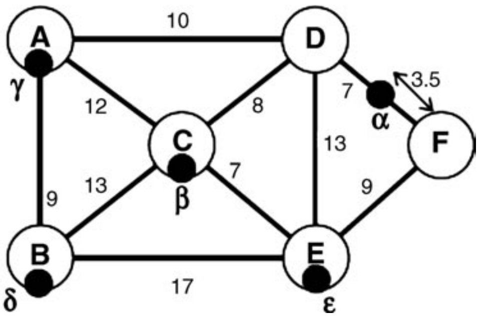

Figure 5.9 Optimal solution to the absolute 5-center problem for Figure 5.3.

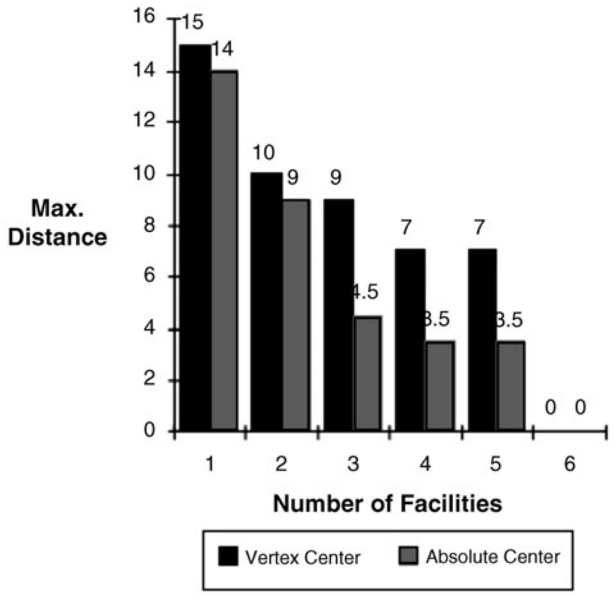

Figure 5.10 Comparison of absolute and vertex *P*-center solutions for Figure 5.3.

the network shown in Figure 5.3. Facility locations are shown by black dots and are denoted by Greek letters $(\alpha, \beta, \gamma, \delta,$ and $\varepsilon)$. In all cases, the facility at node α is the one whose distance to a demand node defines the absolute *P*-center objective function value. Figure 5.10 compares the objective function values for the vertex and absolute center problems for the network of Figure 5.3.

5.2 VERTEX *P*-CENTER FORMULATION

In this section we formulate the vertex *P*-center problem. We define the following notation:

Inputs and Sets

I = set of demand nodes

J = set of candidate locations

d_{ij} = distance from demand node $i \in I$ to candidate facility site $j \in J$

h_i = demand at node $i \in I$

P = number of facilities to locate

Decision Variables

$$X_j = \begin{cases} 1 & \text{if we locate at candidate site } j \in J \\ 0 & \text{if not} \end{cases}$$

Y_{ij} = fraction of demand at node $i \in I$ that is served by a facility at node $j \in J$

W = maximum distance between a demand node and the nearest facility

With this notation, we can formulate the vertex P-center problem as follows:

MINIMIZE
$$W \tag{5.1}$$

SUBJECT TO:
$$\sum_{j \in J} Y_{ij} = 1 \quad \forall i \in I \tag{5.2}$$

$$\sum_{j \in J} X_j = P \tag{5.3}$$

$$Y_{ij} \le X_j \quad \forall i \in I; j \in J \tag{5.4}$$

$$W \ge \sum_{j \in J} d_{ij} Y_{ij} \quad \forall i \in I \tag{5.5}$$

$$X_j \in \{0, 1\} \quad \forall j \in J \tag{5.6}$$

$$Y_{ij} \ge 0 \quad \forall i \in I; j \in J \tag{5.7}$$

The objective function (5.1) minimizes the maximum distance between a demand node and the closest facility to the node. Constraints (5.2) state that all of the demand at node $i \in I$ must be assigned to a facility at some node $j \in J$ for all nodes $i \in I$. Constraint (5.3) stipulates that P facilities be located. Constraints (5.4) state that demands at node $i \in I$ cannot be assigned to a facility at node $j \in J$ unless a facility is located at node $j \in J$. Constraints (5.5) state that the maximum distance between a demand node and the nearest facility to the node (W) must be greater than the distance between *any* demand node $i \in I$ and the facility $j \in J$ to which it is assigned. Constraints (5.6) and (5.7) are the integrality and nonnegativity constraints, respectively. Note that while we do not explicitly require the allocation variables Y_{ij} to be integers, in the optimal solution they will naturally be integers (or, as is more likely, we can find an alternate optimum in which they are all integers). This is so because the facilities are not capacitated and there is no reason to allocate demands at node $i \in I$ to a facility other than the closest one.

In some cases, we want to consider the demand-weighted distance. In that case, constraint (5.5) can be replaced by

$$W \geq \sum_{j \in J} h_i d_{ij} Y_{ij} \quad \forall i \in I \tag{5.8}$$

While this generalization of the problem is of mathematical interest, in most applied contexts, if we are interested in the P-center problem we want to minimize the worst-case level of service of the system, independent of the number of demands that experience that level of service.[1]

For fixed values of the number of facilities to locate, P, both the absolute and vertex P-center problems may be solved in polynomial time. To see this for the vertex P-center problem, all we need to realize is that for a network with n nodes, we need only evaluate each of the $O(\binom{n}{P}) = O(n^P)$ possible combinations of P facility sites. Evaluating each of these can be done in polynomial time, so finding the optimal solution to the vertex P-center problem, for *fixed* values of P, can be done in polynomial time. As indicated below in Section 5.5, we can identify a finite polynomial number of points to add to the network such that the solution to the absolute P-center problem will consist of locating on a subset of the original nodes and/or the additional points. Thus, the absolute P-center problem can also be solved in a polynomial amount of time for fixed values of P.

For a general graph and for variable values of P, the P-center problem is NP-complete (Kariv and Hakimi, 1979a; Garey and Johnson, 1979, pp. 219–220). This is true for both the vertex and absolute P-

[1] Since the solution techniques for the P-center problem depend in part on whether or not we want to solve a demand-weighted or unweighted version of the problem, we will retain this distinction throughout this chapter.

center problems. To see why this is so, note that the time required to solve all vertex P-center problems by enumeration for $P = 1$ to $P = n$ for any given value of n is

$$\sum_{j=1}^{n} \binom{n}{j} = 2^n - 1 = O(2^n)$$

which is exponential in n. The fact that this problem is NP-complete means that there is no known polynomial time optimal algorithm for solving these problems on a general network. However, when the network is a tree, we can often find optimal solutions in polynomial time to problems that are NP-complete on more general graphs. This is the case here. Section 5.3 outlines a solution approach to the absolute 1-center and absolute 2-center problems on a tree. Section 5.4 summarizes a solution approach to the vertex P-center problem on a general graph. Section 5.5 presents characteristics of the solution to the absolute P-center problem on a general graph and outlines a solution procedure for absolute P-center problem on a general graph.

5.3 THE ABSOLUTE 1- AND 2-CENTER PROBLEMS ON A TREE

While the absolute P-center problem is NP-complete on a general graph, if we are only locating a single facility on a tree, there are very efficient algorithms for solving the problem optimally. In this section, we outline a number of those approaches. We begin by considering the absolute 1-center problem on a tree in which all of the demands are equal. We call such a tree an *unweighted* tree. Since all of the demands are equal, we can normalize them so that they are all equal to 1. We then show how that algorithm can be extended to the case in which we want to find two centers on an unweighted tree. This section concludes by considering the case of a single absolute center on a weighted tree.

5.3.1 Absolute 1-Center on an Unweighted Tree

To find the absolute 1-center on an unweighted tree, we can use the following algorithm:

Step 1: Pick any point on the tree and find the vertex that is farthest away from the point that was picked. Call this vertex e_1.