Flow Networks

- Used to model flow of
  - Liquids/gases though pipes
  - Parts though assembly lines
  - Current through electrical networks
  - Information through communication networks,
  - etc.
Flow Networks & Maximum Flow Problem:
- Edges: represent “conduits” with a stated capacity (weight) capacity: maximum rate at which the “material” (water, gas, electricity, bits, etc.) can flow through the conduit (e.g., 200 gallons/hour)
- Vertices: represent conduit junctions, material flows through vertices without collecting in them (except for the sink and source vertices)

Maximum Flow Problem: finding the maximum rate at which material can be shipped from the source to sink without violating any capacity constraints
Illustrative Example

- A company has a factory in a city \( (s) \) and a warehouse in another city \( (t) \), the product is manufactured in \( s \) and stocked in \( t \).
- The company leases space on trucks from another firm to ship the product from the factory to the warehouse.
  - Trucks travel over specified routes between cities and have limited capacity.
  - The truck company can ship at most \( c(u,v) \) product units per day between each pair of cities \( u \) and \( v \).
- Determine the largest rate \( p \) of product units that can be shipped.

Illustrative Example

- Every day, \( p \) product units leave the factory and \( p \) product units arrive at the warehouse.
  - The number of days needed to move \( p \) products from \( s \) to \( t \) is not relevant.
- Steady state: the rate at which units enter an intermediate city in the shipping network must equal the rate at which they leave (flow conservation).
- A Maximum flow in the network determines the maximum number \( p \) of product units per day that can be shipped.
Flow Networks

- A flow network $G = (V, E)$ is a directed graph in which each edge $(u, v) \in E$ has a nonnegative capacity $c(u, v) \geq 0$
  - If $(u, v) \notin E$, we assume $c(u, v) = 0$
- A flow network has two special vertices: the source, denoted $s$, and the sink, denoted $t$.
  - Every vertex lies in some path from the source to sink
  - The graph is connected and $|E| \geq |V| - 1$

Constraints

- Let $G = (V, E)$ be a flow network with an implied capacity function $c$. Let $s$ and $t$ be the source and sink, respectively. A flow in $G$ is a real valued function $f$: $V \times V \rightarrow \mathbb{R}$ that satisfies the following properties:
  - **Capacity constraints**: For all $u, v \in V$, we require $f(u, v) \leq c(u, v)$
  - **Skew symmetry**: For all $u, v \in V$, we require $f(u, v) = -f(v, u)$
  - The net flow from a vertex $u$ to a vertex $v$ is the negative of the net flow in the reverse direction
  - The capacity of a vertex to itself is 0: $f(u, u) = -f(u, u) \Rightarrow f(u, u) = 0$
  - **Flow Conservation**: For all $u \in V - \{s, t\}$, we require
    $$\sum_{v \in V} f(u, v) = 0; \text{ the total net flow out of a vertex is 0}$$
Networks with Multiple Sources and Sinks

- Can be converted to a flow network with 1 source and 1 sink

Net Flow

- net flow/capacity
  - $f(u,v)$: net flow from vertex $u$ to vertex $v$ can be positive or negative
- **Skew symmetry**: For all $u,v \in V$, we require $f(u,v) = -f(v,u)$. By skew symmetry, flow conservation can be re-stated as:
  - the total net flow into a vertex (other than source or sink) is 0

<table>
<thead>
<tr>
<th>Vertex</th>
<th>Net Flow Out</th>
<th>Net Flow In</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_2$</td>
<td>$\sum_{v \in V} f(v_2, v) = 15 - 7 - 4 - 12 = 0$</td>
<td>$\sum_{v \in V} f(v, v_2) = -15 + 7 + 4 + 12 = 0$</td>
</tr>
</tbody>
</table>
Net Flow
- net flow/capacity
  - The value of a flow $f$ is defined as the total net flow out of the source.
  \[ |f| = \sum_{v \in V} f(s, v) \]

Maximum Flow Problem
- Given a flow network $G$ with source $s$ and sink $t$, find a flow of maximum value from $s$ to $t$.

\[ \text{maximize } |f| = \sum_{v \in V} f(s, v) \]
### Observation

- There can be no net flow between $u$ and $v$ if there is no edge between them
  - If $(u, v) \notin E$ and $(v, u) \notin E$ then $c(u, v) = c(v, u) = 0$

- By capacity constraint:
  - $f(u, v) \leq c(u, v), f(u, v) \leq 0$
  - $f(v, u) \leq c(v, u), f(v, u) \leq 0$

- By skew symmetry:
  - $f(u, v) = -f(v, u) \Rightarrow f(u, v) = -f(v, u) = 0$

- Thus, **nonzero** net flow from vertex $v$ to vertex $u$ implies that $(u, v) \in E$ or $(v, u) \in E$ (or both)

---

### Positive Net Flow

- The positive net flow *entering* a vertex $v$ is defined by
  $$\sum_{u \in V \atop f(u, v) > 0} f(u, v)$$

- The positive net flow *leaving* a vertex is defined symmetrically.

- Interpretation of flow conservation property:
  - the positive net flow entering a vertex other than the source or sink must equal the positive net flow leaving the vertex
**Positive Net Flows**

- net flow/capacity

Convention for representing net flows: show only positive net flows, since they represent actual “shipments”: positive flows in one direction only (actual “shipments” are made in one direction only)

Cancellation

- Net flow between $v_1$ and $v_3$:
  - Flow from $v_1$ to $v_3$: $f(v_1, v_3) = 8 - 3 = 5$
  - Flow from $v_3$ to $v_1$: $f(v_3, v_1) = 3 - 8 = -5$
- Net flow is from $v_1$ to $v_3$: 5 actual “shipments” from $v_1$ to $v_3$
- By canceling flow in opposite directions, we can represent positive flows in one direction only (actual “shipments” are made in one direction only)

**Cancellation**

- allows to represent the shipments between two cities by a **positive net flow along at most one of the edges** between the corresponding vertices
  - if there is a zero or negative net flow from one vertex to another, no shipments need to be made in that direction
Identities Involving Flows

- Let $G=(V, E)$ be a flow network, and let $f$ be a flow in $G$.

- For $X \subseteq V$, $f(X, X) = 0$. (flow of a vertex to itself is 0)

- For $X, Y \subseteq V$, $f(X, Y) = -f(Y, X)$. (skew symmetry constraint)

- For $X, Y, Z \subseteq V$ with $X \cap Y = \emptyset$
  
  $f(X \cup Y, Z) = f(X, Z) + f(Y, Z)$
  
  and
  
  $f(Z, X \cup Y) = f(Z, X) + f(Z, Y)$
Flow Value

- \( |f| = f(s, V) \)
- \( = f(V, V) - f(V-s, V) \)
- \( = f(V, V-s) \)
- \( = f(V, t) + f(V, V-s-t) \)
- thus, \( |f| = f(s, V) = f(V, t) \)

by flow conservation:

\[ f(u, V) = 0 \text{ for all } u \in V \setminus \{s, t\} \]

Solving the Maximum Flow Problem

- The Ford-Fulkerson iterative method
  - Starts with \( f(u, v) = 0 \) for all \( u, v \in V \)
    - gives an initial flow of value zero
  - At each iteration, increases the flow value by finding an augmenting path
    - \textit{augmenting path}: a path from the source \( s \) to the sink \( t \) along which we can push more flow
    - flow is augmented along this path
  - repeat until no augmenting path can be found
- \textit{The max-flow min-cut theorem will show that upon termination, this method yields the maximum flow}
Ford-Fulkerson Method

Ford-Fulkerson-Method\((G,s,t)\)

Initialize flow \(f\) to 0

While there exists an augmenting path \(p\)

do augment flow along \(p\)

return \(f\)

Relies on three important concepts:

- Residual Networks
- Augmenting Paths
- Cuts

Residual Networks

- Given a flow network and a flow, the residual network consists of edges that can admit more flow.

- Let \(G=(V, E)\) be a flow network, with source \(s\) and sink \(t\). Let \(f\) be a flow in \(G\), and consider a pair of vertices \(u,v \in V\)

- the amount of additional net flow we can push from \(u\) to \(v\) before exceeding the capacity \(c(u,v)\) is the residual capacity of \((u,v)\) given by \(c_f(u,v)=c(u,v) - f(u,v)\)

\[
\begin{align*}
\text{(u)} & \quad c(u,v) = 16 \\
\text{(v)} & \quad f(u,v) = 11 \quad \Rightarrow \quad c_f(u,v) = 5
\end{align*}
\]
Residual Capacity

- When the net flow $f(u,v)$ is negative, the residual capacity $c_f(u,v)$ is greater than the capacity $c(u,v)$.

\[ c(u,v) = 16 \]
\[ f(u,v) = -4 \]
\[ c_f(u,v) = c(u,v) - f(u,v) = 16 - 4 = 20 \]

- Interpretation of residual capacity on edge:
  - Net flow of 4 units from $v$ to $u$. Can be cancelled by pushing a net flow of 4 units from $u$ to $v$.
  - Another 16 units can be pushed from $u$ to $v$ before violating the capacity constraint on edge $(u,v)$.

Residual Networks

- Given a flow network $G = (V, E)$ and a flow $f$, the residual network of $G$ induced by $f$ is $G_f = (V, E_f)$, where
  \[ E_f = \{(u,v) \in V \times V : c_f(u,v) > 0\} \]

- Each residual edge can admit a strictly positive net flow.
Residual Network: Example

Flow network with flow $f$

Residual edges between vertices $s$ and $v_1$

$(u,v)$ may be a residual edge in $E_f$ even if it was not an edge in $E$
(capacity is zero in this case)
Relating a Flow in $G$ with a Flow in $G_f$

- Let $G=(V, E)$ be a flow network, with source $s$ and sink $t$, and let $f$ be a flow in $G$. Let $G_f$ be the residual network of $G$ induced by $f$, and let $f'$ be a flow in $G_f$. Then, the flow sum $f + f'$ is a flow in $G$ with value $|f + f'| = |f| + |f'|$.

- We must verify that skew symmetry, capacity constraints and flow conservation are obeyed by $f + f'$ in $G$.

- **Skew symmetry**
  
  for all $u, v \in V$
  
  \[
  (f + f')(u, v) = f(u, v) + f'(u, v) \\
  = - f(v, u) - f'(v, u) \\
  = - (f(v, u) + f'(v, u)) = -(f + f')(v, u)
  \]

- **Capacity constraints**
  
  - By definition, $f'(u, v) \leq c_f(u, v)$ for all $u, v \in V$ i.e., net flow in residual edge $\leq$ capacity of residual edge
  
  - \[ (f + f')(u, v) = f(u, v) + f'(u, v) \]
  
  \[
  \leq f(u, v) + c_f(u, v) \\
  = f(u, v) + (c(u, v) - f(u, v)) \\
  = c(u, v)
  \]
  
  i.e., resulting total flow $\leq$ capacity

  **Definition of $c_f(u, v)$**
Relating a Flow in $G$ with a Flow in $G_f$

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

  \[
  |f + f'| = \sum_{v \in V} (f + f')(s, v)
  = \sum_{v \in V} (f(s, v) + f'(s, v))
  = \sum_{v \in V} f(s, v) + \sum_{v \in V} f'(s, v)
  = |f| + |f'|
  \]

Finally, we need to show that the flow sum $f + f'$ is a flow in $G$ with value $|f + f'| = |f| + |f'|$

\[
|f + f'| = \sum_{v \in V} (f + f')(s, v)
= \sum_{v \in V} f(s, v) + \sum_{v \in V} f'(s, v)
= \sum_{v \in V} f(s, v) + \sum_{v \in V} f'(s, v)
= |f| + |f'|
\]
Augmenting Paths

- Given a flow network $G = (V, E)$ and a flow $f$, an augmenting path $p$ is a simple path from $s$ to $t$ in the residual network $G_f$.
- By the definition of the residual network, each edge $(u,v)$ on an augmenting path admits some additional positive net flow from $u$ to $v$ without violating the capacity constraint on the edge.

Residual Capacity of Augmenting Path

- The residual capacity of an augmenting path $p$ is the maximum amount of net flow that we can ship along the edges of $p$.

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

Residual capacity of augmenting path $p$ is $C_f(v,v2) = 4$. 
Ford-Fulkerson Method

- **Ford-Fulkerson-Method** \((G,s,t)\)
  - Initialize flow \(f\) to 0
  - while there exists an augmenting path \(p\)
  - do augment flow along \(p\)
  - return \(f\)

- Repeatedly augments the flow along augmenting paths until a maximum flow has been found
- The max-flow min-cut theorem tells us that a flow is maximum if and only if its residual network contains no augmenting paths proof needs notion of cut (of a flow network)

Cuts of Flow Networks

- A cut \((S,T)\) of flow network \(G = (V,E)\) is a partition of \(V\) into \(S\) and \(T=V\setminus S\) such that \(s \in S\) and \(t \in T\).
- If \(f\) is a flow, then the net flow across the cut \((S,T)\) is defined to be \(f(S,T)\). The capacity of the cut \((S,T)\) is \(c(S,T)\).

Net flow across cut:

\[
f(S,T) = f(v_1,v_2) + f(v_3,v_2) + f(v_3,v_4)
\]

\[
= 12 + (-4) + 11 = 19
\]

Capacity of the cut:

\[
c(S,T) = c(v_1,v_2) + c(v_3,v_2) + c(v_3,v_4)
\]

\[
= 12 + 0 + 14 = 26
\]
Net Flow Across a Cut

- The value of a flow in a network \( G = (V, E) \), \(|f|\), is the net flow across any cut \((S, T)\) of the network, denoted \(f(S, T)\).

- \( f(S, T) = f(S, V) - f(S, S) \)
  
  - \( f(S, V) = f(S, V) + f(S-s, V) \)
  
  - \( f(s, V) \)
  
  - \( |f| \)

\[ f(u, V) = 0 \text{ for all } u \in V-\{s, t\} \text{ -- flow conservation} \]

Cut Capacities Bound the Value of a Flow

- The value of a flow \( f \) in a network \( G = (V, E) \) is upper bounded by the capacity of any cut of \( G \).

\[
|f| = \sum_{u \in S} \sum_{v \in T} f(u, v) 
\leq \sum_{u \in S} \sum_{v \in T} c(u, v) 
= c(S, T) 
\]
Max-Flow Min-Cut Theorem

- If \( f \) is a flow in a flow network \( G = (V, E) \) with source \( s \) and sink \( t \), then the following conditions are equivalent:

  1. \( f \) is a maximum flow in \( G \)
  2. The residual network \( G_f \) contains no augmenting path
  3. \( |f| = c(S, T) \) for some cut \((S, T)\) of \( G \). (There is a cut for which the net flow is equal to the capacity of the cut)

Max-Flow Min-Cut Theorem

- (1) \( \Rightarrow \) (2) (by contradiction): Suppose that \( f \) is a maximum flow in \( G \) but that \( G_f \) has an augmenting path \( p \). Then the flow sum \( f + f_p \) is a flow in \( G \) with value strictly greater than \( |f| \), contradicting the assumption.

- (2) \( \Rightarrow \) (3) Suppose that \( G_f \) has no augmenting path (i.e., no path from \( s \) to \( t \)). Define a cut such that all vertices is \( S \) have a path from \( s \) in \( G_f \), and \( T = V - S \). (Note that this is indeed a cut, since \( t \) is not in \( S \), or we would have an augmenting path.) If \( u \in S \) and \( v \in T \), we have \( f(u, v) = c(u, v) \) (or else \( (u, v) \in E_f \) and \( v \) is in set \( S \)). Therefore \( |f| = f(S, T) = c(S, T) \)

- (3) \( \Rightarrow \) (1): \( |f| \leq c(S, T) \) for all cuts \((S, T)\).
  - Thus \( |f| = c(S, T) \Rightarrow f \) is a maximum flow.
Example

\[ |\mathcal{F}| = 0 \]

\[ |\mathcal{F}| = 12 \]

\[ c(\mathcal{F}) = 12 \]

Example

\[ |\mathcal{F}| = 16 \]

\[ c(\mathcal{F}) = 4 \]
Example

\[ |f| = 16 \]

\[ |f| = 23 \]

\[ \text{max flow was found: } |f| = 23 \]

\[ \text{Cut Capacity: } 12 + 7 + 4 = 23 \]
Ford-Fulkerson Algorithm: Time Complexity

- The running time of the Ford-Fulkerson algorithm depends on how the augmenting path $p$ is determined.
- If the augmenting path is chosen using a breadth-first search (Edmonds-Karp), it can be shown that the number of iterations performed by the algorithm is at most $O(VE)$.
- Each iteration takes $O(E)$ time.
- Total running time: $O(VE^2)$