Elementary definitions in Graph Theory

Pramesh Kumar

IIT Delhi

December 31, 2023

Outline

Introduction

Definition (Network). A network is interconnection among set of items.

Outline

Internet network

Figure: Source: https://www.discovery.org/a/25/

Social network

Figure: Source: Medium

Highway network

Figure: Twin cities highway network (Source:CEGE5214)

Transit network

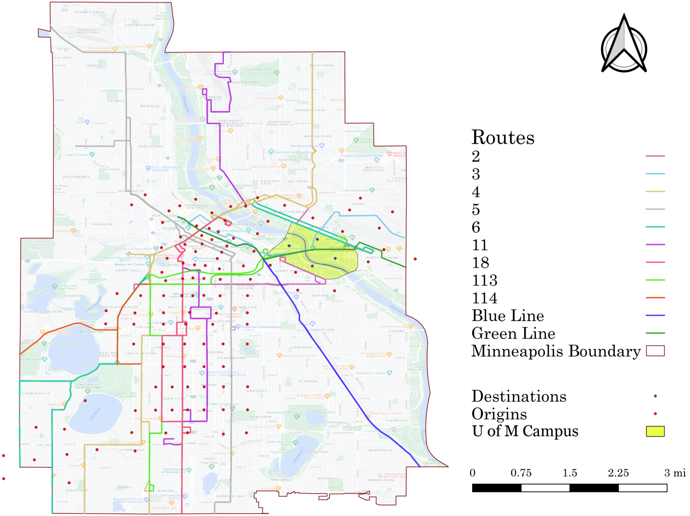


Figure: South Minneapolis transit network

Airline network

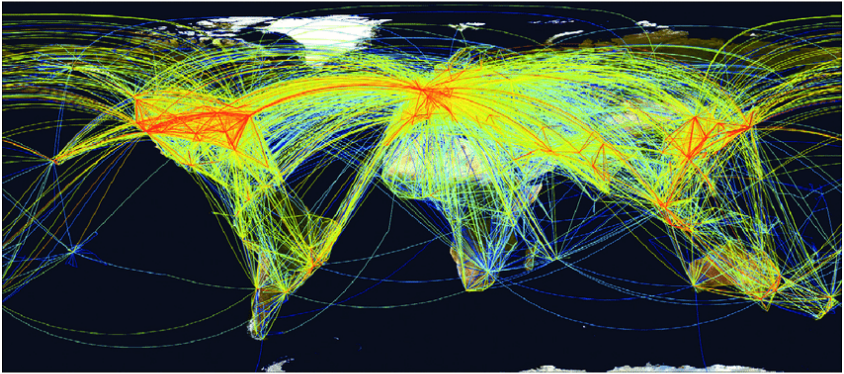


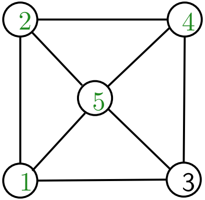
Figure: Source: Sarah Randolph on ResearchGate

Outline

Undirected graph

Definition (Undirected graph/network). An undirected graph G is a pair (N, A), where N is the set of nodes and A is the set of links whose elements are unordered pair of distinct nodes.

Example(s). $N = \{1, 2, 3, 4, 5\}$,
$A = \{(1, 2), (1, 3), (1, 5), (5, 4), (5, 3), (5, 2), (2, 4), (3, 4)\}$

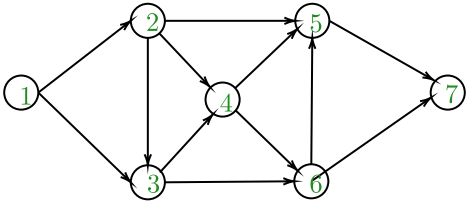


Remark. Let $|N| = n$. Then, $|E| = m \leq \frac{n(n-1)}{2}$.

Directed graph

Definition (Directed network/graph). A directed graph is pair (N, A), where N denotes the set of nodes/vertices and $A \subseteq N \times N$ denotes the set of links/edges/arcs whose elements are ordered pair of distinct nodes.

Example(s). $N = \{1, 2, 3, 4, 5, 6, 7\}$
$A = \{(1,2), (1,3), (2,3), (2,4), (2,5), (3,4), (3,6), (4,5), (4,6), (5,7), (6,5), (6,7)\}$



Definition (). If $e = (i, j) \in A$, then

1. i and j are endpoints of e.
2. i is the tail node and j is the head node of e.
3. (i, j) emanates from i and terminates at node j.
4. (i, j) is incident to nodes i and j.
5. (i, j) is outgoing link of node i and incoming link of node j.

Definition (Degree). The number of incoming and outgoing links of a node $i \in N$ are called indegree and outdegree respectively. The sum of indegree and outegree is called degree.

Definition (Multilinks). Two or more links with same head and tail nodes.

Definition (Loop). A link whose tail and head nodes are the same.

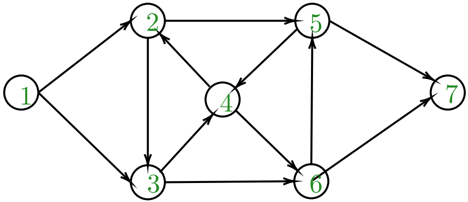
Note: In this course, we assume that graphs contain no loops or multiarcs.

Definition (Subgraph). A graph $G^{'}(N^{'}, A^{'})$ is a subgraph of $G(N, A)$ if $N^{'} \subseteq N$ and $A^{'} \subseteq A$. A subgraph $G^{'}(N^{'}, A^{'})$ of $G(N, A)$ is said to be induced by $N^{'}$ if $A^{'}$ contains links with their end points in $N^{'}$.

Definition (Walk). A collection of links $W = \{(u_1, v_1), \cdots, (u_q, v_q)\}$ is an $s - t$ walk if

1. $u_1 = s$
2. $v_i = u_{i+1}, \forall i = 1, ..., q - 1$
3. $v_q = t$

Example(s).



$W_1 = \{(1,2),(2,5),(5,7)\}$,
$W_2 = \{(1,2),(2,3),(3,4),(4,2),(2,5),(5,7)\}$,
$W_3 = \{(1,3),(3,6),(6,5),(5,4),(4,6),(6,7)\}$
are all exmples of $1-7$ walks.

Definition (Path). An $s-t$ path is an $s-t$ walk without any repeated nodes.
In above example, W_1 is a $1-7$ path while W_2 and W_3 are not.

Definition (Cycle). A cycle is a path with same first and last nodes.

Definition (Tour). A tour is a cycle including all nodes of the graph.

Definition (Acyclic graph). A graph without any cycles is acyclic.

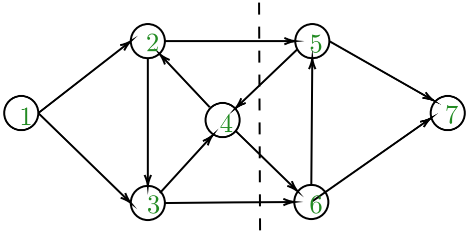
Definition ().

1. Nodes $i \in N$ and $j \in N$ are said to be connected if there exists at least one path between i and j.
2. A graph is said to be connected graph if every pair of its nodes are connected. Otherwise, the graph is called disconnected.

Definition (Cut). A cut is a partition of nodes into two subsets S and $\bar{S} = N \backslash S$.

- Each cut defines a set of links with one endpoint in S and another in $\bar{S}$. This set of links is denoted by $(S, \bar{S})$.
- An $s - t$ cut is a cut $(S, \bar{S})$ with $s \in S$ and $t \in \bar{S}$.

Example(s).



$S = \{1, 2, 3, 4\}, \bar{S} = \{5, 6, 7\}$, and $(S, \bar{S}) = \{(2, 5), (5, 4), (4, 6), (3, 6)\}$ defines a $1 - 7$ cut.

Definition (Tree). A tree is a connected graph that contains no cycles.

Proposition

1. *A tree on n nodes contains exactly $n-1$ links.*
2. *A tree has at least 2 leaf nodes (i.e., nodes with degree 1).*
3. *Every pair of nodes are connected by a unique path.*

Proof.

1. (Proof by induction) Let $P(n)$ be the statement that a tree on n nodes contains exactly $n-1$ links. $P(1) = 0$ since there is only one node and a link requires at least two nodes. Let us assume that $P(k)$ is true, i.e., a tree on k nodes contains exactly $k-1$ links. Then, we can add another node to this graph with one link and that would still be a tree with k links, which means that $P(k+1)$ is true.
2. Assuming $n < \infty$, we prove this by contradiction. Assume that a tree on n nodes has only one leaf u. Then, find the longest path from u in the tree. The longest path cannot end at u because that is not a path but cycle. Let us assume that it ends at v. If v has degree 1 then we are done. If it has degree 2, then it is not a longest path.
3. Proof by induction. Not possible to add another node without creating a cycle.

□

Definition (Forest). A graph that contains no cycles is a forest or a forest is a collection of trees.

Definition (Subtree). A connected subgraph of a tree is called a subtree.

Definition (Spanning tree). A spanning tree of a graph is a subgraph which is a tree connecting all the nodes.

Definition (Fundamental cycle). Adding a nontree link to the spanning tree creates a cycle. Such cycle is known as fundamental cycle. There are $m + n - 1$ fundamental cycles in the graph.

Definition (Fundamental cut). Any non-empty partition of a spanning tree nodes into two subset is a fundamental cut. There are $n - 1$ fundamental cuts.

Bipartite graphs

Definition (Bipartite graph). A graph $G(N, A)$ is bipartite if we can partition N into two subsets N_1 and N_2 such that for every link $(i, j) \in A$, we have either $i \in N_1$ and $j \in N_2$ or $j \in N_1$ and $i \in N_2$.

Proposition

A graph G is bipartite if and only if every cycle in G contains an even number of links.

Proof.

($\Rightarrow$) Assume that G is bipartite. Then, every step of a walk will take you either from N_1 to N_2 or N_2 to N_1. To form a cycle, you need to come back where you started requires even number of steps.

($\Leftarrow$) Assume that every cycle in G is even. Then, starting from one node $u \in C_1$ along a path/cycle, put nodes at odd distance in C_2 and nodes at even distance in C_1. Do this for every connected components of G. We cannot have link between two nodes within C_1 or C_2, otherwise cycle will be odd. $\square$

Outline

Network representation

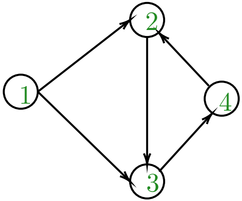
- The performance of a network algorithm depends not only on the algorithm but also on which data structure we use to store the network.
- We need to store how nodes are connected as well as capacities or costs associated to links.

Data structures

1. Node-link incidence matrix
2. Node-node adjacency matrix
3. Adjacency list
4. Forward (Backward) Star

Node-link incidence matrix

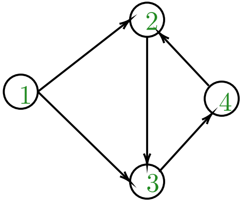
- Nodes on the rows and links on the columns.
- For every $(i, j) \in A$, we have $+1$ in row i and -1 in row j of column (i, j).
- Only 2 non-zero entries in every column and $2m$ non-zero entries in total.
- Not space efficient data structure



$$\mathcal{N} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccccc} (1,2) & (1,3) & (2,3) & (3,4) & (4,2) \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & -1 \\ 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix} \end{array}$$

Node-node adjacency matrix

- $|N| \times |N|$ matrix $\mathcal{H}$.
- $H_{ij} = 1$ if $(i, j) \in A$, 0, otherwise.
- We can store capacities and cost of edges using similar matrix.
- m non-zero elements.



$$\mathcal{H} = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

Adjacency lists

- The node adjacency list of a node $i \in N$ is $A(i) = \{j \in N \mid (i,j) \in A\}$ (set of emanating nodes)
- Stored as a linked list for every node
- Need a linked list with $|A(i)|$ cells for every node $i \in N$
- We can also store costs and capacities associated to the links in these cells

Forward and reverse star

- Forward star stores node adjacency lists in a large array.
- Assigns a unique sequence number to each link in a specific order: first those emanating from node 1, then node 2, and so on.
- Store information about $tail, head, cost$, and $capacity$ in separate arrays.
- If the sequence number of arc (i, j) is 10,then one can call $tail[10], head[10], cost[10]$,and $capacity[10]$ to get the information about (i, j).
- Also maintains a pointer for each node i, i.e., $point(i)$ that indicates the smallest numbered link in the list of links for that node.
- FS will store the outgoing links of node i at positions $point(i)$ and $point(i + 1) - 1$.
- Reverse star stores the incoming links in the similar fashion. The sequence starts with node 1 and stores all its incoming links, and so on.
- This is more space efficient than adjacency list.

Network representations	Storage space	Features
Node–arc incidence matrix	nm	1. Space inefficient 2. Too expensive to manipulate 3. Important because it represents the constraint matrix of the minimum cost flow problem
Node–node adjacency matrix	kn^2 for some constant k	1. Suited for dense networks 2. Easy to implement
Adjacency list	$k_1 n + k_2 m$ for some constants k_1 and k_2	1. Space efficient 2. Efficient to manipulate 3. Suited for dense as well as sparse networks
Forward and reverse star	$k_3 n + k_4 m$ for some constants k_3 and k_4	1. Space efficient 2. Efficient to manipulate 3. Suited for dense as well as sparse networks

Figure 2.25 Comparison of various network representations.

Figure: Source: AMO

Thank you!