Modelling and Optimization

INF170

#7:Network Optimization

AHMAD HEMMATI

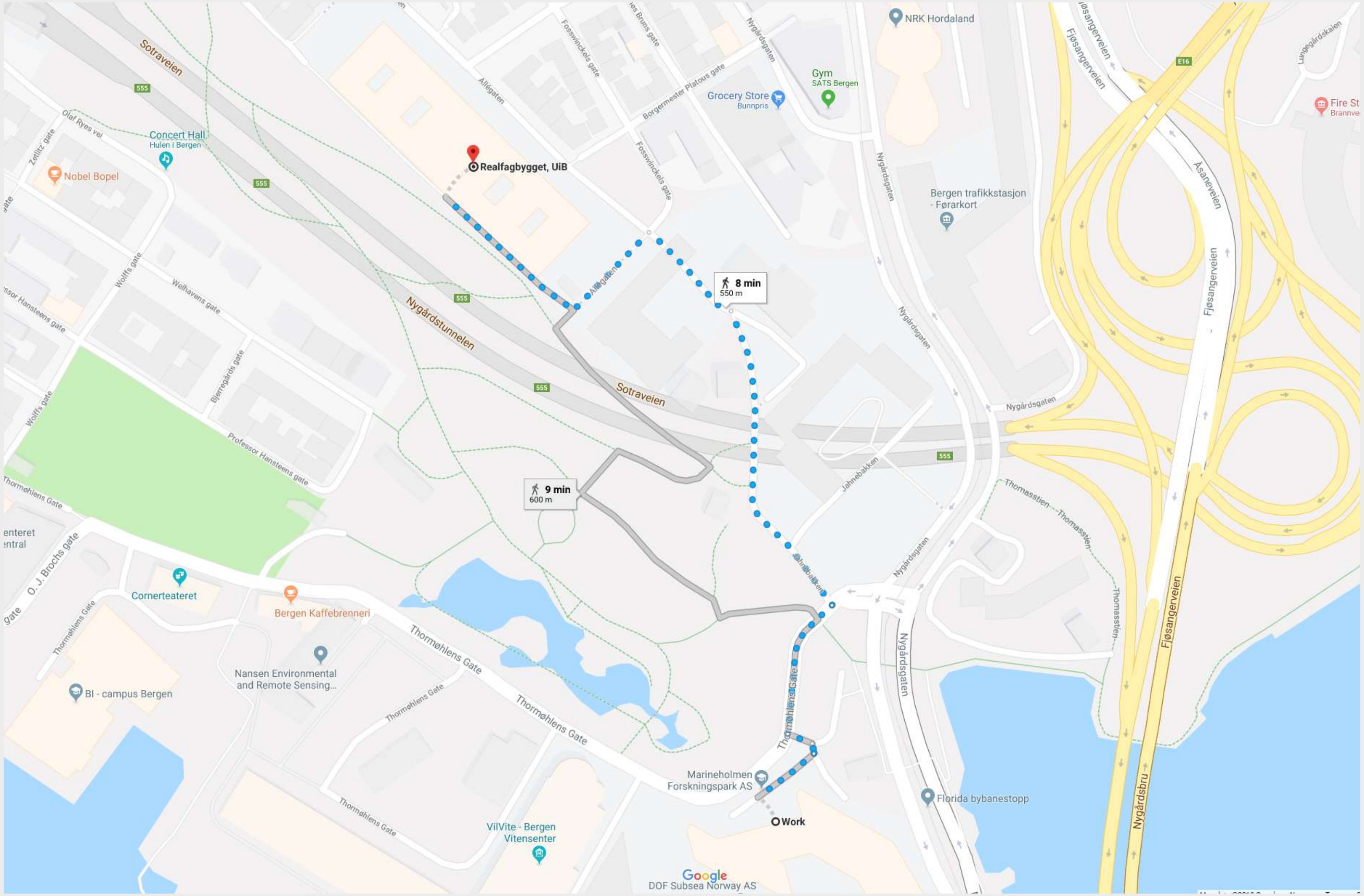
Optimization Group
Dept. of Informatics
University of Bergen

Fall Semester
2018

AGENDA

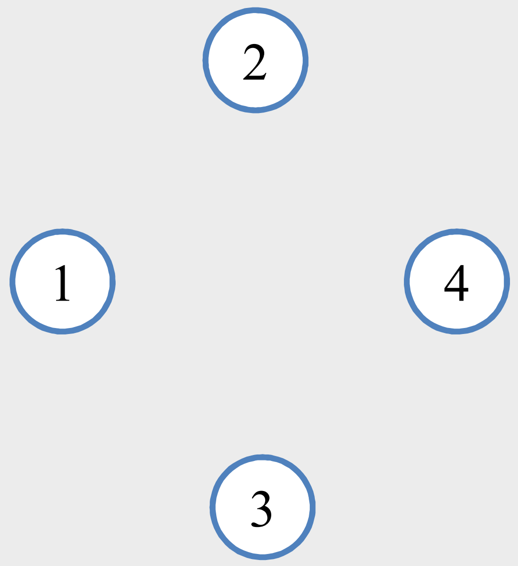
- Graphs and Networks
- Shortest Path Problem
- Critical Path Problem
- Maximum Flow Problem
- Minimum Cost Flow Problem

Graphs and Networks



- Set of nodes N

  (Also known as vertices)



$N = \{1, 2, 3, 4\}$

Graphs and Networks

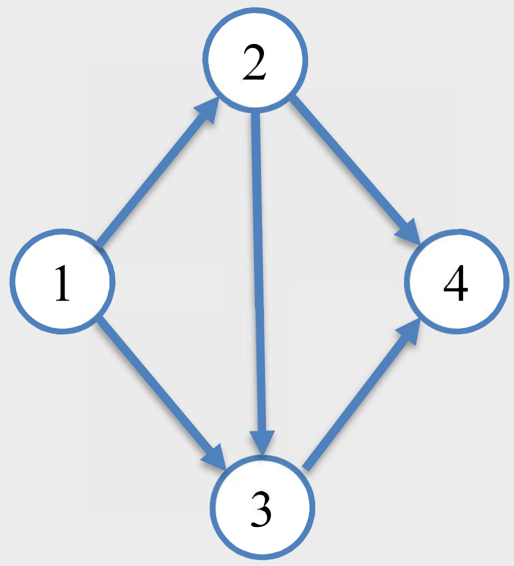
- Set of nodes N

  (Also known as vertices)

- Set of arcs (links) A

  Directed from one node to another

  Denoted $(i\ ,\ j)$ for some $i\ ,\ j \in N$



$N = \{1, 2, 3, 4\}$

$A = \{(1, 2), (1, 3), (2, 3), (2, 4), (3, 4)\}$

Graphs and Networks

- Set of nodes N

  (Also known as vertices)

- Set of arcs (links) A

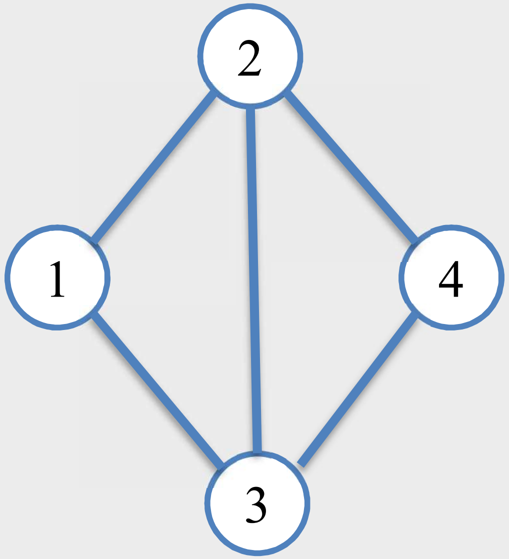
  Directed from one node to another

  Denoted (i, j) for some $i, j \in N$

- Set of edges E

  Two nodes connected

  without specified direction

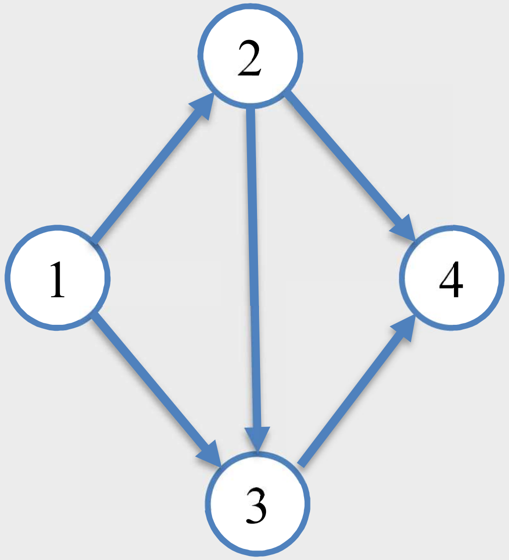


$N = \{1, 2, 3, 4\}$

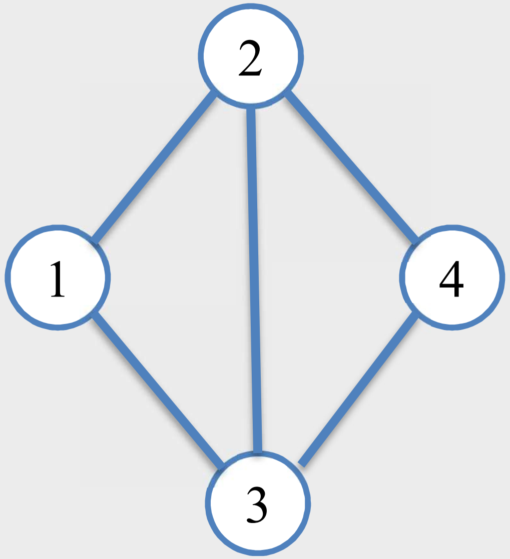
$E = \{\{1, 2\}, \{1, 3\}, \{2, 3\}, \{2, 4\}, \{3, 4\}\}$

Graphs and Networks

Directed graph or **network** (N, A)
(nodes are connected by arcs)



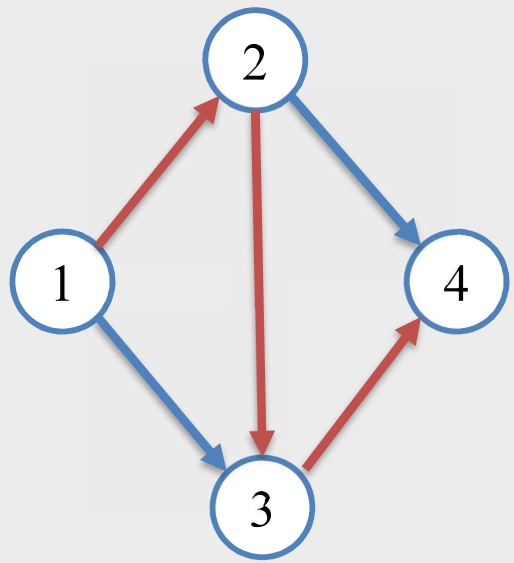
Undirected graph (N, E)
(nodes are connected by edges)



Graphs and Networks

- Physical networks
  - ✓ Road networks
  - ✓ Airline traffic networks
  - ✓ Electrical networks
- Abstract networks
  - ✓ Organizational charts
  - ✓ Social networks
  - ✓ Precedence relationships in projects

- A path is a sequence of arcs or edges connecting two specified nodes in a graph:
  - Each arc or edge must have exactly one node in common with its predecessor in the sequence
  - Any arcs must be passed in the forward direction
  - No node may be visited more than once

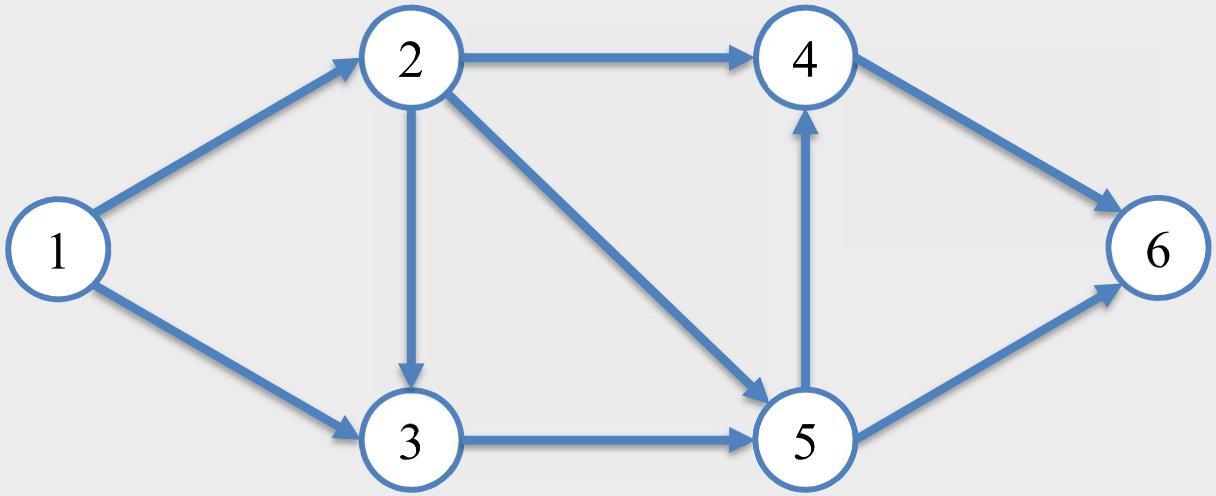


1-2-3-4 is a path from 1 to 4

Sequence of arcs: (1, 2), (2, 3), (3, 4)

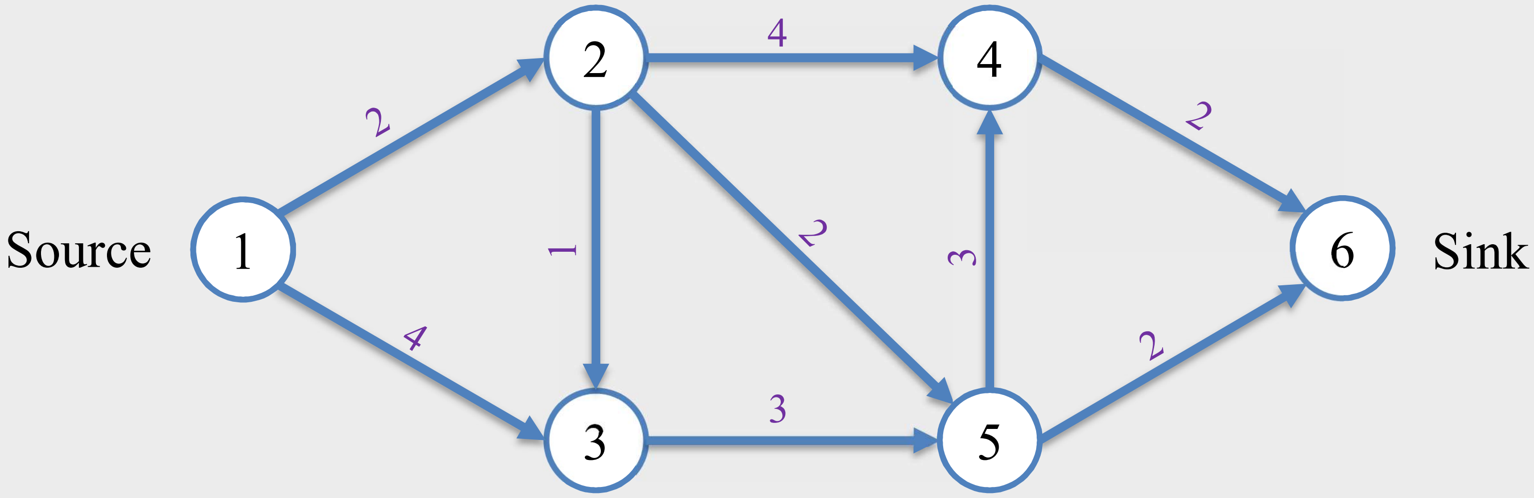
SHORTEST PATH PROBLEM

Shortest Path Problem

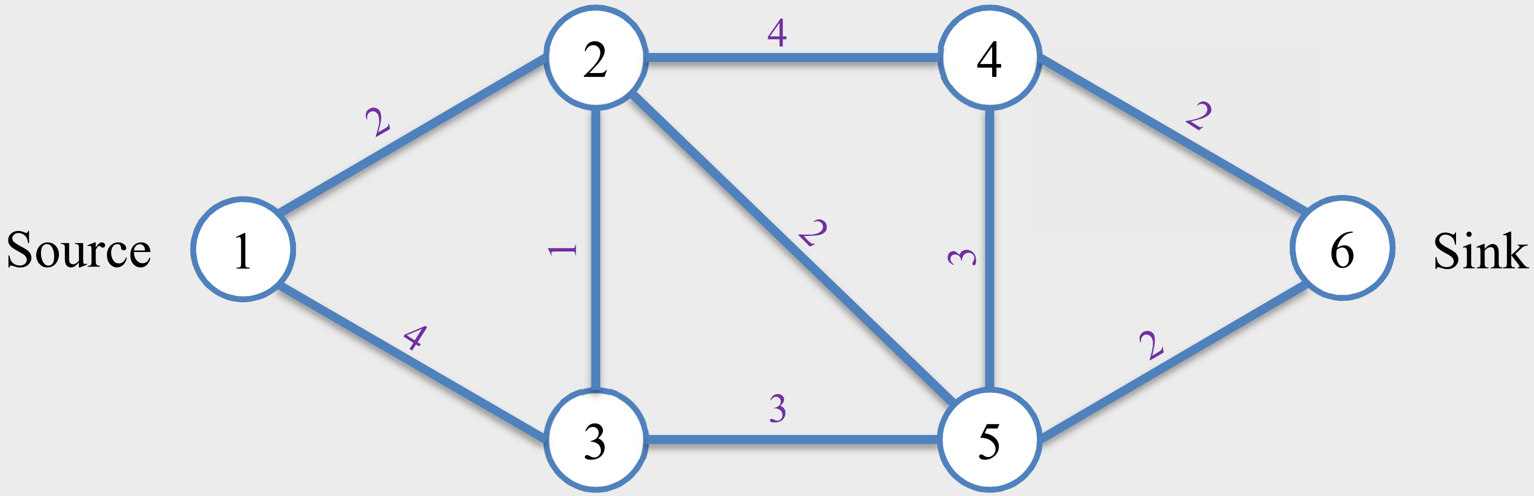


- Network (N, A)

SHORTEST PATH PROBLEM

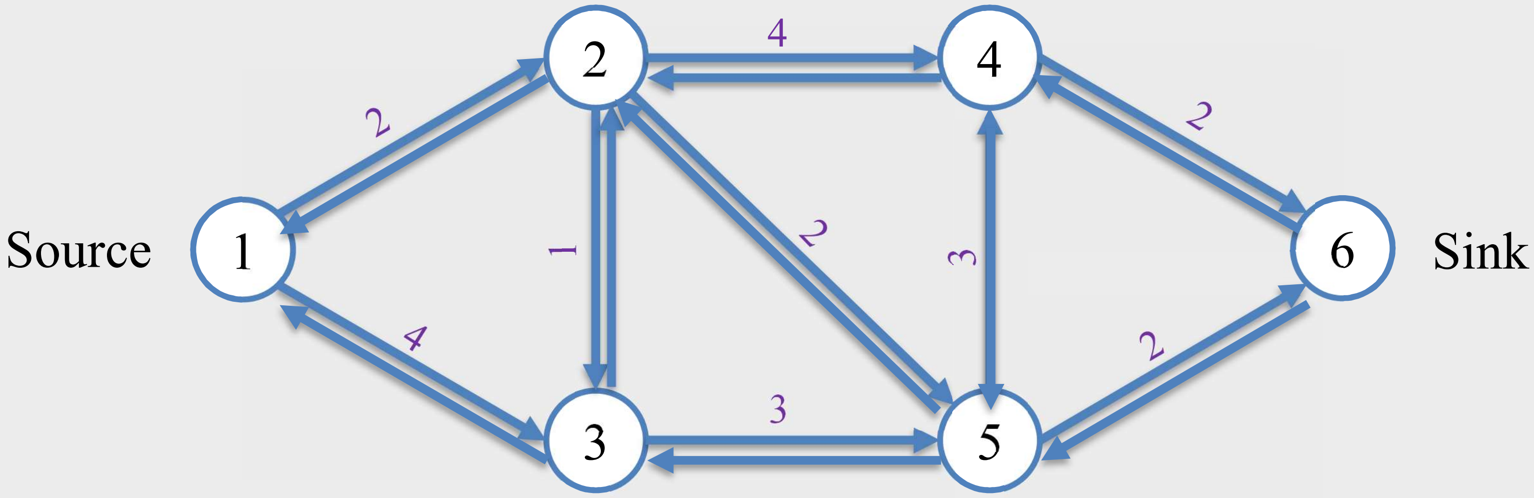


- Network (N, A)
- Each arc (i, j) in A has a length (or cost) $c_{i,j}$
- Designate
  - one node in the network as the source s
  - another node in the network as the sink t
- What is the shortest path from s to t?



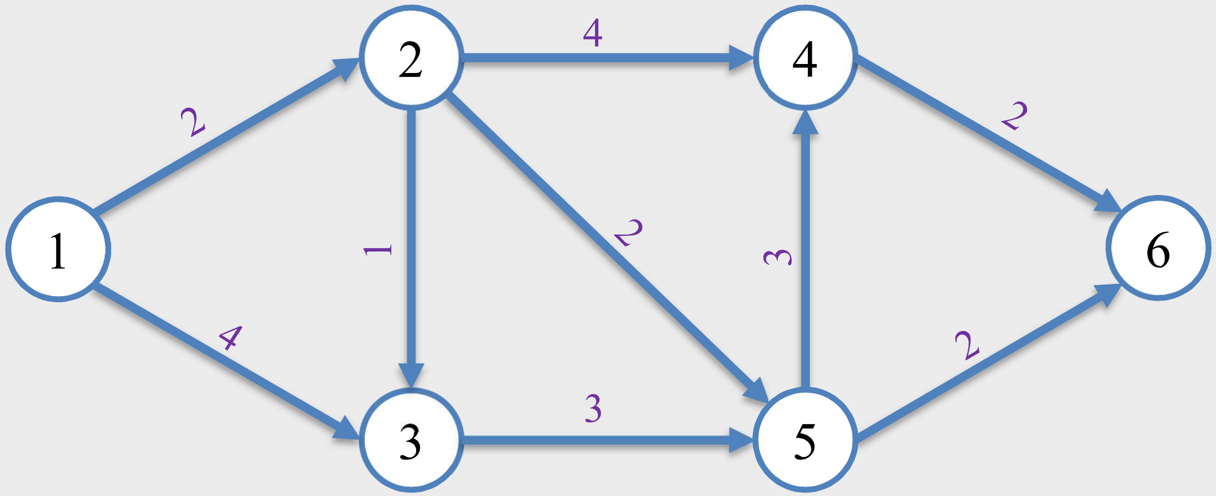
- What about shortest paths in undirected graphs?

Shortest Path Problem



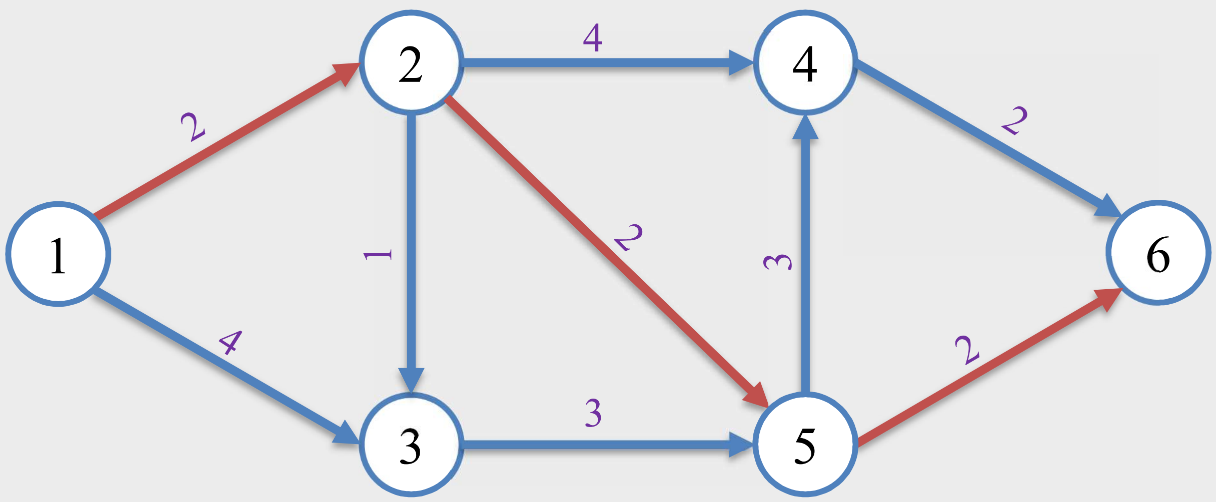
- What about shortest paths in undirected graphs?
- Convert to problem on network
  - Replace each edge with two arcs
- Note: whether the link is directed or not, a path will use the link no more than once because paths cannot repeat nodes

Shortest Path Problem



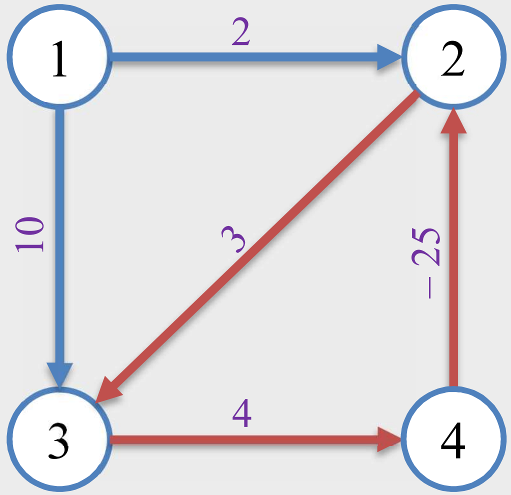
- Find the shortest path from 1 to 6
- Find the shortest path from 1 to 5

Shortest Path Problem



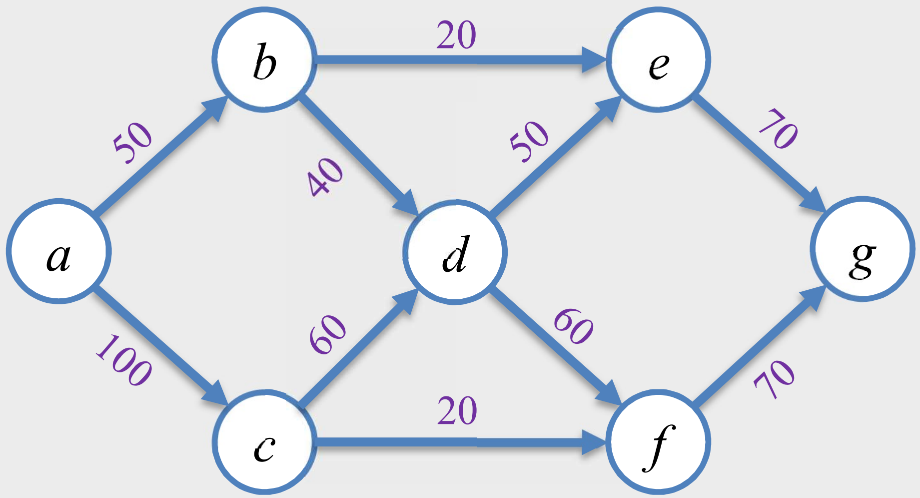
- Shortest path from 1 to 6 is 1-2-5-6 with length 6
- Consider going from 1 to 5
  - 1-2-5 is one such path, with length 4
- Could there be a shorter path?
- No! Otherwise, 1-2-5-6 is not the shortest path from 1 to 6

Shortest Path Problem



- A directed cycle (or a dicycle) in a network is a path from a source node *s* to a sink node *t* plus an arc (*t*, *s*)
- A negative dicycle has negative total length
- Principle of optimality (for the shortest path problem)
  - ✓ In a graph with no negative dicycles, optimal paths must have optimal subpaths

Shortest Path Problem



SHORTEST PATH PROBLEM

$$minimize \sum_{(i,j)\in A} d_{i,j} x_{i,j}$$

subject to

$$\sum_{j:(i,j)\in A} x_{i,j} - \sum_{k:(k,i)\in A} x_{k,i} = \begin{cases} 1 & i = s \\ 0 & i \neq s,t \\ -1 & i = t \end{cases}$$

$x_{i,j} \geq 0$ for all $(i, j) \in A$

Shortest Path Problem

```
set   INTER;                                # intersections
param entr symbolic in INTER;               # entrance to road network
param exit symbolic in INTER,<> entr;# exit from road network
set   ROADS within (INTER diff {exit}) cross (INTER diff {entr});
param time {ROADS} >= 0;                    # times to travel roads
var   Use {(i,j) in ROADS} >= 0;            # 1 iff (i,j) in shortest path

minimize   Total_Time: sum {(i,j) in ROADS} time[i,j] * Use[i,j];
subject to Start: sum {(entr,j) in ROADS} Use[entr,j] = 1;
subject to Balance {k in INTER diff {entr,exit}}:
     sum {(i,k) in ROADS} Use[i,k] = sum {(k,j) in ROADS} Use[k,j];

data;
set    INTER := a b c d e f g ;
param  entr := a ;
param  exit := g ;
param: ROADS: time :=
       a b 50, a c 100
       b d 40, b e 20
       c d 60, c f 20
       d e 50, d f 60
       e g 70, f g 70 ;
```

CRITICAL PATHS PROBLEM

Critical Paths Problem

- Lots of stuff to get done
- Some need to be done in a certain order
- Collection of activities, say $\{1, \ldots, n\}$
- Each activity has an estimated duration, say

$$a_k = \text{time required to accomplish activity } k$$

- Activity j is a predecessor of activity k if activity j must be completed before activity k can begin

Critical Paths Problem

- For example, preparing breakfast

k	Activity	Duration (min.)	Predecessors
1	Boil water	5	None
2	Get dishware	1	None
3	Make tea	3	1 , 2
4	Pour cereal	1	2
5	Fruit on cereal	2	4
6	Milk on cereal	1	4
7	Make toast	4	None
8	Butter toast	3	7

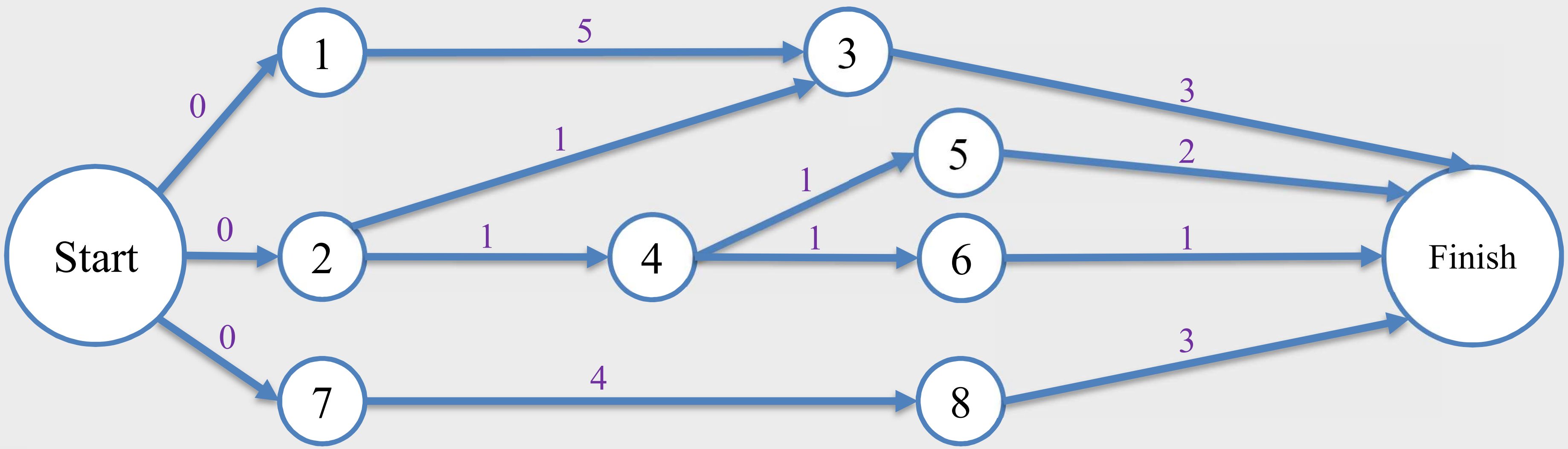
The Critical Path Method (CPM)

project networks consist of

- special start and finish nodes
- one node for each activity
- arcs with length 0 connect the start node to all activities without predecessors
- arcs of length a_k connect each activity k to all activities of which it is a predecessor, or to the finish node if there are no such activities

Critical Paths Problem

k	Activity	Duration (min.)	Predecessors
1	Boil water	5	None
2	Get dishware	1	None
3	Make tea	3	1 , 2
4	Pour cereal	1	2
5	Fruit on cereal	2	4
6	Milk on cereal	1	4
7	Make toast	4	None
8	Butter toast	3	7



Critical Paths Problem

- Earliest start time of any activity k in a project

  = length of longest path from start to node k in the corresponding project network

- Such longest paths are called critical paths

- Think of the finish node as an activity

  (e.g. celebrating finishing the project)

- What is the earliest time when you can start celebrating?

  (What is the earliest time when you can finish the project?)

  → Length of critical path from start node to finish node in the project network

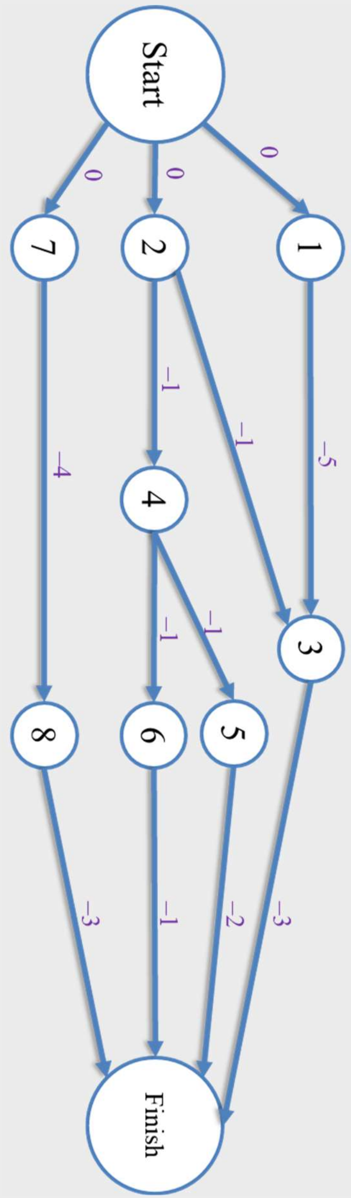
Critical Paths Problem

- What if a project network contains a cycle?
- Nonsensical!
- An acyclic network is a network that contains no directed cycles
- (Well-formed) project networks are acyclic

Critical Paths Problem

- Want to find the longest path from start to k
- Can recast this as a shortest path problem
  - In a project network, length of arc $(i, j) = a_i$
  - Form new identical network, with length of arc $(i, j) = -a_i$
  - Shortest path from start to k in new network = Longest path from start to k in old network
- Project networks are acyclic

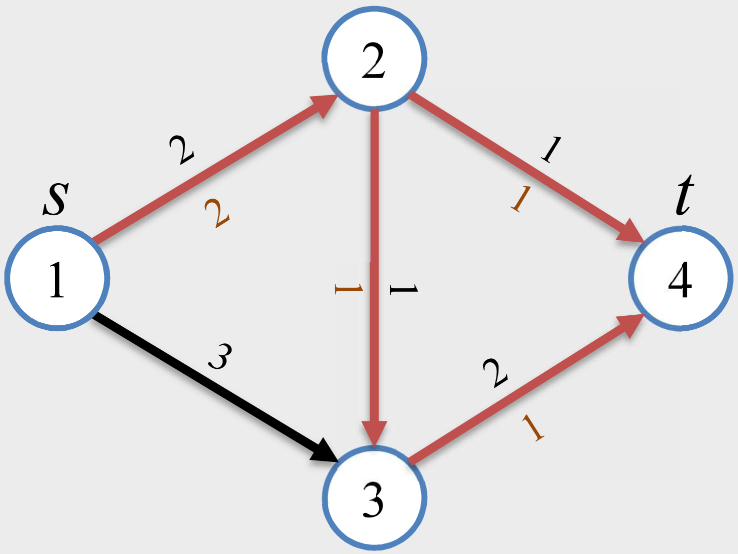
  no negative dicycles



MAXIMUM FLOW PROBLEM

Maximum Flow Problem

- Network (N, A)
- Source node $s \in N$, sink node $t \in N$
- Each arc $(i, j) \in A$ has a capacity $u_{i,j}$
- A flow is an assignment of values to each arc $(i, j) \in A$
- Feasible flow:
  - Flow on arc (i, j) is at most $u_{i,j}$
  - Flow conservation: for each node $i \in N$ $(i \neq s, t)$, total flow into node i = total flow out of node i
- Think of network as pipes

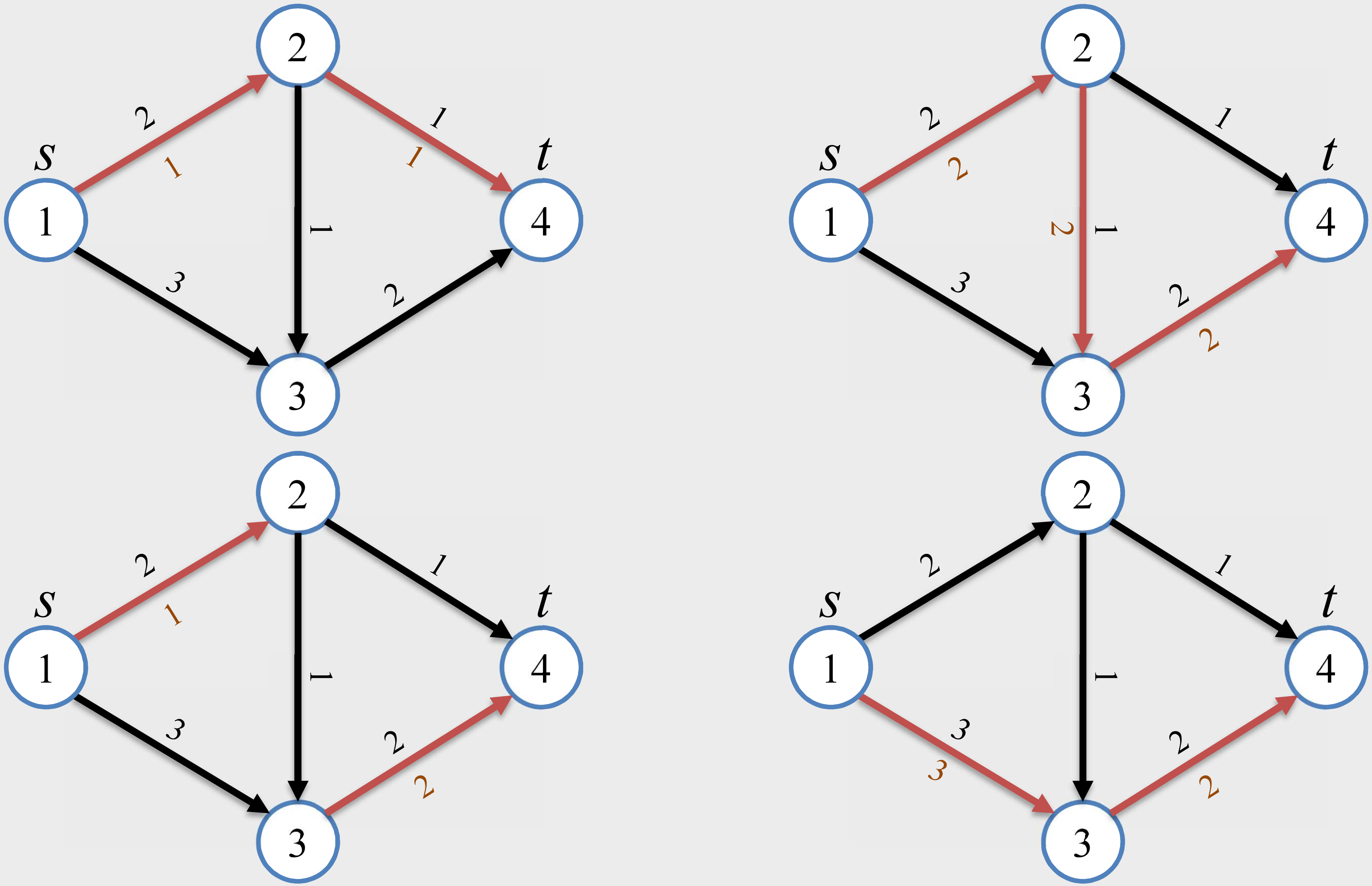


Black arc labels are capacities

Red arc labels are flows

Maximum Flow Problem

➢ Are these feasible flows? Why or why not?



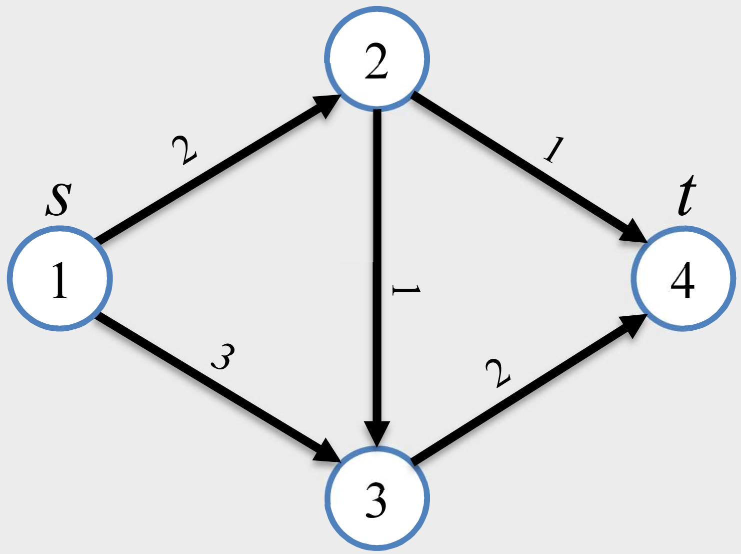
Maximum Flow Problem

- Network (N, A)
- Source node $s \in N$
- Sink node $t \in N$
- Capacity $u_{i,j}$ for each arc $(i, j) \in A$
- Determine a feasible flow that maximizes the flow out of s
- Flow conservation → (flow out of s) = (flow into t))

  → Equivalently, what is the maximum possible flow we can send from s to t?

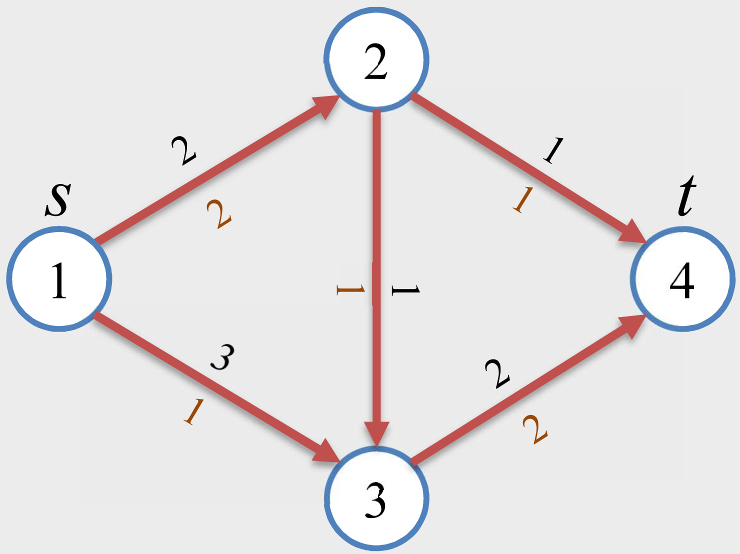
MAXIMUM FLOW PROBLEM

➢ What is the maximum possible flow from *s* to *t* in this network?

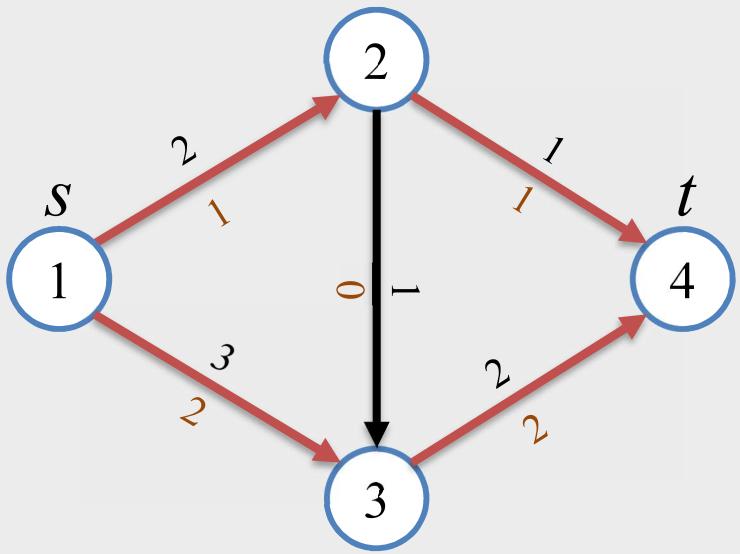


Maximum Flow Problem

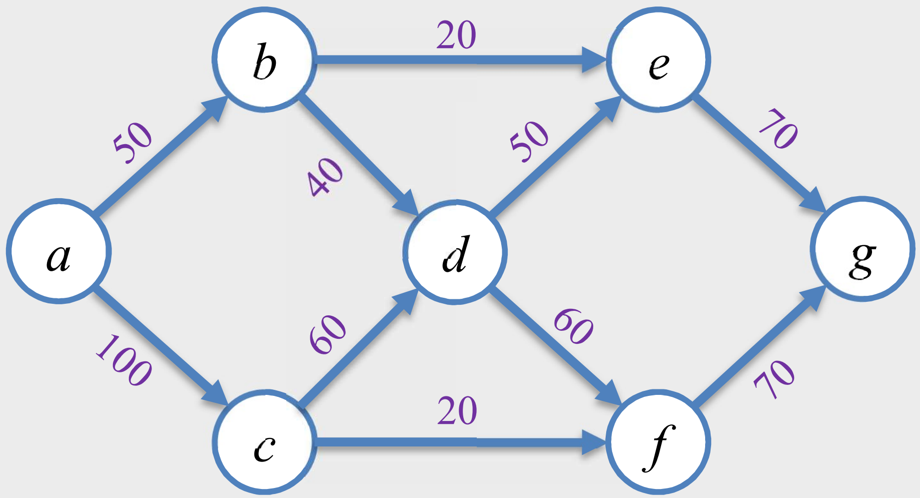
➢ What is the maximum possible flow from s to t in this network?



- What is the maximum possible flow from *s* to *t* in this network?



Maximum Flow Problem



Maximum Flow Problem

```
set   INTER;                               # intersections
param entr symbolic in INTER;              # entrance to road network
param exit symbolic in INTER,<> entr;# exit from road network
set   ROADS within (INTER diff {exit}) cross (INTER diff {entr});
param cap {ROADS} >= 0;                              # capacities
var Traff {(i,j) in ROADS} >= 0, <= cap[i,j];  # traffic loads

maximize Entering_Traff: sum {(entr,j) in ROADS} Traff[entr,j];

subject to Balance {k in INTER diff {entr,exit}}:
 sum {(i,k) in ROADS} Traff[i,k] = sum {(k,j) in ROADS} Traff[k,j];

data;
set    INTER := a b c d e f g ;
param  entr := a ;
param  exit := g ;
param: ROADS: time :=
       a b 50, a c 100
       b d 40, b e 20
       c d 60, c f 20
       d e 50, d f 60
       e g 70, f g 70 ;
```

Maximum Flow Problem

Shortest path

```
set   INTER;
param entr symbolic in INTER;
param exit symbolic in INTER,
                       <> entr;
set   ROADS
within (INTER diff {exit})
cross (INTER diff {entr});
param time {ROADS} >= 0;
var   Use {(i,j) in ROADS} >= 0;


minimize   Total_Time: sum
{(i,j) in ROADS}
time[i,j] * Use[i,j];

subject to Start: sum {(entr,j)
in ROADS} Use[entr,j] = 1;
subject to Balance {k in INTER
diff {entr,exit}}:
sum {(i,k) in ROADS} Use[i,k] =
sum {(k,j) in ROADS} Use[k,j];
```

Maximum flow

```
set   INTER;
param entr symbolic in INTER;
param exit symbolic in INTER,
                       <> entr;
set   ROADS
within (INTER diff {exit})
cross (INTER diff {entr});
param cap {ROADS} >= 0;
var Traff {(i,j) in ROADS} >= 0,
<= cap[i,j];

maximize Entering_Traff: sum
{(entr,j) in ROADS}
Traff[entr,j];


subject to Balance {k in INTER
diff {entr,exit}}:
sum {(i,k) in ROADS} Traff[i,k] =
sum {(k,j) in ROADS} Traff[k,j];
```

Maximum Flow Problem

Some applications of the maximum flow problem

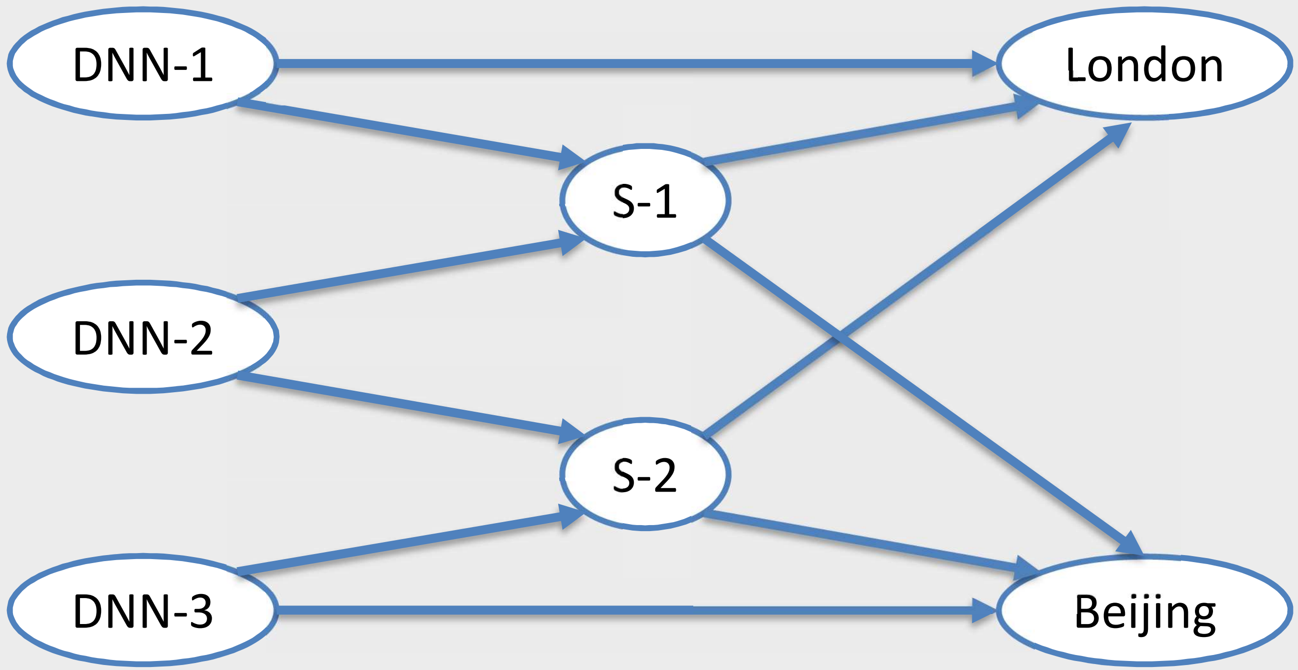
- Emergency, maximum rate for evacuation!
- What is the maximum number of packets that can be sent from server A to server B in a communications network?
- What is the maximum number of iPads that can be sent from the factories in China to the retail stores in the US?

Minimum Cost Flow Problem

Minimum Cost Flow Problem

- The Dantzig News Network (DNN) is preparing for the incredible demand for its upcoming Election Day video coverage, streamed over the Internet
- DNN has set up 3 dedicated servers to handle the load
- The major demand for the video stream is in 2 cities: London and Beijing
- There are some direct links from each of the DNN sites
- The streams can also be sent through 2 intermediate satellites

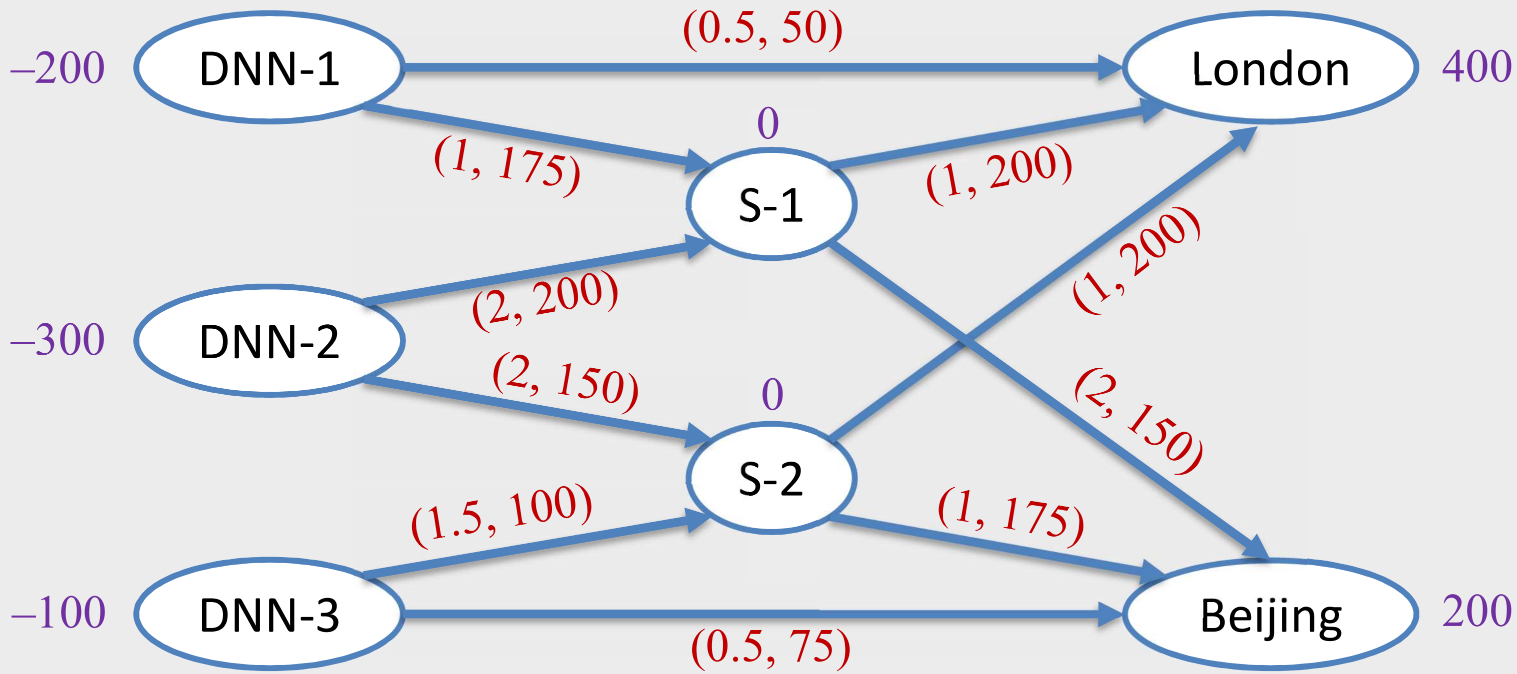
- Network (N, A)



Minimum Cost Flow Problem

- Each server has a specified supply per day (GB/day)
- Each city has a specified demand per day (GB/day)
- Each satellite has 0 demand
- Each link/arc (server-city, server-satellite, satellite-city) has
  - a cost per-unit-flow ($ per GB/day)
  - a capacity (GB/day)

Minimum Cost Flow Problem



- Numbers next to nodes are net demands:
  - positive numbers are demands: demand or sink nodes
  - negative numbers are supplies: supply or source nodes
  - 0 = no demand/supply: transhipment node
- Arc labels: (cost, capacity)

Minimum Cost Flow Problem

- How can we handle the required load at minimum cost?
- Decisions that need to be made:
  - How much flow (GB/day) should be sent from one node to another?

Minimum Cost Flow Problem

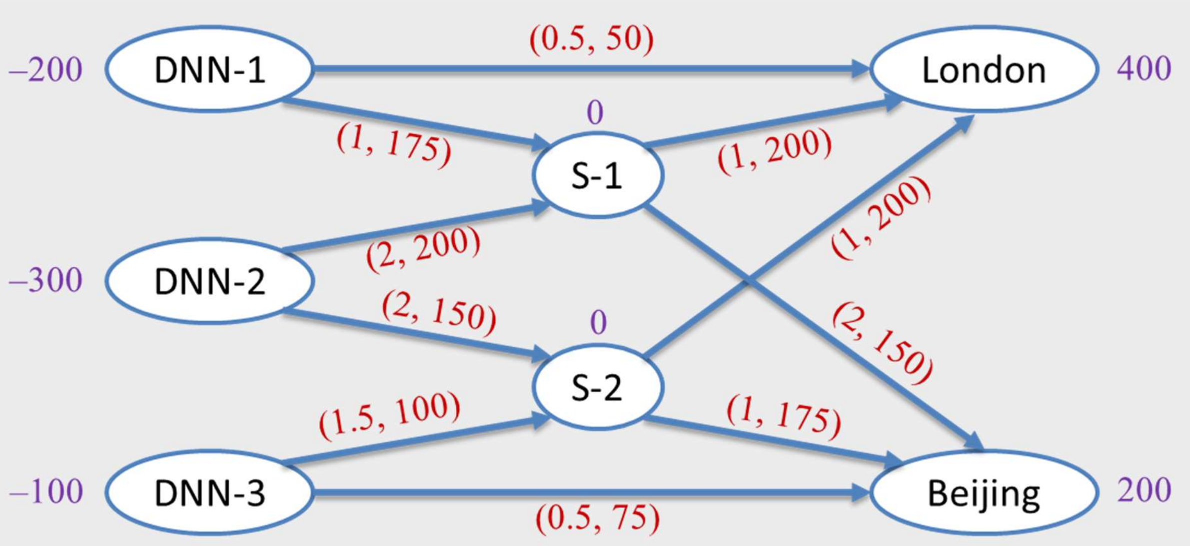
- A flow is an assignment of values to each arc $(i, j) \in A$
- Feasible flow (slightly different from max flow problem):
  - Flow on arc (i, j) is at most the capacity of arc (i, j)
  - Flow conservation: for each node $i \in N$

    total flow into node i – total flow out of node i = net demand of node i
- Demand node:

  net demand > 0 $\rightarrow$ (total flow in) > (total flow out)
- Supply node:

  net demand < 0 $\rightarrow$ (total flow in) < (total flow out)
- Transhipment node:

  net demand = 0 $\rightarrow$ (total flow in) = (total flow out)



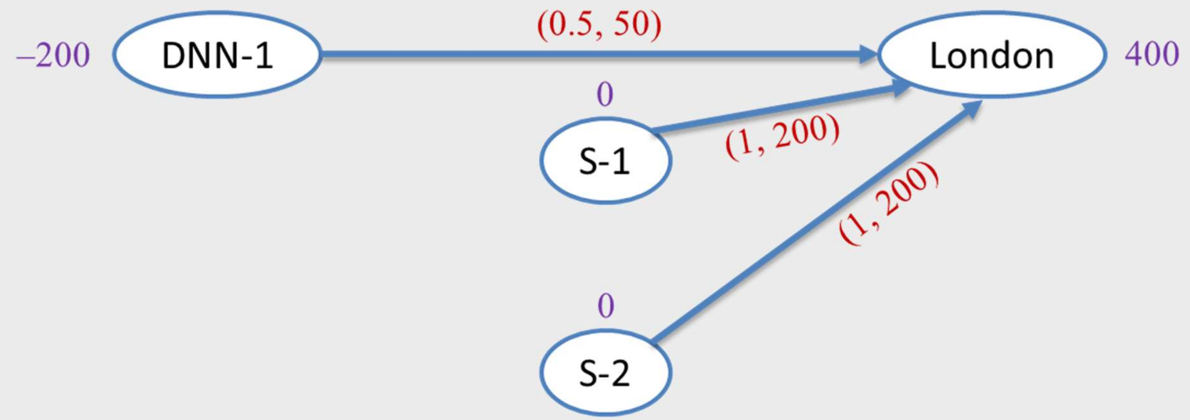
Decision variables:

$x_{i,j}$ = GB/day sent on arc $(i\ ,j)$ for $(i\ ,j) \in A$

Objective function:

minimize $0.5\ x_{DNN\text{-}1,London} + 1\ x_{DNN\text{-}1,S\text{-}1} + 2\ x_{DNN\text{-}2,S\text{-}1} + 2\ x_{DNN\text{-}2,S\text{-}2} +$

$1.5\ x_{DNN\text{-}3,S\text{-}2} + 0.5\ x_{DNN\text{-}3,Beijing} + 1\ x_{S\text{-}1,London} + 2\ x_{S\text{-}1,Beijing} + 1\ x_{S\text{-}2,London} + 1\ x_{S\text{-}2,Beijing}$



Constraints:

The demand node London:

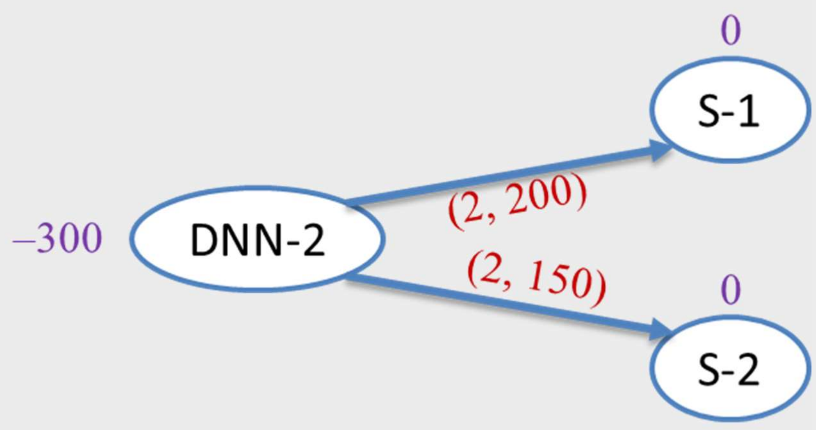
We need to impose flow conservation:

(total flow in) – (total flow out) = net demand

So, for the London node:

$$(x_{DNN\text{-}1,London} + x_{S\text{-}1,London} + x_{S\text{-}2,London}) = 400$$

MINIMUM COST FLOW PROBLEM



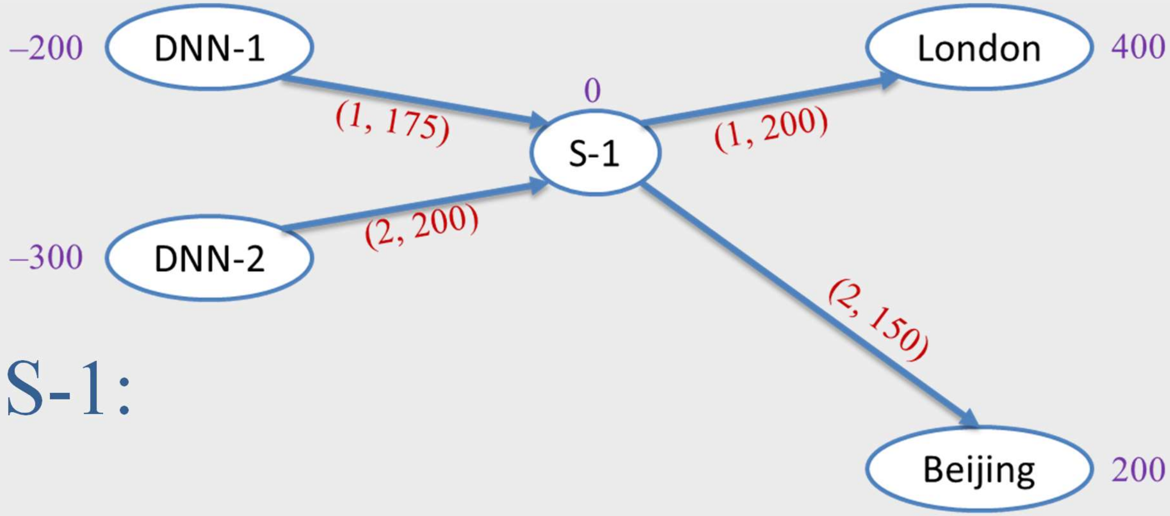
The supply node DNN-2:

Flow conservation:

$$(\text{total flow in}) - (\text{total flow out}) = \text{net demand}$$

So, for the node DNN-2:

$$-\left(x_{DNN\text{-}2,S\text{-}1} + x_{DNN\text{-}2,S\text{-}2}\right) = -\,300$$



The transhipment node S-1:

Flow conservation:

(total flow in) – (total flow out) = net demand

So, for the S-1 node:

$$(x_{DNN\text{-}1,S\text{-}1} + x_{DNN\text{-}2,S\text{-}1}) - (x_{S\text{-}1,London} + x_{S\text{-}1,Beijing}) = 0$$

Minimum Cost Flow Problem

Flow conservation constraints for every node:

$$-(x_{DNN\text{-}1,London} + x_{DNN\text{-}2,S\text{-}1}) = -200 \qquad (DNN\text{-}1)$$

$$-(x_{DNN\text{-}2,S\text{-}1} + x_{DNN\text{-}2,S\text{-}2}) = -300 \qquad (DNN\text{-}2)$$

$$-(x_{DNN\text{-}3,S\text{-}2} + x_{DNN\text{-}2,Beijing}) = -100 \qquad (DNN\text{-}3)$$

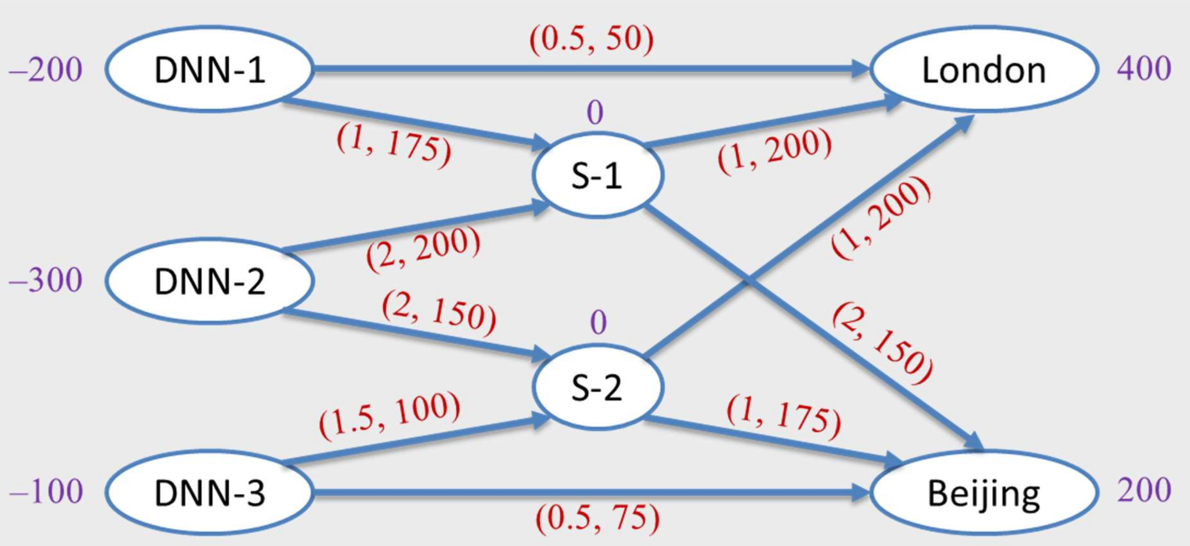
$$(x_{DNN\text{-}1,S\text{-}1} + x_{DNN\text{-}2,S\text{-}1}) - (x_{S\text{-}1,London} + x_{S\text{-}1,Beijing}) = 0 \qquad (S\text{-}1)$$

$$(x_{DNN\text{-}2,S\text{-}2} + x_{DNN\text{-}3,S\text{-}2}) - (x_{S\text{-}2,London} + x_{S\text{-}2,Beijing}) = 0 \qquad (S\text{-}2)$$

$$(x_{DNN\text{-}1,London} + x_{S\text{-}1,London} + x_{S\text{-}2,London}) = 400 \qquad (\text{London})$$

$$(x_{S\text{-}1,Beijing} + x_{S\text{-}2,Beijing} + x_{DNN\text{-}3,Beijing}) = 200 \qquad (\text{Beijing})$$

MINIMUM COST FLOW PROBLEM



What else?

- ✓ Flows should be nonnegative
- ✓ Flows on arcs have capacities

So, for example, arc (*DNN-2*, *S-1*):

$$x_{DNN\text{-}2,S\text{-}1} \geq 0 \qquad x_{DNN\text{-}2,S\text{-}1} \leq 200$$

Minimum Cost Flow Problem

$$\textit{minimize} \quad 0.5\, x_{DNN\text{-}1,London} + 1\, x_{DNN\text{-}1,S\text{-}1} + 2\, x_{DNN\text{-}2,S\text{-}1} + 2\, x_{DNN\text{-}2,S\text{-}2} + 1.5\, x_{DNN\text{-}3,S\text{-}2}$$

$$+ 0.5\, x_{DNN\text{-}3,Beijing} + 1\, x_{S\text{-}1,London} + 2\, x_{S\text{-}1,Beijing} + 1\, x_{S\text{-}2,London} + 1\, x_{S\text{-}2,Beijing}$$

$$\left[\begin{array}{l} -(x_{DNN\text{-}1,London} + x_{DNN\text{-}2,S\text{-}1}) = -200 \\ -(x_{DNN\text{-}2,S\text{-}1} + x_{DNN\text{-}2,S\text{-}2}) = -300 \\ -(x_{DNN\text{-}3,S\text{-}2} + x_{DNN\text{-}2,Beijing}) = -100 \end{array}\right.$$

$$\left[\begin{array}{l} (x_{DNN\text{-}1,S\text{-}1} + x_{DNN\text{-}2,S\text{-}1}) - (x_{S\text{-}1,London} + x_{S\text{-}1,Beijing}) = 0 \\ (x_{DNN\text{-}2,S\text{-}2} + x_{DNN\text{-}3,S\text{-}2}) - (x_{S\text{-}2,London} + x_{S\text{-}2,Beijing}) = 0 \end{array}\right.$$

$$\left[\begin{array}{l} (x_{DNN\text{-}1,London} + x_{S\text{-}1,London} + x_{S\text{-}2,London}) = 400 \\ (x_{S\text{-}1,Beijing} + x_{S\text{-}2,Beijing} + x_{DNN\text{-}3,Beijing}) = 200 \end{array}\right.$$

$$\left[\begin{array}{l} x_{DNN\text{-}1,London} \leq 50;\ x_{DNN\text{-}1,S\text{-}1} \leq 175;\ x_{DNN\text{-}2,S\text{-}1} \leq 200;\ x_{DNN\text{-}2,S\text{-}2} \leq 150 \\ x_{DNN\text{-}3,S\text{-}2} \leq 100;\ x_{DNN\text{-}3,Beijing} \leq 75;\ x_{S\text{-}1,London} \leq 200;\ x_{S\text{-}1,Beijing} \leq 150 \\ x_{S\text{-}2,London} \leq 200;\ x_{S\text{-}2,Beijing} \leq 175 \end{array}\right.$$

$$\left[\begin{array}{l} x_{DNN\text{-}1,London} \geq 0;\ x_{DNN\text{-}1,S\text{-}1} \geq 0;\ x_{DNN\text{-}2,S\text{-}1} \geq 0;\ x_{DNN\text{-}2,S\text{-}2} \geq 0;\ x_{DNN\text{-}3,S\text{-}2} \geq 0; \\ x_{DNN\text{-}3,Beijing} \geq 0;\ x_{S\text{-}1,London} \geq 0;\ x_{S\text{-}1,Beijing} \geq 0;\ x_{S\text{-}2,London} \geq 0;\ x_{S\text{-}2,Beijing} \geq 0 \end{array}\right.$$

MINIMUM COST FLOW PROBLEM

- Network (N, A)
- Specified net demand b_i for every node $i \in N$
  - $bi > 0$: demand at node i
  - $bi < 0$: supply at node i
  - $bi = 0$: node i is a transhipment node
- Cost per unit flow $c_{i,j}$ for every arc $(i\ ,j) \in A$
- Flow capacity $u_{i,j}$ for every arc $(i\ ,j) \in A$

Decision variables:

$$x_{i,j} = \text{flow sent on arc } (i\ ,j) \qquad \text{for } (i\ ,j) \in A$$

Minimum Cost Flow Problem

$$\textit{minimize} \sum_{(i,j)\in A} c_{i,j} x_{i,j}$$

subject to

$$\sum_{(i,k)\in A} x_{i,k} - \sum_{(k,j)\in A} x_{k,j} = b_k \qquad \text{for } k \in N$$

$$0 \le x_{i,j} \le u_{i,j} \qquad \text{for } (i,j) \in A$$

ASSIGNMENT #4:

AMPL BOOK
CHAPTER 15. EXERCISES(1-8)

LECTURE #8:

MILP