Graphs and Their Applications (5)

K.M. Koh*
Department of Mathematics
National University of Singapore, Singapore 117543

F.M. Dong and E.G. Tay
Mathematics and Mathematics Education
National Institute of Education
Nanyang Technological University, Singapore 637616

14 Trees and Their Properties

This is the first of a few articles in which we shall study a special and important family of graphs, called trees, discuss their properties and introduce some of their applications.

A **tree** is a connected graph which contains no cycle as a subgraph.

In Figure 14.1, the graph (a) is *not* a tree as it is disconnected; the graph (b) is connected but *not* a tree as it contains a cycle (for instance, $uvxyu$) as a subgraph; and the remaining ones are all trees (with special names).

*Corresponding author. Email: matkohkm@nus.edu.sg.

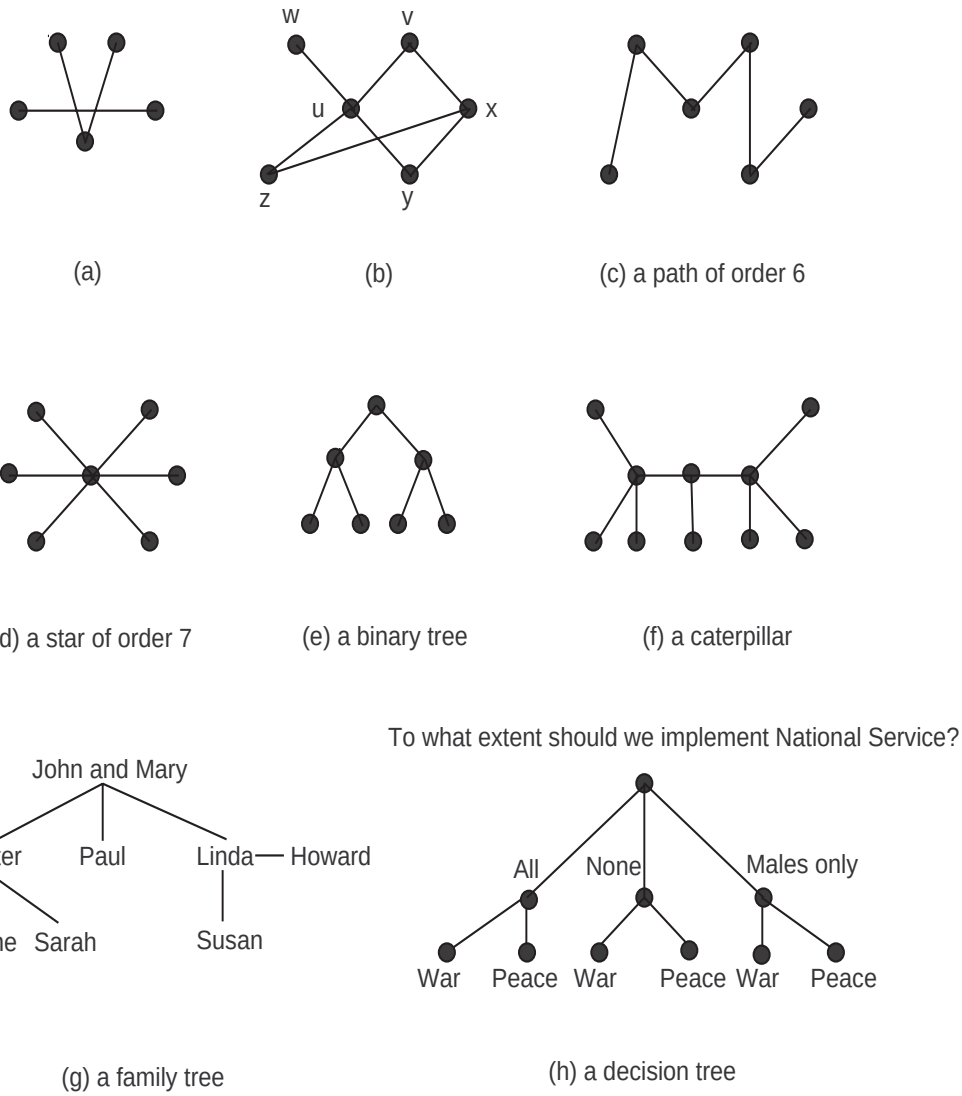

Figure 14.1

Look at the trees shown in Figure 14.1. As every tree is connected, it is not surprising that any two vertices in it are joined by a path; but a related feature of trees that other connected graphs do not have is the *uniqueness* of the existence of such a path between any two vertices. Indeed, we have:

Theorem 14.1 *Let G be a connected graph. Then G is a tree if and only if every two vertices in G are joined by a unique path.*

Proof. [Necessity] Suppose that G is a tree and assume on the contrary that there exist two distinct vertices x, y in G which are joined by two different $x - y$ paths, say P and Q. Denote by u the first vertex (as we traverse from x to y) on both P and Q such that

its successors on P and Q are distinct. Denote by v the next vertex which lies on both P and Q (note that both u and v exist). It is now clear that the union of the two $u-v$ paths on P and Q forms a cycle in G, a contradiction.

[Sufficiency] Suppose that every two vertices in G are joined by a unique path and assume on the contrary that G is not a tree. Then, by definition, G contains a cycle C. Take any two distinct vertices x and y on C. Clearly, x and y are joined by two paths along C in G, a contradiction. □

Look at those trees shown in Figure 14.1 again. How many vertices and edges are there in each tree? Is there any relation between these two numbers in each case?

Theorem 14.2 *Let G be a connected graph of order n and size m. Then G is a tree if and only if $m = n - 1$.*

Proof. [Necessity] We prove it by induction on n. The result is trivial if $n = 1$. Assume that the result is true for all trees of order less than n, where $n \geq 2$, and let G be a tree of order n. Choose an edge, say xy, in G. By Theorem 14.1, xy is the unique $x-y$ path in G. Thus, $G - xy$ is a disconnected graph having two disjoint components, say G_1 and G_2, both of which are also trees. Let n_i and m_i be, respectively, the order and size of G_i, $i = 1, 2$. By the induction hypothesis, we have $m_i = n_i - 1$ for each $i = 1, 2$. Thus,

$$m = m_1 + m_2 + 1 = (n_1 - 1) + (n_2 - 1) + 1 = n_1 + n_2 - 1 = n - 1,$$

as was to be shown.

[Sufficiency] Assume that G is connected and $m = n - 1$. We shall show that G is a tree. We first make the following:

Claim. G contains an end-vertex.

If not, then $d(v) \geq 2$ for each vertex v in G, and we have, by applying Euler's Handshaking Lemma (see [1]),

$$2m = \sum_{v \in V(G)} d(v) \geq 2n,$$

which implies that $m \geq n$, a contradiction. Thus, G contains an end-vertex, as claimed.

We shall now prove that G is a tree by induction on n. The result is trivial if $n = 1$. Assume that $n \geq 2$. By the above claim, G contains an end-vertex, say w. Clearly, $G - w$

is connected, and $e(G-w) = v(G-w) - 1$. By the induction hypothesis, $G-w$ is a tree. It follows that G is a tree, as desired. □

Recall that an edge e in a connected graph G is called a **bridge** if $G-e$ is disconnected (see [2]). It follows from the necessity part of Theorem 14.1 that if G is a tree, then every edge in G is a bridge. Conversely, if every edge in a connected graph G is a bridge, then G must be a tree by definition. Thus, we have (see Exercise 14.4):

Theorem 14.3 *Let G be a connected graph. Then G is a tree if and only if every edge in G is a bridge.* □

It follows from the proof of Theorem 14.2 that if T is a tree, then T contains at least one end-vertex. Indeed, one may apply Theorem 14.2 together with Euler's Handshaking Lemma to obtain the following expression for the number of end-vertices of a tree in terms of the numbers of vertices of degree $3, 4$ and so on. (See Exercise 14.5.)

Theorem 14.4 *Let T be a tree having p_i vertices of degree i, where $i = 1, 2, \cdots$. Then*

$$p_1 = 2 + p_3 + 2p_4 + \cdots + (i-2)p_i + \cdots.$$

□

It follows from Theorem 14.4 that every tree has at least two end-vertices. (Indeed, there are trees, namely paths, which have exactly two end-vertices.) Also, it is observed from the above expression that p_1 is independent of p_2, the number of vertices of degree two in T.

Consider the tree T of Figure 14.2. It can be checked in T that $p_3 = 4$, $p_4 = 3$ and $p_5 = 1$. By Theorem 14.4,

$$p_1 = 2 + p_3 + 2p_4 + 3p_5 = 2 + 4 + 6 + 3 = 15;$$

i.e., there are 15 end-vertices in T.

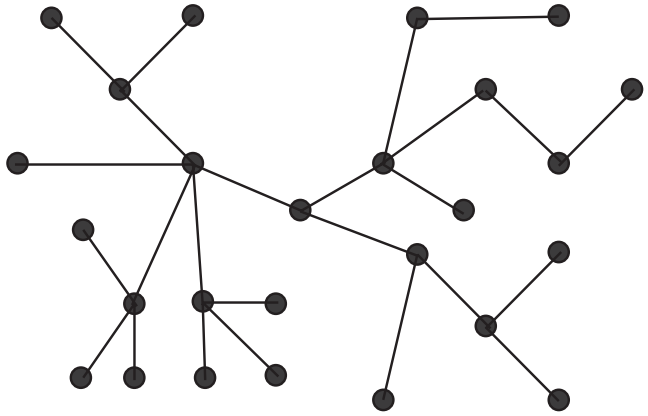

Figure 14.2

Exercise 14.1 *Draw all nonisomorphic trees of order n, where $2 \leq n \leq 7$.*

Exercise 14.2 *A* **forest** *is a graph which contains no cycle as a subgraph. Thus, a tree is a connected forest, and a forest is a graph in which each component is a tree (the graph (a) in Figure 14.1 is a forest with two components; the 'Chinese forest' with five components is shown in Figure 14.3). Let G be a forest having n vertices and k components, where $n \geq k \geq 1$. Express $e(G)$ in terms of n and k.*

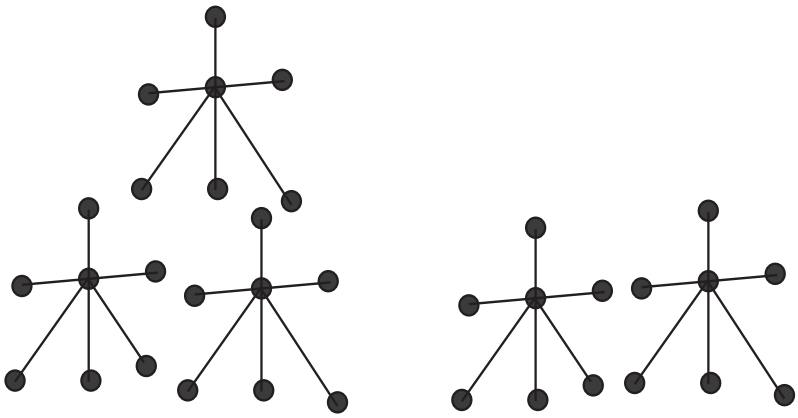

Figure 14.3

Exercise 14.3 *Let G be a graph of order n and size $n-1$, where $n \geq 4$. Must G be a tree?*

Exercise 14.4 *Prove Theorem 14.3.*

Exercise 14.5 *Prove Theorem 14.4.*

Exercise 14.6 *Let T be a tree of order n, where $n \geq 3$. What is T if $diam(T) = 2$? What is T if $diam(T) = n - 1$?*

Exercise 14.7 *In each tree T of Figure 14.4, find*

(i) the eccentricity of each vertex (see [4]);

(ii) $rad(T)$;

(iii) $diam(T)$; and

(iv) $C(T)$.

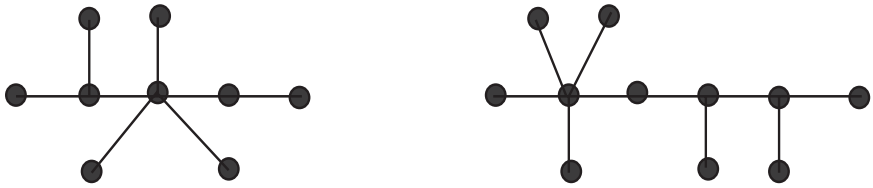

Figure 14.4

Exercise 14.8 *Let T be a tree. Show that T has either one central vertex or two adjacent central vertices (see [4]).*

15 Spanning Trees of a Multigraph

Consider the graph G of Figure 15.1. It is connected. Indeed, it remains connected even if some edges are removed. For instance, $T = G - \{uv, xy\}$ is still connected. However, we cannot afford to miss any other edge from T to maintain the connectedness. Observe that T is both a spanning subgraph of G and a tree. It is called a spanning tree of G.

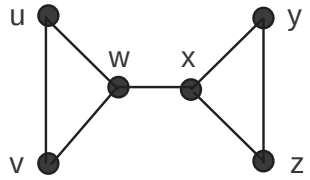

Figure 15.1

In general, a graph H is called a **spanning tree** of a multigraph G if H is a spanning subgraph of G (see[3]) as well as a tree. Besides T above, two other spanning trees are

shown in Figure 15.2. In fact, there are altogether nine spanning trees of G of Figure 15.1 (why?). Observe also that all the spanning trees of G contain the edge wx (why?).

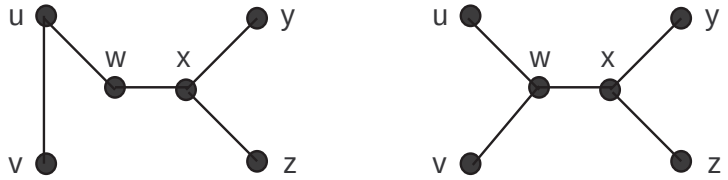

Figure 15.2

The following simple result relating the connectedness of a multigraph and the existence of its spanning trees is useful.

Theorem 15.1 *A multigraph G is connected if and only if it contains a spanning tree.*

Proof. [Sufficiency] Suppose that G contains a spanning tree, say T. We shall show that G is connected by showing that every two vertices in G are joined by a path in G. Thus, let x and y be any two vertices in G. Then, as T is spanning, x and y are in T. Since T is connected, there is a $x - y$ path in T. As T is a subgraph of G, this $x - y$ path is also in G, as required.

[Necessity] Suppose now that G is connected. If G contains no cycles, then G is itself a spanning tree of G. Otherwise, let C be a cycle of G and e be an edge in C. Then $G - e$ is still spanning and connected (why?). If $G - e$ contains no cycles, then $G - e$ is a spanning tree of G. Otherwise, we proceed as before by deleting an edge from an existing cycle. We continue this procedure repeatedly until a spanning tree of G is eventually found after a finite number of steps. □

If G is a connected multigraph, then, by Theorem 15.1, G contains a spanning tree T as a subgraph. Thus, by Theorem 14.2, we have:

$$e(G) \geq e(T) = v(T) - 1 = v(G) - 1.$$

Corollary 15.1 *If G is a connected multigraph, then $e(G) \geq v(G) - 1$.* □

An application. Figure 15.3 shows a 3-D folded structure and one of its flat layouts. Let us introduce a graph to study the relationship between the 3-D folded structure and

its flat layout. The 3-D folded structure has nine faces as indicated. Construct a graph G with $V(G) = \{a, b, c, d, e, f, g, h, i\}$, where each vertex represents a face, such that two vertices are adjacent in G if and only if the faces they represent have one edge in common. This graph G, called the **face adjacency graph** (FAG) of the 3-D folded structure, is shown in Figure 15.4. If we construct the FAG of the flat layout of Figure 15.3, then we obtain a graph, which is actually a spanning tree of G. In Figure 15.4, this spanning tree of G is shown with the bold edges. In the study of unfolding of 3-D folded structures (see [5], for instance), one fundamental problem is: given a 3-D folded structure, what are the flat layouts that could be folded into the structure? It is obvious from the above discussion that this problem is actually the problem of finding the spanning trees of the FAG of the given 3-D structure. On the other hand, some 3-D folded structures are 'non-manifold'. Not all the spanning trees of the FAG of such a non-manifold structure give rise to feasible flat layouts of the structure. The challenge here is to find out which spanning trees are feasible and to develop efficient algorithms for the search.

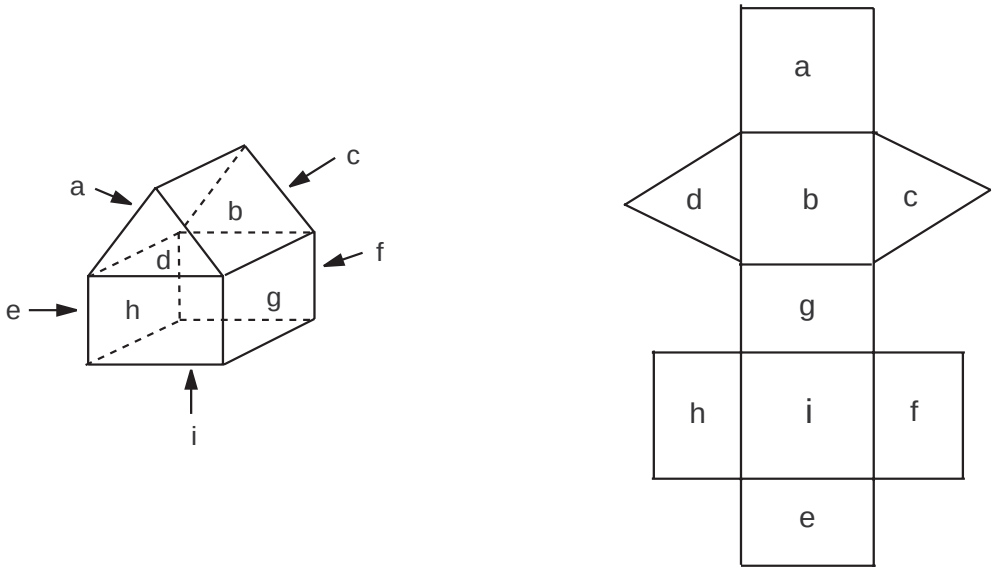

Figure 15.3

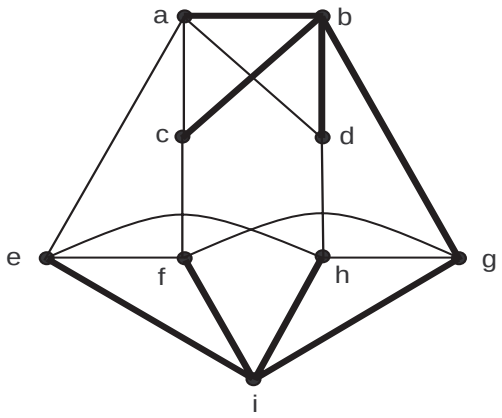

Figure 15.4

Exercise 15.1 *Find all spanning trees of the multigraph of Figure 15.5.*

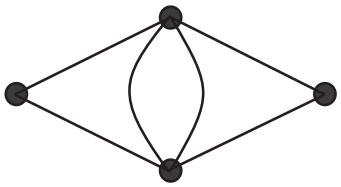

Figure 15.5

Exercise 15.2 *Let H be a graph of order 100 and size 98. Can H be connected? Why?*

Exercise 15.3 *Is the converse of the corollary to Theorem 15.1 true? That is: if G is a multigraph such that $e(G) \geq v(G) - 1$, must G be connected?*

Exercise 15.4 *Let G be a connected multigraph and e be a bridge in G. Must e be contained in any spanning tree of G? Why?*

Exercise 15.5 *Let G be a connected graph which contains a unique cycle C as a subgraph. Suppose that C is of order k, where $k \geq 3$. How many spanning trees does G have?*

16 Depth-first Search Trees and the One-way Street Problem

Given a connected multigraph G, how do we construct a spanning tree? The proof of the necessity part of Theorem 15.1 suggests that this could be done by 'breaking a cycle' (deleting one of its edges) in G step by step. This method is, however, not efficient as algorithmically detecting the existence of a cycle in G is a difficult task. In what follows, we shall introduce an efficient algorithm, called the **depth-first search algorithm**, to construct a spanning tree.

Algorithm (Finding a spanning tree in a connected multigraph G of order n.)

Start by picking a vertex at random and labeling it '1'. Pick any unlabeled neighbour of the vertex '1', and label it '2'; traverse from '1' to '2' via an edge and colour the edge chosen. In general, having labeled vertices with the labels '1', '2', $\cdots$, 'k' and coloured the edges chosen, search through the neighbours of the vertex 'k'. If among these, there are vertices which are unlabeled, pick one and label it '$k+1$'; traverse from 'k' to '$k+1$' via an edge and colour the edge chosen. Otherwise, find the **highest** label 'j' such that there is an unlabeled neighbour of the vertex 'j', pick such a vertex and label it '$k+1$'; traverse from 'j' to '$k+1$' via an edge and colour the edge chosen. Stop when all the vertices are labeled. The n vertices together with the coloured edges form a spanning tree of G.

Remark. The spanning tree of G obtained from the above algorithm is called a **depth-first search spanning tree** of G and the labeling '1', '2', $\cdots$, 'n' of the vertices in G is called a **depth-first search labeling** of G.

We give an example in Figure 16.1 to illustrate the above procedure step by step:

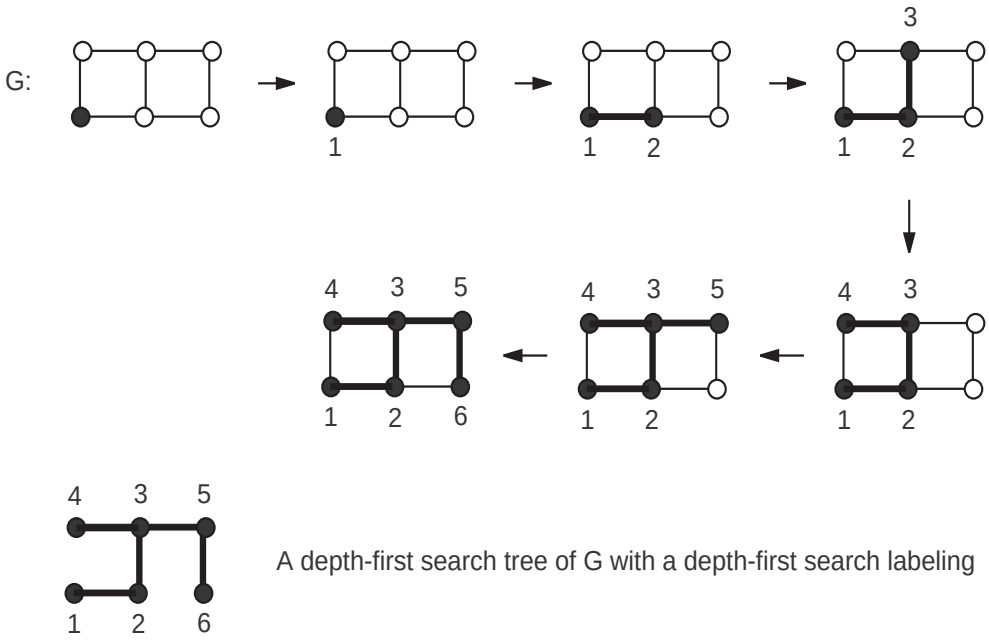

Figure 16.1

To end this section, we will show an application of the notion of a depth-first search tree to solve an interesting problem, the so-called **One-way Street Problem**. The graphs of Figure 16.2 model certain sections of the street systems of some towns where the roads are two-way. The edges represent the roads and the vertices represent the road junctions. To speed up traffic flow when the number of vehicles increases or during certain occasions such as when there is a large sports event at the nearby stadium, it may be better to make the roads one-way. Thus, the question is: Can we convert these two-way systems into one-way systems where every two vertices are still mutually reachable in each system?

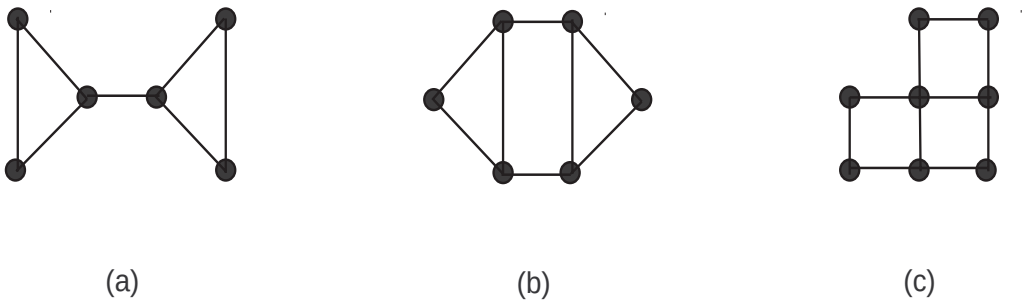

Figure 16.2

An **orientation** of a connected multigraph G is a one-way system obtained from G by assigning a direction to each edge of G. An orientation of G is said to be **strong** if every two vertices of G are mutually reachable (via directed paths) in it. An example is shown in Figure 16.3.

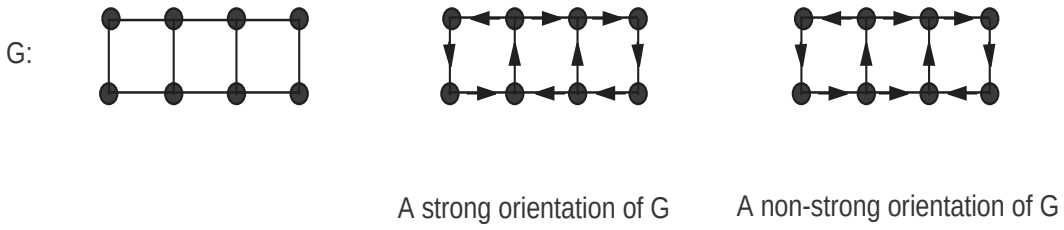

Figure 16.3

The One-way Street Problem may now be formulated using graph terminology as follows: **Under what conditions can a connected multigraph have a strong orientation?**

While the graphs (b) and (c) of Figure 16.2 have their strong orientations as shown in Figure 16.4, the graph (a) of the same figure doesn't have one due to the very obvious fact that it contains a 'bridge'. Indeed, it is trivial to see that if a connected multigraph does contain a bridge, then it has no strong orientations. Is the converse true? That is, does every bridgeless connected multigraph always have a strong orientation? The answer is in the affirmative as shown below.

Theorem 16.1 (Robbins [6]) *A connected multigraph has a strong orientation if and only if it contains no bridges.* □

Figure 16.4

The proof given by Robbins [6] on the existence of a strong orientation for a bridgeless connected multigraph is basically by induction. No explicit procedure for obtaining such

a strong orientation is spelled out therein. In what follows, we shall see how Roberts [7] made use of the depth-first search tree to obtain such a strong orientation.

Algorithm (Roberts [7], designing a strong orientation of a bridgeless connected multigraph G)

Step 1. Construct a depth-first search spanning tree T of G with a depth-first search labeling of G.

Step 2. For each edge in T, orient it from lower label to higher label; for each of the remaining edges in G, orient it from higher label to lower label.

Example 16.1 *Consider the graph (a) of Figure 16.5. Three of its depth-first search trees are shown in (b), (c) and (d) of the figure. By Roberts' algorithm, we design three respective strong orientations of the graph (a) as shown in (e), (f) and (g) of the figure respectively.*

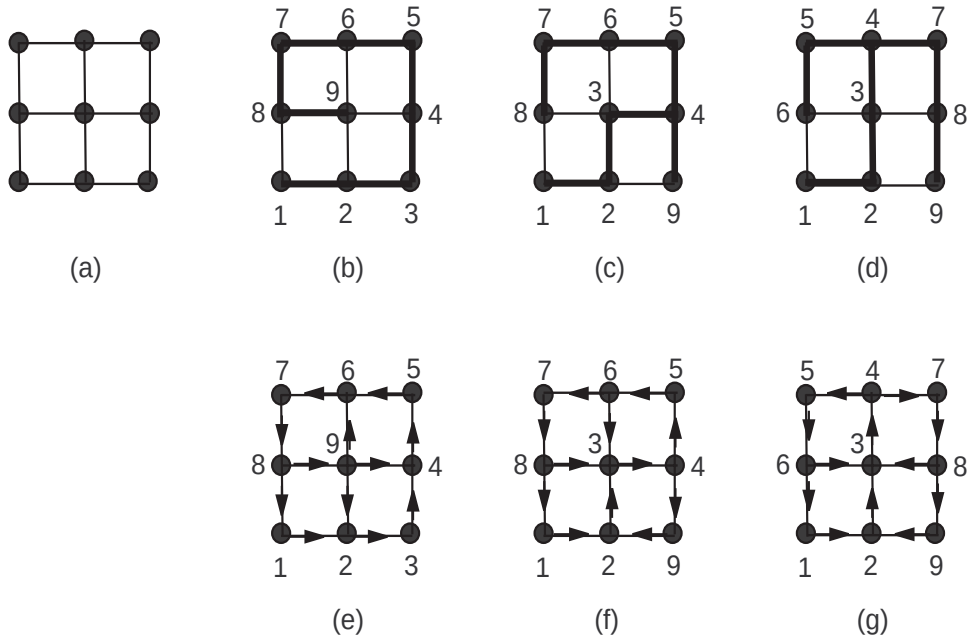

Figure 16.5

Exercise 16.1 *Design a strong orientation of each of the following graphs by applying Roberts' algorithm.*

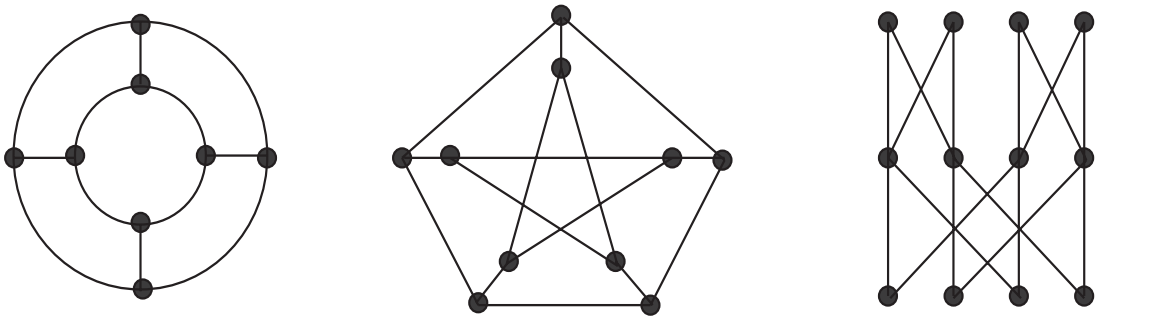

Figure 16.6

Exercise 16.2 *In the strong orientation of Figure 16.5(e), we observe that every two vertices are mutually reachable within eight steps (via at most eight arrows), and we do need eight steps to go from vertex '*1*' to vertex '*9*'. In the strong orientation of Figure 16.5(f), every two vertices are mutually reachable within seven steps, and we do need seven steps to go from vertex '*1*' to vertex '*8*'. In the strong orientation of Figure 16.5(g), it is noted that every two vertices are mutually reachable within six steps, and we do need six steps to go from vertex '*1*' to vertex '*9*'. Is it possible to design a strong orientation of the graph (a) of Figure 16.5 in which every two vertices are mutually reachable within*

(i) three steps?

(ii) four steps?

(iii) five steps? (You may have to ignore Roberts' algorithm.)

References

[1] K.M. Koh, Graphs and their applications (1), *Mathematical Medley* **29** (2) (2002) 86-94.

[2] K.M. Koh, Graphs and their applications (2), *Mathematical Medley* **30** (1) (2003) 11-22.

[3] K.M. Koh, F.M. Dong and E.G. Tay, Graphs and their applications (3), *Mathematical Medley* **30** (2) (2003) 102-116.

[4] K.M. Koh, F.M. Dong and E.G. Tay, Graphs and their applications (4), *Mathematical Medley* **31** (1) (2004) 9-19.

[5] W. Liu and K. Tai, Computational geometric modeling and unfolding of 3-D folded structures, in *ASME 2002 Design Engineering Technical Conference (28th Design Automation Conference), Montreal, Canada, 2002*, paper no. DETC2002/DAC-34046.

[6] H.E. Robbins, A theorem on graphs with an application to a problem of traffic control, *Am. Math. Monthly 46* (1939) 281-283.

[7] F.S. Roberts, Discrete Mathematical Models, with Applications to Social, Biological, and Environmental Problems, Prentice-Hall, Englewood Cliffs, NJ (1976).