

**Problem Set 6**

(Due: Wednesday, December 6, 2006)

**Problem 1**

Problem 6.6 in Larson and Odoni

**Problem 2**

Exercise 6.7 (page 442) in Larson and Odoni.

**Problem 3**

Suppose we have a network  $G(N, A)$  such as the one pictured in Figure 1, which can be separated by an “isthmus edge”,  $(s, t)$  into two distinct sub-networks  $G(S, A_s)$  and  $G(T, A_t)$  such that  $S \cup T = N$  and  $A_s \cup A_t \cup (s, t) = A$ . (Note that the set of nodes  $S$  includes node  $s$  and the set of nodes  $T$  includes node  $t$ .) Let  $H(T)$  be the sum of the weights,  $h_j$ , of the nodes in the set  $T$  and  $H(S)$  be the sum of the weights,  $h_j$ , of the nodes in the set  $S$ .

(a) The following is known as Goldman’s majority theorem”: “If  $H(T) \geq H(S)$  then the set of nodes  $T$  contains at least one solution to the 1-median problem on  $G(N, A)$ .”

Prove the theorem. To do so, assume that the solution is at some node  $y \in S$  and argue that  $J(y) \geq J(t)$ , a contradiction.  $J(\cdot)$  is the objective function for the 1-median problem – see book. Note as well that  $t$  is the node on the  $G(T, A_t)$  “side” of  $(s, t)$ .

(b) Prove the following theorem: “If  $H(T) \geq H(S)$  then one can find a solution to the original 1-median problem on  $G(N, A)$  by solving the 1-median problem on the sub-network  $G'(T, A_t)$  which is identical to  $G(T, A_t)$  except that the weight  $h_t$  of the node  $t$  (on which the edge  $(s, t)$  is incident) is replaced by  $H(S) + h_t$ .”

To prove this statement argue as follows: We know from part (a) that the 1-median is in  $T$ . Show that for any node  $y \in T$ :

$$J(y) = C + [H(S) + h_t] \cdot d(y, t) + \sum_{j \in (T-t)} h_j \cdot d(y, j) \quad (1)$$

where  $C$  is a constant and  $(T-t)$  indicates the set of nodes,  $T$ , not including the node  $t$ . Why does (1) prove our theorem?

(c) Using the theorems of parts (a) and (b) find very quickly the 1-median of the network shown in Figure 2. (For each node, an identification letter followed by the node’s weight

is indicated; link lengths are noted next to each link.) Note that you do not have to consider the lengths of the edges in solving this problem.

#### **Problem 4**

Consider the Traveling Salesman Problem with Backhauls (TSPB), a version of the TSP which is as follows: Suppose we have one “station point,”  $s$ , a set  $D$  of “delivery” points ( $|D| = n$ ) and a set  $P$  of “pick-up” points ( $|P| = m$ ). Assume the travel medium is the Euclidean plane and that all  $n+m+1$  points in the problem are distinct, so that the Euclidean distance between any pair of points is positive. We want to design a tour of minimum length that has this description: a vehicle will begin from  $s$ , will visit first all the  $n$  points in  $D$  to deliver packages, will then (without first returning to  $s$ ) visit all  $m$  points in  $P$  to pick up packages and will finally go back to  $s$  (where the tour ends).

The following heuristic, based on the idea of the Christofides heuristic for the TSP, has been proposed recently for the TSPB (we shall leave Step 4 incomplete until the second question of our problem):

Step 1: Construct: (a)  $TD$ , the minimum spanning tree of  $D$ , i.e., the MST that connects all  $n$  delivery points; (b)  $TP$ , the minimum spanning tree of  $P$ , i.e., the MST that connects all  $m$  pick-up points.

Now connect  $s$  to the delivery point in  $D$  which is closest to  $s$ . Also connect  $s$  to the pick-up point in  $P$  which is closest to  $s$ .

In this way a single tree,  $T$ , is formed which consists of  $TD$ ,  $TP$ ,  $s$ , and the two links connecting  $s$  to  $TD$  and  $TP$ , respectively.

Step 2: Transform the  $(n+m+1) \times (n+m+1)$  matrix of Euclidean distances between pairs of points in the problem in the following way: leave all distances between pairs of points in  $D$  unchanged, i.e., equal to the Euclidean distances between these points; similarly, leave all distances between pairs of points in  $P$  unchanged; then add a large constant  $K$  to all the other distances in the matrix. This means that if the Euclidean distance between any point  $i \in D$  and any point  $j \in P$  is  $d(i, j)$  units, it will be changed to  $d(i, j) + K$  in the transformed distance matrix; similarly the distance between  $s$  and  $i$  in the transformed distance matrix will be  $d(s, i) + K$  and the distance between  $s$  and  $j$  will be  $d(s, j) + K$ . Note that the same “large constant”  $K$  is added in all cases.  $K$  is chosen so that it is much larger than any of the Euclidean distances (or sums of Euclidean distances) encountered in the problem.

Step 3: Let  $R$  be the set of odd-degree nodes in  $T$ , the tree obtained in Step 1. Find the minimum-cost pair-wise matching between the nodes (points) in  $R$ , *using the transformed matrix prepared in Step 2*. Let  $H$  be the (disjoint) graph consisting of the set of links (straight lines) which correspond to the minimum-cost pair-wise matching. (For example, if the points  $i \in D$  and  $j \in P$  are both in  $R$  and have been matched together in the minimum-cost matching, then the link  $(i, j)$  will be in  $H$ ; similarly, if the points  $v \in D$  and  $w \in D$  are both in  $R$  and have been matched together, the link  $(v, w)$  will be in  $H$ ; and so on.) Note that, in drawing the graph  $H$ , *we forget about the large constant  $K$* .

Step 4: Merge  $H$  with  $T$  to obtain a graph  $G$  (i.e.,  $G = T \cup H$ ).

(a) Explain carefully but briefly why it is true that the number of (odd-degree) nodes in  $R$  which are also delivery points is an *odd* number (i.e.,  $|R \cap D|$  is an odd number).

(a) Given the result of part (a), argue carefully, but briefly, that:

- (i) The graph  $G$  constructed in Step 4 of the algorithm has an Euler tour.
- (ii) It is possible, using only links in  $G$ , to find an Euler tour, which is a feasible (not necessarily) optimal solution to the TSPB. In other words, it is possible to find on  $G$  an Euler tour that begins at  $s$ , visits all the points in  $D$  at least once, then visits all the points in  $P$  at least once, and finally returns to the origin  $s$ .

[Note: the heuristic described in this problem is the best currently available for the TSPB, in terms of worst-case performance.]

## Problem 5

Consider the following routing problem, which we shall call the Delivery Truck Problem (DTP). Suppose a truck must perform a set of pickups and deliveries in a Euclidean travel environment. Each load which is picked up fills the truck completely and goes to a single destination, so no pickups and deliveries can be combined. This is shown in Figure 3a, for a problem involving four loads: points  $A$ ,  $C$ ,  $E$ , and  $G$  are the pick-up points and the corresponding set of directed edges between each pick-up and each delivery is denoted as  $R = \{(A,B), (C,D), (E,F), (G,H)\}$ . We shall call the pick-up points “head vertices” and the delivery points ( $B$ ,  $D$ ,  $F$ ,  $H$ ) “tail vertices”. The DTP is to find a tour that begins and ends at some head vertex (we shall use point  $A$  in our example without loss of generality) and is of minimal total length.

The following heuristic has been suggested for this problem:

Step 1: Construct an undirected graph  $G_1$  whose vertices are all the head and tail vertices and whose edges are all possible (head vertex, tail vertex) edges, except those edges appearing in  $R$ . In other words, in building  $G_1$ , each head vertex is connected to all tail vertices except the one it is connected to in  $R$ . For example, in Figure 3b, we draw an edge between  $A$  and  $D$ ,  $A$  and  $F$ , and  $A$  and  $H$ , but not between  $A$  and  $B$ . The cost (or length) of each edge of  $G_1$  is equal to the Euclidean distance between the points that it connects.  $G_1$  for our example is shown in Figure 3b.

Step 2: Find a minimum cost pairwise matching,  $M$ , of the vertices of  $G_1$ . For our example, let us assume that  $M = \{(A,D), (C,B), (E,H), (G,F)\}$ . Construct a graph  $G_2$ , by merging  $R$  with  $M$  (Figure 3c). Note that  $G_2$  has some directed edges (the ones associated with  $R$ ) and some undirected ones (those associated with  $M$ ).

Step 3: Construct a graph  $G_3$  with one vertex corresponding to each of the  $k$  disjoint cycles of  $G_2$  and one vertex between each pair of these vertices. [Note that in our example  $k=2$  and thus  $G_3$  will have one vertex corresponding to the set of nodes  $V = (A, B, C, D)$  and another corresponding to the set  $W = (E, F, G, H)$  – see Figure 3d.] The cost (or length) of each edge in  $G_3$  is set equal to the distance between the corresponding pair of cycles, i.e., to the distance between two nodes, one in each cycle, which are closest to each other. [In our example,

$$c(V, W) = \min c(i, j) \quad \text{for all vertices } i \in V \text{ and } j \in W$$

where  $c(i, j)$  indicates the distance between vertices  $i$  and  $j$ .] Find the minimum spanning tree  $T$  of  $G_3$ . The minimum spanning tree  $T$  of  $G_3$  for our example is shown in Figure 3d; note that because, in this case,  $k=2$ ,  $T$  and  $G_3$  are identical.

Step 4: Construct a new graph  $G_4$  by adding to  $G_2$  two directed edges for each edge in  $T$ . The two edges corresponding to an edge  $(V, W)$  in  $T$  join the two vertices  $i \in V$  and  $j \in W$ , whose distance  $c(i, j)$  equals  $c(V, W)$  and go in opposite directions. [For our example, assume that  $c(V, W) = c(E, C)$ . Hence, we draw a directed edge from  $C$  to  $E$  and a directed edge from  $E$  to  $C$  and add them to  $G_2$  to obtain  $G_4$ , as shown in Figure 3e.]

Step 5: Draw an Euler tour through the mixed graph  $G_4$ , beginning and ending at the required point and making sure to traverse all the directed edges in the correct direction. This tour is a solution to the DTP.

(a) Please explain briefly why the mixed graph  $G_4$  has an Euler tour.

(b) Argue briefly that, if  $k=1$  after Step 2, the above heuristic provides an optimal solution to the DTP.

(c) Let  $L(G_4)$  denote the length of the tour found in Step 5 (i.e., the length of the solution produced by our heuristic),  $L(\text{DTP})$  be the length of an optimal DTP tour and  $L(R)$  the length of the distance the truck will travel fully loaded. (Note that  $L(R)$  will always be part of the distance that any legitimate DTP solution, exact or approximate, must cover.)

Argue that

$$L(G_4) \leq 3L(\text{DTP}) - 2L(R)$$

[Hint: This is not difficult. Consider how  $G_4$  was obtained.]

## Problem 6

Consider the following version of the  $k$ -traveling salesmen problem in Euclidean space: Suppose we have  $n$  cities of which city 1 is the home location of the  $k$  salesmen. We want to design a set of  $k$  tours such that every city is visited by one salesman and the length of the longest tour traveled by any salesman is minimized. In other words, our

measure of the quality of a given collection of  $k$  routes is the length of the longest of these routes. We call this longest route the “maximum length subtour.”

Suppose we use the following algorithm for this problem:

Step 1: Beginning and ending at city 1, generate a single traveling salesman tour  $T$  that includes all  $n$  cities. Do this by using your favorite TSP heuristic. Number the cities  $i_1, i_2, \dots, i_n$  in the order they appear in the tour (note that  $i_1 = 1$ ). Let  $L$  be the length of  $T$  and let  $d_{\max}$  be the maximum *direct* distance between any city and city 1.

Step 2: Generate  $k$  subtours in the following way: For each  $j, 1 \leq j < k$ , define  $p(j)$  to be the largest integer such that the length of the path in  $T$  from 1 to node  $i_{p(j)}$  does not exceed the quantity  $\frac{j}{k}(L - 2d_{\max}) + d_{\max}$ .

Step 3: Let the  $k$  subtours be  $T_1 = \{1, i_2, \dots, i_{p(1)}, 1\}$ ,  $T_2 = \{1, i_{p(1)+1}, \dots, i_{p(2)}, 1\}, \dots,$   
 $T_k = \{1, i_{p(k-1)+1}, \dots, i_n, 1\}$ .

(a) Show that the lengths  $L_1, L_2, \dots, L_k$  of all the subtours  $T_1, T_2, \dots, T_k$  generated in this way must be less than or equal to the quantity

$$\frac{1}{k}(L - 2d_{\max}) + 2d_{\max} \quad (1)$$

[Hint: Show this separately for  $L_1$ , for  $L_j$  such that  $2 \leq j < k - 1$ , and for  $L_k$ .]

(b) From part (a) we know that the length of the maximum length subtour generated in this way [call this length  $L(T_{LONG}) = \max(L_1, L_2, \dots, L_k)$ ] must be less than or equal to (1) above. Suppose you had used the Christofides heuristic to draw the initial single traveling salesman tour  $T$ . Let also  $L(T^*_{LONG})$  denote the length of the maximum length subtour in the *optimum solution* to this  $k$ -traveling salesmen problem (assuming you could somehow solve the problem optimally).

Show that

$$L(T_{LONG}) \leq \left(\frac{5}{2} - \frac{1}{k}\right) \cdot L(T^*_{LONG}) \quad (2)$$

i.e., that our heuristic produces a solution which is within less than 150% of the optimum no matter what  $k$  is.

[Hint: How do the quantities  $L(T_{LONG})$  and  $L$  compare to the length  $L^*$  of the optimum (single) traveling salesman tour through all  $n$  cities? How does  $d_{\max}$  compare with one of these quantities?]

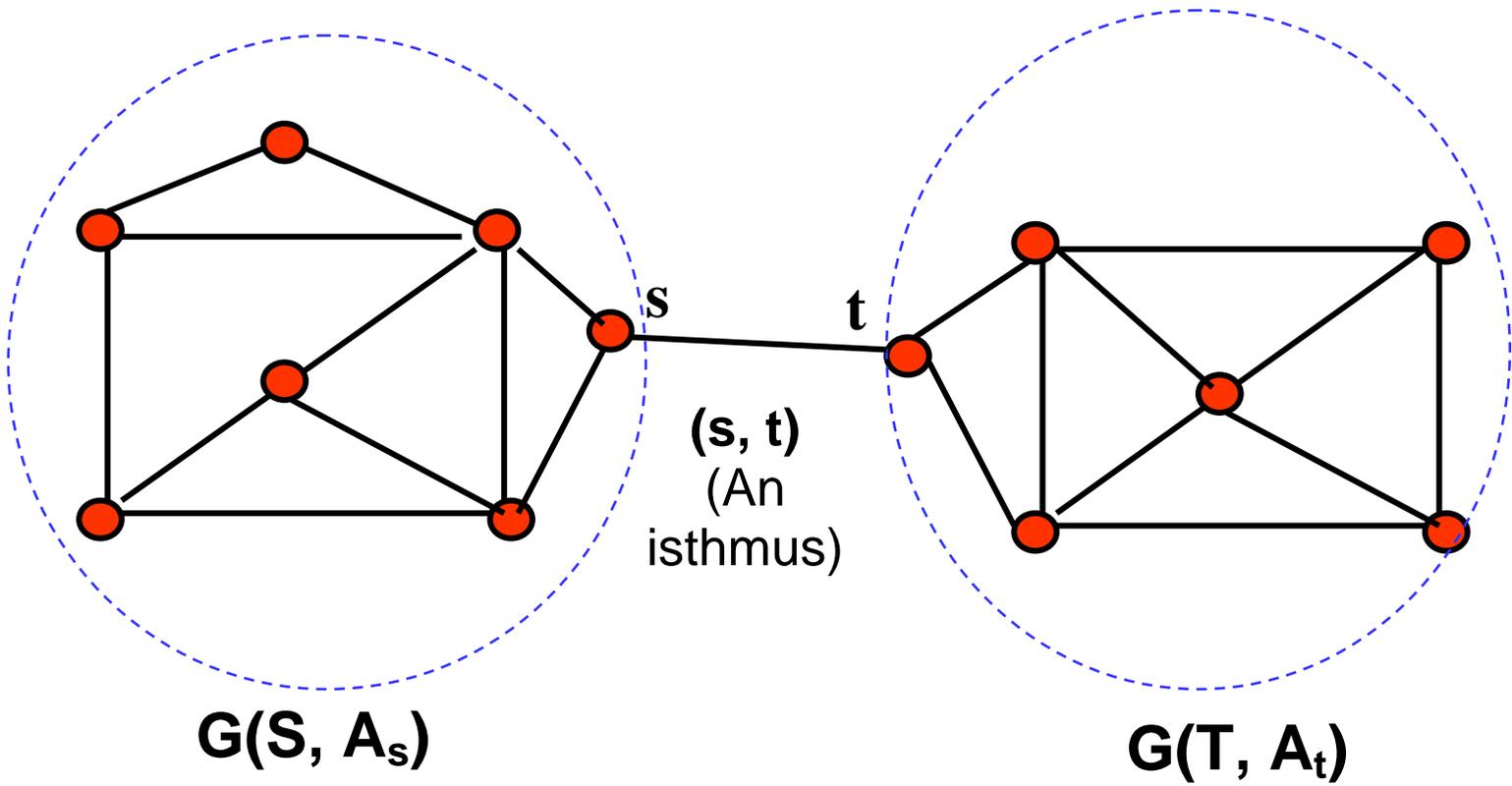
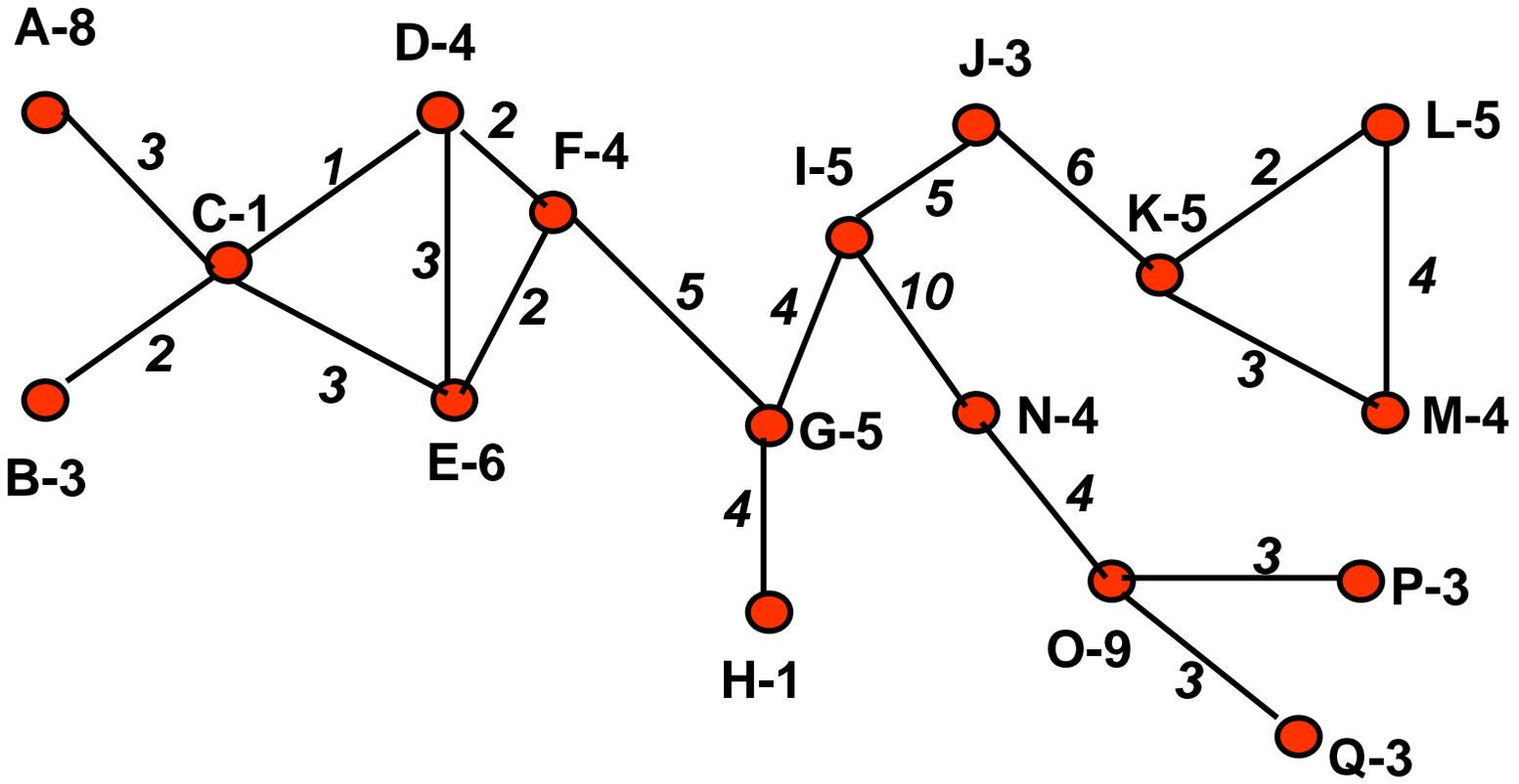


Figure 1

Figure 2



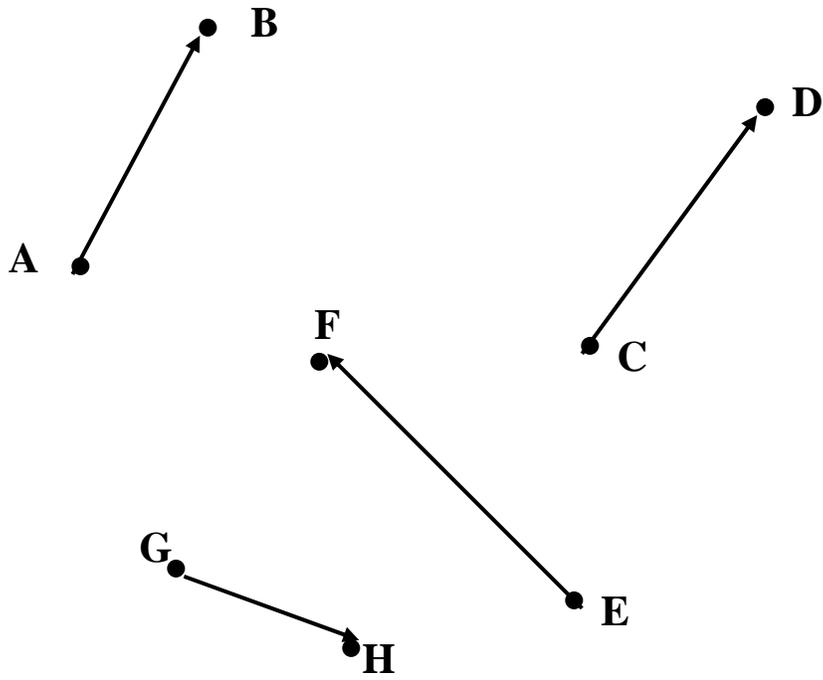


Figure 3a

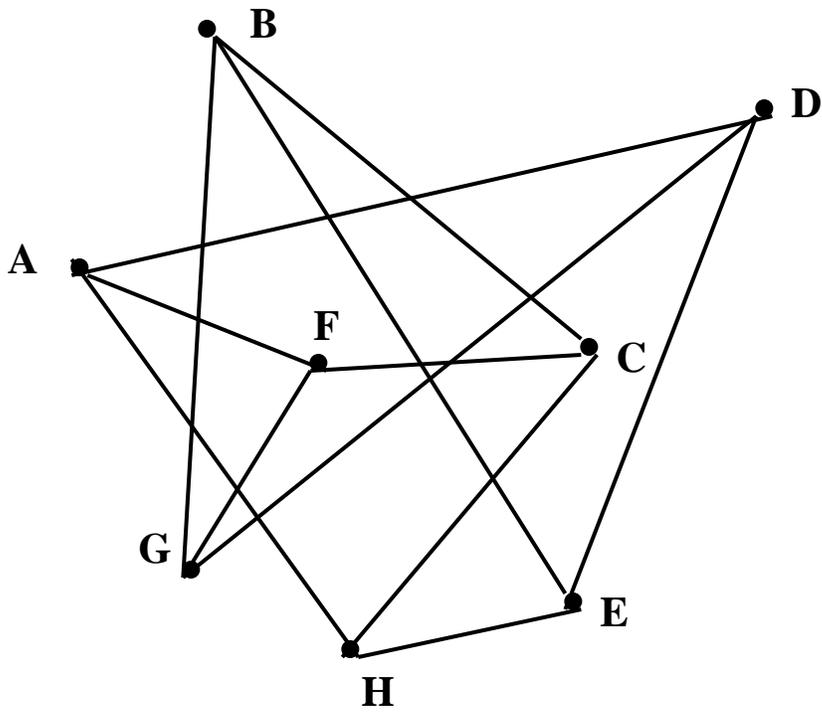


Figure 3b

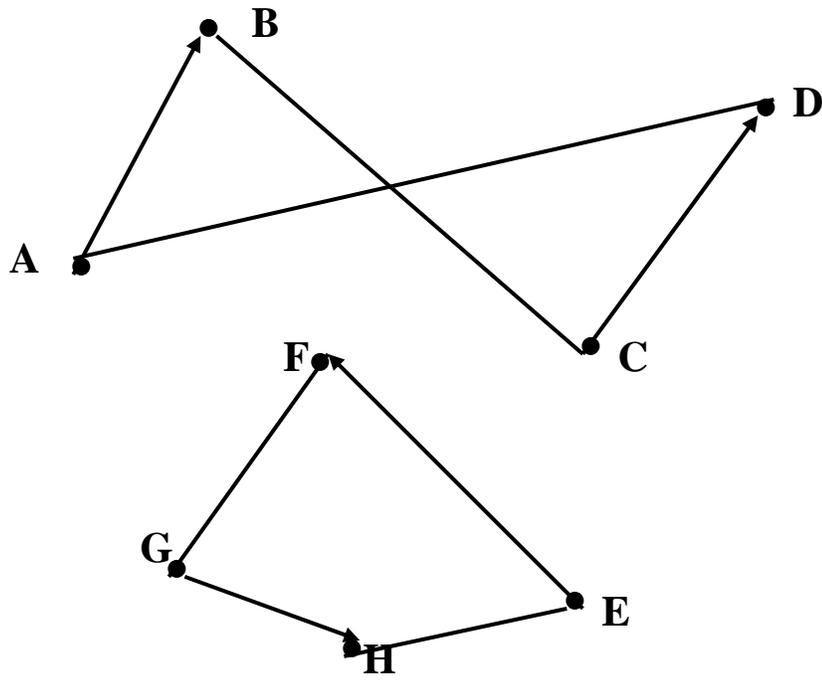


Figure 3c



Figure 3d

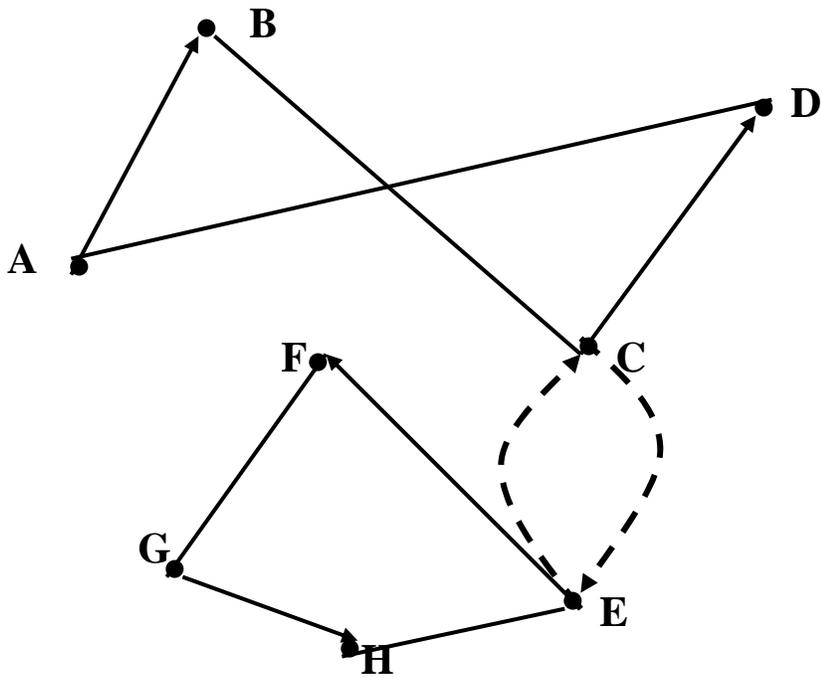


Figure 3e