Algorithms

Graph Theory

Definitions

- A **graph** G = (V,E) consists of a set of **vertices**, V, and a set of **edges**, E.
- Each edge in E joins 2 different vertices of V and is denoted by the tuple (i,j) where i and j are the 2 vertices joined by E.
- Some of the edges in a graph are oriented (i.e. they have arrowheads) whereas others are not.
- An edge with an orientation is a **directed** edge, whereas one with no orientation is an **undirected** edge. The undirected edges *(i,j)* and *(j,i)* are the same, but the directed edge *(i,j)* is different from the directed edge *(j,i)*.
- If all of the edges in a graph are undirected, then the graph is an **undirected graph**.

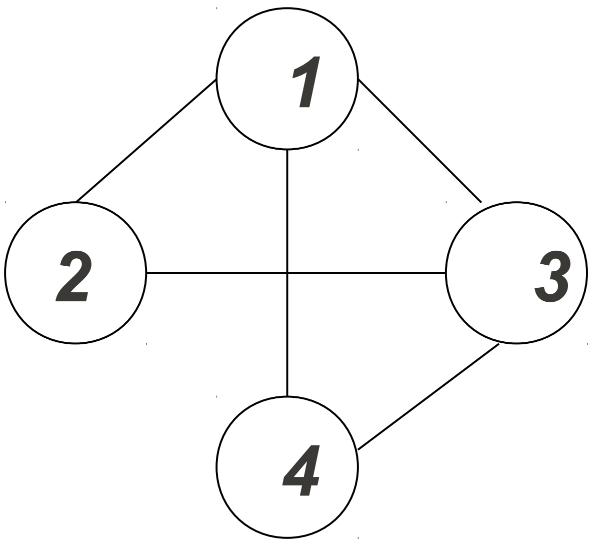
Definitions

- If all of the edges in a graph are directed then the graph is a **directed graph**.
- A graph that has both directed and undirected edges is called a **mixed graph**.

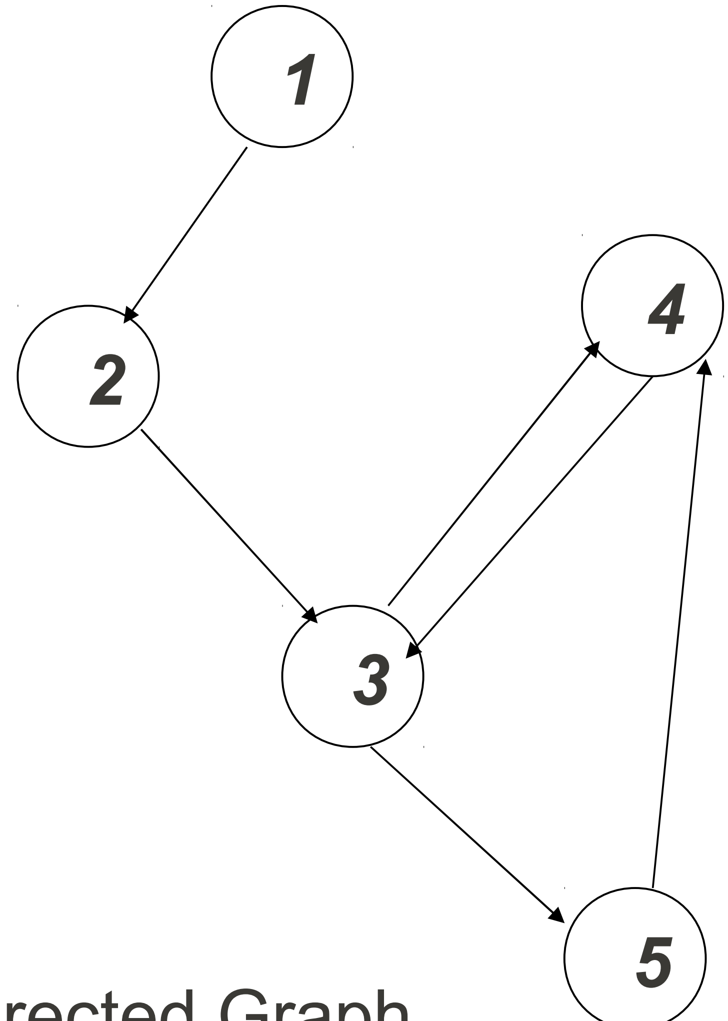
Example

A city map can be modelled by a graph whose vertices are intersections or dead-ends and whose edges are stretches of streets without intersections. This graph has both undirected edges, which correspond to 2 way streets and directed edges which correspond to one way streets. A graph modelling a city map is thus a mixed graph.

Definitions



Undirected Graph



Directed Graph

Definitions

- Vertices *i* and *j* are **adjacent** vertices iff (i,j) is an edge in the graph. The edge (i,j) is **incident** on the vertices i and j.
- In a directed graph, the directed edge (i,j) is **incident to** vertex j and **incident from** vertex i.
- Vertex i is **adjacent to** vertex j, and vertex j is **adjacent from** vertex i.
- By definition, a graph does not contain multiple copies of the same edge. Therefore, an undirected graph can have at most one edge between any pairs of vertices, and a directed graph can have at most one edge from vertex i to vertex j and one from j to i.
- A graph cannot contain any **self-edges**; that is edges of the form (i,i) are not permitted. A self edge is also called a **loop**.

Definitions

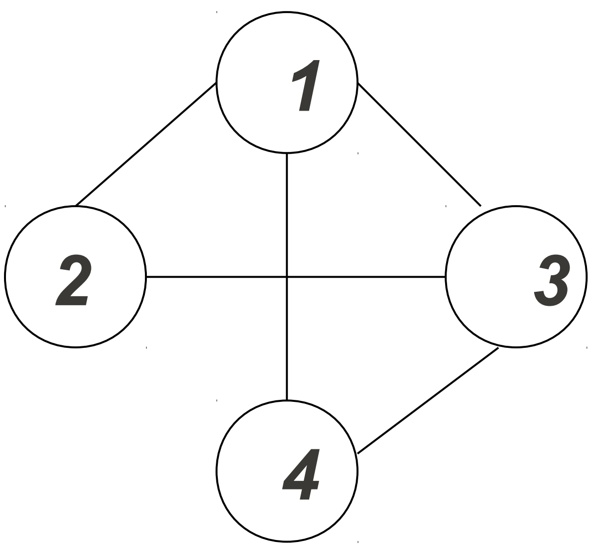
- In the undirected graph shown previously, the following vertices are adjacent
  - 1 and 2, 1 and 3, 1 and 4, 2 and 3, 3 and 4
- Also in this graph, the edge (1,2) is incident on vertices 1 and 2 and the edge (2,3) is incident on vertices 2 and 3 etc.
- In the directed graph shown previously, the following vertices are adjacent
  - 1 is adjacent to 2, 2 is adjacent from 1
  - 3 is adjacent to and from 4, 4 is adjacent to and from 3
  - 3 is adjacent to 5, 5 is adjacent from 3
- Also in this graph, the edge (1,2) is incident from 1 and incident to 2 and the edge (3,4) is incident from 3 and incident to 4

Definitions

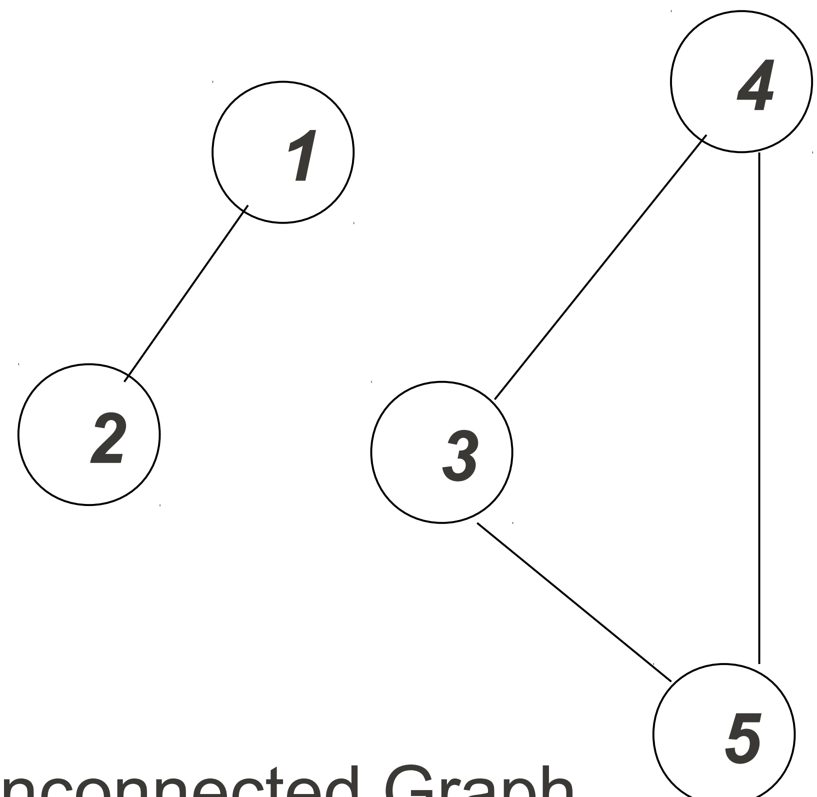
- A **path** of a graph is a sequence of alternating vertices and edges that starts at a vertex and ends at a vertex such that each edge is incident to its predecessor and successor vertex.
- A **cycle** is a path such that its start and end vertices are the same.
- A path is **simple** if each vertex in the path is distinct and a cycle is simple if each vertex in the cycle is distinct except the first and last one.
- A **subgraph** of a graph G, is a graph H whose vertices and edges are subsets of the vertices and edges of G respectively.
- A graph is **connected** iff there is a path between every pair of vertices in G.

Definitions

- A connected undirected graph that contains no cycles is a **tree**.
- In some applications we will assign a **weight** or cost to each edge e.g. in a map the weight might be the distance between 2 towns.



Connected Graph



Unconnected Graph

Representation of Graphs

The most frequently used representation schemes for unweighted graphs are adjacency based: adjacency matrices, array adjacency lists and linked adjacency lists.

Weighted graphs are generally represented with schemes that are simple extensions of those for unweighted graphs.

Adjacency Matrices

Adjacency Matrix

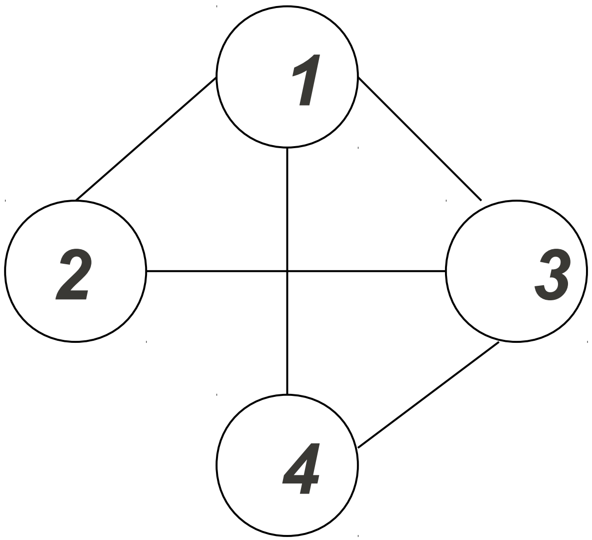
The adjacency matrix of an n-vertex graph G = (V,E) is an n x n matrix A. Each element of A is either 0 or 1. If G is an undirected graph, then the elements of A are defined as follows:

$$A(i,j) = \begin{cases} 1 & \text{if } (i,j) \ \varepsilon\ E \text{ or } (j,i) \ \varepsilon\ E \\ 0 & \text{otherwise} \end{cases}$$

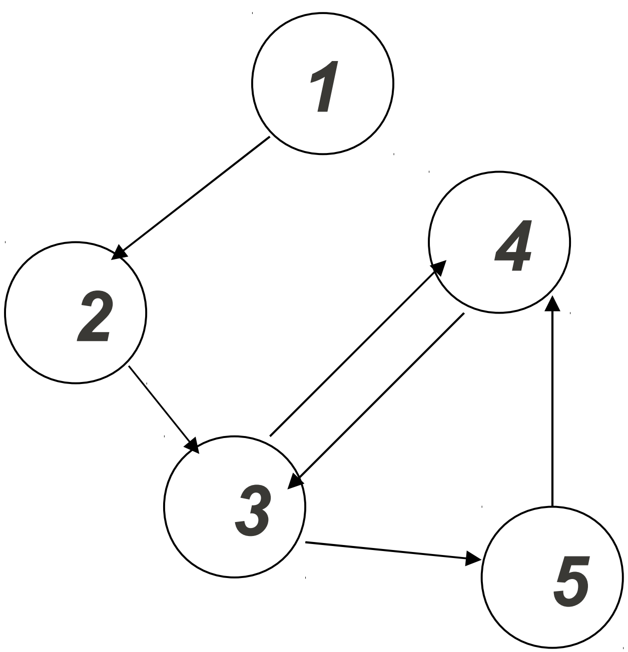
If G is a directed graph, then the elements of A are defined as follows:

$$A(i,j) = \begin{cases} 1 & \text{if } (i,j) \ \varepsilon\ E \\ 0 & \text{otherwise} \end{cases}$$

Adjacency Matrices



$$\begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \end{array}$$



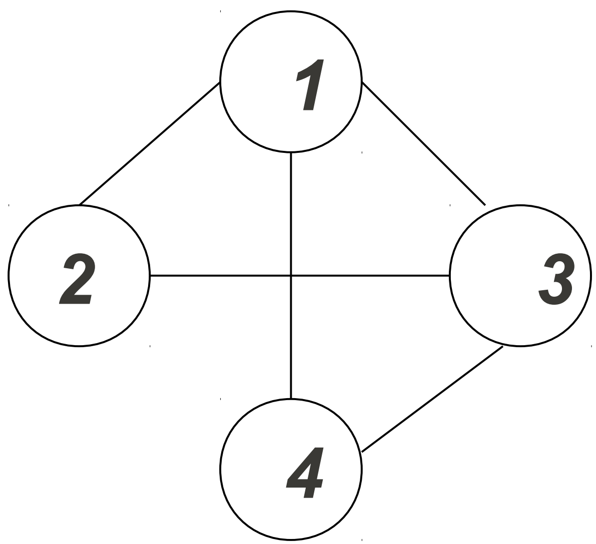
$$\begin{array}{c c} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

Adjacency Lists

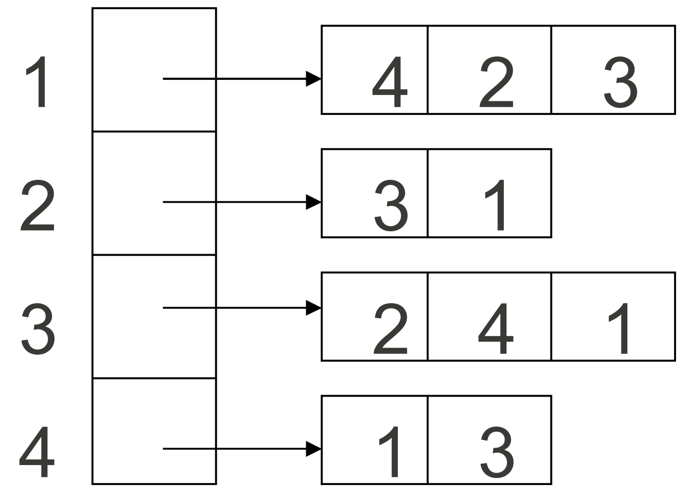
Adjacency Lists

- The adjacency list for vertex i is a linear list that includes all vertices adjacent from i.
- In an adjacency list representation of a graph, we maintain an adjacency list for each vertex of the graph.
- When these adjacency lists are represented as chains, we get the **linked adjacency list** representation.
- When the adjacency lists are represented as an array based linear list, we get the **array adjacency list** representation.

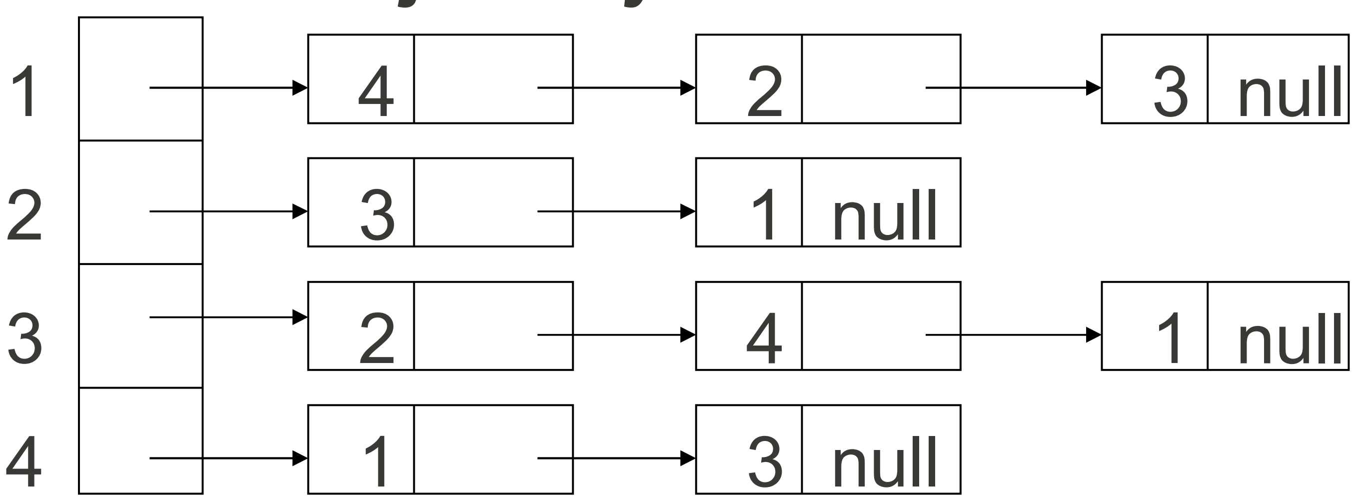
Adjacency Lists



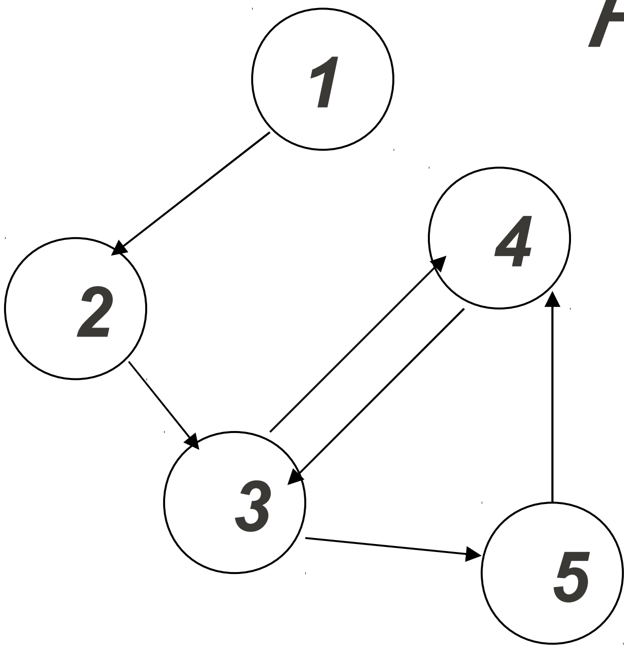
Array Adjacency List



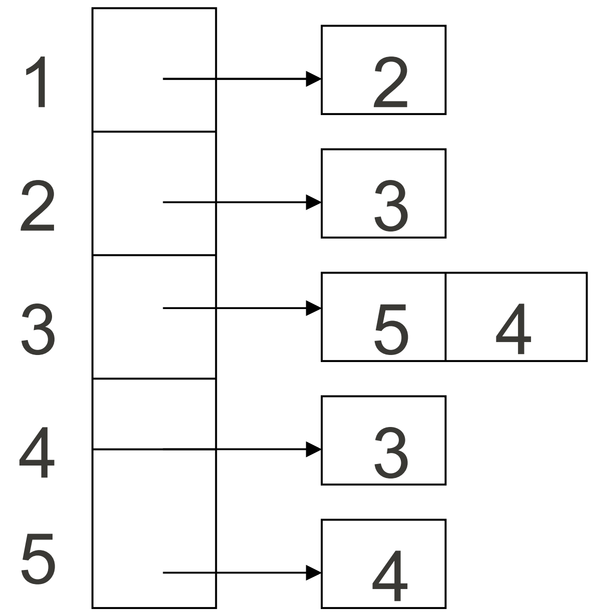
Linked Adjacency List



Adjacency Lists



Array Adjacency List



Linked Adjacency List