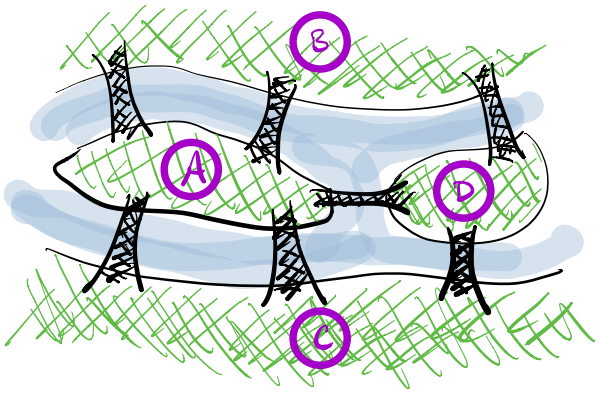
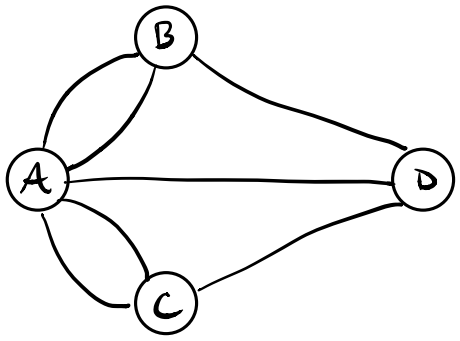
CHAPTER 8 - Network Models

Section 8.1: Basic Definitions

- In the Prussian City of Königsberg, founded in 1254, on the banks of the river Pregel, the city consists of seven bridges connecting four regions:



Network Representation



Question: Is it possible to visit all four cities in a round-trip, crossing all bridges exactly once?

- Euler solved the problem, proving it was impossible by using a parity construction argument. Later, the city was depicted by a graph with nodes and edges as above. This problem was a seed for graph theory.

- **Def:** A graph, or network, call it G, is an ordered pair, $G = (V, E)$, of two sets V and E, where

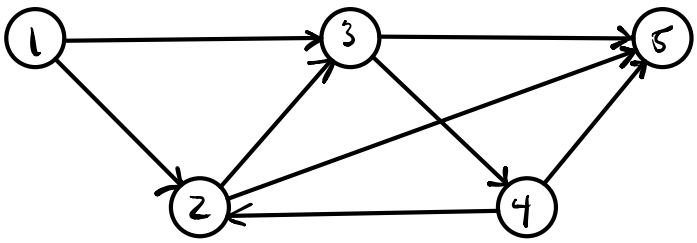
- V is the set of vertices or nodes
- E is the set of edges or arcs

- **Def:** A edge is weighted if there is a cost, length, flow, or value associated to it.
  - the associated weight is problem dependent.

- **Def:** An edge is directed or oriented if there is a positive flow in one direction only.

- **Def:** A directed graph or directed network is a graph whose edges are directed.
  - a graph is undirected if the edges link two vertices symmetrically, both directions.

- **Ex:**



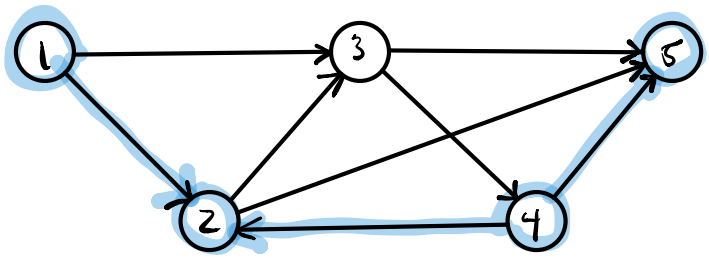
Unweighted, Directed Graph

$G = (V, E)$ where

- $V = \{1, 2, 3, 4, 5\}$
- $E = \{(1,2), (1,3), (2,3), (2,5), (3,4), (3,5), (4,2), (4,5)\}$

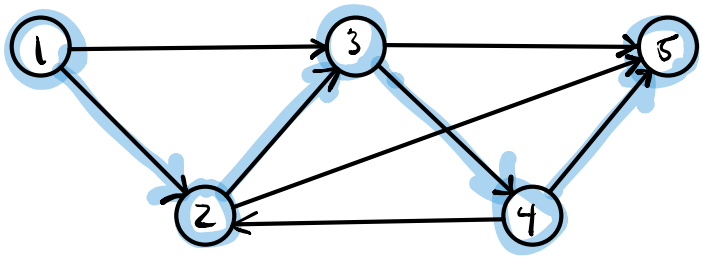
- **Def:** A chain is a sequence of edges such that every edge has exactly one vertex in common with the previous edge.

  Ex: From the graph above, $\{(1,2),(4,2),(4,5)\}$ is a chain.



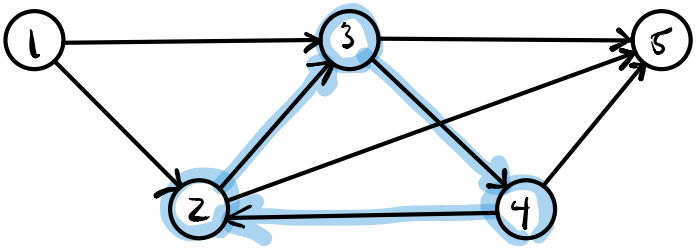
- **Def:** A path is a chain in which the terminal vertex of each edge is the initial vertex of the next edge.
  - for our purposes, a path is a directed path in. all edges in the path flow the same direction.

  Ex: From the graph above, $\{(1,2),(2,3),(3,4),(4,5)\}$ is a path from 1 to 5.



- **Def:** A cycle or loop is a path that connects a node back to itself.

  Ex: From the graph above, $\{(2,3),(3,4),(4,2)\}$ is a cycle.



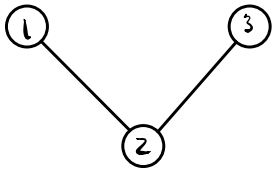
- **Def:** A graph is connected if every two distinct nodes are linked by at least one edge.

  Ex: The graph above is connected.

- **Def:** A tree is a cycle-free connected network comprised solely of a subset of vertices.
  - a tree is an undirected subgraph of the original graph with no cycles.

- **Def:** A spanning tree is a tree that links all the nodes of a graph.

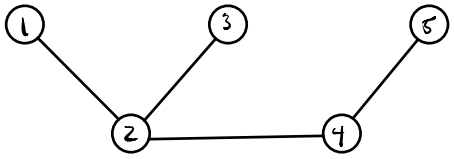
Ex: Tree:



$T = (V_T, E_T)$ where

- $V_T = \{1, 2, 3\}$
- $E_T = \{(1,2), (2,3)\}$

Spanning tree:



$S = (V_S, E_S)$ where

- $V_S = \{1, 2, 3, 4, 5\}$
- $E_S = \{(1,2), (2,3), (2,4), (4,5)\}$

- We'll study three types of optimization problems associated to the network:

1.) Shortest Path Problem: problem of finding the path connecting the source node to the destination node with the least/minimum length/cost. (8.2)

- Dijkstra's Algorithm

Generalized to Min-cost Network Flow Problem (8.5)

2.) Maximum Flow Problem: problem of finding the maximum amount to transport from a source node to a destination node; weights represent maximum capacity for transport.

- Ford-Fulkerson Method. (8.6)

3.) Minimum Spanning Tree: finding the minimum cardinality spanning tree (8.6).

WS #1, #2 working with some basic definitions.

Section 8.2: Shortest Path Problem

- In many optimization problems on networks, we may be interested in finding the shortest path (in terms of distance or overall weight of the edges) from one node (called the source) to another terminal node (called the sink or destination).
- The main algorithm we'll discuss is Dijkstra's Algorithm.

1.) Definitions:

- Let u_i be the shortest distance/weight from the source to node i.

- Let d_{ij} be the distance/weight of the edge connecting node i to node j. ($d_{ij} > 0$).
- Each node will receive a temporary or permanent labeling. The labeling given to an immediately succeeding node j is

Predecessor node

Current shortest distance to node i from source

$$[u_j, i] = [u_i + d_{ij}, i].$$

- The label for the starting source node is

$$[0, -].$$

- A temporary label is modified/replaced if a shorter route can be found. Otherwise, it is changed to permanent.

2.) Algorithm

Step 0: Label source as $[0, -]$. Set $k = 1$.

Step k: a.) Let i be the current node such that u_i is the shortest distance/weight away from the source. (If $k=1$, i is the source node.) Compute the temporary labels $[u_i + d_{ij}, i]$ for every node j (not permanent) such that $d_{ij} > 0$. If node j already has a temporary label $[u_j, s]$ and $u_i + d_{ij} < u_j$, then replace it.

b.) If all nodes have permanent labels, stop. Otherwise, select the label $[u_r, t]$ with the shortest distance (which would be equal to u_r) among all temporary labels. (Note: t could be i or some other predecessor.) Break ties

arbitrarily. Set $i = r$ and $k = k+1$.

[US] #1, #2, #3 work with Dijkstra's Algorithm.

- We may formulate a Shortest Path problem as a transportation LP (actually as a transshipment LP) by defining the source node as the sole source, called the Supply point, and the destination node as the demand point. All other nodes act as transshipment points that have a supply and demand.

- We can think of the shortest path problem as an attempt to ship 1 unit of good from the source to the destination using the intermediate nodes as "stops" along the way. Each stop comes with an associated cost (weight of the edge). Every trans-shipment point has a supply and demand equal to 1. Thus, transportation LP solution

$$x_{ij} = \begin{cases} 1 & \text{uses edge } (i,j) \\ 0 & \text{doesn't use edge } (i,j) \end{cases}.$$

More like an assignment problem with a different interpretation. Some additional ideas:

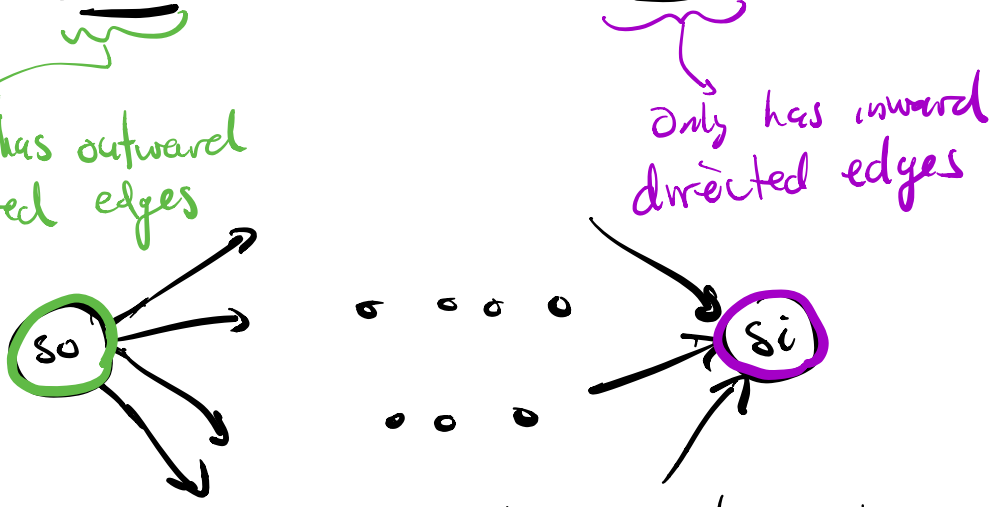
- if there is no edge (i,j), cost is set to M.
- cost of shipping from node i to node i is 0.

- If $X_{ii}=1$, then that node i is not used.

WS #1, #2. Formulate Shortest Path as Transportation LP.

Section 8.3: Maximum-Flow Problems

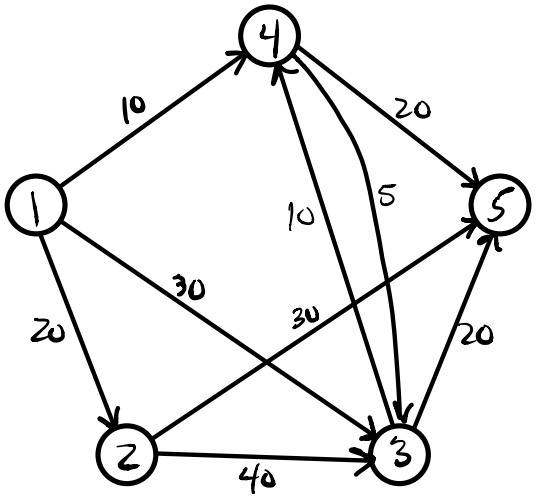
- Many scenarios arise where a network is connected via edges where edge weight corresponds to a maximum capacity that limits some quantity. (e.g. crude oil flowing through a pipeline.)
- Goal: Transport the maximum amount of the quantity from a source node to a sink node.



- An LP Formulation can be somewhat easily obtained by applying two basic assumptions:
  1.) $0 \leq$ (flow through an edge) $\leq$ (edge capacity)
  2.) (flow into node i) = (flow out of node i)

- Note: Assumption is called the conservation of flow constraint.

- We'll demonstrate with the following example:



- Suppose we want to maximize the flow from node 1 to node 5.
- Define the following decision variables:

  x_{ij} = units of quantity flowing from node i to node j.

- Define Objective Function:

  Maximize $Z = x_{12} + x_{13} + x_{14}$.

- Constraints:

(Capacity Constraints)

$$x_{12} \leq 20$$
$$x_{13} \leq 30$$
$$x_{14} \leq 10$$
$$x_{23} \leq 40$$
$$x_{25} \leq 30$$
$$x_{34} \leq 10$$
$$x_{35} \leq 20$$
$$x_{43} \leq 5$$
$$x_{45} \leq 20$$

(Conservation Constraints)

$$x_{12} = x_{23} + x_{25}$$
$$x_{13} + x_{23} + x_{43} = x_{34} + x_{35}$$
$$x_{14} + x_{34} = x_{43} + x_{45}$$

$$x_{ij} \geq 0.$$

- Every maximum flow problem of this nature will have an LP representation using the capacity and conservation of flow constraints

IOS #1-#3 work with formulating Max flow problems.

Section 8.5: Minimum Cost Network Flow Problems.

- Any transportation, transshipment, assignment, shortest path, or maximum flow problem is a special case of a Minimum Cost Network Flow Problem (MCNFP).
- note: The MNCFP formulation is essentially identical to the LP formulation of a Max Flow problem. The key is to define an edge capacity even if it is not explicitly stated. Also, define the net flow from any node.
- A MNCFP is characterized by the following definitions:
  - Decision Variables:
    x_{ij} = number of units of flow sent from node i to node j.
  - Obj. Function Coeffs:
    c_{ij} = cost of transporting one unit from node i to node j.
  - Right-hand side of constraints:
    b_i = net supply (outflow − inflow) at node i
    * the values of b_i are dictated by problem type

- Lower and upper bounds on flow:

L_{ij} = lower bounds (usually 0)

U_{ij} = upper bounds (dictated by capacities)

- With these definitions, an MCNFP is formulated as follows:

$$\text{Minimize } Z = \sum_{\text{all edges}} c_{ij} x_{ij}$$

Subject to

$$\sum_j x_{ij} - \sum_k x_{ki} = b_i \quad \text{(for each node } i\text{)}$$

$$L_{ij} \le x_{ij} \le U_{ij} \quad \text{(for each edge)}$$

WS #1-#3 work with formulating MCNFPs.

Section 8.6: Minimum Spanning Tree Problems

- Recall that a spanning tree of a graph with n nodes is a set of n-1 edges that connects all nodes, where direction of edges is ignored and no cycles are present.
- A minimum spanning tree is a spanning tree with minimum edge cost. (Hence, we are essentially considering undirected weighted graphs.)

• A simple algorithm: Let V be the set of vertices of the graph, define

C_k = set of nodes permanently connected at k^{th} steps.

$\overline{C}_k$ = set of nodes yet to be connected at k^{th} step.

Step 0: Set $C_0 = \emptyset$ and $\overline{C}_0 = V$.

Step 1: Start with any node i in the unconnected set $\overline{C}_0$. Set $C_1 = \{i\}$, rendering $\overline{C}_1 = V \setminus \{i\}$. Set $k = 2$.

General Step k: Select node j from $\overline{C}_{k-1}$ that yields the shortest edge to a node in C_k. Link node j to C_{k-1} and remove it from $\overline{C}_{k-1}$ to obtain C_k and $\overline{C}_k$, respectively. Stop if $\overline{C}_k$ is empty; otherwise set $k = k+1$.

WS #1, #2 working with minimum spanning tree problems