Transshipment

The Transshipment Problem: Given m suppliers and n customers,
Is it possible for the customers (suppliers) to have their orders filled?

Transshipment

The Transshipment Problem: Given m suppliers and n customers, Is it possible for the customers (suppliers) to have their orders filled?

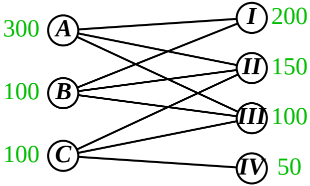
- Each supplier has some amount of product.
- Each customer desires some amount of product.
- Not all suppliers deliver to each customer.

Transshipment

The Transshipment Problem: Given m suppliers and n customers, Is it possible for the customers (suppliers) to have their orders filled?

- Each supplier has some amount of product.
- Each customer desires some amount of product.
- Not all suppliers deliver to each customer.

Example. Suppliers A, B, C have 300, 100, 100 units of product. Customers I, II, III, IV, desire 200, 150, 100, 50 units of product. Neither A nor B delivers to IV, and C does not deliver to I.

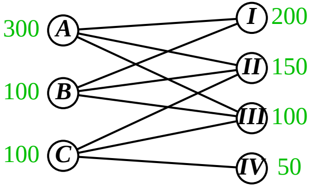


Transshipment

The Transshipment Problem: Given m suppliers and n customers, Is it possible for the customers (suppliers) to have their orders filled?

- Each supplier has some amount of product.
- Each customer desires some amount of product.
- Not all suppliers deliver to each customer.

Example. Suppliers A, B, C have 300, 100, 100 units of product. Customers I, II, III, IV, desire 200, 150, 100, 50 units of product. Neither A nor B delivers to IV, and C does not deliver to I.



Q: Is there a transshipment that satisfies all the suppliers?

Transshipment

Key: Convert the transshipment problem to a network flow problem.

Transshipment

Key: Convert the transshipment problem to a network flow problem.

- Start with G, with edges **from** suppliers x **to** customers y.

Transshipment

Key: Convert the transshipment problem to a network flow problem.

- Start with G, with edges **from** suppliers x **to** customers y.
- Create a network $\widehat{G}$ by adding two vertices:

Transshipment

Key: Convert the transshipment problem to a network flow problem.

- Start with G, with edges **from** suppliers x **to** customers y.
- Create a network $\widehat{G}$ by adding two vertices:
  - A "super-source" s that is **adjacent to** every supplier x.
  - A "super-sink" t that is **adjacent from** every customer y.

Transshipment

Key: Convert the transshipment problem to a network flow problem.

- Start with G, with edges **from** suppliers x **to** customers y.
- Create a network $\widehat{G}$ by adding two vertices:
  - A "super-source" s that is **adjacent to** every supplier x.
  - A "super-sink" t that is **adjacent from** every customer y.
- Assign capacities to the edges as follows:

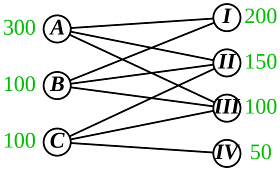
$$\begin{cases} \text{if } e : s \to x, \text{ set} & c_e = \text{supplier } x\text{'s supply} \\ \text{if } e : x \to y, \text{ set} & c_e = \infty \\ \text{if } e : y \to t, \text{ set} & c_e = \text{customer } y\text{'s demand} \end{cases}$$

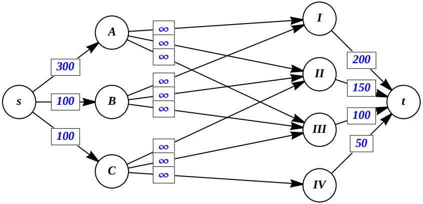
Transshipment

Key: Convert the transshipment problem to a network flow problem.

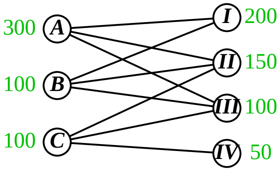
- Start with G, with edges **from** suppliers x **to** customers y.
- Create a network $\widehat{G}$ by adding two vertices:
  - A "super-source" s that is **adjacent to** every supplier x.
  - A "super-sink" t that is **adjacent from** every customer y.
- Assign capacities to the edges as follows:

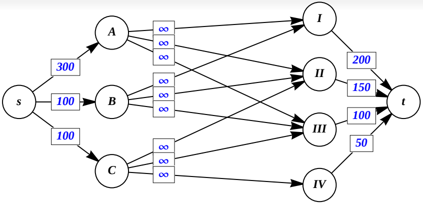
$$\begin{cases} \text{if } e: s \to x, \text{ set} & c_e = \text{supplier } x\text{'s supply} \\ \text{if } e: x \to y, \text{ set} & c_e = \infty \\ \text{if } e: y \to t, \text{ set} & c_e = \text{customer } y\text{'s demand} \end{cases}$$





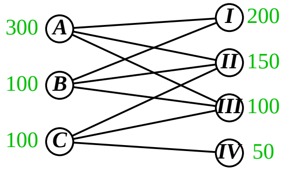
Transshipment

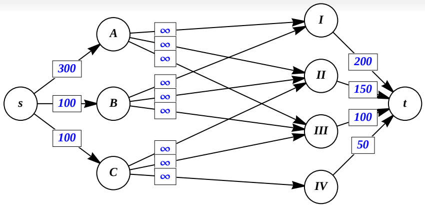




Important: a transshipment in $G \iff$ a flow in $\widehat{G}$.

Transshipment

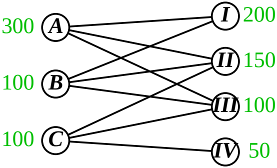


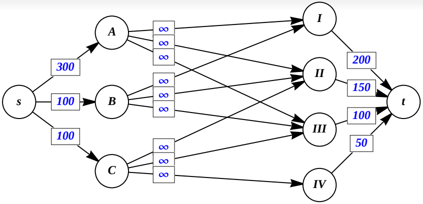


Important: a transshipment in $G \iff$ a flow in $\widehat{G}$.

$\therefore$ a maximum transshipment in $G \iff$ a maximum flow in $\widehat{G}$.

Transshipment



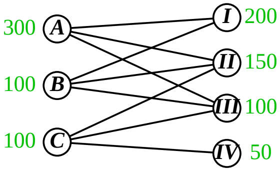


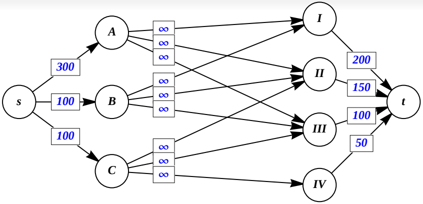
Important: a transshipment in $G \iff$ a flow in $\widehat{G}$.

$\therefore$ a maximum transshipment in $G \iff$ a maximum flow in $\widehat{G}$.

Run the Ford-Fulkerson algorithm. **Interpret the min cut.**

Transshipment





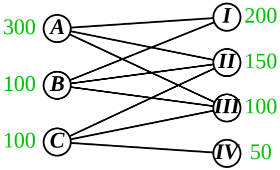
Important: a transshipment in $G \iff$ a flow in $\widehat{G}$.

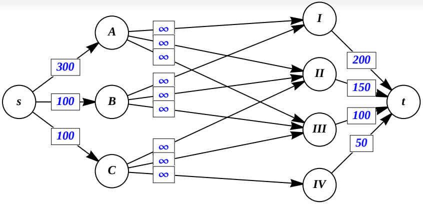
$\therefore$ a maximum transshipment in $G \iff$ a maximum flow in $\widehat{G}$.

Run the Ford-Fulkerson algorithm. **Interpret the min cut.**

- When all suppliers are satisfied in G, the min cut in $\widehat{G}$ is ______.

Transshipment





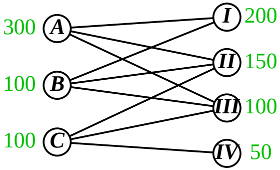
Important: a transshipment in $G \iff$ a flow in $\widehat{G}$.

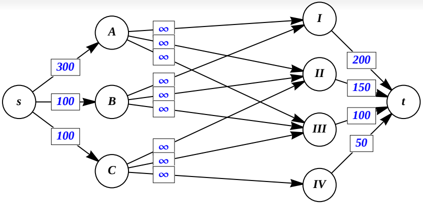
$\therefore$ a maximum transshipment in $G \iff$ a maximum flow in $\widehat{G}$.

Run the Ford-Fulkerson algorithm. **Interpret the min cut.**

- When all suppliers are satisfied in G, the min cut in $\widehat{G}$ is ______.
- Otherwise, the min cut tells the problem: there exists a set of suppliers whose customers demand less than the suppliers supply.

Transshipment





Important: a transshipment in $G \iff$ a flow in $\widehat{G}$.

$\therefore$ a maximum transshipment in $G \iff$ a maximum flow in $\widehat{G}$.

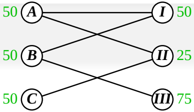
Run the Ford-Fulkerson algorithm. **Interpret the min cut.**

- When all suppliers are satisfied in G, the min cut in $\widehat{G}$ is ______.
- Otherwise, the min cut tells the problem: there exists a set of suppliers whose customers demand less than the suppliers supply.

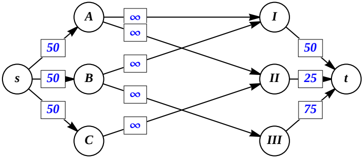
If you are customer-centric, orient the edges from right to left.

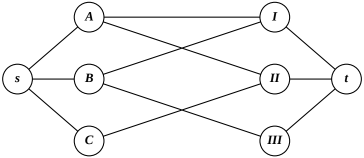
Gives a set of customers who can not be satisfied by their suppliers.

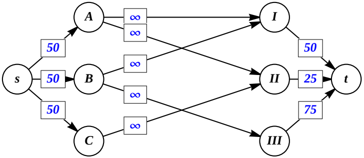
Transshipment Example

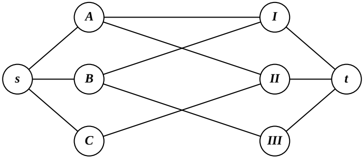


Supplier-centric:



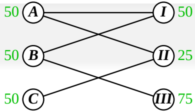




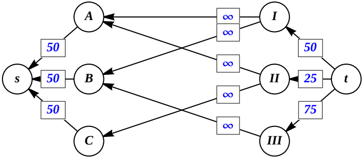


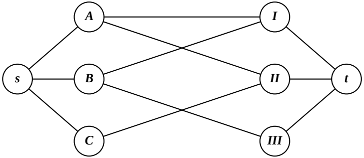
Problem:

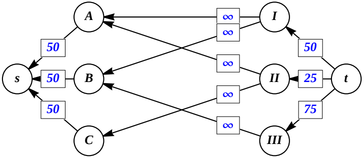
Transshipment Example

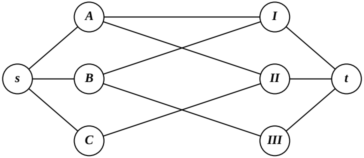


Customer-centric:









Problem:

Dynamic Networks

- Ford–Fulkerson gives the max throughput of a static network.
- Use dynamic networks to model the act of sending shipments.

Dynamic Networks

- Ford–Fulkerson gives the max throughput of a static network.
- Use dynamic networks to model the act of sending shipments.

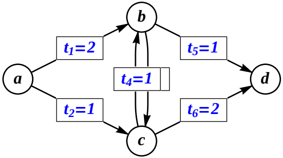
Definition: In a **dynamic network**, every edge e has both a capacity c_e and a travel time t_e.

Dynamic Networks

- Ford–Fulkerson gives the max throughput of a static network.
- Use dynamic networks to model the act of sending shipments.

Definition: In a **dynamic network**, every edge e has both a capacity c_e and a travel time t_e.

Example. Consider four cities with warehouses (a, b, c, and d) such that one truck per day can leave along any route, and the travel time for each route is given by:

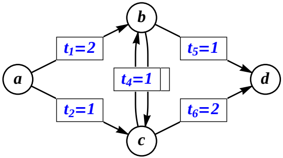


Dynamic Networks

- Ford–Fulkerson gives the max throughput of a static network.
- Use dynamic networks to model the act of sending shipments.

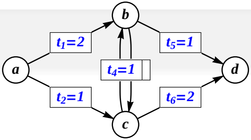
Definition: In a **dynamic network**, every edge e has both a capacity c_e and a travel time t_e.

Example. Consider four cities with warehouses (a, b, c, and d) such that one truck per day can leave along any route, and the travel time for each route is given by:

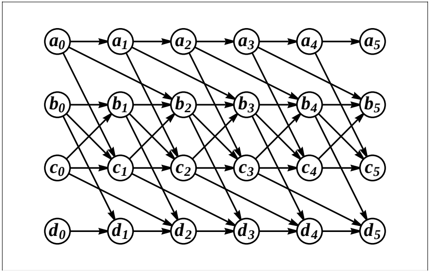


We wish to determine the maximum number of shipments which can make it from city a on day 0 and arrive at city d by day 5.

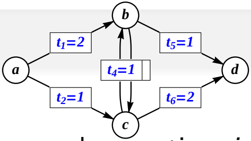
Dynamic Networks



Create a new, static network.

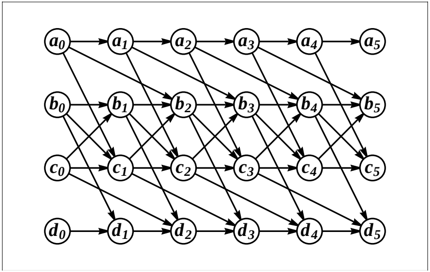


Dynamic Networks

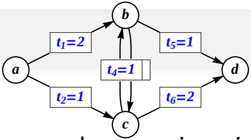


Create a new, static network.

- Create a vertex v_i for every warehouse v and every time i.

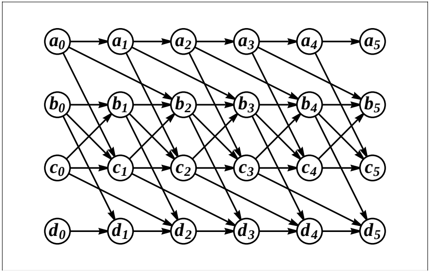


Dynamic Networks

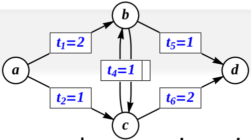


Create a new, static network.

- Create a vertex v_i for every warehouse v and every time i.
- For all original edges $e : v \to w$ with capacity c_e and time t_e, create edges from $v_i \to w_{i+t_e}$ with capacity c_e for all i.

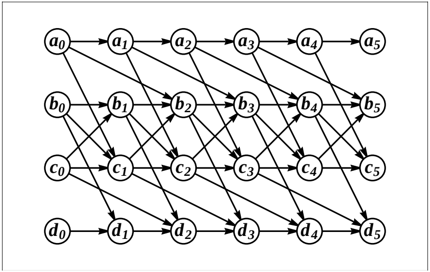


Dynamic Networks

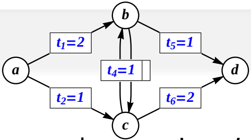


Create a new, static network.

- Create a vertex v_i for every warehouse v and every time i.
- For all original edges $e : v \longrightarrow w$ with capacity c_e and time t_e, create edges from $v_i \longrightarrow w_{i+t_e}$ with capacity c_e for all i.
- For all v and i, create an edge from v_i to v_{i+1} with ∞ capacity. This represents shipping no product.

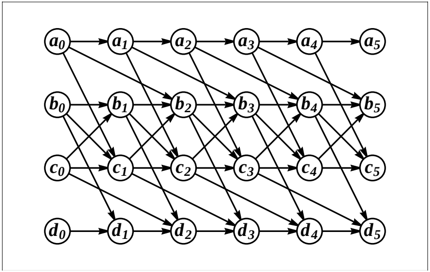


Dynamic Networks

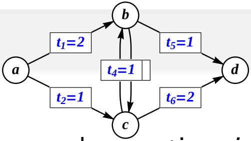


Create a new, static network.

- Create a vertex v_i for every warehouse v and every time i.
- For all original edges $e : v \longrightarrow w$ with capacity c_e and time t_e, create edges from $v_i \longrightarrow w_{i+t_e}$ with capacity c_e for all i.
- For all v and i, create an edge from v_i to v_{i+1} with ∞ capacity. This represents shipping no product.
- Find the max flow from source(s) at time 0 to sink(s) at time n.



Dynamic Networks



Create a new, static network.

- Create a vertex v_i for every warehouse v and every time i.
- For all original edges $e : v \to w$ with capacity c_e and time t_e, create edges from $v_i \to w_{i+t_e}$ with capacity c_e for all i.
- For all v and i, create an edge from v_i to v_{i+1} with ∞ capacity. This represents shipping no product.
- Find the max flow from source(s) at time 0 to sink(s) at time n.

Example. In the graph below, calculate the max flow from a_0 to d_5.