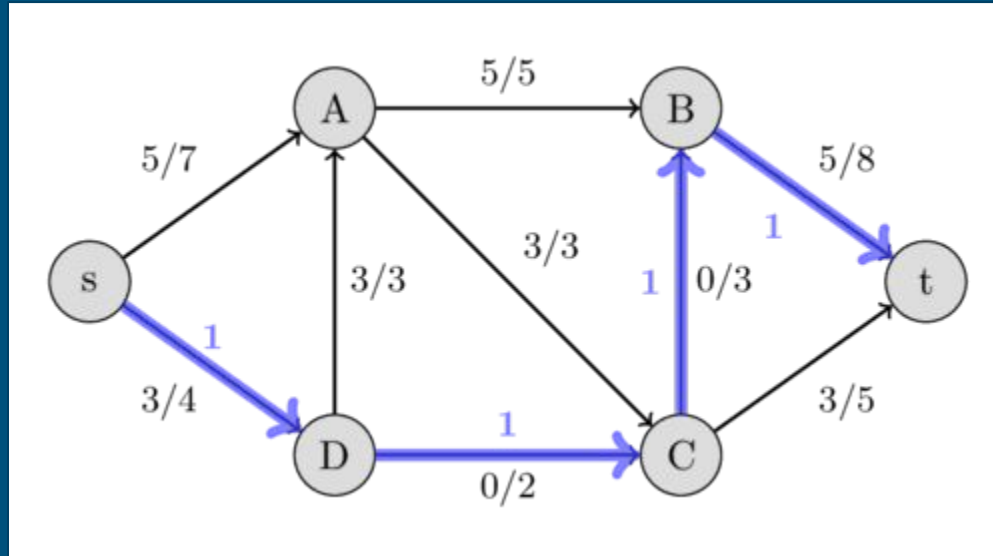
Ford-Fulkerson Method: Example

flow = 8



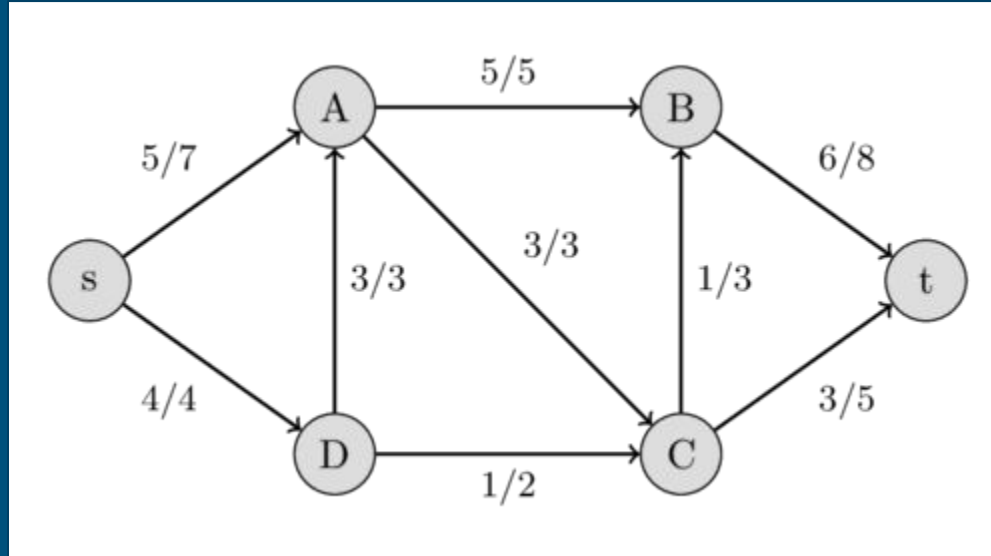
Ford-Fulkerson Method: Example

flow = 8



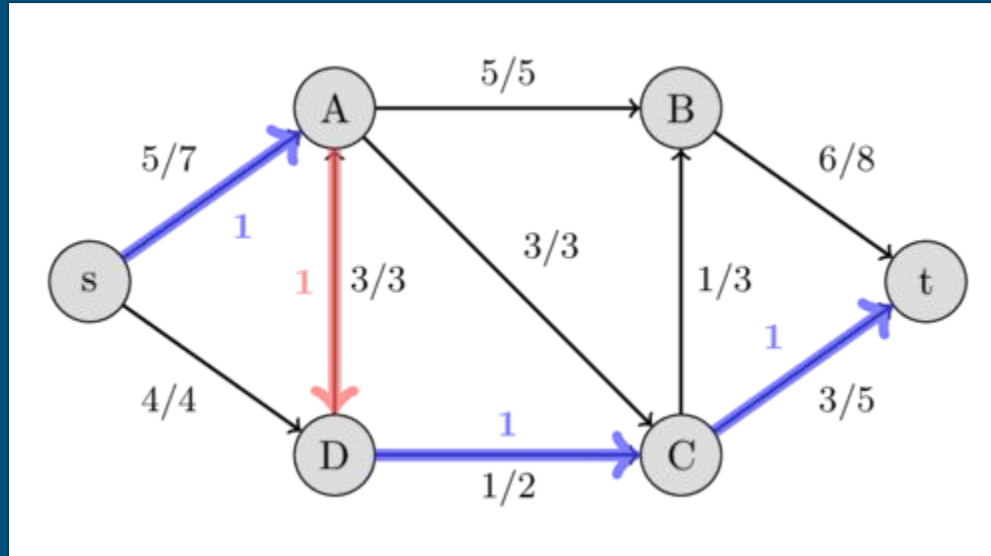
Ford-Fulkerson Method: Example

flow = 9



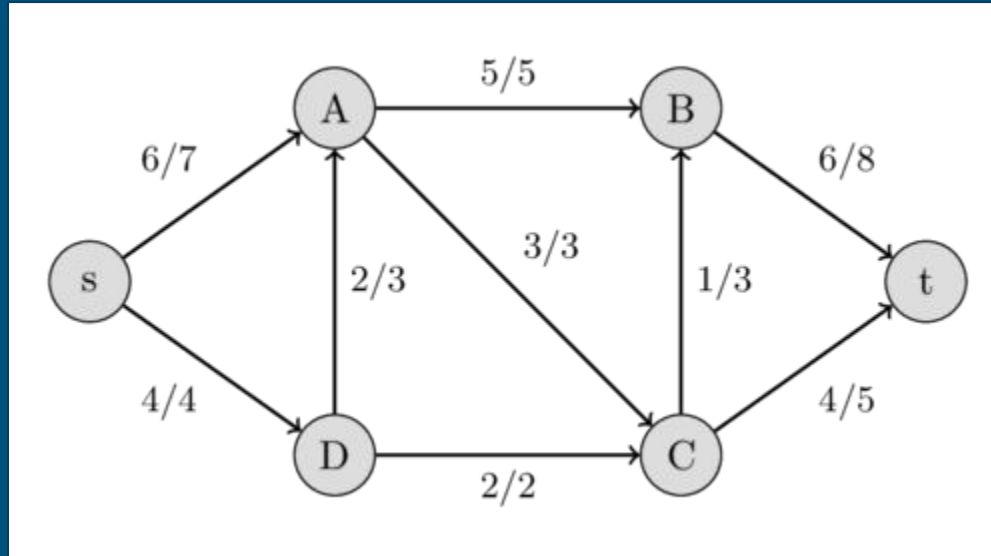
Ford-Fulkerson Method: Example

flow = 9



Ford-Fulkerson Method: Example

flow = 10



Edmonds-Karp Algorithm (1972)

- **Edmonds-Karp algorithm** is just an implementation of the Ford-Fulkerson method that uses BFS for finding augmenting paths.
- First published by Yefim Dinitz in 1970; later, independently published by Jack Edmonds and Richard Karp in 1972.
- The complexity can be given *independently* of the maximum flow. The algorithm runs in $O(VE^2)$ time (yes! even for irrational capacities).
- **Intuition**
 - Every time we find an augmenting path one of the edges becomes saturated and the distance from the edge to s will be longer if it appears again in an augmenting path later. The path length is bounded by V .

Integral Flow Theorem: Statement

- Suppose a given graph has each of its capacity integral. Then, there exists a maximum flow f where every flow value $f(e)$ is an integer.

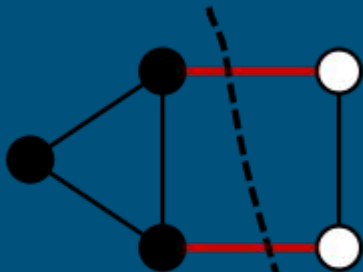
Integral Flow Theorem: Proof

- Suppose a given graph has each of its capacity integral. Then, there exists a maximum flow f where every flow value $f(e)$ is an integer.
- In each step of Edmonds-Karp algorithm, the augmenting path will change a flow value and each residual capacity by **integral** amount.
- Therefore, as long as the algorithm terminates in finite steps, a maximum flow would have flow values that are integral too!

s-t cut and its capacity

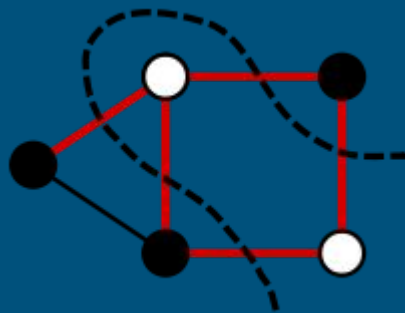
- An **s-t cut** is a partition (A, B) of the vertices with $s \in A$ and $t \in B$.
- The capacity of a cut, **cap**(**A**, **B**), is the sum of capacities of the edges from A to B.

Minimum Cut



Capacity = 2

Maximum Cut



Capacity = 5

Flow Value Lemma: Statement

- Let f be any flow and let (A, B) be any cut. Then,

$$\text{val}(f) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e).$$

Flow Value Lemma: Proof

- Let f be any flow and let (A, B) be any cut. Then,

$$\text{val}(f) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e).$$

$$\text{val}(f) = \sum_{e \text{ out of } s} f(e) - \sum_{e \text{ in to } s} f(e)$$

$$= \sum_{v \in A} (\sum_{e \text{ out of } v} f(e) - \sum_{e \text{ in to } v} f(e))$$

$$= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) //$$

Definition of $\text{val}(f)$

All terms except $v = s$ are 0

Edges completely in A cancel

Flow Weak Duality Lemma: Statement

- Let f be any flow and let (A, B) be any cut. Then,
$$\text{val}(f) \leq \text{cap}(A, B).$$

Flow Weak Duality Lemma: Proof

- Let f be any flow and let (A, B) be any cut. Then,
 $\text{val}(f) \leq \text{cap}(A, B)$.

$$\begin{aligned}\text{val}(f) &= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \\ &\leq \sum_{e \text{ out of } A} f(e) \\ &\leq \sum_{e \text{ out of } A} c(e) \\ &= \text{cap}(A, B) //\end{aligned}$$

Flow Value Lemma

Flows are non-negative

Capacity constraint

Flow Certificate of Optimality: Statement

- Let f be a flow and let (A, B) be any cut. If $\text{val}(f) = \text{cap}(A, B)$ then f is a maximum flow and (A, B) is a minimum cut.

Flow Certificate of Optimality: Proof

- Let f be a flow and let (A, B) be any cut. If $\text{val}(f) = \text{cap}(A, B)$ then f is a maximum flow and (A, B) is a minimum cut.
- By Weak Duality Lemma, for any flow f' , we have $\text{val}(f') \leq \text{cap}(A, B) = \text{val}(f)$.
- Similarly, for any cut (A', B') , we have $\text{cap}(A', B') \geq \text{val}(f) = \text{cap}(A, B)$ //

Max-flow Min-cut Theorem: Statement

- The maximum value of an s-t flow is equal to the minimum capacity over all s-t cuts.

Max-flow Min-cut Theorem: Proof Sketch

- Suffices to show the following three conditions are equivalent:
 - There exists a cut (A, B) such that $\text{cap}(A, B) = \text{val}(f)$;
 - f is a maximum flow;
 - There is no augmenting path with respect to f ;
- **(1 \Rightarrow 2)**
 - Weak duality!
- **(2 \Rightarrow 3)**
 - We prove the contrapositive. If there is an augmenting path then f is not a maximum flow. If f' is the flow achievable in an augmenting path then $f + f'$ is a flow in G , which means f is not a maximum flow.
- **(3 \Rightarrow 1)**
 - Let f be a flow with no augmenting paths. Let A be the set of vertices reachable from s in the residual network. By the definition of A , $s \in A$. By the definition of flow f , $t \notin A$. So,

$$\text{val}(f) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) = \sum_{e \text{ out of } A} f(e) - 0 = \text{cap}(A, B).$$

Menger's Theorem

- Let G be a finite directed graph and let s and t be two nonadjacent vertices.
- The size of the minimum vertex cut for s and t (i.e., the minimum number of vertices, distinct from s and t , whose removal disconnects s and t) is equal to the maximum number of pairwise internally vertex-disjoint paths from s to t .

Menger's Theorem: Proof Sketch

- Let G be a finite directed graph and let s and t be two nonadjacent vertices.
- The size of the minimum vertex cut for s and t (i.e., the minimum number of vertices, distinct from s and t , whose removal disconnects s and t) is equal to the maximum number of pairwise internally vertex-disjoint paths from s to t .
- Consider the following flow network:
 - Split each vertex $v \in V$ into v_{in} and v_{out} and add the edge (v_{in}, v_{out}) with capacity 1.
 - Each edge $(u, v) \in E$ becomes (u_{out}, v_{in}) with capacity ∞ .
- Use the maximum-flow minimum-cut theorem!

Further Topics in Flows

- Push-relabel algorithm
- Dinic's algorithm (most of the time, this algorithm is *sufficiently fast*!)
- Malhotra-Pramodh-Maheshwari (MPM) algorithm
- Flows with demands
- Minimum cost flow
- Assignment problem
- Nearly-linear(!) time algorithm (Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, Sushant Sachdeva)
- Weighted/Unweighted Bipartite/General Matching
- ...
- Dynamic graphs! (Professor Richard Peng's lecture, scheduled on July 14)
- ...

References

- https://cp-algorithms.com/graph/edmonds_karp.html
- https://faculty.cc.gatech.edu/~rpeng/CS7510_F19/Sep3MaxFlowMinCut.pdf
- <https://www.cs.cmu.edu/~avrim/451f11/lectures/lect1025.pdf>
- <https://web.stanford.edu/class/archive/cs/cs161/cs161.1168/lecture16.pdf>
- <http://www.cs.toronto.edu/~lalla/373s16/notes/MFMC.pdf>
- Introduction to Algorithms (CLRS), 4th Edition
- Wikipedia

Contact Information

- yongwhan@mit.edu or yongwhan.lim@columbia.edu
 - <https://cs.columbia.edu/~yongwhan>
 - <https://www.linkedin.com/in/yongwhan/>
-
- For those who are interested in **quantitative analysis in finance**, feel free to send me an email to schedule 1:1 zoom meeting.