
THE UNIVERSITY OF
AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

Algorithms and Data Structures

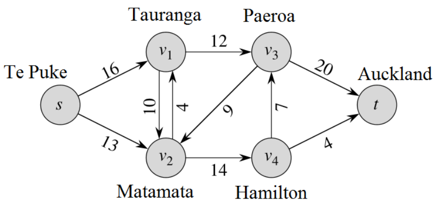
Lecture 22 Flow Network

Jiamou Liu
The University of Auckland

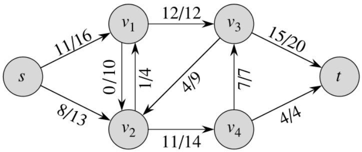
Part I: Flow Network – Basic Terminology

A Shipping Problem

A farmer in plans to lease trucks to ship his fruits to Auckland. Due to road and traffic conditions, the trucks can ship at most $c(u, v)$ cartons per day between each pair of towns u and v. What is the maximum amount of fruits that he can ship per day?

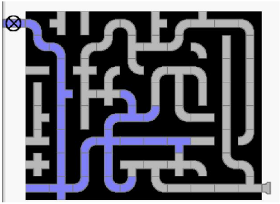


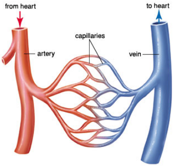
A possible shipping arrangement

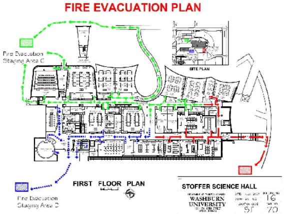


Flow Network

- Liquid (pipeline/blood vessel) networks
- Electricity networks
- Traffic networks
- Evacuation networks
- Telecommunication networks



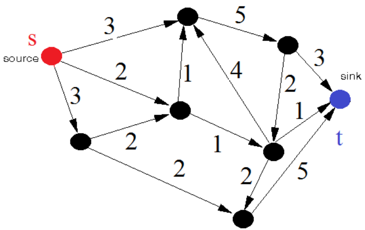




Flow Network

Definition [Flow Network]

- A source node in a directed graph is a node with no incoming edges.
- A sink node in a directed graph is a node with no outgoing edges.
- A flow network is a directed graph (V, E, c, s, t) where (V, E, c) forms a weighted graph with positive edge weights, $s \in V$ is a source node and $t \in V$ is a sink node.

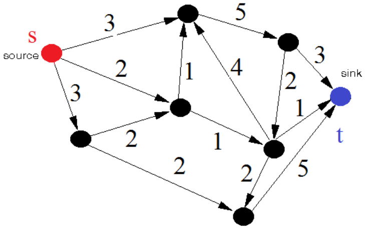


Definition [Flow]

A flow in a flow network (V, E, c, s, t) is a function $f : E \to \mathbb{N}$ such that $f(u, v) \le c(u, v)$ for every $(u, v) \in E$ and

for any node u that is not s nor t, we have $\sum_{(u,x)\in E} f(u, x) = \sum_{(y,u)\in E} f(y, u)$.

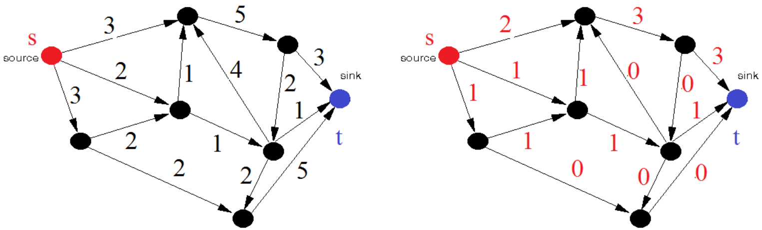
Intuitively, a flow defines a way to send oil from s to t without exceeding the pipeline capacity, and without any leak on the way.



Definition [Size of a Flow]

The size of a flow f in a flow network is $size(f) = \sum\{f(s,u) \mid (s,u) \in E\}$. In other words, it is the total quantity of stuff sent from s to t.

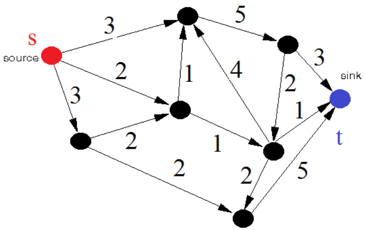
Example: Size of the following flow = 4

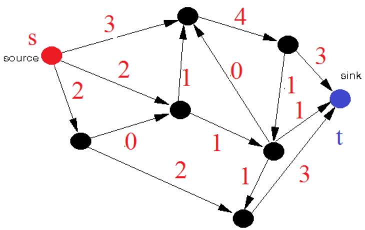


Definition [Size of a Flow]

The size of a flow f in a flow network is $size(f) = \sum\{f(s,u) \mid (s,u) \in E\}$. In other words, it is the total quantity of stuff sent from s to t.

Example: Size of the following flow = 7



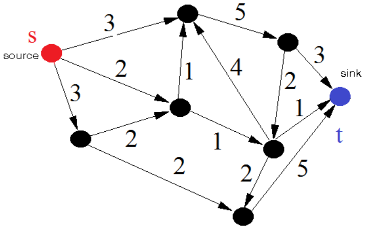


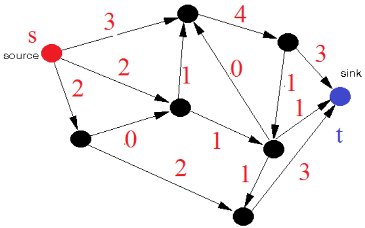
Maximal Flows in a Flow Network

Definition [Maximal Flow]

A flow in a flow network is maximal if it has maximal size.

Example: the following is a maximal flow.



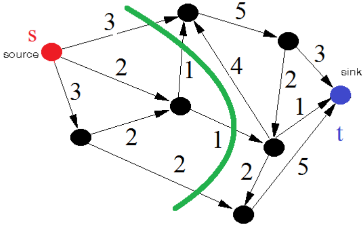


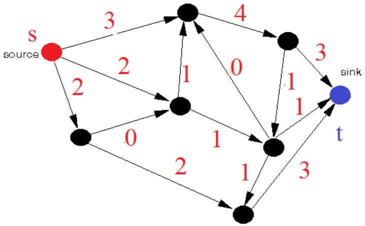
Cuts in a Flow Network

Definition [Cut]

A cut in a graph is a set C of edges such that removing them would disconnect the graph into $left(C)$ (nodes reachable from s) and the right part $right(C)$ (nodes that may reach t).

The green line shows a cut:





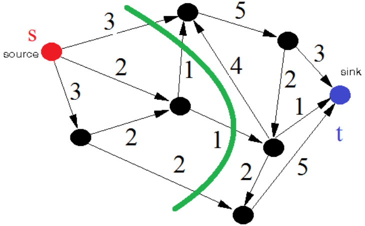
Definition [Capacity of a Cut]

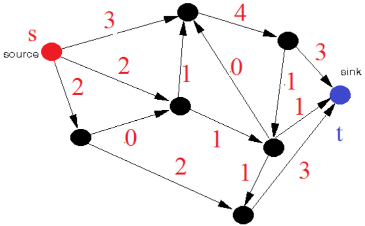
The capacity of a cut C in a flow network is the sum of the capacity of edges that goes from left to right:

$$capacity(C) = \sum \{c(u,v) \mid (u,v) \in E, u \in left(C), v \in right(C)\}$$

A minimal cut is a cut with minimal capacity.

For example, the green line shows a minimal cut of capacity 7.

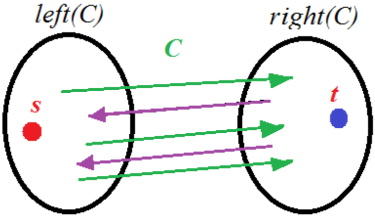




Cuts in a Flow

Let C be a cut in a network G and f be a flow. The net flow of f over C

$$f(C) = \sum \{f(u,v) \mid (u,v) \in E, u \in left(C), v \in right(C)\}$$
$$- \sum \{f(v,u) \mid (v,u) \in E, u \in left(C), v \in right(C)\}$$

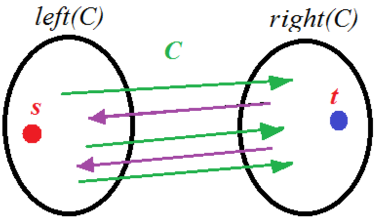


Flow and Cut

Fact 1

For any cut C and flow f,

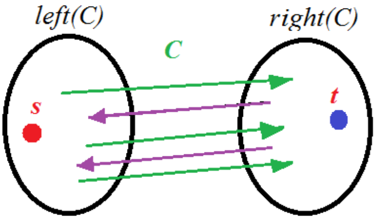
$$size(f) = f(C)$$



Fact 2

For any flow network G,

$$\text{the size of max-flow} \leq \text{the capacity of min-cut}$$



Two Problems

Max-Flow Problem

Given a flow network, find a maximal flow in the network?
In other words, what is the maximal quantity we can send from s to t?

Min-Cut Problem

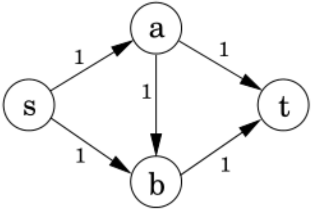
Given a flow network, find a minimal cut in the network?
In other words, what is the bottleneck across the entire network?

Later we will show that these two are the same problem.

Part II: Max-Flow Min-Cut

A Network Example

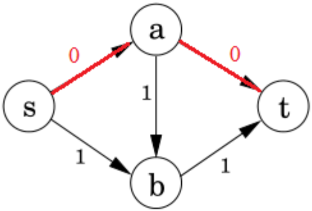
Example:

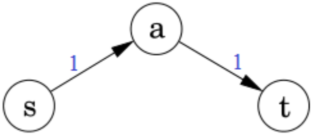


1. Start with zero flow
2. Repeat: Choose a path from s to t and maximize flow along this path.

A Network Example

Find a path from s to t with positive capacity:

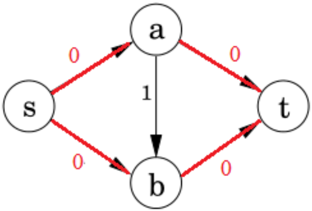


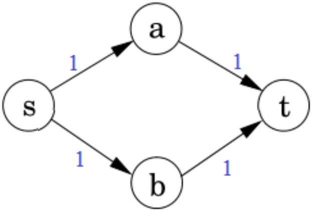


1. Start with zero flow
2. Repeat: Choose a path from s to t and maximize flow along this path.

A Network Example

Find a remaining path from s to t with positive capacity:

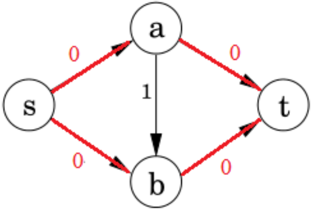


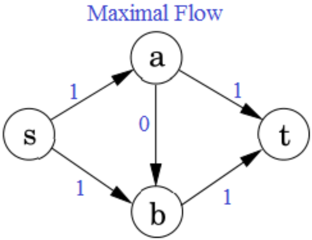


1. Start with zero flow
2. Repeat: Choose a path from s to t and maximize flow along this path.

A Network Example

The final flow

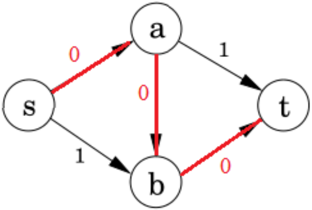


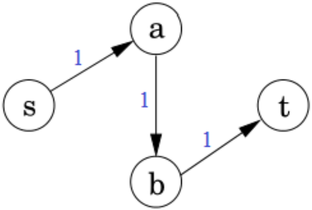


1. Start with zero flow
2. Repeat: Choose a path from s to t and maximize flow along this path.

A Network Example

Potential Problem:

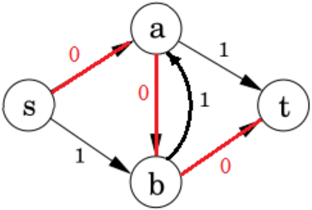


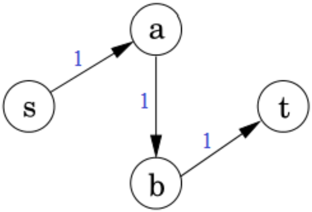


The network on the left represents the amount of additional flow that could be pushed along the edges.
Formally, this is captured using residual networks.

A Network Example

Add a reversed edge for cancelation:

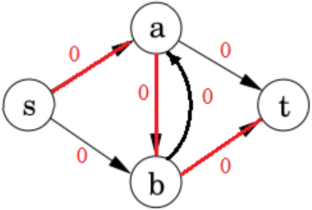


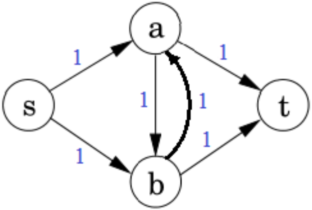


The network on the left represents the amount of additional flow that could be pushed along the edges.
Formally, this is captured using residual networks.

A Network Example

Find a path with reversed edge:

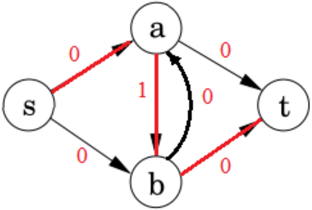




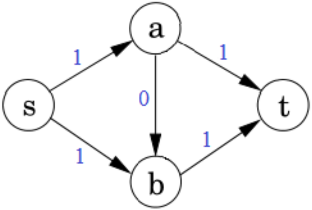
The network on the left represents the amount of additional flow that could be pushed along the edges.
Formally, this is captured using residual networks.

A Network Example

Cancel the two edges:



Maximal Flow



The network on the left represents the amount of additional flow that could be pushed along the edges.
Formally, this is captured using residual networks.

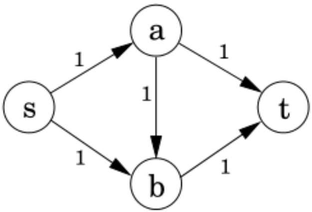
Residual Network

Let $G = (V, E, c, s, t)$ be a flow network and f be a flow in G. The residual network G_f of G and F is another flow network (V, E_f, c_f, s, t):

(a) if $(u, v) \in E$, and $f(u, v) < c(u, v)$, put $(u, v) \in E_f$ and set $c_f(u, v) = c(u, v) - f(u, v)$.

(b) if $f(u, v) > 0$, put $(v, u) \in E_f$ and set $c_f(v, u) = f(u, v)$.

Flow Network G



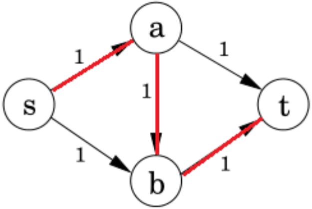
Residual Network

Let $G = (V, E, c, s, t)$ be a flow network and f be a flow in G. The residual network G_f of G and F is another flow network (V, E_f, c_f, s, t):

(a) if $(u, v) \in E$, and $f(u, v) < c(u, v)$, put $(u, v) \in E_f$ and set $c_f(u, v) = c(u, v) - f(u, v)$.

(b) if $f(u, v) > 0$, put $(v, u) \in E_f$ and set $c_f(v, u) = f(u, v)$.

Flow Network G



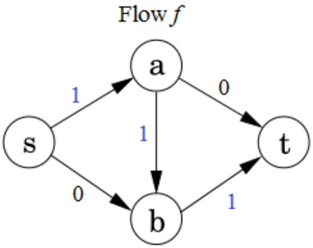
Residual Network

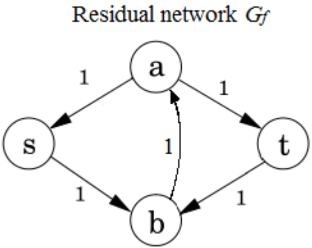
Residual Network

Let $G = (V, E, c, s, t)$ be a flow network and f be a flow in G. The residual network G_f of G and F is another flow network (V, E_f, c_f, s, t):

(a) if $(u, v) \in E$, and $f(u, v) < c(u, v)$, put $(u, v) \in E_f$ and set $c_f(u, v) = c(u, v) - f(u, v)$.

(b) if $f(u, v) > 0$, put $(v, u) \in E_f$ and set $c_f(v, u) = f(u, v)$.





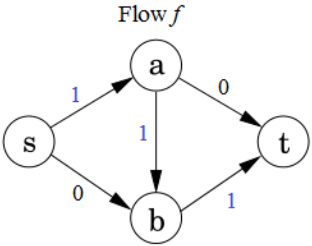
Residual Network

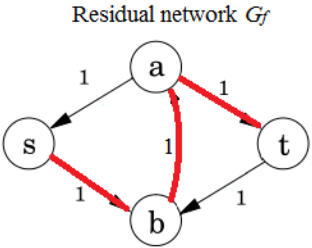
Residual Network

Let $G = (V, E, c, s, t)$ be a flow network and f be a flow in G. The residual network G_f of G and F is another flow network (V, E_f, c_f, s, t):

(a) if $(u, v) \in E$, and $f(u, v) < c(u, v)$, put $(u, v) \in E_f$ and set $c_f(u, v) = c(u, v) - f(u, v)$.

(b) if $f(u, v) > 0$, put $(v, u) \in E_f$ and set $c_f(v, u) = f(u, v)$.





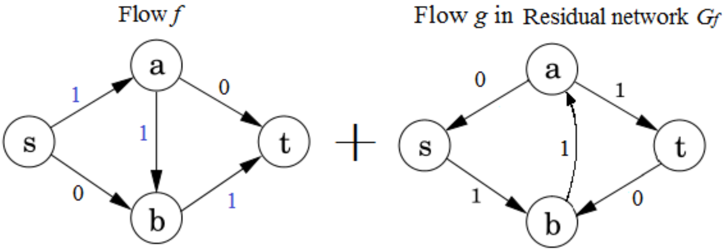
Combining Flows

Combining Flows

Let f be a flow in the network G. Let g be a flow in the residual network G_f. Then we can obtain another flow h in G by adding f and g: For all $(u, v) \in E$, let

$$h(u,v) = f(u,v) + g(u,v) - g(v,u)$$

Note: The flow h is larger than f.



Combining Flows

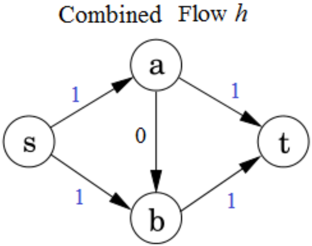
Combining Flows

Let f be a flow in the network G. Let g be a flow in the residual network G_f. Then we can obtain another flow h in G by adding f and g: For all $(u, v) \in E$, let

$$h(u, v) = f(u, v) + g(u, v) - g(v, u)$$

Note: The flow h is larger than f.

$=$



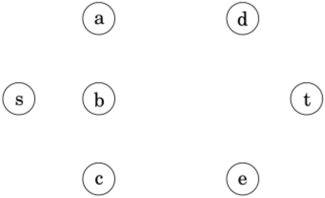
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

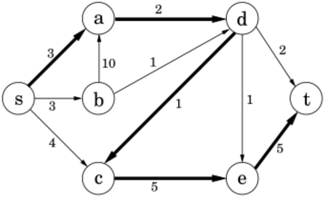
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
  1) Pick a single-path flow from the residual network.
  2) Combine this flow with the current flow
  3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 0:

Current Flow



Current Residual Network



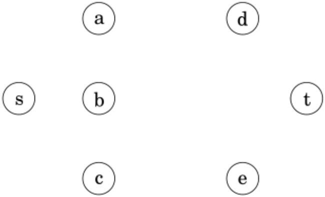
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

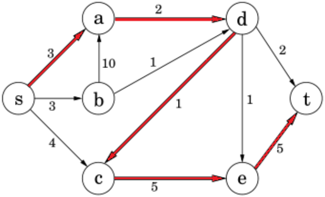
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 1: Pick

Current Flow



Current Residual Network



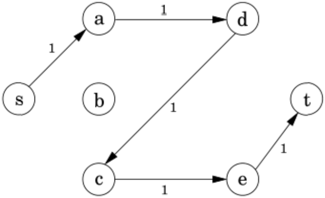
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

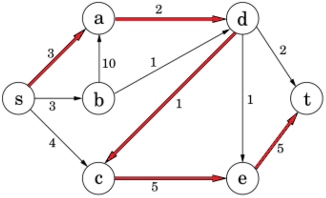
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 1: Combine

Current Flow



Current Residual Network



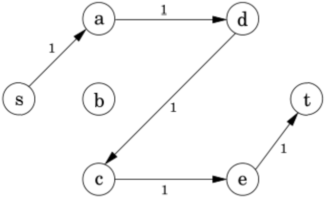
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

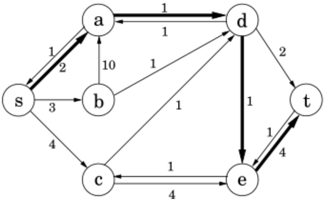
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 1: Update

Current Flow



Current Residual Network



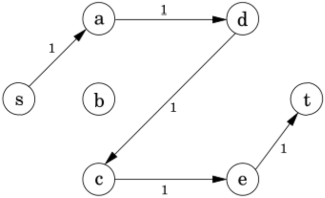
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

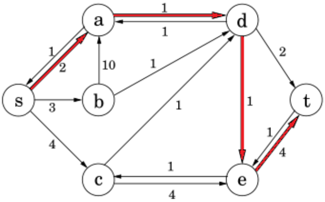
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 2: Pick

Current Flow



Current Residual Network

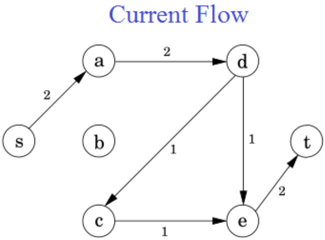


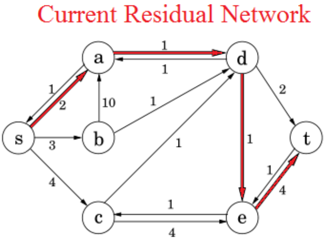
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 2: Combine





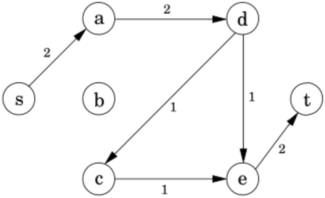
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

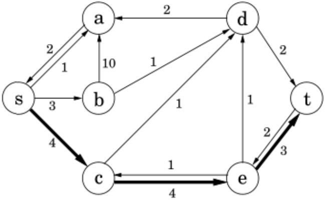
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 2: Update

Current Flow



Current Residual Network



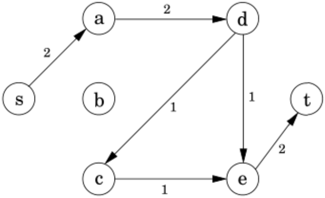
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

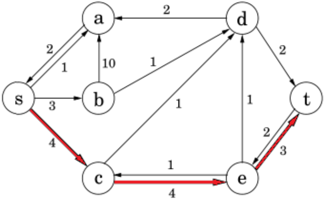
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 3: Pick

Current Flow



Current Residual Network

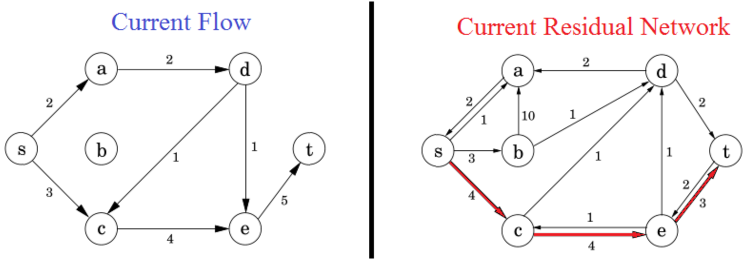


Solving Max-Flow Problem

Ford-Fulkerson Algorithm

INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
  1) Pick a single-path flow from the residual network.
  2) Combine this flow with the current flow
  3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 3: Combine



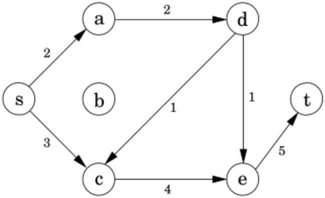
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

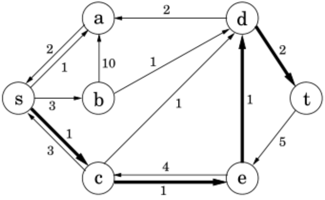
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
  1) Pick a single-path flow from the residual network.
  2) Combine this flow with the current flow
  3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 3: Update

Current Flow



Current Residual Network

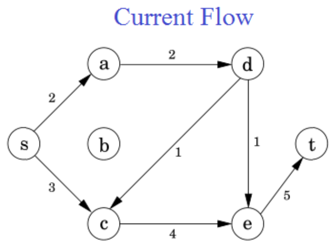


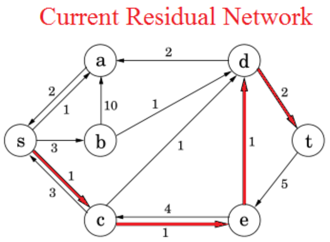
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
  1) Pick a single-path flow from the residual network.
  2) Combine this flow with the current flow
  3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 4: Pick





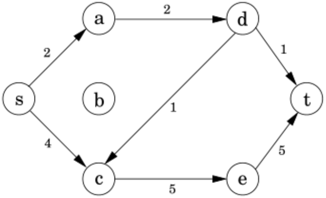
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

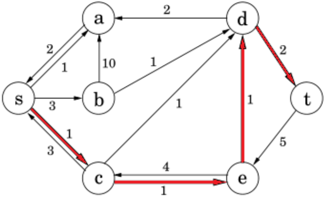
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 4: Combine

Current Flow



Current Residual Network



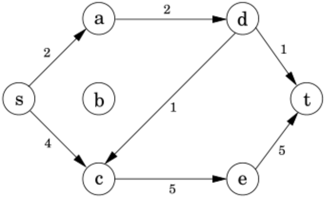
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

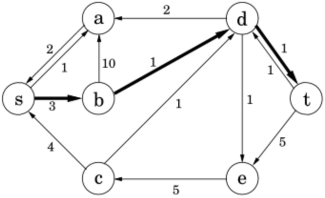
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 4: Update

Current Flow



Current Residual Network

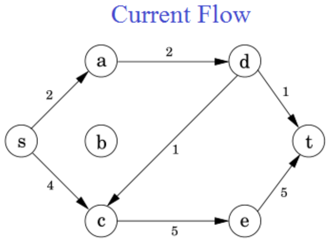


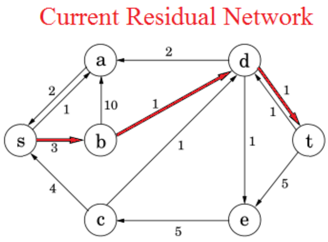
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 5: Pick





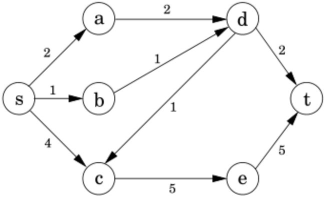
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

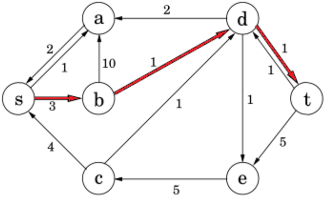
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
  1) Pick a single-path flow from the residual network.
  2) Combine this flow with the current flow
  3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 5: Combine

Current Flow



Current Residual Network



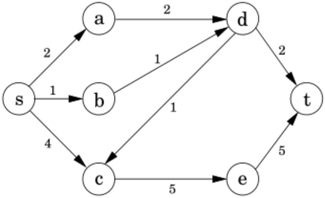
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

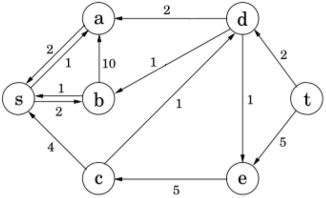
INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Iterationp 5: Update

Current Flow



Current Residual Network

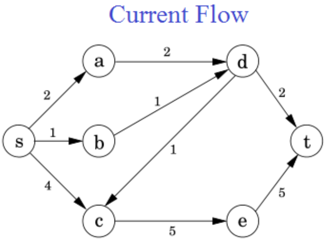


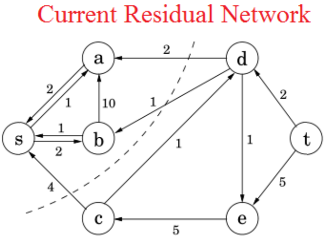
Solving Max-Flow Problem

Ford-Fulkerson Algorithm

INPUT: A flow network $G = (V, E, c, s, t)$
OUTPUT: A maximal flow in G
- Start with empty flow, and full residual network.
- Repeat:
    1) Pick a single-path flow from the residual network.
    2) Combine this flow with the current flow
    3) Update the residual network
- Stop when we cannot find any single-path flow

Stop p

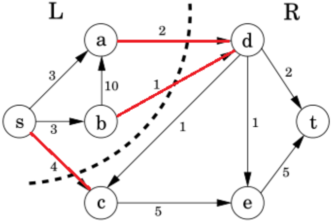




Ford-Fulkerson Algorithm: Correctness

Why does Ford-Fulkerson algorithm find a maximal flow?

Because this flow actually matched the capacity of a cut!



Fact 3

Let f be the flow produced by Ford-Fulkerson. Consider the residual network G_f. Let L be the nodes that are reachable from s in G_f. Let $R = V \setminus L$. Let C be the cut between L and R. Then

$$size(f) = capacity(C)$$

Max-Flow, Min-Cut

Max-Flow Min-Cut Theorem

In any flow network, the size of the maximal flow equals the capacity of the minimal cut.

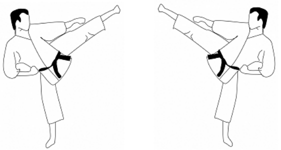
Furthermore

Ford-Fulkerson algorithm finds a maximal flow, as well as a minimal cut in a network.

Part III: Further Applications

Application 1: Community Detection

Zachary's Karate Club [1] A social network of a karate club was studied by Wayne W. Zachary, with 34 members and 78 links between members who interacted outside the club. During the study a conflict arose between the administrator "John A" and instructor "Mr. Hi", which led to the split of the club into two. Half of the members formed a new club around Mr. Hi, members from the other part found a new instructor or gave up karate. Basing on collected data Zachary assigned correctly all but one member of the club to the groups they actually joined after the split.



[1] "An Information Flow Model for Conflict and Fission in Small Groups", Wayne W. Zachary (1977)

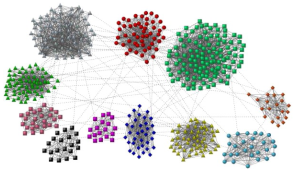
Application 1: Community Detection

Question: How did Zachary come up with the partition?

Community Detection Problem

Partition a given network into subgraphs that capture "realistic communities" in real-life networks:

- Circles of friends
- Political bloggers with similar opinions
- Consumers with similar purchasing behaviours
- Functional groups of proteins (in protein-protein network)

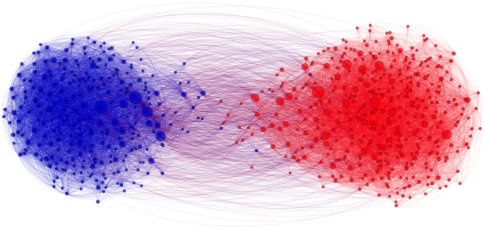


Application 1: Community Detection

Community Detection using Minimum Cut Method

Polarisation: Divide the network into two parts, between which there exist the smallest number of links.

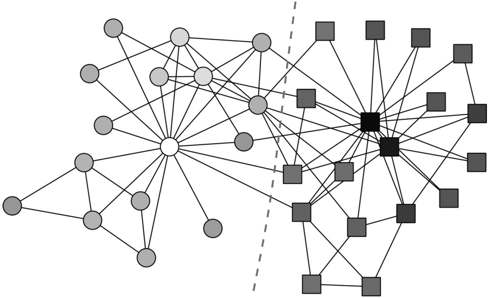
Useful in analysing e.g. political campaign



Note: This is essentially the minimal cut problem.

Application 1: Community Detection

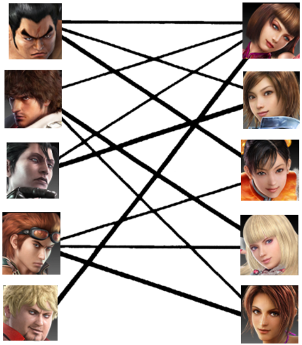
- Ford-Fulkerson's algorithm is the earliest algorithm for community detection
- Zachary run Ford-Fulkerson's algorithm on the social interaction graph of the karate club, with s, t being the administrator and the instructor, respectively, to obtain the following partition:



Application 2: Bipartite Matching

A Problem of Match Making

There are m male and n female users in an online dating website. By analysing the their profiles, we represent possible matchings as a bipartite graph. Now it needs to find a way for matching men with women so that there are as many matched users as possible.

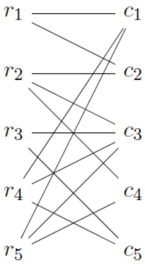


Application 2: Bipartite Matching

Bipartite Matching

Let $G = (X \cup Y, E)$ be a bipartite graph. A matching is a set $M \subseteq E$ of edges that are independent, that is, they do not share any nodes in common.
A node $u \in V = X \cup Y$ is called matched if there is v such that $\{u, v\} \in E$.

Example.

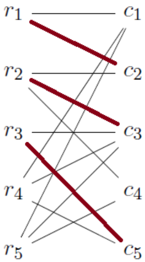


Application 2: Bipartite Matching

Bipartite Matching

Let $G = (X \cup Y, E)$ be a bipartite graph. A matching is a set $M \subseteq E$ of edges that are independent, that is, they do not share any nodes in common.
A node $u \in V = X \cup Y$ is called matched if there is v such that $\{u, v\} \in E$.

Example.



Application 2: Bipartite Matching

Bipartite Matching

A matching M of G is maximal if it is not a proper subset of another matching of G.

A matching of G is perfect if every node in G is matched.

Maximal Bipartite Matching Problem: Given a bipartite graph, compute a maximal matching

Applications of Matching:

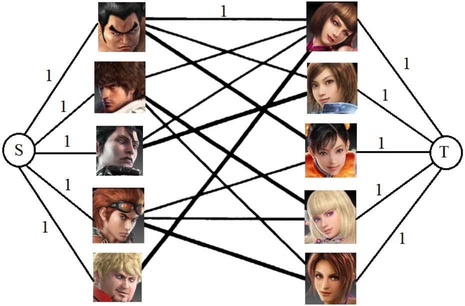
- Consumer v.s. products
- People with required skill set v.s. Tasks

Application 2: Bipartite Matching

We can reduce the bipartite matching problem to maximal flow problem: Given a bipartite graph $(X \cup Y, E)$

1. Add a source node s and a sink node t
2. Add edges between s and all nodes in X
3. Add edges between t and all nodes in Y
4. Set capacity of each edge to 1

Then a maximal flow in the resulting network corresponds to a maximal matching.

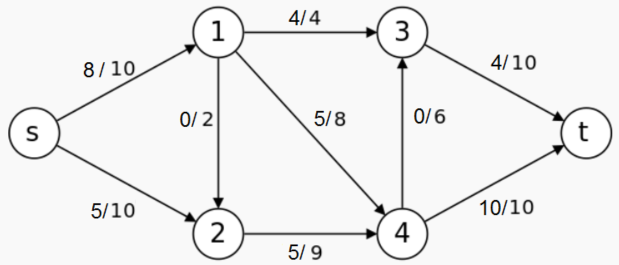


Summary

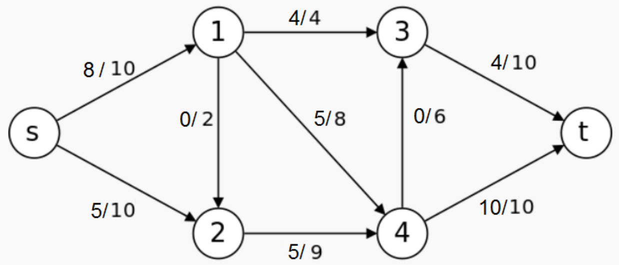
The following are the topics covered in this lecture:

- Maximal flow ≡ Minimal cut
- Ford and Fulkerson algorithm
- Application 1: Community detection (polarisation)
- Application 2: Bipartite matching

Question 1. Consider the following flow in the flow network. Write down its residual network.

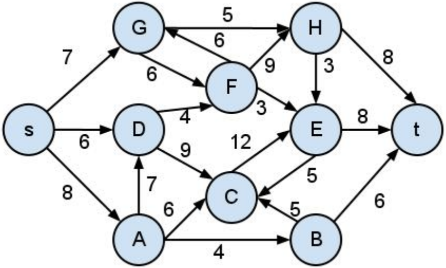


Question 2. Identify a path flow in the residual network you found for the question above, and combine it with the flow given.



Exercise

Question 3. Consider the flow network below. Apply Ford-Fulkerson's algorithm to find a maximal flow and also identify a minimal cut in the network.




THE UNIVERSITY OF
AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND