A Puzzle

How many loads of grain can you ship from $s$ to $t$? Which boats are used?

Max-Flow Problem: Flow Network

Problem input is a flow network:
- Directed graph
- Source node $s$
- Target node or sink $t$
- Edge capacities $c(e) \geq 0$

Solution: A Flow

A network flow is an assignment of values $f(e)$ to each edge $e$, which satisfy:
- Capacity constraints: $0 \leq f(e) \leq c(e)$ for all $e$
- Flow conservation:
  $$\sum_{e \text{ into } v} f(e) = \sum_{e \text{ out of } v} f(e)$$
  for all $v \notin \{s, t\}$.
- Max flow problem: find a flow of maximum value
- Value $v(f)$ of flow $f =$ total flow on edges leaving source

Network Flow

- Previous topics were design techniques
  (Greedy, Divide-and-Conquer, Dynamic Programming)
- Network flow: a specific class of problems with many applications
- Direct applications:
  - commodities in networks
  - transporting goods on the rail network
  - packets on the internet
  - gas through pipes
- Indirect applications:
  - Matching in graphs
  - Airline scheduling
  - Baseball elimination

Plan: design and analyze algorithms for max-flow problem, then apply to solve other problems
First, a Story About Flow and Cuts

Key theme: flows in a network are intimately related to cuts

Soviet rail network (Harris & Ross, RAND report, 1955)


Clicker Question

Let’s recall how a cut is defined:

A: A partition of graph vertices into two subsets
B: A partition of nodes so that the graph is bipartite
C: A set of edges that give a matching between two node sets
D: A set of edges between two node sets so that no two edges cross

Designing a Max-Flow Algorithm

First idea: initialize to zero flow, then repeatedly “augment” flow on paths from $s$ to $t$ until we can no longer do so.

Problem: we are stuck, all paths from $s$ to $t$ have a saturated edge.

“In dealing with the usual railway networks a single flooding, followed by removal of bottlenecks, should lead to a maximal flow.” (Boldyreff, RAND report, 1955)

We’d like to “augment” $s \xrightarrow{+1} v \xleftarrow{-1} u \xrightarrow{+1} t$, but this is not a real $s \rightarrow t$ path. How can we identify such an opportunity?

Residual Graph

The residual graph $G_f$ identifies ways to increase flow on edges with leftover capacity, or decrease flow on edges already carrying flow:

For each original edge $e = (u, v)$ in $G$, it has:

- A forward edge $e = (u, v)$ with residual capacity $c(e) - f(e)$
- A reverse edge $e' = (v, u)$ with residual capacity $f(e)$

Edges with zero residual capacity are omitted

Exercise: Residual Graph

Let $G$ and $f$ be as depicted above.

How many edges in the residual graph have capacity $> 10$?

A. 0
B. 1
C. 3
D. 4
Augment Operation

Revised Idea: use paths in the residual graph to augment flow

Augment($f, P$)

Let $b = \text{bottleneck}(P, f)$

for each edge $(u, v)$ in $P$
do

if $(u, v)$ is a forward edge then

Let $e = (u, v)$ be the original edge

$f(e) = f(e) + b$ \quad \text{\# increase flow on forward edges}

else $(u, v)$ is a backward edge

Let $e = (v, u)$ be the original edge

$f(e) = f(e) - b$ \quad \text{\# decrease flow on backward edges}

end if

end for

Clicker Question

What is the highest bottleneck capacity of an augmenting path?

A. 1
B. 4
C. 5
D. 11

Augment Operation

Revised Idea: use paths in the residual graph to augment flow

Augment($f, P$)

Let $b = \text{bottleneck}(P, f)$

for each edge $(u, v)$ in $P$
do

if $(u, v)$ is a forward edge then

Let $e = (u, v)$ be the original edge

$f(e) = f(e) + b$ \quad \text{\# increase flow on forward edges}

else $(u, v)$ is a backward edge

Let $e = (v, u)$ be the original edge

$f(e) = f(e) - b$ \quad \text{\# decrease flow on backward edges}

end if

end for

Augment Example

Augment($f, P$)

Let $b = \text{bottleneck}(P, f)$

for each edge $(u, v)$ in $P$
do

if $(u, v)$ is a forward edge then

Let $e = (u, v)$ be the original edge

$f(e) = f(e) + b$ \quad \text{\# increase flow on forward edges}

else $(u, v)$ is a backward edge

Let $e = (v, u)$ be the original edge

$f(e) = f(e) - b$ \quad \text{\# decrease flow on backward edges}

end if

end for
Augmenting Path

New Flow

Ford-Fulkerson Algorithm

Clicker Question

Ford-Fulkerson Analysis

Step 1: F-F returns a flow

Claim: If $f$ is a flow then $f' = \text{Augment}(f, P)$ is also a flow.

Proof idea. Verify two conditions for $f'$ to be a flow: capacity and flow conservation.
Prove: Capacity

- Suppose original edge is $e = (u, v)$
- If $e$ appears in $P$ as a forward edge ($u \xrightarrow{b} v$), then flow increases by bottleneck capacity $b$, which is at most $c(e) - f(e)$, so does not exceed $c(e)$
- If $e$ appears in $P$ as a reverse edge ($v \xleftarrow{b} u$), then flow decreases by bottleneck capacity $b$, which is at most $f(e)$

Step 2: Termination and Running Time

Assumption: All capacities are integers. By nature of F-F, all flow values and residual capacities remain integers during the algorithm.

Running time:
- In each F-F iteration, flow increases by at least 1. Therefore, number of iterations is at most $v(f^*)$, where $f^*$ is the final flow.
- Let $C_s$ be the total capacity of edges leaving source $s$.
- Then $v(f^*) \leq C_s$.
- So F-F terminates in at most $C_s$ iterations

Running time per iteration? Cost of finding an augmenting path 
How to find one? Any graph search: $O(m + n)$

Prove: Flow Conservation

- Consider any node $v$ in the augmenting path, and do a case analysis on the types of the incoming and outgoing edge:

original graph: $P = s \xrightarrow{b} u \xrightarrow{b} v \xleftarrow{b} w \xrightarrow{b} t$
residual graph: $P = s \xrightarrow{b} u \xrightarrow{b} v \xleftarrow{b} w$

- In all cases, the change in incoming flow to $v$ is equal to the change in outgoing flow from $v$.

Step 3: F-F returns a maximum flow

We will prove this by establishing a deep connection between flows and cuts in graphs: the max-flow min-cut theorem.
- An $s$-$t$ cut $(A, B)$ is a partition of the nodes into sets $A$ and $B$ where $s \in A$, $t \in B$
- Capacity of cut $(A, B)$ equals
  \[ c(A, B) = \sum_{e \text{ from } A \text{ to } B} c(e) \]
- Flow across a cut $(A, B)$ equals
  \[ f(A, B) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e) \]