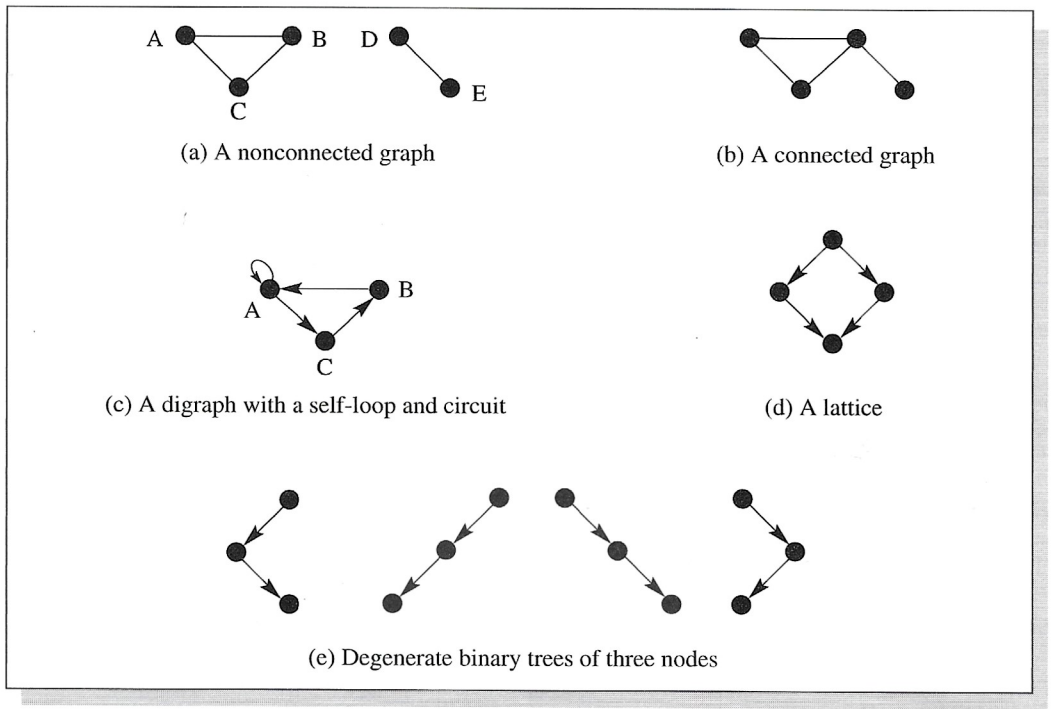
Figure 3.2 Simple Graphs



(a) A nonconnected graph

(b) A connected graph

(c) A digraph with a self-loop and circuit

(d) A lattice

(e) Degenerate binary trees of three nodes

net to another, which modifies the weights. A simple graph has no links that come immediately back on the node itself, as shown in Figure 3.2a. A **circuit** or **cycle** is a path through a graph that begins and ends on the same node, as does the path ABCA in Figure 3.2a An **acyclic graph** has no cycles. A **connected graph** has links to all its nodes as shown in Fig. 3.2b. A graph with directed links, called a **digraph,** and a **self-loop** is shown in Figure 3.2c. A directed acyclic graph is a **lattice**, and an example is shown in Figure 3.2d. A tree with only a single path from the root to its one leaf is a **degenerate tree**. The degenerate binary trees of three nodes are shown in Figure 3.2e. Generally in a tree, the arrows are *not* explicitly shown because they are assumed to be pointing down.

Trees and lattices are useful for classifying objects because of their hierarchical nature, with parents above children. An example is a family tree, which shows the relationships and ancestry of related people. Another application of trees and lattices is making decisions; these are called **decision trees** or **decision lattices** and are very common for simple reasoning. We will use the term **structure** to refer to both trees and lattices. A decision structure is both a knowledge representation scheme and a method of reasoning about its knowledge. An example of a decision tree for classifying animals is shown in Figure 3.3. This example is for the classic game of twenty questions. The nodes contain questions, the branches "yes" or "no" responses to the questions, and the leaves contain the guesses of what the animal is.