Graphs

Bhadrachalam Chitturi

University of Texas at Dallas.

Relations

- A binary relation from set A to set B is a subset of A × B, where A × B is the Cartesian product of A and B.
- A × B can be represented either as a set of ordered pairs or as an adjacency matrix.
- Let A={a, b, c} and B]{1,2}. Then C= A × B = {(a,1), (a,2), (b,1), (b,2), (c,1), (c,2)}. If |A|=m and |B|=n then |C|=mn. C can also be denoted as a 3×2 matrix M1. M2 and M3 denote subsets of C. M1, M2 and M3 are relations from A to B. Rows correspond to a, b, c and columns correspond to 1, 2. Each entity of the matrix is either a 1 or a 0.

M1

	1	**2**
a	1	1
b	1	1
c	1	1

M2

	1	**2**
a	1	1
b	0	1
c	1	0

M3

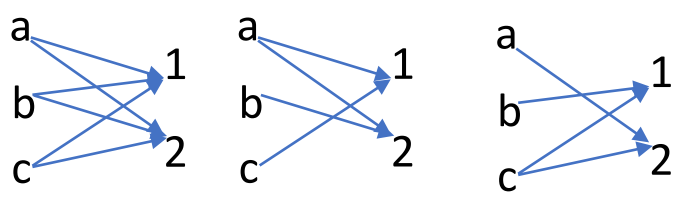
	1	**2**
a	0	1
b	1	0
c	1	1

The above relations can be denoted with a graph also.

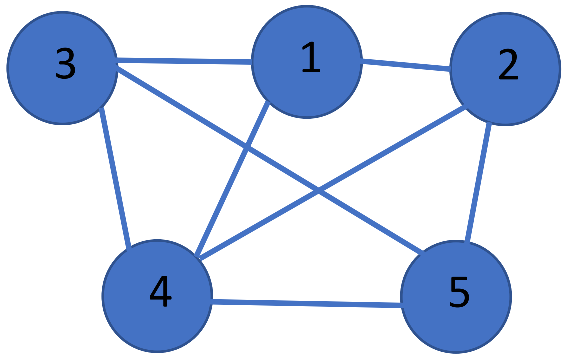
- A graph G=(V,E) is an ordered pair of sets. Where set V is the set of objects called as vertices or nodes and set E denotes the relation from V to V or E is a subset of V×V.

Graphs

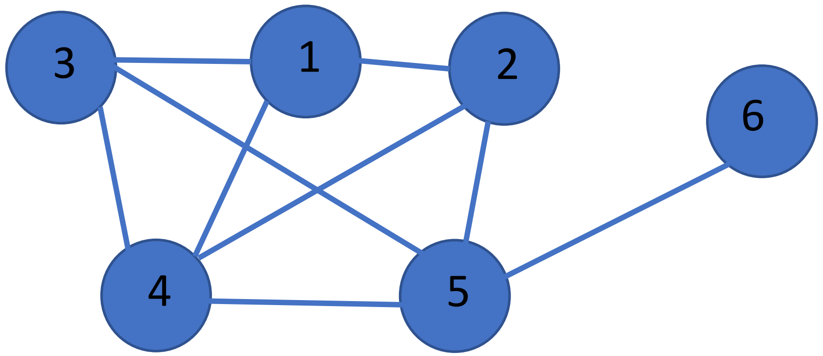
- A graph G=(V,E) is an ordered pair of sets. Where set V is the set of objects called as vertices or nodes and set E denotes the relation from V to V or E is a subset of V×V.
- The graphs G1, G2 and G3 corresponding to M1, M2 and M3 resp. are:



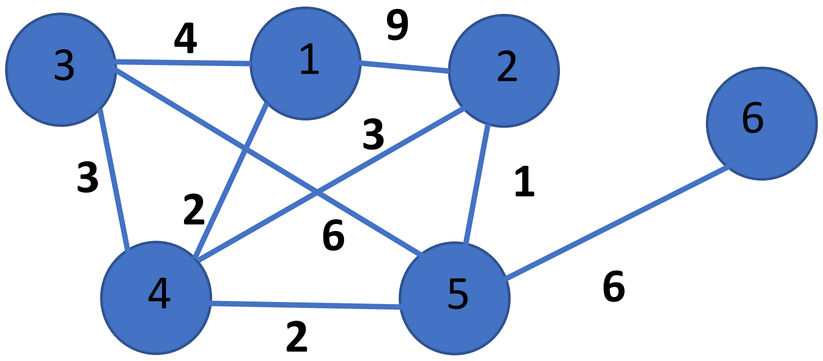
- Here the V={a, b, c, 1, 2}. Another example follows where V={1,2,3,4,5}
- 0 1 1 1 0
1 0 0 1 1
1 0 0 1 1
1 1 1 0 1
0 1 1 1 0



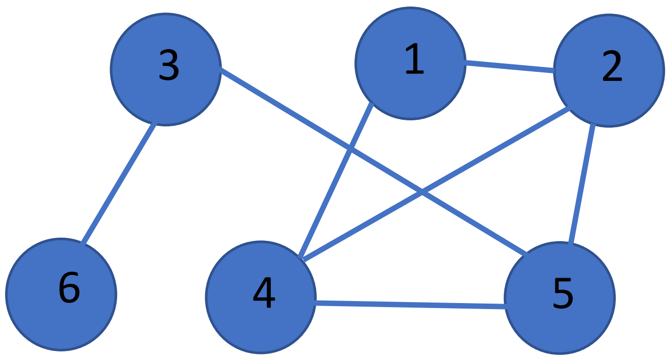
- Write the adjacency matrix for:

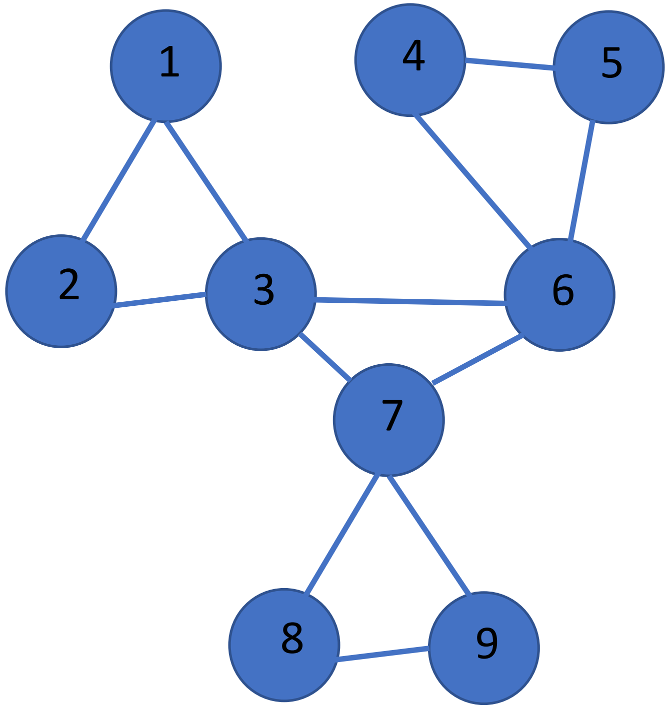


- Write the adjacency matrix for:



- Identify articulation points and bridges





Graphs

- The last graph is symmetric, that is if (a,b) is in the edge set E, i.e. (a,b) is an edge then (b,a) is also an edge.
  Directed/Undirected graphs:
  A graph that is symmetric is denoted an undirected graph. Here the edges do not have arrows indicating that a line segment denotes bi-directional relation.
  Otherwise, edges have directions. In such a case if (a,b) is an edge then (b,a) need not be an edge.

Correspondence of sets, matrices and graphs

- A binary relation R has the corresponding representations as
- a set S_R, a matrix M_R, and a graph $G_R = (V_R, E_R)$. All are equivalent.

An ordered pair (a,b) is in S_R

$\leftrightarrow M_R[a,b]=1$

$\leftrightarrow$ (a,b) is in E_R.

Types of Graphs

- An edge in a graph can have an associated weight. That is, there is a function f that maps an edge e_i into a real number w_i. $\forall_x\ f(e_x)=w_x$. Such a graph is weighted. If weights are not specified then the graph is called unweighted and all edges are presumed to have a weight of one.
- Thus we have four types of graphs corresponding to the combination of directed/undirected (2 choices) and weighted/unweighted (2 choices).
- Fig 1 shows weighted undirected graph and Fig 2 unweighted directed graph.

Fig 1

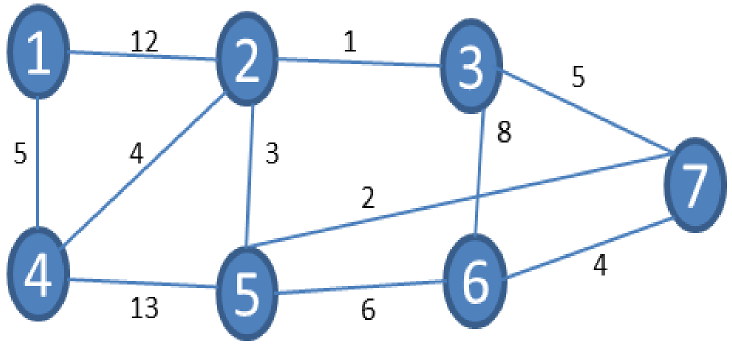
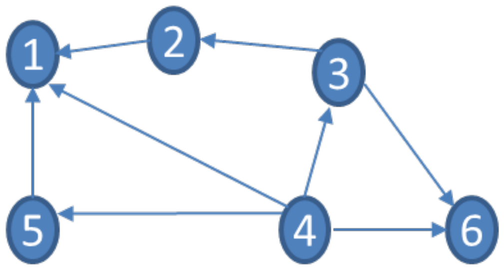


Fig 2



Types of Graphs

- In Fig 1 each vertex can denote a city. An edge may denote a 2-way highway. The weight can any one of: distance, cost, time, gas consumed etc.. Here the questions one might ask are w.r.t. reachability and the cost of the path.
- In Fig 2 each vertex can denote an intersection point of roads in a city where intersection points are connected via one way streets. Here the questions one might ask are w.r.t. reachability and the number of hops.

Fig 1

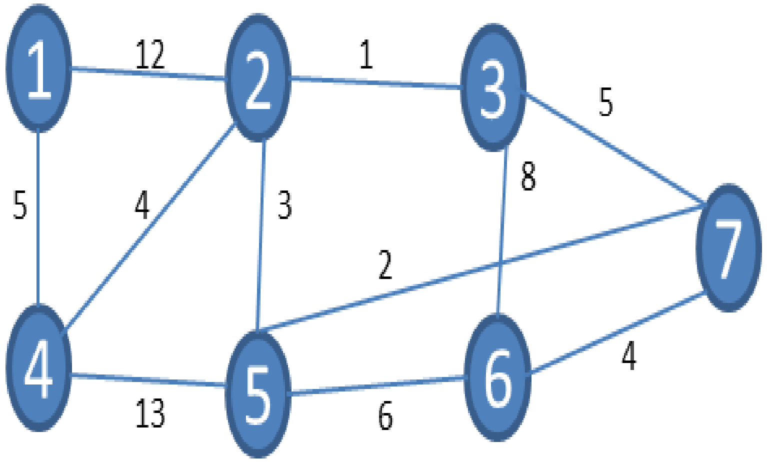
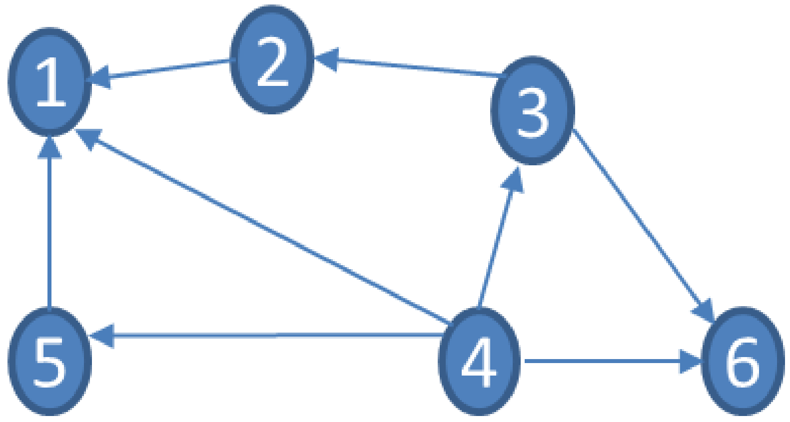


Fig 2



Definitions

- Path: A path exist from a vertex u to a vertex v if there is a sequence of vertices (i_0, i_1, i_2, i_3, …i_k) where u=i_1 and v=i_k and for all j, (i_j, i_{j+1}) $\in$E.
  - In an unweighted graph the path length is *k* (k edges are traversed).
    - In Fig2 the path length (3,2,1) is 2.
  - In a weighted graph the path length equals $\sum_{j=0}^{k-1} w(i_j,\ i_{j+1})$.
    - In Fig1 the path length of (4, 2, 5) is 7.

- Distance from u to v is defined as the minimum value of the path length of any path from u to v.
    - In Fig2 distance from 3 to 1 is 2 corr. to the path (3,2,1).
    - In Fig 1 distance from 3 to 4 is 5 corr. to the path (3,2,4).

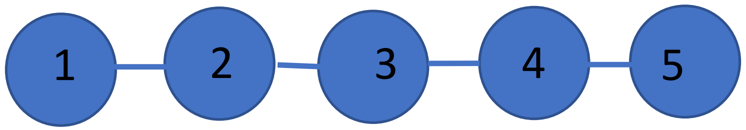
Definitions

- A path is simple if no intermediate vertex can repeat. That is, $\{i_0, i_1, i_2, i_3, \ldots i_{k-1}\}$ are unique.
- Self-loop is an edge from a vertex to itself.
- A graph is simple if it has at most one directed edge from u to v or at most one undirected edge between u and v. All graphs are presumed to be simple and self-loop free unless specified otherwise.
- Cycle: If there is a simple path from u to u with at least 2 intermediate vertices (undirected) then you have a cycle in the graph. For directed graphs you need at least one intermediate vertex.

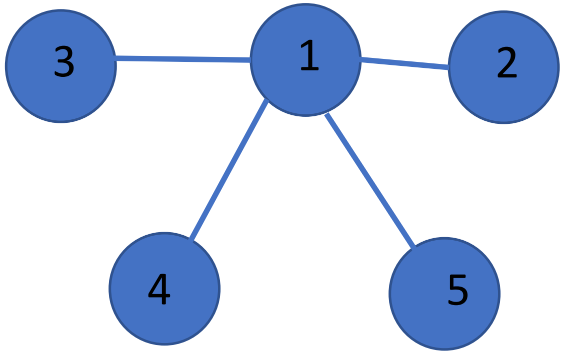
Definitions: Undirected Graphs

- Types of Undirected Graphs. Cyclic or Acyclic (cycle free).
- Connected component: It is the maximal subgraph of given graph such that every vertex can be reached from every other vertex. Note that in an undirected graph u can reach v implies the other way around. If the entire graph is one connected component then the graph is **connected**.
- Tree: An acyclic connected graph is also called as a tree.
  - If a unique vertex is designated as a root then we obtain a rooted tree otherwise the tree is unrooted
- Types of trees: Path tree with n vertices P_n, Star tree with one central vertex and n vertices connected to it with their corresponding edges: $K_{1,n}$ etc.

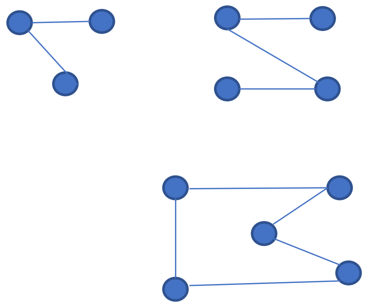
P_5



$K_{1,4}$



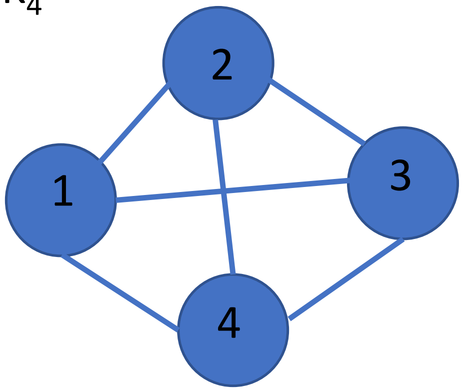
G with 3 connected components



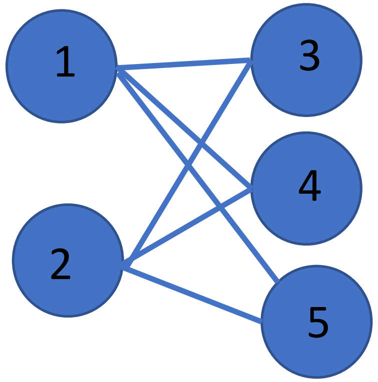
Definitions: Undirected Graphs

- Bipartite graph: If the vertex set V can be partitioned into 2 sets V_1 and V_2 ($V= V_1 \cup V_2$ and $V_1 \cap V_2 = \phi$) where every edge has endpoints in different sets then we have a bipartite graph. One example would be vertices consisting of the set of classrooms and the set of courses and the course to classroom assignments denoting the edges.
- Complete graph on n vertices: K_n. An edge exists between every pair of vertices.
- Complete bipartite graph consisting of two sets with m and n vertices each: $K_{m,n}$. A vertex in one set has an edge to every vertex in the other set.

K_4



$K_{2,3}$



Graph algorithms

- Graph Explorations
- Distance computations
- Computation of Minimum spanning trees
- Graph Exploration: Visiting all vertices of G in a systematic way.

BFS: Breadth First Search

- Breadth First Search is a process of systematically visiting vertices that are closer to a source vertex before visiting any vertices that are farther away.
- BFS mimics a radially expanding process.
  - E.g. sun rays,
  - ripples in the water,
  - forest fire,
  - expansion of bee swamp (closest gets bitten first),
  - spread of smoke,
  - spread of microbes among farm animals, (herd is relatively stationary).
  - flow of lawa etc.
  - aroma of a dish

BFS: Breadth First Search

- Breadth First Search is a process of systematically visiting vertices that are closer to a source vertex before visiting any vertices that are farther away.
- BFS mimics a radially expanding process.
  - E.g. sun rays, source: sun.
  - ripples in the water, source: the location where ripples started.
  - forest fire, source: the location where fire started.
  - expansion of bee swamp (closest gets bitten first), source: the beehive.
  - spread of smoke, source: the location where smoke started.
  - spread of microbes among farm animals, source: initially infected animal. We assume that herd is relatively stationary.
  - flow of lawa etc. source: location of volcano.
  - aroma of a dish. source: dish.

DFS

- An entity traversing unexplored paths and marking off the paths that are already traversed.

E.g. A mouse trying to escape a maze. Human trying to exit a building