2

Graphs

Some Basic Terminology

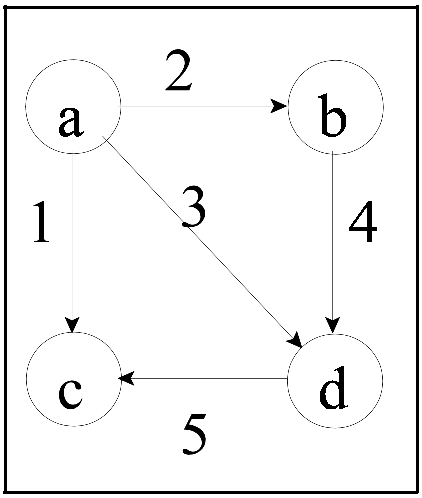

Figure 1 A Connected Graph

We are concerned with *graphs*. Every graph contains objects that are known as *nodes* or *vertices* depending on the mood of the speaker. A graph always contains at least one node. A graph usually also contains lines that go between nodes. These lines are known as *arcs* or *edges*, again depending on the mood of the speaker. In this book arcs will almost always be *directed*. That is, each arc will leave a vertex and go to a vertex. Hence, the representation of an arc is most like an arrow. An arc from a node to itself is sometimes referred to as a *loop*. Furthermore, graphs frequently have labels attached to the arcs and/or the vertices. The nature and purpose of the labels is determined by the user.

Since our arcs are directed from one vertex to another (which may be the same vertex) we can speak of transversing the arc *with* or *against* the direction of the arc. A graph is said to be *connected*, if it is possible to travel from any vertex to any other vertex of the graph by transversing the arcs with the understanding that an arc can be transversed against its direction. A graph that is not connected is said to be *disconnected*.

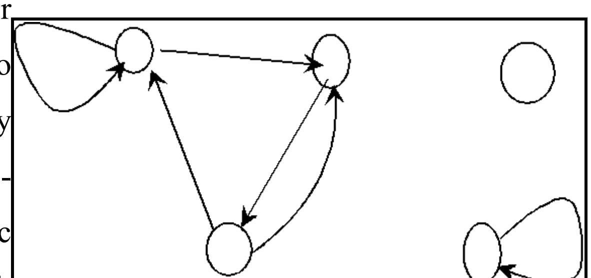

Figure 2 A Disconnected Graph.

Figure 1 shows a connected graph with labeled nodes and arcs. **Figure 2** shows a disconnected graph. Note that one of the vertices in **Figure 2** has no incident arcs (that is it has no arcs going to or from it).

Suppose that it is possible to go from vertex A to Vertex B via a sequence of arcs a_1, a_2, a_3, ..., a_n where each arc a_i meets arc a_{i+1} at some vertex. Then the sequence of arcs is known as a *path* from A to B. If each arc is transversed in the proper direction, the path is said to be a *directed* path. A path from a vertex to itself is called a *circuit*. A graph without circuits is called a *forest*, and a connected forest is a *tree*. **Figure 3** shows a forest. Each connected subgraph[1] is a tree. Whereas in real life a forest is a bunch of trees, in Graph Theory, a tree is a minimal forest. In a graph, if there is a directed path from any vertex to any other, it is *strongly connected*. A directed path from a vertex to itself is called a *directed circuit*. A nice way of characterizing a tree is that in a tree, there is exactly one path from any node to any other node.

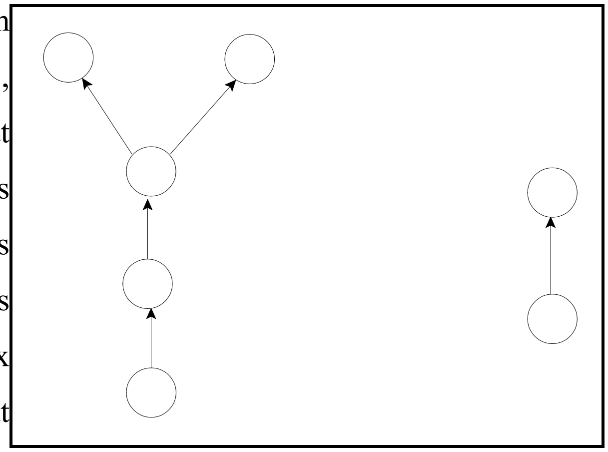

Figure 3 A Forest

Answers to <u>all</u> exercises are given in the back of the book.

☐ **Exercise 1** Prove the above assertion: A connected graph is a tree if and only if given any two nodes A and B, there is exactly one path from A to B (the direction of an arc can be ignored). This problem like this whole chapter is a little abstract. Feel free to go to the next chapter.

[1]A subgraph of a graph is a graph made up of some of the arcs and nodes of the original graph and no arcs or nodes that are not part of the original graph. Since the subgraph is a graph, if it contains an arc it must contain both end nodes from the original graph; that is, an arc must always go between nodes.

A particularly important type of graph to this text is the *probability graph*. It is a graph with numeric labels on each arc. These numbers are all positive and the arcs leaving a particular node must have labels that add up to one. **Figure 4** is a probability graph. In a probability graph an arc from A to B is labeled with x, then x represents the probability of going to B starting from A. That is why the arcs leaving a node must (have labels that) add up to one.

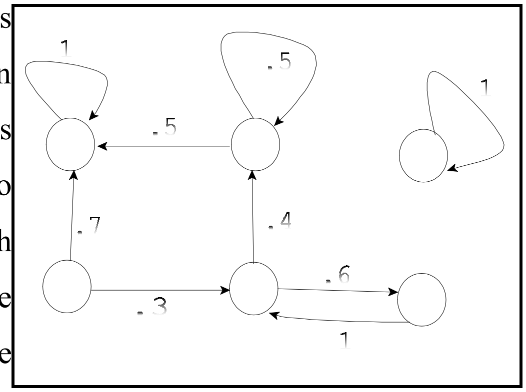

Figure 4 A Probability Graph

Relations and Graphs

To mathematicians, arcs are *binary relations* on nodes. That is because each arc can be considered as merely an ordered pair of nodes. For example, the arc from vertex A to vertex B can be represented as the ordered pair (A, B). Again, the pair is said to be *ordered* because the pair (B, A) would represent a different arc: the arc from B to A. If each node has no more than one arc leaving it, the binary relation is in fact a *function*. The relation is said to be *binary* because it is defined on pairs of objects (which is the only type of relation we discussed in the previous chapter).

Graphs of Equivalence Relations

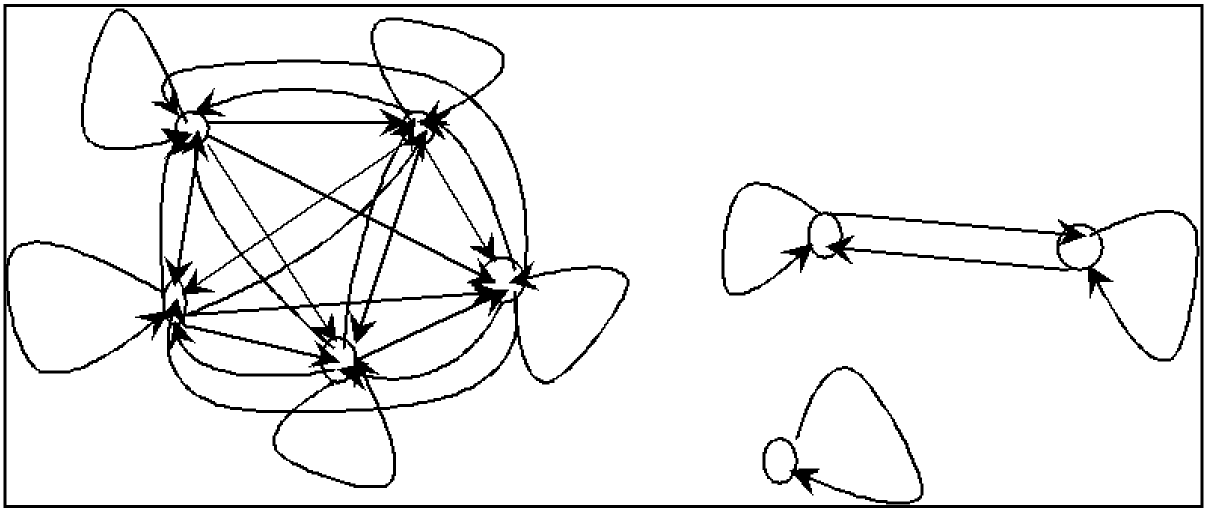

Figure 5 A Graph of an Equivalence Relation.

The arcs in **Figure 6** are an equivalence relation on the nodes of the graph. In this case there are three equivalence classes consisting of 5, 2, and 1 nodes respectively. Again, the equivalence relation given by the arcs partitions the graph into the three classes.

Graphs of Permutations

A (finite) graph shows a permutation of its nodes if there is precisely one arc leaving and entering each node. If there is an arc from node A to node B then we say that the permutation takes A to B. **Figure 6** represents a permutation on the first eight positive integers. It takes 1 to 2, 2 to 4, 4 to 3, 3 to 1, 5 to 5, 6 to 7, 7 to 8, and 8 to 6. If these numbers were passengers on a train, then when we perform a permutation 1 takes 2's seat and 2 takes 4's seat etc. If we keep performing the permutation 1 takes 2's seat then he takes 4's seat and then he takes 3's seat. If we perform the permutation four times then 1 winds up in his original seat, and 6 winds up in 7's original seat.

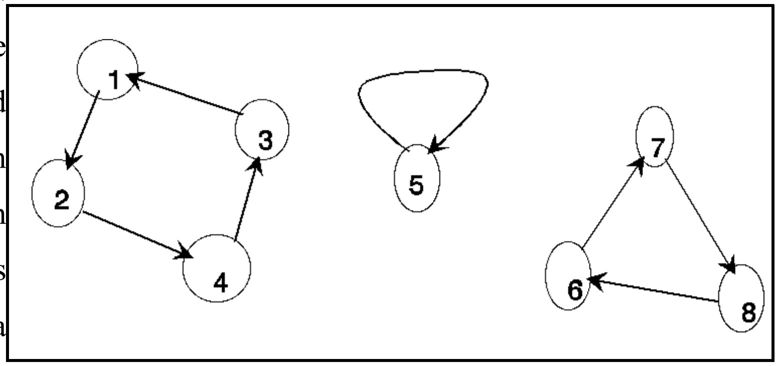

Figure 6 A Permutation Graph.

☐ **Exercise 2** (This is a theoretical problem; do not take it too seriously!) Why is the relation in **Figure 6** not an equivalence relation? Having answered that, notice that in **Figure 6** there is an obvious partition of the eight integers into three partition classes. Therefore, there must be an equivalence relation corresponding to the permutation. Define a new graph in terms of the graph in **Figure 6** that gives an equivalence relation. More generally, given an arbitrary permutation, define an equivalence relation based on it. (This may be the toughest problem in this whole text; feel free to go on!)

1. If there is no path from one vertex to another then the graph is not connected and it can't be a tree. Suppose now that the graph is connected and that there are two distinct paths from node A to node B. Starting at node A there must be a first node, call it C, where the two paths diverge (differ). This node might be A itself. Similarly, since both paths go to B, there must be a first node, call it D, where the two paths first meet again. This node could be B itself. The two nodes C and D along with the two different paths that connect them form a circuit. Hence, the graph is not a tree. (It might help you to fallow this proof if you draw a picture of the graph and the nodes A, B, C, and D.)

2. That we do not have an equivalence relation is because we do not satisfy that a~b implies b~a. For example, there is an arc form node 2 to node 4 but there is not an arc going the other direction. On the other hand we can easily see how to define an obvious partition of the 8 nodes. The classes would be {1,2,3,4}, {5}, and {6,7,8}. The graph of the corresponding equivalence relation would have an arc from each node in a class to each other node in that class. The corresponding relation x~y implies that y is *eventually* permuted to y if you keep permuting. That is the relation x~y implies that node x is permuted to a node that is permuted to a node etc until the sequence arrives at y.