COMP H2026
Information Technology
Mathematics

Graphs

Lecture Outline

- Graph Theory
  - Graphs & their basic properties
    - their representation
    - Paths
  - Graph Traversals
  - Will try investigate some practical problems in which they can be applied

  - Overall want to learn about graphs (& later Trees) and how they model structures in computing

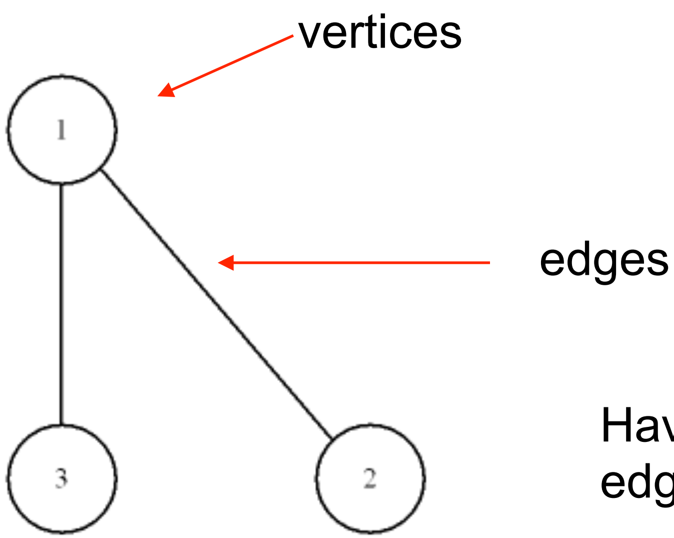
Graph Theory

- In maths and computing *graph theory* is the study of graphs
  - graphs are a mathematical structure used to model pairwise relations between object from a certain collection (not graphs drawn with *x* and *y* axes!)
  - represented by a structure consisting of points (called ‘vertices’), some of which may be joined to other vertices by lines (called ‘edges’) to form a network
- Structures of this type are used in computing
  - e.g. the computers on a site may be connected into a local area network, which in turn may be linked to a wider network, including the internet.
  - the circuitry inside a computer is another example of a graph structure

example

Graphs - basic properties

- ‘A graph consists of a non-empty set of points, called *vertices* (singular: *vertex*) and a set of lines, called *edges*, such that every edge is attached at each end to a vertex.’ Grossman, chpt. 10
- e.g.



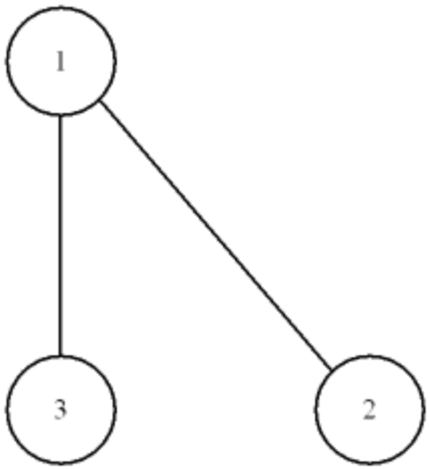
Have vertices 1, 2, and 3 with edges connecting 1 to 2 and 1 to 3

Representations of graphs

- From a computational point of view want to represent graphs as data
- In computer science a graph is an abstract data type that consists of a set of vertices and a set of edges that establish relationships (connections between the nodes. The graph data type follows directly from the graph concept from mathematics
- A graph G is defined as G = (V,E) where V is a set of vertices and E is a set of edges
  - In practice, information is associated with each vertex and edge

Graphs - basic properties

- Two vertices that are joined by an edge are *adjacent*

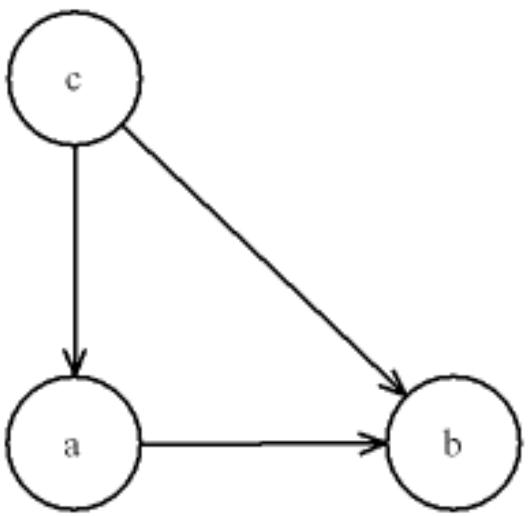


1 and 3 are adjacent but 2 and 3 are not…

- Graphs can arise in many practical situations – a graph could represent a road system in which they edges are the roads and the vertices are the towns. Or the vertices may represent computer labs in a Local Area Network (LAN) with the edges representing the links in the network.

Graphs - basic properties

- Directed graph – a graph where each edge points in one direction (using arrows to show the direction)



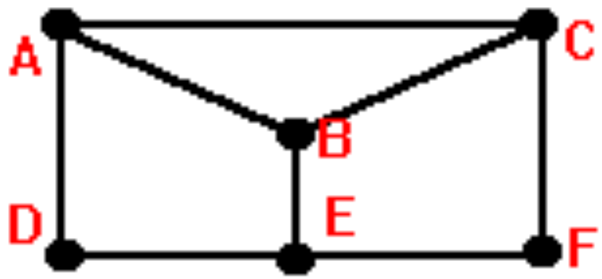
Directed graph with 3 vertices a, b, and c and edges from c to a, c to b and a to b

Graphs - basic properties

- When we draw a graph all that matter is which vertices are adjacent (joined by an edge and which are not; the precise location of the vertices and the lengths and shapes of the edges are irrelevant.
  - There can be many different ways of drawing the same graph some of which can look very different from others!

Graphs - basic properties

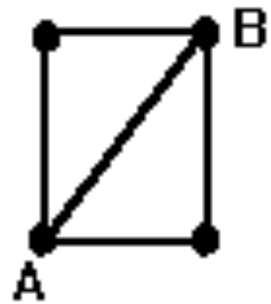
- The degree of a vertex is the number of edges it touches



In this example: vertices A, B, C and E have degree 3

vertices D and F have degree 2

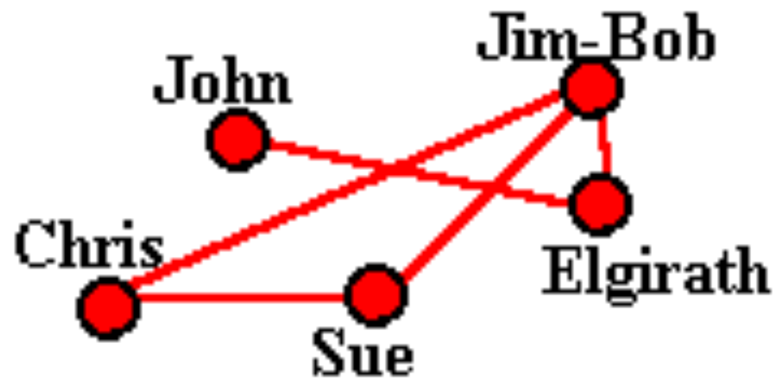
- In the graph below the vertices A and B have the same degree, what is the degree?



Answer = ?

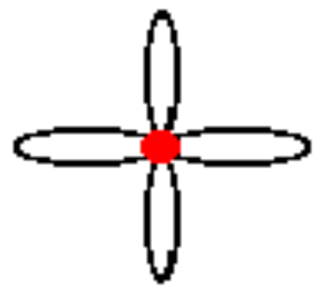
Graphs - basic properties

- Below is a graph showing the relationships between a set of school children. **The vertices are adjacent if the children they represent are friends.** Use this graph to answer TRUE or FALSE to the following questions:
  - Every child has at least one friend
  - Chris has the fewest friends
  - Jim-Bob has the most friends.



Graphs - basic properties

- A graph may have loops (an edge from one vertex to the same vertex)



- The degree of the red vertex in the above graph is 8

Paths

- Problems that use graphs often involves moving from one vertex to another along a sequence of edges, where each edge shares a vertex with the next edge in the sequence
- A *path* of length *n* in a graph is a sequence of vertices and edges of the form: v_0, e_1, v_1, e_2, v_2, …, e_n, v_n
  - Where e_i is an edge joining v_{i-1} and v_i for all i elements of {1,2,…,n}

- A path can be thought of as a sequence of vertices and edges that can be traced without lifting your pen off the paper. A path may include repeat edges of vertices.
- The length of a path is the number of edges in the path
- A circuit is a path that starts and ends at the sane vertex $v_0 = v_n$

Paths

- Consider the following graph

- In this graph
  - A, B, D, E is a path of length 3
  - C, D, B, C, D is a path of length 4
  - E, D, B, C, D, E is a circuit of length 5

- Connected graph: a graph is connected if there is a path between every pair of vertices
- Any path that contains each edge of a graph exactly once is called a *Euler path*

Graph Traversals

- There are many different applications of graphs – we need different algorithms for manipulating them
- graph algorithms systematically visit all the vertices in the graph.
  - i.e., the algorithm walks through the graph data structure and performs some computation at each vertex in the graph.
- This process of walking through the graph is called a *graph traversal*
- Graph traversal: starts at some vertex v and visits all vertices x that can be reached from v by travelling along some path from v to x, if a vertex has already been visited it is not visited again

Graph Traversals

- Two popular traversals algorithms are *Breadth-first* and *Depth-first*
- Breadth-first traversal - specify the vertex at which to start the traversal
  - breadth-first traversal first visits the starting vertex, then all the vertices adjacent to the starting vertex, and the all the vertices adjacent to those, and so on
  - Procedure visit(v, k) = visit every vertex x not yet visited for which there is a length k path from v to x
  - If the graph has n vertices the breadth-first traversal starting at v can be

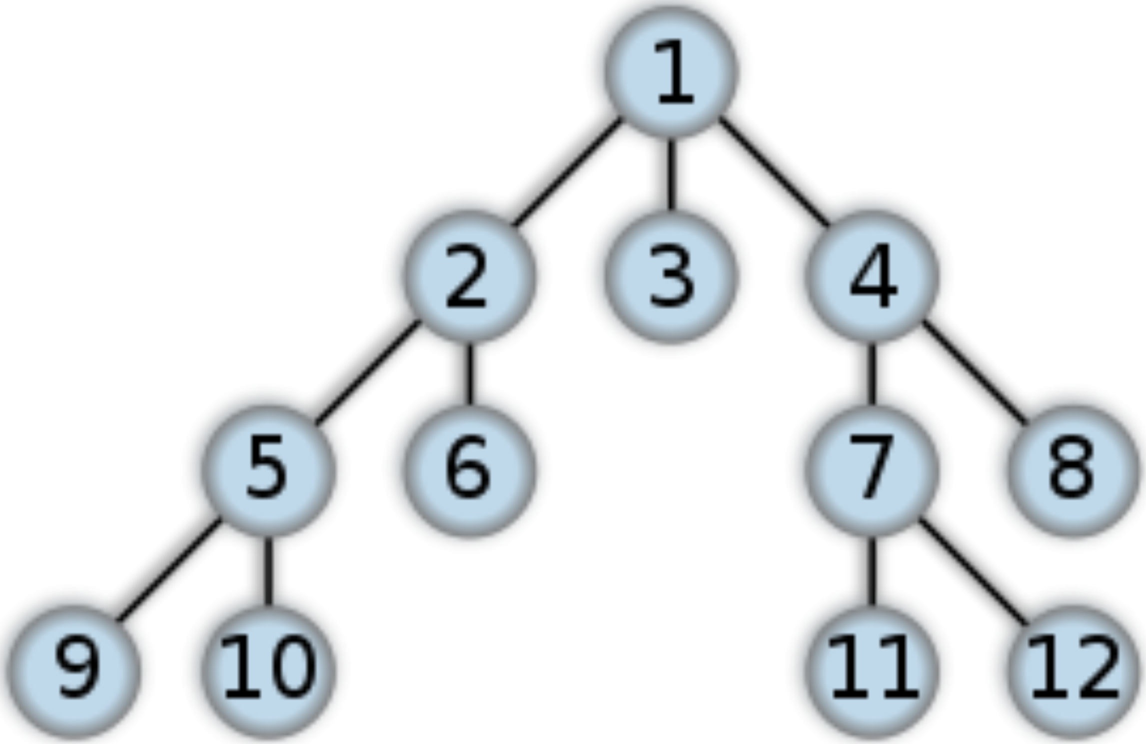
    for k = 0 to n do

    visit (v, k)

There are usually several different traversals from any given starting vertex

Breadth-first traversal

Example

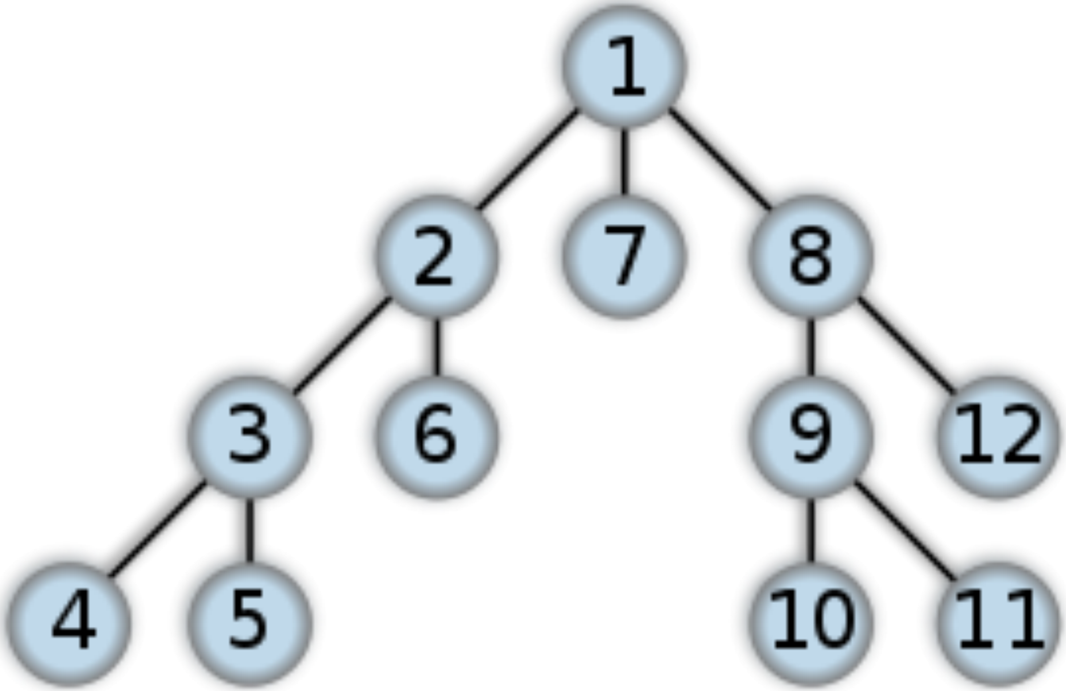


Depth-first traversal

- In depth-first traversal - must specify the vertex at which to begin
- Depth-first traversal of a graph visits a vertex and then recursively visits all the vertices adjacent to that node
- The catch is that the graph may contain *cycles*, but the traversal must visit every vertex at most once.
  - *Cycles* - A path whose beginning & ending vertices are equal and in which no edge occurs more than once (acyclic – graph with no cycles)
- The solution to the problem is to keep track of the nodes that have been visited, so that the traversal does not suffer the fate of infinite recursion.
- A depth-first traversal only follows edges that lead to unvisited vertices

Depth-first traversal

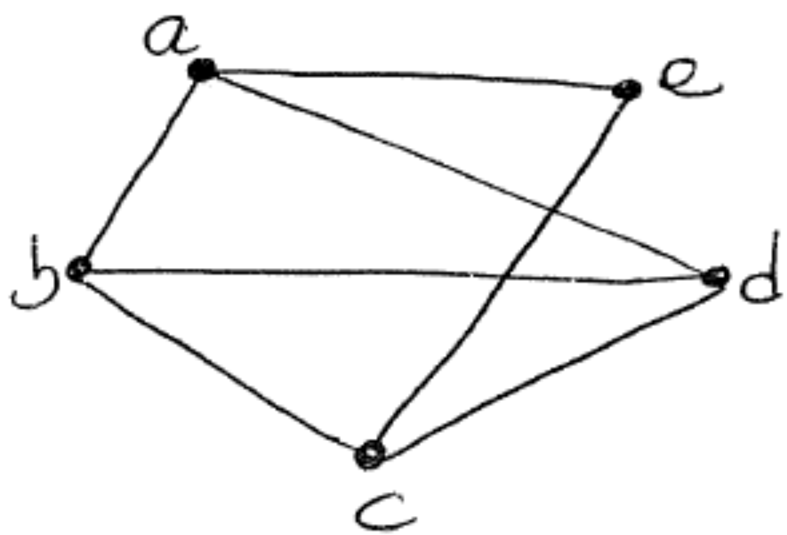
Example



Representations of a graph

- In typical graph implementation the vertices are implemented as structures or objects
- Adjacency Matrix: if G is a graph with n vetices, labelled $v_1, v_2, \ldots, v_n$ then the adjacency matrix of G is an n x n matrix whose entries are given by the following rule:
  - The entry in the ith row and jth column is the number of edges from v_i to v_j

Example (on board)



Representations of a graph

- Adjacency list: associates each vertex with an array of incident edges
  - These arrays can simply be pointers to other vertices

Summary

- Graphs used to represent data and information
- Graphs represented graphically by a dot for every vertex and drawing an arc between two vertices
  - Use arc if connected
  - Use arrow if a directed graph
- Can follow paths through a graph - a sequence of vertices and edges that can be traced where the length is the number of edges in a path
- Graph structure can be manipulated using graph traversal
  - Breadth-first
  - Depth- first
- Matrix representation of graph

Exercise

- Research (lookup and give me a description of) the "Travelling Salesman Problem' (classic computer science problem)
- What is Dijkstra's algorithm?

Graph Theory

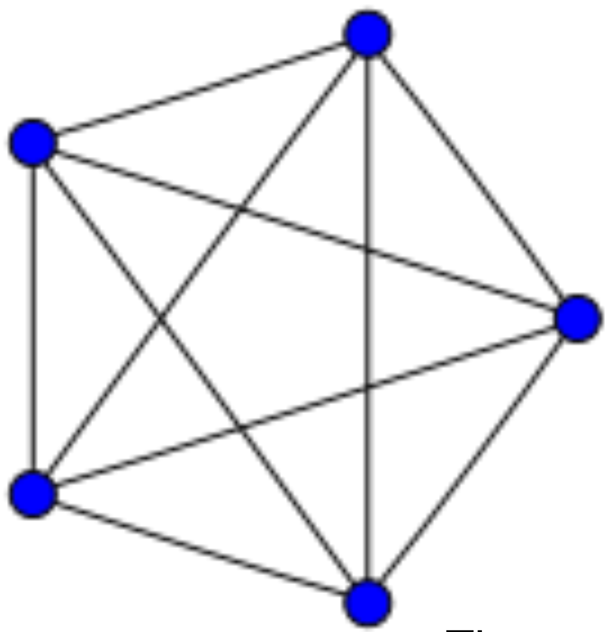
- The degree of a vertex *v* in a graph is the number of edges incident to *v*, with a loop counting 2 towards the degree of the vertex to which it is incident
- Can denote the degree by *deg(v)*
  - Example (on board)
  - A vertex with degree 0 is isolated
- There is a relationship between the degrees of the vertices of a graph and the number of edges it has…
  - In any graph, the sum of the degrees of the vertices equals twice the number of edges (Grossman, Chpt. 10, Theorem pg. 178)
  - i.e. if V and E denote respectively the set of vertices and the set of edges of a graph then we can write:

$$\sum_{v \in V} \deg(v) = 2|E|$$

  - Example (on board)

Graph Theory

- A graph is *complete* if each vertex of the graph is adjacent to every other vertex
- Example (on board) – complete graph with five vertices



The complete graph on n vertices has $n(n-1)/2$ edges, and is denoted by K_n

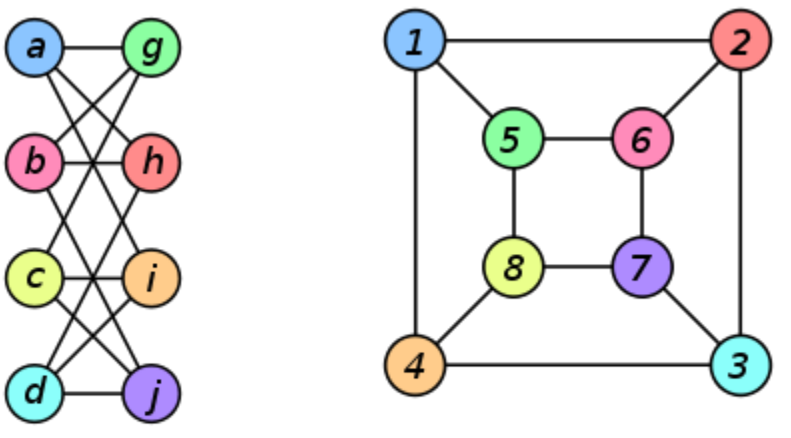
Isomorphism of Graphs

- Can describe two graphs as being isomorphic if one of them can be moved around until it looks like the other (subject to certain rules such as not breaking edges)
  - Vague definition
  - Example (on board)

- More precise way to describe isomorphism of graphs is to define *isomorphic* as the idea of matching up the corresponding vertices in two graphs
  - i.e. If two graphs are isomorphic then we can associate with each vertex of one graph a corresponding vertex in the other graph

Isomorphism

- $f(a) = 1$
- $f(b) = 6$
- $f(c) = 8$
- $f(d) = 3$
- $f(g) = 5$
- $f(h) = 2$
- $f(i) = 4$
- $f(j) = 7$



Isomorphism of Graphs

- Given the graphs *G* and *H*
- Let *V(G)* and *V(H)* denote the vertex set of *G* and *H* respectively
- If *G* and *H* are isomorphic then can show this by associating with each element of *V(G)* the corresponding element of *V(H)*
- i.e. an isomorphism from *G* to *H* is a function

  $f: V(G) \rightarrow V(H)$
- With the properties
  - *f* is one-to-one and onto
  - For any two vertices *u* and *v* of *G*, if *u* and *v* are adjacent in *G*, then *f(u)* and *f(v)* are adjacent in *H* and if *u* and *v* are not adjacent in *G*, then *f(u)* and *f(v)* are not adjacent in *H*

If there exists an isomorphism from G to H, then G and H are isomorphic (Grossman, pg 183)

Example (on board)

Isomorphism of graphs

- How can we show that two graphs G and H are not isomorphic?
  - Impractical to test all functions from V(G) to V(H)
- Find a graph-theoretic property that one graph has but the other does not
  - graph-theoretic property – a property that is retained if the graph is rearranged (as we did in our first look at isomorphism)
  - Graphs with different numbers of edges or vertices cannot be isomorphic
  - Useful to look at the degrees of vertices, e.g. if one graph has two vertices with degree 3 and the other has only one, then the graphs are not isomorphic….
  - Example (on board)

Hamiltonian Path

- A Hamiltonian Path is a path that includes every vertex of G exactly once except only the first and the last vertices may coincide.
- A Hamiltonian circuit is a Hamiltonian path for which the first and the last vertices coincide
  - Need not use all of the edges of a graph

Scifest 2013:

- http://newsroom.intel.com/community/en_ie/blog/2013/11/29/st-paul-s-college-raheny-student-picks-up-top-prize-at-scifest-2013