3 Graphs

We have already seen examples of directed graphs, used to illustrate relations.

Definition 3.1 *A directed graph consists of a set V of vertices, and a set E of directed edges each of which joins a vertex of V to a vertex of V. If e is a directed edge in E, then the vertex at which e starts is called its initial vertex, and the vertex at which it ends its terminal vertex; the two vertices may coincide (in which case the edge is called a loop). Two distinct edges cannot share both their initial and their end vertices.*

The last condition in the definition is another way of saying that E is a subset of $V \times V$; so really a directed graph is just a binary relation on its vertex set. Of course we represent it pictorially, and then represent an edge as an arc with an arrow on it. Where two vertices are joined by directed edges running in both directions, we often draw just one arc, without an arrow, instead of two directed ones. We often abbreviate 'directed graph' to 'digraph'.

Examples.

A digraph is completely defined by the two sets V and E, and not by the way the vertices and directed edges are shown in the diagram; two different diagrams may look quite different.

Sometimes we don't want to specify directions on the edges.

Definition 3.2 *Graph*

Examples, including complete graphs.

Every digraph is associated with a graph, formed by ignoring the directions on its edges.

And sometimes we want to allow multiple edges between pairs of vertices.

Definition 3.3 *Multigraph, directed multigraph.*

Definition 3.4 *We define a path (or length n) in a directed graph or graph to be a sequence of vertices $v_0, v_1, \ldots v_n$ such that for any i, (v_i, v_{i+1}) is a (directed) edge. Such a path is called a circuit (of length n) if $v_0 = v_n$, and a cycle if $v_0 = v_n$ and all of $v_0, \ldots v_{n-1}$ are distinct. The length is the number of edges in the path/circuit. A graph is said to be connected if any two vertices x, y are connected by a path; a directed graph is said to be connected if its underlying graph is connected.*

Definition 3.5 *Subgraphs, components.*

Definition 3.6 *A bipartite graph is one whose vertex set V can be written as a disjoint union of two subsets V_1 and V_2 in such a way that any edge in the graph joins a vertex in V_1 to a vertex in V_2. More generally a k-partite graph is one whose vertex set V can be written as a disjoint union of k subsets $V_1, \ldots V_k$ in such a way that every edge joins two vertices in different subsets.*

An alternative way of phrasing this is to say that a graph is k-partite if the vertices can be coloured with k colours in such a way that any edge joins two vertices of different colours.

It can be proved that any graph which can be drawn on a piece of paper with no edges crossing can be coloured with 4 colours (and so is 4-partite). This is the famous 'four colour theorem'.

It is easy to check whether or not a graph is bipartite (2-partite). A graph is bipartite if and only all its cycles have even length.

In computer science, certain graphs, known as trees are important.

Definition 3.7 *A tree is a connected graph without any cycles. It may or not have directions assigned to its edges. A vertex with just one edge through it is called a leaf. Sometimes one vertex is selected to be called the root of the tree; then the tree is known as a rooted tree. In that case, the tree is called a binary tree if every non-leaf is joined to 2 vertices which are further from the root than it. If x is a vertex in a rooted tree, then the vertex joined to x and nearer to the root than x is called its parent, and the other vertices joined to x (and further from the root than x) are called its children.*

Some diagrams of examples.

Rooted trees are used to define directory stucture.

3.8 *Parsing trees*

Trees are also used for parsing, e.g. of arithmetic expressions. Then often we use rooted trees in which each vertex has either one of two children. Examples, e.g. of $x + (x * (y + z) + y * (x - z))$. For the purposes of this course we consider that any arithmetic expression is formed from a combination of binary operations ($*$, $+$ and $-$) and unary operations ^{n}, for any integer n) applied to a set of constants and variables. We form a parsing tree as follows. Each leaf is labelled by a constant or variable, and each non-leaf by an operation. A non-leaf is then also associated with the sub-expression which is formed by applying the operation which labels that vertex to the one or two (maximal) subtrees whose root(s) is/are the children of that vertex.

Definition 3.9 *A spanning tree for a graph is a subgraph, involving all the vertices, which is also a tree.*