CHAPTER 6

# Network Models

There is a multitude of operations research situations that can be modeled and solved as networks (nodes connected by branches). Some recent surveys report that as much as 70% of the real-world mathematical programming problems can be represented by network-related models. The following list illustrates possible applications of networks.

1. Design of an offshore natural gas pipeline network connecting wellheads in the Gulf of Mexico to an inshore delivery point. The objective of the model is to minimize the cost of constructing the pipeline.
2. Determination of the shortest route between two cities in a network of roads.
3. Determination of the maximum capacity (in tons per year) of a coal slurry pipeline network joining the coal mines in Wyoming with the power plants in Houston. (Slurry pipelines transport coal by pumping water through specially designed pipes.)
4. Determination of the minimum-cost flow schedule from oil fields to refineries through a pipeline network.
5. Determination of the time schedule (start and completion dates) for the activities of a construction project.

The solution of these situations, and others like it, is accomplished through a variety of network optimization algorithms. This chapter will present five of these algorithms.

1. Minimal spanning tree (situation 1)
2. Shortest-route algorithm (situation 2)
3. Maximum flow algorithm (situation 3)
4. Minimum-cost capacitated network algorithm (situation 4)
5. Critical path (CPM) algorithm (situation 5)

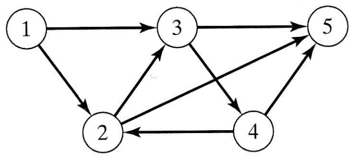
The situations for which these algorithms apply can also be formulated and solved as explicit linear programs. However, the proposed network-based algorithms are more efficient than the simplex method.

## 6.1 NETWORK DEFINITIONS

A network consists of a set of **nodes** linked by **arcs** (or **branches**). The notation for describing a network is $(N, A)$, where $N$ is the set of nodes, and $A$ is the set of arcs. As an illustration, the network in Figure 6.1 is described as

$$N = \{1,2,3,4,5\}$$
$$A = \{(1,2),(1,3),(2,3),(2,5),(3,4),(3,5),(4,2),(4,5)\}$$

FIGURE 6.1
Example of $(N, A)$ network



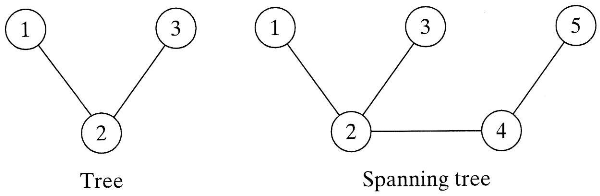

Associated with each network is some type of flow (e.g., oil products flow in a pipeline and automobile traffic flows on highways). In general, the flow in a network is limited by the capacity of its arcs, which may be finite or infinite.

An arc is said to be **directed** or **oriented** if it allows positive flow in one direction and zero flow in the opposite direction. A **directed network** has all directed arcs.

A **path** is a sequence of distinct arcs that join two nodes through other nodes regardless of the direction of flow in each arc. A path forms a **cycle** if it connects a node to itself through other nodes. For example, in Figure 6.1, arcs (2,3), (3,5), and (5,2) form a loop. A cycle is *directed* if it consists of a directed path; e.g., (2, 3), (3, 4), and (4, 2) in Figure 6.1.

A **connected network** is such that every two distinct nodes are linked by at least one path. The network in Figure 6.1 demonstrates this type of network. A **tree** is a connected network that may involve only a *subset* of all the nodes of the network with no cycles allowed, and a **spanning tree** is a tree that links *all* the nodes of the network, also with no cycles allowed. Figure 6.2 provides examples of a tree and a spanning tree for the network in Figure 6.1.

FIGURE 6.2
Examples of a tree and a spanning tree given the network in Figure 6.1



## PROBLEM SET 6.1A

1. For each network in Figure 6.3 determine (a) a path, (b) a cycle, (c) a directed cycle, (d) a tree, and (e) a spanning tree.

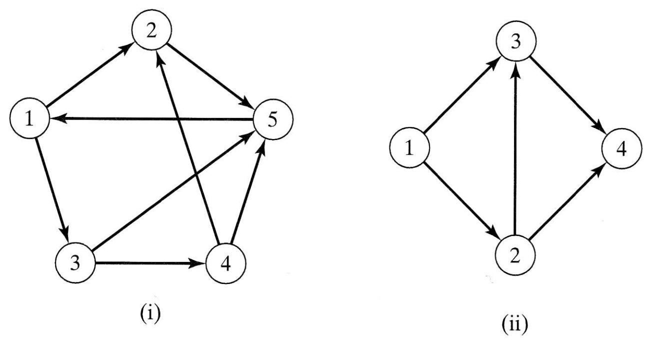



FIGURE 6.3
Networks for Problem 1, Set 6.1a

2. Determine the sets $N$ and $A$ for the networks in Figure 6.3.
3. Draw the network defined by

$$N = \{1,2,3,4,5,6\}$$

$$A = \{(1,2),(1,5),(2,3),(2,4),(3,5),(3,4),(4,3),(4,6),(5,2),(5,6)\}$$

4. Consider eight equal squares arranged in three rows, with two squares in the first row, four in the second, and two in the third. The squares of each row are arranged symmetrically about the vertical axis. It is desired to fill the squares with distinct numbers in the range 1, 2, ..., and 8 so that no two *adjacent* vertical, horizontal, or diagonal squares hold consecutive numbers. Use network representation as a vehicle to find the solution in a systematic way.
5. Three inmates escorted by 3 guards must be transported by boat from San Francisco to the Alcatraz penitentiary island to serve their sentences. The boat cannot transfer more than two persons in either direction. The inmates are certain to overpower the guards if they outnumber them at any time. Develop a network model that designs the boat trips in a manner that ensures a safe transfer of the inmates. Assume that the inmates will not flee if given a chance.

## 6.2 MINIMAL SPANNING TREE ALGORITHM

The minimal spanning tree algorithm deals with linking the nodes of a network, directly or indirectly, using the shortest length of connecting branches. A typical application occurs in the construction of paved roads that link several towns. The road between two towns may pass through one or more other towns. The most economical design of the road system calls for minimizing the total miles of paved roads, a result that is achieved by implementing the minimal spanning tree algorithm.

The steps of the procedure are given as follows. Let $N = \{1, 2, \ldots, n\}$ be the set of nodes of the network and define

$C_k$ = Set of nodes that have been permanently connected at iteration $k$

$\overline{C}_k$ = Set of nodes as yet to be connected permanently

**Step 0.** Set $C_0 = \emptyset$ and $\overline{C}_0 = N$.

**Step 1.** Start with *any* node, $i$, in the unconnected set $\overline{C}_0$ and set $C_1 = \{i\}$, which renders $\overline{C}_1 = N - \{i\}$. Set $k = 2$.

**General Step k.** Select a node, $j^*$, in the unconnected set $\overline{C}_{k-1}$ that yields the shortest arc to a node in the connected set $C_{k-1}$. Link $j^*$ permanently to $C_{k-1}$ and remove it from $\overline{C}_{k-1}$, that is,

$$C_k = C_{k-1} + \{j^*\}, \overline{C}_k = \overline{C}_{k-1} - \{j^*\}$$

If the set of unconnected nodes, $\overline{C}_k$, is empty, stop. Otherwise, set $k = k + 1$ and repeat the step.

## Example 6.2-1

Midwest TV Cable Company is in the process of providing cable service to five new housing development areas. Figure 6.4 depicts possible TV linkages among the five areas. The cable miles are shown on each arc. Determine the most economical cable network.

The algorithm starts at node 1 (any other node will do as well), which gives

$$C_1 = \{1\}, \overline{C}_1 = \{2,3,4,5,6\}$$

The iterations of the algorithm are summarized in Figure 6.5. The thin arcs provide all the candidate links between $C$ and $\overline{C}$. The thick branches represent the permanent links among the nodes of the connected set $C$, and the dashed branch represents the new (permanent) link added at each iteration. For example, in iteration 1, branch (1,2) is the shortest link ( = 1 mile) among all the candidate branches from node 1 to nodes 2, 3, 4, and 5 of the unconnected set $\overline{C}_1$. Hence, link (1,2) is made permanent and $j^* = 2$, which yields

$$C_2 = \{1,2\}, \overline{C}_2 = \{3,4,5,6\}$$

The solution is given by the minimal spanning tree shown in iteration 6 of Figure 6.5. The resulting minimum cable miles needed to provide the desired cable service are $1 + 3 + 4 + 3 + 5 = 16$ miles.

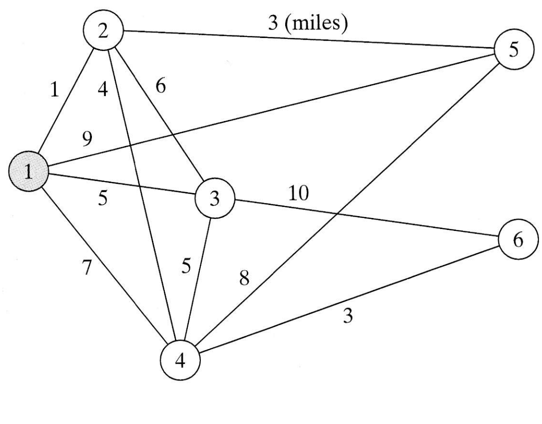



FIGURE 6.4

Cable connections for Midwest TV Cable Company

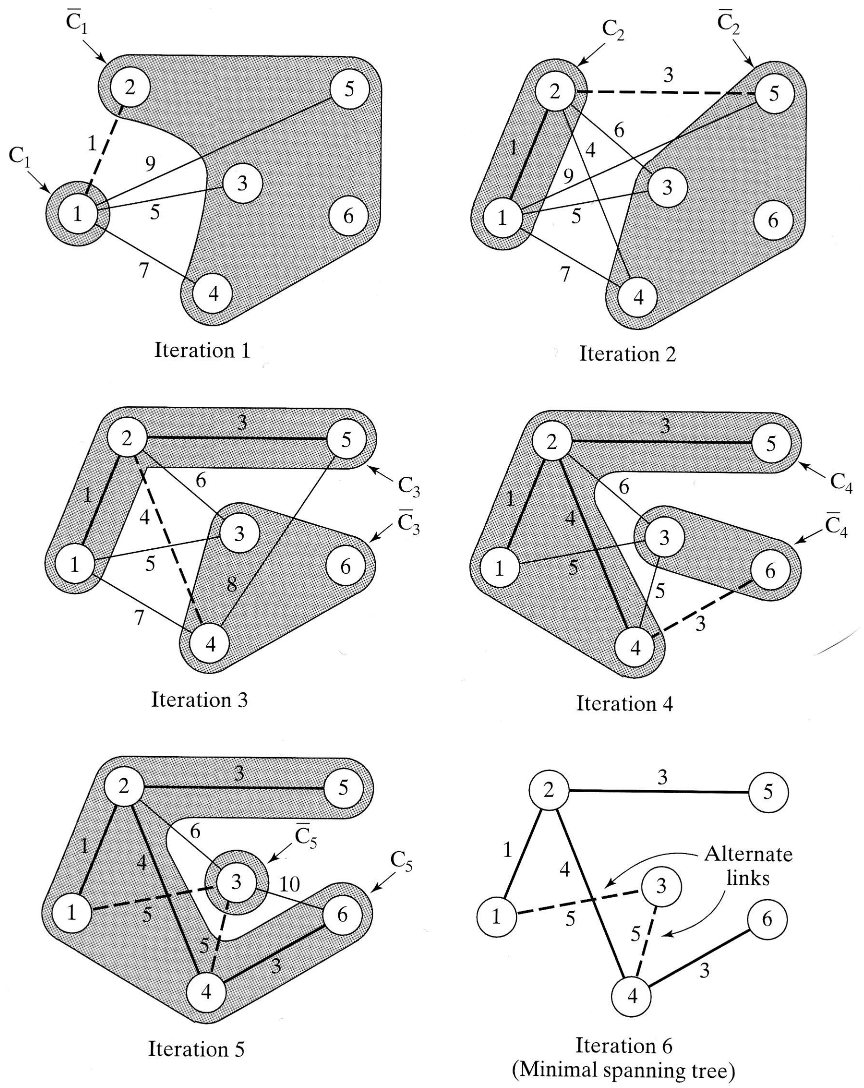



FIGURE 6.5
Solution iterations for Midwest TV Cable Company

You can use TORA to generate the iterations of the minimal spanning tree. From `Main menu`, select `Network models` ⇒ `Minimal spanning tree`. Next, from `SOLVE/MODIFY` menu, select `Solve problem` ⇒ `Go to output screen`. In the output screen, select a `Starting node` and then use `Next iteration` or `All iterations` to generate the successive iterations. You can restart the iterations by selecting a new `Starting node`. Figure 6.6 gives TORA output for Example 6.2-1 (file ch6ToraMinSpanEx6-2-1.txt).

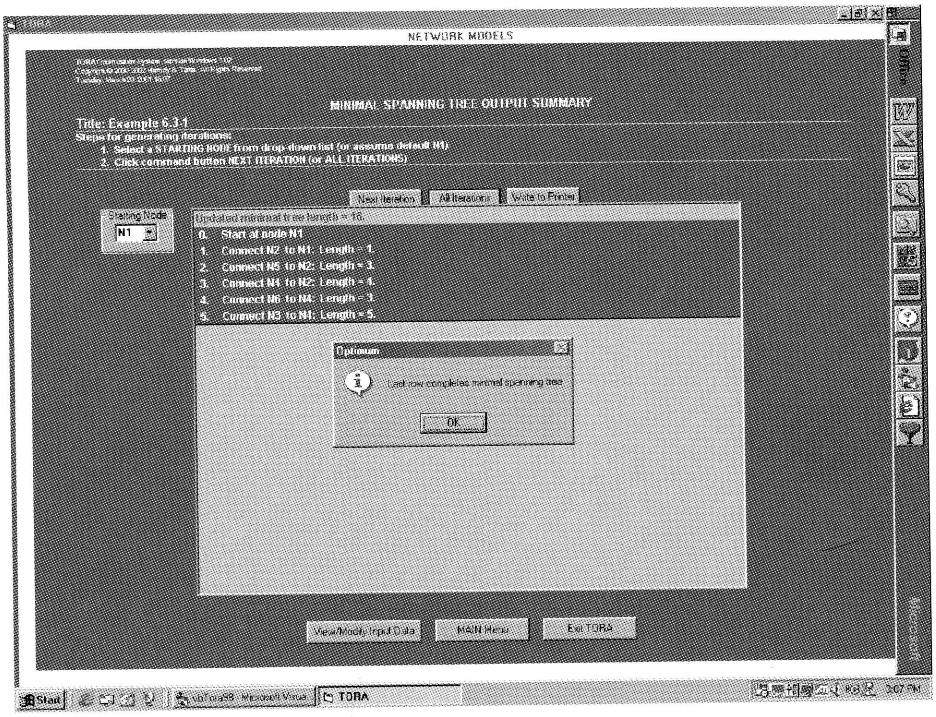



FIGURE 6.6
Output of the minimal spanning tree of Example 6.2-1

## PROBLEM SET 6.2A

1. Solve Example 6.2-1 starting at node 5 (instead of node 1), and show that the algorithm produces the same solution.
2. Determine the minimal spanning tree of the network of Example 6.2-1 under each of the following separate conditions:
   (a) Nodes 5 and 6 are linked by a 2-mile cable.
   (b) Nodes 2 and 5 cannot be linked.
   (c) Nodes 2 and 6 are linked by a 4-mile cable.
   (d) The cable between nodes 1 and 2 is 8 miles long.
   (e) Nodes 3 and 5 are linked by a 2-mile cable.
   (f) Node 2 cannot be linked directly to nodes 3 and 5.
3. In intermodal transportation, loaded truck trailers are shipped between railroad terminals by placing the trailer on special flatbed carts. Figure 6.7 shows the location of the main railroad terminals in the United States and the existing railroad tracks. The objective is to decide which tracks should be "revitalized" to handle the intermodal traffic. In particular, the Los Angeles (LA) terminal must be linked directly to Chicago (CH) to accommodate expected heavy traffic. Other than that, all the remaining terminals can be linked, directly or indirectly, such that the total length (in miles) of the selected tracks is minimized. Determine the segments of the railroad tracks that must be included in the revitalization program.

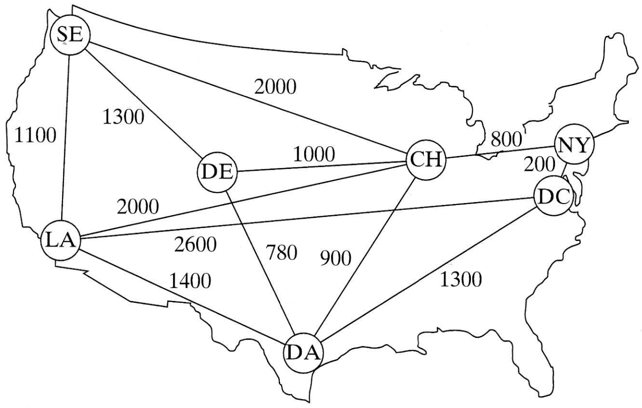



FIGURE 6.7
Network for Problem 3, Set 6.2a

4. Figure 6.8 gives the mileage of the feasible links connecting nine offshore natural gas wellheads with an inshore delivery point. Because the location of wellhead 1 is the closest to shore, it is equipped with sufficient pumping and storage capacity to pump the output of the remaining eight wells to the delivery point. Determine the minimum pipeline network that links the wellheads to the delivery point.

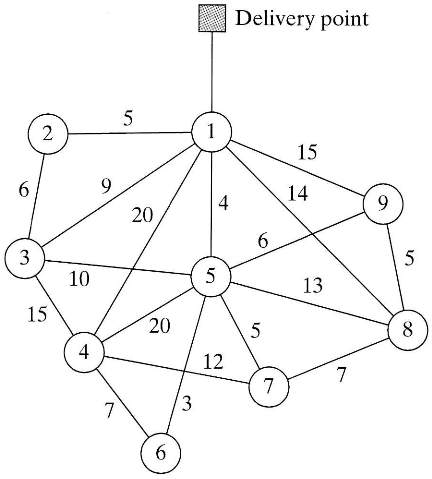



FIGURE 6.8
Network for Problem 4, Set 6.2a

5. In Figure 6.8 of Problem 4, suppose that the wellheads can be divided into two groups depending on gas pressure: a high-pressure group that includes wells 2, 3, 4, and 6; and a low-pressure group that includes wells 5, 7, 8, and 9. Because of pressure difference, wellheads from the two groups cannot be linked. At the same time, both groups must be connected to the delivery point through wellhead 1. Determine the minimum pipeline network for this situation.

6. Electro produces 15 electronic parts on 10 machines. The company wants to group the machines into cells designed to minimize the "dissimilarities" among the parts processed

in each cell. A measure of "dissimilarity," $d_{ij}$, among the parts processed on machines $i$ and $j$ can be expressed as

$$d_{ij} = 1 - \frac{n_{ij}}{n_{ij} + m_{ij}}$$

where $n_{ij}$ is the number of parts shared between machines $i$ and $j$, and $m_{ij}$ is the number of parts that are used by either machine $i$ or $j$ only.

The following table assigns the parts to machines:

| Machine | Assigned parts |
|---|---|
| 1 | 1, 6 |
| 2 | 2, 3, 7, 8, 9, 12, 13, 15 |
| 3 | 3, 5, 10, 14 |
| 4 | 2, 7, 8, 11, 12, 13 |
| 5 | 3, 5, 10, 11, 14 |
| 6 | 1, 4, 5, 9, 10 |
| 7 | 2, 5, 7, 8, 9, 10 |
| 8 | 3, 4, 15 |
| 9 | 4, 10 |
| 10 | 3, 8, 10, 14, 15 |

**(a)** Express the problem as a network model.

**(b)** Show that the determination of the cells can be based on the minimal spanning tree solution.

**(c)** For the data given in the preceding table, construct the two- and three-cell solutions.

## 6.3 SHORTEST-ROUTE PROBLEM

The shortest-route problem determines the shortest route between a source and destination in a transportation network. Other situations can be represented by the same model as illustrated by the following examples.

### 6.3.1 Examples of the Shortest-Route Applications

**Example 6.3-1 (Equipment Replacement)**

RentCar is developing a replacement plan for its car fleet for a 4-year planning horizon that starts January 1, 2001, and terminates December 31, 2004. At the start of each year, a decision is made as to whether a car should be kept in operation or replaced. A car must be in service a minimum of 1 year and a maximum of 3 years. The following table provides the replacement cost as a function of the year a car is acquired and the number of years in operation.

| *Equipment acquired at start of* | Replacement cost ($) for given years in operation | | |
|---|---|---|---|
| | *1* | *2* | *3* |
| 2001 | 4000 | 5400 | 9800 |
| 2002 | 4300 | 6200 | 8700 |
| 2003 | 4800 | 7100 | — |
| 2004 | 4900 | — | — |

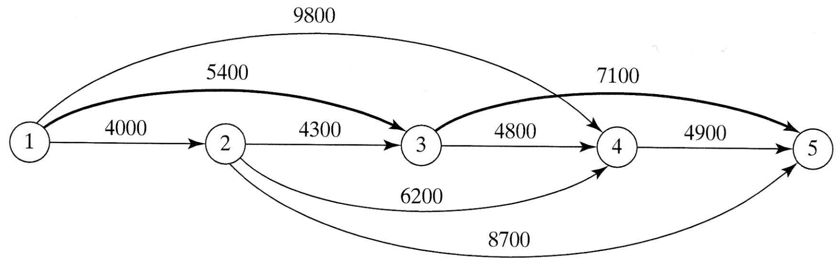



FIGURE 6.9
Equipment replacement problem as a shortest-route model

The problem can be formulated as a network in which nodes 1 to 5 represent the start of years 2001 to 2005. Arcs from node 1 (year 2001) can reach only nodes 2, 3, and 4 because a car must be in operation between 1 and 3 years. The arcs from the other nodes can be interpreted similarly. The length of each arc equals the replacement cost. The solution of the problem is equivalent to finding the shortest route between nodes 1 and 5.

Figure 6.9 shows the resulting network. Using TORA,[1] the shortest route (shown by the thick path) is $1 \rightarrow 3 \rightarrow 5$. The solution means that a car acquired at the start of 2001 (node 1) must be replaced after 2 years at the start of 2003 (node 3). The replacement car will then be kept in service until the end of 2004. The total cost of this replacement policy is \$12,500 (= \$5400 + \$7100).

### Example 6.3-2 (Most Reliable Route)

I. Q. Smart drives daily to work. Having just completed a course in network analysis, Smart is able to determine the shortest route to work. Unfortunately, the selected route is heavily patrolled by police, and with all the fines paid for speeding, the shortest route may not be the best choice. Smart has thus decided to choose a route that maximizes the probability of *not* being stopped by police.

The network in Figure 6.10 shows the possible routes between home and work, and the associated probabilities of not being stopped on each segment. The probability of not being stopped on the way to work is the product of the probabilities associated with the successive segments of the selected route. For example, the probability of not

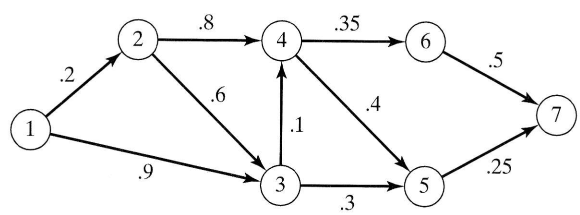



FIGURE 6.10
Most-reliable-route network model

[1]From `Main menu`, select `Network models` ⇒ `Shortest route`. From `SOLVE/MODIFY` menu, select `Solve problem` ⇒ `Shortest routes`.

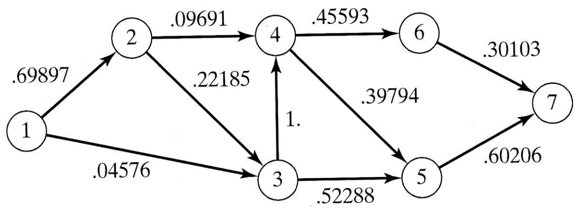



FIGURE 6.11
Most-reliable-route representation as a shortest-route model

receiving a fine on the route $1 \rightarrow 3 \rightarrow 5 \rightarrow 7$ is $.9 \times .3 \times .25 = .0675$. Smart's objective is to select the route that *maximizes* the probability of not being fined.

The problem can be formulated as a shortest-route model by using a logarithmic transformation that converts the product probability into the sum of the logarithms of probabilities—that is, if $p_{1k} = p_1 \times p_2 \times \ldots \times p_k$ is the probability of not being stopped, then $\log p_{1k} = \log p_1 + \log p_2 + \ldots + \log p_k$.

Mathematically, the maximization of $p_{1k}$ is equivalent to the maximization of $\log p_{1k}$. Because $\log p_{1k} \leq 0$, the maximization of $\log p_{1k}$ is, in turn, equivalent to the minimization of $-\log p_{1k}$. Using this transformation, the individual probabilities $p_j$ in Figure 6.10 are replaced with $-\log p_j$ for all $j$ in the network, thus yielding the shortest-route network in Figure 6.11.

Using TORA, nodes 1, 3, 5, and 7 define the shortest route in Figure 6.11 with a corresponding "length" of 1.1707 ($= -\log p_{17}$). Thus, the maximum probability of not being stopped is $p_{17} = .0675$.

**Example 6.3-3 (Three-Jug Puzzle)**

An 8-gallon jug is filled with fluid. Given two empty 5- and 3-gallon jugs, we want to divide the 8 gallons of fluid into two equal parts using the three jugs. No other measuring devices are allowed. What is the smallest number of pourings needed to achieve this result?

You probably can guess the solution of this puzzle. Nevertheless, the solution process can be systematized by representing the problem as a shortest-route problem.

A node is defined to represent the amount of fluid in the 8-, 5-, and 3-gallon jugs, respectively. This means that the network starts with node (8, 0, 0) and terminates with the desired solution node (4, 4, 0). A new node is generated from the current node by pouring fluid from one jug into another.

Figure 6.12 shows different routes that lead from start node (8, 0, 0) to end node (4, 4, 0). The arc between two successive nodes represents a single pouring, and hence can be assumed to have a length of 1 unit. The problem reduces to determining the shortest route between node (8, 0, 0) and node (4, 4, 0).

The optimal solution, given by the bottom path in Figure 6.12, requires 7 pourings.

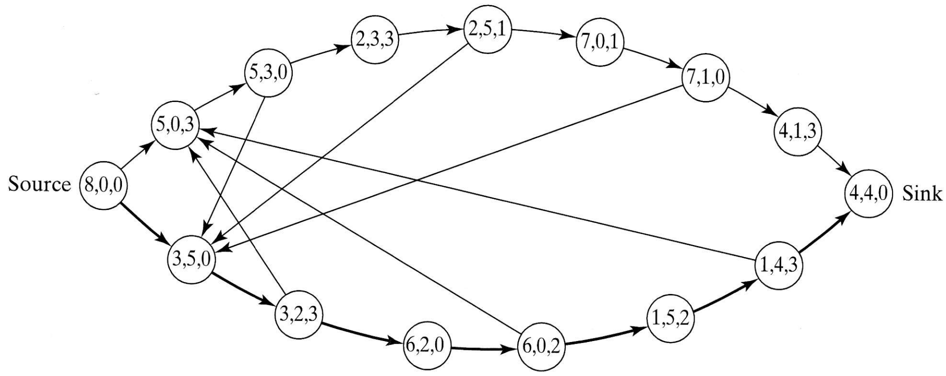



FIGURE 6.12

Three-jug puzzle representation as a shortest-route model

## PROBLEM SET 6.3A

1. Reconstruct the equipment replacement model of Example 6.3-1, assuming that a car must be kept in service at least 2 years, with a maximum service life of 4 years. The planning horizon is from the start of 2001 to the end of 2005. The following table provides the necessary data.

| | Replacement cost ($) for given years in operation | | |
|---|---|---|---|
| *Year acquired* | *2* | *3* | *4* |
| 2001 | 3800 | 4100 | 6800 |
| 2002 | 4000 | 4800 | 7000 |
| 2003 | 4200 | 5300 | 7200 |
| 2004 | 4800 | 5700 | — |
| 2005 | 5300 | — | — |

2. Figure 6.13 provides the communication network between two stations, 1 and 7. The probability that a link in the network will operate without failure is shown on each arc. Messages are sent from station 1 to station 7, and the objective is to determine the route that will maximize the probability of a successful transmission. Formulate the situation as a shortest-route model, and solve with TORA.
3. An old-fashioned electric toaster has two spring-loaded base-hinged doors. The two doors open outward in opposite directions away from the heating element. A slice of bread is toasted one side at a time by pushing open one of the doors with one hand and placing the slice with the other hand. After one side is toasted, the slice is turned over to get the other side toasted. It is desired to determine the sequence of operations (placing, toasting, turning, and removing) needed to toast three slices of bread in the shortest possible

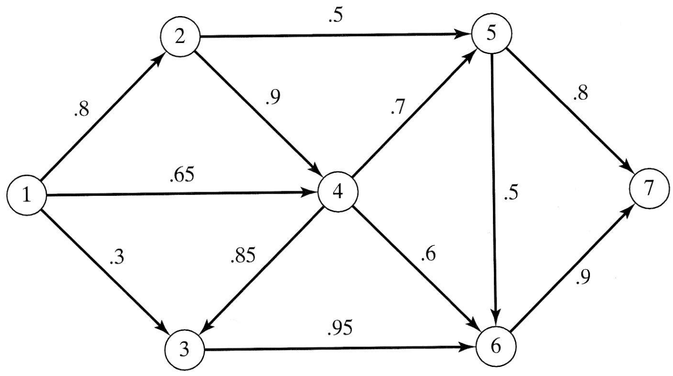



FIGURE 6.13
Network for Problem 2, Set 6.3a

time. Formulate the problem as a shortest-route model using the following elemental times for the different operations:

| Operation | Time (seconds) |
|---|---|
| Place one slice in either side | 3 |
| Toast one side | 30 |
| Turn slice already in toaster | 1 |
| Remove slice from either side | 3 |

4. *Production Planning.* DirectCo sells an item whose demand over the next 4 months is 100, 140, 210, and 180 units, respectively. The company can stock just enough supply to meet each month's demand, or it can overstock to meet the demand for two or more successive and consecutive months. In the latter case, a holding cost of $1.20 is charged per overstocked unit per month. DirectCo estimates the unit purchase prices for the next 4 months to be $15, $12, $10, and $14, respectively. A setup cost of $200 is incurred each time a purchase order is placed. The company wants to develop a purchasing plan that will minimize the total costs of ordering, purchasing, and holding the item in stock. Formulate the problem as a shortest-route model, and use TORA to find the optimum solution.
5. *Knapsack Problem.* A hiker has a 5-ft$^3$ backpack and needs to decide on the most valuable items to take on the hiking trip. There are three items from which to choose. Their volumes are 2, 3, and 4 ft$^3$, and the hiker estimates their associated values on a scale from 0 to 100 as 30, 50, and 70, respectively. Express the problem as a longest-route network, and find the optimal solution. (*Hint:* A node in the network may be defined as $[i, v]$, where $i$ is the item number considered for packing, and $v$ is the volume remaining immediately before the decision is made on $i$.)

### 6.3.2 Shortest-Route Algorithms

This section presents two algorithms for solving both cyclic (i.e., containing loops) and acyclic networks:

1. Dijkstra's algorithm
2. Floyd's algorithm

Dijkstra's algorithm is designed to determine the shortest routes between the source node and every other node in the network. Floyd's algorithm is general because it allows the determination of the shortest route between *any* two nodes in the network.

**Dijkstra's Algorithm.** Let $u_i$ be the shortest distance from source node 1 to node *i*, and define $d_{ij}$ ($\geq 0$) as the length of arc $(i, j)$. Then the algorithm defines the label for an immediately succeeding node *j* as

$$[u_j, i] = [u_i + d_{ij}, i], \ d_{ij} \geq 0$$

The label for the starting node is [0,—], indicating that the node has no predecessor.

Node labels in Dijkstra's algorithm are of two types: *temporary* and *permanent.* A temporary label is modified if a shorter route to a node can be found. At the point when no better routes can be found, the status of the temporary label is changed to permanent.

**Step 0.** Label the source node (node 1) with the *permanent* label [0,—]. Set $i = 1$.

**Step *i*.** **(a)** Compute the *temporary* labels $[u_i + d_{ij}, i]$ for each node *j* that can be reached from node *i*, *provided j is not permanently labeled.* If node *j* is already labeled with $[u_j, k]$ through another node *k* and if $u_i + d_{ij} < u_j$, replace $[u_j, k]$ with $[u_i + d_{ij}, i]$.

**(b)** If *all* the nodes have *permanent* labels, stop. Otherwise, select the label $[u_r, s]$ having the shortest distance ($=u_r$) among all the *temporary* labels (break ties arbitrarily). Set $i = r$ and repeat step *i*.

## Example 6.3-4

The network in Figure 6.14 gives the routes and their lengths in miles between city 1 (node 1) and four other cities (nodes 2 to 5). Determine the shortest routes between city 1 and each of the remaining four cities.

**Iteration 0.** Assign the *permanent* label [0,—] to node 1.

**Iteration 1.** Nodes 2 and 3 can be reached from (the last permanently labeled) node 1. Thus, the list of labeled nodes (temporary and permanent) becomes

| Node | Label | Status |
|---|---|---|
| **1** | **[0,—]** | **Permanent** |
| 2 | [0 + 100, 1] = [100, 1] | Temporary |
| 3 | [0 + 30, 1] = [30, 1] | Temporary |

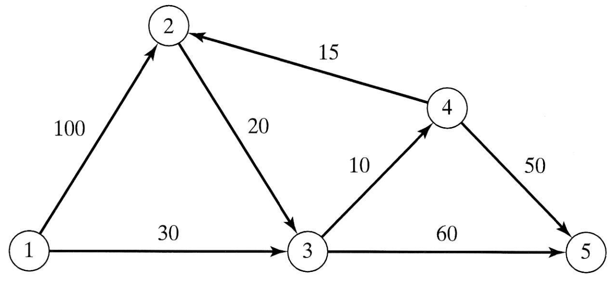



FIGURE 6.14

Network example for Dijkstra's shortest-route algorithm

For the two temporary labels [100, 1] and [30, 1], node 3 yields the smaller distance ($u_3 = 30$). Thus, the status of node 3 is changed to permanent.

**Iteration 2.** Nodes 4 and 5 can be reached from node 3, and the list of labeled nodes becomes

| Node | Label | Status |
|---|---|---|
| **1** | **[0,—]** | Permanent |
| 2 | [100, 1] | Temporary |
| **3** | **[30, 1]** | **Permanent** |
| 4 | [30 + 10, 3] = [40, 3] | Temporary |
| 5 | [30 + 60, 3] = [90, 3] | Temporary |

The status of the temporary label [40, 3] at node 4 is changed to permanent ($u_4 = 40$).

**Iteration 3.** Nodes 2 and 5 can be reached from node 4. Thus, the list of labeled nodes is updated as

| Node | Label | Status |
|---|---|---|
| **1** | **[0,—]** | Permanent |
| 2 | [40 + 15, 4] = [55, 4] | Temporary |
| **3** | **[30, 1]** | Permanent |
| **4** | **[40, 3]** | **Permanent** |
| 5 | [90, 3] or [40 + 50, 4] = [90, 4] | Temporary |

Node 2's temporary label [100, 1] in iteration 2 is changed to [55, 4] in iteration 3 to indicate that a shorter route has been found through node 4. Also, in iteration 3, node 5 has two alternative labels with the same distance $u_5 = 90$.

The list for iteration 3 shows that the label for node 2 is now permanent.

**Iteration 4.** Only node 3 can be reached from node 2. However, node 3 has a permanent label and cannot be relabeled. The new list of labels remains the same as in iteration 3 except that the label at node 2 is now permanent. This leaves node 5 as the only temporary label. Because node 5 does not lead to other nodes, its status is converted to permanent, and the process ends.

The computations of the algorithm can be carried out more easily on the network as Figure 6.15 demonstrates.

The shortest route between nodes 1 and any other node in the network is determined by starting at the desired destination node and backtracking through the nodes using the information given by the permanent labels. For example, the following sequence determines the shortest route from node 1 to node 2:

$$(2) \rightarrow [55, 4] \rightarrow (4) \rightarrow [40, 3] \rightarrow (3) \rightarrow [30, 1] \rightarrow (1)$$

Thus, the desired route is $1 \rightarrow 3 \rightarrow 4 \rightarrow 2$ with a total length of 55 miles.

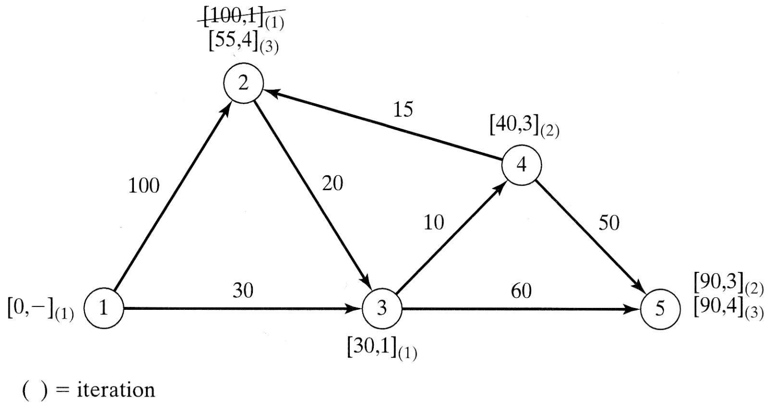



FIGURE 6.15
Dijkstra's labeling procedure

TORA can be used to generate Dijkstra's iterations. From the `SOLVE/MODIFY` menu, select `Solve problem` ⇒ `Iterations` ⇒ `Dijkstra's algorithm`. Figure 6.16 provides TORA's iterations output for Example 6.3-4 (file ch6ToraDijkstraEx6-3-4.txt).

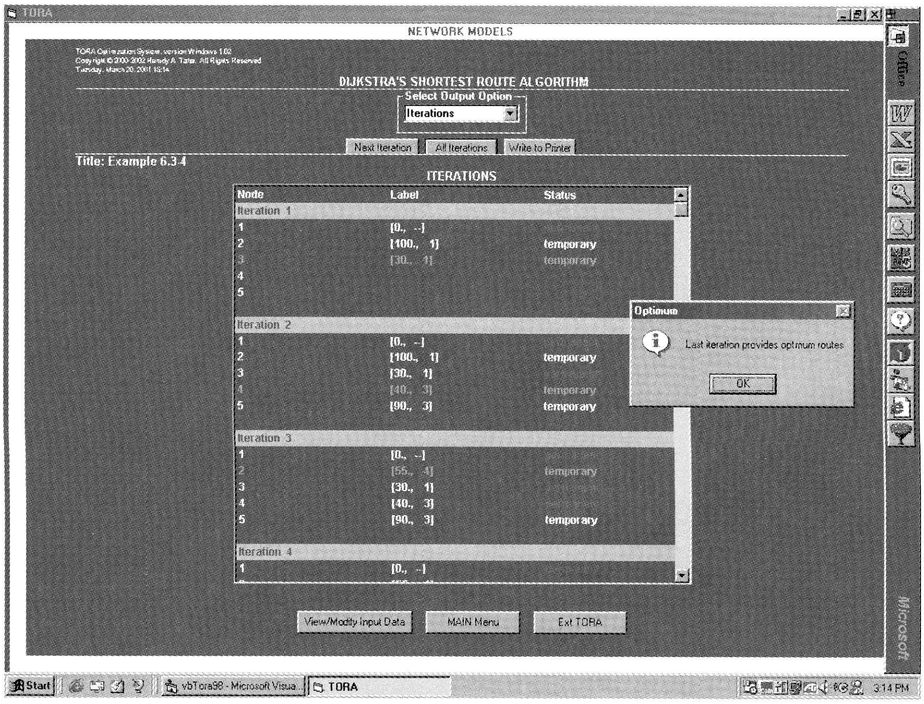
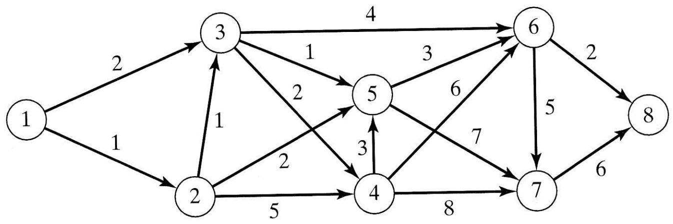



FIGURE 6.16
TORA Dijkstra iterations for Example 6.3-4

## PROBLEM SET 6.3B

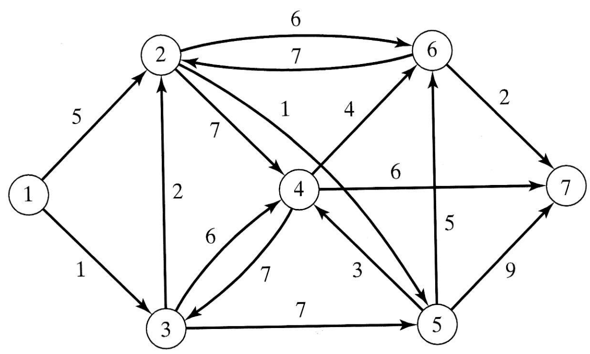
1. The network in Figure 6.17 gives the distances in miles between pairs of cities 1, 2, ..., and 8. Use Dijkstra's algorithm to find the shortest route between the following cities:
   **(a)** Cities 1 and 8
   **(b)** Cities 1 and 6
   **(c)** Cities 4 and 8
   **(d)** Cities 2 and 6

FIGURE 6.17
Network for Problem 1, Set 6.3b




2. Use Dijkstra's algorithm to find the shortest route between node 1 and every other node in the network of Figure 6.18.

FIGURE 6.18
Network for Problem 2, Set 6.3b




3. Use Dijkstra's algorithm to determine the optimal solution of each of the following problems:
   **(a)** Problem 1, Set 6.3a
   **(b)** Problem 2, Set 6.3a
   **(c)** Problem 4, Set 6.3a

**Floyd's Algorithm.** Floyd's algorithm is more general than Dijkstra's because it determines the shortest route between *any* two nodes in the network. The algorithm represents an $n$-node network as a square matrix with $n$ rows and $n$ columns. Entry $(i,j)$ of the matrix gives the distance $d_{ij}$ from node $i$ to node $j$, which is finite if $i$ is linked directly to $j$, and infinite otherwise.

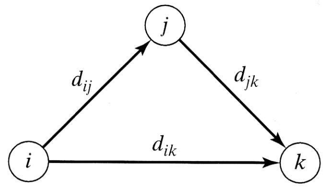



FIGURE 6.19
Floyd's triple operation

The idea of Floyd's algorithm is straightforward. Given three nodes $i$, $j$, and $k$ in Figure 6.19 with the connecting distances shown on the three arcs, it is shorter to reach $k$ from $i$ passing through $j$ if

$$d_{ij} + d_{jk} < d_{ik}$$

In this case, it is optimal to replace the direct route from $i \rightarrow k$ with the indirect route $i \rightarrow j \rightarrow k$. This **triple operation** exchange is applied systematically to the network using the following steps:

**Step 0.** Define the starting distance matrix $D_0$ and node sequence matrix $S_0$ as given below. The diagonal elements are marked with (—) to indicates that they are blocked. Set $k = 1$.

$D_0 =$

| | 1 | 2 | ... | $j$ | ... | $n$ |
|---|---|---|---|---|---|---|
| 1 | — | $d_{12}$ | ... | $d_{ij}$ | ... | $d_{1n}$ |
| 2 | $d_{21}$ | — | ... | $d_{2j}$ | ... | $d_{2n}$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| $i$ | $d_{i1}$ | $d_{i2}$ | ... | $d_{ij}$ | ... | $d_{in}$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| $n$ | $D_{n1}$ | $d_{n2}$ | ... | $d_{nj}$ | ... | — |

$S_0 =$

| | 1 | 2 | ... | $j$ | ... | $n$ |
|---|---|---|---|---|---|---|
| 1 | — | 2 | ... | $j$ | ... | $n$ |
| 2 | 1 | — | ... | $j$ | ... | $n$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| $i$ | 1 | 2 | ... | $j$ | ... | $n$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| $n$ | 1 | 2 | ... | $j$ | ... | — |

**General Step $k$.** Define row $k$ and column $k$ as *pivot row* and *pivot column*. Apply the *triple operation* to each element $d_{ij}$ in $D_{k-1}$ for all $i$ and $j$. If the condition

$$d_{ik} + d_{kj} < d_{ij}, \ (i \neq k, \ j \neq k, \ \text{and } i \neq j)$$

is satisfied, make the following changes:

**(a)** Create $D_k$ by replacing $d_{ij}$ in $D_{k-1}$ with $d_{ik} + d_{kj}$.

**(b)** Create $S_k$ by replacing $s_{ij}$ in $S_{k-1}$ with $k$. Set $k = k + 1$, and repeat step $k$.

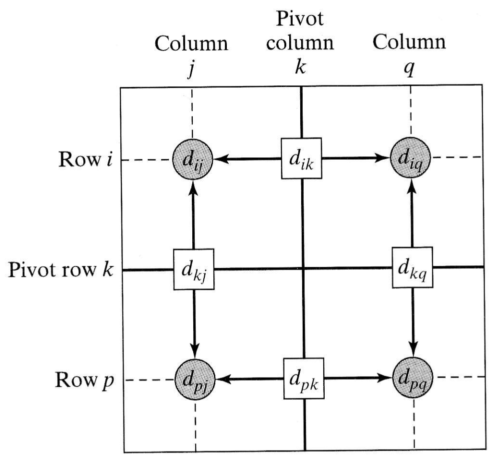



FIGURE 6.20
Implementation of triple operation in matrix form

Step $k$ of the algorithm can be explained by representing $D_{k-1}$ as shown in Figure 6.20. Here, row $k$ and column $k$ define the current pivot row and column. Row $i$ represents any of the rows $1, 2, \ldots,$ and $k - 1$, and row $p$ represents any of the rows $k + 1, k + 2, \ldots,$ and $n$. Similarly, column $j$ represents any of the columns $1, 2, \ldots,$ and $k - 1$, and column $q$ represents any of the columns $k + 1, k + 2, \ldots,$ and $n$. With the *triple operation*, if the sum of the elements on the pivot row and the pivot column (shown by squares) is smaller than the associated intersection element (shown by a circle), then it is optimal to replace the intersection distance by the sum of the pivot distances.

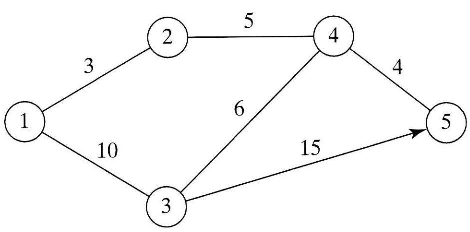
After $n$ steps, we can determine the shortest route between nodes $i$ and $j$ from the matrices $D_n$ and $S_n$ using the following rules:

1. From $D_n$, $d_{ij}$ gives the shortest distance between nodes $i$ and $j$.
2. From $S_n$, determine the intermediate node $k = s_{ij}$ that yields the route $i \rightarrow k \rightarrow j$. If $s_{ik} = k$ and $s_{kj} = j$, stop; all the intermediate nodes of the route have been found. Otherwise, repeat the procedure between nodes $i$ and $k$, and between nodes $k$ and $j$.

## Example 6.3-5

For the network in Figure 6.21, find the shortest routes between every two nodes. The distances (in miles) are given on the arcs. Arc (3,5) is directional so that no traffic is allowed from node 5 to node 3. All the other arcs allow traffic in both directions.

FIGURE 6.21
Network for Example 6.3-5



**Iteration 0.** The matrices $D_0$ and $S_0$ give the initial representation of the network. $D_0$ is symmetrical except that $d_{53} = \infty$ because no traffic is allowed from node 5 to node 3.

$D_0$

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | — | 3 | 10 | ∞ | ∞ |
| 2 | 3 | — | ∞ | 5 | ∞ |
| 3 | 10 | ∞ | — | 6 | 15 |
| 4 | ∞ | 5 | 6 | — | 4 |
| 5 | ∞ | ∞ | ∞ | 4 | — |

$S_0$

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | — | 2 | 3 | 4 | 5 |
| 2 | 1 | — | 3 | 4 | 5 |
| 3 | 1 | 2 | — | 4 | 5 |
| 4 | 1 | 2 | 3 | — | 5 |
| 5 | 1 | 2 | 3 | 4 | — |

**Iteration 1.** Set $k = 1$. The pivot row and column are shown by the lightly shaded first row and first column in the $D_0$-matrix. The darker cells, $d_{23}$ and $d_{32}$, are the only ones that can be improved by the *triple operation.* Thus, $D_1$ and $S_1$ are obtained from $D_0$ and $S_0$ in the following manner:

1. Replace $d_{23}$ with $d_{21} + d_{13} = 3 + 10 = 13$ and set $s_{23} = 1$.
2. Replace $d_{32}$ with $d_{31} + d_{12} = 10 + 3 = 13$ and set $s_{32} = 1$.

These changes are shown in bold in matrices $D_1$ and $S_1$.

$D_1$

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | — | 3 | 10 | ∞ | ∞ |
| 2 | 3 | — | **13** | 5 | ∞ |
| 3 | 10 | **13** | — | 6 | 15 |
| 4 | ∞ | 5 | 6 | — | 4 |
| 5 | ∞ | ∞ | ∞ | 4 | — |

$S_1$

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | — | 2 | 3 | 4 | 5 |
| 2 | 1 | — | **1** | 4 | 5 |
| 3 | 1 | **1** | — | 4 | 5 |
| 4 | 1 | 2 | 3 | — | 5 |
| 5 | 1 | 2 | 3 | 4 | — |

**Iteration 2.** Set $k = 2$, as shown by the lightly shaded row and column in $D_1$. The *triple operation* is applied to the darker cells in $D_1$ and $S_1$. The resulting changes are shown in bold in $D_2$ and $S_2$.

$D_2$

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | — | 3 | 10 | **8** | ∞ |
| 2 | 3 | — | 13 | 5 | ∞ |
| 3 | 10 | 13 | — | 6 | 15 |
| 4 | **8** | 5 | 6 | — | 4 |
| 5 | ∞ | ∞ | ∞ | 4 | — |

$S_2$

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | — | 2 | 3 | **2** | 5 |
| 2 | 1 | — | 1 | 4 | 5 |
| 3 | 1 | 1 | — | 4 | 5 |
| 4 | **2** | 2 | 3 | — | 5 |
| 5 | 1 | 2 | 3 | 4 | — |

**Iteration 3.** Set $k = 3$, as shown by the shaded row and column in $D_2$. The new matrices are given by $D_3$ and $S_3$.

$D_3$

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | — | 3 | 10 | 8 | **25** |
| 2 | 3 | — | 13 | 5 | **28** |
| 3 | 10 | 13 | — | 6 | 15 |
| 4 | 8 | 5 | 6 | — | 4 |
| 5 | ∞ | ∞ | ∞ | 4 | — |

$S_3$

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | — | 2 | 3 | 2 | **3** |
| 2 | 1 | — | 1 | 4 | **3** |
| 3 | 1 | 1 | — | 4 | 5 |
| 4 | 2 | 2 | 3 | — | 5 |
| 5 | 1 | 2 | 3 | 4 | — |

**Iteration 4.** Set $k = 4$, as shown by the lightly-shaded row and column in $D_3$. The new matrices are given by $D_4$ and $S_4$.

$D_4$

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | — | 3 | 10 | 8 | **12** |
| 2 | 3 | — | **11** | 5 | **9** |
| 3 | 10 | **11** | — | 6 | **10** |
| 4 | 8 | 5 | 6 | — | 4 |
| 5 | **12** | **9** | **10** | 4 | — |

$S_4$

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | — | 2 | 3 | 2 | **4** |
| 2 | 1 | — | **4** | 4 | **4** |
| 3 | 1 | **4** | — | 4 | **4** |
| 4 | 2 | 2 | 3 | — | 5 |
| 5 | **4** | **4** | **4** | 4 | — |

**Iteration 5.** Set $k = 5$, as shown by the shaded row and column in $D_4$. No further improvements are possible in this iteration. Hence, $D_5$ and $S_5$ are the same as $D_4$ and $S_4$.

The final matrices $D_5$ and $S_5$ contain all the information needed to determine the shortest route between any two nodes in the network. For example, consider determining the shortest route from node 1 to node 5. First, the associated shortest distance is given by $d_{15} = 12$ miles. To determine the associated route, recall that a segment $(i, j)$ represents a direct link only if $s_{ij} = j$. Otherwise, $i$ and $j$ are linked through at least one other intermediate node. Because $s_{15} = 4$, the route is initially given as $1 \rightarrow 4 \rightarrow 5$. Now, because $s_{14} = 2 \neq 4$, the segment $(1, 4)$ is not a *direct* link, and $1 \rightarrow 4$ must be replaced with $1 \rightarrow 2 \rightarrow 4$, and the route $1 \rightarrow 4 \rightarrow 5$ now becomes $1 \rightarrow 2 \rightarrow 4 \rightarrow 5$. Next, because $s_{12} = 2$, $s_{24} = 4$, and $s_{45} = 5$, the route $1 \rightarrow 2 \rightarrow 4 \rightarrow 5$ needs no further "dissecting" and the process ends.

As in Dijkstra's algorithm, TORA can be used to generate Floyd's iterations. From the `SOLVE/MODIFY` menu, select `Solve problem` ⇒ `Iterations` ⇒ `Floyd's algorithm`. Figure 6.22 illustrates TORA's output for Floyd's Example 6.3-5 (file ch6ToraFloydEx6-3-5.txt).

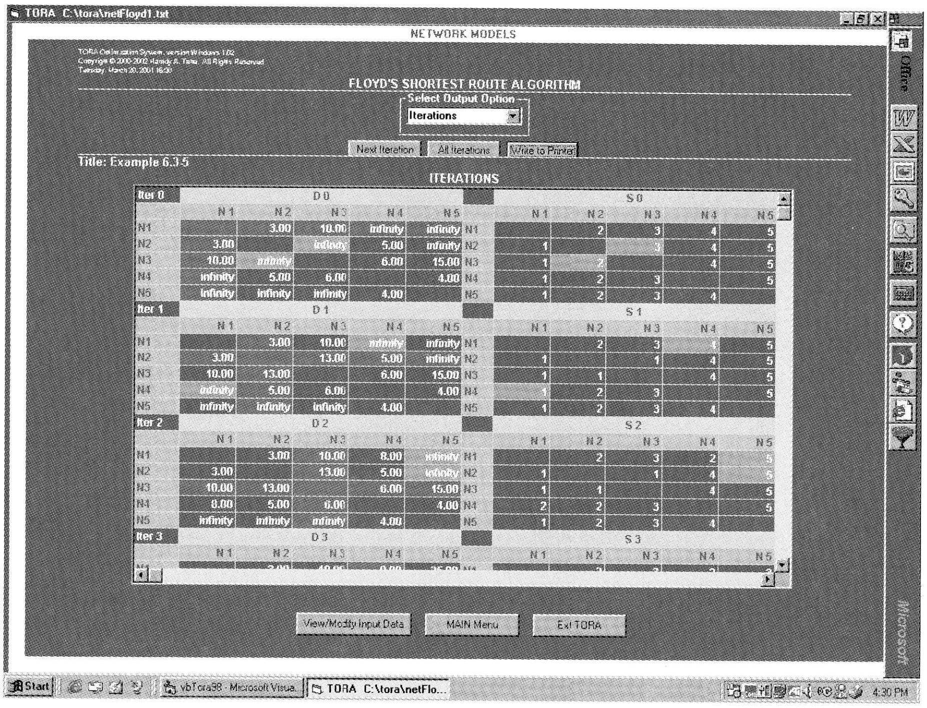



FIGURE 6.22
TORA Floyd iterations for Example 6.3-5

## PROBLEM SET 6.3C

1. In Example 6.3-5, use Floyd's algorithm to determine the shortest routes between each of the following pairs of nodes:
   - **(a)** From node 5 to node 1
   - **(b)** From node 3 to node 5
   - **(c)** From node 5 to node 3
   - **(d)** From node 5 to node 2
2. Apply Floyd's algorithm to the network in Figure 6.23. Arcs (7, 6) and (6, 4) are unidirectional, and all the distances are in miles. Determine the shortest route between the following pairs of nodes:
   - **(a)** From node 1 to node 7
   - **(b)** From node 7 to node 1
   - **(c)** From node 6 to node 7
3. The Tell-All mobile phone company services six geographical areas. The satellite distances (in miles) among the six areas are given in Figure 6.24. Tell-All needs to determine the most efficient message routes that should be established between each two areas in the network.
4. Six kids—Joe, Kay, Jim, Bob, Rae, and Kim—play a variation of the game of hide and seek. The hiding place of a child is known only to a select few of the other children. A

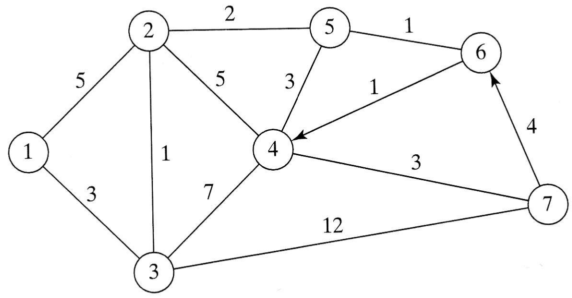



FIGURE 6.23
Network for Problem 2, Set 6.3c

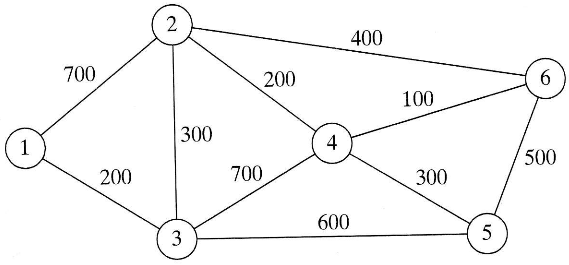



FIGURE 6.24
Network for Problem 3, Set 6.3c

child is then paired with another with the objective of finding his or her hiding place. This may be achieved through a chain of other kids who eventually will lead to discovering where the designated child is hiding. For example, suppose that Joe needs to find Kim and that Joe knows where Jim is hiding, who in turn knows where Kim is. Thus, Joe can find Kim by first finding Jim, who in turn will lead Joe to Kim. The following list provides the whereabouts of the children:

Joe knows the hiding places of Bob and Kim.

Kay knows the hiding places of Bob, Jim, and Rae.

Jim and Bob know the hiding place of Kay only.

Rae knows where Kim is hiding.

Kim knows where Joe and Bob are hiding.

Devise a plan for each child to find every other child through the smallest number of contacts. What is the largest number of contacts?

### 6.3.3 Linear Programming Formulation of the Shortest-Route Problem

This section provides two LP formulations for the shortest-route problem. The formulations are general in the sense that they can be used to find the shortest route between any two nodes in the network. In this regard, the LP formulations are equivalent to Floyd's algorithm.

Suppose that the shortest-route network includes $n$ nodes and that we desire to determine the shortest route between any two nodes $s$ and $t$ in the network.

**Formulation 1:** This formulation assumes that an external one unit of flow enters the network at node $s$ and leaves it at node $t$, where $s$ and $t$ are the two target nodes between which we seek to determine the shortest route.

Define

$$x_{ij} = \text{amount of flow in arc } (i, j), \text{ for all feasible } i \text{ and } j$$

$$c_{ij} = \text{length of arc } (i, j), \text{ for all feasible } i \text{ and } j$$

Because only *one* unit of flow can be in any arc at any one time, the variable $x_{ij}$ must assume binary values (0 or 1) only. Thus, the objective function of the linear program becomes

$$\text{Minimize } z = \sum_{\text{all defined arcs}(i,\, j)} c_{ij}x_{ij}$$

There is one constraint that represents the conservation of flow at each node—that is, for any node $j$,

$$\text{Total input flow} = \text{Total output flow}$$

**Formulation 2:** The second formulation is actually the dual problem of the LP in Formulation 1. Because the number of constraints in Formulation 1 equals the number of nodes, the dual problem will have as many variables as the number of nodes in the network. Also, all the dual variables must be unrestricted because all the constraints in Formulation 1 are equations.

Let

$$y_j = \text{dual constraint associated with node } j$$

Given $s$ and $t$ are the start and terminal nodes of the network, the dual problem is defined as

$$\text{Maximize } z = y_t - y_s$$

subject to

$$y_j - y_i \leq c_{ij}, \text{ for all feasible } i \text{ and } j$$

$$\text{all } y_i \text{ and } y_j \text{ unrestricted in sign}$$

### Example 6.3-6

Consider the shortest route network of Example 6.3-4. Suppose that we want to determine the shortest route from node 1 to node 2; that is, $s = 1$ and $t = 2$. Figure 6.25 shows how the unit of flow enters at node 1 and leaves at node 2.

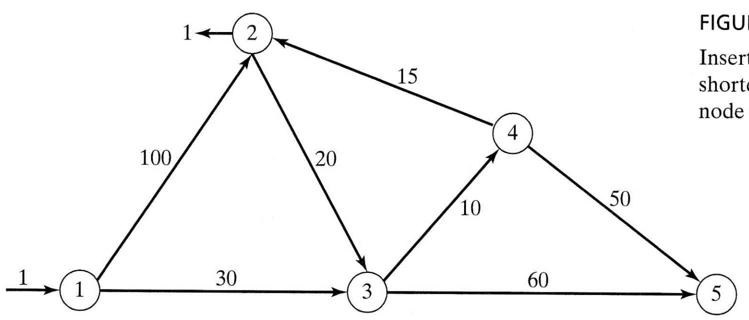



FIGURE 6.25

Insertion of unit flow to determine shortest route between node $s = 1$ and node $t = 2$

Using Formulation 1, the associated LP is listed below.

| | $x_{12}$ | $x_{13}$ | $x_{23}$ | $x_{34}$ | $x_{35}$ | $x_{42}$ | $x_{45}$ | |
|---|---|---|---|---|---|---|---|---|
| Minimize $z$ = | 100 | 30 | 20 | 10 | 60 | 15 | 50 | |
| Node 1 | −1 | −1 | | | | | | = −1 |
| Node 2 | 1 | | −1 | | | 1 | | = 1 |
| Node 3 | | 1 | 1 | −1 | −1 | | | = 0 |
| Node 4 | | | | 1 | | −1 | −1 | = 0 |
| Node 5 | | | | | 1 | | 1 | = 0 |

The constraints represent flow conservation at each node. For example, at node 2, "input flow = output flow" yields $x_{12} + x_{42} = 1 + x_{23}$. Note that one of the constraints is always redundant. For example, adding the last four constraints simultaneously yields $x_{12} + x_{13} = 1$, which is the same as constraint 1.

The optimal solution (obtained by TORA)[2] is

$$z = 55,\ x_{13} = 1,\ x_{34} = 1,\ x_{42} = 1$$

This solution gives the shortest route from node 1 to node 2 as $1 \rightarrow 3 \rightarrow 4 \rightarrow 2$ and the associated distance is $z = 55$ (miles).

To use Formulation 2, the dual problem associated with the LP above is given as

$$\text{Maximize } z = y_2 - y_1$$

subject to

$$\begin{aligned}
y_2 - y_1 &\le 100 \text{ (Route 1-2)} \\
y_3 - y_1 &\le 30 \text{ (Route 1-3)} \\
y_3 - y_2 &\le 20 \text{ (Route 2-3)} \\
y_4 - y_3 &\le 10 \text{ (Route 3-4)} \\
y_5 - y_3 &\le 60 \text{ (Route 3-5)} \\
y_2 - y_4 &\le 15 \text{ (Route 4-2)} \\
y_5 - y_4 &\le 50 \text{ (Route 4-5)} \\
y_1, y_2, \ldots, y_5 &\text{ unrestricted}
\end{aligned}$$

Although the dual problem given above is a pure mathematical definition derived from the primal problem, we actually can interpret the problem in a logical manner. Define

$$y_i = \text{Distance to node } i$$

[2]TORA does not accept a negative right-hand side. You can get around this inconvenience by selecting the redundant constraint as the one having the negative right-hand side, then make it redundant by changing = to ≤ and setting the right-hand side to a very large value. Another trick is to add a new variable whose upper and lower bounds equal 1, effectively forcing it to equal 1 in any solution. The constraint coefficients of the new variable equal those of the current right-hand side, but with opposite sign. The right-hand side of the "new" problem must be changed to zero for all the constraints (see file ch6ToraLpShortRoute Ex6-3-6.txt).

With this definition, the shortest distance from the start node 1 to the terminal node 2 is determined by maximizing $y_2 - y_1$. The constraint associated with route $(i, j)$ says that the distance from node $i$ to node $j$ cannot exceed the direct length of that route. It can be less if node $j$ can be reached from node $i$ through other nodes that provide a shorter path. For example, the distance from node 1 to node 2 is at most 100. With the definition of $y_i$ as the distance to node $i$, we can assume that all the variables are nonnegative (instead of being unrestricted). We can also assume that $y_1 = 0$ as the distance to node 1.

Based on the discussion above, and assuming that all the variables are nonnegative, the optimum solution is given as

$$z = 55,\ y_1 = 0,\ y_2 = 55,\ y_3 = 30,\ y_4 = 40,\ y_5 = 0$$

The value of $z = 55$ gives the shortest distance from node 1 to node 2, which also equals $y_2 - y_1 = 55 - 0 = 55$.

The determination of the route itself from this solution is somewhat tricky. We note that the solution satisfies *in equation form* the constraints of routes 1-3, 3-4, and 4-2 because their slacks equal zero—that is, $y_3 - y_1 = 30$, $y_4 - y_3 = 10$, and $y_2 - y_4 = 15$. This result identifies the shortest route as $1 \rightarrow 3 \rightarrow 4 \rightarrow 2$.

Another way for identifying the constraints that are satisfied in equation form is to consult the dual solution of the LP of Formulation 2. Any constraint that has a nonzero dual value must be satisfied in equation form (see Section 4.2.4). The following table pairs the routes (constraints) with their associated dual values.

| Route (constraint) | 1-2 | 1-3 | 2-3 | 3-4 | 3-5 | 4-2 | 4-5 |
|---|---|---|---|---|---|---|---|
| Associated dual value | 0 | 1 | 0 | 1 | 0 | 1 | 0 |

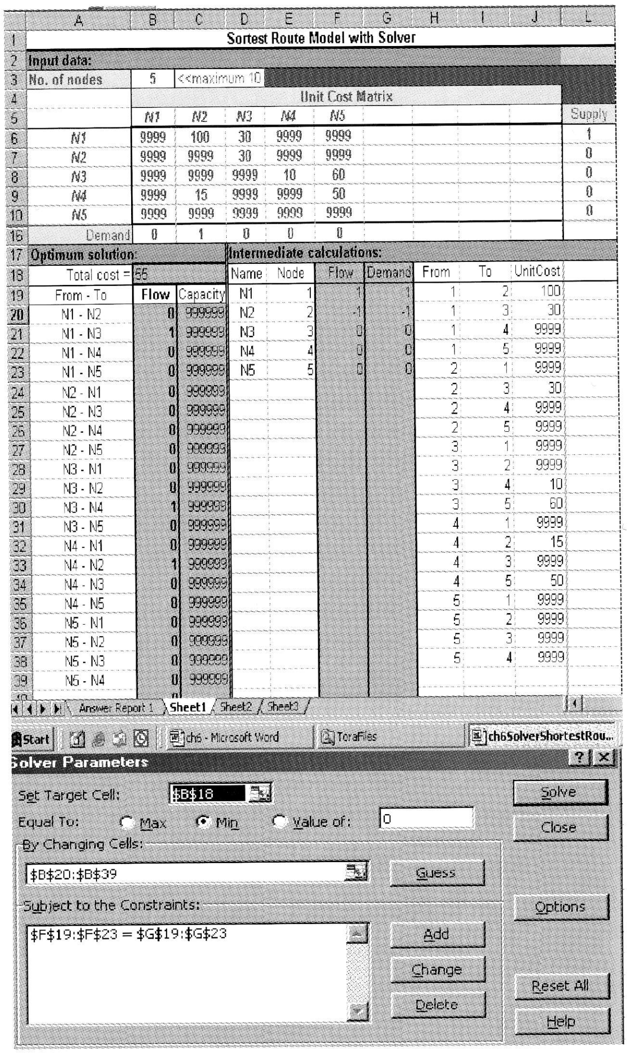
## PROBLEM SET 6.3D

1. In Example 6.3-6, use the two LP formulations to determine the shortest routes between the following pairs of nodes:
   **(a)** Node 1 to node 5.
   **(b)** Node 2 to node 5.

### 6.3.4 Excel Spreadsheet Solution of the Shortest-Route Problem

The Excel spreadsheet developed for the general transportation model (Section 5.3.3) can be modified readily to find the shortest route between two nodes. The spreadsheet is based on Formulation 1, Section 6.3.3, and is designed for problems with a maximum of 10 nodes. Figure 6.26 shows the application of the spreadsheet to Example 6.3-4 (file ch6SolverShortestRoute.xls). The distance matrix resides in cells B6:K15.[3] An infinite distance (= 9999, or any relatively large value) is entered for nonexisting arcs. Because we are seeking the shortest route between nodes 1 and 2, the supply amount for node 1 and the demand amount for node 2 is 1 unit. A zero amount is entered for the remaining supply and demand entries.

[3]In Figure 6.26, rows 11 through 15 and column K are hidden to conserve space.



| | A | B | C | D | E | F | G | H | I | J | L |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Sortest Route Model with Solver | | | | | | | | | | |
| 2 | Input data: | | | | | | | | | | |
| 3 | No. of nodes | 5 | <<maximum 10 | | | | | | | | |
| 4 | | | | | Unit Cost Matrix | | | | | | |
| 5 | | N1 | N2 | N3 | N4 | N5 | | | | | Supply |
| 6 | N1 | 9999 | 100 | 30 | 9999 | 9999 | | | | | 1 |
| 7 | N2 | 9999 | 9999 | 30 | 9999 | 9999 | | | | | 0 |
| 8 | N3 | 9999 | 9999 | 9999 | 10 | 60 | | | | | 0 |
| 9 | N4 | 9999 | 15 | 9999 | 9999 | 50 | | | | | 0 |
| 10 | N5 | 9999 | 9999 | 9999 | 9999 | 9999 | | | | | 0 |
| 16 | Demand | 0 | 1 | 0 | 0 | 0 | | | | | |
| 17 | Optimum solution: | | | Intermediate calculations: | | | | | | | |
| 18 | Total cost = | 55 | | Name | Node | Flow | Demand | From | To | UnitCost | |
| 19 | From - To | Flow | Capacity | N1 | 1 | 1 | 1 | 1 | 2 | 100 | |
| 20 | N1 - N2 | 0 | 999999 | N2 | 2 | -1 | -1 | 1 | 3 | 30 | |
| 21 | N1 - N3 | 1 | 999999 | N3 | 3 | 0 | 0 | 1 | 4 | 9999 | |
| 22 | N1 - N4 | 0 | 999999 | N4 | 4 | 0 | 0 | 1 | 5 | 9999 | |
| 23 | N1 - N5 | 0 | 999999 | N5 | 5 | 0 | 0 | 2 | 1 | 9999 | |
| 24 | N2 - N1 | 0 | 999999 | | | | | 2 | 3 | 30 | |
| 25 | N2 - N3 | 0 | 999999 | | | | | 2 | 4 | 9999 | |
| 26 | N2 - N4 | 0 | 999999 | | | | | 2 | 5 | 9999 | |
| 27 | N2 - N5 | 0 | 999999 | | | | | 3 | 1 | 9999 | |
| 28 | N3 - N1 | 0 | 999999 | | | | | 3 | 2 | 9999 | |
| 29 | N3 - N2 | 0 | 999999 | | | | | 3 | 4 | 10 | |
| 30 | N3 - N4 | 1 | 999999 | | | | | 3 | 5 | 60 | |
| 31 | N3 - N5 | 0 | 999999 | | | | | 4 | 1 | 9999 | |
| 32 | N4 - N1 | 0 | 999999 | | | | | 4 | 2 | 15 | |
| 33 | N4 - N2 | 1 | 999999 | | | | | 4 | 3 | 9999 | |
| 34 | N4 - N3 | 0 | 999999 | | | | | 4 | 5 | 50 | |
| 35 | N4 - N5 | 0 | 999999 | | | | | 5 | 1 | 9999 | |
| 36 | N5 - N1 | 0 | 999999 | | | | | 5 | 2 | 9999 | |
| 37 | N5 - N2 | 0 | 999999 | | | | | 5 | 3 | 9999 | |
| 38 | N5 - N3 | 0 | 999999 | | | | | 5 | 4 | 9999 | |
| 39 | N5 - N4 | 0 | 999999 | | | | | | | | |



FIGURE 6.26

Excel Solver solution of the shortest route between nodes 1 and 2 in Example 6.3-4

Once the unit cost and supply/demand data are entered, the remainder of the spreadsheet (*intermediate calculations* and *optimum solution* sections) is generated automatically. Solver parameters must correspond to the input data of the problem as shown in highlighted columns B, C, F, and G. Column B specifies the changing cells (arcs flow) of the problem (cells B20:B39). Column C specifies the capacities of the arcs of the network (cells C20:C39). In the shortest-route model, these capacities do not play a role in the computations and hence are infinite (=999999). The constraints of the model represent the balance equation for each node. Cells F19:F23 define the left-hand side and cells G19:G23 represent the right-hand side of the flow equations. As explained in Section 5.3.3, SUMIF is used to generate the proper net flow in each node using the information in columns I and J. These calculations are automated by the spreadsheet. Thus, all you need to do after entering the input data is to update Changing Cells and Constraints specifications of Solver to match the input data. The Target Cell remains the same for all input data. In Example 6.3-4, we have

```
Changing Cells: B20:B39
Constraints: F19:F23=G19:G23
```

The output in Figure 6.26 yields the solution (N1-N3 = 1, N3-N4 = 1, N4-N2 = 1) with a total distance of 55 miles. This means that the optimal route is $1 \rightarrow 3 \rightarrow 4 \rightarrow 2$.

## PROBLEM SET 6.3E

1. Modify spreadsheet ch6SolverShortestRoute.xls (applied to Example 6.3-4) to find the shortest route between the following pairs of nodes:
   **(a)** Node 1 to node 5
   **(b)** Node 4 to node 3
2. Adapt spreadsheet ch6SolverShortestRoute.xls for Problem 2, Set 6.3a, to find the shortest routes between node 4 and node 7.

## 6.4 MAXIMAL FLOW MODEL

Consider a network of pipelines that transports crude oil from oil wells to refineries. Intermediate booster and pumping stations are installed at appropriate design distances to move the crude in the network. Each pipe segment has a finite maximum rate of crude flow (or capacity). A pipe segment may be unidirectional or bidirectional, depending on its design. A unidirectional segment has a finite capacity in one direction and a zero capacity in the opposite direction. Figure 6.27 demonstrates a typical pipeline network. How can we determine the maximum capacity of the network between the wells and the refineries?

The solution of the proposed problem requires converting the network into one with a single source and a single sink. This requirement can be accomplished by using unidirectional infinite capacity arcs as shown by dashed arcs in Figure 6.27.

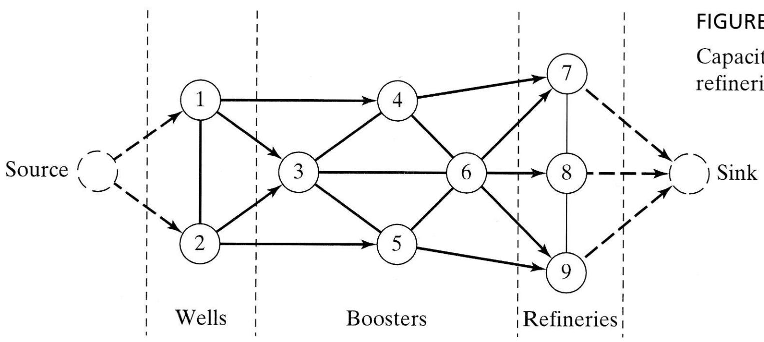
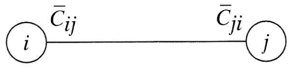



FIGURE 6.27
Capacitated network connecting wells and refineries through booster stations

Given arc $(i, j)$ with $i < j$, we use the notation $(\overline{C}_{ij}, \overline{C}_{ji})$ to represent the flow capacities in the two directions $i \rightarrow j$ and $j \rightarrow i$, respectively. To eliminate ambiguity, we place $\overline{C}_{ij}$ on the arc next to node $i$ with $\overline{C}_{ji}$ placed next to node $j$, as shown in Figure 6.28.

FIGURE 6.28

Arc flows $\overline{C}_{ij}$ from $i \rightarrow j$ and $\overline{C}_{ji}$ from $j \rightarrow i$




## 6.4.1 Enumeration of Cuts

A **cut** defines a set of arcs which when deleted from the network will cause a complete disruption of flow between the source and sink nodes. The **cut capacity** equals the sum of the capacities of the associated arcs. Among *all* possible cuts in the network, the cut with the *smallest capacity* gives the maximum flow in the network.

### Example 6.4-1

Consider the network in Figure 6.29. The bidirectional capacities are shown on the respective arcs using the convention in Figure 6.28. For example, for arc (3, 4), the flow limit is 10 units from 3 to 4 and 5 units from 4 to 3.

FIGURE 6.29

Examples of cuts in flow networks

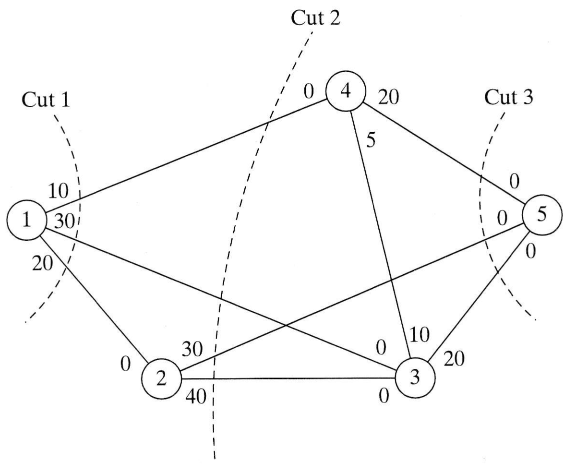



Figure 6.29 illustrates three cuts whose capacities are computed in the following table.

| Cut | Associated arcs | Capacity |
|---|---|---|
| 1 | (1, 2), (1, 3), (1, 4) | 20 + 30 + 10 = 60 |
| 2 | (1, 3), (1, 4), (2, 3), (2, 5) | 30 + 10 + 40 + 30 = 110 |
| 3 | (2, 5), (3, 5), (4, 5) | 30 + 20 + 20 = 70 |

We cannot tell what the maximal flow in the network is unless we exhaustively enumerate all possible cuts. The only piece of information we can get from the partial enumeration of three cuts is that the maximum flow in the network cannot exceed 60 units. Unfortunately, exhaustive enumeration of all cuts is not a simple task, thus making it necessary to develop the efficient algorithm in Section 6.4.2.

## PROBLEM SET 6.4A

1. For the network in Figure 6.29, determine two additional cuts, and find their capacities.

### 6.4.2 Maximal Flow Algorithm

The maximal flow algorithm is based on finding **breakthrough paths** with net *positive* flow between the source and sink nodes. Each path commits part or all the capacities of its arcs to the total flow in the network.

Consider arc $(i, j)$ with (initial) capacities $(\overline{C}_{ij}, \overline{C}_{ji})$. As portions of these capacities are committed to the flow in the arc, the **residuals** (or remaining capacities) of the arc are updated. The network with the updated residuals is referred to as the **residue network**. We use the notation $(c_{ij}, c_{ji})$ to represent these residuals.

For a node $j$ that receives flow from node $i$, we define a label $[a_j, i]$, where $a_j$ is the flow from node $i$ to node $j$. The steps of the algorithm are summarized as follows.

**Step 1.** For all arcs $(i, j)$, set the residual capacity equal to the initial capacity—that is $(c_{ij}, c_{ji}) = (\overline{C}_{ij}, \overline{C}_{ji})$. Let $a_1 = \infty$ and label source node 1 with $[\infty, -]$. Set $i = 1$, and go to step 2.

**Step 2.** Determine $S_i$ as the set of unlabeled nodes $j$ that can be reached directly from node $i$ by arcs with *positive* residuals (that is, $c_{ij} > 0$ for all $j \in S_i$). If $S_i \neq \emptyset$, go to step 3. Otherwise, go to step 4.

**Step 3.** Determine $k \in S_i$ such that

$$c_{ik} = \max_{j \in S_i} \{c_{ij}\}$$

Set $a_k = c_{ik}$ and label node $k$ with $[a_k, i]$. If $k = n$, the sink node has been labeled, and a *breakthrough path* is found, go to step 5. Otherwise, set $i = k$, and go to step 2.

**Step 4.** *(Backtracking).* If $i = 1$, no further breakthroughs are possible; go to step 6. Otherwise, let $r$ be the node that has been labeled *immediately* before the current node $i$ and remove $i$ from the set of nodes that are adjacent to $r$. Set $i = r$, and go to step 2.

**Step 5.** *(Determination of Residue Network).* Let $N_p = (1, k_1, k_2, \ldots, n)$ define the nodes of the $p$th breakthrough path from source node 1 to sink node $n$. Then the maximum flow along the path is computed as

$$f_p = \min \{a_1, a_{k_1}, a_{k_2}, \ldots, a_n\}$$

The residual capacity of each arc along the breakthrough path is *decreased* by $f_p$ in the direction of the flow and *increased* by $f_p$ in the reverse direction—that is, for nodes $i$ and $j$ on the path, the residual flow is changed from the current $(c_{ij}, c_{ji})$ to

**(a)** $(c_{ij} - f_p, c_{ji} + f_p)$ if the flow is from $i$ to $j$

**(b)** $(c_{ij} + f_p, c_{ji} - f_p)$ if the flow is from $j$ to $i$

Reinstate any nodes that were removed in step 4. Set $i = 1$, and return to step 2 to attempt a new breakthrough path.

**Step 6.** *(Solution)*

**(a)** Given that $m$ breakthrough paths have been determined, the maximal flow in the network is

$$F = f_1 + f_2 + \dots + f_m$$

**(b)** Given that the *initial* and *final* residuals of arc $(i, j)$ are given by $(\overline{C}_{ij}, \overline{C}_{ji})$ and $(c_{ij}, c_{ji})$, respectively, the optimal flow in arc $(i, j)$ is computed as follows: Let $(\alpha, \beta) = (\overline{C}_{ij} - c_{ij}, \overline{C}_{ji} - c_{ji})$. If $\alpha > 0$, the optimal flow from $i$ to $j$ is $\alpha$. Otherwise, if $\beta > 0$, the optimal flow from $j$ to $i$ is $\beta$. (It is impossible to have both $\alpha$ and $\beta$ positive.)

The backtracking process of step 4 is invoked when the algorithm becomes inadvertently "dead-ended" at an intermediate node before a breakthrough can be realized. The flow adjustment in step 5 can be explained via the simple flow network in Figure 6.30. Network (a) gives the first breakthrough path $N_1 = \{1, 2, 3, 4\}$ with its maximum flow $f_1 = 5$. Thus, the residuals of each of arcs (1,2), (2,3), and (3,4) are changed from (5,0) to (0,5), per step 5. Network (b) now gives the second breakthrough path $N_2 = \{1, 3, 2, 4\}$ with $f_2 = 5$. After making the necessary flow adjustments, we get network (c), where no further breakthroughs are possible. What happened in the transition from (b) to (c) is nothing but a cancellation of a previously committed flow in the direction $2 \rightarrow 3$. The algorithm is able to "remember" that a flow from 2 to 3 has been committed previously only because we have increased the capacity in the reverse direction from 0 to 5 (per step 5).

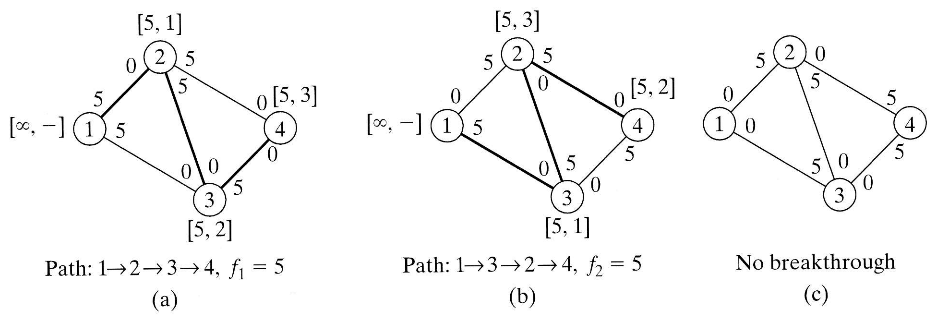



FIGURE 6.30

Use of residual to calculate maximum flow

### Example 6.4-2

Determine the maximal flow in the network of Example 6.4-1 (Figure 6.29). Figure 6.31 provides a graphical summary of the iterations of the algorithm. You will find it helpful to compare the description of the iterations with the graphical summary.

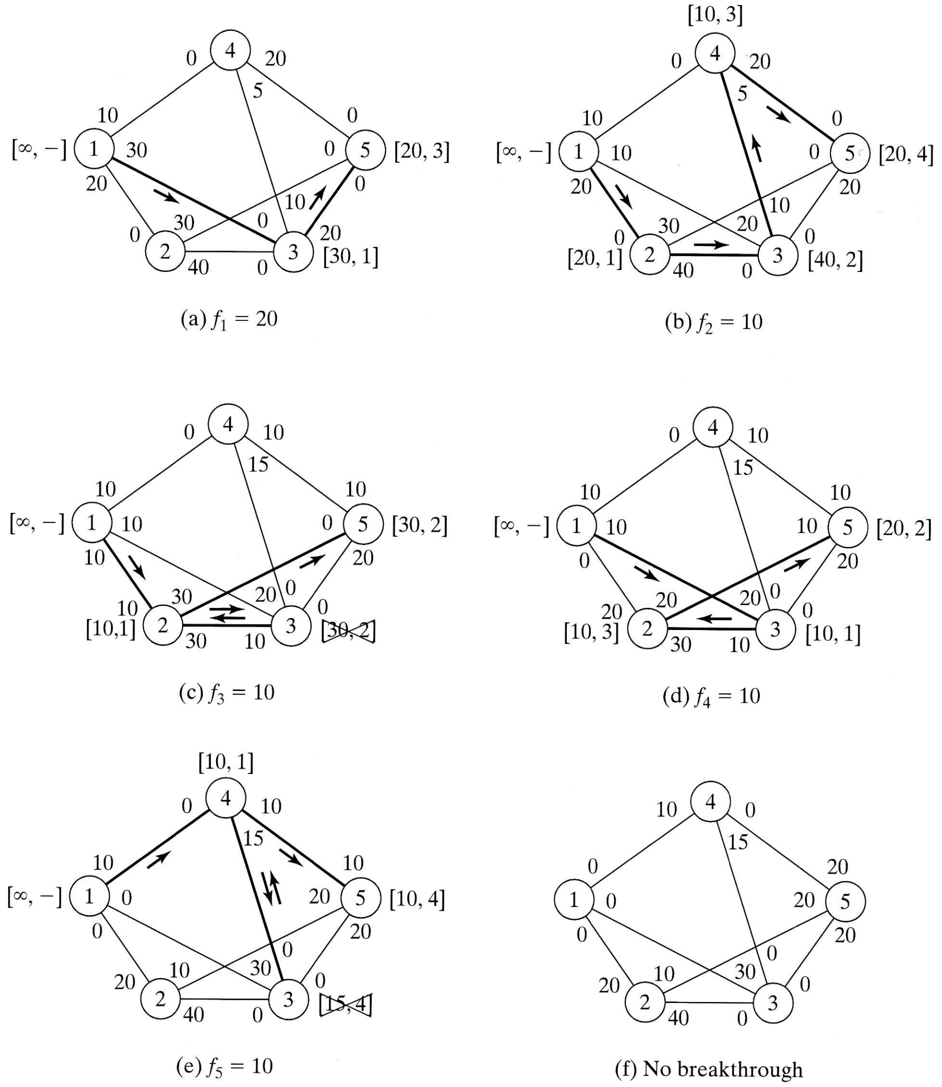



FIGURE 6.31

Iterations of the maximum flow algorithm of Example 6.4-2

**Iteration 1.** Set the initial residuals $(c_{ij}, c_{ji})$ equal to the initial capacities $(\overline{C}_{ij}, \overline{C}_{ji})$.

**Step 1.** Set $a_1 = \infty$ and label node 1 with $[\infty, —]$. Set $i = 1$.
**Step 2.** $S_1 = \{2, 3, 4\} (\neq \emptyset)$.
**Step 3.** $k = 3$ because $c_{13} = \max\{c_{12}, c_{13}, c_{14}\} = \max\{20, 30, 10\} = 30$. Set $a_3 = c_{13} = 30$, and label node 3 with [30, 1]. Set $i = 3$, and repeat step 2.
**Step 2.** $S_3 = (4, 5)$.
**Step 3.** $k = 5$ and $a_5 = c_{35} = \max\{10, 20\} = 20$. Label node 5 with [20, 3]. Breakthrough is achieved. Go to step 5.
**Step 5.** Breakthrough path is determined from the labels starting at node 5 and ending at node 1—that is, **(5)** → [20, **3**] → **(3)** → [30, **1**] → **(1)**. Thus, $N_1 = \{1, 3, 5\}$ and $f_1 = \min\{a_1, a_3, a_5\} = \{\infty, 30, 20\} = 20$. The residual capacities along path $N_1$ are

$$(c_{13}, c_{31}) = (30 - 20, 0 + 20) = (10, 20)$$

$$(c_{35}, c_{53}) = (20 - 20, 0 + 20) = (0, 20)$$

**Iteration 2.**

**Step 1.** Set $a_1 = \infty$, and label node 1 with $[\infty, —]$. Set $i = 1$.
**Step 2.** $S_1 = \{2, 3, 4\}$.
**Step 3.** $k = 2$ and $a_2 = c_{12} = \max\{20, 10, 10\} = 20$. Set $i = 2$, and repeat step 2.
**Step 2.** $S_2 = \{3, 5\}$.
**Step 3.** $k = 3$ and $a_3 = c_{23} = 40$. Label node 3 with [40, 2]. Set $i = 3$, and repeat step 2.
**Step 2.** $S_3 = \{4\}$ (note that $c_{35} = 0$—hence, node 5 cannot be included in $S_3$).
**Step 3.** $k = 4$ and $a_4 = c_{34} = 10$. Label node 4 with [10, 3]. Set $i = 4$, and repeat step 2.
**Step 2.** $S_4 = \{5\}$ (note that nodes 1 and 3 are already labeled—hence, they cannot be included in $S_4$).
**Step 3.** $k = 5$ and $a_5 = c_{45} = 20$. Label node 5 with [20,4]. Breakthrough has been achieved. Go to step 5.
**Step 5.** $N_2 = \{1, 2, 3, 4, 5\}$ and $f_2 = \min\{\infty, 20, 40, 10, 20\} = 10$. The residuals along the path of $N_2$ are

$$(c_{12}, c_{21}) = (20 - 10, 0 + 10) = (10, 10)$$

$$(c_{23}, c_{32}) = (40 - 10, 0 + 10) = (30, 10)$$

$$(c_{34}, c_{43}) = (10 - 10, 5 + 10) = (0, 15)$$

$$(c_{45}, c_{54}) = (20 - 10, 0 + 10) = (10, 10)$$

**Iteration 3.**

**Step 1.** Set $a_1 = \infty$ and label node 1 with $[\infty, —]$. Set $i = 1$.
**Step 2.** $S_1 = \{2, 3, 4\}$.
**Step 3.** $k = 2$ and $a_2 = c_{12} = \max\{10, 10, 10\} = 10$ (though ties are broken arbitrarily, TORA always selects the tied node with the smallest index; we will

use this convention throughout the example). Label node 2 with [10,1]. Set $i = 2$, and repeat step 2.

**Step 2.** $S_2 = \{3, 5\}$.

**Step 3.** $k = 3$ and $a_3 = c_{23} = 30$. Label node 3 with [30, 2]. Set $i = 3$, and repeat step 2.

**Step 2.** $S_3 = \emptyset$ (because $c_{34} = c_{35} = 0$). Go to step 4 to backtrack.

**Step 4.** The label [30, 2] at node 3 gives the immediately preceding node $r = 2$. Remove node 3 from further consideration *in this iteration* by crossing it out. Set $i = r = 2$, and repeat step 2.

**Step 2.** $S_2 = \{5\}$ (note that node 3 has been removed in the backtracking step).

**Step 3.** $k = 5$ and $a_5 = c_{25} = 30$. Label node 5 with [30, 2]. Breakthrough has been achieved; go to step 5.

**Step 5.** $N_3 = \{1, 2, 5\}$ and $c_5 = \min\{\infty, 10, 30\} = 10$. The residuals along the path of $N_3$ are

$$(c_{12}, c_{21}) = (10 - 10, 10 + 10) = (0, 20)$$

$$(c_{25}, c_{52}) = (30 - 10, 0 + 10) = (20, 10)$$

**Iteration 4.** This iteration yields $N_4 = \{1, 3, 2, 5\}$ with $f_4 = 10$ (verify!).

**Iteration 5.** This iteration yields $N_5 = \{1, 4, 5\}$ with $f_5 = 10$ (verify!).

**Iteration 6.** All the arcs out of node 1 have zero residuals. Hence, no further breakthroughs are possible. We turn to step 6 to determine the solution.

**Step 6.** Maximal flow in the network is $F = f_1 + f_2 + \ldots + f_5 = 20 + 10 + 10 + 10 + 10 = 60$ units. The flow in the different arcs is computed by subtracting the last residuals $(c_{ij}, c_{ji})$ in iterations 6 from the initial capacities $(\overline{C}_{ij}, \overline{C}_{ji})$, as the following table shows.

| Arc | $(\overline{C}_{ij}, \overline{C}_{ji}) - (c_{ij}, c_{ji})_6$ | Flow amount | Direction |
|---|---|---|---|
| (1, 2) | (20, 0) − (0, 20) = (20, −20) | 20 | $1 \to 2$ |
| (1, 3) | (30, 0) − (0, 30) = (30, −30) | 30 | $1 \to 3$ |
| (1, 4) | (10, 0) − (0, 10) = (10, −10) | 10 | $1 \to 4$ |
| (2, 3) | (40, 0) − (40, 0) = (0, 0) | 0 | — |
| (2, 5) | (30, 0) − (10, 20) = (20, −20) | 20 | $2 \to 5$ |
| (3, 4) | (10, 5) − (0, 15) = (10, −10) | 10 | $3 \to 4$ |
| (3, 5) | (20, 0) − (0, 20) = (20, −20) | 20 | $3 \to 5$ |
| (4, 5) | (20, 0) − (0, 20) = (20, −20) | 20 | $4 \to 5$ |

You can use TORA to solve the maximum flow model in an automated mode or to produce the iterations outlined above. From the `SOLVE/MODIFY` menu select `Solve Problem`. After specifying the output format, go to output screen and select either `Maximum Flows` or `Iterations`. Figure 6.32 illustrates the first two iterations of Example 6.4-2 (file ch6ToraMaxFlowEx6-4-2.txt).

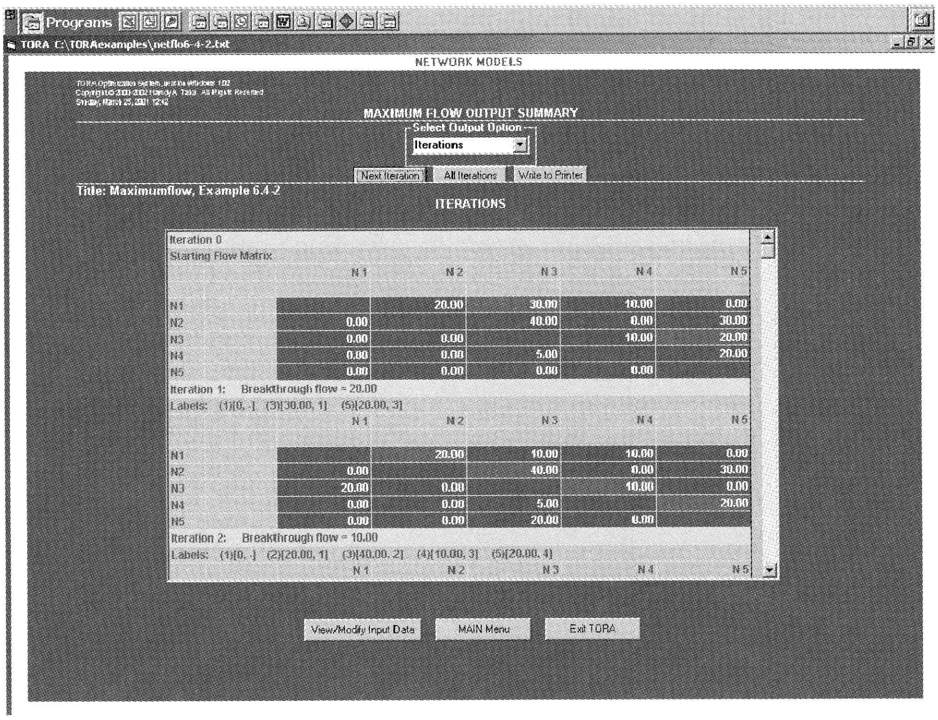



FIGURE 6.32

TORA's maximum flow iterations for Example 6.4-2

## PROBLEM SET 6.4B

1. In Example 6.4-2,
   (a) Determine the surplus capacities for all the arcs.
   (b) Determine the amount of flow through nodes 2, 3, and 4.
   (c) Can the network flow be increased by increasing the capacities in the directions $3 \rightarrow 5$ and $4 \rightarrow 5$?
2. Determine the maximal flow and the optimum flow in each arc for the network in Figure 6.33.
3. Three refineries send a gasoline product to two distribution terminals through a pipeline network. Any demand that cannot be satisfied through the network is acquired from other sources. The pipeline network is served by three pumping stations as shown in Figure 6.34. The product flows in the network in the direction shown by the arrows. The capacity of each pipe segment (shown directly on the arcs) is in million bbl per day. Determine the following:
   (a) The daily production at each refinery that matches the maximum capacity of the network.
   (b) The daily demand at each terminal that matches the maximum capacity of the network.
   (c) The daily capacity of each pump that matches the maximum capacity of the network.

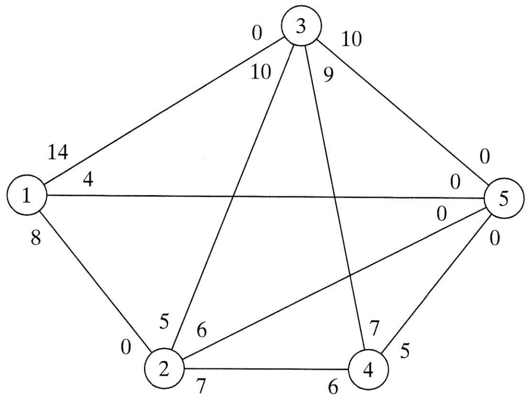


FIGURE 6.33
Network for Problem 2, Set 6.4b

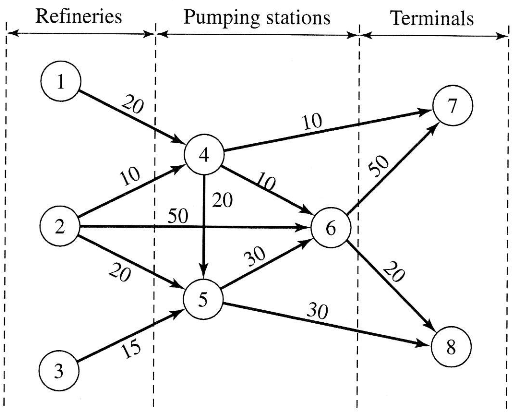



FIGURE 6.34
Network for Problem 3, Set 6.4b

4. Suppose that the maximum daily capacity of pump 6 in the network of Figure 6.34 is limited to 60 million bbl per day. Remodel the network to include this restriction. Then determine the maximum capacity of the network.
5. Chicken feed is transported by trucks from three silos to four chicken farms. Some of the silos cannot ship directly to some of the farms. The capacities of the other routes are limited by the number of trucks available and the number of trips made daily. The following table shows the daily amounts of supply at the silos and demand at the farms (in thousands of pounds). The cell entries of the table specify the daily capacities of the associated routes.

| | Farm 1 | Farm 2 | Farm 3 | Farm 4 | |
|---|---|---|---|---|---|
| Silo 1 | 30 | 5 | 0 | 40 | **20** |
| Silo 2 | 0 | 0 | 5 | 90 | **20** |
| Silo 3 | 100 | 40 | 30 | 40 | **200** |
| | **200** | **10** | **60** | **20** | |

**(a)** Determine the schedule that satisfies the most demand.

**(b)** Will the proposed schedule satisfy all the demand at the farms?

**6.** In Problem 5, suppose that transshipping is allowed between silos 1 and 2 and silos 2 and 3. Suppose also that transshipping is allowed between farms 1 and 2, 2 and 3, and 3 and 4. The maximum two-way daily capacity on the proposed transshipping routes is 50 (thousand) lb. What is the effect of transshipping on the unsatisfied demands at the farms?

**7.** A parent has five (teenage) children and five household chores to assign to them. Past experience has shown that forcing chores on a child is counterproductive. With this in mind, the children are asked to list their preferences among the five chores, as the following table shows:

| Child | Preferred chore |
|---|---|
| Rif | 3, 4, or 5 |
| Mai | 1 |
| Ben | 1 or 2 |
| Kim | 1, 2, or 5 |
| Ken | 2 |

The parent's modest goal now is to finish as many chores as possible while abiding by the children's preferences. Determine the maximum number of chores that can be completed and the assignment of chores to children.

**8.** Four factories are engaged in the production of four types of toys. The following table lists the toys that can be produced by each factory.

| Factory | Toy productions mix |
|---|---|
| 1 | 1, 2, 3 |
| 2 | 2, 3 |
| 3 | 1, 4 |
| 4 | 3, 4 |

All toys require the same per unit labor and material. The daily capacities of the four factories are 250, 180, 300, and 100 toys, respectively. The daily demands for the four toys are 200, 150, 350, and 100 units, respectively. Determine the production schedules that will most satisfy the demands for the four toys.

**9.** The academic council at the U of A is seeking representation from among six students who are affiliated with four honor societies. The academic council representation includes three areas: mathematics, art, and engineering. At most two students in each area can be on the council. The following table shows the membership of the six students in the four honor societies:

| Society | Affiliated students |
|---|---|
| 1 | 1, 2, 3 |
| 2 | 1, 3, 5 |
| 3 | 3, 4, 5 |
| 4 | 1, 2, 4, 6 |

The students who are skilled in the areas of mathematics, art, and engineering are shown in the following table:

| Area | Skilled students |
|---|---|
| Mathematics | 1, 2, 4 |
| Art | 3, 4 |
| Engineering | 4, 5, 6 |

A student who is skilled in more than one area must be assigned exclusively to one area only. Can all four honor societies be represented on the council?

**10.** *Maximal/Minimal Flow in Networks with Lower Bounds.* The maximal flow algorithm given in this section assumes that all the arcs have zero lower bounds. In some models, the lower bounds may be strictly positive, and we may be interested in finding the maximal or minimal flow in the network (see Comprehensive Problem 6-3). The presence of the lower bound poses difficulty because the network may not have a feasible flow at all. The objective of this exercise is to show that any maximal and minimal flow model with positive lower bounds can be solved using two steps.

**Step 1.** Find an initial feasible solution for the network with positive lower bounds.

**Step 2.** Using the feasible solution in step 1, find the maximal or minimal flow in the original network.

**(a)** Show that an arc $(i, j)$ with flow limited by $l_{ij} \leq x_{ij} \leq u_{ij}$ can be represented equivalently by a *sink* with demand $l_{ij}$ at node $i$ and a *source* with supply $l_{ij}$ at node $j$ with flow limited by $0 \leq x'_{ij} \leq u_{ij} - l_{ij}$.

**(b)** Show that finding a feasible solution for the original network is equivalent to finding the maximal flow $x'_{ij}$ in the network after (1) modifying the bounds on $x_{ij}$ to $0 \leq x'_{ij} \leq u_{ij} - l_{ij}$, (2) "lumping" all the resulting sources into one supersource with outgoing arc capacities $l_{ij}$, (3) "lumping" all the resulting sinks into one supersink with incoming arc capacities $l_{ij}$, and (4) connecting the terminal node $t$ to the source node $s$ in the original network by a return infinite capacity arc. A feasible solution exists if the maximal flow in the new network equals the sum of the lower bounds in the original network. Apply the procedure to the following network and find a feasible flow solution:

| Arc $(i, j)$ | $(l_{ij}, u_{ij})$ |
|---|---|
| (1, 2) | (5, 20) |
| (1, 3) | (0, 15) |
| (2, 3) | (4, 10) |
| (2, 4) | (3, 15) |
| (3, 4) | (0, 20) |

**(c)** Use the feasible solution for the network in (b) together with the maximal flow algorithm to determine the *minimal* flow in the original network. (*Hint:* First compute the residue network given the initial feasible solution. Next, determine the maximum flow *from the end node to the start node.* This is equivalent to finding the maximum flow that should be canceled from the start node to the end node. Now, combining the feasible and maximal flow solutions yields the minimal flow in the original network.)

**(d)** Use the feasible solution for the network in (b) together with the *maximal* flow model to determine the maximal flow in the original network. (*Hint:* As in part [c], start with the residue network. Next apply the breakthrough algorithm to the resulting residue network exactly as in the regular maximal flow model.)

### 6.4.3 Linear Programming Formulation of the Maximum Flow Model

Define $x_{ij}$ as the amount of flow in arc $(i, j)$ and let $c_{ij}$ be the capacity of the same arc. Assume that $s$ and $t$ are the start and terminal nodes between which we need to determine the maximum flow in the capacitated network.

The constraints of the problem preserve the in-out flow at each node, with the exception of start and terminal nodes. The objective function maximizes either the total "out" flow from start node $s$ or the total "in" flow to terminal node $t$.

#### Example 6.4-3

In the maximum flow model of Figure 6.29 (Example 6.4-2), $s = 1$ and $t = 5$. The following table summarizes the associated LP with two different objective functions depending on whether we are maximizing the output from node 1 ($=z_1$) or the input to node 5 ($=z_2$).

| | $x_{12}$ | $x_{13}$ | $x_{14}$ | $x_{23}$ | $x_{25}$ | $x_{34}$ | $x_{35}$ | $x_{43}$ | $x_{45}$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| Maximize $z_1$ = | 1 | 1 | 1 | | | | | | | |
| Maximize $z_2$ = | | | | | 1 | | 1 | | 1 | |
| Node 2 | 1 | | | −1 | −1 | | | | | = 0 |
| Node 3 | | 1 | | 1 | | −1 | −1 | 1 | | = 0 |
| Node 4 | | | 1 | | | 1 | | −1 | −1 | = 0 |
| Capacity | 20 | 30 | 10 | 40 | 30 | 10 | 20 | 5 | 20 | |

The optimal solution using either objective function is

$$x_{12} = 20,\ x_{13} = 30,\ x_{14} = 10,\ x_{25} = 20,\ x_{34} = 10,\ x_{35} = 20,\ x_{45} = 20$$

The associated maximum flow is $z_1 = z_2 = 60$.

#### PROBLEM SET 6.4C

1. Rework Problem 2, Set 6.4b using linear programming.
2. Rework Problem 5, Set 6.4b using linear programming.

### 6.4.4 Excel Spreadsheet Solution of the Maximum Flow Model

The network-based Excel spreadsheet developed for the transportation model (Section 5.3.3) is modified to determine the maximum flow in a capacitated network. The spreadsheet is designed for problems with a maximum of 10 nodes. Figure 6.35 shows the application of the spreadsheet to Example 6.4-2 (file ch6SolverMax Flow.xls). The capacity flow matrix resides in cells B6:K15.[4] A blank cell in the capacity matrix indicates that the associated arc has infinite capacity. A zero entry corresponds to a nonexisting flow arc. Otherwise, all the remaining arcs must have finite capacities.

Once the flow capacity data have been entered, the remainder of the spreadsheet (*intermediate calculations* and *optimum solution* sections) is created automatically. All

[4]In Figure 6.35, rows 11 through 16 and column K are hidden to conserve space.

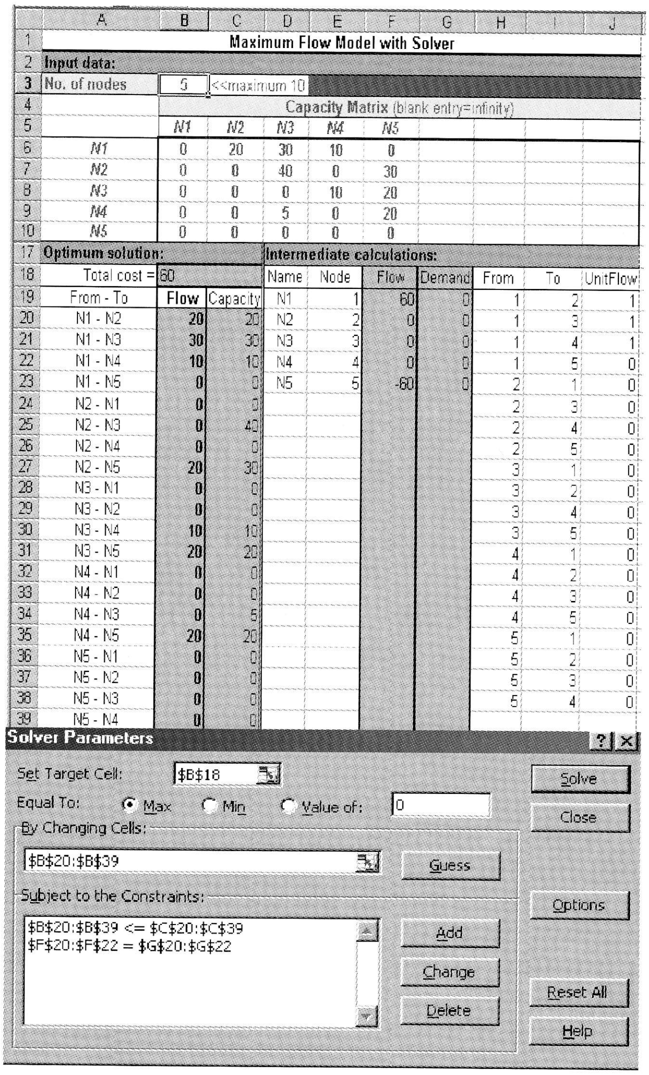



FIGURE 6.35

Excel Solver solution of the maximum flow model of Example 6.4-2

that is needed now is to update Solver parameters to match the input data. Column B specifies the changing cells (arcs flow) of the problem. The range for Changing Cells must encompass all the arcs specified in column A (make sure that you give each node a name in the input data matrix, else column A will only show a hyphen in the associated cells). In the present example, cells B20:B39 provide Changing Cells range. Column C specifies the capacities of the arcs of the network (cells C20:C39).

The constraints of the model represent the flow balance equation for each node. The LP formulation in Section 6.4.3 shows that it is not necessary to construct flow equations for the first and last nodes of the network (nodes 1 and 5 in Figure 6.35). Thus, cells F20:F22 define the left-hand side and cells G20:G22 represent the right-hand side of the flow equations.

Based on given information, Solver parameters for the example in Figure 6.26 are entered as

```
Changing Cells: B20:B39
Constraints: B20:B39<=C20:C39      (Arc capacity)
             F20:F22=G20:G22       (Flow equations for nodes 2, 3, and 4)
```

Note that Target Cell is automated and need not be changed. The Equal to parameter is Max because this is a maximization problem.

The output in Figure 6.35 yields the solution (N1-N2 = 20, N1-N3 = 30, N1-N4 = 10, N2-N5 = 20, N3-N4 = 10, N3-N5 = 20, N4-N5 = 20) with a maximum flow of 60 units.

### PROBLEM SET 6.4D

1. Solve Problem 2, Set 6.4b using Excel Solver.
2. Solve Problem 5, Set 6.4b using Excel Solver.

## 6.5 MINIMUM-COST CAPACITATED FLOW PROBLEM

The minimum-cost capacitated flow problem is based on the following assumptions:

1. A (nonnegative) unit flow cost is associated with each arc.
2. Arcs may have positive lower capacity limits.
3. Any node in the network may act as a source or as a sink.

The new model determines the flows in the different arcs that minimize the total cost while satisfying the flow restrictions on the arcs and the supply and demand amounts at the nodes. We first present the capacitated network flow model and its equivalent linear programming formulation. The linear programming formulation is the basis for the development of a special capacitated simplex algorithm for solving the network flow model. The section ends with a presentation of a spreadsheet template of the minimum-cost capacitated network.

### 6.5.1 Network Representation

Consider a capacitated network $G = (N, A)$, where $N$ is the set of nodes, and $A$ is the set of arcs and define

$$x_{ij} = \text{amount of flow from node } i \text{ to node } j$$

$$u_{ij}\,(l_{ij}) = \text{upper (lower) capacity of arc } (i, j)$$

$$c_{ij} = \text{unit flow cost from node } i \text{ to node } j$$

$$f_i = \text{net flow at node } i$$

Figure 6.36 depicts these definitions on arc $(i, j)$. The label $[f_i]$ assumes a positive (negative) value when a net supply (demand) is associated with node $i$.

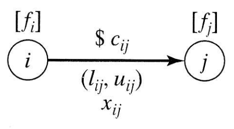



FIGURE 6.36

Capacitated arc with external flow

### Example 6.5-1

GrainCo supplies corn from three silos to three poultry farms. The supply amounts at the three silos are 100, 200, and 50 thousand bushels; and the demand at the three farms is 150, 80, and 120 thousand bushels. GrainCo mostly uses railroads to transport the corn to the farms, with the exception of three routes where trucks are used.

Figure 6.37 shows the available route between the silos and the farms. The silos are represented by nodes 1, 2, and 3 whose supply amounts are [100], [200], and [50], respectively. The farms are represented by nodes 4, 5, and 6 whose demand amounts are [−150], [−80], and [−120], respectively. The routes allow transshipping between the silos. Arcs (1, 4), (3, 4), and (4, 6) are truck routes with minimum and maximum capacities. For example, the capacity of route (1, 4) is between 50 and 80 thousand bushels. All other routes use trainloads, whose maximum capacity is practically unlimited. The transportation costs per bushel are indicated on the respective arcs.

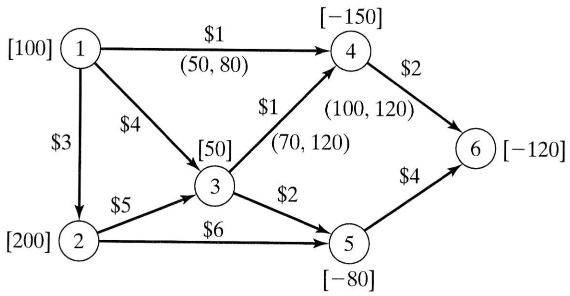



FIGURE 6.37

Capacitated network for Example 6.5-1

## PROBLEM SET 6.5A

**1.** A product is manufactured to satisfy demand over a 4-period planning horizon according to the following data:

| Period | Units of demand | Unit production cost ($) | Unit holding cost ($) |
|---|---|---|---|
| 1 | 100 | 24 | 1 |
| 2 | 110 | 26 | 2 |
| 3 | 95 | 21 | 1 |
| 4 | 125 | 24 | 2 |

Given that no back-ordering is allowed, represent the problem as a network model.

**2.** In Problem 1, suppose that back-ordering is allowed at a penalty of $1.50 per unit per period. Formulate the problem as a network model.

3. In Problem 1, suppose that the production capacities of periods 1 to 4 are 110, 95, 125, and 100 units, respectively, in which case the given demand cannot be satisfied without back-ordering. Assuming that the penalty cost for back-ordering is \$1.50 per unit per period, formulate the problem as a network model.
4. Daw Chemical owns two plants that manufacture a basic chemical compound for two customers at the rate of 660 and 800 tons per month. The monthly production capacity of plant 1 is between 400 and 800 tons and that of plant 2 is between 450 and 900 tons. The production costs per ton in plants 1 and 2 are \$25 and \$28, respectively. Raw material for the plants is provided by two suppliers, who are contracted to ship at least 500 and 700 tons per month for plants 1 and 2 at the costs of \$200 and \$210 per ton, respectively. Daw Chemical also assumes the transportation cost of both the raw material and the final compound. The costs per ton of transporting the raw material from supplier 1 to plants 1 and 2 are \$10 and \$12. Similar costs from supplier 2 are \$9 and \$13, respectively. The transportation costs per ton from plant 1 to clients 1 and 2 are \$3 and \$4, and from plant 2 costs are \$5 and \$2, respectively. Assuming that 1 ton of raw material produces 1 ton of the final compound, formulate the problem as a network model.
5. Two nonintegrated public schools are required to change the racial balance of their enrollments by accepting minority students. Minority enrollment must be between 30% and 40% in both schools. Nonminority students live in two communities, and minority students live in three other communities. Traveled distances, in miles, from the five communities to the two schools are summarized in the following table:

| | | Round-trip miles from school to | | | | |
|---|---|---|---|---|---|---|
| | | *Minority areas* | | | *Nonminority areas* | |
| *School* | *Maximum enrollment* | *1* | *2* | *3* | *1* | *2* |
| 1 | 1500 | 20 | 12 | 10 | 4 | 5 |
| 2 | 2000 | 15 | 18 | 8 | 6 | 5 |
| Student population | | 500 | 450 | 300 | 1000 | 1000 |

Formulate the problem as a network model to determine the number of minority and nonminority students enrolled in each school.

### 6.5.2 Linear Programming Formulation

The formulation of the capacitated network model as a linear program provides the foundation for the development of the capacitated simplex algorithm, which we will present in the next section. Using the notation introduced in Section 6.5.1, the linear program for the capacitated network is given as

$$\text{Minimize } z = \sum_{(i,j)\in A}\sum c_{ij}x_{ij}$$

subject to

$$\sum_{\substack{k \\ (j,k)\in A}} x_{jk} - \sum_{\substack{i \\ (i,j)\in A}} x_{ij} = f_j,\ j \in N$$

$$l_{ij} \leq x_{ij} \leq u_{ij}$$

The equation for node $j$ measures the net flow $f_j$ in node $j$ as

$$(\text{Outgoing flow from node } j) - (\text{Incoming flow into node } j) = f_j$$

Node $j$ acts as a source if $f_j > 0$ and as a sink if $f_j < 0$.

We can always remove the lower bound $l_{ij}$ from the constraints by using the substitution

$$x_{ij} = x'_{ij} + l_{ij}$$

The new flow variable, $x'_{ij}$, has an upper limit of $u_{ij} - l_{ij}$. Additionally, the net flow at node $i$ becomes $f_i - l_{ij}$, and that at node $j$ is $f_j + l_{ij}$. Figure 6.38 shows the transformation of activity $(i, j)$ after the lower bound is substituted out.

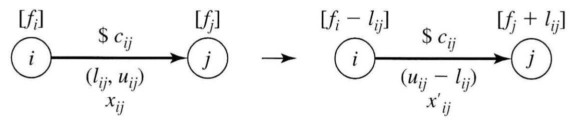



**FIGURE 6.38**
Removal of the lower bound in arcs

## Example 6.5-2

Write the linear program for the network in Figure 6.37, before and after the lower bounds are substituted out.

The main constraints of the linear program relate the input-output flow at each node, which yields the following LP:

| | $x_{12}$ | $x_{13}$ | $x_{14}$ | $x_{23}$ | $x_{25}$ | $x_{34}$ | $x_{35}$ | $x_{46}$ | $x_{56}$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| Minimize | 3 | 4 | 1 | 5 | 6 | 1 | 2 | 2 | 4 | |
| Node 1 | 1 | 1 | 1 | | | | | | | = 100 |
| Node 2 | −1 | | | 1 | 1 | | | | | = 200 |
| Node 3 | | −1 | | −1 | | 1 | 1 | | | = 50 |
| Node 4 | | | −1 | | | −1 | | 1 | | = −150 |
| Node 5 | | | | | −1 | | −1 | | 1 | = −80 |
| Node 6 | | | | | | | | −1 | −1 | = −120 |
| Lower bounds | 0 | 0 | 50 | 0 | 0 | 70 | 0 | 100 | 0 | |
| Upper bounds | ∞ | ∞ | 80 | ∞ | ∞ | 120 | ∞ | 120 | ∞ | |

Note the arrangement of the constraints coefficients. The column associated with variable $x_{ij}$ has exactly one +1 in row $i$ and one −1 in row $j$. The rest of the coefficients are 0. This structure is typical of network flow models.

The variables with lower bounds are substituted as

$$x_{14} = x'_{14} + 50$$

$$x_{34} = x'_{34} + 70$$

$$x_{46} = x'_{46} + 100$$

The resulting linear program is

| | $x_{12}$ | $x_{13}$ | $x'_{14}$ | $x_{23}$ | $x_{25}$ | $x'_{34}$ | $x_{35}$ | $x'_{46}$ | $x_{56}$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| Minimize | 3 | 4 | 1 | 5 | 6 | 1 | 2 | 2 | 4 | |
| Node 1 | 1 | 1 | 1 | | | | | | | = 50 |
| Node 2 | −1 | | | 1 | 1 | | | | | = 200 |
| Node 3 | | −1 | | −1 | | 1 | 1 | | | = −20 |
| Node 4 | | | −1 | | | −1 | | 1 | | = −130 |
| Node 5 | | | | | −1 | | −1 | | 1 | = −80 |
| Node 6 | | | | | | | | −1 | −1 | = −20 |
| Upper bounds | ∞ | ∞ | 30 | ∞ | ∞ | 50 | ∞ | 20 | ∞ | |

The corresponding network after substituting out the lower bounds is shown in Figure 6.39. Note that the lower-bound substitution can be effected directly from Figure 6.37 using the substitution in Figure 6.38, and without the need to express the problem as a linear program first.

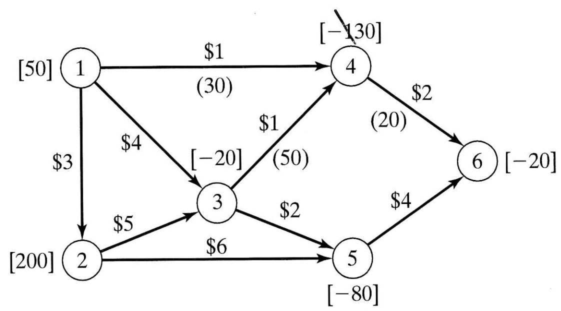



FIGURE 6.39
Network of Example 6.5-2 after substituting out lower bounds

## Example 6.5-3 (Employment Scheduling)

This example illustrates a network model that initially does not satisfy the "node flow" requirement (i.e., node output flow less node input flow equals node net flow), but that can be converted to this form readily through special manipulation of the constraints of the linear program.

Tempo Employment Agency has a contract to provide workers over the next 4 months (January to April) according to the following schedule:

| Month | Jan. | Feb. | Mar. | Apr. |
|---|---|---|---|---|
| No. of workers | 100 | 120 | 80 | 170 |

Because of change in demand, it may be economical to retain more workers than needed in a given month. The cost of recruiting and maintaining a worker is a function of their employment period as the following table shows:

| Employment period (months) | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| Cost per worker ($) | 100 | 130 | 180 | 220 |

Let

$x_{ij}$ = number of workers hired at the *start* of month $i$ and terminated at the *start* of month $j$

For example, $x_{12}$ gives the number of workers hired in January for 1 month only.

To formulate the problem as a linear program for the 4-month period, we add May as a dummy month (month 5), so that $x_{45}$ defines hiring in April for April. The constraints recognize that the demand for period $k$ can be satisfied by all $x_{ij}$ such that $i \le k < j$. Letting $s_i \ge 0$ be the surplus number of workers in month $i$, the linear program is given as

| | $x_{12}$ | $x_{13}$ | $x_{14}$ | $x_{15}$ | $x_{23}$ | $x_{24}$ | $x_{25}$ | $x_{34}$ | $x_{35}$ | $x_{45}$ | $s_1$ | $s_2$ | $s_3$ | $s_4$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Minimize | 100 | 130 | 180 | 220 | 100 | 130 | 180 | 100 | 130 | 100 | | | | | |
| Jan. | 1 | 1 | 1 | 1 | | | | | | | −1 | | | | =100 |
| Feb. | | 1 | 1 | 1 | 1 | 1 | 1 | | | | | −1 | | | =120 |
| Mar. | | | 1 | 1 | | 1 | 1 | 1 | 1 | | | | −1 | | = 80 |
| Apr. | | | | 1 | | | 1 | | 1 | 1 | | | | −1 | =170 |

The preceding LP does not have the (−1, +1) special structure of the network flow model (see Example 6.5-2). Nevertheless, the given linear program can be converted into an equivalent network flow model by using the following arithmetic manipulations:

1. In an $n$-equation linear program, create a new equation, $n + 1$, by multiplying equation $n$ by $-1$.
2. Leave equation 1 unchanged.
3. For $i = 2, 3, \ldots, n$, replace each equation $i$ with (equation $i$) − (equation $i - 1$).

The application of these manipulations to the employment scheduling example yields the following linear program whose structure fits the network flow model:

| | $x_{12}$ | $x_{13}$ | $x_{14}$ | $x_{15}$ | $x_{23}$ | $x_{24}$ | $x_{25}$ | $x_{34}$ | $x_{35}$ | $x_{45}$ | $s_1$ | $s_2$ | $s_3$ | $s_4$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Minimize | 100 | 130 | 180 | 220 | 100 | 130 | 180 | 100 | 130 | 100 | | | | | |
| Jan. | 1 | 1 | 1 | 1 | | | | | | | −1 | | | | = 100 |
| Feb. | −1 | | | | 1 | 1 | 1 | | | | 1 | −1 | | | = 20 |
| Mar. | | −1 | | | −1 | | | 1 | 1 | | | 1 | −1 | | = −40 |
| Apr. | | | −1 | | | −1 | | −1 | | 1 | | | 1 | −1 | = 90 |
| May | | | | −1 | | | −1 | | −1 | −1 | | | | 1 | =−170 |

Using the preceding formulation, the employment scheduling model can be represented equivalently by the minimum-cost flow network shown in Figure 6.40. Actually, because the arcs have no upper bounds, the problem can be solved also as a transshipment model (see Section 5.5).

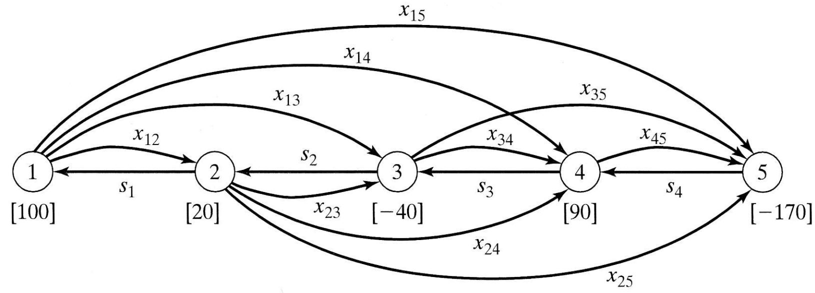



FIGURE 6.40
Network representation of employment scheduling problem

## PROBLEM SET 6.5B

1. Write the linear program associated with the minimum-cost flow network in Figure 6.41, before and after the lower bounds are substituted out.

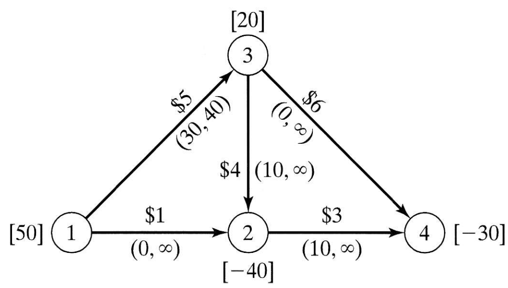



FIGURE 6.41
Network for Problem 1, Set 6.5b

2. Use inspection to find a feasible solution to the minimum-cost network model of the employment scheduling problem in Example 6.5-3 (Figure 6.40). Interpret the solution by showing the pattern of hiring and firing that satisfies the demand for each month, and compute the associated total cost.
3. Reformulate the employment scheduling model of Example 6.5-3, assuming that a worker must be hired for at least 2 months. Write the linear program, and convert it to a minimum-cost flow network.
4. Develop the linear program and the associated minimum-cost flow network for the employment scheduling model of Example 6.5-3 using the following 5-month demand data. The per worker costs of recruiting and maintaining a worker for periods of 1 to 5 months are \$50, \$70, \$85, \$100, and \$130, respectively.

**(a)**

| Month | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| No. of workers | 300 | 180 | 90 | 170 | 200 |

**(b)**

| Month | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| No. of workers | 200 | 220 | 300 | 50 | 240 |

**5.** *Conversion of a Capacitated Network into an Uncapacitated Network.* Show that an arc $(i \rightarrow j)$ with capacitated flow $x_{ij} \le u_{ij}$ can be replaced with two *uncapacitated* arcs $(i \rightarrow k)$ and $(j \rightarrow k)$ with a net (output) flow of $[-u_{ij}]$ at node $k$ and an additional (input) flow of $[+u_{ij}]$ at node $j$. The result is that the *capacitated* network can be converted to an *uncapacitated* transportation cost model (Section 5.1). Apply the resulting transformation to the network in Figure 6.42 and find the optimum solution to the original network by applying TORA to the uncapacitated transportation model.

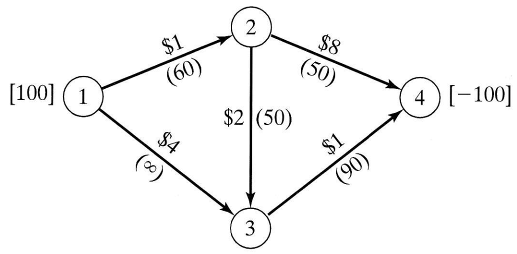



FIGURE 6.42

Network for Problem 5, Set 6.5b

### 6.5.3 Capacitated Network Simplex Algorithm

The algorithm is based on the exact steps of the regular simplex method, but designed to exploit the special network structure of the minimum-cost flow model.

Given $f_i$ is the net flow at node $i$ as defined in the linear program of Section 6.5.2, the capacitated simplex algorithm stipulates that the network must satisfy

$$\sum_{i=1}^{n} f_i = 0$$

The condition says that the total supply in the network equals the total demand. We can always satisfy this requirement by adding a balancing dummy source or destination, which we connect to all other nodes in the network by zero unit cost and infinite capacity arcs. However, the balancing of the network does not guarantee a feasible solution as this depends on the restricting capacities of the arcs.

We will now present the steps of the capacitated algorithm. Familiarity with the simplex method and duality theory (Chapters 3 and 4) is essential. Also, knowledge of the upper-bounded simplex method (Section 7.3) is helpful.

**Step 0.** Determine a starting basic feasible solution (set of arcs) for the network. Go to step 1.

**Step 1.** Determine an entering arc (variable) using the simplex method optimality condition. If the solution is optimal, stop; otherwise, go to step 2.

**Step 2.** Determine the leaving arc (variable) using the simplex method feasibility condition. Determine the new solution, and then go to step 1.

An $n$-node network with zero net flow (i.e., $f_1 + f_2 + \dots + f_n = 0$) consists of $n - 1$ *independent* constraint equations. Thus, an associated basic solution must include $n - 1$ arcs. It can be proved that a basic solution always corresponds to a *spanning tree* of the network (see Section 6.2).

The entering arc (step 1) is determined by computing $z_{ij} - c_{ij}$, the objective coefficients, for all the current nonbasic arcs $(i, j)$. If $z_{ij} - c_{ij} \leq 0$ for all $i$ and $j$, the current basis is optimum. Otherwise, we select the nonbasic arc with the most positive $z_{ij} - c_{ij}$ to enter the basis.

The computation of objective coefficients is based on duality, exactly as we did with the transportation model (see Section 5.3.4). Using the linear program defined in Section 6.5.2, let $w_i$ be the dual variable associated with the constraint of node $i$; then the dual problem (excluding the upper bounds) is given as

$$\text{Maximize } z = \sum_{i=1}^{n} f_i w_i$$

subject to

$$w_i - w_j \leq c_{ij}, \ (i, j) \in A$$

$$w_i \text{ unrestricted in sign, } i = 1, 2, \ldots n$$

From the theory of linear programming, we have

$$w_i - w_j = c_{ij}, \quad \text{for basic arc } (i, j)$$

Because the original linear program (Section 6.5.2) has one redundant constraint by definition, we can assign an arbitrary value to one of the dual variables (compare with the transportation algorithm, Section 5.3). For convenience, we will set $w_1 = 0$. We then solve the (basic) equations $w_i - w_j = c_{ij}$ to determine the remaining dual values. From Section 4.2.3, Method 2, we know that the objective coefficient of nonbasic $x_{ij}$ is the difference between the left-hand side and the right-hand side of the dual associated dual constraint—that is

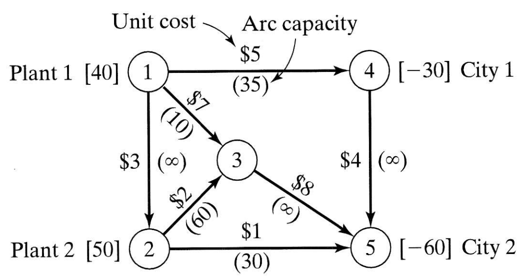
$$z_{ij} - c_{ij} = w_i - w_j - c_{ij}$$

The only remaining detail is to show how the leaving variable is determined. We do so by using a numeric example.

### Example 6.5-4

A network of pipelines connects two water desalinization plants to two cities. The daily supply amounts at the two plants are 40 and 50 million gallons, and the daily demand amounts at cities 1 and 2 are 30 and 60 million gallons. Nodes 1 and 2 represent plants 1 and 2, and nodes 4 and 5 represent cities 1 and 2. Node 3 is a booster station between the plants and the cities. The model is already balanced because the supply at nodes 1 and 2 equals the demand at nodes 4 and 5. Figure 6.43 gives the associated network.

FIGURE 6.43
Network for Example 6.5-4



**Iteration 0.**

**Step 0.** *Determination of a Starting Basic Feasible Solution:* The starting *feasible* spanning tree in Figure 6.44 (shown with solid arcs) is obtained by inspection. Normally, we use an artificial variable technique to find such a solution (for details, see Bazaraa et al., 1990, pp. 440–46).

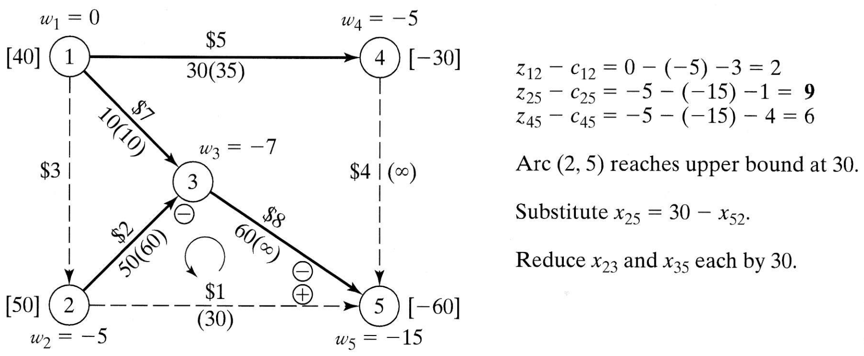



FIGURE 6.44
Network for iteration 0

In Figure 6.44, the basic feasible solution consists of (solid) arcs (1, 3), (1,4), (2, 3), and (3, 5) with the *feasible* flows of 10, 30, 50, and 60 units, respectively. This leaves (dashed) arcs (1, 2), (2, 5), and (4, 5) to represent the nonbasic variables. The notation $x(c)$ shown on the arcs indicates that a flow of $x$ units is assigned to an arc with capacity $c$. The default values for $x$ and $c$ are 0 and $\infty$, respectively.

**Iteration 1.**

**Step 1.** *Determination of the Entering Arc:* We obtain the dual values by solving the current basic equations

$$w_1 = 0$$
$$w_i - w_j = c_{ij}, \quad \text{for basic } (i, j)$$

We thus get,

$$\text{Arc }(1, 3): w_1 - w_3 = 7, \text{ hence } w_3 = -7$$
$$\text{Arc }(1, 4): w_1 - w_4 = 5, \text{ hence } w_4 = -5$$
$$\text{Arc }(2, 3): w_2 - w_3 = 2, \text{ hence } w_2 = -5$$
$$\text{Arc }(3, 5): w_3 - w_5 = 8, \text{ hence } w_5 = -15$$

Now, we compute $z_{ij} - c_{ij}$ for the nonbasic variables as

$$\text{Arc }(1, 2): w_1 - w_2 - c_{12} = 0 - (-5) - 3 = 2$$
$$\text{Arc }(2, 5): w_2 - w_5 - c_{25} = (-5) - (-15) - 1 = 9$$
$$\text{Arc }(4, 5): w_4 - w_5 - c_{45} = (-5) - (-15) - 4 = 6$$

Thus, arc (2, 5) enters the basic solution.

**Step 2.** *Determination of the Leaving Arc:* From Figure 6.44, arc (2, 5) forms a loop with basic arcs (2, 3) and (3, 5). *From the definition of the spanning tree, no other loop can be formed.* Because the flow in the new arc (2, 5) must be increased, we adjust the flow in the arcs of the loop by an equal amount to maintain the feasibility of the new solution. To achieve this, we identify the positive (+) flow in the loop by the direction of flow of the entering arc (i.e., from 2 to 5). We then assign (+) or (−) to the remaining arcs of the loop, depending on whether the flow of each arc is *with* or *against* the direction of flow of the entering arc. These sign conventions are shown in Figure 6.44.

Determination of the maximum level of flow in the entering arc (2, 5) is based on two conditions:

1. New flow in current basic arcs of the loop cannot be negative.
2. New flow in the entering arc cannot exceed its capacity.

The application of condition 1 shows that the flows in arcs (2, 3) and (3, 5) cannot be decreased by more than min {50, 60} = 50 units. Condition 2 stipulates that the flow in arc (2, 5) can be increased to at most the arc capacity (=30 units). Thus, the maximum flow change in the loop is min {30, 50} = 30 units. The new flows in the loop are thus 30 units in arc (2, 5), 50 − 30 = 20 units in arc (2, 3), and 60 − 30 = 30 units in arc (3, 5).

Because none of the current basic arcs leave the basis at zero level, the new arc (2, 5) must remain nonbasic at the upper bound. However, to avoid dealing with nonbasic arcs that are at capacity (or upper bound) level, we implement the substitution

$$x_{25} = 30 - x_{52},\ 0 \le x_{52} \le 30$$

This substitution is effected in the flow equations associated with nodes 2 and 5 as follows. Consider

$$\text{Current flow equation at node 2: } 50 + x_{12} = x_{23} + x_{25}$$

$$\text{Current flow equation at node 5: } x_{25} + x_{35} + x_{45} = 60$$

Then, the substitution $x_{25} = 30 - x_{52}$ gives

$$\text{New flow equation at node 2: } 20 + x_{12} + x_{52} = x_{23}$$

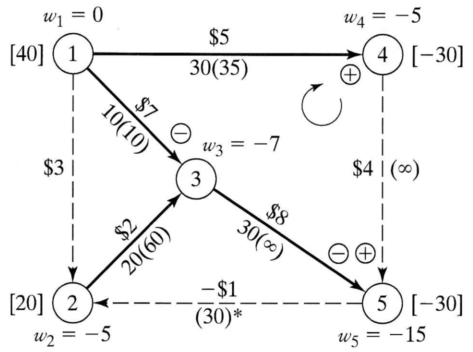
$$\text{New flow equation at node 5: } x_{35} + x_{45} = x_{52} + 30$$

The results of these changes are shown in Figure 6.45. The direction of flow in arc (2, 5) is now reversed to 5 → 2 with $x_{52} = 0$, as desired. The substitution also requires changing the unit cost of arc (5, 2) to −\$1. We will indicate this direction reversal on the network by tagging the arc with an asterisk.

**Iteration 2.** Figure 6.45 summarizes the new values of $z_{ij} - c_{ij}$ (verify!) and shows that arc (4, 5) enters the basic solution. It also defines the loop associated with the new entering arc and assigns the signs to its arcs.

The flow in arc (4, 5) can be increased by the smallest of

1. Maximum allowable *increase* in entering arc (4, 5) = ∞
2. Maximum allowable *increase* in arc (1, 4) = 35 − 30 = 5 units




$z_{12} - c_{12} = 0 - (-5) - 3 = 2$
$z_{52} - c_{52} = -15 - (-5) - (-1) = -9$
$z_{45} - c_{45} = -5 - (-15) - 4 = \mathbf{6}$

Arc (4, 5) enters at level 5.

Arc (1, 4) leaves at upper bound.

Substitute $x_{14} = 35 - x_{41}$.

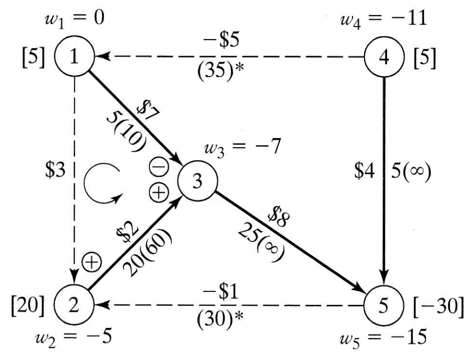
Reduce $x_{13}$ and $x_{35}$ each by 5.

FIGURE 6.45
Network for iteration 1

3. Maximum allowable *decrease* in arc (1, 3) = 10 units
4. Maximum allowable *decrease* in arc (3, 5) = 30 units

Thus, the flow in arc (4, 5) can be increased to 5 units, which will make (4, 5) basic and will force basic arc (1, 4) to be nonbasic at its upper bound ( = 35).

Using the substitution $x_{14} = 35 - x_{41}$, the network is changed as shown in Figure 6.46, with arcs (1, 3), (2, 3), (3, 5), and (4, 5) forming the basic (spanning tree) solution. The reversal of flow in arc (1, 4) changes its unit cost to −$5. Also, convince yourself that the substitution in the flow equations of nodes 1 and 4 will net 5 *input* units at each node.




$z_{12} - c_{12} = 0 - (-5) - 3 = \mathbf{2}$
$z_{41} - c_{41} = -11 - 0 - (-5) = -6$
$z_{52} - c_{52} = -15 - (-5) - (-1) = -9$

Arc (1, 2) enters at level 5.

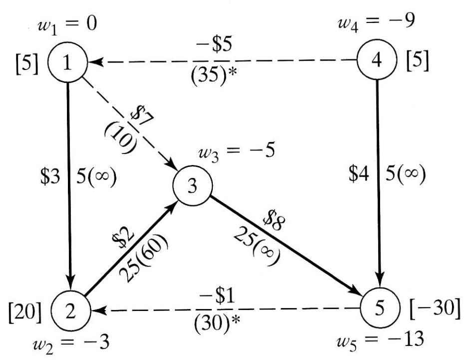
Arc (1, 3) leaves at level 0.

Increase $x_{23}$ by 5.

FIGURE 6.46
Network for iteration 2

**Iteration 3.** The computations of the new $z_{ij} - c_{ij}$ for the nonbasic arcs (1, 2), (4, 1), and (5, 2) are summarized in Figure 6.46, which shows that arc (1, 2) enters at level 5, and arc (1, 3) becomes nonbasic at level 0. The new solution is depicted in Figure 6.47.

**Iteration 4.** The new $z_{ij} - c_{ij}$ in Figure 6.47 shows that the solution is optimum. The values of the original variables are obtained by back substitution as shown in Figure 6.47.




$z_{13} - c_{13} = 0 - (-5) - 7 = -2$
$z_{41} - c_{41} = -9 - 0 - (-5) = -4$
$z_{52} - c_{52} = -13 - (-3) - (-1) = -9$

Optimum solution:
$x_{12} = 5, x_{13} = 0$
$x_{14} = 35 - 0 = 35$
$x_{23} = 25$
$x_{25} = 30 - 0 = 30$
$x_{35} = 25, x_{45} = 5$
Total cost = $490

FIGURE 6.47
Network for iteration 3

## PROBLEM SET 6.5C

1. Solve Problem 1, Set 6.5a by the capacitated simplex algorithm, and also show that it can be solved by the transshipment model.
2. Solve Problem 2, Set 6.5a by the capacitated simplex algorithm, and also show that it can be solved by the transshipment model.
3. Solve Problem 3, Set 6.5a by the capacitated simplex algorithm.
4. Solve Problem 4, Set 6.5a by the capacitated simplex algorithm.
5. Solve Problem 5, Set 6.5a by the capacitated simplex algorithm.
6. Solve the employment scheduling problem of Example 6.5-3 by the capacitated simplex algorithm.
7. Wyoming Electric uses existing slurry pipes to transport coal (carried by pumped water) from three mining areas (1, 2, and 3) to three power plants (4, 5, and 6). Each pipe can transport at most 10 tons per hour. The transportation costs per ton and the supply and demand per hour are given in the following table.

| | 4 | 5 | 6 | Supply |
|---|---|---|---|---|
| 1 | $5 | $8 | $4 | **8** |
| 2 | $6 | $9 | $12 | **10** |
| 3 | $3 | $1 | $5 | **18** |
| Demand | **16** | **6** | **14** | |

Determine the optimum shipping schedule.

8. The network in Figure 6.48 gives the distances among seven cities. Use the capacitated simplex algorithm to find the shortest distance between nodes 1 and 7. (*Hint:* Assume that nodes 1 and 7 have net flows of [+1] and [−1], respectively. All the other nodes have zero net flow.)
9. Show how the capacitated minimum-cost flow model can be specialized to represent the maximum flow model of Section 6.4. Apply the transformation to the network in Example 6.4-2. For convenience, assume that the flow capacity from 4 to 3 is zero. All the remaining data are unchanged.

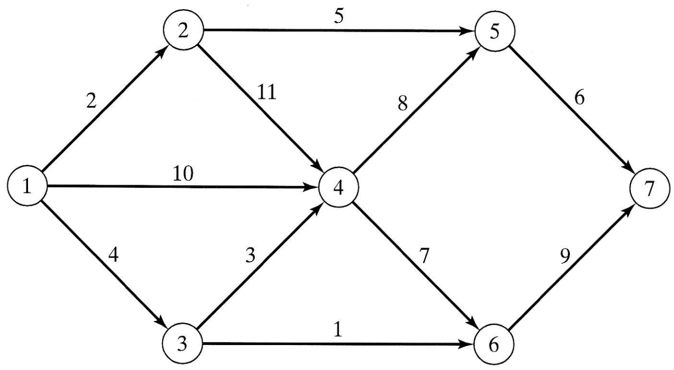



FIGURE 6.48
Network for Problem 8, Set 6.5c

### 6.5.4 Excel Spreadsheet Solution of the Minimum-Cost Capacitated Flow Model

As in the cases of the shortest-route and maximum flow models, the Excel spreadsheet developed for the general transportation model (Section 5.3.3) applies readily to the capacitated network flow model. Figure 6.49 shows the application of the spreadsheet to Example 6.5-4 (file ch6SolverMinCostCapacitatedNetwork.xls). The spreadsheet is designed for networks with a maximum of 10 nodes. In the capacity matrix (cells N6:W15),[5] a blank entry signifies an infinite capacity arc. A nonexisting arc is represented by a zero-capacity entry. As an illustration, in Example 6.5-4, infinite capacity arc 1-2 is represented by a blank entry in cell O6, and nonexisting arc 3-4 is shown by a zero entry in cell Q8. The unit cost matrix resides in cells B6:K15. We arbitrarily assign zero unit cost to all nonexisting arcs.

Once the unit cost and capacity matrices are created, the remainder of the spreadsheet (*intermediate calculations* and *optimum solution* sections) is created automatically, delineating the cells needed to update Solver parameters for Changing Cells and Constraints. Target Cell is already defined for any network (with 10 nodes or less). Specifically, for Example 6.5-4, we have,

```
Changing cells: B20:B39
Constraints: B20:B39<=C20:C39     (Arc capacity)
             F19:F23=G19:G23      (Node flow equation)
```

Figure 6.49 provides the following solution: N1-N2 $=$ 5, N1-N4 $=$ 35, N2-N3 $=$ 25, N2-N5 $=$ 30, N3-N5 $=$ 25, and N4-N5 $=$ 5. The associated total cost is \$490.

**PROBLEM SET 6.5D**

1. Solve the following problem using the spreadsheet in Section 6.5.4:
   **(a)** Problem 3, Set 6.5c
   **(b)** Problem 4, Set 6.5c

[5]In Figure 6.49, rows 11 through 15 and column K are hidden to conserve space.

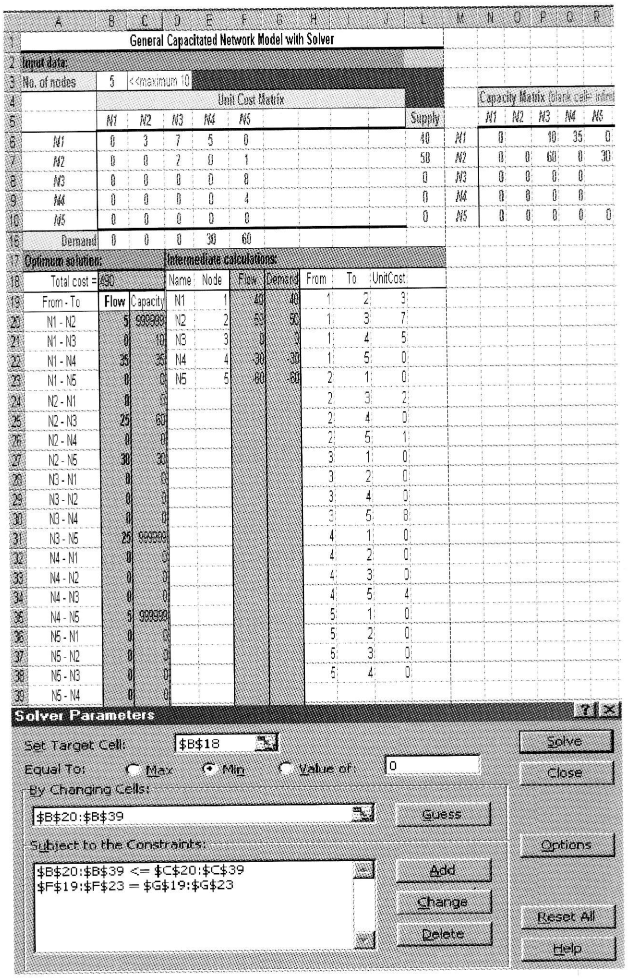



FIGURE 6.49
Excel Solver output for Example 6.5-4

**(c)** Problem 7, Set 6.5c

**(d)** Problem 8, Set 6.5c

## 6.6 CPM AND PERT

CPM (Critical Path Method) and PERT (Program Evaluation and Review Technique) are network-based methods designed to assist in the planning, scheduling, and control of projects. A project is defined as a collection of interrelated activities with each activity consuming time and resources. The objective of CPM and PERT is to provide analytic means for scheduling the activities. Figure 6.50 summarizes the steps of the techniques. First, we define the activities of the project, their precedence relationships, and their

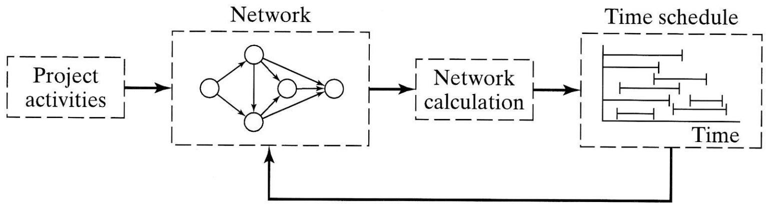



FIGURE 6.50
Phases for project planning with CPM-PERT

time requirements. Next, the project is translated into a network that shows the precedence relationships among the activities. The third step involves specific network computations that form the basis for the development of the time schedule for the project.

During the execution of the project, the schedule may not be realized as planned, causing some of the activities to be expedited or delayed. In this case, it will be necessary to update the schedule to reflect the realities on the ground. This is the reason for including a feedback loop between the time schedule phase and the network phase as shown in Figure 6.50.

The two techniques, CPM and PERT, which were developed independently, differ in that CPM assumes deterministic activity durations, whereas PERT assumes probabilistic durations. This presentation will start with CPM and then provide the details of PERT.

### 6.6.1 Network Representation

Each activity of the project is represented by an arc pointing in the direction of progress in the project. The nodes of the network establish the precedence relationships among the different activities of the project.

Two rules are available for constructing the network.

**Rule 1.** *Each activity is represented by one, and only one, arc.*

**Rule 2.** *Each activity must be identified by two distinct end nodes.*

Figure 6.51 shows how a dummy activity can be used to represent two concurrent activities, $A$ and $B$. By definition, a dummy activity, which normally is depicted by a dashed arc, consumes no time or resources. Inserting a dummy activity in one of the four ways shown in Figure 6.51, we maintain the concurrence of $A$ and $B$, and also provide unique end nodes for the two activities (to satisfy rule 2).

**Rule 3.** *To maintain the correct precedence relationships, the following questions must be answered as each activity is added to the network:*

**(a)** *What activities must immediately precede the current activity?*

**(b)** *What activities must follow the current activity?*

**(c)** *What activities must occur concurrently with the current activity?*

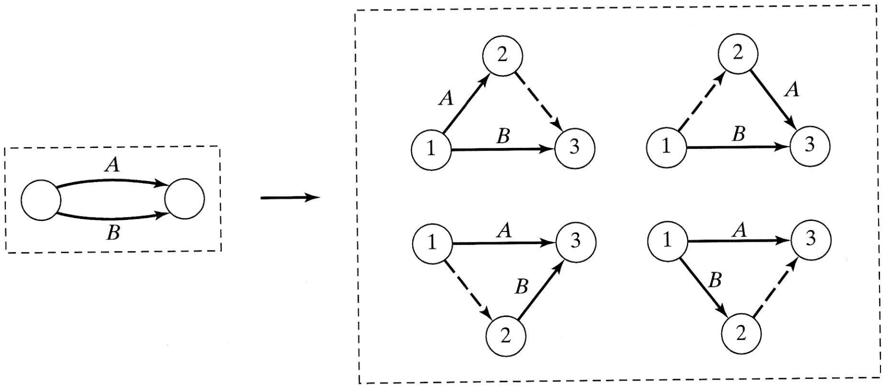


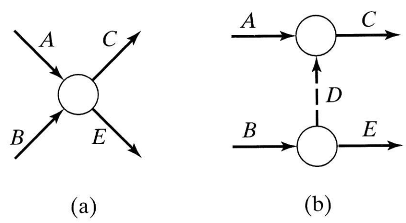

FIGURE 6.51
Use of dummy activity to produce unique representation of concurrent activities *A* and *B*

The answers to these questions may require the use of dummy activities to ensure correct precedences among the activities. For example, consider the following segment of a project:

1. Activity *C* starts immediately after *A* and *B* have been completed.
2. Activity *E* starts after *B* only has been completed.

Part (a) of Figure 6.52 shows the incorrect representation of the precedence relationship because it requires both *A* and *B* to be completed before *E* can start. In part (b), the use of a dummy activity rectifies the situation.

FIGURE 6.52
Use of dummy activity to ensure correct precedence relationship




## Example 6.6-1

A publisher has a contract with an author to publish a textbook. The (simplified) activities associated with the production of the textbook are given below. Develop the associated network for the project.

| Activity | Predecessor(s) | Duration (weeks) |
|---|---|---|
| *A*: Manuscript proofreading by editor | — | 3 |
| *B*: Sample pages prepared by typesetter | — | 2 |
| *C*: Book cover design | — | 4 |
| *D*: Preparation of artwork for book figures | — | 3 |
| *E*: Author's approval of edited manuscript and sample pages | *A*, *B* | 2 |

| | | |
|---|---|---|
| *F*: Book typesetting | *E* | 2 |
| *G*: Author checks typeset pages | *F* | 2 |
| *H*: Author checks artwork | *D* | 1 |
| *I*: Production of printing plates | *G, H* | 2 |
| *J*: Book production and binding | *C, I* | 4 |

Figure 6.53 provides the network describing the precedence relationships among the different activities. Dummy activity (2, 3) produces unique end nodes for concurrent activities *A* and *B*. The numbering of the nodes is done in a manner that indicates the direction of progress in the project.

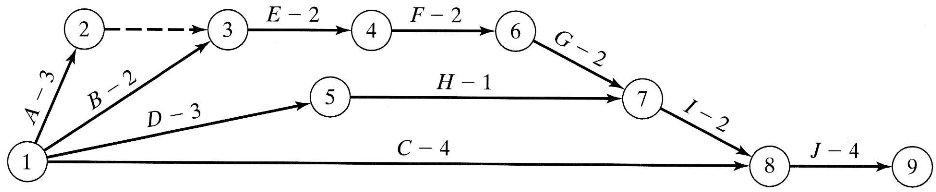



FIGURE 6.53

Project network for Example 6.6-1

## PROBLEM SET 6.6A

1. Construct the project network comprised of activities *A* to *L* with the following precedence relationships:
   - **(a)** *A, B,* and *C,* the first activities of the project, can be executed concurrently.
   - **(b)** *A* and *B* precede *D*.
   - **(c)** *B* precedes *E, F,* and *H*.
   - **(d)** *F* and *C* precede *G*.
   - **(e)** *E* and *H* precede *I* and *J*.
   - **(f)** *C, D, F,* and *J* precede *K*.
   - **(g)** *K* precedes *L*.
   - **(h)** *I, G,* and *L* are the terminal activities of the project.
2. Construct the project network comprised of activities *A* to *P* that satisfies the following precedence relationships:
   - **(a)** *A, B,* and *C,* the first activities of the project, can be executed concurrently.
   - **(b)** *D, E,* and *F* follow *A*.
   - **(c)** *I* and *G* follow both *B* and *D*.
   - **(d)** *H* follows both *C* and *G*.
   - **(e)** *K* and *L* follow *I*.
   - **(f)** *J* succeeds both *E* and *H*.
   - **(g)** *M* and *N* succeed *F,* but cannot start until both *E* and *H* are completed.
   - **(h)** *O* succeeds *M* and *I*.
   - **(i)** *P* succeeds *J, L,* and *O*.
   - **(j)** *K, N,* and *P* are the terminal activities of the project.

3. The footings of a building can be completed in four connected sections. The activities for each section include (1) digging, (2) placing steel, and (3) pouring concrete. The digging of one section cannot start until that of the preceding section has been completed. The same restriction applies to pouring concrete. Develop the project network.
4. In Problem 3, suppose that 10% of the plumbing work can be started simultaneously with the digging of the first section. After each section of the footings is completed, an additional 5% of the plumbing can be started provided that the preceding 5% portion is completed. The remaining plumbing can be completed at the end of the project. Construct the project network.
5. An opinion survey involves designing and printing questionnaires, hiring and training personnel, selecting participants, mailing questionnaires, and analyzing the data. Construct the project network, stating all assumptions.
6. The activities in the following table describe the construction of a new house. Develop the associated project network.

| Activity | Predecessor(s) | Duration (days) |
|---|---|---|
| *A*: Clear site | — | 1 |
| *B*: Bring utilities to site | — | 2 |
| *C*: Excavate | *A* | 1 |
| *D*: Pour foundation | *C* | 2 |
| *E*: Outside plumbing | *B*, *C* | 6 |
| *F*: Frame house | *D* | 10 |
| *G*: Do electric wiring | *F* | 3 |
| *H*: Lay floor | *G* | 1 |
| *I*: Lay roof | *F* | 1 |
| *J*: Inside plumbing | *E*, *H* | 5 |
| *K*: Shingling | *I* | 2 |
| *L*: Outside sheathing insulation | *F*, *J* | 1 |
| *M*: Install windows and outside doors | *F* | 2 |
| *N*: Do brick work | *L*, *M* | 4 |
| *O*: Insulate walls and ceiling | *G*, *J* | 2 |
| *P*: Cover walls and ceiling | *O* | 2 |
| *Q*: Insulate roof | *I*, *P* | 1 |
| *R*: Finish interior | *P* | 7 |
| *S*: Finish exterior | *I*, *N* | 7 |
| *T*: Landscape | *S* | 3 |

7. A company is in the process of preparing a budget for launching a new product. The following table provides the associated activities and their durations. Construct the project network.

| Activity | Predecessor(s) | Duration (days) |
|---|---|---|
| *A*: Forecast sales volume | — | 10 |
| *B*: Study competitive market | — | 7 |
| *C*: Design item and facilities | *A* | 5 |
| *D*: Prepare production schedule | *C* | 3 |
| *E*: Estimate cost of production | *D* | 2 |
| *F*: Set sales price | *B*, *E* | 1 |
| *G*: Prepare budget | *E*, *F* | 14 |

**8.** The activities involved in a candlelight choir service are listed in the following table. Construct the project network.

| Activity | Predecessor(s) | Duration (days) |
|---|---|---|
| *A*: Select music | — | 2 |
| *B*: Learn music | *A* | 14 |
| *C*: Make copies and buy books | *A* | 14 |
| *D*: Tryouts | *B*, *C* | 3 |
| *E*: Rehearsals | *D* | 70 |
| *F*: Rent candelabra | *D* | 14 |
| *G*: Decorate candelabra | *F* | 1 |
| *H*: Set up decorations | *D* | 1 |
| *I*: Order choir robe stoles | *D* | 7 |
| *J*: Check out public address system | *D* | 7 |
| *K*: Select music tracks | *J* | 14 |
| *L*: Set up public address system | *K* | 1 |
| *M*: Final rehearsal | *E*, *G*, *L* | 1 |
| *N*: Choir party | *H*, *L*, *M* | 1 |
| *O*: Final program | *I*, *N* | 1 |

**9.** The widening of a road section requires relocating ("reconductoring") 1700 feet of 13.8-kV overhead primary line. The following table summarizes the activities of the project. Construct the associated project network.

| Activity | Predecessor(s) | Duration (days) |
|---|---|---|
| *A*: Job review | — | 1 |
| *B*: Advise customers of temporary outage | *A* | .5 |
| *C*: Requisition stores | *A* | 1 |
| *D*: Scout job | *A* | .5 |
| *E*: Secure poles and material | *C*, *D* | 3 |
| *F*: Distribute poles | *E* | 3.5 |
| *G*: Pole location coordination | *D* | .5 |
| *H*: Re-stake | *G* | .5 |
| *I*: Dig holes | *H* | 3 |
| *J*: Frame and set poles | *F*, *I* | 4 |
| *K*: Cover old conductors | *F*, *I* | 1 |
| *L*: Pull new conductors | *J*, *K* | 2 |
| *M*: Install remaining material | *L* | 2 |
| *N*: Sag conductor | *L* | 2 |
| *O*: Trim trees | *D* | 2 |
| *P*: De-energize and switch lines | *B*, *M*, *N*, *O* | .1 |
| *Q*: Energize and switch new line | *P* | .5 |
| *R*: Clean up | *Q* | 1 |
| *S*: Remove old conductor | *Q* | 1 |
| *T*: Remove old poles | *S* | 2 |
| *U*: Return material to stores | *R*, *T* | 2 |

**10.** The following table gives the activities for buying a new car. Construct the project network.

| Activity | Predecessor(s) | Duration (days) |
|---|---|---|
| $A$: Conduct feasibility study | — | 3 |
| $B$: Find potential buyer for present car | $A$ | 14 |
| $C$: List possible models | $A$ | 1 |
| $D$: Research all possible models | $C$ | 3 |
| $E$: Conduct interview with mechanic | $C$ | 1 |
| $F$: Collect dealer propaganda | $C$ | 2 |
| $G$: Compile pertinent data | $D, E, F$ | 1 |
| $H$: Choose top three models | $G$ | 1 |
| $I$: Test-drive all three choices | $H$ | 3 |
| $J$: Gather warranty and financing data | $H$ | 2 |
| $K$: Choose one car | $I, J$ | 2 |
| $L$: Choose dealer | $K$ | 2 |
| $M$: Search for desired color and options | $L$ | 4 |
| $N$: Test-drive chosen model once again | $L$ | 1 |
| $O$: Purchase new car | $B, M, N$ | 3 |

### 6.6.2 Critical Path (CPM) Computations

The ultimate result in CPM is the construction of the time schedule for the project (see Figure 6.50). To achieve this objective conveniently, we carry out special computations that produce the following information:

1. Total duration needed to complete the project
2. Classification of the activities of the project as *critical* and *noncritical*

An activity is said to be **critical** if there is no "leeway" in determining its start and finish times. A **noncritical** activity allows some scheduling slack, so that the start time of the activity may be advanced or delayed within limits without affecting the completion date of the entire project.

To carry out the necessary computations, we define an **event** as a point in time at which activities are terminated and others are started. In terms of the network, an event corresponds to a node. Define

$$\square_j = \text{Earliest occurrence time of event } j$$

$$\Delta_j = \text{Latest occurrence time of event } j$$

$$D_{ij} = \text{Duration of activity } (i, j)$$

The definitions of the *earliest* and *latest* occurrence times of event $j$ are specified relative to the start and completion dates of the entire project.

The critical path calculations involve two passes: The **forward pass** determines the *earliest* occurrence times of the events, and the **backward pass** calculates their latest occurrence times.

**Forward Pass (Earliest Occurrence Times, $\square$).** The computations start at node 1 and advance recursively to end node $n$.

**Initial Step.** Set $\square_1 = 0$ to indicate that the project starts at time 0.

**General Step *j*.** Given that nodes $p, q, \ldots,$ and $v$ are linked *directly* to node $j$ by incoming activities $(p, j), (q, j), \ldots,$ and $(v, j)$ and that the earliest occurrence times of events (nodes) $p, q, \ldots,$ and $v$ have already been computed, then the earliest occurrence time of event $j$ is computed as

$$\square_j = \max\{\square_p + D_{pj}, \square_q + D_{qj}, \ldots, \square_v + D_{vj}\}$$

The forward pass is complete when $\square_n$ at node $n$ has been computed. By definition $\square_j$ represents the longest path (duration) to node $j$.

**Backward Pass (Latest Occurrence Times, $\Delta$).** Following the completion of the forward pass, the backward pass computations start at node $n$ and end at node 1.

**Initial Step.** Set $\Delta_n = \square_n$ to indicate that the earliest and latest occurrences of the last node of the project are the same.

**General Step *j*.** Given that nodes $p, q, \ldots,$ and $v$ are linked *directly* to node $j$ by *outgoing* activities $(j, p), (j, q), \ldots,$ and $(j, v)$ and that the latest occurrence times of nodes $p, q, \ldots,$ and $v$ have already been computed, the latest occurrence time of node $j$ is computed as

$$\Delta_j = \min\{\Delta_p - D_{jp}, \Delta_q - D_{jq}, \ldots, \Delta_v - D_{jv}\}$$

The backward pass is complete when $\Delta_1$ at node 1 is computed.

Based on the preceding calculations, an activity $(i, j)$ will be *critical* if it satisfies three conditions.

1. $\Delta_i = \square_i$
2. $\Delta_j = \square_j$
3. $\Delta_j - \Delta_i = \square_j - \square_i = D_{ij}$

The three conditions state that the earliest and latest occurrence times of nodes $i$ and $j$ are equal, and the duration $D_{ij}$ fits "tightly" in the specified time span. An activity that does not satisfy all three conditions is *noncritical.*

The critical activities of a network must constitute an uninterrupted path that spans the entire network from start to finish.

## Example 6.6-2

Determine the critical path for the project network in Figure 6.54. All the durations are in days.

**Forward Pass**

**Node 1.** Set $\square_1 = 0$

**Node 2.** $\square_2 = \square_1 + D_{12} = 0 + 5 = 5$

**Node 3.** $\square_3 = \max\{\square_1 + D_{13}, \square_2 + D_{23}\} = \max\{0 + 6, 5 + 3\} = 8$
**Node 4.** $\square_4 = \square_2 + D_{24} = 5 + 8 = 13$
**Node 5.** $\square_5 = \max\{\square_3 + D_{35}, \square_4 + D_{45}\} = \max\{8 + 2, 13 + 0\} = 13$
**Node 6.** $\square_6 = \max\{\square_3 + D_{36}, \square_4 + D_{46}, \square_5 + D_{56}\}$
$= \max\{8 + 11, 13 + 1, 13 + 12\} = 25$

The computations show that the project can be completed in 25 days.

**Backward Pass**

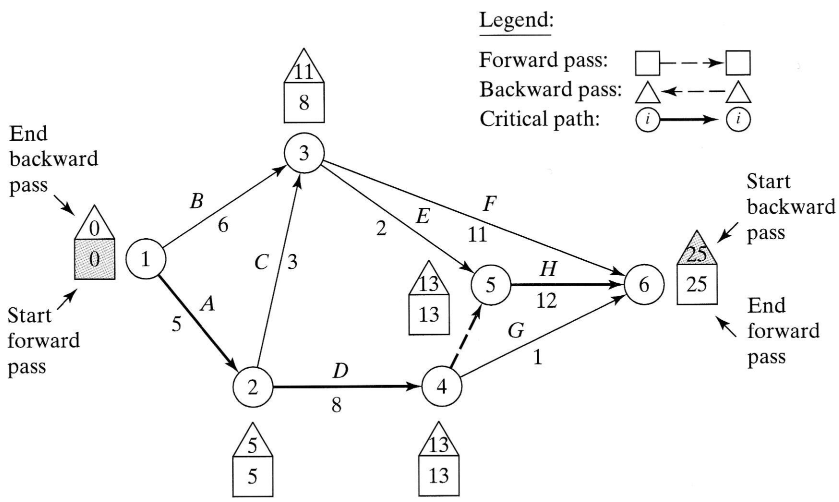
**Node 6.** Set $\Delta_6 = \square_6 = 25$
**Node 5.** $\Delta_5 = \Delta_6 - D_{56} = 25 - 12 = 13$
**Node 4.** $\Delta_4 = \min\{\Delta_6 - D_{46}, \Delta_5 - D_{45}\} = \min\{25 - 1, 13 - 0\} = 13$
**Node 3.** $\Delta_3 = \min\{\Delta_6 - D_{36}, \Delta_5 - D_{35}\} = \min\{25 - 11, 13 - 2\} = 11$
**Node 2.** $\Delta_2 = \min\{\Delta_4 - D_{24}, \Delta_3 - D_{23}\} = \min\{13 - 8, 11 - 3\} = 5$
**Node 1.** $\Delta_1 = \min\{\Delta_3 - D_{13}, \Delta_2 - D_{12}\} = \min\{11 - 6, 5 - 5\} = 0$

Correct computations will always end with $\Delta_1 = 0$.

The forward and backward pass computations are summarized in Figure 6.54. The rules for determining the critical activities show that the critical path is defined by $1 \rightarrow 2 \rightarrow 4 \rightarrow 5 \rightarrow 6$, which spans the network from start (node 1) to finish (node 6). The sum of the durations of the critical activities [(1, 2), (2, 4), (4, 5), and (5, 6)] equals the duration of the project (= 25 days). Observe that activity (4, 6) satisfies the first two conditions for a critical activity ($\Delta_4 = \square_4 = 13$ and $\Delta_5 = \square_5 = 25$) but not the third ($\square_6 - \square_4 \neq D_{46}$). Hence, the activity is not critical.

FIGURE 6.54

Forward and backward pass calculations for the project of Example 6.6-2



## PROBLEM SET 6.6B

**1.** Determine the critical path for the project network in Figure 6.55.

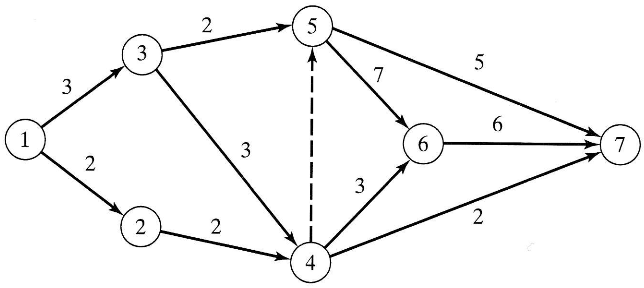
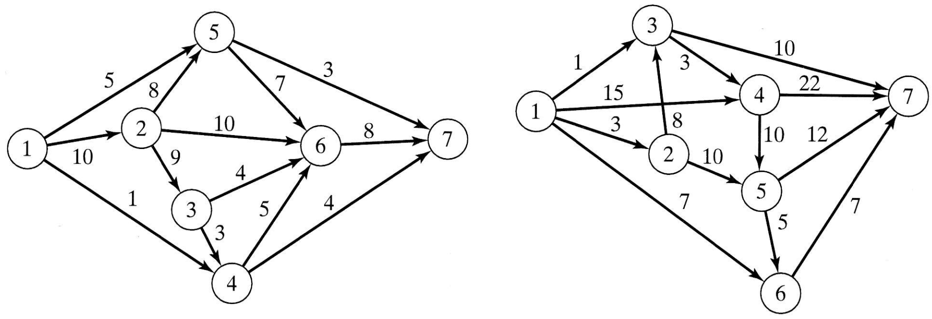


FIGURE 6.55
Project network for Problem 1, Set 6.6b

**2.** Determine the critical path for the project networks in Figure 6.56.



Project (a) Project (b)

FIGURE 6.56
Project network for Problem 2, Set 6.6b

**3.** Determine the critical path for the project in Problem 6, Set 6.6a.
**4.** Determine the critical path for the project in Problem 8, Set 6.6a.
**5.** Determine the critical path for the project in Problem 9, Set 6.6a.
**6.** Determine the critical path for the project in Problem 10, Set 6.6a.

### 6.6.3 Construction of the Time Schedule

This section shows how the information obtained from the calculations in Section 6.6.2 can be used to develop the time schedule. We recognize that for an activity $(i, j)$, $\square_i$ represents the *earliest start time,* and $\Delta_j$ represents the *latest completion time.* This means that $(\square_i, \Delta_j)$ delineates the (maximum) span during which activity $(i, j)$ may be scheduled.

**Construction of Preliminary Schedule.** The method for constructing a preliminary schedule is illustrated by an example.

### Example 6.6-3

Determine the time schedule for the project of Example 6.6-2 (Figure 6.54).

We can get a preliminary time schedule for the different activities of the project by delineating their respective time spans as shown in Figure 6.57. Two observations are in order.

1. The critical activities (shown by solid lines) must be scheduled one right after the other to ensure that the project is completed within its specified 25-day duration.
2. The noncritical activities (shown by dashed lines) encompass spans that are larger than their respective durations, thus allowing slack (or "leeway") in scheduling them within their allotted spans.

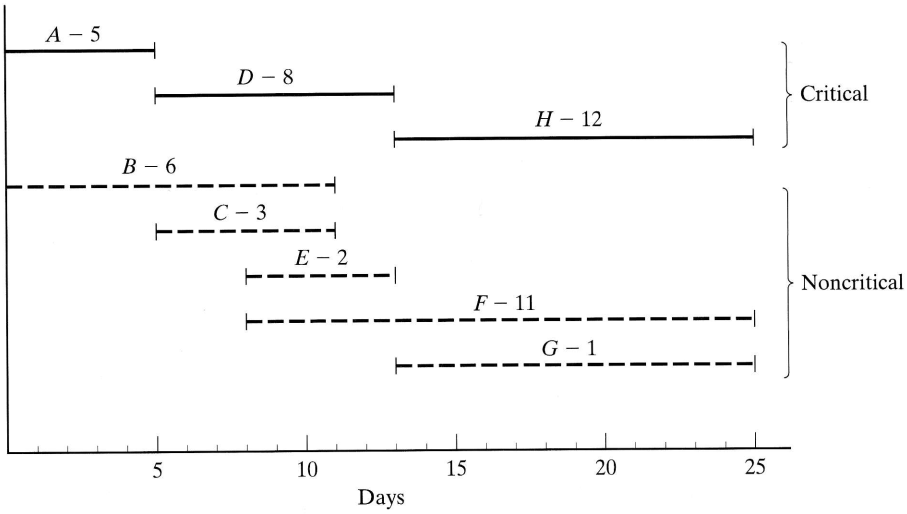



FIGURE 6.57
Preliminary schedule for the project of Example 6.6-2

How should we schedule the noncritical activities within their respective spans? Normally, it is preferable to start each noncritical activity as early as possible. In this manner, slack periods will remain opportunely available at the end of the allotted span, where they can be used to absorb unexpected delays in the execution of the activity. It may be necessary, however, to delay the start of a noncritical activity past its earliest time. For example, in Figure 6.57, suppose that each of the noncritical activities $E$ and $F$ requires the use of a bulldozer, and that only one is available. Scheduling both $E$ and $F$ as early as possible requires two bulldozers between times 8 and 10. We can remove the overlap by starting $E$ at time 8 and pushing the start time of $F$ to somewhere between times 10 and 14.

If all the noncritical activities can be scheduled as early as possible, the resulting schedule automatically is feasible. Otherwise, some precedence relationships may be violated if noncritical activities are delayed past their earliest time. Take, for example, activities $C$ and $E$ in Figure 6.57. In the project network (Figure 6.54), though C must be completed before $E$, the spans of $C$ and $E$ in Figure 6.57 allow us to schedule $C$ between times 6 and 9, and E between times 8 and 10. These spans, however, do not ensure that $C$ will precede $E$. The need for a "red flag" that automatically reveals schedule conflict is thus evident. Such information is provided by computing the *floats* for the noncritical activities.

---

**Determination of the Floats.** Floats are the slack times available within the allotted span of the noncritical activity. The two most common floats are the **total float** and the **free float**.

Figure 6.58 gives a convenient summary for computing the total float ($TF_{ij}$) and the free float ($FF_{ij}$) for an activity $(i, j)$. The total float is the excess of the time span defined from the *earliest* occurrence of event $i$ to the *latest* occurrence of event $j$ over the duration of $(i, j)$—that is,

$$TF_{ij} = \Delta_j - \square_i - D_{ij}$$

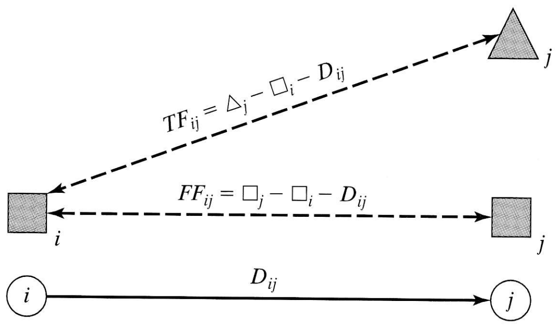



FIGURE 6.58
Computation of total and free floats

The free float is the excess of the time span defined from the *earliest* occurrence of event $i$ to the *earliest* occurrence of event $j$ over the duration of $(i, j)$—that is,

$$FF_{ij} = \square_j - \square_i - D_{ij}$$

By definition, $FF_{ij} \leq TF_{ij}$.

***Red-Flagging Rule.*** *For a noncritical activity (i, j)*

**(a)** *If $FF_{ij} = TF_{ij}$, then the activity can be scheduled anywhere within its $(\square_i, \Delta_j)$ span without causing schedule conflict.*

**(b)** *If $FF_{ij} < TF_{ij}$, then the start of activity $(i, j)$ can be delayed by at most $FF_{ij}$ relative to its earliest start time ($\square_i$) without causing schedule conflict. Any delay larger than $FF_{ij}$ (but not more than $TF_{ij}$) must be accompanied by an equal delay relative to $\square_j$ in the start time of all the activities leaving node j.*

The implication of the rule is that a noncritical activity $(i, j)$ will be red-flagged if its $FF_{ij} < TF_{ij}$. This red flag is important only if we decide to delay the start of the activity past its earliest start time, $\square_i$, in which case we must pay attention to the start times of the activities leaving node $j$ to avoid schedule conflicts.

## Example 6.6-4

Compute the floats for the noncritical activities of the network in Example 6.6-2, and discuss their use in finalizing a schedule for the project.

The following table summarizes the computations of the total and free floats. It is more convenient to do the calculations directly on the network using the procedure in Figure 6.54.

| Noncritical activity | Duration | Total float ($TF$) | Free float ($FF$) |
|---|---|---|---|
| $B(1,3)$ | 6 | $11 - 0 - 6 = 5$ | $8 - 0 - 6 = 2$ |
| $C(2,3)$ | 3 | $11 - 5 - 3 = 3$ | $8 - 5 - 3 = 0$ |
| $E(3,5)$ | 2 | $13 - 8 - 2 = 3$ | $13 - 8 - 2 = 3$ |
| $F(3,6)$ | 11 | $25 - 8 - 11 = 6$ | $25 - 8 - 11 = 6$ |
| $G(4,6)$ | 1 | $25 - 13 - 1 = 11$ | $25 - 13 - 1 = 11$ |

The computations red-flag activities $B$ and $C$ because their $FF < TF$. The remaining activities ($E$, $F$, and $G$) have $FF = TF$, and hence may be scheduled anywhere between their earliest start and latest completion times.

To investigate the significance of the red-flagged activities, consider activity $B$. Because its $TF = 5$ days, this activity can start as early as time 0 or as late as time 5 (see Figure 6.57). However, because its $FF = 2$ days, starting $B$ anywhere between time 0 and time 2 will have no effect on the succeeding activities $E$ and $F$. If, however, activity $B$ must start at time $2 + d\,(<5)$, then the start times of the immediately succeeding activities $E$ and $F$ must be pushed forward past their earliest start time ($= 8$) by at least $d$. In this manner, the precedence relationship between $B$ and its successors $E$ and $F$ is preserved.

Turning to red-flagged activity $C$, we note that its $FF = 0$. This means that *any* delay in starting $C$ past its earliest start time ($= 5$) must be coupled with at least an equal delay in the start of its successor activities $E$ and $F$.

---

TORA provides useful tutorial tools for CPM calculations and for constructing the time schedule. To use these tools, select `Project Planning` ⇒ `CPM-Critical Path Method` from `Main Menu`. In the output screen, you have the option to select `CPM Calculations` to produce step-by-step computations of the forward pass, backward pass, and the floats or `CPM Bar Chart` to construct and experiment with the time schedule.

Figure 6.59 shows TORA output for the CPM calculations of Example 6.6-2 (file ch6ToraCPMEx6-6-2.xls). If you elect to generate the output using the `Next Step` option, TORA will guide you through the details of the forward and backward pass calculations.

Figure 6.60 provides the TORA schedule produced by `CPM Bar Chart` option for the project of Example 6.6-2. The default bar chart automatically schedules all the noncritical activities as early as possible. You can study the impact of delaying the start time of a noncritical activity by using the self-explanatory drop-down lists inside the bottom left frame of the screen. The impact of a delay of a noncritical activity will be shown directly on the bar chart together with an accompanying explanation. For example, if you delay the start of activity $B$ by more than 2 time units, the succeeding activities $E$ and $F$ will be delayed by an amount equal to the difference between the delay

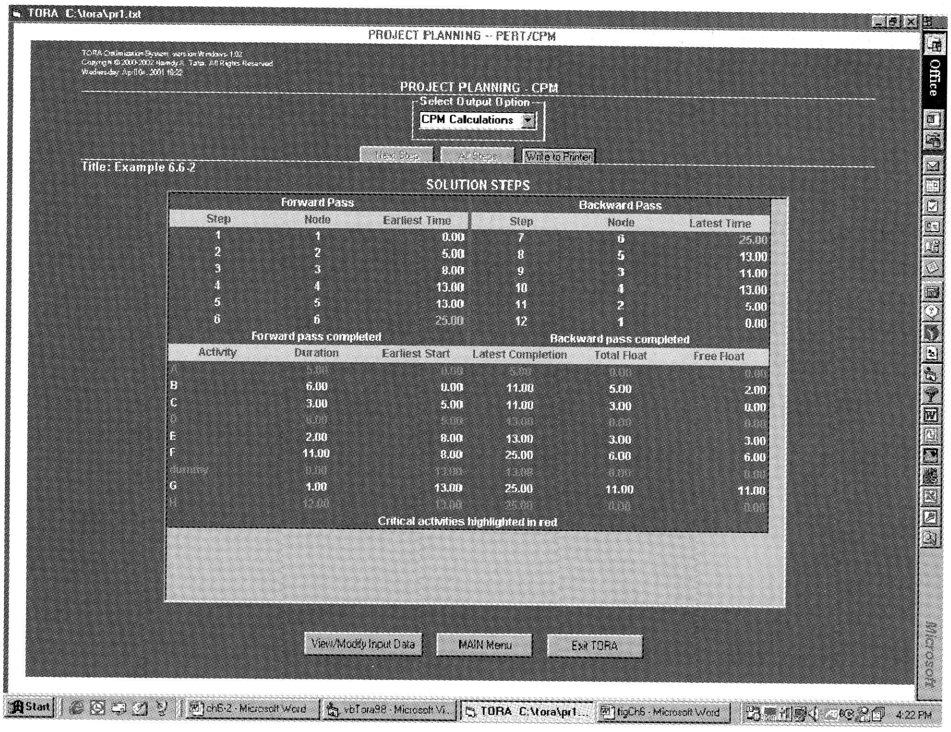



FIGURE 6.59

TORA step-by-step CPM calculations of forward pass, backward pass, and floats for Example 6.6-2

and the free float of activity *B*. Specifically, given the free float for *B* is 2 time units, if *B* is delayed by 3 time units, then *E* and *F* must be delayed by at least $3 - 2 = 1$ time unit. This situation is demonstrated in Figure 6.60.

## PROBLEM SET 6.6C

1. Given an activity $(i, j)$ with duration $D_{ij}$ and its earliest start time $\square_i$ and its latest completion time $\Delta_j$, determine the earliest completion and the latest start times of $(i, j)$.
2. What are the total and free floats of a critical activity?
3. For each of the following activities, determine the maximum delay in the starting time relative to its earliest start time that will allow all the immediately succeeding activities to be scheduled anywhere between their earliest and latest completion times.
   - **(a)** $TF = 10,\ FF = 10,\ D = 4$
   - **(b)** $TF = 10,\ FF = 5,\ D = 4$
   - **(c)** $TF = 10,\ FF = 0,\ D = 4$
4. In Example 6.6-4, use the floats to answer the following:
   - **(a)** Suppose that activity *B* is started at time 1, and activity *C* is started at time 5, determine the earliest start times for *E* and *F*.

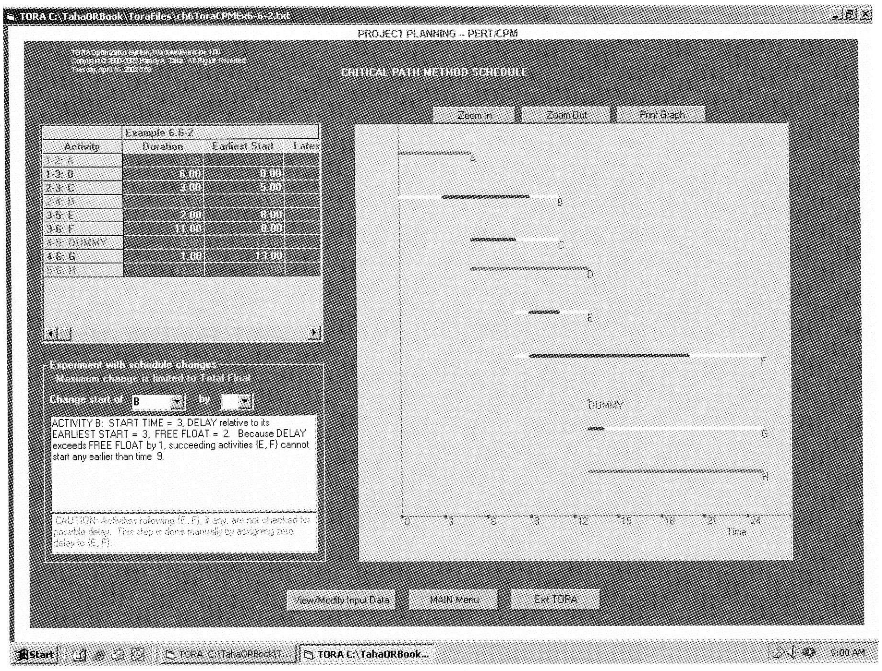



FIGURE 6.60

TORA bar chart output for Example 6.6-2

**(b)** Suppose that activity $B$ is started at time 3, and activity $C$ is started at time 7, determine the earliest start times for $E$ and $F$.

**(c)** If activity $B$ starts at time 6, what effect will this have on other activities of the project?

**5.** In the project of Example 6.6-2 (Figure 6.54), assume that the durations of activities $B$ and $F$ are changed from 6 and 11 days to 20 and 25 days, respectively.

**(a)** Determine the critical path.

**(b)** Determine the total and free floats for the network, and identify the red-flagged activities.

**(c)** Suppose that activity $A$ is started at time 5, determine the earliest possible start times for activities $C$, $D$, $E$, and $G$.

**(d)** Suppose that activities $F$, $G$, and $H$ require the same equipment. Determine the minimum number of units needed of this equipment.

**6.** Compute the floats and identify the red-flagged activities for projects (a) and (b) in Figure 6.56, and then develop the time schedules under the following conditions:

**Project (a)**

**(i)** Activity (1, 5) cannot start any earlier than time 14.

**(ii)** Activities (5, 6) and (5, 7) use the same equipment, of which only one unit is available.

**(iii)** All other activities start as early as possible.

**Project (b)**

**(i)** Activity (1, 3) must be scheduled at its earliest start time while accounting for the requirement that (1, 2), (1, 3), and (1, 6) use special equipment, of which 1 unit only is available.

**(ii)** All other activities start as early as possible.

### 6.6.4 Linear Programming Formulation of CPM

A CPM problem can be thought of as the opposite of the shortest-route problem (Section 6.3), in the sense that we are interested in finding the *longest* route from start to finish. We can thus apply the shortest-route LP formulation in Section 6.3.3 to CPM in the following manner. We assume that a unit flow enters the network at the start node and leaves at the finish node. Define

$$x_{ij} = \text{Amount of flow in activity } (i,j) \text{ for all defined } i \text{ and } j$$

$$D_{ij} = \text{Duration of activity } (i,j) \text{ for all defined } i \text{ and } j$$

Thus, the objective function of the linear program becomes

$$\text{Maximize } z = \sum_{\text{all defined activities } (i,\,j)} D_{ij}x_{ij}$$

(Compare with the shortest-route LP formulation in Section 6.3.3 where the objective function is minimized.) There is one constraint that represents the conservation of flow at each node—that is, for all node $j$,

$$\text{Total input flow} = \text{Total output flow}$$

Naturally, all the variables, $x_{ij}$, are nonnegative. Note that one of the constraints is redundant.

Again, as in the shortest-route problem, we can use the dual of the LP to solve the CPM problem. The following example applies the two formulations to the project in Example 6.6-2.

#### Example 6.6-5

The LP formulation of the project of Example 6.6-2 (Figure 6.54) is given below. Note that nodes 1 and 6 are the start and finish nodes, respectively.

| | *A* | *B* | *C* | *D* | *E* | *F* | Dummy | *G* | *H* | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $x_{12}$ | $x_{13}$ | $x_{23}$ | $x_{24}$ | $x_{35}$ | $x_{36}$ | $x_{45}$ | $x_{46}$ | $x_{56}$ | |
| Maximize z = | 5 | 6 | 3 | 8 | 2 | 11 | 0 | 1 | 12 | |
| Node 1 | −1 | −1 | | | | | | | | = −1 |
| Node 2 | 1 | | −1 | −1 | | | | | | = 0 |
| Node 3 | | 1 | 1 | | −1 | −1 | | | | = 0 |
| Node 4 | | | | 1 | | | −1 | −1 | | = 0 |
| Node 5 | | | | | 1 | | 1 | | −1 | = 0 |
| Node 6 | | | | | | 1 | | 1 | 1 | = 1 |

TORA gives the optimum solution as

$$z = 25,\ x_{12}(A) = 1,\ x_{24}(D) = 1,\ x_{45}\,(\text{Dummy}) = 1,\ x_{56}(H) = 1,\ \text{ all others} = 0$$

The solution defines the critical path as $A \rightarrow D \rightarrow \text{Dummy} \rightarrow H$, and the duration of the project is 25 days.

The dual problem of the LP given above is:

$$\text{Minimize } w = y_6 - y_1$$

subject to

$$\begin{aligned}
y_2 - y_1 &\geq 5 && (A)\\
y_3 - y_1 &\geq 6 && (B)\\
y_3 - y_2 &\geq 3 && (C)\\
y_4 - y_2 &\geq 8 && (D)\\
y_5 - y_3 &\geq 2 && (E)\\
y_6 - y_3 &\geq 11 && (F)\\
y_5 - y_4 &\geq 0 && (\text{Dummy})\\
y_6 - y_4 &\geq 1 && (G)\\
y_6 - y_5 &\geq 12 && (H)\\
&\text{all } y_i \text{ unrestricted}
\end{aligned}$$

The dual formulation, though purely mathematical, reveals an interesting definition of the dual variables that is consistent with the precedence relationships of the CPM network. Specifically, consider the following definition:

$$y_j = \text{Occurrence time of node } j$$

In this case, $w = y_6 - y_1$ will represent the duration of the project. Each constraint is associated with an activity, and it specifies the precedence relationships among the different activities. For example, $y_2 - y_1 \geq 5$ is equivalent to $y_2 \geq y_1 + 5$, which says that $y_2$, the earliest occurrence time for node 2, cannot be any earlier than time $y_1 + 5$. By minimizing the objective function, we obtain the shortest time span in which all precedence relationships are satisfied. Also, notice that with the (new) practical meaning used to describe the dual variables, these variables can be restricted to nonnegative values. In fact, the start time, $y_1$, of the project can be set equal to zero, in which case the objective function reduces to minimizing $w = y_6$. Setting $y_1 = 0$ is also consistent with the fact that one of the primal constraints is redundant.

Under the nonnegativity restriction, the optimal dual solution (obtained by TORA) is given as

$$w = 25,\ y_1 = 0,\ y_2 = 5,\ y_3 = 11,\ y_4 = 13,\ y_5 = 13,\ y_6 = 25$$

The solution shows that the duration of the project is $w = 25$ days.

The critical activities correspond to the constraints that are satisfied as strict equations by the given solution; namely, $A \rightarrow D \rightarrow \text{Dummy} \rightarrow H$. These constraints are identified by their zero surplus variables or by realizing that if a constraint is satisfied in equation form in the solution, then its associated dual value must be positive.

Indeed, pairing the constraints with their associated dual solution (as determined by TORA) we get,

| Constraint | $A$ | $B$ | $C$ | $D$ | $E$ | $F$ | Dummy | $G$ | $H$ |
|---|---|---|---|---|---|---|---|---|---|
| Associated dual value | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |

The conclusion is that the critical path is given as $A \rightarrow D \rightarrow \text{Dummy} \rightarrow H$. Observe that *positive* dual values will always equal 1 because a delay of one day in any critical activity will increase the duration of the project by one day (remember that the dual variable is interpreted as the worth per unit of a resource, see Section 4.3.1).

### PROBLEM SET 6.6D

1. Use LP to determine the critical path for the project network in Figure 6.55.
2. Use LP to determine the critical path for the project networks in Figure 6.56.

## 6.6.5 PERT Networks

PERT differs from CPM in that it bases the duration of an activity on three estimates:

1. **Optimistic time**, $a$, which assumes that execution goes extremely well.
2. **Most likely time**, $m$, which assumes that execution is done under normal conditions.
3. **Pessimistic time**, $b$, which assumes that execution goes extremely poorly.

The range $(a, b)$ is assumed to enclose all possible estimates of the duration of an activity. The estimate $m$ thus must lie somewhere in the range $(a, b)$. Based on the estimates, the average duration time, $\overline{D}$, and variance, $v$, are computed as follows:

$$\overline{D} = \frac{a + 4m + b}{6}$$

$$v = \left(\frac{b - a}{6}\right)^2$$

CPM calculations given in Sections 6.6.2 and 6.6.3 may be applied directly, with $\overline{D}$ replacing the single estimate $D$.

It is now possible to estimate the probability that a node $j$ in the network will occur by a prespecified scheduled time, $S_j$. Let $e_j$ be the earliest occurrence time of node $j$. Because the durations of the activities leading from the start node to node $j$ are random variables, $e_j$ also must be a random variable. Assuming that all the activities in the network are statistically independent, we can determine the mean, $E\{e_j\}$, and variance, var $\{e_j\}$, in the following manner. If there is only one path from the start node to node $j$, then the mean is the sum of expected durations, $\overline{D}$, for all the activities along this path and the variance is the sum of the variances, $v$, of the same activities. On the other hand, if more than one path leads to node $j$, then it is necessary first to compute the statistical distribution of the duration of the longest path before the correct mean and variance can be calculated. This problem is rather difficult because it is equivalent to determining the distribution of the maximum of several random variables. A simplifying assumption thus calls for computing the mean and variance, $E\{e_j\}$ and $\text{var}\{e_j\}$, as those of the path to node $j$ that has the largest sum of *expected* activity durations. If two or more paths have the same mean, the one with the largest variance is selected because it reflects the most uncertainty, hence leads to a more conservative estimate of probabilities.

Once the mean and variance of the path to node $j$, $E\{e_j\}$ and $\text{var}\{e_j\}$, have been computed, the probability that node $j$ will be realized by a preset time $S_j$ is calculated using the following formula:

$$P\{e_j \le S_j\} = P\left\{\frac{e_j - E\{e_j\}}{\sqrt{\text{var}\{e_j\}}} \le \frac{S_j - E\{e_j\}}{\sqrt{\text{var}\{e_j\}}}\right\} = P\{z \le K_j\}$$

where

$$z = \text{Standard normal random variable}$$

$$K_j = \frac{S_j - E\{e_j\}}{\sqrt{\text{var}\{e_j\}}}$$

The standard normal variable $z$ has mean 0 and standard deviation 1 (see Appendix C). Justification for the use of the normal distribution is that $e_j$ is the sum of independent random variables. According to the *Central Limit Theorem* (see Section 12.5.4), $e_j$ is approximately normally distributed.

## Example 6.6-6

Consider the project of Example 6.6-2. To avoid repeating critical path calculations, the values of $a$, $m$, and $b$ in the table below are selected such that $\overline{D}_{ij} = D_{ij}$ for all $i$ and $j$ in Example 6.6-2.

| Activity | $i$-$j$ | $(a, m, b)$ | Activity | $i$-$j$ | $(a, m, b)$ |
|---|---|---|---|---|---|
| $A$ | 1-2 | (3, 5, 7) | $E$ | 3-5 | (1, 2, 3) |
| $B$ | 1-3 | (4, 6, 8) | $F$ | 3-6 | (9, 11, 13) |
| $C$ | 2-3 | (1, 3, 5) | $G$ | 4-6 | (1, 1, 1) |
| $D$ | 2-4 | (5, 8, 11) | $H$ | 5-6 | (10, 12, 14) |

The mean $\overline{D}_{ij}$ and variance $V_{ij}$ for the different activities are given in the following table. Note that for a dummy activity $(a, b, m) = (0, 0, 0)$, hence its mean and variance also equal zero.

| Activity | $i$-$j$ | $\overline{D}_{ij}$ | $V_{ij}$ | Activity | $i$-$j$ | $\overline{D}_{ij}$ | $V_{ij}$ |
|---|---|---|---|---|---|---|---|
| $A$ | 1-2 | 5 | .444 | $E$ | 3-5 | 2 | .111 |
| $B$ | 1-3 | 6 | .444 | $F$ | 3-6 | 11 | .444 |
| $C$ | 2-3 | 3 | .444 | $G$ | 4-6 | 1 | .000 |
| $D$ | 2-4 | 8 | 1.000 | $H$ | 5-6 | 12 | .444 |

The next table gives the longest path from node 1 to the different nodes, together with their associated mean and variance.

| Node | Longest path based on mean durations | Path mean | Path standard deviation |
|---|---|---|---|
| 2 | 1-2 | 5.00 | 0.67 |
| 3 | 1-2-3 | 8.00 | 0.94 |
| 4 | 1-2-4 | 13.00 | 1.20 |
| 5 | 1-2-4-5 | 13.00 | 1.20 |
| 6 | 1-2-4-5-6 | 25.00 | 1.37 |

Finally, the following table computes the probability that each node is realized by a preset time, $S_j$, specified by the analyst.

| Node $j$ | Longest path | Path mean | Path standard deviation | $S_j$ | $K_j$ | $P\{z \le K_j\}$ |
|---|---|---|---|---|---|---|
| 2 | 1-2 | 5.00 | 0.67 | 5.00 | 0 | .5000 |
| 3 | 1-2-3 | 8.00 | 0.94 | 11.00 | 3.19 | .9993 |
| 4 | 1-2-4 | 13.00 | 1.20 | 12.00 | −.83 | .2033 |
| 5 | 1-2-4-5 | 13.00 | 1.20 | 14.00 | .83 | .7967 |
| 6 | 1-2-4-5-6 | 25.00 | 1.37 | 26.00 | .73 | .7673 |

TORA provides a module for carrying out PERT calculations. To use this module, select `Project Planning` ⇒ `PERT-Program Evaluation and Review Technique` from `Main Menu`. In the output screen, you have the option to select `Activity Mean/Var` to compute the mean and variance for each activity or `PERT Calculations` to compute the mean and variance of the longest path to each node in the network.

Figure 6.61 shows TORA output for the PERT calculations of Example 6.6-6 (file ch6ToraPERTEx6-6-6.txt).

## PROBLEM SET 6.6E

**1.** Consider Problem 2, Set 6.6b. The estimates $(a, m, b)$ are listed below. Determine the probabilities that the different nodes of the project will be realized without delay.

| Project (a) | | | | Project (b) | | | |
|---|---|---|---|---|---|---|---|
| *Activity* | $(a, m, b)$ | *Activity* | $(a, m, b)$ | *Activity* | $(a, m, b)$ | *Activity* | $(a, m, b)$ |
| 1-2 | (5, 6, 8) | 3-6 | (3, 4, 5) | 1-2 | (1, 3, 4) | 3-7 | (12, 13, 14) |
| 1-4 | (1, 3, 4) | 4-6 | (4, 8, 10) | 1-3 | (5, 7, 8) | 4-5 | (10, 12, 15) |
| 1-5 | (2, 4, 5) | 4-7 | (5, 6, 8) | 1-4 | (6, 7, 9) | 4-7 | (8, 10, 12) |
| 2-3 | (4, 5, 6) | 5-6 | (9, 10, 15) | 1-6 | (1, 2, 3) | 5-6 | (7, 8, 11) |
| 2-5 | (7, 8, 10) | 5-7 | (4, 6, 8) | 2-3 | (3, 4, 5) | 5-7 | (2, 4, 8) |
| 2-6 | (8, 9, 13) | 6-7 | (3, 4, 5) | 2-5 | (7, 8, 9) | 6-7 | (5, 6, 7) |
| 3-4 | (5, 9, 19) | | | 3-4 | (10, 15, 20) | | |

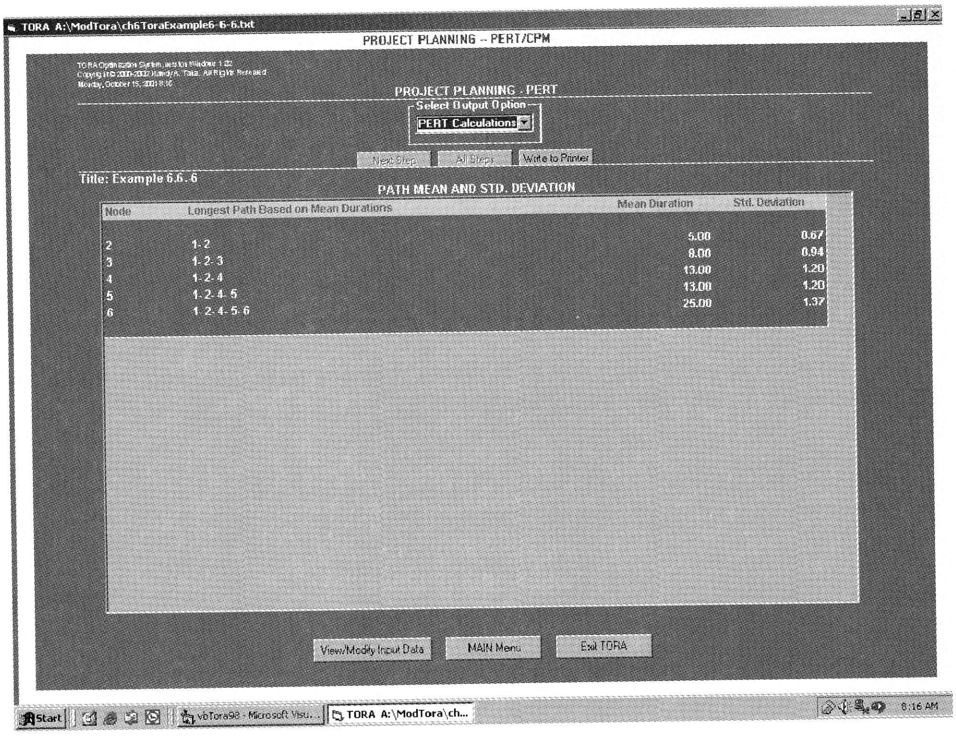



FIGURE 6.61
TORA PERT calculations for Example 6.6-6

## SELECTED REFERENCES


Ahuja, R., T. Magnati, and J. Orlin, *Network Flows: Theory, Algorithms, and Applications,* Prentice Hall, Upper Saddle River, NJ, 1993.

Bazaraa, M., J. Jarvis, and H. Sherali, *Linear Programming and Network Flow,* 2nd ed., Wiley, New York, 1990.

Evans, J. R., and E. Minieka, *Optimization Algorithms for Networks and Graphs,* 2nd ed., Marcel Dekker, New York, 1992.

Murty, K., *Network Programming,* Prentice Hall, Upper Saddle River, NJ, 1992.


## COMPREHENSIVE PROBLEMS

**6.1** An outdoors person who lives in San Francisco (SF) wishes to spend a 15-day vacation visiting four national parks: Yosemite (YO), Yellowstone (YE), Grand Teton (GT), and Mount Rushmore (MR). The tour, which starts and ends in San Francisco, visits the parks in the following order and includes a 2-day stay at each park: SF → YO → YE → GT → MR → SF. Travel from one park location to another is either by air or car. Each leg of the trip takes 1/2 day if traveled by air. Travel by car takes 1/2 day from SF to YO, 3 days from YO to YE, one day from YE to GT, 2 days from GT to MR, and 3 days from MR back to SF. The trade-off is that car travel generally costs less but takes longer. Considering the fact that the individual must return to work in 15 days, the objective is to make the tour as inexpensive as possible within the 15-day limit. The following table provides the one-way cost of traveling by car and air. Determine the mode of travel on each leg of the tour.

| | Air travel cost ($) to | | | | | Car travel cost ($) to | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| *From* | *SF* | *YO* | *YE* | *GT* | *MR* | *SF* | *YO* | *YE* | *GT* | *MR* |
| SF | — | 150 | 350 | 380 | 450 | — | 130 | 175 | 200 | 230 |
| YO | 150 | — | 400 | 290 | 340 | 130 | — | 200 | 145 | 180 |
| YE | 350 | 400 | — | 150 | 320 | 175 | 200 | — | 70 | 150 |
| GT | 380 | 290 | 150 | — | 300 | 200 | 145 | 70 | — | 100 |
| MR | 450 | 340 | 320 | 300 | — | 230 | 180 | 150 | 100 | — |

**6.2**[6] A benefactor has donated valuable books to the Springdale Public Library. The books come in four heights: 12, 10, 8, and 6 inches. The head librarian estimates that 12 feet of shelving will be needed for the 12-inch books, 18 feet for the 10-inch ones, 9 feet for the 8-inch books, and 10 feet for the 6-inch ones. The construction cost of a shelf includes both a fixed cost and a variable cost per foot length as the following table shows.

| Shelf height (in) | Fixed cost ($) | Variable cost ($/ft length) |
|---|---|---|
| 12 | 25 | 5.50 |
| 10 | 25 | 4.50 |
| 8 | 22 | 3.50 |
| 6 | 22 | 2.50 |

Given that smaller books can be stored on larger shelves, how should the shelves be designed?

**6.3** A shipping company wants to deliver five cargo shipments from ports *A, B,* and *C* to ports *D* and *E*. The delivery dates for the five shipments are

| Shipment | Shipping route | Delivery date |
|---|---|---|
| 1 | *A* to *D* | 10 |
| 2 | *A* to *E* | 15 |
| 3 | *B* to *D* | 4 |
| 4 | *B* to *E* | 5 |
| 5 | *C* to *E* | 18 |

The following table gives trip times (in days) between ports (the return trip is assumed to take less time).

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | | | | 3 | 4 |
| B | | | | 3 | 2 |
| C | | | | 3 | 5 |
| D | 2 | 2 | 2 | | |
| E | 3 | 1 | 4 | | |

[6]Based on A. Ravindran, "On Compact Storage in Libraries," *Opsearch,* Vol. 8, No. 3, pp. 245–52, 1971.

The company wants to determine the minimum number of ships needed to carry out the given shipping schedule.

**6.4**[7] Several individuals have set up separate brokerage firms that traded in highly speculative stocks. The brokers operated under a loose financial system that allowed extensive inter-brokerage transactions, including buying, selling, borrowing, and lending. For the group of brokers as a whole, the main source of income was the commission they received from sales to outside clients.

Eventually, the risky trading in speculative stocks became unmanageable, and all the brokers declared bankruptcy. At the time the bankruptcy was declared, the financial situation was that all brokers owed money to outside clients and the interbroker financial entanglements were so complex that almost every broker owed money to every other broker in the group.

The brokers whose assets could pay for their debts were declared solvent. The remaining brokers were referred to a legal body whose purpose was to resolve the debt situation in the best interest of outside clients. Because the assets and receivables of the nonsolvent brokers were less than their payables, all debts were prorated. The final effect was a complete liquidation of all the assets of the nonsolvent brokers.

In resolving the financial entanglements within the group of nonsolvent brokers, it was decided that the transactions would be executed only to satisfy certain legal requirements because, in effect, none of the brokers would be keeping any of the funds owed by others. As such, the legal body requested that the number of interbroker transactions be reduced to an absolute minimum. This means that if $A$ owed $B$ an amount $X$, and $B$ owed $A$ an amount $Y$, the two "loop" transactions were reduced to one whose amount is $|X - Y|$. This amount would go from $A$ to $B$ if $X > Y$ and from $B$ to $A$ if $Y > X$. If $X = Y$, the transactions were completely eliminated. The idea was to be extended to all loop transactions involving any number of brokers.

How would you handle this situation? Specifically, you are required to answer two questions.

1. How should the debts be prorated?
2. How should the number of interbroker transactions be reduced to a minimum?

---

[7]Based on H. Taha, "Operations Research Analysis of a Stock Market Problem," *Computers and Operations Research*, Vol. 18, No. 7, pp. 597–602, 1991.

C H A P T E R 7

# Advanced Linear Programming

This chapter presents a matrix version of linear programming that allows the development of a number of computationally efficient algorithms: revised simplex method, upper and lower bounding, decomposition, and parametric programming. The chapter also presents the totally different Karmarkar interior-point algorithm, which appears quite efficient in handling very large LPs.

## 7.1 SIMPLEX METHOD FUNDAMENTALS

In linear programming, the feasible solution space is said to form a **convex set** if the line segment joining any two *distinct* feasible points also falls in the set. An **extreme point** of the convex set is a feasible point that cannot lie on a line segment joining any two *distinct* feasible points in the set. Actually, extreme points are the same as corner points, the more apt name used in Chapters 2, 3, and 4.

Figure 7.1 illustrates two convex sets. Set (a), which is typical of the solution space of a linear program, is convex (with six extreme points), whereas set (b) is nonconvex.

In the graphical LP solution given in Section 2.3, we demonstrated that the optimum solution can always be associated with a feasible extreme (corner) point of the solution space. This result makes sense intuitively because in LP every feasible point

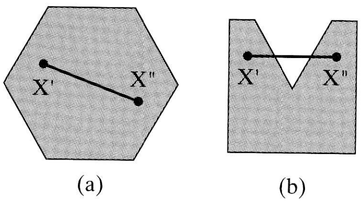



FIGURE 7.1
Examples of a convex and a nonconvex set

can be determined as a function of the extreme points. For example, in convex set (a) of Figure 7.1, given the extreme points $\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_4, \mathbf{X}_5$, and $\mathbf{X}_6$, a feasible point $\mathbf{X}$ can be expressed as a **convex combination** of the extreme points using

$$\mathbf{X} = \alpha_1\mathbf{X}_1 + \alpha_2\mathbf{X}_2 + \alpha_3\mathbf{X}_3 + \alpha_4\mathbf{X}_4 + \alpha_5\mathbf{X}_5 + \alpha_6\mathbf{X}_6$$

where

$$\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 + \alpha_6 = 1$$

$$\alpha_i \geq 0, i = 1,2, \ldots, 6$$

This observation shows that extreme points provide all that is needed to define the solution space completely.

---

**Example 7.1-1**

Show that the following set is convex:

$$C = \{(x_1,x_2)|x_1 \leq 2, x_2 \leq 3, x_1 \geq 0, x_2 \geq 0\}$$

Let $\mathbf{X}_1 = \{x_1^1, x_2^1\}$ and $\mathbf{X}_2 = \{x_1^2, x_2^2\}$ be any two distinct points in $C$. If $C$ is convex, then $\mathbf{X} = (x_1,x_2) = \alpha_1\mathbf{X}_1 + \alpha_2\mathbf{X}_2$ must also be in $C$. To show that this is true, we need to show that all the constraints of $C$ are satisfied by the line segment $\mathbf{X}$—that is,

$$x_1 = \alpha_1 x_1^1 + \alpha_2 x_1^2 \leq \alpha_1(2) + \alpha_2(2) = 2$$

$$x_2 = \alpha_1 x_2^1 + \alpha_2 x_2^2 \leq \alpha_1(3) + \alpha_2(3) = 3$$

Thus, $x_1 \leq 2$ and $x_2 \leq 3$ because $\alpha_1 + \alpha_2 = 1$. Additionally, the nonnegativity conditions are satisfied because $\alpha_1$ and $\alpha_2$ are nonnegative.

---

## PROBLEM SET 7.1A

1. Show that the set $Q = \{x_1,x_2 | x_1 + x_2 \leq 1, x_1 \geq 0, x_2 \geq 0\}$ is convex. Is the nonnegativity condition essential for the proof?
2. Show that the set $Q = \{x_1,x_2 \mid x_1 \geq 1 \text{ or } x_2 \geq 2\}$ is not convex.
3. Determine graphically the extreme points of the following convex set:

   $$Q = \{x_1,x_2 \mid x_1 + x_2 \leq 2, x_1 \geq 0, x_2 \geq 0\}$$

   Show that the entire feasible solution space can be determined as a convex combination of its extreme points. Hence conclude that any convex (bounded) solution space is totally defined once its extreme points are known.
4. In the solution space in Figure 7.2 (drawn to scale), express the interior point (3, 1) as a convex combination of the extreme points $A$, $B$, $C$, and $D$ where each extreme point carries a strictly positive weight.

### 7.1.1 From Extreme Points to Basic Solutions

It is convenient to express the general LP problem in equation form (see Section 3.1) using matrix notation. Define $\mathbf{X}$ as an $n$-vector representing the variables, $\mathbf{A}$ as an

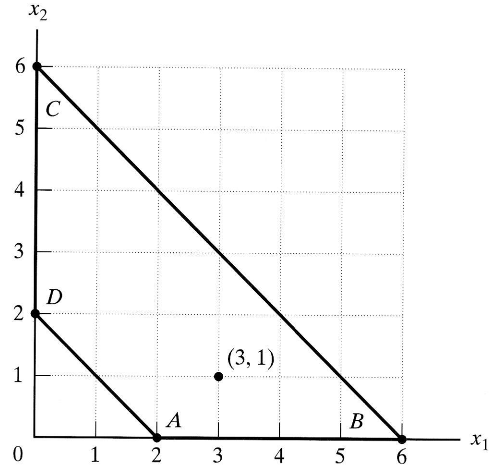



FIGURE 7.2
Solution space for Problem 4, Set 7.1a

($m \times n$)-matrix representing the constraint coefficients, and **C** as an $n$-vector representing the objective function coefficients. The LP is then written as

$$\text{Maximize or minimize } z = \mathbf{CX}$$

subject to

$$\mathbf{AX} = \mathbf{b}$$

$$\mathbf{X} \geq \mathbf{0}$$

Using the format of Chapter 3 (see also Figure 4.1), the rightmost $m$ columns of **A** always can be made to represent the identity matrix **I** through proper arrangements of the slack/artificial variables associated with the starting basic solution.

A **basic solution** of $\mathbf{AX} = \mathbf{b}$ is determined by setting $n - m$ variables equal to zero, and then solving the resulting $m$ equations in the remaining $m$ unknowns, provided that the resulting solution is unique. Given this definition, the theory of linear programming establishes the following result between the geometric definition of extreme points and the algebraic definition of basic solutions:

$$\text{Extreme points of } \{\mathbf{X} \mid \mathbf{AX} = \mathbf{b}\} \Leftrightarrow \text{Basic solutions of } \mathbf{AX} = \mathbf{b}$$

The relationship means that the extreme points of the LP solution space are totally defined by the basic solutions of the system $\mathbf{AX} = \mathbf{b}$, and vice versa. Thus, we conclude that the basic solutions of $\mathbf{AX} = \mathbf{b}$ contain all the information we need to determine the optimum solution of the LP problem. Furthermore, if we impose the nonnegativity restriction, $\mathbf{X} \geq \mathbf{0}$, the search for the optimum solution is confined to the *feasible* basic solutions only.

To formalize the definition of a basic solution, the system $\mathbf{AX} = \mathbf{b}$ can be expressed in vector form as follows:

$$\sum_{j=1}^{n} \mathbf{P}_j x_j = \mathbf{b}$$

The vector $\mathbf{P}_j$ is the $j$th column of $\mathbf{A}$. A subset of $m$ vectors is said to form a **basis**, $\mathbf{B}$, if, and only if, the selected $m$ vectors are **linearly independent**. In this case, the matrix $\mathbf{B}$ is **nonsingular**. If $\mathbf{X}_B$ is the set of $m$ variables associated with the vectors of nonsingular $\mathbf{B}$, then $\mathbf{X}_B$ must be a **basic solution**. In this case, we have

$$\mathbf{B}\mathbf{X}_B = \mathbf{b}$$

Given the inverse $\mathbf{B}^{-1}$ of $\mathbf{B}$, we then get the corresponding basic solution as

$$\mathbf{X}_B = \mathbf{B}^{-1}\mathbf{b}$$

If $\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$, then $\mathbf{X}_B$ is feasible. The definition, of course, assumes that the remaining $n - m$ variables are **nonbasic** at zero level.

The previous result shows that in a system of $m$ equations and $n$ unknowns, the maximum number of (feasible and infeasible) basic solutions is given by

$$C_m^n = \frac{n!}{m!(n-m)!}$$

## Example 7.1-2

Determine and classify (as feasible and infeasible) all the basic solutions of the following system of equations.

$$\begin{pmatrix} 1 & 3 & -1 \\ 2 & -2 & -2 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix}$$

The following table summarizes the results. The inverse of $\mathbf{B}$ is determined by using one of the methods in Section A.2.7.

| $\mathbf{B}$ | $\mathbf{BX}_B = b$ | Solution | Status |
|---|---|---|---|
| $(\mathbf{P}_1, \mathbf{P}_2)$ | $\begin{pmatrix} 1 & 3 \\ 2 & -2 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix}$ | $\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{4} & \frac{3}{8} \\ \frac{1}{4} & -\frac{1}{8} \end{pmatrix}\begin{pmatrix} 4 \\ 2 \end{pmatrix} = \begin{pmatrix} \frac{7}{4} \\ \frac{3}{4} \end{pmatrix}$ | Feasible |
| $(\mathbf{P}_1, \mathbf{P}_3)$ | (Not a basis) | — | — |
| $(\mathbf{P}_2, \mathbf{P}_3)$ | $\begin{pmatrix} 3 & -1 \\ -2 & -2 \end{pmatrix}\begin{pmatrix} x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix}$ | $\begin{pmatrix} x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \frac{1}{4} & -\frac{1}{8} \\ -\frac{1}{4} & -\frac{3}{8} \end{pmatrix}\begin{pmatrix} 4 \\ 2 \end{pmatrix} = \begin{pmatrix} \frac{3}{4} \\ -\frac{7}{4} \end{pmatrix}$ | Infeasible |

We can also investigate the problem by expressing it in vector form as follows:

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix}x_1 + \begin{pmatrix} 3 \\ -2 \end{pmatrix}x_2 + \begin{pmatrix} -1 \\ -2 \end{pmatrix}x_3 = \begin{pmatrix} 4 \\ 2 \end{pmatrix}$$

Each of $\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3$, and $\mathbf{b}$ is a two-dimensional vector, which can be represented generically as $(a_1, a_2)^T$. Figure 7.3 graphs these vectors on the $(a_1, a_2)$-plane. For example, for $\mathbf{b} = (4, 2)^T, a_1 = 4$ and $a_2 = 2$.

Because we are dealing with two equations ($m = 2$), a basis must include exactly two vectors, selected from among $\mathbf{P}_1, \mathbf{P}_2$, and $\mathbf{P}_3$. From Figure 7.3, the combinations $(\mathbf{P}_1, \mathbf{P}_2)$ and $(\mathbf{P}_2, \mathbf{P}_3)$ form bases because their associated vectors are independent. In the combination $(\mathbf{P}_1, \mathbf{P}_3)$ the two vectors are dependent, and hence do not constitute a basis.

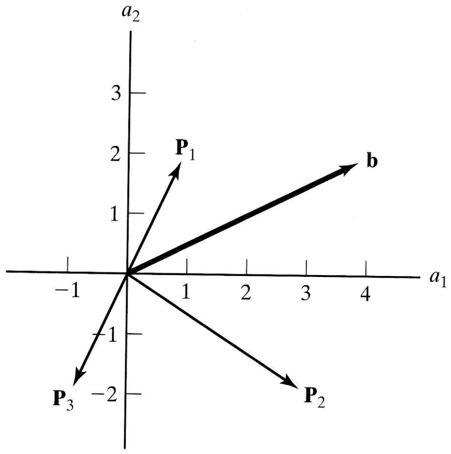



FIGURE 7.3
Vector representation of LP solution space

Algebraically, a combination forms a basis if its determinant is not zero (see Section A.2.5). The following computations show that the combinations $(\mathbf{P}_1, \mathbf{P}_2)$ and $(\mathbf{P}_2, \mathbf{P}_3)$ are bases, and the combination $(\mathbf{P}_1, \mathbf{P}_3)$ is not.

$$\det(\mathbf{P}_1, \mathbf{P}_2) = \det\begin{pmatrix} 1 & 3 \\ 2 & -2 \end{pmatrix} = (1 \times -2) - (2 \times 3) = -8 \neq 0$$

$$\det(\mathbf{P}_2, \mathbf{P}_3) = \det\begin{pmatrix} 3 & -1 \\ -2 & -2 \end{pmatrix} = (3 \times -2) - (-2 \times -1) = -8 \neq 0$$

$$\det(\mathbf{P}_1, \mathbf{P}_3) = \det\begin{pmatrix} 1 & -1 \\ 2 & -2 \end{pmatrix} = (1 \times -2) - (2 \times -1) = 0$$

We can take advantage of the vector representation of the problem to discuss how the starting solution of the simplex method is determined. From the vector representation in Figure 7.3, the basis $\mathbf{B} = (\mathbf{P}_1, \mathbf{P}_2)$ can be used to start the simplex iterations because it produces the basic feasible solution $\mathbf{X}_B = (x_1, x_2)^T$. However, in the absence of the vector representation, the only course of action available to us is to try all possible bases (3 in this example, as shown above). The difficulty with trial and error is that it is not suitable for automatic computations. In a typical LP with thousands of variables and constraints where the use of the computer is a must, trial and error simply is not a practical option because of the tremendous computational overhead. To alleviate this problem, the simplex method always uses an identity matrix, $\mathbf{B} = \mathbf{I}$, to start the iterations. Why does a starting $\mathbf{B} = \mathbf{I}$ offer an advantage? The answer is that it will always provide a *feasible* starting basic solution (provided that the right-hand side vector of the equations is nonnegative). You can see this result in Figure 7.3 by graphing the vectors of $\mathbf{B} = \mathbf{I}$ and noting that they coincide with the horizontal and vertical axes, thus always guaranteeing a starting basic feasible solution.

The basis $\mathbf{B} = \mathbf{I}$ automatically forms part of the LP equations if all the original constraints are $\leq$. In other cases, we simply add the unit vectors where needed. This is what the artificial variables accomplish (Section 3.4). We then penalize these extraneous variables in the objective function to force them to zero level in the final solution.

## PROBLEM SET 7.1B

1. In the following sets of equations, (a) and (b) have unique (basic) solutions, (c) has infinity of solutions, and (d) has no solution. Show how these results can be verified using graphical vector representation. From this exercise, state the general conditions for vector dependence-independence that lead to unique solution, infinity of solutions, and no solution.

   **(a)** $x_1 + 3x_2 = 2$, $3x_1 + x_2 = 3$

   **(b)** $2x_1 + 3x_2 = 1$, $2x_1 - x_2 = 2$

   **(c)** $2x_1 + 6x_2 = 4$, $x_1 + 3x_2 = 2$

   **(d)** $2x_1 - 4x_2 = 2$, $-x_1 + 2x_2 = 1$

2. Determine graphically (using vectors) if each of the sets of equations below has a unique solution, infinity of solutions, or no solution. For the cases of unique solutions, indicate from the vector representation (and without solving the equations algebraically) whether the values of the $x_1$ and $x_2$ are positive, zero, or negative.

   **(a)** $\begin{pmatrix} 5 & 4 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ **(b)** $\begin{pmatrix} 2 & -2 \\ 1 & 3 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \end{pmatrix}$

   **(c)** $\begin{pmatrix} 2 & 4 \\ 1 & 3 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -2 \\ -1 \end{pmatrix}$ **(d)** $\begin{pmatrix} 2 & 4 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 6 \\ 3 \end{pmatrix}$

   **(e)** $\begin{pmatrix} -2 & 4 \\ 1 & -2 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$ **(f)** $\begin{pmatrix} 1 & -2 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

3. Consider the following system of equations:

$$\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} x_1 + \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix} x_2 + \begin{pmatrix} 1 \\ 4 \\ 2 \end{pmatrix} x_3 + \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} x_4 = \begin{pmatrix} 3 \\ 4 \\ 2 \end{pmatrix}$$

   Determine if any of the following combinations forms a basis.

   **(a)** $(\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3)$

   **(b)** $(\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_4)$

   **(c)** $(\mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4)$

   **(d)** $(\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4)$

4. **True or False?**

   **(a)** The system $\mathbf{BX} = \mathbf{b}$ has a unique solution if $\mathbf{B}$ is nonsingular.

   **(b)** The system $\mathbf{BX} = \mathbf{b}$ has no solution if $\mathbf{B}$ is singular and $\mathbf{b}$ is independent of $\mathbf{B}$.

   **(c)** The system $\mathbf{BX} = \mathbf{b}$ has infinity of solutions if $\mathbf{B}$ is singular and $\mathbf{b}$ is dependent.

### 7.1.2 Generalized Simplex Tableau in Matrix Form

In this section, we use matrices to develop the general simplex tableau. This representation will be the basis for subsequent developments in the chapter.

Consider the LP in equation form:

$$\text{Maximize } z = \mathbf{CX}, \text{ subject to } \mathbf{AX} = \mathbf{b}, \mathbf{X} \geq \mathbf{0}$$

The problem can be written equivalently as

$$\begin{pmatrix} 1 & -\mathbf{C} \\ \mathbf{0} & \mathbf{A} \end{pmatrix}\begin{pmatrix} z \\ \mathbf{X} \end{pmatrix} = \begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix}$$

Suppose that $\mathbf{B}$ is a feasible basis of the system $\mathbf{AX} = \mathbf{b}, \mathbf{X} \geq \mathbf{0}$, and let $\mathbf{X}_B$ be the corresponding set of basic variables with $\mathbf{C}_B$ as its associated objective vector. The corresponding solution may then be computed as follows (the method for inverting partitioned matrices is given in Section A.2.7):

$$\begin{pmatrix} z \\ \mathbf{X}_B \end{pmatrix} = \begin{pmatrix} 1 & -\mathbf{C}_B \\ \mathbf{0} & \mathbf{B} \end{pmatrix}^{-1} \begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{C}_B\mathbf{B}^{-1} \\ \mathbf{0} & \mathbf{B}^{-1} \end{pmatrix} \begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix} = \begin{pmatrix} C_B\mathbf{B}^{-1}\mathbf{b} \\ \mathbf{B}^{-1}\mathbf{b} \end{pmatrix}$$

The general simplex tableau in matrix form can be derived from the original equations as follows:

$$\begin{pmatrix} 1 & \mathbf{C}_B\mathbf{B}^{-1} \\ \mathbf{0} & \mathbf{B}^{-1} \end{pmatrix} \begin{pmatrix} 1 & -\mathbf{C} \\ \mathbf{0} & \mathbf{A} \end{pmatrix} \begin{pmatrix} z \\ \mathbf{X} \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{C}_B\mathbf{B}^{-1} \\ \mathbf{0} & \mathbf{B}^{-1} \end{pmatrix} \begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix}$$

Matrix manipulations yield the following set of equations:

$$\begin{pmatrix} 1 & \mathbf{C}_B\mathbf{B}^{-1}\mathbf{A} - \mathbf{C} \\ \mathbf{0} & \mathbf{B}^{-1}\mathbf{A} \end{pmatrix} \begin{pmatrix} z \\ \mathbf{X} \end{pmatrix} = \begin{pmatrix} C_B\mathbf{B}^{-1}\mathbf{b} \\ \mathbf{B}^{-1}\mathbf{b} \end{pmatrix}$$

Given $\mathbf{P}_j$ is the $j$th vector of $\mathbf{A}$, the simplex tableau column associated with variable $x_j$ can be represented as follows:

| Basic | $x_j$ | Solution |
|---|---|---|
| $z$ | $\mathbf{C}_B\mathbf{B}^{-1}\mathbf{P}_j - c_j$ | $\mathbf{C}_B\mathbf{B}^{-1}\mathbf{b}$ |
| $\mathbf{X}_B$ | $\mathbf{B}^{-1}\mathbf{P}_j$ | $\mathbf{B}^{-1}\mathbf{b}$ |

In fact, the tableau above is the same as the one we presented in Chapter 3 (see Problem 5 of Set 7.1c). An important property of this table is that the inverse, $\mathbf{B}^{-1}$, is the only element that changes from one tableau to the next, and that the *entire* tableau can be generated once $\mathbf{B}^{-1}$ is known. This point is important because the computational roundoff error in any tableau can be controlled by controlling the accuracy of $\mathbf{B}^{-1}$. This result is the main reason for the development of the revised simplex method in Section 7.2.

### Example 7.1-3

Consider the following LP:

$$\text{Maximize } z = x_1 + 4x_2 + 7x_3 + 5x_4$$

subject to

$$2x_1 + x_2 + 2x_3 + 4x_4 = 10$$

$$3x_1 - x_2 - 2x_3 + 6x_4 = 5$$

$$x_1, x_2, x_3, x_4 \geq 0$$

Generate the simplex tableau associated with the basis $\mathbf{B} = (\mathbf{P}_1, \mathbf{P}_2)$.

Given $\mathbf{B} = (\mathbf{P}_1, \mathbf{P}_2)$, then $\mathbf{X}_B = (x_1, x_2)^T$ and $\mathbf{C}_B = (1, 4)$. Thus,

$$\mathbf{B}^{-1} = \begin{pmatrix} 2 & 1 \\ 3 & -1 \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{5} & \frac{1}{5} \\ \frac{3}{5} & -\frac{2}{5} \end{pmatrix}$$

We then get

$$\mathbf{X}_B = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \mathbf{B}^{-1}\mathbf{b} = \begin{pmatrix} \frac{1}{5} & \frac{1}{5} \\ \frac{3}{5} & -\frac{2}{5} \end{pmatrix}\begin{pmatrix} 10 \\ 5 \end{pmatrix} = \begin{pmatrix} 3 \\ 4 \end{pmatrix}$$

To compute the constraint columns in the body of the tableau, we have

$$\mathbf{B}^{-1}(\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4) = \begin{pmatrix} \frac{1}{5} & \frac{1}{5} \\ \frac{3}{5} & -\frac{2}{5} \end{pmatrix}\begin{pmatrix} 2 & 1 & 2 & 4 \\ 3 & -1 & -2 & 6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 2 & 0 \end{pmatrix}$$

Next, we compute the objective row as follows:

$$\mathbf{C}_B[\mathbf{B}^{-1}(\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_2, \mathbf{P}_4)] - \mathbf{C} = (1,4)\begin{pmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 2 & 0 \end{pmatrix} - (1,4,7,5) = (0,0,1,-3)$$

Finally, we compute the value of the objective function as follows:

$$z = \mathbf{C}_B\mathbf{B}^{-1}\mathbf{b} = \mathbf{C}_B\mathbf{X}_B = (1,4)\begin{pmatrix} 3 \\ 4 \end{pmatrix} = 19$$

Thus, the entire tableau can be summarized as shown below.

| Basic | $x_1$ | $x_2$ | $x_3$ | $x_4$ | Solution |
|---|---|---|---|---|---|
| $z$ | 0 | 0 | 1 | −3 | 19 |
| $x_1$ | 1 | 0 | 2 | 0 | 3 |
| $x_2$ | 0 | 1 | 0 | 2 | 4 |

The main conclusion from this example is that once the inverse, $\mathbf{B}^{-1}$, is known, the entire simplex tableau can be generated from $\mathbf{B}^{-1}$ and the *original* data of the problem.

## PROBLEM SET 7.1C

1. In Example 7.1-3, consider $\mathbf{B} = (\mathbf{P}_3, \mathbf{P}_4)$. Show that the corresponding basic solution is feasible, and then generate the corresponding simplex tableau.
2. Consider the following LP:

$$\text{Maximize } z = 5x_1 + 12x_2 + 4x_3$$

subject to

$$x_1 + 2x_2 + x_3 + x_4 = 10$$
$$2x_1 - 2x_2 - x_3 = 2$$
$$x_1, x_2, x_3, x_4 \geq 0$$

Check if each of the following vector sets forms a (feasible or infeasible) basis: $(\mathbf{P}_1, \mathbf{P}_2)$, $(\mathbf{P}_2, \mathbf{P}_3)$, $(\mathbf{P}_3, \mathbf{P}_4)$.

3. In the following LP, compute the entire simplex tableau associated with $\mathbf{X}_B = (x_1, x_2, x_5)^T$:

$$\text{Minimize } z = 2x_1 + x_2$$

7.2

subject to

$$
\begin{aligned}
3x_1 + x_2 - x_3 \qquad\qquad &= 3\\
4x_1 + 3x_2 \qquad - x_4 \qquad &= 6\\
x_1 + 2x_2 \qquad\qquad + x_5 &= 3\\
x_1, x_2, x_3, x_4, x_5 &\geq 0
\end{aligned}
$$

**4.** The following is an optimal LP tableau:

| Basic | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | Solution |
|---|---|---|---|---|---|---|
| $z$ | 0 | 0 | 0 | 3 | 2 | ? |
| $x_3$ | 0 | 0 | 1 | 1 | −1 | 2 |
| $x_2$ | 0 | 1 | 0 | 1 | 0 | 6 |
| $x_1$ | 1 | 0 | 0 | −1 | 1 | 2 |

The variables $x_3, x_4$, and $x_5$ are slacks in the original problem. Use matrix manipulations to reconstruct the original LP, and then compute the optimum objective value.

**5.** In the generalized simplex tableau, suppose that $\mathbf{X} = (\mathbf{X}_{\text{I}}, \mathbf{X}_{\text{II}})^T$, where $\mathbf{X}_{\text{II}}$ corresponds to a typical *starting* basic solution (consisting of slack and/or artificial variables) with $\mathbf{B} = \mathbf{I}$; and let $\mathbf{C} = (\mathbf{C}_{\text{I}}, \mathbf{C}_{\text{II}})$ and $\mathbf{A} = (\mathbf{D}, \mathbf{I})$ be the corresponding partitions of $\mathbf{C}$ and $\mathbf{A}$, respectively. Show that the matrix form of the simplex tableau reduces to the following form, which is exactly the form used in Chapter 3.

| Basic | $\mathbf{X}_{\text{I}}$ | $\mathbf{X}_{\text{II}}$ | Solution |
|---|---|---|---|
| $z$ | $\mathbf{C}_B\mathbf{B}^{-1}\mathbf{D} - \mathbf{C}_{\text{I}}$ | $\mathbf{C}_B\mathbf{B}^{-1} - \mathbf{C}_{\text{II}}$ | $\mathbf{C}_B\mathbf{B}^{-1}\mathbf{b}$ |
| $\mathbf{X}_B$ | $\mathbf{B}^{-1}\mathbf{D}$ | $\mathbf{B}^{-1}$ | $\mathbf{B}^{-1}\mathbf{b}$ |

## 7.2 REVISED SIMPLEX METHOD

Section 7.1.1 shows that the optimum solution of a linear program is always associated with a basic (feasible) solution. The simplex method search for the optimum starts by selecting a feasible basis, $\mathbf{B}$, and then moving to another feasible basis, $\mathbf{B}_{\text{next}}$, that leads to a better (or, at least, no worse) value of the objective function. Continuing in this manner, the optimum feasible basis is eventually reached.

The iterative steps of the *revised* simplex method are exactly the same as in the *tableau* simplex method presented in Chapter 3. The main difference is that the computations in the revised method are based on matrix algebra rather than on row operations. The use of matrix algebra reduces the adverse effect of machine roundoff error by controlling the accuracy of computing $\mathbf{B}^{-1}$. This result follows because, as Section 7.1 shows, the entire simplex tableau can be computed from the *original* data and the current $\mathbf{B}^{-1}$. In the tableau simplex method of Chapter 3, each tableau is generated from the immediately preceding one, which tends to worsen the problem of roundoff error.

### 7.2.1 Development of the Optimality and Feasibility Conditions

The general LP problem can be written as follows:

$$\text{Maximize or minimize } z = \sum_{j=1}^{n} c_j x_j \text{ subject to } \sum_{j=1}^{n} \mathbf{P}_j x_j = \mathbf{b}, x_j \geq 0, j = 1, 2, \ldots, n$$

For a given basic vector $\mathbf{X}_B$ and its corresponding basis $\mathbf{B}$ and objective vector $\mathbf{C}_B$, the general simplex tableau developed in Section 7.1.2 shows that any simplex iteration can be represented by the following equations:

$$z + \sum_{j=1}^{n} (z_j - c_j) x_j = \mathbf{C}_B \mathbf{B}^{-1} \mathbf{b}$$

$$(\mathbf{X}_B)_i + \sum_{j=1}^{n} (\mathbf{B}^{-1} \mathbf{P}_j)_i x_j = (\mathbf{B}^{-1} \mathbf{b})_i$$

where

$$z_j - c_j = \mathbf{C}_B \mathbf{B}^{-1} \mathbf{P}_j - c_j$$

The notation $(\mathbf{V})_i$ is used to represent the $i$th element of the vector $\mathbf{V}$.

**Optimality Condition.** From the $z$-equation given above, an increase in nonbasic $x_j$ above its current zero value will improve the value of $z$ relative to its current value, $\mathbf{C}_B \mathbf{B}^{-1} \mathbf{b}$, only if its $z_j - c_j$ is strictly negative in the case of maximization and strictly positive in the case of minimization. Otherwise, $x_j$ cannot improve the solution and must remain nonbasic at zero level. Though any nonbasic variable satisfying the given condition can be chosen to improve the solution, the simplex method uses a rule of thumb that selects the entering variable as the one with the *most* negative (*most* positive) $z_j - c_j$ in case of maximization (minimization).

**Feasibility Condition.** The determination of the leaving vector is based on examining the constraint equation associated with the $i$th *basic* variable. Specifically, we have

$$(\mathbf{X}_B)_i + \sum_{j=1}^{n} (\mathbf{B}^{-1} \mathbf{P}_j)_i x_j = (\mathbf{B}^{-1} \mathbf{b})_i$$

When the vector $\mathbf{P}_j$ is selected by the optimality condition to enter the basis, its associated variable $x_j$ will increase above zero level. At the same time, all the remaining nonbasic variables remain at zero level. Thus, the $i$th constraint equation reduces to

$$(\mathbf{X}_B)_i = (\mathbf{B}^{-1} \mathbf{b})_i - (\mathbf{B}^{-1} \mathbf{P}_j)_i x_j$$

The equation shows that if $(\mathbf{B}^{-1} \mathbf{P}_j)_i > 0$, an increase in $x_j$ can cause $(\mathbf{X}_B)_i$ to become negative, which violates the nonnegativity condition, $(\mathbf{X}_B)_i \geq 0$ for all i. Thus, we have

$$(\mathbf{B}^{-1} \mathbf{b})_i - (\mathbf{B}^{-1} \mathbf{P}_j)_i x_j \geq 0, \text{ for all } i$$

This condition yields the maximum value of the entering variable $x_j$ as

$$x_j = \min_i \left\{ \frac{(\mathbf{B}^{-1} \mathbf{b})_i}{(\mathbf{B}^{-1} \mathbf{P}_j)_i} \middle| (\mathbf{B}^{-1} \mathbf{P}_j)_i > 0 \right\}$$

The basic variable responsible for producing the minimum ratio leaves the basic solution to become nonbasic at zero level.

## PROBLEM SET 7.2A

**1.** Consider the following LP:

$$\text{Maximize } z = c_1x_1 + c_2x_2 + c_3x_3 + c_4x_4$$

subject to

$$\mathbf{P}_1x_1 + \mathbf{P}_2x_2 + \mathbf{P}_3x_3 + \mathbf{P}_4x_4 = \mathbf{b}$$

$$x_1, x_2, x_3, x_4 \geq 0$$

The vectors $\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3$, and $\mathbf{P}_4$ are shown in Figure 7.4. Assume that the basis $\mathbf{B}$ of the current iteration is comprised of $\mathbf{P}_1$ and $\mathbf{P}_2$.

**(a)** If the vector $\mathbf{P}_3$ enters the basis, which of the current two basic vectors must leave in order for the resulting basic solution to be feasible.

**(b)** Can the vector $\mathbf{P}_4$ be part of a feasible basis?

**2.** Prove that, in any simplex iteration, $z_j - c_j = 0$ for all the associated *basic* variables.

**3.** Prove that if $z_j - c_j > 0$ $(<0)$ for all the nonbasic variables $x_j$ of a maximization (minimization) LP problem, then the optimum is unique. Else, if $z_j - c_j$ equals zero for a nonbasic $x_j$, then the problem has an alternative optimum solution.

**4.** In an all-slack starting basic solution, show using the matrix form of the tableau that the mechanical procedure used in Section 3.3 in which the objective equation is set as

$$z - \sum_{j=1}^{n} c_jx_j = 0$$

automatically computes the proper $z_j - c_j$ for all the variables in the starting tableau.

**5.** Using the matrix form of the simplex tableau, show that in an all-artificial starting basic solution, the procedure employed in Section 3.4.1 that calls for substituting out the artificial variables in the objective function (using the constraint equations) actually computes the proper $z_j - c_j$ for all the variables in the starting tableau.

**6.** Consider an LP in which the variable $x_k$ is unrestricted in sign. Prove that by substituting $x_k = x_k^+ - x_k^-$, where $x_k^+$ and $x_k^-$ are nonnegative, it is impossible that $x_k^+$ and $x_k^-$ will replace one another in an alternative optimum solution.

**7.** Given the general LP in equation form with $m$ equations and $n$ unknowns, determine the maximum number of *adjacent* extreme points that can be reached from a nondegenerate extreme point of the solution space.

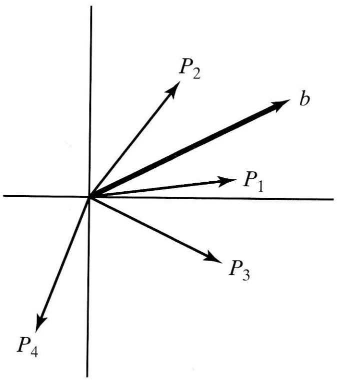



FIGURE 7.4
Vector representation of Problem 1, Set 7.2a

8. In applying the feasibility condition of the simplex method, suppose that $x_r = 0$ is a basic variable and that $x_j$ is the entering variable. Why is it necessary for the leaving variable $x_r$ to have $(\mathbf{B}^{-1}\mathbf{P}_j)_r > 0$? What is the fallacy if $(\mathbf{B}^{-1}\mathbf{P}_j)_r \leq 0$? (*Hint:* Basic $x_r$ must remain nonnegative.)
9. In the implementation of the feasibility condition of the simplex method, what are the conditions for encountering a degenerate solution for the first time? For continuing to obtain a degenerate solution in the next iteration? For removing degeneracy in the next iteration? Explain the answer mathematically.
10. What are the relationships between extreme points and basic solutions under degeneracy and nondegeneracy. What is the maximum number of iterations that can be performed at a given extreme point assuming no cycling?
11. Consider the LP

$$\text{Maximize } z = \mathbf{CX} \text{ subject to } \mathbf{AX} \leq \mathbf{b}, \mathbf{X} \geq \mathbf{0}, \mathbf{b} \geq \mathbf{0}$$

Suppose that the entering vector $\mathbf{P}_j$ is such that at least one element of $\mathbf{B}^{-1}\mathbf{P}_j$ is positive.

**(a)** If $\mathbf{P}_j$ is replaced with $\alpha\mathbf{P}_j$, where $\alpha$ is a positive scalar, and provided $x_j$ remains the entering variable, find the relationship between the values of $x_j$ corresponding to $\mathbf{P}_j$ and $\alpha\mathbf{P}_j$.

**(b)** Answer Part (a) if, additionally, $\mathbf{b}$ is replaced with $\beta\mathbf{b}$, where $\beta$ is a positive scalar.

12. Consider the LP

$$\text{Maximize } z = \mathbf{CX} \text{ subject to } \mathbf{AX} \leq \mathbf{b}, \mathbf{X} \geq \mathbf{0}, \mathbf{b} \geq \mathbf{0}$$

After obtaining the optimum solution, it is suggested that a nonbasic variable $x_j$ can be made basic (profitable) by reducing the requirements per unit of $x_j$ for the different resources to $\frac{1}{\alpha}$ of their original values, $\alpha > 1$. Because the requirements per unit are reduced, it is expected that the profit per unit of $x_j$ will also be reduced to $\frac{1}{\alpha}$ of its original value. Will these changes make $x_j$ a profitable variable? Explain.

13. Consider the LP

$$\text{Maximize } z = \mathbf{CX} \text{ subject to } (\mathbf{A}, \mathbf{I})\mathbf{X} = \mathbf{b}, \mathbf{X} \geq \mathbf{0}$$

Define $\mathbf{X}_B$ as the current basic vector with $\mathbf{B}$ as its associated basis and $\mathbf{C}_B$ as its vector of objective coefficients. Show that if $\mathbf{C}_B$ is replaced with the new coefficients $\mathbf{D}_B$, the values of $z_j - c_j$ for the basic vector $\mathbf{X}_B$ will remain equal to zero. What is the significance of this result?

## 7.2.2 Revised Simplex Algorithm

Having developed the optimality and feasibility conditions in Section 7.2.1, we now present the computational steps of the revised simplex method.

**Step 0.** Construct a starting basic feasible solution and let $\mathbf{B}$ and $\mathbf{C}_B$ be its associated basis and objective coefficients vector, respectively.

**Step 1.** Compute the inverse $\mathbf{B}^{-1}$ by using an appropriate inversion method.[1]

[1]In most LP presentations, including the first six editions of this book, the *product form* method for inverting a basis (see Section A.2.7) is integrated into the revised simplex algorithm because the *product form* lends itself neatly to the revised computations where successive bases differ in exactly one column. The author has removed this detail from this presentation because it makes the algorithm appear more complex than it really is. Moreover, the *product form* is rarely used in the development of LP codes because it is not designed for automatic computations where machine roundoff error can be a serious issue. Normally, some advanced numeric analysis method, such as the *LU decomposition* method, is used to obtain the inverse. Incidentally, TORA matrix inversion module is based on LU decomposition.

**Step 2.** For each *nonbasic* variable $x_j$, compute

$$z_j - c_j = \mathbf{C}_B\mathbf{B}^{-1}\mathbf{P}_j - c_j$$

If $z_j - c_j \geq 0$ in maximization ($\leq 0$ in minimization) for all nonbasic $x_j$, stop; the optimal solution is given by

$$\mathbf{X}_B = \mathbf{B}^{-1}\mathbf{b},\ z = \mathbf{C}_B\mathbf{X}_B$$

Else, apply the optimality condition and determine the *entering* variable $x_j$ as the nonbasic variable with the most negative (positive) $z_j - c_j$ in case of maximization (minimization).

**Step 3.** Compute $\mathbf{B}^{-1}\mathbf{P}_j$. If all the elements of $\mathbf{B}^{-1}\mathbf{P}_j$ are negative or zero, stop; the problem has no bounded solution. Else, compute $\mathbf{B}^{-1}\mathbf{b}$. Then for all the *strictly positive* elements of $\mathbf{B}^{-1}\mathbf{P}_j$, determine the ratios defined by the feasibility condition. The basic variable $x_i$ associated with the smallest ratio is the *leaving* variable.

**Step 4.** From the current basis $\mathbf{B}$, form a new basis by replacing the leaving vector $\mathbf{P}_i$ with the entering vector $\mathbf{P}_j$. Go to step 1 to start a new iteration.

## Example 7.2-1

The Reddy Mikks model (Section 2.1) is solved by the revised simplex algorithm. The same model was solved by the tableau method in Section 3.3.2. A comparison between the two methods will show that they are one and the same.

The equation form of the Reddy Mikks model can be expressed in matrix form as

$$\text{Maximize } z = (5,4,0,0,0,0)(x_1,x_2,x_3,x_4,x_5,x_6)^T$$

subject to

$$\begin{pmatrix} 6 & 4 & 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 & 0 \\ -1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{pmatrix} = \begin{pmatrix} 24 \\ 6 \\ 1 \\ 2 \end{pmatrix}$$

$$x_1, x_2, \ldots, x_6 \geq 0$$

We use the notation $\mathbf{C} = (c_1, c_2, \ldots, c_6)$ to represent the objective function coefficients and $(\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_6)$ to represent the column vectors of the constraint equations. The right-hand side of the constraints gives the vector $\mathbf{b}$.

In the computations below, we will give the algebraic formula for each step and its final numeric answer without detailing the arithmetic operations. You will find it instructive to fill in the gaps in each step.

**Iteration 0.**

$$\mathbf{X}_{B_0} = (x_3, x_4, x_5, x_6), \mathbf{C}_{B_0} = (0,0,0,0)$$
$$\mathbf{B}_0 = (\mathbf{P}_3, \mathbf{P}_4, \mathbf{P}_5, \mathbf{P}_6) = \mathbf{I}, \mathbf{B}_0^{-1} = \mathbf{I}$$

Thus,

$$\mathbf{X}_{B_0} = \mathbf{B}_0^{-1}\mathbf{b} = (24,6,1,2)^T, z = \mathbf{C}_{B_0}\mathbf{X}_{B_0} = 0$$

***Optimality Computations:***

$$\mathbf{C}_{B_0}\mathbf{B}_0^{-1} = (0,0,0,0)$$
$$\{z_j - c_j\}_{j=1,2} = \mathbf{C}_{B_0}\mathbf{B}_0^{-1}(\mathbf{P}_1,\mathbf{P}_2) - (c_1,c_2) = (-5,-4)$$

Thus, $\mathbf{P}_1$ is the entering vector.

***Feasibility Computations:***

$$\mathbf{X}_{B_0} = (x_3,x_4,x_5,x_6)^T = (24,6,1,2)^T$$
$$\mathbf{B}_0^{-1}\mathbf{P}_1 = (6,1,-1,0)^T$$

Hence,

$$x_1 = \min\left\{\frac{24}{6},\frac{6}{1},-,-\right\} = \min\{\mathbf{4},6,-,-\} = 4$$

and $\mathbf{P}_4$ becomes the leaving vector.

The results above can be summarized in the familiar simplex tableau format. The presentation should help convince you that the two methods are essentially the same.

| Basic | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ | Solution |
|---|---|---|---|---|---|---|---|
| $z$ | $-5$ | $-4$ | 0 | 0 | 0 | 0 | 0 |
| $x_3$ | 6 | | | | | | 24 |
| $x_4$ | 1 | | | | | | 6 |
| $x_5$ | $-1$ | | | | | | 1 |
| $x_6$ | 0 | | | | | | 2 |

**Iteration 1.**

$$\mathbf{X}_{B_1} = (x_1,x_4,x_5,x_6), \mathbf{C}_{B_1} = (5,0,0,0)$$
$$\mathbf{B}_1 = (\mathbf{P}_1,\mathbf{P}_4,\mathbf{P}_5,\mathbf{P}_6)$$
$$= \begin{pmatrix} 6 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

By using an appropriate inversion method (see Section A.2.6, in particular the *product form* method), the inverse is given as

$$\mathbf{B}_1^{-1} = \begin{pmatrix} \frac{1}{6} & 0 & 0 & 0 \\ -\frac{1}{6} & 1 & 0 & 0 \\ \frac{1}{6} & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Thus,

$$\mathbf{X}_{B_1} = \mathbf{B}_1^{-1}\mathbf{b} = (4,2,5,2)^T, z = \mathbf{C}_{B_1}\mathbf{X}_{B_1} = 20$$

***Optimality Computations:***

$$\mathbf{C}_{B_1}\mathbf{B}_1^{-1} = (\tfrac{5}{6},0,0,0)$$

$$\{z_j - c_j\}_{j=2,3} = \mathbf{C}_{B_1}\mathbf{B}_1^{-1}(\mathbf{P}_2,\mathbf{P}_3) - (c_2,c_3) = (-\tfrac{2}{3},\tfrac{5}{6})$$

Thus, $\mathbf{P}_2$ is the entering vector.

***Feasibility Computations:***

$$\mathbf{X}_{B_1} = (x_1,x_4,x_5,x_6)^T = (4,2,5,2)^T$$

$$\mathbf{B}_1^{-1}\mathbf{P}_2 = (\tfrac{2}{3},\tfrac{4}{3},\tfrac{5}{3},1)^T$$

Hence,

$$x_2 = \min\left\{\frac{4}{\frac{2}{3}},\frac{2}{\frac{4}{3}},\frac{5}{\frac{5}{3}},\frac{2}{1}\right\} = \min\{6,\tfrac{3}{2},3,2\} = \tfrac{3}{2}$$

and $\mathbf{P}_4$ becomes the leaving vector. (You will find it helpful to summarize the results above in the simplex tableau format as we did in iteration 0.)

**Iteration 2.**

$$\mathbf{X}_{B_2} = (x_1,x_2,x_5,x_6)^T, \mathbf{C}_{B_2} = (5,4,0,0)$$

$$\mathbf{B}_2 = (\mathbf{P}_1,\mathbf{P}_2,\mathbf{P}_5,\mathbf{P}_6)$$

$$= \begin{pmatrix} 6 & 4 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ -1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

Hence,

$$\mathbf{B}_2^{-1} = \begin{pmatrix} \frac{1}{4} & -\frac{1}{2} & 0 & 0 \\ -\frac{1}{8} & \frac{3}{4} & 0 & 0 \\ \frac{3}{8} & -\frac{5}{4} & 1 & 0 \\ \frac{1}{8} & -\frac{3}{4} & 0 & 1 \end{pmatrix}$$

Thus,

$$\mathbf{X}_{B_2} = \mathbf{B}_2^{-1}\mathbf{b} = (3,\tfrac{3}{2},\tfrac{5}{2},\tfrac{1}{2})^T, z = \mathbf{C}_{B_2}\mathbf{X}_{B_2} = 21$$

***Optimality Computations:***

$$\mathbf{C}_{B_2}\mathbf{B}_2^{-1} = (\tfrac{3}{4},\tfrac{1}{2},0,0)$$

$$\{z_j - c_j\}_{j=3,4} = \mathbf{C}_{B_2}\mathbf{B}_2^{-1}(\mathbf{P}_3,\mathbf{P}_4) - (c_3,c_4) = (\tfrac{3}{4},\tfrac{1}{2})$$

Thus, $\mathbf{X}_{B_2}$ is optimal and the computations end.

***Summary of Optimal Solution:***

$$x_1 = 3, x_2 = 1.5, z = 21$$

## PROBLEM SET 7.2B

1. In Example 7.2-1, summarize the data of iteration 1 in the tableau format of Section 3.3.
2. Solve the following LPs by the revised simplex method:
   **(a)** Maximize $z = 6x_1 - 2x_2 + 3x_3$
   subject to

$$\begin{aligned} 2x_1 - x_2 + 2x_3 &\leq 2 \\ x_1 \qquad + 4x_3 &\leq 4 \\ x_1, x_2, x_3 &\geq 0 \end{aligned}$$

   **(b)** Maximize $z = 2x_1 + x_2 + 2x_3$
   subject to

$$\begin{aligned} 4x_1 + 3x_2 + 8x_3 &\leq 12 \\ 4x_1 + x_2 + 12x_3 &\leq 8 \\ 4x_1 - x_2 + 3x_3 &\leq 8 \\ x_1, x_2, x_3 &\geq 0 \end{aligned}$$

   **(c)** Minimize $z = 2x_1 + x_2$
   subject to

$$\begin{aligned} 3x_1 + x_2 &= 3 \\ 4x_1 + 3x_2 &\geq 6 \\ x_1 + 2x_2 &\leq 3 \\ x_1, x_2 &\geq 0 \end{aligned}$$

   **(d)** Minimize $z = 5x_1 - 4x_2 + 6x_3 + 8x_4$
   subject to

$$\begin{aligned} x_1 + 7x_2 + 3x_3 + 7x_4 &\leq 46 \\ 3x_1 - x_2 + x_3 + 2x_4 &\leq 20 \\ 2x_1 + 3x_2 - x_3 + x_4 &\geq 18 \\ x_1, x_2, x_3, x_4 &\geq 0 \end{aligned}$$

7.3

3. Solve the following LP by the revised simplex method given the starting basic feasible vector $\mathbf{X}_{B_0} = (x_2, x_4, x_5)^T$.

$$\text{Minimize } z = 7x_2 + 11x_3 - 10x_4 + 26x_6$$

subject to

$$\begin{aligned} x_2 - x_3 \qquad + x_5 + x_6 &= 6 \\ x_2 - x_3 + x_4 \qquad + 3x_6 &= 8 \\ x_1 + x_2 - 3x_3 + x_4 + x_5 \qquad &= 12 \\ x_1, x_2, x_3, x_4, x_5, x_6 &\geq 0 \end{aligned}$$

4. Solve the following using the two-phase revised simplex method:
   **(a)** Problem 2-c
   **(b)** Problem 2-d
   **(c)** Problem 3 (ignore the given starting $\mathbf{X}_{B_0}$ )
5. *Revised Dual Simplex Method.* The steps of the revised dual simplex method (using matrix manipulations) can be summarized as follows:

   **Step 0.** Let $\mathbf{B}_0 = \mathbf{I}$ be the starting basis and that at least one of the elements of $\mathbf{X}_{B_0}$ is negative (infeasible).

   **Step 1.** Compute $\mathbf{X}_B = \mathbf{B}^{-1}\mathbf{b}$, the current values of the basic variables. Select the leaving variable $x_r$ as the one having the most negative value. If all the elements of $\mathbf{X}_B$ are nonnegative, stop; the current solution is feasible (and optimal).

   **Step 2.** **(a)** Compute $z_j - c_j = \mathbf{C}_B\mathbf{B}^{-1}\mathbf{P}_j - c_j$ for all the nonbasic variables $x_j$.
   **(b)** For all the nonbasic variables $x_j$, compute the constraint coefficients $(\mathbf{B}^{-1}\mathbf{P}_j)_r$ associated with the row of the leaving variable $x_r$.
   **(c)** The entering variable is associated with

$$\theta = \min_j \left\{ \left| \frac{z_j - c_j}{(\mathbf{B}^{-1}\mathbf{P}_j)_r} \right|, (\mathbf{B}^{-1}\mathbf{P}_j)_r < 0 \right\}$$

   If all $(\mathbf{B}^{-1}\mathbf{P}_j)_r \geq 0$, no feasible solution exists.

   **Step 3.** Obtain the new basis by interchanging the entering and leaving vectors ( $\mathbf{P}_j$ and $\mathbf{P}_r$ ). Compute the new inverse and go to Step 1.

   Apply the method to the following problem:

$$\text{Minimize } z = 2x_1 + x_2$$

   subject to

$$\begin{aligned} 3x_1 + x_2 &\geq 3 \\ 4x_1 + 3x_2 &\geq 6 \\ x_1 + x_2 &\leq 3 \\ x_1, x_2 &\geq 0 \end{aligned}$$

## 7.3 BOUNDED VARIABLES ALGORITHM

In LP models, variables may have explicit positive upper and lower bounds. For example, in production facilities, lower and upper bounds can represent the minimum and maximum demands for certain products. Bounded variables also arise prominently in the course of solving integer programming problems by the branch-and-bound algorithm (see Section 9.3.1).

The bounded algorithm is efficient computationally because it accounts for the bounds *implicitly*. We consider the lower bounds first because it is simpler. Given $\mathbf{X} \geq \mathbf{L}$, we use the substitution

$$\mathbf{X} = \mathbf{L} + \mathbf{X}', \quad \mathbf{X}' \geq \mathbf{0}$$

and solve the problem in terms of $\mathbf{X}'$ (whose lower bound now equals zero). The original $\mathbf{X}$ is determined by back-substitution, which is legitimate because it guarantees that $\mathbf{X} = \mathbf{X}' + \mathbf{L}$ will remain nonnegative for all $\mathbf{X}' \geq \mathbf{0}$.

Next, consider the upper bounding constraints, $\mathbf{X} \leq \mathbf{U}$. The idea of direct substitution (i.e., $\mathbf{X} = \mathbf{U} - \mathbf{X}'', \mathbf{X}'' \geq \mathbf{0}$ ) is not correct because back-substitution, $\mathbf{X} = \mathbf{U} - \mathbf{X}''$, does not ensure that $\mathbf{X}$ will remain nonnegative. A different procedure is thus needed.

Define the upper bounded LP model as

$$\text{Maximize } z = \{\mathbf{CX} | (\mathbf{A}, \mathbf{I})\mathbf{X} = \mathbf{b}, \mathbf{0} \leq \mathbf{X} \leq \mathbf{U}\}$$

The bounded algorithm uses only the constraints $(\mathbf{A}, \mathbf{I})\mathbf{X} = \mathbf{b}, \mathbf{X} \geq \mathbf{0}$ explicitly and accounts for $\mathbf{X} \leq \mathbf{U}$ implicitly by modifying the simplex feasibility condition.

Let $\mathbf{X}_B = \mathbf{B}^{-1}\mathbf{b}$ be a current basic feasible solution of $(\mathbf{A}, \mathbf{I})\mathbf{X} = \mathbf{b}, \mathbf{X} \geq \mathbf{0}$ and suppose that, according to the (regular) optimality condition, $\mathbf{P}_j$ is the entering vector. Then, *given that all the nonbasic variables are zero,* the constraint equation of the $i$th basic variable can be written as

$$(\mathbf{X}_B)_i = (\mathbf{B}^{-1}\mathbf{b})_i - (\mathbf{B}^{-1}\mathbf{P}_j)_i\, x_j$$

When the entering variable $x_j$ increases above zero level, $(\mathbf{X}_B)_i$ will *increase or decrease* depending on whether $(\mathbf{B}^{-1}\mathbf{P}_j)_i$ is negative or positive, respectively. Thus, in determining the value of the entering variable $x_j$, three conditions must be satisfied:

1. The basic variable $(\mathbf{X}_B)_i$ remains nonnegative—that is, $(\mathbf{X}_B)_i \geq 0$.
2. The basic variable $(\mathbf{X}_B)_i$ does not exceed its upper bound—that is, $(\mathbf{X}_B)_i \leq (\mathbf{U}_B)_i$, where $\mathbf{U}_B$ comprises the ordered elements of $\mathbf{U}$ corresponding to $\mathbf{X}_B$.
3. The entering variable $x_j$ cannot assume a value larger than its upper bound—that is, $x_j \leq u_j$, where $u_j$ is the $j$th element of $\mathbf{U}$.

The first condition $(\mathbf{X}_B)_i \geq 0$ requires that

$$(\mathbf{B}^{-1}\mathbf{b})_i - (\mathbf{B}^{-1}\mathbf{P}_j)_i x_j \geq 0$$

It is satisfied if

$$x_j \leq \theta_1 = \min_i \left\{ \frac{(\mathbf{B}^{-1}\mathbf{b})_i}{(\mathbf{B}^{-1}\mathbf{P}_j)_i} \middle| (\mathbf{B}^{-1}\mathbf{P}_j)_i > 0 \right\}$$

This condition is the same as the feasibility condition of the regular simplex method.

Next, the condition $(\mathbf{X}_B)_i \leq (\mathbf{U}_B)_i$ specifies that

$$(\mathbf{B}^{-1}\mathbf{b})_i - (\mathbf{B}^{-1}\mathbf{P}_j)_i x_j \leq (\mathbf{U}_B)_i$$

It is satisfied if

$$x_j \leq \theta_2 = \min_i \left\{ \frac{(\mathbf{B}^{-1}\mathbf{b})_i - (\mathbf{U}_B)_i}{(\mathbf{B}^{-1}\mathbf{P}_j)_i} \middle| (\mathbf{B}^{-1}\mathbf{P}_j)_i < 0 \right\}$$

Combining the three restrictions, $x_j$ enters the solution at the level that satisfies

$$x_j = \min(\theta_1, \theta_2, u_j)$$

The change of basis for the next iteration depends on whether $x_j$ enters the solution at level $\theta_1, \theta_2$, or $u_j$. Assuming that $(\mathbf{X}_B)_r$ is the leaving variable, then we have the following rules:

1. $x_j = \theta_1$: $(\mathbf{X}_B)_r$ leaves the basic solution (becomes nonbasic) at level zero. The new iteration is generated in the normal simplex manner by using $x_j$ and $(\mathbf{X}_B)_r$ as the entering and the leaving variables, respectively.
2. $x_j = \theta_2$: $(\mathbf{X}_B)_r$ becomes nonbasic *at its upper bound*. The new iteration is generated as in the case of $x_j = \theta_1$, with one modification that accounts for the fact that $(\mathbf{X}_B)_r$ will be nonbasic at *upper bound*. Because the values of $\theta_1$ and $\theta_2$ are developed under the assumption that *all nonbasic variables are at zero level* (convince yourself that this is the case!), we must convert the new nonbasic $(\mathbf{X}_B)_r$ at upper bound to a nonbasic variable at zero level. This is achieved by using the substitution $(\mathbf{X}_B)_r = (\mathbf{U}_B)_r - (\mathbf{X}_B)'_r$, where $(\mathbf{X}_B)'_r \geq 0$. It is immaterial whether the substitution is made before or after the new basis is computed.
3. $x_j = u_j$: The basic vector $\mathbf{X}_B$ remains unchanged because $x_j = u_j$ stops short of forcing any of the current basic variables to reach its lower (= 0) or upper bound. This means that $x_j$ will remain nonbasic *but at upper bound*. Following the argument just presented, the new iteration is generated by using the substitution $x_j = u_j - x'_j$.

A tie among $\theta_1, \theta_2$, and $u_j$ may be broken arbitrarily. However, it is preferable, where possible, to implement the rule for $x_j = u_j$ because it entails less computations.

The substitution $x_j = u_j - x'_j$ will change the original $c_j, \mathbf{P}_j$, and $\mathbf{b}$ to $c'_j = -c_j, \mathbf{P}'_j = -\mathbf{P}_j$, and $\mathbf{b}$ to $\mathbf{b}' = \mathbf{b} - u_j\mathbf{P}_j$. This means that if the revised simplex method is used, all the computations (e.g., $\mathbf{B}^{-1}, \mathbf{X}_B$, and $z_j - c_j$) should be based on the updated values of **C**, **A**, and **b** at each iteration (see Problem 5, Set 7.3a for further details).

## Example 7.3-1

Solve the following LP model by the upper-bounding algorithm.[2]

$$\text{Maximize } z = 3x_1 + 5y + 2x_3$$

subject to

$$x_1 + y + 2x_3 \leq 14$$

$$2x_1 + 4y + 3x_3 \leq 43$$

$$0 \leq x_1 \leq 4, 7 \leq y \leq 10, 0 \leq x_3 \leq 3$$

The lower bound on $y$ is accounted for using the substitution $y = x_2 + 7$, where $0 \leq x_2 \leq 10 - 7 = 3$.

We will not use the revised simplex method to carry out the computations, to avoid being "sidetracked" by the computational details. Instead, we will use the compact tableau form. Problems 5, 6, and 7, Set 7.3a address the revised version of the algorithm.

[2]You can use TORA's `Linear Programming` ⇒ `Solve problem` ⇒ `Algebraic` ⇒ `Iterations` ⇒ `Bounded simplex` to produce the associated simplex iterations.

**Iteration 0.**

| Basic | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | Solution |
|---|---|---|---|---|---|---|
| $z$ | $-3$ | $-5$ | $-2$ | 0 | 0 | 35 |
| $x_4$ | 1 | 1 | 2 | 1 | 0 | 7 |
| $x_5$ | 2 | 4 | 3 | 0 | 1 | 15 |

We have $\mathbf{B} = \mathbf{B}^{-1} = \mathbf{I}$ and $\mathbf{X}_B = (x_4, x_5)^T = \mathbf{B}^{-1}\mathbf{b} = (7,15)^T$. Given $x_2$ is the entering variable ($z_2 - c_2 = -5$), we get

$$\mathbf{B}^{-1}\mathbf{P}_2 = (1,4)^T$$

which yields

$$\theta_1 = \min\left\{\frac{7}{1}, \frac{15}{4}\right\} = 3.75, \text{ corresponding to } x_5$$

$$\theta_2 = \infty \text{ (because } \mathbf{B}^{-1}\mathbf{P}_2 > \mathbf{0})$$

Next, given the upper bound on the entering variable, $x_2 \leq 3$, it follows that

$$x_2 = \min\{3.75, \infty, 3\} = 3\ (=u_2)$$

Because $x_2$ enters at its upper bound ($= u_2 = 3$), $\mathbf{X}_B$ remains unchanged, and $x_2$ becomes nonbasic *at its upper bound.* We use the substitution $x_2 = 3 - x_2'$ to obtain the new tableau as

| Basic | $x_1$ | $x_2'$ | $x_3$ | $x_4$ | $x_5$ | Solution |
|---|---|---|---|---|---|---|
| $z$ | $-3$ | 5 | $-2$ | 0 | 0 | 50 |
| $x_4$ | 1 | $-1$ | 2 | 1 | 0 | 4 |
| $x_5$ | 2 | $-4$ | 3 | 0 | 1 | 3 |

The substitution in effect changes the original right-hand side vector from $\mathbf{b} = (7,15)^T$ to $\mathbf{b}' = (4,3)^T$. This change should be considered in future computations.

**Iteration 1.** The entering variable is $x_1$. The basic vector $\mathbf{X}_B$ and $\mathbf{B}^{-1}$ ($=\mathbf{I}$) are the same as in Iteration 0. Next,

$$\mathbf{B}^{-1}\mathbf{P}_1 = (1,2)^T$$

$$\theta_1 = \min\left\{\frac{4}{1}, \frac{3}{2}\right\} = 1.5, \text{ corresponding to basic } x_5$$

$$\theta_2 = \infty \text{ (because } \mathbf{B}^{-1}\mathbf{P}_1 > \mathbf{0})$$

Thus,

$$x_1 = \min\{1.5, \infty, 4\} = 1.5\ (=\theta_1)$$

Thus, the entering variable $x_1$ becomes basic, and the leaving variable $x_5$ becomes nonbasic at zero level, which yields

| Basic | $x_1$ | $x_2'$ | $x_3$ | $x_4$ | $x_5$ | Solution |
|---|---|---|---|---|---|---|
| $z$ | 0 | −1 | $\frac{5}{2}$ | 0 | $\frac{3}{2}$ | $\frac{109}{2}$ |
| $x_4$ | 0 | 1 | $\frac{1}{2}$ | 1 | $-\frac{1}{2}$ | $\frac{5}{2}$ |
| $x_1$ | 1 | −2 | $\frac{3}{2}$ | 0 | $\frac{1}{2}$ | $\frac{3}{2}$ |

**Iteration 2.** The new inverse is

$$\mathbf{B}^{-1} = \begin{pmatrix} 1 & -\frac{1}{2} \\ 0 & \frac{1}{2} \end{pmatrix}$$

Now

$$\mathbf{X}_B = (x_4, x_1)^T = \mathbf{B}^{-1}\mathbf{b}' = (\tfrac{5}{2}, \tfrac{3}{2})^T$$

where $\mathbf{b}' = (4,3)^T$ as computed at the end of Iteration 0. We select $x_2'$ as the entering variable, and, noting that $\mathbf{P}_2' = -\mathbf{P}_2$, we get

$$\mathbf{B}^{-1}\mathbf{P}_2' = (1, -2)^T$$

Thus,

$$\theta_1 = \min\left\{\frac{\frac{5}{2}}{1}, —\right\} = 2.5, \text{corresponding to basic } x_4$$

$$\theta_2 = \min\left\{—, \frac{\frac{3}{2} - 4}{-2}\right\} = 1.25, \text{corresponding to basic } x_1$$

We then have

$$x_2' = \min\{2.5, 1.25, 3\} = 1.25 \; (=\theta_2)$$

Because $x_1$ becomes nonbasic at its upper bound, we apply the substitution $x_1 = 4 - x_1'$ to obtain

| Basic | $x_1'$ | $x_2'$ | $x_3$ | $x_4$ | $x_5$ | Solution |
|---|---|---|---|---|---|---|
| $z$ | 0 | −1 | $\frac{5}{2}$ | 0 | $\frac{3}{2}$ | $\frac{109}{2}$ |
| $x_4$ | 0 | 1 | $\frac{1}{2}$ | 1 | $-\frac{1}{2}$ | $\frac{5}{2}$ |
| $x_1'$ | −1 | −2 | $\frac{3}{2}$ | 0 | $\frac{1}{2}$ | $-\frac{5}{2}$ |

Next, the entering variable $x_2'$ becomes basic and the leaving variable $x_1'$ becomes nonbasic at zero level, which yields

| Basic | $x_1'$ | $x_2'$ | $x_3$ | $x_4$ | $x_5$ | Solution |
|---|---|---|---|---|---|---|
| $z$ | $\frac{1}{2}$ | 0 | $\frac{7}{4}$ | 0 | $\frac{5}{4}$ | $\frac{223}{4}$ |
| $x_4$ | $-\frac{1}{2}$ | 0 | $\frac{5}{4}$ | 1 | $-\frac{1}{4}$ | $\frac{5}{4}$ |
| $x_2'$ | $\frac{1}{2}$ | 1 | $-\frac{3}{4}$ | 0 | $-\frac{1}{4}$ | $\frac{5}{4}$ |

The last tableau is feasible and optimal. Note that the last two steps could have been reversed—meaning that we could first make $x_2'$ basic and then apply the substitution $x_1 = 4 - x_1'$ (try it!). The sequence presented here involves less computations, however.

The optimal values of $x_1,x_2$, and $x_3$ are obtained by back-substitution as $x_1 = u_1 - x_1' = 4 - 0 = 4$, $x_2 = u_2 - x_2' = 3 - \frac{5}{4} = \frac{7}{4}$, and $x_3 = 0$. Finally, we get $y = l_2 + x_2 = 7 + \frac{7}{4} = \frac{35}{4}$. The associated optimal value of $z$ is $\frac{223}{4}$.

## PROBLEM SET 7.3A

**1.** Consider the following linear program:

$$\text{Maximize } z = 2x_1 + x_2$$

subject to

$$x_1 + x_2 \leq 3$$

$$0 \leq x_1 \leq 2, 0 \leq x_2 \leq 2$$

**(a)** Solve the problem graphically, and trace the sequence of extreme points leading to the optimal solution.

**(b)** Solve the problem by the upper bounding algorithm and show that the method produces the same sequence of extreme points as in the graphical optimal solution (use TORA to generate the iterations).

**(c)** How does the upper-bounding algorithm recognize the extreme points?

**2.** Solve the following problem by the bounded algorithm:

$$\text{Maximize } z = 6x_1 + 2x_2 + 8x_3 + 4x_4 + 2x_5 + 10x_6$$

subject to

$$8x_1 + x_2 + 8x_3 + 2x_4 + 2x_5 + 4x_6 \leq 13$$

$$0 \leq x_j \leq 1, j = 1,2,\ldots,6$$

**3.** Solve the following problems by the bounded algorithm:

**(a)** Minimize $z = 6x_1 - 2x_2 - 3x_3$
subject to

$$2x_1 + 4x_2 + 2x_3 \leq 8$$

$$x_1 - 2x_2 + 3x_3 \leq 7$$

$$0 \leq x_1 \leq 2, 0 \leq x_2 \leq 2, 0 \leq x_3 \leq 1$$

**(b)** Maximize $z = 3x_1 + 5x_2 + 2x_3$
subject to

$$x_1 + 2x_2 + 2x_3 \leq 10$$

$$2x_1 + 4x_2 + 3x_3 \leq 15$$

$$0 \leq x_1 \leq 4, 0 \leq x_2 \leq 3, 0 \leq x_3 \leq 3$$

4. In the following problems, some of the variables have positive lower bounds. Use the bounded algorithm to solve these problems.

(a) Maximize $z = 3x_1 + 2x_2 - 2x_3$
subject to

$$2x_1 + x_2 + x_3 \le 8$$
$$x_1 + 2x_2 - x_3 \ge 3$$
$$1 \le x_1 \le 3, 0 \le x_2 \le 3, 2 \le x_3$$

(b) Maximize $z = x_1 + 2x_2$
subject to

$$-x_1 + 2x_2 \ge 0$$
$$3x_1 + 2x_2 \le 10$$
$$-x_1 + x_2 \le 1$$
$$1 \le x_1 \le 3, 0 \le x_2 \le 1$$

(c) Maximize $z = 4x_1 + 2x_2 + 6x_3$
subject to

$$4x_1 - x_2 \le 9$$
$$-x_1 + x_2 + 2x_3 \le 8$$
$$-3x_1 + x_2 + 4x_3 \le 12$$
$$1 \le x_1 \le 3, 0 \le x_2 \le 5, 0 \le x_3 \le 2$$

5. Consider the matrix definition of the bounded variables problem. Suppose that the vector $\mathbf{X}$ is partitioned into $(\mathbf{X}_z, \mathbf{X}_u)$, where $\mathbf{X}_u$ represents the basic *and* nonbasic variables that are substituted at upper bound. The problem may thus be written as

$$\begin{pmatrix} 1 & -\mathbf{C}_z & -\mathbf{C}_u \\ 0 & \mathbf{D}_z & \mathbf{D}_u \end{pmatrix} \begin{pmatrix} z \\ \mathbf{X}_z \\ \mathbf{X}_u \end{pmatrix} = \begin{pmatrix} 0 \\ \mathbf{b} \end{pmatrix}$$

Using $\mathbf{X}_u = \mathbf{U}_u - \mathbf{X}'_u$ where $\mathbf{U}_u$ is a subset of $\mathbf{U}$ representing the upper bounds for $\mathbf{X}_u$, let $\mathbf{B}$ (and $\mathbf{X}_B$) be the basis of the current simplex iteration after $\mathbf{X}_u$ has been substituted out. Show that the associated general simplex tableau is given as

| Basic | $\mathbf{X}_z^T$ | $\mathbf{X'}_u^T$ | Solution |
|---|---|---|---|
| $z$ | $\mathbf{C}_B\mathbf{B}^{-1}\mathbf{D}_z - \mathbf{C}_z$ | $-\mathbf{C}_B\mathbf{B}^{-1}\mathbf{D}_u + \mathbf{C}_u$ | $\mathbf{C}_u\mathbf{B}^{-1}\mathbf{b}' + \mathbf{C}_u\mathbf{U}_u$ |
| $\mathbf{X}_B$ | $\mathbf{B}^{-1}\mathbf{D}_z$ | $-\mathbf{B}^{-1}\mathbf{D}_u$ | $\mathbf{B}^{-1}\mathbf{b}'$ |

where $\mathbf{b}' = \mathbf{b} - \mathbf{D}_u\mathbf{U}_u$.

6. In Example 7.3-1, do the following:

(a) In Iteration 1, verify that $\mathbf{X}_B = (x_4, x_1)^T = (\frac{5}{2}, \frac{3}{2})^T$ by using matrix manipulation.

(b) In Iteration 2, show how $\mathbf{B}^{-1}$ can be computed from the original data of the problem. Then verify the given values of basic $x_4$ and $x'_2$ using matrix manipulation.

7. Solve part (a) of Problem 3 using the revised simplex (matrix) version for upper bounded variables.

8. *Bounded Dual Simplex Algorithm.* The dual simplex algorithm (Section 4.4) can be modified to accommodate the bounded variables as follows. Given the upper bound constraint $x_j \leq u_j$ for all $j$ (if $u_j$ is infinite, replace it with a sufficiently large upper bound $M$), the LP problem is converted to a dual feasible (i.e., primal optimal) form by using the substitution $x_j = u_j - x_j'$, where necessary.

**Step 1.** If any of the current basic variables $(\mathbf{X}_B)_i$ exceeds its upper bound, use the substitution $(\mathbf{X}_B)_i = (\mathbf{U}_B)_i - (\mathbf{X}_B)_i'$. Go to step 2.

**Step 2.** If all the basic variables are feasible, stop. Otherwise, select the leaving variable $x_r$ as the basic variable having the most negative value. Go to step 3.

**Step 3.** Select the entering variable using the optimality condition of the regular dual simplex method. Go to step 4.

**Step 4.** Perform a change of basis. Go to step 1.

Apply the given algorithm to the following problems:

**(a)** Minimize $z = 3x_1 - 2x_2 + 2x_3$

subject to

$$2x_1 + x_2 + x_3 \leq 8$$
$$-x_1 + 2x_2 + x_3 \geq 13$$
$$0 \leq x_1 \leq 2, 0 \leq x_2 \leq 3, 0 \leq x_3 \leq 1$$

**(b)** Maximize $z = x_1 + 5x_2 - 2x_3$

subject to

$$4x_1 + 2x_2 + 2x_3 \leq 26$$
$$x_1 + 3x_2 + 4x_3 \geq 17$$
$$0 \leq x_1 \leq 2, 0 \leq x_2 \leq 3, x_3 \geq 0$$

## 7.4 DECOMPOSITION ALGORITHM

Consider the situation of developing a master corporate plan for several production facilities. Although each facility has its own independent capacity and production constraints, the different facilities are tied together at the corporate level by budgetary considerations. The resulting model includes two types of constraints: *common*, representing the corporate budgetary constraints, and *independent,* representing the internal capacity and production restrictions of each facility. Figure 7.5 depicts the layout of the resulting constraints for $n$ activities (facilities). In the absence of the common constraints, all activities operate independently.

The decomposition algorithm improves the computational efficiency of the problem depicted in Figure 7.5 by breaking it down into $n$ subproblems that can be solved almost independently. We point out, however, that the need for the decomposition algorithm was more justifiable in the past when the speed and memory of the computer were modest. Today, computers boast impressive capabilities, and the need for the decomposition algorithm may not be warranted. Nevertheless, we present the algorithm here because of its interesting theoretical contribution.

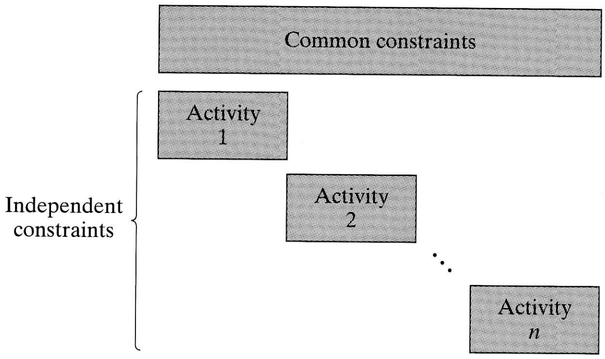



FIGURE 7.5
Layout of a decomposable linear program

The corresponding mathematical model is given as

$$\text{Maximize } z = \mathbf{C}_1\mathbf{X}_1 + \mathbf{C}_2\mathbf{X}_2 + \dots + \mathbf{C}_n\mathbf{X}_n$$

subject to

$$\begin{aligned}
\mathbf{A}_1\mathbf{X}_1 + \mathbf{A}_2\mathbf{X}_2 + \dots + \mathbf{A}_n\mathbf{X}_n &\le \mathbf{b}_0 \\
\mathbf{D}_1\mathbf{X}_1 \qquad\qquad &\le \mathbf{b}_1 \\
\mathbf{D}_2\mathbf{X}_2 \qquad &\le \mathbf{b}_2 \\
\ddots \qquad\qquad &\quad \vdots \\
\mathbf{D}_n\mathbf{X}_n &\le \mathbf{b}_n \\
\mathbf{X}_j \ge \mathbf{0}, \ j = 1,2,\dots,n &
\end{aligned}$$

The slack and surplus variables are added as necessary to convert all the inequalities into equations.

The decomposition principle is based on representing the entire problem in terms of the *extreme points* of the sets $\mathbf{D}_j\mathbf{X}_j \le \mathbf{b}_j, \mathbf{X}_j \ge \mathbf{0}, j = 1,2,\dots,n$. To do so, the solution space described by each $\mathbf{D}_j\mathbf{X}_j \le \mathbf{b}_j, \mathbf{X}_j \ge \mathbf{0}$ must be bounded. This requirement can always be satisfied for any set $j$ by adding the artificial restriction $\mathbf{1X}_j \le M$, where $M$ is sufficiently large.

Suppose that the extreme points of $\mathbf{D}_j\mathbf{X}_j \le \mathbf{b}_j, \mathbf{X}_j \ge \mathbf{0}$ are defined as $\hat{\mathbf{X}}_{jk}, k = 1,2,\dots,K_j$. We then have

$$\mathbf{X}_j = \sum_{k=1}^{K_j} \beta_{jk}\hat{\mathbf{X}}_{jk}, \ j = 1,2,\dots,n$$

where $\beta_{jk} \ge 0$ for all $k$ and $\sum_{k=1}^{k_j} \beta_{jk} = 1$

We can reformulate the entire problem in terms of the extreme points to obtain the following **master problem**:

$$\text{Maximize } = \sum_{k=1}^{K_1} \mathbf{C}_1\hat{\mathbf{X}}_{1k}\beta_{1k} + \sum_{k=1}^{K_2} \mathbf{C}_2\hat{\mathbf{X}}_{2k}\beta_{2k} + \dots + \sum_{k=1}^{K_n} \mathbf{C}_n\hat{\mathbf{X}}_{nk}\beta_{nk}$$

subject to

$$\sum_{k=1}^{K_1} \mathbf{A}_1\hat{\mathbf{X}}_{1k}\beta_{1k} + \sum_{k=1}^{K_2} \mathbf{A}_2\hat{\mathbf{X}}_{2k}\beta_{2k} + \dots + \sum_{k=1}^{K_n} \mathbf{A}_n\hat{\mathbf{X}}_{nk}\beta_{nk} \le \mathbf{b}_0$$

$$\sum_{k=1}^{K_1} \beta_{1k} = 1$$

$$\sum_{k=1}^{K_2} \beta_{2k} = 1$$

$$\vdots$$

$$\sum_{k=1}^{K_n} \beta_{nk} = 1$$

$$\beta_{jk} \ge 0, \text{ for all } j \text{ and } k$$

The new variables in the master problem are $\beta_{jk}$. Once their optimal values, $\beta^*_{jk}$, are determined, we can find the optimal solution to the original problem by back-substitution as

$$\mathbf{X}_j = \sum_{k=1}^{K_j} \beta^*_{jk}\hat{\mathbf{X}}_{jk}, j = 1,2,\dots,n$$

It may appear that the solution of the master problem requires prior determination of *all* the extreme points $\hat{\mathbf{X}}_{jk}$, a difficult task indeed! Fortunately, it is not so.

To solve the master problem by the revised simplex method (Section 7.2), we need to determine the entering and the leaving variables at each iteration. Let us start first with the entering variable. Given $\mathbf{C}_B$ and $\mathbf{B}^{-1}$ of the current basis of the master problem, then for nonbasic $\beta_{jk}$, we have

$$z_{jk} - c_{jk} = \mathbf{C}_B\mathbf{B}^{-1}\mathbf{P}_{jk} - c_{jk}$$

where

$$c_{jk} = \mathbf{C}_j\hat{\mathbf{X}}_{jk} \text{ and } \mathbf{P}_{jk} = \begin{pmatrix} \mathbf{A}_j\hat{\mathbf{X}}_{jk} \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix}$$

Now, to decide which, if any, of the variable $\beta_{jk}$ should enter the solution, we need to determine

$$z_{j^*k^*} - c_{j^*k^*} = \min_{\text{all } j \text{ and } k} \{z_{jk} - c_{jk}\}$$

If $z_{j^*k^*} - c_{j^*k^*} < 0$, then, according to the maximization optimality condition, $\beta_{j^*k^*}$ must enter the solution; otherwise, the optimum has been reached.

We still have not shown how $z_{j^*k^*} - c_{j^*k^*}$ is computed numerically. The idea lies in the following identity

$$\min_{\text{all } j \text{ and } k} \{z_{jk} - c_{jk}\} = \min_j\{\min_k\{z_{jk} - c_{jk}\}\}$$

The reason we are able to establish this identity is that each convex set $\mathbf{D}_j\mathbf{X}_j \leq \mathbf{b}_j$, $\mathbf{X}_j \geq \mathbf{0}$ has its independent set of extreme points. In effect, what the identity says is that we can determine $z_{j^*k^*} - c_{j^*k^*}$ in two steps:

**Step 1.** For each convex set $\mathbf{D}_j\mathbf{X}_j \leq \mathbf{b}_j, \mathbf{X}_j \geq \mathbf{0}$, determine the extreme point $\hat{\mathbf{X}}_{jk^*}$ that yields the smallest $z_{jk} - c_{jk}$—that is, $z_{jk^*} - c_{jk^*} = \min_k\{z_{jk} - c_{jk}\}$.

**Step 2.** Determine $z_{j^*k^*} - c_{j^*k^*} = \min\limits_j\{z_{jk^*} - c_{jk^*}\}$.

From LP theory, we know that the optimum solution, when finite, must be associated with an extreme point of the solution space. Because each of the sets $\mathbf{D}_j\mathbf{X}_j \leq \mathbf{b}_j$, $\mathbf{X}_j \geq \mathbf{0}$ is bounded by definition, step 1 is mathematically equivalent to solving $n$ linear programs of the form

$$\text{Minimize } w_j = \{z_j - c_j | \mathbf{D}_j\mathbf{X}_j \leq \mathbf{b}_j, \mathbf{X}_j \geq \mathbf{0}\}$$

Actually, the objective function $w_j$ is a linear function in $\mathbf{X}_j$ (see Problem 8, Set 7.4a).

The determination of the entering variable $\beta_{j^*k^*}$ in the master problem reduces to solving $n$ (smaller) linear programs to determine the "entering" extreme point $\hat{\mathbf{X}}_{j^*k^*}$. This approach precludes the need to determine all the extreme points of all $n$ convex sets. Once the desired extreme point is located, all the elements of the column vector $\mathbf{P}_{j^*k^*}$ are at hand. Given that information, we can then determine the leaving variable and, subsequently, compute the next $\mathbf{B}^{-1}$ using the revised simplex method computations.

---

**Example 7.4-1**

Solve the following LP by the decomposition algorithm:

$$\text{Maximize } z = 3x_1 + 5x_2 + x_4 + x_5$$

subject to

$$\begin{aligned} x_1 + x_2 + x_3 + x_4 &\leq 40 \\ 5x_1 + x_2 \qquad\qquad &\leq 12 \\ x_3 + x_4 &\geq 5 \\ x_3 + 5x_4 &\leq 50 \\ x_1, x_2, x_3, x_4 &\geq 0 \end{aligned}$$

The problem has two subproblems that correspond to the following sets of variables:

$$\mathbf{X}_1 = (x_1, x_2)^T, \mathbf{X}_2 = (x_3, x_4)^T$$

The master problem corresponding to the problem above may thus be represented as follows:

| Subproblem 1 | | | | Subproblem 2 | | | | Starting basic solution | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\beta_{11}$ | $\beta_{12}$ | $\dots$ | $\beta_{1K_1}$ | $\beta_{21}$ | $\beta_{22}$ | $\dots$ | $\beta_{2K_2}$ | $x_5$ | $x_6$ | $x_7$ | |
| $\mathbf{C}_1\hat{\mathbf{X}}_{11}$ | $\mathbf{C}_1\hat{\mathbf{X}}_{12}$ | $\dots$ | $\mathbf{C}_1\hat{\mathbf{X}}_{1K_1}$ | $\mathbf{C}_2\hat{\mathbf{X}}_{21}$ | $\mathbf{C}_2\hat{\mathbf{X}}_{22}$ | $\dots$ | $\mathbf{C}_2\hat{\mathbf{X}}_{2K_2}$ | 0 | $-M$ | $-M$ | |
| $\mathbf{A}_1\hat{\mathbf{X}}_{11}$ | $\mathbf{A}_1\hat{\mathbf{X}}_{12}$ | $\dots$ | $\mathbf{A}_1\hat{\mathbf{X}}_{1K_1}$ | $\mathbf{A}_2\hat{\mathbf{X}}_{21}$ | $\mathbf{A}_2\hat{\mathbf{X}}_{22}$ | $\dots$ | $\mathbf{A}_2\hat{\mathbf{X}}_{2K_2}$ | 1 | 0 | 0 | = 40 |
| 1 | 1 | $\dots$ | 1 | 0 | 0 | $\dots$ | 0 | 0 | 1 | 0 | = 1 |
| 0 | 0 | $\dots$ | 0 | 1 | 1 | $\dots$ | 1 | 0 | 0 | 1 | = 1 |
| $\mathbf{C}_1 = (3,5)$<br>$\mathbf{A}_1 = (1,1)$<br>Solution space, $\mathbf{D}_1\mathbf{X}_1 \le \mathbf{b}_1$:<br>$5x_1 + x_2 \le 12$<br>$x_1, x_2 \ge 0$ | | | | $\mathbf{C}_2 = (1,1)$<br>$\mathbf{A}_2 = (1,1)$<br>Solution space, $\mathbf{D}_2\mathbf{X}_2 \le \mathbf{b}_2$:<br>$x_3 + x_4 \ge 5$<br>$x_3 + 5x_4 \le 50$<br>$x_3, x_4 \ge 0$ | | | | | | | |

Notice that $x_5$ is the slack variable that converts the common constraint to the following equation

$$x_1 + x_2 + x_3 + x_4 + x_5 = 40$$

Recall that subproblems 1 and 2 account for variables $x_1, x_2, x_3$, and $x_4$ only. This is the reason $x_5$ must appear explicitly in the master problem. The remaining starting basic variables, $x_6$ and $x_7$, are artificial.

**Iteration 0.**

$$\mathbf{X}_B = (x_5, x_6, x_7)^T = (40, 1, 1)^T$$

$$\mathbf{C}_B = (0, -M, -M), \mathbf{B} = \mathbf{B}^{-1} = \mathbf{I}$$

**Iteration 1.**

***Subproblem 1*** $(j = 1)$. We have

$$z_1 - c_1 = \mathbf{C}_B\mathbf{B}^{-1}\begin{pmatrix} \mathbf{A}_1\mathbf{X}_1 \\ 1 \\ 0 \end{pmatrix} - \mathbf{C}_1\mathbf{X}_1$$

$$= (0, -M, -M)\begin{pmatrix} (1,1)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ 1 \\ 0 \end{pmatrix} - (3, 5)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$= -3x_1 - 5x_2 - M$$

Thus, the corresponding LP is

$$\text{Minimize } w_1 = -3x_1 - 5x_2 - M$$

subject to

$$5x_1 + x_2 \le 12$$
$$x_1, x_2 \ge 0$$

The solution of this problem (by the simplex method) yields

$$\hat{\mathbf{X}}_{11} = (0, 12)^T, z_1^* - c_1^* = w_1^* = -60 - M$$

***Subproblem 2*** $(j = 2)$. The associated linear program is given as

$$\text{Minimize } z_2 - c_2 = \mathbf{C}_B\mathbf{B}^{-1}\begin{pmatrix}\mathbf{A}_2\mathbf{X}_2\\ 0\\ 1\end{pmatrix} - \mathbf{C}_2\mathbf{X}_2$$

$$= (0, -M, -M)\begin{pmatrix}(1,1)\begin{pmatrix}x_3\\ x_4\end{pmatrix}\\ 0\\ 1\end{pmatrix} - (1,\ 1)\begin{pmatrix}x_3\\ x_4\end{pmatrix}$$

$$= -x_3 - x_4 - M$$

subject to

$$x_3 + x_4 \ge 5$$
$$x_3 + 5x_4 \le 50$$
$$x_3, x_4 \ge 0$$

The optimal solution of the problem yields

$$\hat{\mathbf{X}}_{21} = (50, 0)^T, z_2^* - c_2^* = -50 - M$$

Because the master problem is of the maximization type and $z_1^* - c_1^* < z_2^* - c_2^*$ and $z_1^* - c_1^* < 0$, it follows that $\beta_{11}$ associated with extreme point $\hat{\mathbf{X}}_{11}$ must enter the solution. To determine the leaving variable,

$$\mathbf{P}_{11} = \begin{pmatrix}\mathbf{A}_1\hat{\mathbf{X}}_{11}\\ 1\\ 0\end{pmatrix} = \begin{pmatrix}(1,1)\begin{pmatrix}0\\ 12\end{pmatrix}\\ 1\\ 0\end{pmatrix} = \begin{pmatrix}12\\ 1\\ 0\end{pmatrix}$$

Thus, $\mathbf{B}^{-1}\mathbf{P}_{11} = (12, 1, 0)^T$. Given $\mathbf{X}_B = (x_5, x_6, x_7)^T = (40, 1, 1)^T$, it follows that $x_6$ (an artificial variable) leaves the basic solution (permanently).

The new basis is determined by replacing the vector associated with $x_6$ with the vector $\mathbf{P}_{11}$, which gives (verify!)

$$\mathbf{B} = \begin{pmatrix}1 & 12 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}$$

Thus,

$$\mathbf{B}^{-1} = \begin{pmatrix}1 & -12 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}$$

The new basic solution is

$$\mathbf{X}_B = (x_5, \beta_{11}, x_7)^T = \mathbf{B}^{-1}(40,1,1)^T = (28,1,1)^T$$
$$\mathbf{C}_B = (0, \mathbf{C}_1\hat{\mathbf{X}}_{11}, -M) = (0, 60, -M)$$

**Iteration 2.**

***Subproblem 1*** *(j = 1).* The associated objective function is

$$\text{Minimize } w_1 = -3x_1 - 5x_2 + 60$$

(verify!). The optimum solution yields $z_1^* - c_1^* = w_1 = 0$, which means that none of the remaining extreme points in subproblem 1 can improve the solution to the master problem.

***Subproblem 2*** *(j = 2).* The associated objective function is (coincidentally) the same as for $j = 2$ in Iteration 1 (verify!). The optimum solution yields

$$\hat{\mathbf{X}}_{22} = (50,0)^T, z_2^* - c_2^* = -50 - M$$

Note that $\hat{\mathbf{X}}_{22}$ is actually the same extreme point as $\hat{\mathbf{X}}_{21}$. We use the subscript 2 for notational convenience to represent Iteration 2.

From the results of the two subproblems, $z_2^* - c_2^* < 0$ indicates that $\beta_{22}$ associated with $\hat{\mathbf{X}}_{22}$ enters the basic solution.

To determine the leaving variable, consider

$$\mathbf{P}_{22} = \begin{pmatrix} \mathbf{A}_2\hat{\mathbf{X}}_{22} \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} (1,1)\begin{pmatrix} 50 \\ 0 \end{pmatrix} \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 50 \\ 0 \\ 1 \end{pmatrix}$$

Thus, $\mathbf{B}^{-1}\mathbf{P}_{22} = (50,0,1)^T$. Because $\mathbf{X}_B = (x_5, \beta_{11}, x_7)^T = (28,1,1)^T, x_5$ leaves.

The new basis and its inverse are given as (verify!)

$$\mathbf{B} = \begin{pmatrix} 50 & 12 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

$$\mathbf{B}^{-1} = \begin{pmatrix} \frac{1}{50} & -\frac{12}{50} & 0 \\ 0 & 1 & 0 \\ -\frac{1}{50} & \frac{12}{50} & 1 \end{pmatrix}$$

$$\mathbf{X}_B = (\beta_{22}, \beta_{11}, x_7)^T = \mathbf{B}^{-1}(40,1,1)^T = (\tfrac{14}{25}, 1, \tfrac{11}{25})^T$$
$$\mathbf{C}_B = (\mathbf{C}_2\hat{\mathbf{X}}_{22}, \mathbf{C}_1\hat{\mathbf{X}}_{11}, -M) = (50, 60, -M)$$

**Iteration 3.**

***Subproblem 1*** *(j = 1).* You should verify that the associated objective function is

$$\text{Minimize } w_1 = (\tfrac{M}{50} - 2)x_1 + (\tfrac{M}{50} - 4)x_2 - \tfrac{12M}{50} + 48$$

The associated optimum solution is

$$\hat{\mathbf{X}}_{13} = (0,0)^T, z_1^* - c_1^* = -\tfrac{12M}{50} + 48$$

***Subproblem 2*** ($j = 2$). The objective function can be shown to equal (verify!)

$$\text{Minimize } w_2 = \left(\tfrac{M}{50}\right)(x_3 + x_4) - M$$

The associated optimum solution is

$$\hat{\mathbf{X}}_{23} = (5,0)^T, z_2^* - c_2^* = -\tfrac{9M}{10}$$

***Nonbasic Variable*** $x_5$. From the definition of the master problem, $z_j - c_j$ of $x_5$ must be computed and compared separately. Thus,

$$\begin{aligned} z_5 - c_5 &= \mathbf{C}_B\mathbf{B}^{-1}\mathbf{P}_5 - c_5 \\ &= \left(1 + \tfrac{M}{50}, 48 - \tfrac{12M}{50}, -M\right)(1,0,0)^T - 0 \\ &= 1 + \tfrac{M}{50} \end{aligned}$$

Thus, $x_5$ cannot improve the solution.

From the preceding information, $\beta_{23}$ associated with $\hat{\mathbf{X}}_{23}$ enters the basic solution. To determine the leaving variable, consider

$$\mathbf{P}_{23} = \begin{pmatrix} \mathbf{A}_2\hat{\mathbf{X}}_{23} \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} (1,1)\begin{pmatrix} 5 \\ 0 \end{pmatrix} \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 5 \\ 0 \\ 1 \end{pmatrix}$$

Thus, $\mathbf{B}^{-1}\mathbf{P}_{23} = \left(\tfrac{1}{10}, 0, \tfrac{9}{10}\right)^T$. Given $\mathbf{X}_B = (\beta_{22}, \beta_{11}, x_7)^T = \left(\tfrac{14}{25}, 1, \tfrac{11}{25}\right)^T$, the artificial variable $x_7$ leaves the basic solution (permanently).

The new basis and its inverse are thus given as (verify!)

$$\mathbf{B} = \begin{pmatrix} 50 & 12 & 5 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

$$\mathbf{B}^{-1} = \begin{pmatrix} \tfrac{1}{45} & -\tfrac{12}{45} & -\tfrac{5}{45} \\ 0 & 1 & 0 \\ -\tfrac{1}{45} & \tfrac{12}{45} & \tfrac{50}{45} \end{pmatrix}$$

$$\mathbf{X}_B = (\beta_{22}, \beta_{11}, \beta_{23})^T = \mathbf{B}^{-1}(40,1,1)^T = \left(\tfrac{23}{45}, 1, \tfrac{22}{45}\right)^T$$

$$\mathbf{C}_B = (\mathbf{C}_2\hat{\mathbf{X}}_{22}, \mathbf{C}_1\hat{\mathbf{X}}_{11}, \mathbf{C}_2\hat{\mathbf{X}}_{23}) = (50, 60, 5)$$

**Iteration 4.**

***Subproblem 1*** ($j = 1$). $w_1 = -2x_1 - 4x_2 + 48$. It yields $z_1^* - c_1^* = w_1^* = 0$.

***Subproblem 2*** ($j = 2$). $w_2 = 0x_3 + 0x_4 + 48 = 48$.

***Nonbasic Variable*** $x_5 : z_5 - c_5 = 1$. The preceding information shows that Iteration 3 is optimal.

We can compute the optimum solution of the original problem by back-substitution:

$$\mathbf{X}_1^* = (x_1, x_2)^T = \beta_{11}\hat{\mathbf{X}}_{11} = 1(0, 12)^T = (0, 12)^T$$

$$\mathbf{X}_2^* = (x_3, x_4)^T = \beta_{22}\hat{\mathbf{X}}_{22} + \beta_{23}\hat{\mathbf{X}}_{23}$$

$$= \left(\tfrac{23}{45}\right)(50, 0)^T + \left(\tfrac{22}{45}\right)(5, 0)^T$$

$$= (28, 0)^T$$

The optimum value of the objective function can be obtained by direct substitution.

## PROBLEM SET 7.4A

1. In each of the following cases, determine the feasible extreme points graphically and express the feasible solution space as a function of these extreme points. If the solution space is unbounded, add a proper artificial constraint.

**(a)**

$$\begin{aligned} x_1 + 2x_2 &\le 6 \\ 2x_1 + x_2 &\le 8 \\ -x_1 + x_2 &\le 1 \\ x_2 &\le 2 \\ x_1, x_2 &\le 0 \end{aligned}$$

**(b)**

$$\begin{aligned} 2x_1 + x_2 &\le 2 \\ 3x_1 + 4x_2 &\ge 12 \\ x_1, x_2 &\ge 0 \end{aligned}$$

**(c)**

$$\begin{aligned} x_1 - x_2 &\le 10 \\ 2x_1 \quad &\le 40 \\ x_1, x_2 &\ge 0 \end{aligned}$$

2. In Example 7.4-1, the feasible extreme points of subspaces $\mathbf{D}_1\mathbf{X}_1 = \mathbf{b}_1, \mathbf{X}_1 \ge \mathbf{0}$ and $\mathbf{D}_2\mathbf{X}_2 = \mathbf{b}_2, \mathbf{X}_2 \ge \mathbf{0}$ can be determined graphically. Use this information to express the associated master problem explicitly. Then show that the application of the simplex method to the master problem produces the same entering variable $\beta_{jk}$ as that generated by solving subproblems 1 and 2. Hence, convince yourself that the determination of the entering variable $\beta_{jk}$ is exactly equivalent to solving the two minimization subproblems.
3. Consider the following linear program:

$$\text{Maximize } z = x_1 + 3x_2 + 5x_3 + 2x_4$$

subject to

$$\begin{aligned} x_1 + 4x_2 \qquad\quad &\le 8 \\ 2x_1 + x_2 \qquad\quad &\le 9 \\ 5x_1 + 3x_2 + 4x_3 &\ge 10 \end{aligned}$$

$$x_3 - 5x_4 \le 4$$
$$x_3 + x_4 \le 10$$
$$x_1, x_2, x_3, x_4 \ge 0$$

Construct the master problem explicitly by using the extreme points of the subspaces, and then solve the resulting problem directly by the simplex method.

**4.** Solve Problem 3 using the decomposition algorithm and compare the two procedures.

**5.** Apply the decomposition algorithm to the following problem:

$$\text{Maximize } z = 6x_1 + 7x_2 + 3x_3 + 5x_4 + x_5 + x_6$$

subject to

$$\begin{aligned}
x_1 + x_2 + x_3 + x_4 + x_5 + x_6 &\le 50 \\
x_1 + x_2 &\le 10 \\
x_2 &\le 8 \\
5x_3 + x_4 &\le 12 \\
x_5 + x_6 &\ge 5 \\
x_5 + 5x_6 &\le 50 \\
x_1, x_2, x_3, x_4, x_5, x_6 &\ge 0
\end{aligned}$$

**6.** Indicate the necessary changes for applying the decomposition algorithm to minimization LPs. Then solve the following problem:

$$\text{Minimize } z = 5x_1 + 3x_2 + 8x_3 - 5x_4$$

subject to

$$\begin{aligned}
x_1 + x_3 + x_3 + x_4 &\ge 25 \\
5x_1 + x_2 &\le 20 \\
5x_1 - x_2 &\ge 5 \\
x_3 + x_4 &= 20 \\
x_1, x_2, x_3, x_4 &\ge 0
\end{aligned}$$

**7.** Solve the following problem by the decomposition algorithm:

$$\text{Minimize } z = 10y_1 + 2y_2 + 4y_3 + 8y_4 + y_5$$

subject to

$$\begin{aligned}
y_1 + 4y_2 - y_3 &\ge 8 \\
2y_1 + y_2 + y_3 &\ge 2 \\
3y_1 + y_4 + y_5 &\ge 4 \\
y_1 + 2y_4 - y_5 &\ge 10 \\
y_1, y_2, y_3, y_4, y_5 &\ge 0
\end{aligned}$$

(*Hint:* Solve the dual problem first by decomposition.)

8. In the decomposition algorithm, suppose that the number of common constraints in the original problem is $r$. Show that the objective function for subproblem $j$ can be written as

$$\text{Minimize } w_j = z_j - c_j = (\mathbf{C}_B\mathbf{R}\mathbf{A}_j - \mathbf{C}_j)\mathbf{X}_j + \mathbf{C}_B\mathbf{V}_{r+j}$$

The vector $\mathbf{R}$ represents the first $r$ columns of $\mathbf{B}^{-1}$ and $\mathbf{V}_{r+j}$ is its $(r + j)$ th column.

## 7.5 DUALITY

We have dealt with the dual problem at an elementary level in Chapter 4. This section presents a more rigorous treatment of duality and allows us to verify the primal-dual relationships that formed the basis for sensitivity analysis in Chapter 4. The presentation also lays the foundation for the development of parametric programming.

### 7.5.1 Matrix Definition of the Dual Problem

Suppose that the primal problem in equation form with $m$ constraints and $n$ variables is defined as

$$\text{Maximize } z = \mathbf{CX}$$

subject to

$$\mathbf{AX} = \mathbf{b}$$

$$\mathbf{X} \geq \mathbf{0}$$

Letting the vector $\mathbf{Y} = (y_1, y_2, \ldots, y_m)$ represent the dual variables, the rules in Table 4.2 produce the following dual problem:

$$\text{Minimize } w = \mathbf{Yb}$$

subject to

$$\mathbf{YA} \geq \mathbf{C}$$

$$\mathbf{Y} \text{ unrestricted}$$

Note that some of the constraints $\mathbf{YA} \geq \mathbf{C}$ may override unrestricted $\mathbf{Y}$.

#### PROBLEM SET 7.5A

1. Prove that the dual of the dual is the primal.
2. Suppose that the primal is given as $\min z = \{\mathbf{CX} | \mathbf{AX} \geq \mathbf{b}, \mathbf{X} \geq \mathbf{0}\}$. Define the corresponding dual problem.

### 7.5.2 Optimal Dual Solution

This section establishes relationships between the primal and dual problems and shows how the optimal dual solution can be determined from the optimal primal solution. Let $\mathbf{B}$ be the current *optimal* primal basis, and define $\mathbf{C}_B$ as the objective function coefficients associated with the optimal vector $\mathbf{X}_B$.

**Theorem 7.5-1. (Weak Duality Theory).** *For any pair of* feasible *primal and dual solutions* $(\mathbf{X}, \mathbf{Y})$, *the value of the objective function in the* minimization *problem sets an upper bound on the value of the objective function in the* maximization *problem. For the* optimal *pair* $(\mathbf{X}^*, \mathbf{Y}^*)$, *the values of the objective functions in the two problems are equal.*

***Proof.*** The feasible pair $(\mathbf{X}, \mathbf{Y})$ satisfies all the restrictions of the two problems. Premultiplying both sides of the constraints of the maximization problem with (unrestricted) **Y,** we get

$$\mathbf{YAX} = \mathbf{Yb} = w \tag{1}$$

Also, for the minimization problem, postmultiplying both sides by $\mathbf{X}(\geq \mathbf{0})$, we get

$$\mathbf{YAX} \geq \mathbf{CX}$$

or

$$\mathbf{YAX} \geq \mathbf{CX} = z \tag{2}$$

(The nonnegativity of the vector **X** is essential for maintaining the direction of the inequality.) Combining (1) and (2), we get $z \leq w$ for any *feasible* pair $(\mathbf{X}, \mathbf{Y})$.

Note that the theorem does *not* depend on labeling the problems as primal or dual. What is important is the sense of optimization in each problem. Specifically, for any pair of feasible solutions, the objective value in the maximization problem does not exceed the objective value in the minimization problem.

The implication of the theorem is that, given $z \leq w$ for any feasible solutions, the maximum of $z$ and the minimum of $w$ are achieved when the two objective values are equal. A consequence of this result is that the "goodness" of any feasible primal and dual solutions relative to the optimum may be checked by comparing the difference $(w - z)$ to $\frac{z + w}{2}$. The smaller the ratio $\frac{2(w - z)}{z + w}$, the closer the two solutions are to being optimal. The suggested *rule of thumb* does *not* imply that the optimal objective value is $\frac{z + w}{2}$.

What happens if one of the two problems has an unbounded objective value? The answer is that the other problem must be infeasible. For if it is not, then both problems have feasible solutions, and the relationship $z \leq w$ must hold—an impossible result because either $z = +\infty$ or $w = -\infty$ by assumption.

The next question is: If one problem is infeasible, is the other problem unbounded? Not necessarily. The following counterexample shows that both the primal and the dual can be infeasible (verify graphically!):

***Primal.*** Maximize $z = \{x_1 + x_2 \mid x_1 - x_2 \leq -1, -x_1 + x_2 \leq -1, x_1, x_2 \geq 0\}$

***Dual.*** Minimize $w = \{-y_1 - y_2 \mid y_1 - y_2 \geq 1, -y_1 + y_2 \geq 1, y_1, y_2 \geq 0\}$

**Theorem 7.5-2.** *Given the* optimal *primal basis* **B** *and its associated objective coefficient vector* $\mathbf{C}_B$, *the optimal solution of the dual problem is*

$$\mathbf{Y} = \mathbf{C}_B\mathbf{B}^{-1}$$

***Proof.*** The proof rests on verifying two points: $\mathbf{Y} = \mathbf{C}_B\mathbf{B}^{-1}$ is a feasible dual solution and $z = w$ per Theorem 7.5-1.

The feasibility of $\mathbf{Y} = \mathbf{C}_B\mathbf{B}^{-1}$ is guaranteed by the optimality of the primal, $z_j - c_j \geq 0$ for all $j$—that is,

$$\mathbf{C}_B\mathbf{B}^{-1}\mathbf{A} - \mathbf{C} \geq \mathbf{0}$$

(See Section 7.2.1.) Thus, $\mathbf{YA} - \mathbf{C} \geq \mathbf{0}$ or $\mathbf{YA} \geq \mathbf{C}$, which shows that $\mathbf{Y} = \mathbf{C}_B\mathbf{B}^{-1}$ is a feasible dual solution.

Next, we show that the associated $w = z$ by noting that

$$w = \mathbf{Yb} = \mathbf{C}_B\mathbf{B}^{-1}\mathbf{b} \qquad (1)$$

Similarly, given the primal solution $\mathbf{X}_B = \mathbf{B}^{-1}\mathbf{b}$, we get

$$z = \mathbf{C}_B\mathbf{X}_B = \mathbf{C}_B\mathbf{B}^{-1}\mathbf{b} \qquad (2)$$

From relations (1) and (2), we conclude $z = w$.

The dual variables $\mathbf{Y} = \mathbf{C}_B\mathbf{B}^{-1}$ are sometimes referred to as the *simplex multipliers.* They are also known as the *shadow prices,* a name that evolved from the economic interpretation of the dual variables (see Section 4.3.1).

Given $\mathbf{P}_j$ is the $j$th column of $\mathbf{A}$, we note from Theorem 7.5-2 that

$$z_j - c_j = \mathbf{C}_B\mathbf{B}^{-1}\mathbf{P}_j - c_j = \mathbf{YP}_j - c_j$$

represents the difference between the left- and right-hand sides of the dual constraints. The maximization primal starts with $z_j - c_j < 0$ for at least one $j$, which means that the corresponding dual constraint, $\mathbf{YP}_j \geq c_j$, is not satisfied. When the primal optimal is reached, we get $z_j - c_j \geq 0$, for all $j$, which means that the corresponding dual solution $\mathbf{Y} = \mathbf{C}_B\mathbf{B}^{-1}$ becomes feasible. We conclude that while the primal is seeking optimality, the dual is automatically seeking feasibility. This point is the basis for the development of the *dual simplex method* (Section 4.4) in which the iterations start better than optimal and infeasible and remain so until feasibility is acquired at the last iteration. This is in contrast with the (primal) simplex method (Chapter 3), which remains worse than optimal but feasible until the optimal iteration is reached.

### Example 7.5-1

The *optimal* basis for the following LP is $\mathbf{B} = (\mathbf{P}_1, \mathbf{P}_4)$. Write the dual and find its optimum solution using the optimal primal basis.

$$\text{Maximize } z = 3x_1 + 5x_2$$

subject to

$$\begin{aligned} x_1 + 2x_2 + x_3 \qquad &= 5 \\ -x_1 + 3x_2 \qquad + x_4 &= 2 \\ x_1, x_2, x_3, x_4 &\geq 0 \end{aligned}$$

The dual problem is given as

$$\text{Minimize } w = 5y_1 + 2y_2$$

subject to

$$
\begin{aligned}
y_1 - \quad y_2 &\geq 3 \\
2y_1 + 3y_2 &\geq 5 \\
y_1, y_2 &\geq 0
\end{aligned}
$$

We have $\mathbf{X}_B = (x_1, x_4)^T$; it follows that $\mathbf{C}_B = (3, 0)$. The optimal basis and its inverse are given as

$$\mathbf{B} = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix} \text{ and } \mathbf{B}^{-1} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

The associated primal and dual values are

$$
\begin{aligned}
(x_1, x_4)^T &= \mathbf{B}^{-1}\mathbf{b} = (5, 7)^T \\
(y_1, y_2) &= \mathbf{C}_B\mathbf{B}^{-1} = (3, 0)
\end{aligned}
$$

Both solutions are feasible and $z = w = 15$ (verify!). Thus, the two solutions are optimal.

---

## PROBLEM SET 7.5B

1. Verify that the dual problem of the numeric example given at the end of Theorem 7.5-1 is correct. Then verify graphically that both the primal and dual problems have no feasible solution.
2. Consider the following LP:

$$\text{Maximize } z = 50x_1 + 30x_2 + 10x_3$$

subject to

$$
\begin{aligned}
2x_1 + x_2 \qquad &= 1 \\
2x_2 \qquad &= -5 \\
4x_1 \qquad + x_3 &= 6 \\
x_1, x_2, x_3 &\geq 0
\end{aligned}
$$

   **(a)** Write the dual.
   **(b)** Show by inspection that the primal is infeasible.
   **(c)** Show that the dual in (a) is unbounded.
   **(d)** From Problems 1 and 2, develop a general conclusion regarding the relationship between infeasibility and unboundedness in the primal and dual problems.

3. Consider the following LP:

$$\text{Maximize } z = 5x_1 + 12x_2 + 4x_3$$

subject to

$$
\begin{aligned}
2x_1 - x_2 + 3x_3 \qquad &= 2 \\
x_1 + 2x_2 + x_3 + x_4 &= 5 \\
x_1, x_2, x_3, x_4 &\geq 0
\end{aligned}
$$

**(a)** Write the dual.

**(b)** In each of the following cases, first verify that the given basis **B** is feasible for the primal. Next, using $\mathbf{Y} = \mathbf{C}_B\mathbf{B}^{-1}$, compute the associated dual values and verify whether or not the primal solution is optimal.

**(i)** $\mathbf{B} = (\mathbf{P}_4, \mathbf{P}_3)$ **(iii)** $\mathbf{B} = (\mathbf{P}_1, \mathbf{P}_2)$

**(ii)** $\mathbf{B} = (\mathbf{P}_2, \mathbf{P}_3)$ **(iv)** $\mathbf{B} = (\mathbf{P}_1, \mathbf{P}_4)$

**4.** Consider the following LP:

$$\text{Maximize } z = 2x_1 + 4x_2 + 4x_3 - 3x_4$$

subject to

$$x_1 + x_2 + x_3 = 4$$
$$x_1 + 4x_2 + \quad + x_4 = 8$$
$$x_1, x_2, x_3, x_4 \geq 0$$

**(a)** Write the dual problem.

**(b)** Verify that $\mathbf{B} = (\mathbf{P}_2, \mathbf{P}_3)$ is optimal by computing $z_j - c_j$ for all nonbasic $\mathbf{P}_j$.

**(c)** Find the associated optimal dual solution.

**5.** An LP model includes two variables $x_1$ and $x_2$ and three constraints of the type $\leq$. The associated slacks are $x_3, x_4$, and $x_5$. Suppose that the optimal basis is $\mathbf{B} = (\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3)$, and its inverse is

$$\mathbf{B}^{-1} = \begin{pmatrix} 0 & -1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & -1 \end{pmatrix}$$

The optimal primal and dual solutions are given as

$$\mathbf{X}_B = (x_1, x_2, x_3)^T = (2, 6, 2)^T$$
$$\mathbf{Y} = (y_1, y_2, y_3) = (0, 3, 2)$$

Determine the optimal value of the objective function in two ways using the primal and dual problems.

**6.** Prove the following relationship for the optimal primal and dual solutions:

$$\Sigma_{i=1}^{m} c_i(\mathbf{B}^{-1}\mathbf{P}_k)_i = \Sigma_{i=1}^{m} y_i a_{ik}$$

where $\mathbf{C}_B = (c_1, c_2, \ldots, c_m)$ and $\mathbf{P}_k = (a_{1k}, a_{2k}, \ldots, a_{mk})^T$, for $k = 1, 2, \ldots, n$, and $(\mathbf{B}^{-1}\mathbf{P}_k)_i$ is the $i$th element of $\mathbf{B}^{-1}\mathbf{P}_k$.

**7.** Write the dual of

$$\text{Maximimize } z = \{\mathbf{CX} | \mathbf{AX} = \mathbf{b}, \mathbf{X} \text{ unrestricted}\}$$

**8.** Show that the dual of

$$\text{Maximize } z = \{\mathbf{CX} | \mathbf{AX} \leq \mathbf{b}, \mathbf{0} < \mathbf{L} \leq \mathbf{X} \leq \mathbf{U}\}$$

always possesses a feasible solution.

## 7.6 PARAMETRIC LINEAR PROGRAMMING

Parametric linear programming is an extension of the sensitivity analysis procedures presented in Section 4.5. It investigates the effect of *predetermined* continuous variations in the objective function coefficients and the right-hand side of the constraints on the optimum solution.

Suppose that the LP is defined as

$$\text{Maximize } z = \left\{ \mathbf{CX} \,\middle|\, \sum_{j=1}^{n} \mathbf{P}_j x_j = \mathbf{b}, \mathbf{X} \geq \mathbf{0} \right\}$$

In parametric analysis, the objective function and right-hand side vectors, $\mathbf{C}$ and $\mathbf{b}$, are replaced with the parameterized functions $\mathbf{C}(t)$ and $\mathbf{b}(t)$, where $t$ is the parameter of variation. Mathematically, $t$ can assume any positive or negative value. In practice, however, $t$ usually represents time, and hence it assumes nonnegative values only. In this presentation we will assume $t \geq 0$.

The general idea of parametric analysis is to start with the optimal solution at $t = 0$. Then, using the optimality and feasibility conditions of the simplex method, we determine the range $0 \leq t \leq t_1$ for which the solution at $t = 0$ remains optimal and feasible. In this case, $t_1$ is referred to as a **critical value.** The process continues by determining successive critical values and their corresponding optimal feasible solutions. The process will terminate at $t = t_r$ when there is indication that either the last solution remains unchanged for $t > t_r$ or that no feasible solution exists beyond that critical value.

### 7.6.1 Parametric Changes in C

Let $\mathbf{X}_{B_i}, \mathbf{B}_i, \mathbf{C}_{B_i}(t)$ be the elements that define the optimal solution associated with critical $t_i$ (the computations start at $t_0 = 0$ with $\mathbf{B}_0$ as its optimal basis). Next, the critical value $t_{i+1}$ and its optimal basis, if one exists, is determined. Because changes in $\mathbf{C}$ can only affect the optimality of the problem, the current solution $\mathbf{X}_{B_i} = \mathbf{B}_i^{-1}\mathbf{b}$ will remain optimal for some $t \geq t_i$ so long as the following optimality condition is satisfied:

$$z_j(t) - c_j(t) = \mathbf{C}_{B_i}(t)\mathbf{B}_i^{-1}\mathbf{P}_j - c_j(t) \geq 0, \text{ for all } j$$

The value of $t_{i+1}$ equals the largest $t > t_i$ that satisfies all the optimality conditions.

Note that *nothing* in the inequalities requires $\mathbf{C}(t)$ to be linear in $t$. Any function $\mathbf{C}(t)$, linear or nonlinear, is acceptable. However, with nonlinearity the numerical manipulation of the resulting inequalities may be cumbersome. (See Problem 5, Set 7.6a for an illustration of the nonlinear case.)

**Example 7.6-1**

$$\text{Maximize } z = (3 - 6t)x_1 + (2 - 2t)x_2 + (5 + 5t)x_3$$

subject to

$$\begin{aligned} x_1 + 2x_2 + x_3 &\leq 40 \\ 3x_1 \phantom{+ 2x_2} + 2x_3 &\leq 60 \\ x_1 + 4x_2 \phantom{+ 2x_3} &\leq 30 \\ x_1, x_2, x_3 &\geq 0 \end{aligned}$$

We have

$$\mathbf{C}(t) = (3 - 6t, 2 - 2t, 5 + 5t), t \geq 0$$

The variables $x_4, x_5,$ and $x_6$ are slacks.

**Optimal Solution at $t = t_0 = 0$**

| Basic | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ | Solution |
|---|---|---|---|---|---|---|---|
| $z$ | 4 | 0 | 0 | 1 | 2 | 0 | 160 |
| $x_2$ | $-\frac{1}{4}$ | 1 | 0 | $\frac{1}{2}$ | $-\frac{1}{4}$ | 0 | 5 |
| $x_3$ | $\frac{3}{2}$ | 0 | 1 | 0 | $\frac{1}{2}$ | 0 | 30 |
| $x_6$ | 2 | 0 | 0 | $-2$ | 1 | 1 | 10 |

$$\mathbf{X}_{B_0} = (x_2, x_3, x_6)^T = (5, 30, 10)^T$$

$$\mathbf{C}_{B_0}(t) = (2 - 2t, 5 + 5t, 0)$$

$$\mathbf{B}_0^{-1} = \begin{pmatrix} \frac{1}{2} & -\frac{1}{4} & 0 \\ 0 & \frac{1}{2} & 0 \\ -2 & 1 & 1 \end{pmatrix}$$

The optimality conditions for the current nonbasic vectors $\mathbf{P}_1, \mathbf{P}_4$, and $\mathbf{P}_5$ are

$$\{\mathbf{C}_{B_0}(t)\mathbf{B}_0^{-1}\mathbf{P}_j - c_j(t)\}_{j=1,4,5} = (4 + 14t, 1 - t, 2 + 3t) \geq \mathbf{0}$$

Thus, $\mathbf{X}_{B_0}$ remains optimal so long as the following conditions are satisfied:

$$4 + 14t \geq 0$$

$$1 - t \geq 0$$

$$2 + 3t \geq 0$$

Because $t \geq 0$, the second inequality stipulates that $t \leq 1$ and the remaining two inequalities are satisfied for all $t \geq 0$. We thus have $t_1 = 1$, which means that $\mathbf{X}_{B_0}$ remains optimal (and feasible) for $0 \leq t \leq 1$.

At $t = 1$, $z_4(t) - c_4(t) = 1 - t$ equals zero and becomes negative for $t > 1$. Thus, $\mathbf{P}_4$ must enter the basis for $t > 1$. In this case, $\mathbf{P}_2$ must leave the basis (see the optimal tableau at $t = 0$). The new basic solution $\mathbf{X}_{B_1}$ is the alternative solution obtained at $t = 1$ by letting $\mathbf{P}_4$ enter the basis—that is, $\mathbf{X}_{B_1} = (x_4, x_3, x_6)^T$ and $\mathbf{B}_1 = (\mathbf{P}_4, \mathbf{P}_3, \mathbf{P}_6)$.

**Alternative Optimal Basis at $t = t_1 = 1$**

$$\mathbf{B}_1 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{B}_1^{-1} = \begin{pmatrix} 1 & -\frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Thus,

$$\mathbf{X}_{B_1} = (x_4, x_3, x_6)^T = \mathbf{B}_1^{-1}\mathbf{b} = (10, 30, 30)^T$$

$$\mathbf{C}_{B_1}(t) = (0, 5 + 5t, 0)$$

The associated nonbasic vectors are $\mathbf{P}_1, \mathbf{P}_2$, and $\mathbf{P}_5$, and we have

$$\{\mathbf{C}_{B_1}(t)\mathbf{B}_1^{-1}\mathbf{P}_j - c_j(t)\}_{j=1,2,5} = \left(\tfrac{9 + 27t}{2}, -2 + 2t, \tfrac{5 + 5t}{2}\right) \geq \mathbf{0}$$

According to these conditions, the basic solution $\mathbf{X}_{B_1}$ remains optimal for all $t \geq 1$. Observe that the optimality condition, $-2 + 2t \geq 0$, automatically "remembers" that

$\mathbf{X}_{B_1}$ is optimal for a range of $t$ that starts from the last critical value $t_1 = 1$. This will always be the case in parametric programming computations.

The optimal solution for the entire range of $t$ is summarized below. The value of $z$ is computed by direct substitution.

| $t$ | $x_1$ | $x_2$ | $x_3$ | $z$ |
|---|---|---|---|---|
| $0 \leq t \leq 1$ | 0 | 5 | 30 | $160 + 140t$ |
| $t \geq 1$ | 0 | 0 | 30 | $150 + 150t$ |

## PROBLEM SET 7.6A

1. In Example 7.6-1, suppose that $t$ is unrestricted in sign. Determine the range of $t$ for which $\mathbf{X}_{B_0}$ remains optimal.
2. Solve Example 7.6-1, assuming that the objective function is given as
   - **(a)** Maximize $z = (3 + 3t)x_1 + 2x_2 + (5 - 6t)x_3$
   - **(b)** Maximize $z = (3 - 2t)x_1 + (2 + t)x_2 + (5 + 2t)x_3$
   - **(c)** Maximize $z = (3 + t)x_1 + (2 + 2t)x_2 + (5 - t)x_3$
3. Study the variation in the optimal solution of the following parameterized LP given $t \geq 0$.

$$\text{Minimize } z = (4 - t)x_1 + (1 - 3t)x_2 + (2 - 2t)x_3$$

subject to

$$3x_1 + x_2 + 2x_3 = 3$$
$$4x_1 + 3x_2 + 2x_3 \geq 6$$
$$x_1 + 2x_2 + 5x_3 \leq 4$$
$$x_1, x_2, x_3 \geq 0$$

4. The analysis in this section assumes that the optimal solution of the LP at $t = 0$ is obtained by the (primal) simplex method. In some problems, it may be more convenient to obtain the optimal solution by the dual simplex method (Section 4.4). Show how the parametric analysis can be carried out in this case, and then analyze the LP of Example 4.4-1, assuming that the objective function is given as

$$\text{Minimize } z = (3 + t)x_1 + (2 + 4t)x_2, \; t \geq 0$$

5. In Example 7.6-1, suppose that the objective function is nonlinear in $t (t \geq 0)$ and is defined as

$$\text{Maximize } z = (3 + 2t^2)x_1 + (2 - 2t^2)x_2 + (5 - t)x_3$$

Determine the first critical value $t_1$.

### 7.6.2 Parametric Changes in b

The parameterized right-hand side $\mathbf{b}(t)$ can only affect the feasibility of the problem. The critical values of $t$ are thus determined from the following condition:

$$\mathbf{X}_B(t) = \mathbf{B}^{-1}\mathbf{b}(t) \geq \mathbf{0}$$

**Example 7.6-2**

$$\text{Maximize } z = 3x_1 + 2x_2 + 5x_3$$

subject to

$$\begin{aligned} x_1 + 2x_2 + x_3 &\le 40 - t \\ 3x_1 \quad + 2x_3 &\le 60 + 2t \\ x_1 + 4x_2 \quad &\le 30 - 7t \\ x_1, x_2, x_3 &\ge 0 \end{aligned}$$

Assume that $t \ge 0$.

At $t = t_0 = 0$, the problem is identical with that in Example 7.6-1. We thus have

$$\mathbf{X}_{B_0} = (x_2, x_3, x_6)^T = (5, 30, 10)^T$$

$$\mathbf{B}_0^{-1} = \begin{pmatrix} \frac{1}{2} & -\frac{1}{4} & 0 \\ 0 & \frac{1}{2} & 0 \\ -2 & 1 & 1 \end{pmatrix}$$

To determine the first critical value $t_1$, we apply the condition $\mathbf{X}_{B_0}(t) = \mathbf{B}_0^{-1}\mathbf{b}(t) \ge 0$ which yields

$$\begin{pmatrix} x_2 \\ x_3 \\ x_6 \end{pmatrix} = \begin{pmatrix} 5 - t \\ 30 + t \\ 10 - 3t \end{pmatrix} \ge \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

These inequalities are satisfied for $t \le \frac{10}{3}$, meaning that $t_1 = \frac{10}{3}$ and that the basis $\mathbf{B}_0$ remains feasible for the range $0 \le t \le \frac{10}{3}$. However, the values of the basic variables $x_2, x_3$, and $x_6$ will change with $t$ as given above.

The value of the basic variable $x_6$ $(=10 - 3t)$ will equal zero at $t = t_1 = \frac{10}{3}$ and will become negative for $t > \frac{10}{3}$. Thus, at $t = \frac{10}{3}$, we can determine the alternative basis $\mathbf{B}_1$ by applying the revised dual simplex method (see Problem 5, Set 7.2b for details). The leaving variable is $x_6$.

**Alternative Basis at $t = t_1 = \frac{10}{3}$**

Given $x_6$ is the leaving variable, we determine the entering variable as follows:

$$\mathbf{X}_{B_0} = (x_2, x_3, x_6)^T, \mathbf{C}_{B_0} = (2, 5, 0)$$

Thus,

$$\{z_j - c_j\}_{j=1,4,5} = \{\mathbf{C}_{B_0}\mathbf{B}_0^{-1}\mathbf{P}_j - c_j\}_{j=1,4,5} = (4, 1, 2)$$

Next, for nonbasic $x_j, j = 1, 4, 5$, we compute

$$\begin{aligned} (\text{Row of } \mathbf{B}_0^{-1} \text{ associated with } x_6)(\mathbf{P}_1, \mathbf{P}_4, \mathbf{P}_5) &= (\text{Third row of } \mathbf{B}_0^{-1})(\mathbf{P}_1, \mathbf{P}_4, \mathbf{P}_5) \\ &= (-2, 1, 1)(\mathbf{P}_1, \mathbf{P}_4, \mathbf{P}_5) \\ &= (2, -2, 1) \end{aligned}$$

The entering variable is thus associated with

$$\theta = \min\left\{-, \left|\frac{1}{-2}\right|, -\right\} = \frac{1}{2}$$

Thus, $\mathbf{P}_4$ is the entering vector.

The alternative basis is

$$\mathbf{B}_1 = (\mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4) = \begin{pmatrix} 2 & 1 & 1 \\ 0 & 2 & 0 \\ 4 & 0 & 0 \end{pmatrix}$$

Thus,

$$\mathbf{B}_1^{-1} = \begin{pmatrix} 0 & 0 & \frac{1}{4} \\ 0 & \frac{1}{2} & 0 \\ 1 & -\frac{1}{2} & -\frac{1}{2} \end{pmatrix}$$

The new $\mathbf{X}_{B_1} = (x_2, x_3, x_4)^T$.

The next critical value $t_2$ is determined from the condition $\mathbf{X}_{B_1}(t) = \mathbf{B}_1^{-1}\mathbf{b}(t) \geq \mathbf{0}$, which yields

$$\begin{pmatrix} x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} \frac{30 - 7t}{4} \\ 30 + t \\ \frac{-10 + 3t}{2} \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

These conditions show that $\mathbf{B}_1$ remains feasible for $\frac{10}{3} \leq t \leq \frac{30}{7}$.

At $t = t_2 = \frac{30}{7}$, an alternative basis can be obtained by the revised dual simplex method. The leaving variable is $x_2$ because it corresponds to the condition yielding the critical value $t_2$.

**Alternative Basis at $t = t_2 = \frac{30}{7}$.**
Given $x_2$ is the leaving variable, we determine the entering variable as follows:

$$\mathbf{X}_{B_1} = (x_2, x_3, x_4)^T, \mathbf{C}_{B_1} = (2, 5, 0)$$

Thus,

$$\{z_j - c_j\}_{j=1,5,6} = \{\mathbf{C}_{B_1}\mathbf{B}_1^{-1}\mathbf{P}_j - c_j\}_{j=1,5,6} = (5, \tfrac{5}{2}, \tfrac{1}{2})$$

Next, for nonbasic $x_j$, $j = 1, 5, 6$, we compute

$$\begin{aligned}(\text{Row of } \mathbf{B}_1^{-1} \text{associated with } x_2)(\mathbf{P}_1, \mathbf{P}_5, \mathbf{P}_6) &= (\text{First row of } \mathbf{B}_1^{-1})(\mathbf{P}_1, \mathbf{P}_5, \mathbf{P}_6) \\ &= (0, 0, \tfrac{1}{4})(\mathbf{P}_1, \mathbf{P}_5, \mathbf{P}_6) \\ &= (\tfrac{1}{4}, 0, \tfrac{1}{4})\end{aligned}$$

Because all the denominator elements, $(\frac{1}{4}, 0, \frac{1}{4})$, are $\geq 0$, the problem has no feasible solution for $t > \frac{30}{7}$ and the parametric analysis ends at $t = t_2 = \frac{30}{7}$.

The optimal solution is summarized as

| $t$ | $x_1$ | $x_2$ | $x_3$ | $z$ |
|---|---|---|---|---|
| $0 \leq t \leq \frac{10}{3}$ | 0 | $5 - t$ | $30 + t$ | $160 + 3t$ |
| $\frac{10}{3} \leq t \leq \frac{30}{7}$ | 0 | $\frac{30 - 7t}{4}$ | $30 + t$ | $165 + \frac{3}{2}t$ |
| $t > \frac{30}{7}$ | (No feasible solution exists) | | | |

## PROBLEM SET 7.6B

**1.** In Example 7.6-2, find the first critical value, $t_1$, and define the vectors of $\mathbf{B}_1$ in each of the following cases:

**(a)** $\mathbf{b}(t) = (40 + 2t, 60 - 3t, 30 + 6t)^T$

**(b)** $\mathbf{b}(t) = (40 - t, 60 + 2t, 30 - 5t)^T$

**2.** Study the variation in the optimal solution of the following parameterized LP given $t \geq 0$.

$$\text{Minimize } z = 4x_1 + x_2 + 2x_3$$

subject to

$$3x_1 + \ x_1 + 2x_3 = 3 + 3t$$
$$4x_1 + 3x_2 + 2x_3 \geq 6 + 2t$$
$$x_1 + 2x_2 + 5x_3 \leq 4 - t$$
$$x_1, x_2, x_3 \geq 0$$

**3.** The analysis in this section assumes that the optimal LP solution at $t = 0$ is obtained by the (primal) simplex method. In some problems, it may be more convenient to obtain the optimal solution by the dual simplex method (Section 4.4). Show how the parametric analysis can be carried out in this case, and then analyze the LP of Example 4.4-1, assuming that the right-hand-side vector is

$$\mathbf{b}(t) = (3 + 2t, 6 - t, 3 - 4t)^T$$

Assume $t \geq 0$.

**4.** Solve Problem 2 assuming that the right-hand side is changed to

$$\mathbf{b}(t) = (3 + 3t^2, 6 + 2t^2, 4 - t^2)^T$$

Further assume that $t$ can be positive, zero, or negative.

## 7.7 KARMARKAR INTERIOR-POINT METHOD

The simplex method obtains the optimum solution by following a path of adjacent extreme points along the edges of the solution space. Although in practice the simplex method has served well in solving large problems, theoretically the number of iterations needed to reach the optimum solution can grow exponentially. In fact, researchers have constructed a class of LPs in which all feasible extreme points are visited before the optimum is reached.

In 1984, N. Karmarkar developed a polynomial-time algorithm that cuts across the interior of the solution space. The algorithm is effective for extremely large LPs.

We start by introducing the main idea of the Karmarkar method and then provide the computational details of the algorithm.

### 7.7.1 Basic Idea of the Interior-Point Algorithm

Consider the following (trivial) example:

$$\text{Maximize } z = x_1$$

subject to

$$0 \le x_1 \le 2$$

Using $x_2$ as a slack variable, the problem can be rewritten as

$$\text{Maximize } z = x_1$$

subject to

$$x_1 + x_2 = 2$$

$$x_1, x_2 \ge 0$$

Figure 7.6 depicts the problem. The solution space is given by the line segment $AB$. The direction of increase in $z$ is in the positive direction of $x_1$.

Let us start with any arbitrary *interior* (nonextreme) point $C$ in the feasible space (line $AB$). The **gradient** of the objective function (maximize $z = x_1$) at $C$ is the direction of fastest increase in $z$. If we locate an arbitrary point along the gradient and then project it perpendicularly on the feasible space (line $AB$), we obtain the new point $D$ with a better objective value $z$. Such improvement is obtained by moving in the direction of the **projected gradient** $CD$. If we repeat the procedure at $D$, we will determine a new closer-to-optimum point $E$. Conceivably, if we move (cautiously) in the direction of the projected gradient, we will "stumble" on the optimum point $B$. If we are minimizing $z$ (instead of maximizing), the projected gradient will correctly move us *away* from point $B$ toward the minimum at point $A$ $(x_1 = 0)$.

The given steps hardly define an algorithm in the normal sense, but the idea is intriguing! We need some modifications that will guarantee that (1) the steps generated along the projected gradient will not "overshoot" the optimum point at B, and (2) in the general $n$-dimensional case, the direction created by the projected gradient will

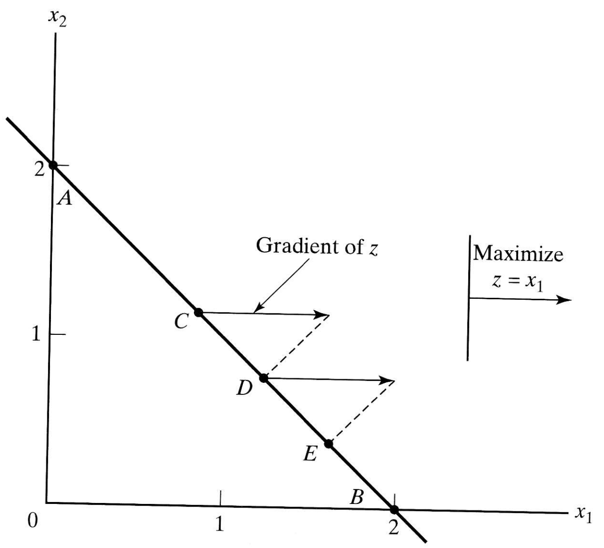



FIGURE 7.6
Illustration of the general idea of Karmarkar's algorithm

not cause an "entrapment" of the algorithm at a nonoptimum point. This, basically, is what Karmarkar's interior-point algorithm accomplishes.

### 7.7.2 Interior-Point Algorithm

Several variants of Karmarkar's algorithm are available in the literature. Our presentation follows the original algorithm. Karmarkar assumes that the LP is given as

$$\text{Minimize } z = \mathbf{CX}$$

subject to

$$\mathbf{AX} = \mathbf{0}$$
$$\mathbf{1X} = 1$$
$$\mathbf{X} \geq \mathbf{0}$$

All the constraints are homogeneous equations except for the constraint $\mathbf{1X} = \sum_{j=1}^{n} x_j = 1$, which defines an $n$-dimensional simplex. The validity of Karmarkar's algorithm rests on satisfying two conditions:

1. $\mathbf{X} = (\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n})$ satisfies $\mathbf{AX} = \mathbf{0}$
2. $\min z = 0$

Karmarkar provides modifications that allow solving the problem when the second condition is not satisfied. These modifications will not be presented here.

The following example illustrates how a general LP may be put in the homogeneous form $\mathbf{AX} = \mathbf{0}$ with $\mathbf{1X} = 1$, which also provides $\mathbf{X} = (\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n})$ as a feasible solution (condition 1). A second example shows how the transformation can be made to satisfy both conditions, albeit involving tedious computations.

---

**Example 7.7-1**

Consider the problem.

$$\text{Maximize } z = y_1 + y_2$$

subject to

$$y_1 + 2y_2 \leq 2$$
$$y_1, y_2 \geq 0$$

The constraint $y_1 + 2y_2 \leq 2$ is converted into an equation by augmenting a slack variable $y_3 \geq 0$ to yield

$$y_1 + 2y_2 + y_3 = 2$$

Now define

$$y_1 + y_2 + y_3 \leq U$$

where $U$ is sufficiently large so as not to eliminate any feasible points in the original solution space. In our example, $U = 5$ will be adequate as can be determined from the equation $y_1 + 2y_2 + y_3 = 2$. Augmenting a slack variable $y_4 \geq 0$, we obtain

$$y_1 + y_2 + y_3 + y_4 = 5$$

We can homogenize the constraint $y_1 + 2y_2 + y_3 = 2$ by multiplying the right-hand side by $\frac{(y_1 + y_2 + y_3 + y_4)}{5}$ because the latter fraction equals 1. This yields, after simplification,

$$3y_1 + 8y_2 + 3y_3 - 2y_4 = 0$$

To convert $y_1 + y_2 + y_3 + y_4 = 5$ to a simplex, we define the new variable $x_i = \frac{y_i}{5}$, $i = 1, 2, 3, 4$, to obtain

$$\text{Maximize } z = 5x_1 + 5x_2$$

subject to

$$3x_1 + 8x_2 + 3x_3 - 2x_4 = 0$$

$$x_1 + x_2 + x_3 + x_4 = 1$$

$$x_j \geq 0,\ j = 1,2,3,4$$

Finally, we can ensure that the center $\mathbf{X} = (\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n})$ of the simplex is a feasible point for homogeneous equations by subtracting from the left-hand side of each equation an artificial variable whose coefficient equals the algebraic sum of all the constraint coefficients on the left-hand side—that is, $3 + 8 + 3 - 2 = 12$. The artificial variables are then added to the simplex equation and are penalized appropriately in the objective function. In our example, the artificial $x_5$ is augmented as follows:

$$\text{Maximize } z = 5x_1 + 5x_2 - Mx_5$$

subject to

$$3x_1 + 8x_2 + 3x_3 - 2x_4 - 12x_5 = 0$$

$$x_1 + x_2 + x_3 + x_4 + x_5 = 1$$

$$x_j \geq 0, j = 1,2,\ldots,5$$

For this system of equations, the new simplex center $(\frac{1}{5}, \frac{1}{5}, \ldots, \frac{1}{5})$ is feasible for the homogeneous equation. The value $M$ in the objective function is chosen sufficiently large to drive $x_5$ to zero level (compare with the $M$-method, Section 3.4.1).

---

### Example 7.7-2

This example shows that any LP can satisfy conditions (1) and (2) required by Karmarkar's algorithm. The transformations are tedious and, hence, not recommended in practice. Instead, a variation of the algorithm that does not require condition (2) is advisable.

Consider the same LP of Example 7.8-1—namely,

$$\text{Maximize } z = y_1 + y_2$$

subject to

$$y_1 + 2y_2 \leq 2$$

$$y_1, y_2 \geq 0$$

We start by defining the primal and dual problems of the LP:

| Primal | Dual |
|---|---|
| Maximize $y_0 = y_1 + y_2$ subject to | Minimize $w_0 = 2w_1$ subject to |
| $y_1 + 2y_2 \le 2$ | $\left.\begin{array}{r} w_1 \ge 1 \\ 2w_1 \ge 1 \end{array}\right\} \Rightarrow w_1 \ge 1$ |
| $y_1, y_2 \ge 0$ | $w_1, w_2 \ge 0$ |

The primal and dual constraints can be put in equation forms as

$$y_1 + 2y_2 + y_3 = 2, y_3 \ge 0 \tag{1}$$
$$w_1 - w_2 = 1, w_2 \ge 0$$

At the optimum $y_0 = w_0$, which yields

$$y_1 + y_2 - 2w_1 = 0 \tag{2}$$

Selecting $M$ sufficiently large, we have

$$y_1 + y_2 + y_3 + w_1 + w_2 \le M \tag{3}$$

Now, converting (3) into an equation we get

$$y_1 + y_2 + y_3 + w_1 + w_2 + s_1 = M,\ s_1 \ge 0 \tag{4}$$

Next, define a new variable $s_2$. From (4) the following two equations hold if, and only if, the condition $s_2 = 1$ holds:

$$y_1 + y_2 + y_3 + w_1 + w_2 + s_1 - Ms_2 = 0$$
$$y_1 + y_2 + y_3 + w_1 + w_2 + s_1 + s_2 = M + 1 \tag{5}$$

Now, given $s_2 = 1$ as stipulated by (5), the primal and dual equations (1) can be written as

$$y_1 + 2y_2 + y_3 - 2s_2 = 0$$
$$w_1 - w_2 - 1s_2 = 0 \tag{6}$$

Now, define

$$y_j = (M + 1)x_j,\ j = 1,2,3$$
$$w_{j-3} = (M + 1)x_j,\ j = 4,5$$
$$s_1 = (M + 1)x_6$$
$$s_2 = (M + 1)x_7$$

Substitution in equations (2), (5), and (6) will produce the following equations:

$$\begin{aligned}
x_1 + x_2 - 2x_4 &= 0 \\
x_1 + x_2 + x_3 + x_4 + x_5 + x_6 - Mx_7 &= 0 \\
x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 &= 1 \\
x_1 + 2x_2 + x_3 - 2x_7 &= 0
\end{aligned}$$

$$x_4 - x_5 \qquad - \quad x_7 = 0$$

$$x_j \geq 0,\ j = 1,2,\ldots,7$$

The final step calls for augmenting the artificial variable $y_8$ in the left-hand side of each equation; the new objective function will call for minimizing $y_8$, whose minimum value must be zero (assuming the primal is feasible). Note, however, that Karmarkar's algorithm requires the solution

$$\mathbf{X} = \left(\tfrac{1}{8}, \tfrac{1}{8}, \tfrac{1}{8}, \tfrac{1}{8}, \tfrac{1}{8}, \tfrac{1}{8}, \tfrac{1}{8}, \tfrac{1}{8}\right)^T$$

to be feasible for $\mathbf{AX} = \mathbf{0}$. This will be true for the homogeneous equations (with zero right-hand side) if the associated coefficient of the artificial $x_8$ equals the (algebraic) sum of all the coefficients on the left-hand side. It thus follows that the transformed LP is given as

$$\text{Minimize } z = x_8$$

subject to

$$\begin{array}{rcl}
x_1 + x_2 \qquad - 2x_4 \qquad\qquad\qquad\qquad - \quad 0x_8 &=& 0 \\
x_1 + x_2 + x_3 + x_4 + x_5 + x_6 - Mx_7 - (6 - M)x_8 &=& 0 \\
x_1 + 2x_2 + x_3 \qquad\qquad\qquad - 2x_7 - \quad 2x_8 &=& 0 \\
x_4 - x_5 \qquad - x_7 + \quad x_8 &=& 0 \\
x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + \quad x_8 &=& 1 \\
x_j \geq 0,\ j = 1, 2, \ldots, 8 & &
\end{array}$$

Note that the solution of this problem automatically yields the optimum solutions of the primal and dual problems through substitution.

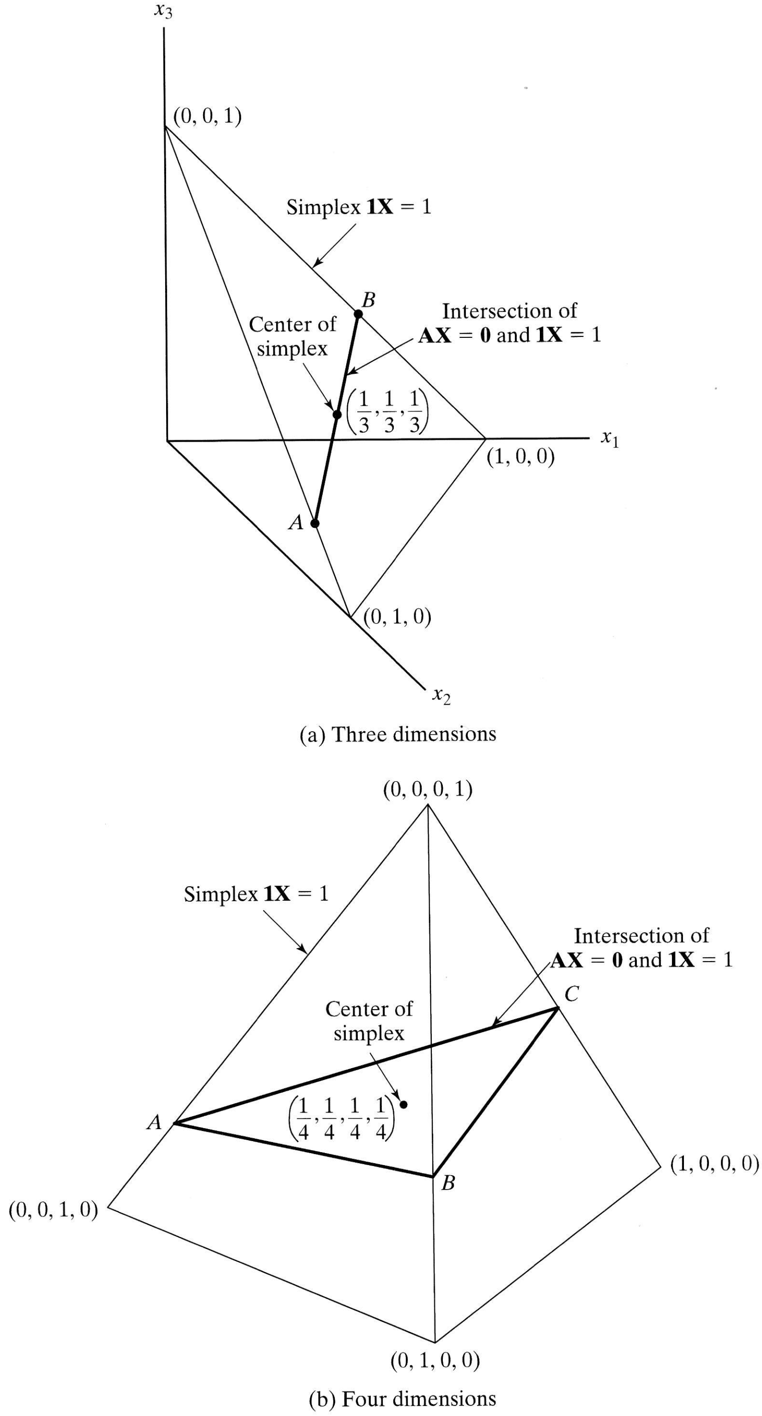
---

We now present the main steps of the algorithm. Figure 7.7 (a) provides a typical illustration of the solution space in three dimensions with the homogeneous set $\mathbf{AX} = \mathbf{0}$ consisting only of one equation. By definition, the solution space consisting of the line segment $AB$ lies entirely in the two-dimensional simplex $\mathbf{1X} = 1$ and passes through the feasible interior point $\left(\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}\right)$. In a similar fashion, Figure 7.7 (b) provides an illustration of the solution space $ABC$ in four dimensions with the homogeneous set again consisting of one constraint only. In this case, the center of the three-dimensional simplex is given by $\left(\tfrac{1}{4}, \tfrac{1}{4}, \tfrac{1}{4}, \tfrac{1}{4}\right)$.

Karmarkar's algorithm starts from an interior point represented by the center of the simplex and then advances in the direction of the *projected gradient* to determine a new solution point. The new point must be strictly interior, meaning that all its coordinates must be positive. The validity of the algorithm rests on this condition.

For the new solution point to be strictly interior, it must not lie on the boundaries of the simplex. (In terms of Figure 7.7, points $A$ and $B$ in three dimensions and lines $AB$, $BC$, and $AC$ in four dimensions must be excluded.) To guarantee this result, a sphere with its center coinciding with that of the simplex is inscribed tightly inside the simplex. In the $n$-dimensional case, the radius $r$ of this sphere equals $\frac{1}{\sqrt{n(n-1)}}$. A smaller sphere with radius $\alpha r$ $(0 < \alpha < 1)$ will be a subset of the sphere, and any point in the intersection of the smaller sphere with the homogeneous system $\mathbf{AX} = \mathbf{0}$ will be

FIGURE 7.7
Illustrations of the simplex $\mathbf{1X} = 1$



an interior point, with strictly positive coordinates. Thus, we can move as far as possible in this restricted space (intersection of $\mathbf{AX} = \mathbf{0}$ and the $\alpha$ $r$-sphere) along the projected gradient to determine the new (necessarily improved) solution point.

The new solution point no longer will be at the center of the simplex. For the procedure to be iterative, we need to bring the *new* solution point to the center of a simplex. Karmarkar satisfies this requirement by proposing the following intriguing idea, called **projective transformation**. Let

$$y_i = \frac{\frac{x_i}{x_{ki}}}{\sum_{j=1}^{n} \frac{x_j}{x_{kj}}}, i = 1,2,\ldots,n$$

where $x_{ki}$ is the $i$th element of the current solution point $\mathbf{X}_k$. The transformation is valid, because all $x_{ki} > 0$ by design. You will also notice that $\sum_{i=1}^{n} y_i = 1$, or $\mathbf{1Y} = 1$, by definition. This transformation is equivalent to

$$\mathbf{Y} = \frac{\mathbf{D}_k^{-1}\mathbf{X}}{\mathbf{1D}_k^{-1}\mathbf{X}}$$

where $\mathbf{D}_k$ is a diagonal matrix whose $i$th diagonal elements equal $x_{ki}$. The transformation maps the $X$-space onto the $Y$-space uniquely because we can directly show that the last equation yields

$$\mathbf{X} = \frac{\mathbf{D}_k\mathbf{Y}}{\mathbf{1D}_k\mathbf{Y}}$$

By definition, min $\mathbf{CX} = 0$. Because $\mathbf{1D}_k\mathbf{Y}$ is always positive, the original linear program is equivalent to

$$\text{Minimize } z = \mathbf{CD}_k\mathbf{Y}$$

subject to

$$\mathbf{AD}_k\mathbf{Y} = 0$$

$$\mathbf{1Y} = 1$$

$$\mathbf{Y} \geq 0$$

The transformed problem has the same format as the original problem. We can thus start with the simplex center $\mathbf{Y} = (\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n})$ and repeat the iterative step. After each iteration, we can compute the values of the original $\mathbf{X}$ variables from the $\mathbf{Y}$ solution.

We show now how the new solution point can be determined for the transformed problem. At any iteration $k$, the problem is given by

$$\text{Minimize } z = \mathbf{CD}_k\mathbf{Y}$$

subject to

$$\mathbf{AD}_k\mathbf{Y} = 0$$

$$\mathbf{Y} \text{ lies in the } \alpha r\text{-sphere}$$

Because the $\alpha r$ -sphere is a subset of the space of the constraints $\mathbf{1X} = \mathbf{1}$ and $\mathbf{X} \geq \mathbf{0}$, these two constraints can be dispensed with. As a result, the optimum solution of the preceding problem lies along the negative projection of the gradient $\mathbf{c}_p$ (minimization) and is given as

$$\mathbf{Y}_{new} = \mathbf{Y}_0 - \alpha r \frac{\mathbf{c}_p}{\|\mathbf{c}_p\|}$$

where $\mathbf{Y}_0 = (\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n})^T$ and $\mathbf{c}_p$ is the projected gradient, which can be shown to be

$$\mathbf{c}_p = [\mathbf{I} - \mathbf{P}^T(\mathbf{P}\mathbf{P}^T)^{-1}\mathbf{P}](\mathbf{C}\mathbf{D}_k)^T$$

where

$$\mathbf{P} = \begin{pmatrix} \mathbf{A}\mathbf{D}_k \\ \mathbf{1} \end{pmatrix}$$

The selection of $\alpha$ is crucial to enhancing the efficiency of the algorithm. Normally, we select $\alpha$ as large as possible to acquire large jumps in the solution. However, by choosing $\alpha$ too large, we may come too close to the prohibited boundaries of the simplex. There is no general answer to this problem, but Karmarkar suggests the use of $\alpha = \frac{n-1}{3n}$

The steps of Karmarkar's algorithm are

**Step 0.** Start with the solution point $\mathbf{X}_0 = (\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n})$ and compute $r = \frac{1}{n\sqrt{(n-1)}}$ and $\alpha = \frac{(n-1)}{3n}$

**General step k.** Define

$$\mathbf{D}_k = \text{diag}\{x_{k1}, \ldots, x_{kn}\}$$

$$\mathbf{P} = \begin{pmatrix} \mathbf{A}\mathbf{D}_k \\ \mathbf{1} \end{pmatrix}$$

and compute

$$\mathbf{Y}_{new} = \left(\frac{1}{n}, \ldots, \frac{1}{n}\right)^T - \alpha r \frac{\mathbf{c}_p}{\|\mathbf{c}_p\|}$$

$$\mathbf{X}_{k+1} = \frac{\mathbf{D}_k \mathbf{Y}_{new}}{\mathbf{1}\mathbf{D}_k \mathbf{Y}_{new}}$$

where

$$\mathbf{c}_p = [\mathbf{I} - \mathbf{P}^T(\mathbf{P}\mathbf{P}^T)^{-1}\mathbf{P}](\mathbf{c}\mathbf{D}_k)^T$$

---

**Example 7.7-3**

$$\text{Minimize } z = 2x_1 + 2x_2 - 3x_3$$

subject to

$$-x_1 - 2x_2 + 3x_3 = 0$$

$$x_1 + x_2 + x_3 = 1$$

$$x_1, x_2, x_3 \geq 0$$

The problem satisfies the two conditions imposed by the interior-point algorithm—namely,

$$\mathbf{X} = (x_1, x_2, x_3)^T = (\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3})^T$$

satisfies both constraints and the optimum solution

$$\mathbf{X}^* = (x_1^*, x_2^*, x_3^*)^T = (0, .6, .4)^T$$

yields $z = 0$.

**Iteration 0.**

$$\mathbf{c} = (2,\ 2,\ -3), \mathbf{A} = (-1, -2, 3)$$

$$\mathbf{X}_0 = (\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3})^T, z_0 = \tfrac{1}{3}, r = \tfrac{1}{\sqrt{6}}, \alpha = \tfrac{2}{9}$$

$$\mathbf{D}_0 = \begin{pmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{3} \end{pmatrix}$$

Using projective transformation, we get

$$\mathbf{Y}_0 = (\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3})^T$$

**Iteration 1.**

$$\mathbf{cD}_0 = (2, 2, -3)\begin{pmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{3} \end{pmatrix} = (\tfrac{2}{3}, \tfrac{2}{3}, -1)$$

$$\mathbf{AD}_0 = (-1, -2, 3)\begin{pmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{3} \end{pmatrix} = (-\tfrac{1}{3}, -\tfrac{2}{3}, 1)$$

$$(\mathbf{PP}^T)^{-1} = \left(\begin{pmatrix} -\frac{1}{3} & -\frac{2}{3} & 1 \\ 1 & 1 & 1 \end{pmatrix}\begin{pmatrix} -\frac{1}{3} & 1 \\ -\frac{2}{3} & 1 \\ 1 & 1 \end{pmatrix}\right)^{-1} = \begin{pmatrix} \frac{9}{14} & 0 \\ 0 & \frac{1}{3} \end{pmatrix}$$

$$\mathbf{I} - \mathbf{P}^T(\mathbf{PP}^T)^{-1}\mathbf{P} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} - \begin{pmatrix} -\frac{1}{3} & 1 \\ -\frac{2}{3} & 1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} \frac{9}{14} & 0 \\ 0 & \frac{1}{3} \end{pmatrix}\begin{pmatrix} -\frac{1}{3} & -\frac{2}{3} & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$= \frac{1}{42}\begin{pmatrix} 25 & -20 & -5 \\ -20 & 16 & 4 \\ -5 & 4 & 1 \end{pmatrix}$$

Thus,

$$\mathbf{c}_p = (\mathbf{I} - \mathbf{P}^T(\mathbf{PP}^T)^{-1}\mathbf{P})(\mathbf{cD}_0)^T = \frac{1}{42}\begin{pmatrix} 25 & -20 & -5 \\ -20 & 16 & 4 \\ -5 & 4 & 1 \end{pmatrix}\begin{pmatrix} \frac{2}{3} \\ \frac{2}{3} \\ -1 \end{pmatrix} = \frac{1}{126}\begin{pmatrix} 25 \\ -20 \\ -5 \end{pmatrix}$$

It then follows that

$$\|\mathbf{c}_p\| = \sqrt{\frac{25^2 + (-20)^2 + (-5)^2}{126^2}} = .257172$$

Thus,

$$\mathbf{Y}_{\text{new}} = \left(\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}\right)^T - \tfrac{2}{9} \times \tfrac{1}{\sqrt{6}} \times \tfrac{1}{.257172} \times \tfrac{1}{126}(25, -20, -5)^T$$
$$= (.263340, .389328, .347332)^T$$

Next,

$$\mathbf{1D}_0\mathbf{Y}_{\text{new}} = \tfrac{1}{3}(1,1,1)(.263340, .389328, .347332)^T = \tfrac{1}{3}$$

Now,

$$\mathbf{X}_1 = \frac{\mathbf{D}_0\mathbf{Y}_{\text{new}}}{\mathbf{1D}_0\mathbf{Y}_{\text{new}}} = \frac{\frac{1}{3}\mathbf{Y}_{\text{new}}}{\frac{1}{3}} = \mathbf{Y}_{\text{new}} = (.263340, .389328, .347332)^T$$
$$z_1 = .26334$$

**Iteration 2.**

$$\mathbf{cD}_1 = (2,2,-3)\begin{pmatrix} .263340 & 0 & 0 \\ 0 & .389328 & 0 \\ 0 & 0 & .347332 \end{pmatrix} = (.526680, .778656, -1.041996)$$

$$\mathbf{AD}_1 = (-1,-2,3)\begin{pmatrix} .263340 & 0 & 0 \\ 0 & .389328 & 0 \\ 0 & 0 & .347332 \end{pmatrix} = (-.263340, -.778656, 1.041996)$$

$$(\mathbf{PP}^T)^{-1} = \left(\begin{pmatrix} -.26334 & -.778656 & 1.041996 \\ 1 & 1 & 1 \end{pmatrix}\begin{pmatrix} -.263340 & 1 \\ -.778656 & 1 \\ 1.041996 & 1 \end{pmatrix}\right)^{-1} = \begin{pmatrix} .567727 & 0 \\ 0 & .333333 \end{pmatrix}$$

$$\mathbf{I} - \mathbf{P}^T(\mathbf{PP}^T)^{-1}\mathbf{P} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} - \begin{pmatrix} -.263340 & 1 \\ -.778656 & 1 \\ 1.041996 & 1 \end{pmatrix}\begin{pmatrix} .567727 & 0 \\ 0 & .333333 \end{pmatrix}$$
$$\begin{pmatrix} -.263340 & -.778656 & 1.041996 \\ 1 & 1 & 1 \end{pmatrix}$$
$$= \begin{pmatrix} .627296 & -.449746 & -.177550 \\ -.449746 & .322451 & .127295 \\ -.177550 & .127295 & .050254 \end{pmatrix}$$

Thus,

$$\mathbf{c}_p = (\mathbf{I} - \mathbf{P}^T(\mathbf{PP}^T)^{-1}\mathbf{P})(\mathbf{cD}_1)^T = \begin{pmatrix} .627296 & -.449746 & -.177550 \\ -.449746 & .322451 & .127295 \\ -.177550 & .127295 & .050254 \end{pmatrix}\begin{pmatrix} .526680 \\ .778656 \\ -1.041996 \end{pmatrix}$$
$$= \begin{pmatrix} .165193 \\ -.118435 \\ -.046757 \end{pmatrix}$$

It then follows that

$$\|\mathbf{c}_p\| = \sqrt{.165193^2 + (-.118435)^2 + (-.046757)^2} = .208571$$

Thus,

$$\mathbf{Y}_{new} = (\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3})^T - \tfrac{2}{9} \times \tfrac{1}{\sqrt{6}} \times \tfrac{1}{.208571}(.165193, -.118435, -.046757)^T$$
$$= (.261479, .384849, .353671)^T$$

Next,

$$\mathbf{D}_1\mathbf{Y}_{new} = \begin{pmatrix} .263340 & 0 & 0 \\ 0 & .389328 & 0 \\ 0 & 0 & .347332 \end{pmatrix} \begin{pmatrix} .261479 \\ .384849 \\ .353671 \end{pmatrix} = \begin{pmatrix} .068858 \\ .149832 \\ .122841 \end{pmatrix}$$

$$\mathbf{1D}_1\mathbf{Y}_{new} = .341531$$

Now,

$$\mathbf{X}_2 = \frac{\mathbf{D}_1\mathbf{Y}_{new}}{\mathbf{1D}_1\mathbf{Y}_{new}} = \begin{pmatrix} .201616 \\ .438707 \\ .359677 \end{pmatrix}$$

$$z_2 = .201615$$

Repeated application of the algorithm will move the solution closer to the optimum point $(0, .6, .4)$. Karmarkar does provide an additional step for rounding the optimal solution to the optimum extreme point.

---

## PROBLEM SET 7.7A

1. Use TORA to show that the solution of the transformed LP given at the end of Example 7.7-2 does yield the optimal primal and dual solutions of the parent problem. (*Hint:* Use $M=10$ and make sure that TORA's output gives at least 5 decimal points accuracy.)
2. Transform the following LP to Karmarkar's format.

$$\text{Maximize } z = y_1 + 2y_2$$

subject to

$$y_1 - y_2 \leq 2$$
$$2y_1 + y_2 \leq 4$$
$$y_1, y_2 \geq 0$$

3. Carry out one additional iteration in Example 7.7-3, and show that the solution is moving toward the optimum $z = 0$.
4. Carry out three iterations of Karmarkar's algorithm for the following problem:

$$\text{Maximize } z = 4x_1 + x_3 + x_4$$

subject to

$$-2x_1 + 2x_2 + x_3 - x_4 = 0$$
$$x_1 + x_2 + x_3 + x_4 = 1$$
$$x_1, x_2, x_3, x_4 \geq 0$$

(*Hint:* The problem must be converted to Karmarkar format first.)

**5.** Carry out three iterations of Karmarkar's algorithm for the following linear program:

$$\text{Maximize } z = 2x_1 + x_2$$

subject to

$$x_1 + x_2 \leq 4$$

$$x_1, x_2 \geq 0$$

(*Hint:* The problem must be converted to Karmarkar format first.)

## SELECTED REFERENCES

Bazaraa, M., J. Jarvis, and H. Sherali, *Linear Programming and Network Flows,* 2nd ed., Wiley, New York, 1990.

Hooker, J., "Karmarkar's Linear Programming Algorithm," *Interfaces,* Vol. 16, No. 4, pp. 75–90, 1986.

Nering, E., and A. Tucker, *Linear Programming and Related Problems,* Academic Press, Boston, 1992.

## COMPREHENSIVE PROBLEMS

**7.1** Suppose that you are given the points

$$A = (6, 4, 6, -2),\ B = (4, 12, -4, 8),\ C = (-4, 0, 8, 4)$$

Develop a systematic procedure that will allow determining whether or not each of the following points can be expressed as a convex combination of *A, B,* and *C:*

**(a)** $(3, 5, 4, 2)$

**(b)** $(5, 8, 4, 9)$

**7.2** Consider the following LP:

$$\text{Maximize } z = 3x_1 + 2x_2$$

subject to

$$x_1 + 2x_2 \leq 6$$

$$2x_1 + x_2 \leq 8$$

$$-x_1 + x_2 \leq 1$$

$$x_1, x_2 \geq 0$$

Determine the optimum simplex tableau (use TORA for convenience), and then directly use the information in the optimum simplex tableau to determine the *second* best extreme-point solution (relative to the "absolute" optimum) for the problem. Verify the answer by solving the problem graphically. (*Hint:* Consult the extreme points that are *adjacent* to the optimum solution.)

**7.3** *Interval Programming.* Consider the following LP:

$$\text{Maximize } z = \{\mathbf{CX} \mid \mathbf{L} \leq \mathbf{AX} \leq \mathbf{U}, \mathbf{X} \geq 0\}$$

where **L** and **U** are constant column vectors. Define the slack vector such that $\mathbf{AX} + \mathbf{Y} = \mathbf{U}$. Show that this LP is equivalent to

$$\text{Maximize } z = \{\mathbf{CX} | \mathbf{AX} + \mathbf{Y} = \mathbf{U}, 0 \le \mathbf{Y} \le \mathbf{U} - \mathbf{L}, \mathbf{X} \ge 0\}$$

Use the proposed procedure to solve the following LP:

$$\text{Minimize } z = 5x_1 - 4x_2 + 6x_3$$

subject to

$$20 \le x_1 + 7x_2 + 3x_3 \le 46$$
$$10 \le 3x_1 - x_2 + x_3 \le 20$$
$$18 \le 2x_1 + 3x_2 - x_3 \le 35$$
$$x_1, x_2, x_3 \ge 0$$

**7.4** Consider the following 0-1 integer LP:

$$\text{Minimize } z = \{\mathbf{CX} | \mathbf{AX} \le \mathbf{b}, \mathbf{X} = (0,1)\}$$

Suppose that $z_{\min}$ is a known upper bound on $z$. Define the constraint

$$\min_{\mu \ge 0} \max_{x=(0,1)} \{\mu(\mathbf{b} - \mathbf{AX}) + (z_{\min} - \mathbf{CX})\} \ge 0$$

where $\mu \ge 0$. This constraint does not violate any of the restrictions of the original 0-1 problem. The min-max problem is one way of identifying the "tightest" such constraint through proper selection of $\mu(\ge 0)$. Show that the proposed mixed 0-1 definition for determining $\mu$ actually reduces to solving an ordinary LP problem. (*Hint:* The integer restriction $\mathbf{X} = [0,1]$ is equivalent to the continuous range $0 \le \mathbf{X} \le 1$. Use the dual problem to define the desired LP.)

**7.5** The optimum solution of the LP in Problem 7-2 is given as $x_1 = \frac{10}{3}$, $x_2 = \frac{4}{3}$, and $z = \frac{38}{3}$. Plot the change in optimum $z$ with $\theta$ given that $x_1 = \frac{10}{3} + \theta$, where $\theta$ is unrestricted in sign. Note that $x_1 = \frac{10}{3} + \theta$ tracks $x_1$ above and below its optimal value.

**7.6** Suppose that the optimum linear program is represented as

$$\text{Maximize } z = c_0 - \sum_{j \in NB} (z_j - c_j) x_j$$

subject to

$$x_i = x_i^* - \sum_{j \in NB} \alpha_{ij} x_j, i = 1, 2, \ldots, m$$

$$\text{all } x_i \text{ and } x_j \ge 0$$

where $NB$ is the set of nonbasic variables. Suppose that for a current basic variable $x_i = x_i^*$ we impose the restriction $x_i \ge d_i$, where $d_i$ is the smallest integer greater than $x_i^*$. Estimate an upper bound on the optimum value of $z$ after the constraint is augmented to the problem. Repeat the same procedure assuming that the imposed restriction is $x_i \le e_i$, where $e_i$ is the largest integer smaller than $x_i^*$.

**7.7** Consider the following minimization LP:

$$\text{Minimize } z = (10t - 4)x_1 + (4t - 8)x_2$$

subject to

$$2x_1 + 2x_2 + x_3 \qquad = 8$$

$$4x_1 + 2x_2 \qquad + x_4 = 6 - 2t$$

$$x_1, x_2, x_3, x_4 \geq 0$$

where $-\infty \leq t \leq \infty$. The parametric analysis of the problem yields the following results:

$-\infty \leq t \leq -5$: Optimal basis is $\mathbf{B} = (\mathbf{P}_1, \mathbf{P}_4)$

$-5 \leq t \leq -1$: Optimal basis is $\mathbf{B} = (\mathbf{P}_1, \mathbf{P}_2)$

$-1 \leq t \leq 2$: Optimal basis is $\mathbf{B} = (\mathbf{P}_2, \mathbf{P}_3)$

Determine all the critical values of $t$ that may exist for $t \geq 2$.

8.1

results:

CHAPTER 8

# Goal Programming

The LP models presented in the preceding chapters are based on the optimization of a *single* objective function. There are situations where multiple (possibly conflicting) objectives may be more appropriate. For example, aspiring politicians may promise to reduce the national debt and, simultaneously, offer income tax relief. In such situations, it may be impossible to find a single solution that optimizes the conflicting objectives. Instead, we may seek a *compromise* solution based on the relative importance of each objective.

This chapter presents the goal programming technique for solving multiobjective models. The principal idea is to convert the original multiple objectives into a single goal. The resulting model yields what is usually referred to as an **efficient solution** because it may not be optimum with respect to *all* the conflicting objectives of the problem.

## 8.1 A GOAL PROGRAMMING FORMULATION

The idea of goal programming is illustrated by an example.

### Example 8.1-1

Fairville is a small city with a population of about 20,000 residents. The city council is in the process of developing an equitable city tax rate table. The annual taxation base for real estate property is $550 million. The annual taxation bases for food and drugs and for general sales are $35 million and $55 million, respectively. Annual local gasoline consumption is estimated at 7.5 million gallons. The city council wants to develop the tax rates based on four main goals.

1. Tax revenues must be at least $16 million to meet the city's financial commitments.
2. Food and drug taxes cannot exceed 10% of all taxes collected.

3. General sales taxes cannot exceed 20% of all taxes collected.
4. Gasoline tax cannot exceed 2 cents per gallon.

Let the variables $x_p$, $x_f$, and $x_s$ represent the tax rates (expressed as proportions of taxation bases) for property, food and drugs, and general sales; and define the variable $x_g$ as the gasoline tax in cents per gallon. The goals of the city council are then expressed as

$$
\begin{aligned}
&550x_p + 35x_f + 55x_s + .075x_g \geq 16 && \text{(Tax revenue)}\\
&35x_f \leq .1(550x_p + 35x_f + 55x_s + .075x_g) && \text{(Food/drug tax)}\\
&55x_s \leq .2(550x_p + 35x_f + 55x_s + .075x_g) && \text{(General tax)}\\
&x_g \leq 2 && \text{(Gasoline tax)}\\
&x_p, x_f, x_s, x_g \geq 0
\end{aligned}
$$

These constraints are then simplified as

$$
\begin{aligned}
550x_p + 35x_f + 55x_s + .075x_g &\geq 16\\
55x_p - 31.5x_f + 5.5x_s + .0075x_g &\geq 0\\
110x_p + 7x_f - 44x_s + .015x_g &\geq 0\\
x_g &\leq 2\\
x_p, x_f, x_s, x_g &\geq 0
\end{aligned}
$$

Each of the inequalities of the model represents a goal that the city council aspires to satisfy. Most likely, however, the best we can do is seek a compromise solution among these conflicting goals.

The manner in which goal programming finds a compromise solution is to convert each inequality into a flexible goal in which the corresponding constraint may be violated, if necessary. In terms of the Fairville model, the flexible goals are expressed as follows:

$$
\begin{aligned}
550x_p + 35x_f + 55x_s + .075x_g + s_1^+ - s_1^- &= 16\\
55x_p - 31.5x_f + 5.5x_s + .0075x_g + s_2^+ - s_2^- &= 0\\
110x_p + 7x_f - 44x_s + .015x_g + s_3^+ - s_3^- &= 0\\
x_g + s_4^+ - s_4^- &= 2\\
x_p, x_f, x_s, x_g &\geq 0\\
s_i^+, s_i^- \geq 0, i &= 1, 2, 3, 4
\end{aligned}
$$

The nonnegative variables $s_i^+$ and $s_i^-$, $i = 1, 2, 3, 4$, are called **deviational variables** because they represent the deviations above and below the right-hand side of constraint $i$.

The deviational variables $s_i^+$ and $s_i^-$ are by definition dependent and, hence, cannot be basic variables simultaneously. This means that in any simplex iteration, at most *one* of the two deviational variables can assume a positive value. If the original $i$th inequality is of the type $\leq$ and its $s_i^+ > 0$, then the $i$th goal will be satisfied; otherwise, if $s_i^- > 0$, goal $i$ will not be satisfied. In essence, the definition of $s_i^+$ and $s_i^-$ allows us to meet or violate the $i$th goal at will. This is the type of flexibility that characterizes goal programming when it seeks a compromise solution. Naturally, a good compromise solution aims at minimizing the amount by which each goal is violated.

In the Fairville model, given that the first three constraints are of the type $\geq$ and the fourth constraint is of the type $\leq$, the deviational variables $s_1^+$, $s_2^+$, $s_3^+$, and $s_4^-$ of the problem represent the amounts by which the respective goals are violated. Thus, the compromise solution tries to satisfy the following four objectives as much as possible:

$$\text{Minimize } G_1 = s_1^+$$

$$\text{Minimize } G_2 = s_2^+$$

$$\text{Minimize } G_3 = s_3^+$$

$$\text{Minimize } G_4 = s_4^-$$

These functions are minimized subject to the constraint equations of the model.

How can we optimize a multiobjective model with possibly conflicting goals? Two methods have been developed for this purpose: (1) the weights method and (2) the preemptive method. Both methods are based on converting the multiple objectives into a single function as detailed in Section 8.2.

## PROBLEM SET 8.1A

1. Formulate the Fairville tax problem, assuming that the town council is specifying an additional goal, $G_5$, that requires gasoline tax to equal at least 10% of the total tax bill.
2. The NW Shopping Mall conducts special events to attract potential patrons. The two most popular events that seem to attract teenagers, the young/middle-aged group, and senior citizens are band concerts and art and craft shows. The costs per presentation of the band and art show are \$1500 and \$3000, respectively. The total (strict) annual budget allocated to the two events is \$15,000. The mall manager estimates the attendance of the events as follows:

| | Number attending per presentation | | |
|---|---|---|---|
| *Event* | *Teenagers* | *Young/middle age* | *Seniors* |
| Band concert | 200 | 100 | 0 |
| Art show | 0 | 400 | 250 |

   The manager has set the minimum annual goals of 1000, 1200, and 800 for the attendance of teenagers, the young/middle-aged group, and seniors, respectively. Formulate the problem as a goal programming model.
3. Ozark University admissions office is processing freshman applications for the upcoming academic year. The applications fall into three categories: instate, out-of-state, and international. The male–female ratios for in-state and out-of-state applicants are 1:1 and 3:2, respectively. For the international students, the corresponding ratio is 8:1. The American College Test (ACT) score is an important factor in accepting new students.

Statistics indicate that the average ACT scores for in-state, out-of-state, and international students are 27, 26, and 23, respectively. The committee on admissions has established the following desirable goals for the new freshman class:

**(a)** The incoming class is at least 1200 freshmen.

**(b)** The average ACT score for all incoming students is at least 25.

**(c)** International students constitute at least 10% of the incoming class.

**(d)** The female–male ratio is at least 3:4.

**(e)** Out-of-state students constitute at least 20% of the incoming class.

Formulate the problem as a goal programming model.

**4.** Circle K farms consume 3 tons of special feed daily. The feed—a mixture of limestone, corn, and soybean meal—must satisfy the following nutritional requirements:

*Calcium.* At least 0.8% but not more than 1.2%

*Protein.* At least 22%

*Fiber.* At most 5%

The following table gives the nutritional content of the feed ingredients.

| | lb per lb of ingredient | | |
|---|---|---|---|
| *Ingredient* | *Calcium* | *Protein* | *Fiber* |
| Limestone | .380 | .00 | .00 |
| Corn | .001 | .09 | .02 |
| Soybean meal | .002 | .50 | .08 |

Formulate the problem as a goal programming model, and state your opinion regarding the applicability of goal programming to this situation.

**5.** Mantel produces a toy carriage, whose final assembly must include four wheels and two seats. The factory producing the parts operates three shifts a day. The following table provides the amounts produced of each part in the three shifts.

| | Units produced per run | |
|---|---|---|
| *Shift* | *Wheels* | *Seats* |
| 1 | 500 | 300 |
| 2 | 600 | 280 |
| 3 | 640 | 360 |

Ideally, the number of produced wheels is exactly twice that of the number of seats. However, because the production rates vary from shift to shift, exact balance in production may not be possible. Mantel is interested in determining the number of production runs in each shift that minimizes the imbalance in the production of the parts. The capacity limitations restrict the number of runs to between 4 and 5 for shift 1, 10 and 20 for shift 2, and 3 and 5 for shift 3. Formulate the problem as a goal programming model.

**6.** Camyo Manufacturing produces four parts that require the use of a lathe and a drill press. The two machines operate 10 hours a day. The following table provides the time in minutes required by each part:

| | Production time in min | |
|---|---|---|
| *Part* | *Lathe* | *Drill press* |
| 1 | 5 | 3 |
| 2 | 6 | 2 |
| 3 | 4 | 6 |
| 4 | 7 | 4 |

It is desired to balance the two machines by limiting the difference between their total operation times to at most 30 minutes. The market demand for each part is at least 10 units. Additionally, the number of units of part 1 may not exceed that of part 2. Formulate the problem as a goal programming model.

**7.** Two products are manufactured on two sequential machines. The following table gives the machining times in minutes per unit for the two products.

| | Machining time in min | |
|---|---|---|
| *Machine* | *Product 1* | *Product 2* |
| 1 | 5 | 3 |
| 2 | 6 | 2 |

The daily production quotas for the two products are 80 and 60 units, respectively. Each machine runs 8 hours a day. Overtime, though not desirable, may be used if necessary to meet the production quota. Formulate the problem as a goal programming model.

**8.** Vista City Hospital plans the short-stay assignment of surplus beds (those that are not already occupied) 4 days in advance. During the 4-day planning period about 30, 25, and 20 patients will require 1-, 2-, or 3-day stays, respectively. Surplus beds during the same period are estimated at 20, 30, 30, and 30. Use goal programming to resolve the problem of overadmission and underadmission in the hospital.

**9.** The Von Trapp family is in the process of moving to a new city where both parents have accepted new jobs. In trying to find an ideal location for their new home, the Von Trapps list the following goals:

**(a)** It should be as close as possible to Mrs. Von Trapp's place of work (within $\frac{1}{4}$ of a mile).

**(b)** It should be as far as possible from the noise of the airport (at least 10 miles).

**(c)** It should be reasonably close to a shopping mall (within 1 mile).

Mr. and Mrs. Von Trapp use a landmark in the city as a reference point and locate the $x$-$y$ coordinates of work, airport, and shopping mall at (1, 1), (20, 15), and (4, 7), respectively (all distances are in miles). Formulate the problem as a goal programming model. (*Note:* The resulting constraints are not necessarily linear.)

**10.** *Regression Analysis.* In a laboratory experiment, suppose that $y_i$ is the $i$th observed (independent) yield associated with the dependent observational measurements $x_{ij}$, $i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n$. It is desired to determine a linear regression fit into these data points. Given $b_j$, $j = 0, 1, \ldots, n$, as the regression coefficients, all $b_j$ are determined such that the sum of the absolute deviations between the observed and the estimated yield is minimized. Formulate the problem as a goal programming model.

**11.** *Chebyshev Problem.* An alternative goal for the regression model in Problem 10 is to minimize over $b_j$ the maximum of the absolute deviations. Formulate the problem as a goal programming model.

## 8.2 GOAL PROGRAMMING ALGORITHMS

This section presents two algorithms for solving goal programming. Both methods convert the multiple goals into a single objective function. In the **weights method**, the single objective function is the weighted sum of the functions representing the goals of the problem. The **preemptive method** starts by prioritizing the goals in order of importance. The model is then optimized using one goal at a time such that the optimum value of a higher priority goal is never degraded by a lower priority goal.

The proposed two methods do not generally produce the same solution. Neither method, however, is superior to the other because each technique is designed to satisfy certain decision-making preferences.

### 8.2.1 The Weights Method

Suppose that the goal programming model has $n$ goals and that the $i$th goal is given as

$$\text{Minimize } G_i,\ i = 1, 2, \ldots, n$$

The combined objective function used in the weights method is defined as

$$\text{Minimize } z = w_1G_1 + w_2G_2 + \ldots + w_nG_n$$

The parameter $w_i$, $i = 1, 2, \ldots, n$, represents positive weights that reflect the decision maker's preferences regarding the relative importance of each goal. For example, $w_i = 1$, for all $i$, signifies that all goals carry equal weights. The determination of the specific values of these weights is subjective. Indeed, the apparently sophisticated analytic procedures developed in the literature (see, e.g., Cohon, 1978) are still rooted in subjective assessments.

#### Example 8.2-1

TopAd, a new advertising agency with 10 employees, has received a contract to promote a new product. The agency can advertise by radio and television. The following table provides data about the number of people reached by each type of advertisement, and the cost and labor requirements.

| | Data/min advertisement | |
|---|---|---|
| | *Radio* | *Television* |
| Exposure (in millions of persons) | 4 | 8 |
| Cost (in thousands of dollars) | 8 | 24 |
| Assigned employees | 1 | 2 |

The contract prohibits TopAd from using more than 6 minutes of radio advertisement. Additionally, radio and television advertisements need to reach at least 45 million people. TopAd has set a budget goal of \$100,000 for the project. How many minutes of radio and television advertisement should TopAd use?

Let $x_1$ and $x_2$ be the minutes allocated to radio and television advertisements. The goal programming formulation for the problem is given as

$$\text{Minimize } G_1 = s_1^+ \text{ (Satisfy exposure goal)}$$

$$\text{Minimize } G_2 = s_2^- \text{ (Satisfy budget goal)}$$

subject to

$$\begin{aligned}
4x_1 + 8x_2 + s_1^+ - s_1^- \qquad\qquad &= 45 \quad \text{(Exposure goal)}\\
8x_1 + 24x_2 \qquad + s_2^+ - s_2^- &= 100 \quad \text{(Budget goal)}\\
x_1 + 2x_2 \qquad\qquad &\le 10 \quad \text{(Personnel limit)}\\
x_1 \qquad\qquad &\le 6 \quad \text{(Radio limit)}\\
x_1, x_2, s_1^+, s_1^-, s_2^+, s_2^-, &\ge 0
\end{aligned}$$

TopAd's management assumes that the exposure goal is twice as important as the budget goal. The combined objective function thus becomes

$$\text{Minimize } z = 2G_1 + G_2 = 2s_1^+ + s_2^-$$

The optimum solution (obtained by TORA) is

$$z = 10$$

$$x_1 = 5 \text{ minutes}, x_2 = 2.5 \text{ minutes}, s_1^+ = 5 \text{ million persons}$$

All the remaining variables equal zero.

The fact that the optimum value of $z$ is not zero indicates that at least one of the goals is not met. Specifically, $s_1^+ = 5$ means that the exposure goal (of at least 45 million persons) is missed by 5 million individuals. Conversely, the budget goal (of not exceeding \$100,000) is not violated because $s_2^- = 0$.

Goal programming yields only an *efficient* solution to the problem, which is not necessarily optimum. For example, the solution $x_1 = 6$ and $x_2 = 2$ yields the same exposure ($4 \times 6 + 8 \times 2 = 40$ million persons) but costs less ($8 \times 6 + 24 \times 2 =$ \$96,000). In essence, what goal programming does is to find a solution that simply *satisfies* the goals of the model with no regard to optimization. Such "deficiency" in finding an optimum solution raises doubts about the viability of goal programming as an optimizing technique (see Example 8.2-3 for further discussion).

## PROBLEM SET 8.2A

1. Consider Problem 1, Set 8.1a dealing with the Fairville tax situation. Solve the problem, assuming that all five goals have the same weight. Does the solution satisfy all the goals?
2. In Problem 2, Set 8.1a, suppose that the goal of attracting young/middle-aged people is twice as important as either of the other two categories (teens and seniors). Find the associated solution, and check if all the goals have been met.
3. In the Ozark University admission situation described in Problem 3, Set 8.1a, suppose that the limit on the size of the incoming freshman class must be met, but the remaining

requirements can be treated as flexible goals. Further, assume that the ACT score goal is twice as important as any of the remaining goals.

(a) Solve the problem, and specify whether or not all the goals are satisfied.

(b) If, in addition, the size of the incoming class can be treated as a flexible goal that is twice as important as the ACT goal, how would this change affect the solution?

4. In the Circle K model of Problem 4, Set 8.1a, is it possible to satisfy all the nutritional requirements?

5. In Problem 5, Set 8.1a, determine the solution, and specify whether or not the daily production of wheels and seats can be balanced.

6. In Problem 6, Set 8.1a, suppose that the market demand goal is twice as important as that of balancing the two machines, and that no overtime is allowed. Solve the problem, and determine if the goals are met.

7. In Problem 7, Set 8.1a, suppose that the production quota for the two products needs to be met, using overtime if necessary. Find a solution to the problem, and specify the amount of overtime, if any, needed to meet the production quota.

8. In the Vista City Hospital of Problem 8, Set 8.1a, suppose that only the bed limits represent flexible goals and that all the goals have equal weights. Can all the goals be met?

9. The Malco Company has compiled the following table from the files of five of its employees to study the relationship between income and age, education (expressed in number of college years completed), and experience (expressed in number of years in the business).

| Age (yr) | Education (yr) | Experience (yr) | Annual income ($) |
|---|---|---|---|
| 30 | 4 | 5 | 40,000 |
| 39 | 5 | 10 | 48,000 |
| 44 | 2 | 14 | 38,000 |
| 48 | 0 | 18 | 36,000 |
| 37 | 3 | 9 | 41,000 |

Use the goal programming formulation in Problem 10, Set 8.1a to fit the data into the linear equation $y = b_0 + b_1x_1 + b_2x_2 + b_3x_3$.

10. Solve Problem 9 using the Chebyshev Method proposed in Problem 11, Set 8.1a.

### 8.2.2 The Preemptive Method

In the preemptive method, the decision maker must rank the goals of the problem in order of importance. Given an $n$-goal situation, the objectives of the problem are written as

$$\text{Minimize } G_1 = \rho_1 \text{ (Highest priority)}$$
$$\vdots$$
$$\text{Minimize } G_n = \rho_n \text{ (Lowest priority)}$$

The variable $\rho_i$ is either $s_i^+$ or $s_i^-$ representing goal $i$. For example, in the TopAd model (Example 8.2-1), $\rho_1 = s_1^+$ and $\rho_2 = s_2^-$.

The solution procedure considers one goal at a time, starting with the highest priority, $G_1$, and terminating with the lowest, $G_n$. *The process is carried out such that* *the solution obtained from a lower priority goal never degrades any higher priority solutions.*

The literature on goal programming presents a "special" simplex method that guarantees the nondegradation of higher priority solutions. The method uses the **column-dropping rule** that calls for eliminating a *nonbasic* variable $x_j$ with $z_j - c_j \neq 0$ from the optimal tableau of goal $G_k$ before solving the problem of goal $G_{k+1}$. The rule recognizes that such nonbasic variables, if elevated above zero level in the optimization of succeeding goals, can degrade (but never improve) the quality of a higher priority goal. The procedure requires modifying the simplex tableau so that it will carry the objective functions of all the goals of the model.

The proposed column-dropping modification needlessly complicates goal programming. In this presentation, we show that the same results can be achieved in a more straightforward manner using the following steps:

**Step 0.** Identify the goals of the model and rank them in order of priority:

$$G_1 = \rho_1 > G_2 = \rho_2 > \ldots > G_n = \rho_n$$

Set $i = 1$.

**Step i.** Solve $LP_i$ that minimizes $G_i$, and let $\rho_i = \rho_i^*$ define the corresponding optimum value of the deviational variable $\rho_i$. If $i = n$, stop; $LP_n$ solves the $n$-goal program. Otherwise, augment the constraint $\rho_i = \rho_i^*$ to the constraints of the $G_i$-problem to ensure that the value of $\rho_i$ will not be degraded in future problems. Set $i = i + 1$, and repeat step $i$.

The successive addition of the special constraints $\rho_i = \rho_i^*$ may not be as "elegant" theoretically as the *column-dropping rule.* Nevertheless, it achieves the exact same result. More important, it is easier to understand.

Some may argue that the column-dropping rule offers computational advantages. Essentially, the rule makes the problem smaller successively by removing variables, whereas our procedure makes the problem larger by adding new constraints. However, considering the nature of the additional constraints ($\rho_i = \rho_i^*$), we should be able to modify the simplex algorithm to implement the additional constraint implicitly through direct substitution of the variable $\rho_i$. This substitution affects only the constraint in which $\rho_i$ appears and, in effect, reduces the number of variables as we move from one goal to the next. Alternatively, we can use the bounded simplex method of Section 7.3 by replacing $\rho_i = \rho_i^*$ with $\rho_i \leq \rho_i^*$, in which case the additional constraints are accounted for implicitly. In this regard, the column-dropping rule, theoretical appeal aside, does not appear to offer a particular computational advantage. For the sake of completeness, however, we will demonstrate in Example 8.2-3 how the column-dropping rule works.

#### Example 8.2-2

The problem of Example 8.2-1 is solved by the preemptive method. Assume that the exposure goal has a higher priority.

**Step 0.** $G_1 > G_2$

$$G_1\text{: Minimize } s_1^+ \text{ (Satisfy exposure goal)}$$
$$G_2\text{: Minimize } s_2^- \text{ (Satisfy budget goal)}$$

**Step 1.** Solve $LP_1$.

$$\text{Minimize } G_1 = s_1^+$$

subject to

$$\begin{aligned}
4x_1 + 8x_2 + s_1^+ - s_1^- \quad &= 45 \quad \text{(Exposure goal)}\\
8x_1 + 24x_2 \qquad + s_2^+ - s_2^- &= 100 \quad \text{(Budget goal)}\\
x_1 + 2x_2 \quad &\le 10 \quad \text{(Personnel limit)}\\
x_1 \quad &\le 6 \quad \text{(Radio limit)}
\end{aligned}$$

$$x_1, x_2, s_1^+, s_1^-, s_2^+, s_2^- \ge 0$$

The optimum solution (determined by TORA) is $x_1 = 5$ minutes, $x_2 = 2.5$ minutes, $s_1^+ = 5$ million people, with the remaining variables equal to zero. The solution shows that the exposure goal, $G_1$, is violated by 5 million persons.

In $LP_1$, we have $\rho_1 = s_1^+$. Thus, the additional constraint we use with the $G_2$-problem is $s_1^+ = 5$.

**Step 2.** We need to solve $LP_2$, whose objective function is

$$\text{Minimize } G_2 = s_2^-$$

subject to the same set of constraints as in step 1 plus the additional constraint $s_1^+ = 5$. We can solve the new problem by using TORA's MODIFY option to add the constraint $s_1^+ = 5$.

The additional constraint $s_1^+ = 5$ can also be accounted for by substituting out $s_1^+$ in the first constraint. The result is that the right-hand side of the exposure goal constraint will be changed from 45 to 40, thus reducing $LP_2$ to

$$\text{Minimize } G_2 = s_2^-$$

subject to

$$\begin{aligned}
4x_1 + 8x_2 - s_1^- \quad &= 40 \quad \text{(Exposure goal)}\\
8x_1 + 24x_2 \qquad + s_2^+ - s_2^- &= 100 \quad \text{(Budget goal)}\\
x_1 + 2x_2 \quad &\le 10 \quad \text{(Personnel limit)}\\
x_1 \quad &\le 6 \quad \text{(Radio limit)}
\end{aligned}$$

$$x_1, x_2, s_1^-, s_2^+, s_2^- \ge 0$$

The new formulation is one variable less than the one in $LP_1$, which is the general idea advanced by the column-dropping rule.

In reality, the optimization of $LP_2$ is not necessary in this example because the optimum solution to problem $G_1$ already yields $s_2^- = 0$. Hence, the solution of $LP_1$ is automatically optimum for $LP_2$ as well (you can verify this answer by solving $LP_2$ with TORA).

Next, we use an example to show that a better solution for the problem of Example 8.2-2 can be obtained if the preemptive method is used to *optimize* objectives rather than to *satisfy* goals. The example also serves to demonstrate the *column-dropping rule* for solving goal programs.

---

**Example 8.2-3**

The goals of Example 8.2-2 can be restated as

Priority 1: Maximize exposure ($P_1$)

Priority 2: Minimize cost ($P_2$)

Mathematically, the two objectives are given as

$$\text{Maximize } P_1 = 4x_1 + 8x_2 \quad \text{(Exposure)}$$
$$\text{Minimize } P_2 = 8x_1 + 24x_2 \quad \text{(Cost)}$$

The specific goal limits for exposure and cost ( = 45 and 100) are removed because the simplex method will determine them optimally.

The new problem can thus be stated as

$$\text{Maximize } P_1 = 4x_1 + 8x_2$$
$$\text{Minimize } P_2 = 8x_1 + 24x_2$$

subject to

$$x_1 + 2x_2 \le 10$$
$$x_1 \le 6$$
$$x_1, x_2 \ge 0$$

We first solve the problem using the procedure introduced in Example 8.2-2.

**Step 1.** Solve $LP_1$.

$$\text{Maximize } P_1 = 4x_1 + 8x_2$$

subject to

$$x_1 + 2x_2 \le 10$$
$$x_1 \le 6$$
$$x_1, x_2 \ge 0$$

The optimum solution (obtained by TORA) is $x_1 = 0$, $x_2 = 5$ with $P_1 = 40$, which shows that the most exposure we can get is 40 million persons.

**Step 2.** Add the constraint $4x_1 + 8x_2 \ge 40$ to ensure that goal $G_1$ is not degraded. Thus, we solve $LP_2$ as

$$\text{Minimize } P_2 = 8x_1 + 24x_2$$

subject to

$$x_1 + 2x_2 \le 10$$
$$x_1 \qquad \le 6$$
$$4x_1 + 8x_2 \ge 40 \quad \text{(Additional constraint)}$$
$$x_1, x_2 \ge 0$$

The TORA optimum solution of $LP_2$ is $P_2 = \$96{,}000$, $x_1 = 6$ minutes, and $x_2 = 2$ minutes. It yields the same exposure ($P_1 = 40$ million people) but at a smaller cost than the one in Example 8.2-2 where the main objective is to satisfy rather than optimize the goals.

The same problem is solved now by using the column-dropping rule. The rule calls for carrying the objective rows associated with all the goals in the simplex tableau.

**$LP_1$ (Exposure Maximization):** The $LP_1$ simplex tableau carries both objective rows, $P_1$ and $P_2$. The optimality condition applies to the $P_1$-objective row only. The $P_2$-row plays a passive role in $LP_1$, but must be updated with the rest of the simplex tableau in preparation for the optimization of $LP_2$.

$LP_1$ is solved in two iterations as follows:

| Iteration | Basic | $x_1$ | $x_2$ | $s_1$ | $s_2$ | Solution |
|---|---|---|---|---|---|---|
| 1 | $P_1$ | −4 | −8 | 0 | 0 | 0 |
| | $P_2$ | −8 | −24 | 0 | 0 | 0 |
| | $s_1$ | 1 | 2 | 1 | 0 | 10 |
| | $s_2$ | 1 | 0 | 0 | 1 | 6 |
| 2 | $P_1$ | 0 | 0 | 4 | 0 | 40 |
| | $P_2$ | 4 | 0 | 12 | 0 | 120 |
| | $x_2$ | $\frac{1}{2}$ | 1 | $\frac{1}{2}$ | 0 | 5 |
| | $s_2$ | 1 | 0 | 0 | 1 | 6 |

The last tableau yields the optimal solution $x_1 = 0$, $x_2 = 5$, and $P_1 = 40$.

The *column-dropping rule* calls for eliminating any *nonbasic* variable $x_j$ with $z_j - c_j \ne 0$ from the optimum tableau of $LP_1$ before $LP_2$ is optimized. The reason for doing so is that these variables, if left unchecked, could become positive in lower priority optimization problems, which would degrade the quality of higher priority solutions.

**$LP_2$ (Cost Minimization):** The column-dropping rule eliminates $s_1$ (with $z_j - c_j = 4$). We can see from the $P_2$-row that if $s_1$ is not eliminated, it will be the entering variable at the start of the $P_2$-iterations and will yield the optimum solution $x_1 = x_2 = 0$, which will degrade the optimum objective value of the $P_1$-problem from $P_1 = 40$ to $P_1 = 0$. (Try it!)

The $P_2$-problem is of the minimization type. Following the elimination of $s_1$, the variable $x_1$ with $z_j - c_j = 4$ ($>0$) can improve the value of $P_2$. The following table shows the $LP_2$ iterations. The elements of $P_1$-row has been deleted because the row no longer serves a purpose in the optimization of $LP_2$.

| Iteration | Basic | $x_1$ | $x_2$ | $s_1$ | $s_2$ | Solution |
|---|---|---|---|---|---|---|
| 1 | $P_1$ | | | | | 40 |
| | $P_2$ | 4 | 0 | | 0 | 120 |
| | $x_2$ | $\frac{1}{2}$ | 1 | | 0 | 5 |
| | $s_2$ | 1 | 0 | | 1 | 6 |
| 2 | $P_1$ | | | | | 40 |
| | $P_2$ | 0 | 0 | | −4 | 96 |
| | $x_2$ | 0 | 1 | | $-\frac{1}{2}$ | 2 |
| | $x_1$ | 1 | 0 | | 1 | 6 |

The optimum solution ($x_1 = 6, x_2 = 2$) with a total exposure of $P_1 = 40$ and a total cost of $P_2 = 96$ is the same as obtained earlier.

## PROBLEM SET 8.2B

**1.** In Example 8.2-2, suppose that the budget goal is increased to $110,000. The exposure goal remains unchanged at 45 million persons. Show how the preemptive method will reach a solution.

**2.** Solve Problem 1, Set 8.1a (Fairville tax model) using the following priority ordering for the goals: $G_1 > G_2 > G_3 > G_4 > G_5$.

**3.** Consider Problem 2, Set 8.1a, which deals with the presentation of band concerts and art shows at the NW Shopping Mall. Suppose that the goals set for teens, the young/middle-aged group, and seniors are referred to as $G_1$, $G_2$, and $G_3$, respectively. Solve the problem for each of the following priority orders:

**(a)** $G_1 > G_2 > G_3$

**(b)** $G_3 > G_2 > G_1$

Show that the satisfaction of the goals (or lack of it) can be a function of the priority order.

**4.** Solve the Ozark University model (Problem 3, Set 8.1a) using the preemptive method, assuming that the goals are prioritized in the same order given in the problem.

## SELECTED REFERENCES


Cohon, T. L., *Multiobjective Programming and Planning,* Academic Press, New York, 1978.

Ignizio, J. P., and T. M. Cavalier, *Linear Programming,* Prentice-Hall, Upper Saddle River, NJ, 1994.

Steuer, R. E., *Multiple Criteria Optimization: Theory, Computations, and Application,* Wiley, New York, 1986.


## COMPREHENSIVE PROBLEMS

**8.1**[1] The Warehouzer Company manages three sites of forestland for timber production and reforestation with the respective areas of 100,000, 180,000, and 200,000 acres. The main

[1]Based on K. P. Rustagi, *Forest Management Planning for Timber Production: A Goal Programming Approach,* Bulletin No. 89, Yale University Press, New Haven, CT, 1976.

timber products include three categories: pulpwood, plywood, and sawlogs. Several reforestation alternatives are available for each site, each with its cost, number of rotation years (i.e., number of years from seedling size until harvesting), return from rent, and production output. The following table summarizes this information.

| | | Annual $/acre | | | Annual $m^3$/acre | | |
|---|---|---|---|---|---|---|---|
| *Site* | *Alternative* | *Cost* | *Rent* | *Rotation yr* | *Pulpwood* | *Plywood* | *Sawlogs* |
| 1 | *A*1 | 1000 | 160 | 20 | 12 | 0 | 0 |
| | *A*2 | 800 | 117 | 25 | 10 | 0 | 0 |
| | *A*3 | 1500 | 140 | 40 | 5 | 6 | 0 |
| | *A*4 | 1200 | 195 | 15 | 4 | 7 | 0 |
| | *A*5 | 1300 | 182 | 40 | 3 | 0 | 7 |
| | *A*6 | 1200 | 180 | 40 | 2 | 0 | 6 |
| | *A*7 | 1500 | 135 | 50 | 3 | 0 | 5 |
| 2 | *A*1 | 1000 | 102 | 20 | 9 | 0 | 0 |
| | *A*2 | 800 | 55 | 25 | 8 | 0 | 0 |
| | *A*3 | 1500 | 95 | 40 | 2 | 5 | 0 |
| | *A*4 | 1200 | 120 | 15 | 3 | 4 | 0 |
| | *A*5 | 1300 | 100 | 40 | 2 | 0 | 5 |
| | *A*6 | 1200 | 90 | 40 | 2 | 0 | 4 |
| 3 | *A*1 | 1000 | 60 | 20 | 7 | 0 | 0 |
| | *A*2 | 800 | 48 | 25 | 6 | 4 | 0 |
| | *A*3 | 1500 | 60 | 40 | 2 | 0 | 4 |
| | *A*4 | 1200 | 65 | 15 | 2 | 0 | 3 |
| | *A*5 | 1300 | 35 | 40 | 1 | 0 | 5 |

To guarantee sustained future production, each acre of reforestation in each alternative requires that as many acres as years in rotation be assigned to that alternative. The rent column represents the stumpage value per acre.

The goals of Warehouzer are as follows:

1. Annual outputs of pulpwood, plywood, and sawlogs are 200,000, 150,000, and 350,000 cubic meters, respectively.
2. Annual reforestation budget is $2.5 million.
3. Annual return from land rent is $100 per acre.

How much land at each site should be assigned to each alternative?

**8.2** A charity organization runs a children's shelter. The organization relies on volunteer service from 8:00 A.M. until 2:00 P.M. Volunteers may begin work at the start of any hour between 8:00 A.M. and 11:00 A.M. A volunteer works a maximum of 6 hours and a minimum of 2 hours, and no volunteers work during lunch hour between 12:00 noon and 1:00 P.M. The charity has estimated its goal of needed volunteers throughout the day (from 8:00 A.M. to 2:00 P.M., and excluding the lunch hour between 12:00 noon and 1:00 P.M.) as 15, 16, 18, 20, and 16, respectively. The objective is to decide on the number of volunteers that should start at each hour (8:00, 9:00, 10:00, 11:00, and 1:00) such that the given goals are met as much as possible.

C H A P T E R 9

# Integer Linear Programming

Integer linear programs (ILPs) are linear programs in which some or all the variables are restricted to integer (or discrete) values. ILP has important practical applications. Unfortunately, despite decades of extensive research, computational experience with ILP has been less than satisfactory. To date, there does not exist an ILP computer code that can solve integer programming problems consistently.

## 9.1 ILLUSTRATIVE APPLICATIONS

The ILP applications in this section start with simple formulations and then graduate to more complex ones. For convenience, we define a **pure** integer problem as the one in which all the variables are integer. Otherwise, the problem is a **mixed** integer program.

### Example 9.1-1 (Capital Budgeting)

Five projects are being evaluated over a 3-year planning horizon. The following table gives the expected returns for each project and the associated yearly expenditures.

| | Expenditures (million $)/yr | | | |
|---|---|---|---|---|
| *Project* | *1* | *2* | *3* | *Returns (million $)* |
| 1 | 5 | 1 | 8 | 20 |
| 2 | 4 | 7 | 10 | 40 |
| 3 | 3 | 9 | 2 | 20 |
| 4 | 7 | 4 | 1 | 15 |
| 5 | 8 | 6 | 10 | 30 |
| Available funds (million $) | 25 | 25 | 25 | |

Which projects should be selected over the 3-year horizon?

The problem reduces to a "yes-no" decision for each project. Define the binary variable $x_j$ as

$$x_j = \begin{cases} 1, & \text{if project } j \text{ is selected} \\ 0, & \text{if project } j \text{ is not selected} \end{cases}$$

The ILP model is thus given as

$$\text{Maximize } z = 20x_1 + 40x_2 + 20x_3 + 15x_4 + 30x_5$$

subject to

$$\begin{aligned} 5x_1 + 4x_2 + 3x_3 + 7x_4 + 8x_5 &\leq 25 \\ x_1 + 7x_2 + 9x_3 + 4x_4 + 6x_5 &\leq 25 \\ 8x_1 + 10x_2 + 2x_3 + x_4 + 10x_5 &\leq 25 \\ x_1, x_2, x_3, x_4, x_5 &= (0, 1) \end{aligned}$$

The optimum integer solution (obtained by TORA[1]) is $x_1 = x_2 = x_3 = x_4 = 1$, $x_5 = 0$, with $z = 95$ (million \$). The solution shows that all but project 5 must be selected.

It is interesting to compare the continuous LP solution with the ILP solution. The LP optimum, obtained by replacing $x_j = (0, 1)$ with $0 \leq x_j \leq 1$ for all $j$, yields $x_1 = .5789$, $x_2 = x_3 = x_4 = 1$, $x_5 = .7368$, and $z = 108.68$ (million \$). The solution is meaningless because two of the variables assume fractional values. We may *round* the solution to the closest integer values, which yields $x_1 = x_5 = 1$. However, the resulting solution is infeasible because the constraints are violated. More important, the concept of *rounding* should not apply here because $x_j$ represents a "yes-no" decision for which fractional values are meaningless.

## PROBLEM SET 9.1A[2]

1. In the capital budgeting model of Example 9.1–1, suppose that project 5 must be selected if either project 1 or project 3 is selected. Modify the model to include the new restriction and find the optimum solution with TORA.
2. Five items are to be loaded in a vessel. The weight $w_i$ and volume $v_i$ together with the value $r_i$ for item $i$ are tabulated below.

| Item $i$ | Unit weight, $w_i$ (tons) | Unit volume, $v_i$ (yd$^3$) | Unit worth, $r_i$ (100 \$) |
|---|---|---|---|
| 1 | 5 | 1 | 4 |
| 2 | 8 | 8 | 7 |
| 3 | 3 | 6 | 6 |
| 4 | 2 | 5 | 5 |
| 5 | 7 | 4 | 4 |

The maximum allowable cargo weight and volume are 112 tons and 109 yd$^3$, respectively. Formulate the ILP model, and find the most valuable cargo using TORA.

[1]To use TORA, select `Integer Programming` from `Main Menu`. After inputting the problem (file Ch9ToraCapitalBudgetEx9-1-1.txt), go to output screen and select `Automated B&B` to obtain the optimum solution.

[2]Problems 3 to 6 are adapted from Malba Tahan, *El Hombre Que Calculaba,* Editorial Limusa, Mexico City, 1994, pp. 39–182.

3. Suppose that you have 7 full wine bottles, 7 half-full, and 7 empty. You would like to divide the 21 bottles among three individuals so that each will receive exactly 7. Additionally, each individual must receive the same quantity of wine. Express the problem as an ILP constraint equations, and find a solution using TORA. (*Hint:* Use a dummy objective function in which all the objective coefficients are zeros.)
4. An eccentric sheikh left a will to distribute a herd of camels among his three children: Tarek receives at least one-half of the herd, Sharif gets at least one-third, and Maisa gets at least one-ninth. The remainder goes to a charity organization. The will does not specify the size of the herd except to say that it is an odd number of camels and that the named charity receives exactly one camel. How many camels did the sheikh leave in the estate, and how many did each child get?
5. A farm couple is sending their three children to the market to sell 90 apples with the objective of educating them about money and numbers. Karen, the oldest, carries 50 apples; Bill, the middle child, carries 30; and John, the youngest, carries only 10. The parents have stipulated five rules: (a) The selling price is either \$1 for 7 apples or \$3 for 1 apple, or a combination of the two prices; (b) each child may exercise one or both options of the selling price; (c) each of the three children must return with exactly the same amount of money; (d) each child's income must be in whole dollars (no cents allowed); and (e) the amount received by each child must be the largest possible under the stipulated conditions. Given that the three children are able to sell all they have, how can they satisfy their parents' conditions?
6. Once upon a time, there was a captain of a merchant ship who wanted to reward three crew members for their valiant effort in saving the ship's cargo during an unexpected storm in the high seas. The captain put aside a certain sum of money in the purser's office and instructed the first officer to distribute it equally among the three mariners after the ship had reached shore. One night, one of the sailors, unbeknownst to the others, went to the purser's office and decided to claim (an equitable) one-third of the money in advance. After dividing the money into three equal shares, an extra coin remained, which the mariner decided to keep (in addition to one-third of the money). The next night, the second mariner got the same idea and, repeating the same three-way division with what was left, ended up keeping an extra coin as well. The third night, the third mariner also took a third of what was left, plus an extra coin that could not be divided. When the ship reached shore, the first officer divided what was left of the money equally among the three mariners, also to be left with an extra coin. To simplify things, the first officer put the extra coin aside and gave the three mariners their allotted equal shares. How much money was in the safe to start with? Formulate the problem as an ILP, and find the solution using TORA. (*Hint:* The problem has a countably infinite number of integer solutions. For convenience, assume that we are interested in determining the smallest sum of money that satisfies the problem. Then, boosting the resulting solution by 1, augment it as a lower bound and obtain the next smallest solution. Continuing in this manner, a general solution pattern will evolve.)
7. You have the following three-letter words: AFT, FAR, TVA, ADV, JOE, FIN, OSF, and KEN. Suppose that we assign numeric values to the alphabet starting with $A = 1$ and ending with $Z = 26$. Each word is scored by adding the numeric codes of its three letters. For example, AFT has a score of $1 + 6 + 20 = 27$. You are to select five of the given eight words that yield the maximum total score. Simultaneously, the selected five words must satisfy the following conditions:

$$\begin{pmatrix}\text{sum of letter 1}\\ \text{scores}\end{pmatrix} < \begin{pmatrix}\text{sum of letter 2}\\ \text{scores}\end{pmatrix} < \begin{pmatrix}\text{sum of letter 3}\\ \text{scores}\end{pmatrix}$$

Formulate the problem as an ILP, and find the optimum solution using TORA.

8. The Record-a-Song Company has contracted with a rising star to record eight songs. The durations of the different songs are 8, 3, 5, 5, 9, 6, 7, and 12 minutes, respectively. Record-a-Song uses a two-sided cassette tape for the recording. Each side has a capacity of 30 minutes. The company would like to distribute the songs on the two sides in a balanced manner. This means that the length of the songs on each side should be about the same, as much as possible. Formulate the problem as an ILP, and find the optimum solution.
9. In Problem 8, suppose that the nature of the melodies dictates that songs 3 and 4 cannot be recorded on the same side. Formulate the problem as an ILP. Would it be possible to use a 25-minute tape (each side) to record the eight songs? If not, use ILP to determine the minimum tape capacity needed to make the recording.

### Example 9.1-2 (Fixed-Charge Problem)

I have been approached by three telephone companies to subscribe to their long distance service in the United States. MaBell will charge a flat \$16 per month plus \$.25 a minute. PaBell will charge \$25 a month but will reduce the per minute cost to \$.21. As for BabyBell, the flat monthly charge is \$18, and the cost per minute is \$.22. I usually make an average of 200 minutes of long-distance calls a month. Assuming that I do not pay the flat monthly fee unless I make calls and that I can apportion my calls among all three companies as I please, how should I use the three companies to minimize my monthly telephone bill?

This problem can be solved readily without ILP. Nevertheless, it is instructive to formulate it as an integer program.

Define

$$x_1 = \text{MaBell long-distance minutes per month}$$
$$x_2 = \text{PaBell long-distance minutes per month}$$
$$x_3 = \text{BabyBell long-distance minutes per month}$$
$$y_1 = 1 \text{ if } x_1 > 0 \text{ and } 0 \text{ if } x_1 = 0$$
$$y_2 = 1 \text{ if } x_2 > 0 \text{ and } 0 \text{ if } x_2 = 0$$
$$y_3 = 1 \text{ if } x_3 > 0 \text{ and } 0 \text{ if } x_3 = 0$$

We can ensure that $y_j$ will equal 1 if $x_j$ is positive by using the constraint

$$x_j \le My_j,\ j = 1, 2, 3$$

The value of $M$ should be selected sufficiently large so as not to restrict the variables $x_j$ artificially. Because I make about 200 minutes of phone calls a month, then $x_j \le 200$ for all $j$, and it is safe to select $M = 200$.

The complete model is

$$\text{Minimize } z = .25x_1 + .21x_2 + .22x_3 + 16y_1 + 25y_2 + 18y_3$$

subject to

$$x_1 + x_2 + x_3 \ge 200$$
$$x_1 \le 200y_1$$
$$x_2 \le 200y_2$$

$$x_3 \qquad \leq 200y_3$$

$$x_1, x_2, x_3 \geq 0$$

$$y_1, y_2, y_3 = (0, 1)$$

The formulation shows that the $j$th monthly flat fee will be part of the objective function $z$ only if $y_j = 1$, which can happen only if $x_j > 0$ (per the last three constraints of the model). If $x_j = 0$ at the optimum, then the minimization of $z$, together with the fact that the objective coefficient of $y_j$ is strictly positive, will force $y_j$ to equal zero, as desired.

The optimum solution (file Ch9ToraFixedChargeEx9-1-2.txt) yields $x_3 = 200$, $y_3 = 1$, and all the remaining variables are equal to zero, which shows that BabyBell should be selected as my long-distance carrier. Observe that the information conveyed by $y_3 = 1$ is redundant because the same result is implied by $x_3 > 0$ $(= 200)$. Actually, the main reason for using $y_1$, $y_2$, and $y_3$ is to account for the monthly flat fee. In effect, the three binary variables convert an ill-behaved (nonlinear) model into an analytically tractable formulation. This conversion has resulted in introducing the integer (binary) variables in an otherwise continuous problem.

The concept of "flat fee" is typical of what is known in the literature as the **fixed charge problem.**

## PROBLEM SET 9.1B

**1.** Jobco is planning to produce at least 2000 widget on three machines. The minimum lot size on any machine is 500 widget. The following table gives the pertinent data of the situation.

| Machine | Setup cost | Production cost/unit | Capacity (units) |
|---|---|---|---|
| 1 | 300 | 2 | 600 |
| 2 | 100 | 10 | 800 |
| 3 | 200 | 5 | 1200 |

Formulate the problem as an ILP, and find the optimum solution using TORA.

**2.** Oilco is considering two potential drilling sites for reaching four targets (possible oil wells). The following table provides the preparation costs at each of the two sites and the cost of drilling from site $i$ to target $j$ $(i = 1, 2;\ j = 1, 2, 3, 4)$.

| | Drilling cost (million $) to target | | | | |
|---|---|---|---|---|---|
| *Site* | *1* | *2* | *3* | *4* | *Preparation cost (million $)* |
| 1 | 2 | 1 | 8 | 5 | 5 |
| 2 | 4 | 6 | 3 | 1 | 6 |

Formulate the problem as an ILP, and find the optimum solution using TORA.

**3.** Three industrial sites are considered for locating manufacturing plants. The plants send their supplies to three customers. The supply at the plants and the demand at the customers, together with the unit transportation cost from the plants to the customers, are given in the following table.

| | 1 | 2 | 3 | Supply |
|---|---|---|---|---|
| 1 | \$10 | \$15 | \$12 | **1800** |
| 2 | \$17 | \$14 | \$20 | **1400** |
| 3 | \$15 | \$10 | \$11 | **1300** |
| Demand | **1200** | **1700** | **1600** | |

In addition to the transportation costs, fixed costs also are incurred at the rate of \$12,000, \$11,000, and \$12,000 for plants 1, 2, and 3, respectively. Formulate the problem as an ILP and find the optimum solution using TORA.

**4.** Repeat Problem 3 assuming that the demands at each of customers 2 and 3 are changed to 800.

## Example 9.1-3 (Set Covering Problem)

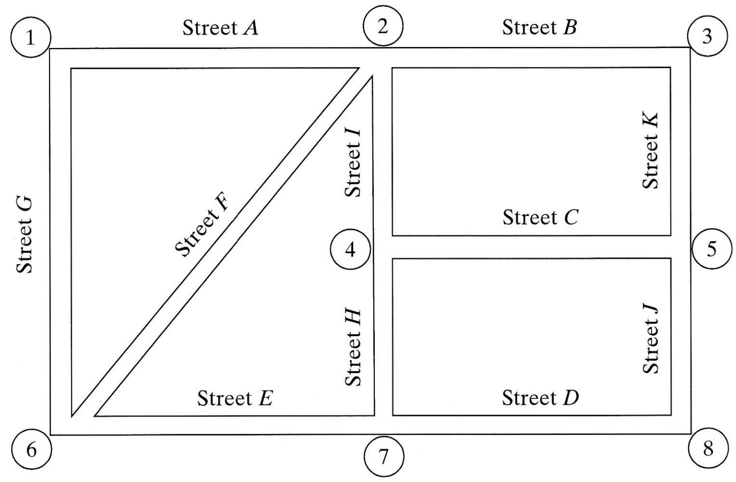
To promote on-campus safety, the U of A Security Department is in the process of installing emergency telephones at selected locations. The department wants to install the minimum number of telephones provided that each of the campus main streets is served by at least one telephone. Figure 9.1 maps the principal streets (A to K) on campus.

It is logical to place the telephones at the intersections of streets so that each telephone will serve at least two streets. Figure 9.1 shows that the layout of the streets requires a maximum of eight telephone locations.

Define

$$x_j = \begin{cases} 1, & \text{a telephone is installed in location } j \\ 0, & \text{otherwise} \end{cases}$$

FIGURE 9.1
Street map of the U of A campus



The constraints of the problem require installing at least one telephone on each of the 11 streets (A to K). Thus, the model becomes

$$\text{Minimize } z = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8$$

subject to

$$\begin{array}{llllllllll}
x_1 + x_2 & & & & & & & \geq 1 & \text{(Street } A\text{)} \\
& x_2 + x_3 & & & & & & \geq 1 & \text{(Street } B\text{)} \\
& & & x_4 + x_5 & & & & \geq 1 & \text{(Street } C\text{)} \\
& & & & & & x_7 + x_8 & \geq 1 & \text{(Street } D\text{)} \\
& & & & & x_6 + x_7 & & \geq 1 & \text{(Street } E\text{)} \\
& x_2 & & & & + x_6 & & \geq 1 & \text{(Street } F\text{)} \\
x_1 & & & & & + x_6 & & \geq 1 & \text{(Street } G\text{)} \\
& & & x_4 & & & + x_7 & \geq 1 & \text{(Street } H\text{)} \\
& x_2 & & + x_4 & & & & \geq 1 & \text{(Street } I\text{)} \\
& & & & x_5 & & + x_8 & \geq 1 & \text{(Street } J\text{)} \\
& & x_3 & & + x_5 & & & \geq 1 & \text{(Street } K\text{)}
\end{array}$$

$$x_j = (0, 1),\ j = 1, 2, \ldots, 8$$

The optimum solution of the problem (obtained by TORA, file Ch9ToraSetCover Ex9-1-3.txt) requires installing four telephones at intersections 1, 2, 5, and 7. The problem has alternative optima.

---

The preceding model is typical of what is generically known as the **set covering problem**. In this model, all the variables are binary. For each constraint, all the left-hand-side coefficients are 0 or 1, and the right-hand side is of the form ($\geq 1$). The objective function always minimizes $c_1x_1 + c_2x_2 + \ldots + c_nx_n$, where $c_j > 0$ for all $j = 1, 2, \ldots, n$. In the present example, $c_j = 1$ for all $j$. However, if $c_j$ represents the installation cost in location $j$, then these coefficients may assume values other than 1.

## PROBLEM SET 9.1C

1. ABC is an LTL trucking company that delivers loads on a daily basis to five customers. The following table provides the customers associated with each route:

| Route | Customers |
|---|---|
| 1 | 1, 2, 3, 4 |
| 2 | 4, 3, 5 |
| 3 | 1, 2, 5 |
| 4 | 2, 3, 5 |
| 5 | 1, 4, 2 |
| 6 | 1, 3, 5 |

The segments of each route are dictated by the capacity of the truck delivering the loads. For example, on route 1, the capacity of the truck is sufficient to deliver the loads to customers 1, 2, 3, and 4 only. The following table lists distances (in miles) among the truck terminal (ABC) and the five customers.

| | ABC | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| ABC | 0 | 10 | 12 | 16 | 9 | 8 |
| 1 | 10 | 0 | 32 | 8 | 17 | 10 |
| 2 | 12 | 32 | 0 | 14 | 21 | 20 |
| 3 | 16 | 8 | 14 | 0 | 15 | 18 |
| 4 | 9 | 17 | 21 | 15 | 0 | 11 |
| 5 | 8 | 10 | 20 | 18 | 11 | 0 |

The objective is to determine the least distance needed to make the daily deliveries to all five customers. Though the solution may result in a customer being served by more than one route, the implementation phase will use only one such route. Formulate the problem as an ILP and solve using TORA.

2. The U of A is in the process of forming a committee to handle the students' grievances. The directive received from the administration is to include at least one female, one male, one student, one administrator, and one faculty member. Ten individuals (identified, for simplicity, by the letters $a$ to $j$) have been nominated. The mix of these individuals in the different categories is given as follows:

| Category | Individuals |
|---|---|
| Females | $a, b, c, d, e$ |
| Males | $f, g, h, i, j$ |
| Students | $a, b, c, j$ |
| Administrators | $e, f$ |
| Faculty | $d, g, h, i$ |

The U of A wants to form the smallest committee with representation from each of the five categories. Formulate the problem as an ILP, and find the optimum solution using TORA.

3. Washington County includes six towns that need emergency ambulance service. Because of the proximity of some of the towns, a single station may serve more than one community. The stipulation is that the station must be within 15 minutes of driving time from the towns it serves. The table below gives the driving times in minutes among the six towns.

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 0 | 23 | 14 | 18 | 10 | 32 |
| 2 | 23 | 0 | 24 | 13 | 22 | 11 |
| 3 | 14 | 24 | 0 | 60 | 19 | 20 |
| 4 | 18 | 13 | 60 | 0 | 55 | 17 |
| 5 | 10 | 22 | 19 | 55 | 0 | 12 |
| 6 | 32 | 11 | 20 | 17 | 12 | 0 |

Formulate an ILP whose solution will produce the smallest number of stations and their locations. Find the solution using TORA.

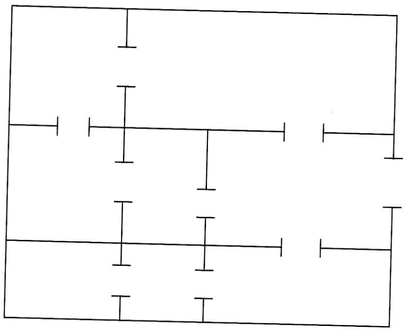


FIGURE 9.2
Museum layout for Problem 4, Set 9.1c

4. The treasures of King Tut are on display in a museum in New Orleans. The layout of the museum is shown in Figure 9.2, with the different rooms joined by open doors. A guard standing at a door can watch two adjoining rooms. The museum wants to ensure guard presence in every room, using the minimum number possible. Formulate the problem as an ILP, and find the optimum solution with TORA.

### Example 9.1-4 (Either-or Constraints)

Jobco uses a single machine to process three jobs. Both the processing time and the due date (in days) for each job are given in the following table. The due dates are measured from the zero datum, the assumed start time of the first job.

| Job | Processing time (days) | Due date (days) | Late penalty $/day |
|---|---|---|---|
| 1 | 5 | 25 | 19 |
| 2 | 20 | 22 | 12 |
| 3 | 15 | 35 | 34 |

The objective of the problem is to determine the minimum late-penalty sequence for processing the three jobs.

Define

$$x_j = \text{Start date in days for job } j \text{ (measured from the zero datum)}$$

The problem has two types of constraints: The noninterference constraints (guaranteeing that jobs are not processed concurrently) and the due date constraints. Consider the noninterference constraints first.

Two jobs $i$ and $j$ with processing time $p_i$ and $p_j$ will not be processed concurrently if either $x_i \geq x_j + p_j$ or $x_j \geq x_i + p_i$, depending on whether job $j$ precedes job $i$, or vice versa. Because all mathematical programs deal with *simultaneous* constraints only, we transform the either-or constraints by introducing the following auxiliary binary variable:

$$y_{ij} = \begin{cases} 1, & \text{if } i \text{ precedes } j \\ 0, & \text{if } j \text{ precedes } i \end{cases}$$

For $M$ sufficiently large, the *either-or* constraint is converted to the following *simultaneous* constraints

$$My_{ij} + (x_i - x_j) \geq p_j \text{ and } M(1 - y_{ij}) + (x_j - x_i) \geq p_i$$

The conversion guarantees that only one of the two constraints can be active at any one time. If $y_{ij} = 0$, the first constraint is active, and the second is redundant (because its left-hand side will include $M$, which is much larger than $p_i$). If $y_{ij} = 1$, the first constraint is redundant, and the second is active.

Next, the due date constraint is considered. Given $d_j$ is the due date for job $j$, let $s_j$ be an unrestricted variable. Then, the associated constraint is

$$x_j + p_j + s_j = d_j$$

If $s_j \geq 0$, the due date is met, and if $s_j < 0$, a late penalty is incurred. Using the substitution

$$s_j = s_j^+ - s_j^-,\ s_j^+,\ s_j^- \geq 0$$

the constraint becomes

$$x_j + s_j^+ - s_j^- = d_j - p_j$$

The late penalty cost is proportional to $s_j^-$.

The model for the given problem is

$$\text{Minimize } z = 19s_1^- + 12s_2^- + 34s_3^-$$

subject to

$$\begin{array}{lllllll}
x_1 - x_2 & + My_{12} & & & & & \geq 20 \\
-x_1 + x_2 & - My_{12} & & & & & \geq 5 - M \\
x_1 \quad - x_3 & & + My_{13} & & & & \geq 15 \\
-x_1 \quad + x_3 & & - My_{13} & & & & \geq 5 - M \\
\quad x_2 - x_3 & & & + My_{23} & & & \geq 15 \\
\quad - x_2 + x_3 & & & - My_{23} & & & \geq 20 - M \\
x_1 & & & & + s_1^+ - s_1^- & & = 25 - 5 \\
\quad x_2 & & & & & + s_2^+ - s_2^- & = 22 - 20 \\
\quad\quad x_3 & & & & & + s_3^+ - s_3^- & = 35 - 15
\end{array}$$

$$x_1, x_2, x_3, s_1^+, s_1^-, s_2^+, s_2^-, s_3^+, s_3^- \geq 0$$

$$y_{12}, y_{13}, y_{23} = (0, 1)$$

The integer variables— $y_{12}$, $y_{13}$, and $y_{23}$ —are introduced to convert the either-or constraints into simultaneous constraints. The resulting model is a *mixed* ILP.

To solve the model, we choose $M = 1000$, a value that is larger than the sum of the processing times for all three activities.

The optimal solution (obtained by TORA, file Ch9ToraEitherOrEx9-1-4.txt[3]) is $x_1 = 20$, $x_2, = 0$, and $x_3 = 25$. This means that job 2 starts at time 0, job 1 starts at time 20, and job 3 starts at time 25, thus yielding the optimal processing sequence 2 → 1 → 3. The solution calls for completing job 2 at time $0 + 20 = 20$, job 1 at time $= 20 + 5 = 25$, and job 3 at $25 + 15 = 40$ days. Job 3 is delayed by $40 - 35 = 5$ days past its due date at a cost of $5 \times \$34 = \$170$.

---

[3]Because TORA does not accept a negative right-hand side, the variable (RHS-), whose value is always 1, assumes the role of the right-hand side of the constraints.

## PROBLEM SET 9.1D

1. A game board consists of nine equal squares. You are required to fill each square with a number between 1 and 9 such that the sum of the numbers in each row, each column, and each diagonal equals 15. Use ILP to determine the number in each square such that no two adjacent numbers in any row, column, or diagonal are equal. Solve with TORA.
2. A machine is used to produce two interchangeable products. The daily capacity of the machine can produce at most 20 units of product 1 and 10 units of product 2. Alternatively, the machine can be adjusted to produce at most 12 units of product 1 and 22 units of product 2 daily. Market analysis shows that the maximum daily demand for the two products combined is 35 units. Given that the unit profits for the two respective products are \$10 and \$12, which of the two machine settings should be selected? Formulate the problem as an ILP, and find the optimum using TORA. (*Note:* This two-dimensional problem can be solved by inspecting the graphical solution space. This is not the case for the $n$-dimensional problem.)
3. Gapco manufactures three products, whose daily labor and raw material requirements are given in the following table.

| Product | Required daily labor (hr/unit) | Required daily raw material (lb/unit) |
|---|---|---|
| 1 | 3 | 4 |
| 2 | 4 | 3 |
| 3 | 5 | 6 |

The profits per unit of the three products are \$25, \$30, and \$22, respectively. Gapco has two options for locating its plant. The two locations differ primarily in the availability of labor and raw material as shown in the following table:

| Location | Available daily labor (hr) | Available daily raw material (lb) |
|---|---|---|
| 1 | 100 | 100 |
| 2 | 90 | 120 |

Formulate the problem as a mixed ILP, and use TORA to determine the optimum location of the plant.

4. Consider the job-shop scheduling problem that produces two end products using a single machine. The precedence relationships among the eight operations are summarized in Figure 9.3. Let $p_j$ be the processing time for operations $j$ $(=1, 2, \ldots, n)$. The due dates, measured from the zero datum, for products 1 and 2, are $d_1$ and $d_2$, respectively. An operation, once started, must be completed before another starts. Formulate the problem as a mixed ILP.

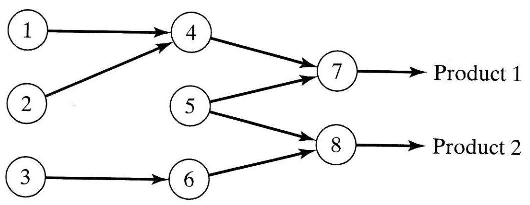



FIGURE 9.3

Precedence relationships for the job-shop situation of Problem 4, Set 9.1d

5. Jaco owns a plant in which three products are manufactured. The labor and raw material requirements for the three products are given in the following table.

| Product | Required daily labor (hr/unit) | Required daily raw material (lb/unit) |
|---|---|---|
| 1 | 3 | 4 |
| 2 | 4 | 3 |
| 3 | 5 | 6 |
| Daily availability | 100 | 100 |

The profits per unit for the three products are \$25, \$30, and \$45, respectively. If product 3 is to be manufactured at all, then its production level must be at least 5 units daily. Formulate the problem as a mixed ILP, and find the optimal mix using TORA.

6. Show how the nonconvex shaded solution spaces in Figure 9.4 can be represented by a set of simultaneous constraints. Then use TORA to find the optimum solution that maximizes $z = 2x_1 + 3x_2$ subject to the solution space given in (a).

7. Suppose that it is required that *any k* out of the following *m* constraints must be active:

$$g_i\ (x_1, x_2, \ldots, x_n) \leq b_i,\ i = 1, 2, \ldots, m$$

Show how this condition may be represented.

8. In the following constraint, the right-hand side may assume one of the values, $b_1, b_2, \ldots$, and $b_m$.

$$g(x_1, x_2, \ldots, x_n) \leq b_1, b_2, \ldots, \text{ or } b_m$$

Show how this condition is represented.

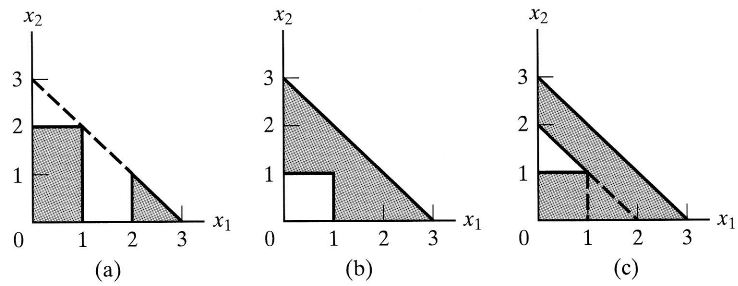



FIGURE 9.4

Solution spaces for Problem 6, Set 9.1d

9.2.1

## 9.2 INTEGER PROGRAMMING ALGORITHMS

The ILP algorithms are based on exploiting the tremendous computational success of LP. The strategy of these algorithms involves three steps.

**Step 1.** Relax the solution space of the ILP by deleting the integer restriction on all integer variables and replacing any binary variable $y$ with the continuous range $0 \leq y \leq 1$. The result of the relaxation is a regular LP.

**Step 2.** Solve the LP, and identify its continuous optimum.

**Step 3.** Starting from the continuous optimum point, add special constraints that iteratively modify the LP solution space in a manner that will eventually render an optimum extreme point satisfying the integer requirements.

Two general methods have been developed for generating the special constraints in step 3.

1. Branch-and-bound (B&B) method
2. Cutting plane method

Although neither method is consistently effective computationally, experience shows that the B&B method is far more successful than the cutting plane method. This point is discussed further in this chapter.

### 9.2.1 Branch-and-Bound (B&B) Algorithm

The first B&B algorithm was developed in 1960 by A. Land and G. Doig for the general mixed and pure ILP problem. Later, in 1965, E. Balas developed the **additive algorithm** for solving ILP problems with pure binary (zero or one) variables. The additive algorithm computations were so simple (mainly addition and subtraction) that it was hailed as a possible breakthrough in the solution of general ILP.[4] Unfortunately, the algorithm failed to produce the desired computational advantages. Moreover, the algorithm, which initially appeared unrelated to the B&B technique, was shown to be but a special case of the general Land and Doig algorithm.

This section will present the general Land-Doig B&B algorithm only. A numeric example is used to explain the details.

#### Example 9.2-1

$$\text{Maximize } z = 5x_1 + 4x_2$$

subject to

$$x_1 + x_2 \le 5$$

$$10x_1 + 6x_2 \le 45$$

$$x_1, x_2 \text{ nonnegative integer}$$

The lattice points (dots) in Figure 9.5 define ILP solution space. The associated LP problem, LP0, is defined by removing the integer restrictions. Its optimum solution is $x_1 = 3.75$, $x_2 = 1.25$, and $z = 23.75$.

Because the optimum LP0 solution does not satisfy the integer requirements, the B&B algorithm modifies the solution space in a manner that eventually identifies the

[4] A general ILP can be expressed in terms of binary (0-1) variables as follows. Given an integer variable $x$ with a finite upper bound $u$ (i.e., $0 \le x \le u$), then

$$x = 2^0 y_0 + 2^1 y_1 + 2^2 y_2 + \dots + 2^k y_k$$

The variables $y_0, y_1, \dots$, and $y_k$ are binary, and the index $k$ is the smallest integer satisfying $2^{k+1} - 1 \ge u$.

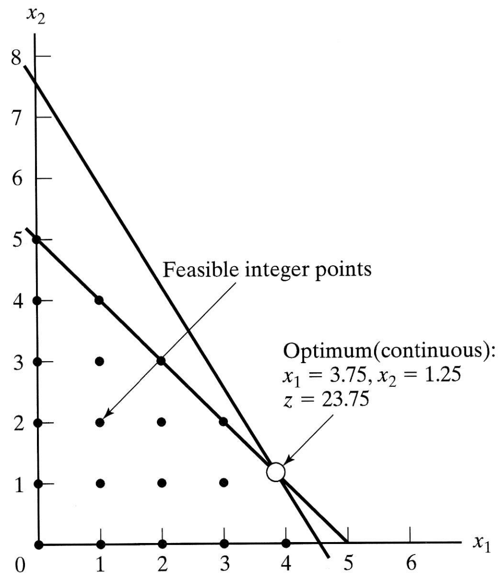



FIGURE 9.5
ILP solution space of Example 9.2-1

ILP optimum. First, we select one of the integer variables whose optimum value at LP0 is not integer. Selecting $x_1$ (=3.75) arbitrarily, the region $3 < x_1 < 4$ of the LP0 solution space contains no integer values of $x_1$ and can be eliminated as nonpromising. This is equivalent to replacing the original LP0 with two new LPs, LP1 and LP2, defined as

$$\text{LP1 space} = \text{LP0 space} + (x_1 \leq 3)$$

$$\text{LP2 space} = \text{LP0 space} + (x_1 \geq 4)$$

Figure 9.6 depicts the LP1 and LP2 spaces. The two spaces contain the same feasible integer points of the original ILP, which means that, from the standpoint of the integer solution, dealing with LP1 and LP2 is the same as dealing with the original LP0.

If we *intelligently* continue to remove the regions that do not include integer solutions by imposing the appropriate constraints (e.g., $3 < x_1 < 4$ at LP0), we will eventually produce LPs whose optimum extreme points satisfy the integer restrictions. In effect, we will be solving the ILP by dealing with a succession of (continuous) LPs.

The new restrictions, $x_1 \leq 3$ and $x_1 \geq 4$, are mutually exclusive, so that LP1 and LP2 must be dealt with as separate LPs as Figure 9.7 shows. This dichotomization gives rise to the concept of **branching** in the B&B algorithm with $x_1$ being the **branching variable**.

The optimum ILP lies in either LP1 or LP2. Hence, both subproblems must be examined. We arbitrarily examine LP1 (associated with $x_1 \leq 3$) first.

$$\text{Maximize } z = 5x_1 + 4x_2$$

subject to

$$\begin{aligned} x_1 + x_2 &\leq 5 \\ 10x_1 + 6x_2 &\leq 45 \\ x_1 &\leq 3 \\ x_1, x_2 &\geq 0 \end{aligned}$$

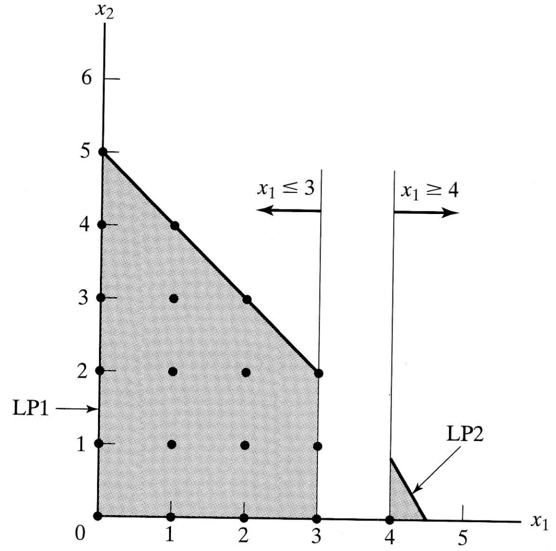



FIGURE 9.6

Solution spaces of LP1 and LP2 for Example 9.2-1

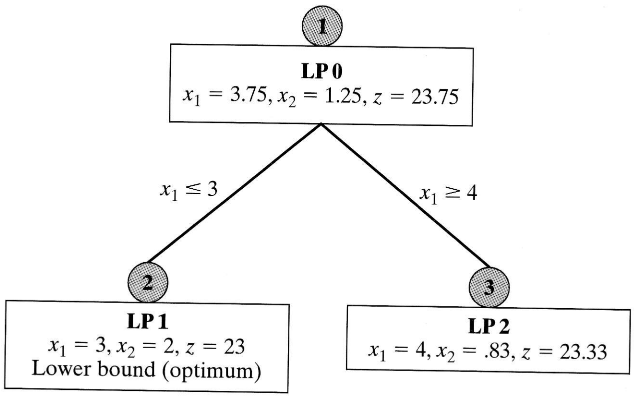



FIGURE 9.7

Using branching variable $x_1$ to create LP1 and LP2 for Example 9.2-1

The solution of LP1 (which can be solved efficiently by the upper-bounded algorithm of Section 7.3) yields the optimum solution

$$x_1 = 3,\ x_2 = 2, \text{ and } z = 23$$

The LP1 solution satisfies the integer requirements for $x_1$ and $x_2$. Hence, LP1 is said to be **fathomed**. This means that LP1 need not be investigated any further because it cannot yield any *better* ILP solution.

We cannot at this point say that the integer solution obtained from LP1 is optimum for the original problem because LP2 may yield a better integer solution (with a higher value of $z$). All we can say is that $z = 23$ is a **lower bound** on the optimum (maximum) objective value of the original ILP. This means that any unexamined subproblem that cannot yield a better objective value than the lower bound must be discarded as nonpromising. If an unexamined subproblem produces a better integer solution, then the lower bound must be updated accordingly.

Given the lower bound $z = 23$, we examine LP2 (the only remaining unexamined subproblem). Because optimum $z = 23.75$ at LP0 *and all the coefficients of the objective function happen to be integers*, it is impossible that LP2 (which is more restrictive than LP0) will produce a better integer solution. As a result, we discard LP2 and conclude that it has been *fathomed.*

The B&B algorithm is now complete because both LP1 and LP2 have been examined and fathomed (the first for producing an integer solution and the second for showing that it cannot produce a better integer solution). We thus conclude that the optimum ILP solution is the one associated with the lower bound—namely, $x_1 = 3$, $x_2 = 2$, and $z = 23$.

Two questions remain unanswered regarding the procedure:

1. At LP0, could we have selected $x_2$ as the *branching variable* in place of $x_1$?
2. When selecting the next subproblem to be examined, could we have solved LP2 first instead of LP1?

The answer to both questions is "yes." However, ensuing computations could differ dramatically. Figure 9.8, in which LP2 is examined first, illustrates this point. The optimum LP2 solution is $x_1 = 4$, $x_2 = .83$, and $z = 23.33$ (verify using TORA LP module). Because $x_2$ (=.83) is noninteger, LP2 is investigated further by creating subproblems LP3 and LP4 using the branches $x_2 \leq 0$ and $x_2 \geq 1$, respectively. This means that

$$\begin{aligned}\text{LP3 space} &= \text{LP2 space} + (x_2 \leq 0)\\ &= \text{LP0 space} + (x_1 \geq 4) + (x_2 \leq 0)\\ \text{LP4 space} &= \text{LP2 space} + (x_2 \geq 1)\\ &= \text{LP0 space} + (x_1 \geq 4) + (x_2 \geq 1)\end{aligned}$$

We have three "dangling" subproblems that must be examined: LP1, LP3, and LP4. Suppose that we arbitrarily examine LP4 first. LP4 has no solution, and hence it is fathomed. Next, let us examine LP3. The optimum solution is $x_1 = 4.5$, $x_2 = 0$, and $z = 22.5$. The noninteger value of $x_1$ (=4.5) leads to the two branches $x_1 \leq 4$ and $x_1 \geq 5$, and the creation of subproblems LP5 and LP6 from LP3.

$$\text{LP5 space} = \text{LP0 space} + (x_1 \geq 4) + (x_2 \leq 0) + (x_1 \leq 4) \equiv \text{LP0 space} + (x_1 = 4) + (x_2 \leq 0)$$

$$\text{LP6 space} = \text{LP0 space} + (x_1 \geq 4) + (x_2 \leq 0) + (x_1 \geq 5) \equiv \text{LP0 space} + (x_1 \geq 5) + (x_2 \leq 0)$$

Now, subproblems LP1, LP5, and LP6 remain unexamined. LP6 is fathomed because it has no feasible solution. Next, LP5 has the integer solution ($x_1 = 4$, $x_2 = 0$, $z = 20$) and, hence, yields a lower bound ($z = 20$) on the optimum ILP solution. We are left with subproblem LP1, whose solution yields a better integer ($x_1 = 3$, $x_2 = 2$, $z = 23$). Thus, the lower bound is updated to $z = 23$. Because *all* the subproblems have been fathomed, the optimum solution is associated with the most up-to-date lower bound—namely, $x_1 = 3$, $x_2 = 2$, and $z = 23$.

The solution sequence in Figure 9.8 (LP0 $\rightarrow$ LP2 $\rightarrow$ LP4 $\rightarrow$ LP3 $\rightarrow$ LP6 $\rightarrow$ LP5 $\rightarrow$ LP1) is a worst-case scenario that, nevertheless, may occur in practice. The example points to a principal weakness of the B&B algorithm: How do we select the next subproblem to be examined, and how do we choose its branching variable?

In Figure 9.7, we were lucky to "stumble" upon a good lower bound at the very first subproblem, LP1, thus allowing us to fathom LP2 without further computations and to terminate the B&B search. In essence, we completed the procedure by solving

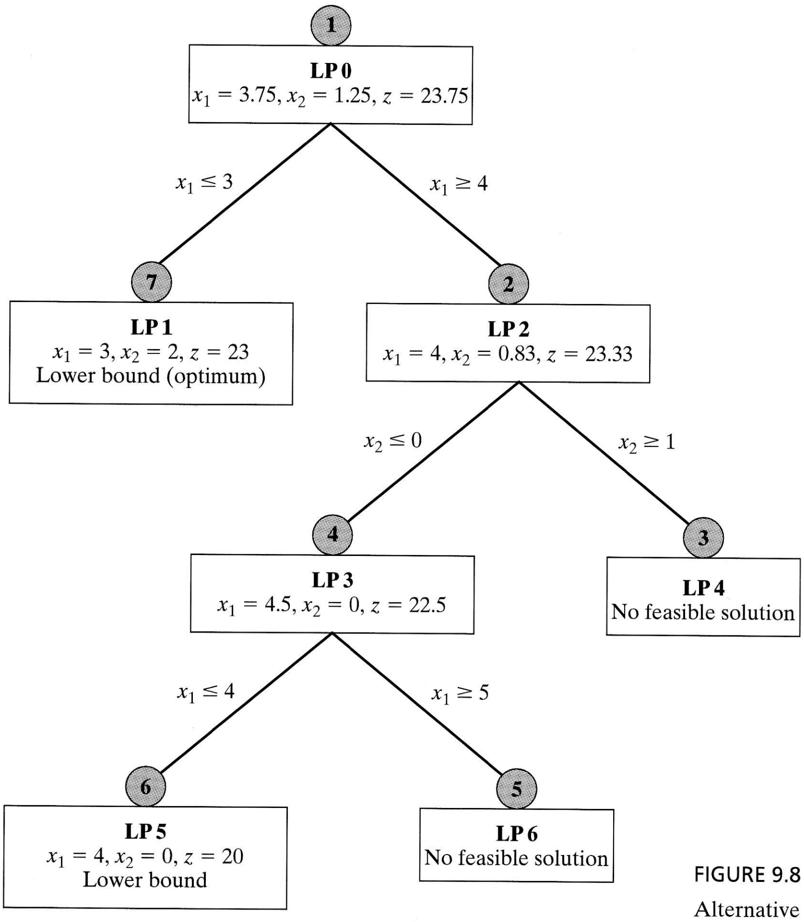



FIGURE 9.8

Alternative B&B tree for Example 9.2-1

one subproblem only. In Figure 9.8, we had to examine seven subproblems before the B&B algorithm could be terminated. Although there are heuristics for enhancing the ability of B&B to "guess" which branch can lead to an improved ILP solution (see Taha, 1975, pp. 154–171), there is no solid theory that will always yield consistent results, and herein lies the difficulty that plagues computations in ILP. Indeed in Section 9.2.2, Problem 1, Set 9.2b, demonstrates with the help of TORA the bizarre behavior of the B&B algorithm, even for a small 16-variable 1-constraint problem, where the optimum is found in 9 iterations (subproblems) but requires over 25,000 iterations to verify optimality. It is no wonder that to this day, and after four decades of research, available computer codes (commercial and academic alike) lack consistency (a la simplex method) in solving ILPs.

---

We now summarize the B&B algorithm. Assuming a maximization problem, set an initial lower bound $\underline{z} = -\infty$ on the optimum objective value of ILP. Set $i = 0$.

**Step 1.** (Fathoming/bounding). Select LP$i$, the next subproblem to be examined. Solve LP$i$, and attempt to fathom it using one of three conditions.

**(a)** The optimal $z$-value of LP$i$ cannot yield a better objective value than the current lower bound.

**(b)** LP$i$ yields a better feasible integer solution than the current lower bound.

**(c)** LP$i$ has no feasible solution.

Two cases will arise.

**(a)** If LP$i$ is fathomed and a better solution is found, update the lower bound. If *all* subproblems have been fathomed, stop; the optimum ILP is associated with the current lower bound, if any. Otherwise, set $i = i + 1$, and repeat step 1.

**(b)** If LP$i$ is not fathomed, go to step 2 for branching.

**Step 2.** (Branching). Select one of the integer variables $x_j$, whose optimum value $x_j^*$ in the LP$i$ solution is not integer. Eliminate the region

$$[x_j^*] < x_j < [x_j^*] + 1$$

(where $[v]$ defines the largest integer $\leq v$) by creating two LP subproblems that correspond to

$$x_j \leq [x_j^*] \text{ and } x_j \geq [x_j^*] + 1$$

Set $i = i + 1$, and go to step 1.

The given steps apply to maximization problems. For minimization, we replace the lower bound with an upper bound (whose initial value is $z = +\infty$).

The B&B algorithm can be extended directly to mixed problems (in which only some of the variables are integer). If a variable is continuous, we simply never select it as a branching variable. A feasible subproblem provides a new bound on the objective value if the values of the discrete variables are integer and the objective value is improved relative to the current bound.

## PROBLEM SET 9.2A

1. Solve the ILP of Example 9.2-1 by the B&B algorithm starting with $x_2$ as the branching variable. Solve the subproblems with TORA using the MODIFY option for the upper and lower bounds. Start the procedure by solving the subproblem associated with $x_2 \leq [x_2^*]$.
2. Develop the B&B tree for each of the following problems. For convenience, always select $x_1$ as the branching variable at node 0.

**(a)** Maximize $z = 3x_1 + 2x_2$

subject to

$$2x_1 + 5x_2 \leq 9$$
$$4x_1 + 2x_2 \leq 9$$
$$x_1, x_2 \geq 0 \text{ and integer}$$

**(b)** Maximize $z = 2x_1 + 3x_2$

9.2.2

subject to

$$5x_1 + 7x_2 \leq 35$$
$$4x_1 + 9x_2 \leq 36$$
$$x_1, x_2 \geq 0 \text{ and integer}$$

**(c)** Maximize $z = x_1 + x_2$
subject to

$$2x_1 + 5x_2 \leq 16$$
$$6x_1 + 5x_2 \leq 27$$
$$x_1, x_2 \geq 0 \text{ and integer}$$

**(d)** Minimize $z = 5x_1 + 4x_2$
subject to

$$3x_1 + 2x_2 \geq 5$$
$$2x_1 + 3x_2 \geq 7$$
$$x_1, x_2 \geq 0 \text{ and integer}$$

**(e)** Maximize $z = 5x_1 + 7x_2$
subject to

$$2x_1 + x_2 \leq 13$$
$$5x_1 + 9x_2 \leq 41$$
$$x_1, x_2 \geq 0 \text{ and integer}$$

**3.** Repeat Problem 2, assuming that $x_1$ is continuous.

**4.** Show graphically that the following ILP has no feasible solution, and then verify the result using B&B.

$$\text{Maximize } z = 2x_1 + x_2$$

subject to

$$10x_1 + 10x_2 \leq 9$$
$$10x_1 + 5x_2 \geq 1$$
$$x_1, x_2 \geq 0 \text{ and integer}$$

**5.** Solve the following problems by B&B.

$$\text{Maximize } z = 18x_1 + 14x_2 + 8x_3 + 4x_4$$

subject to

$$15x_1 + 12x_2 + 7x_3 + 4x_4 + x_5 \leq 37$$
$$x_1, x_2, x_3, x_4, x_5 = (0, 1)$$

### 9.2.2 TORA-Generated B&B Tree

TORA integer programming module is equipped with a facility for generating the B&B tree interactively. To use this facility, select `User-guided B&B` in the output screen

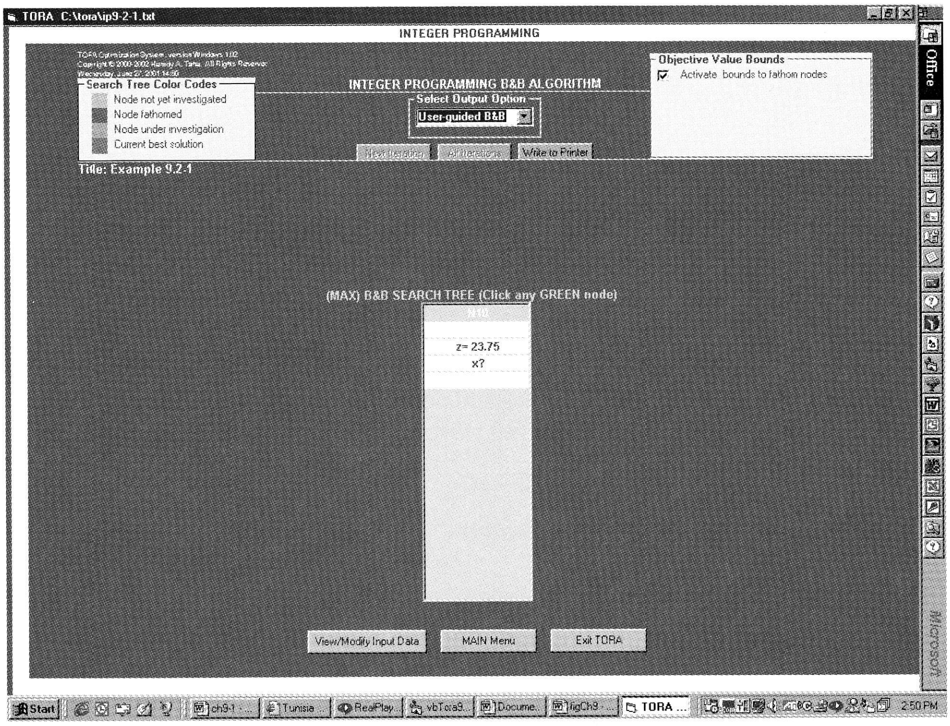



FIGURE 9.9

Starting solution of the B&B tree in Example 9.2-1

of the integer programming module. The resulting screen provides all the information needed to create the B&B tree. Figure 9.9 shows the layout of the screen representing the root of the search tree, N10, which corresponds to LP0 in Figure 9.6 (file ch9ToraB &BEx9-2-1.txt). Each node is identified by two digits, prefixed with the letter N. The left digit identifies the grid row in which the node resides, and the right digit gives a unique numeric value within the same row. Thus, N10 in Figure 9.9 shows that node 0 is situated in row 1 (which is the only node in this row). TORA limits the number of subproblems per row to 10. The reasoning is that once this limit is reached, the tutorial nature of the interactive procedure becomes unwieldy. A message indicating that the algorithm is reverting to automatic mode is given whenever the number of subproblems per row exceeds 10. Keep in mind that in the automated mode, no limit is set in any way on the number of generated subproblems.

The screen is now set for the selection of the branching variable by clicking any node tagged with "x?" Such nodes are highlighted in green. If you click anywhere in the entries of the node, the associated solution is exposed in the area on top of the B&B tree, as shown in Figure 9.10 where the solution of N10 shows that $x_1 = 3.75$ and $x_2 = 1.25$. It also points out which variables are restricted to integer values. Clicking on either variable automatically creates two subproblems that correspond to the selected

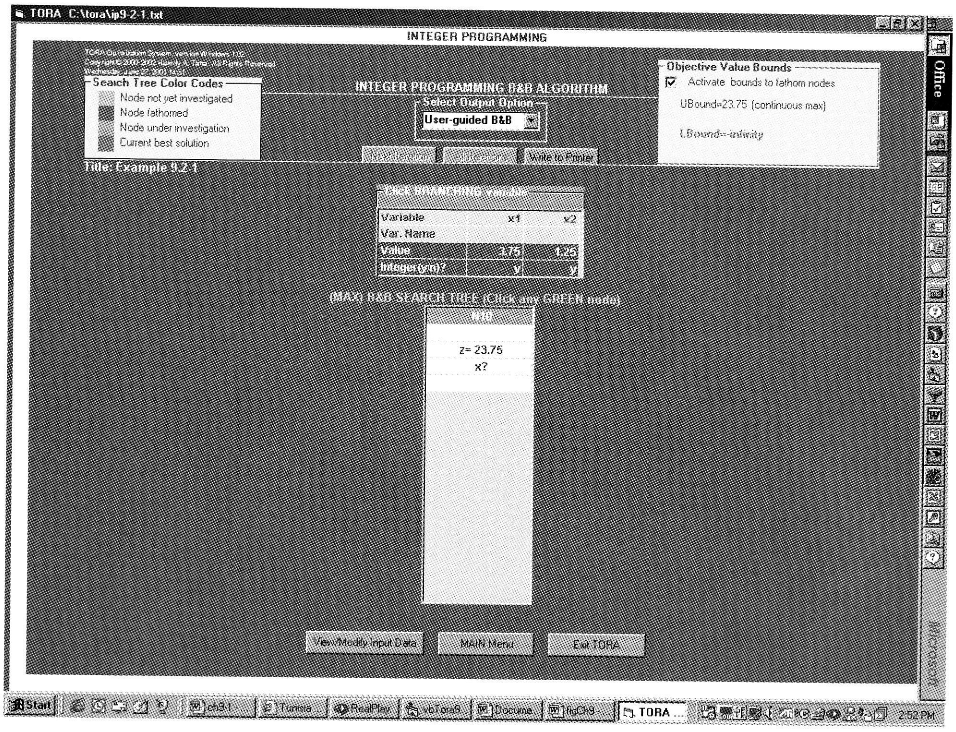



FIGURE 9.10

Selection of the branching variable from the starting solution of Example 9.2-1

branching variable. Figure 9.11 shows the result of selecting $x_1$ as the branching variable at N10. Node N20 (corresponding to $x_1 \leq 3$) and N21 (corresponding to $x_1 \geq 4$) are added to the tree. Node N20 yields an integer solution, and hence it is fathomed. A fathomed node is highlighted in red or magenta. The magenta color is used if the fathomed node provides the current best lower bound, as is the case with node N20. Node N21 has not yet been fathomed, and clicking it will create further nodes in row 3 of the tree. The process continues until all nodes have been fathomed (highlighted in red or magenta).

The top right box in the output screen automatically keeps track of the upper and lower bounds for the problem. The default calls for activating the bounds to fathom the nodes. TORA will automatically discard subproblems whose objective value violates the current bounds. However, you can deactivate the bounds (i.e., remove check in box) to create the entire search tree. In this case, a node is fathomed only if it yields an integer solution or if it is infeasible.

It is important to note that in the search, TORA's automated B&B mode is coded to generate and scan subproblems on a strict LIFO basis. For this reason, most likely the user-guided search may lead to a more efficient search tree, mainly because the user invokes good judgment in selecting the next node to be investigated.

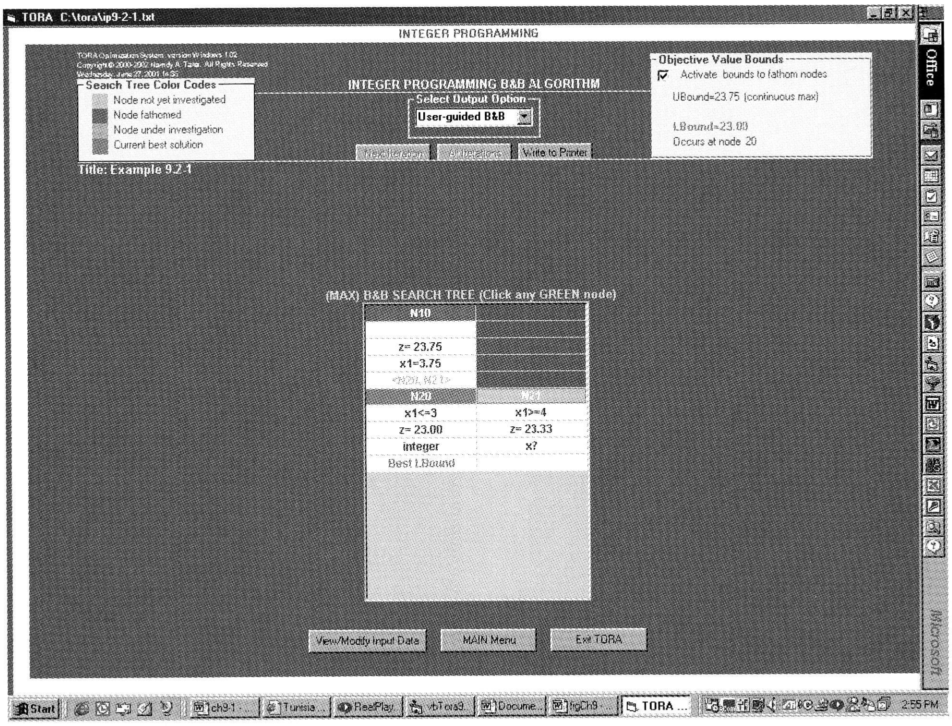



FIGURE 9.11

Creation of the first two subproblems in the B&B tree of Example 9.2-1

## PROBLEM SET 9.2B

1. The following problem is designed to demonstrate the bizarre behavior of the B&B algorithm even for small problems. In particular, note how many subproblems are examined before the optimum is found and how many are needed to verify optimality.

$$\text{Minimize } y$$

subject to

$$2(x_1 + x_2 + \cdots + x_{15}) + y = 15$$

$$\text{All variables are } (0, 1)$$

Use the automated option of TORA to answer the following:

**(a)** How many subproblems are solved before the optimal solution is found?

**(b)** How many subproblems are solved before the optimality of the solution found in (a) is verified?

2. Consider the following ILP:

$$\text{Maximize } z = 18x_1 + 14x_2 + 8x_3$$

subject to

$$15x_1 + 12x_2 + 7x_3 \leq 43$$

$$x_1, x_2, x_3 \text{ nonnegative integers}$$

Use TORA's B&B user-guided option to generate the search tree with and without activating the objective value bound. What is the impact of activating the objective value bound on the number of generated subproblems? For consistency, always select the branching variable as the one with the lowest index and investigate all the subproblems in a current row from left to right before moving to the next row.

**3.** Reconsider Problem 2 above. Convert the problem into an equivalent 0-1 ILP, and then solve it with TORA's automated option. Compare the size of the search trees in the two problems.

**4.** In the following 0-1 ILP use TORA's user-guided option to generate the associated search tree. In each case, show how $z$-bound is used to fathom subproblems.

$$\text{Maximize } z = 3x_1 + 2x_2 - 5x_3 - 2x_4 + 3x_5$$

subject to

$$\begin{aligned} x_1 + x_2 + x_3 + 2x_4 + x_5 &\leq 4 \\ 7x_1 + 3x_3 - 4x_4 + 3x_5 &\leq 8 \\ 11x_1 - 6x_2 + 3x_4 - 3x_5 &\geq 3 \\ x_1, x_2, x_3, x_4, x_5 &= (0, 1) \end{aligned}$$

**5.** Show by using TORA's user-guided option that the following problem has no feasible solution.

$$\text{Maximize } z = 2x_1 + x_2$$

subject to

$$\begin{aligned} 10x_1 + 10x_2 &\leq 9 \\ 10x_1 + 5x_2 &\geq 1 \\ x_1, x_2 &= (0, 1) \end{aligned}$$

**6.** Use TORA's user-guided option to generate the B&B tree associated with the following mixed ILP problem and give the optimum solution.

$$\text{Maximize } z = x_1 + 2x_2 - 3x_3$$

subject to

$$\begin{aligned} 3x_1 + 4x_2 - x_3 &\leq 10 \\ 2x_1 - 3x_2 + 4x_3 &\leq 20 \end{aligned}$$

$x_1, x_2$ nonnegative integers

$$x_3 \geq 0$$

**7.** Use TORA to generate the B&B tree for the following problem assuming that only one of the two constraints holds.

$$\text{Maximize } z = x_1 + 2x_2 - 3x_3$$

subject to

$$\begin{aligned} 20x_1 + 15x_2 - x_3 &\leq 10 \\ 12x_1 + 3x_2 + 4x_3 &\leq 13 \\ x_1, x_2, x_3 &\geq 0 \end{aligned}$$

**8.** Convert the following problem into a mixed ILP, and then use TORA to generate its B&B tree. What is the optimal solution?

$$\text{Maximize } z = x_1 + 2x_2 + 5x_3$$

subject to

$$|-x_1 + 10x_2 - 3x_3| \geq 15$$
$$2x_1 + x_2 + x_3 \leq 10$$
$$x_1, x_2, x_3 \geq 0$$

### 9.2.3 Cutting Plane Algorithm

As in the B&B algorithm, the cutting plane algorithm also starts at the continuous optimum LP solution. Special constraints (called **cuts**) are added to the solution space in a manner that renders an optimum integer extreme point. In Example 9.2-2, we first demonstrate graphically how cuts are used to produce an integer solution and then implement the idea algebraically.

#### Example 9.2-2

Consider the following ILP.

$$\text{Maximize } z = 7x_1 + 10x_2$$

subject to

$$-x_1 + 3x_2 \leq 6$$
$$7x_1 + x_2 \leq 35$$
$$x_1, x_2 \geq 0 \text{ and integer}$$

The cutting plane algorithm modifies the solution space by adding *cuts* that produce an optimum integer extreme point. Figure 9.12 gives an example of two such cuts. Initially, we start with the continuous LP optimum $z = 66\frac{1}{2}$, $x_1 = 4\frac{1}{2}$, $x_2 = 3\frac{1}{2}$ . Next, we add cut I, which produces the (continuous) LP optimum solution $z = 62$, $x_1 = 4\frac{4}{7}$, $x_2 = 3$. Then, we add cut II, which, together with cut I and the original constraints, pro-

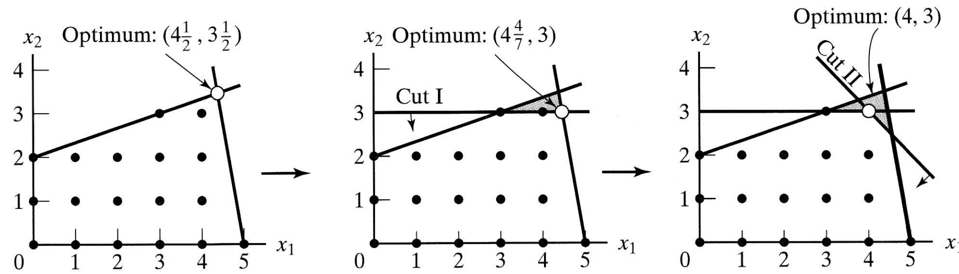



FIGURE 9.12

Illustration of the use of cuts in ILP

duces the LP optimum $z = 58$, $x_1 = 4$, $x_2 = 3$. The last solution is all integer, as desired.

The added cuts do not eliminate any of the original feasible integer points, but must pass through at least one feasible or infeasible integer point. These are basic requirements of any cut.

It is purely accidental that a 2-variable problem used exactly 2 cuts to reach the optimum integer solution. In general, the number of cuts, though finite, is independent of the size of the problem, in the sense that a problem with a small number of variables and constraints may require more cuts than a larger problem.

Next, we use the same example to show how the cuts are constructed and implemented algebraically.

Given the slacks $x_3$ and $x_4$ for constraints 1 and 2, the optimum LP tableau is given as

| Basic | $x_1$ | $x_2$ | $x_3$ | $x_4$ | Solution |
|---|---|---|---|---|---|
| $z$ | 0 | 0 | $\frac{63}{22}$ | $\frac{31}{22}$ | $66\frac{1}{2}$ |
| $x_2$ | 0 | 1 | $\frac{7}{22}$ | $\frac{1}{22}$ | $3\frac{1}{2}$ |
| $x_1$ | 1 | 0 | $-\frac{1}{22}$ | $\frac{3}{22}$ | $4\frac{1}{2}$ |

The optimum continuous solution is $z = 66\frac{1}{2}$, $x_1 = 4\frac{1}{2}$, $x_2 = 3\frac{1}{2}$, $x_3 = 0$, $x_4 = 0$. The cut is developed under the assumption that *all* the variables (including the slacks $x_3$ and $x_4$ ) are integer. Note also that because all the original objective coefficients are integer in this example, the value of $z$ is integer as well.

The information in the optimum tableau can be written explicitly as

$$z + \tfrac{63}{22}x_3 + \tfrac{31}{22}x_4 = 66\tfrac{1}{2} \quad (z\text{-equation})$$

$$x_2 + \tfrac{7}{22}x_3 + \tfrac{1}{22}x_4 = 3\tfrac{1}{2} \quad (x_2\text{-equation})$$

$$x_1 - \tfrac{1}{22}x_3 + \tfrac{3}{22}x_4 = 4\tfrac{1}{2} \quad (x_1\text{-equation})$$

A constraint equation can be used as a **source row** for generating a cut provided its right-hand side is fractional. We also note that the $z$-equation can be used as a source row because $z$ happens to be integer in this example. We will demonstrate how a cut is generated from each of these source rows, starting with the $z$-equation.

First, we factor out all the noninteger coefficients of the equation into an integer value and a fractional component, *provided that the resulting fractional component is strictly positive.* For example,

$$\tfrac{5}{2} = (2 + \tfrac{1}{2})$$

$$-\tfrac{7}{3} = (-3 + \tfrac{2}{3})$$

The factoring of the $z$-equation yields

$$z + (2 + \tfrac{19}{22})x_3 + (1 + \tfrac{9}{22})x_4 = (66 + \tfrac{1}{2})$$

Moving all the integer components to the left-hand side and all the fractional components to the right-hand side, we get

$$z + 2x_3 + 1x_4 - 66 = -\tfrac{19}{22}x_3 - \tfrac{9}{22}x_4 + \tfrac{1}{2} \qquad (1)$$

Because $x_3$ and $x_4$ are nonnegative and all fractions are originally strictly positive, the right-hand side must satisfy the following inequality:

$$-\tfrac{19}{22}x_3 - \tfrac{9}{22}x_4 + \tfrac{1}{2} \le \tfrac{1}{2} \qquad (2)$$

Next, because the left-hand side in Equation (1), $z + 2x_3 + 1x_4 - 66$, is an integer value by construction, the right-hand side, $-\frac{19}{22}x_3 - \frac{9}{22}x_4 + \frac{1}{2}$, must also be integer. It then follows that (2) can be replaced with the inequality:

$$-\tfrac{19}{22}x_3 - \tfrac{9}{22}x_4 + \tfrac{1}{2} \le 0$$

This is the desired cut, and it represents a necessary condition for obtaining an integer solution. It is also referred to as the **fractional cut** because all its coefficients are fractions.

Because $x_3 = x_4 = 0$ in the continuous LP tableau given above, the current continuous optimum violates the cut (because it yields $\frac{1}{2} \le 0$). Thus, if we add this cut to the optimum tableau, the resulting optimum extreme point moves the solution toward satisfying the integer requirements.

Before showing how a cut is implemented in the optimal tableau, we will demonstrate how cuts can also be constructed from the constraint equations. Consider the $x_1$-row:

$$x_1 - \tfrac{1}{22}x_3 + \tfrac{3}{22}x_4 = 4\tfrac{1}{2}$$

Factoring the equation yields

$$x_1 + (-1 + \tfrac{21}{22})x_3 + (0 + \tfrac{3}{22})x_4 = (4 + \tfrac{1}{2})$$

The associated cut is

$$-\tfrac{21}{22}x_2 - \tfrac{3}{22}x_4 + \tfrac{1}{2} \le 0$$

Similarly, the $x_2$-equation

$$x_2 + \tfrac{7}{22}x_3 + \tfrac{1}{22}x_4 = 3\tfrac{1}{2}$$

is factored as

$$x_2 + (0 + \tfrac{7}{22})x_3 + (0 + \tfrac{1}{22})x_4 = 3 + \tfrac{1}{2}$$

Hence, the associated cut is given as

$$-\tfrac{7}{22}x_3 - \tfrac{1}{22}x_4 + \tfrac{1}{2} \le 0$$

Any of the three cuts given above can be used in the first iteration of the cutting plane algorithm. As such, it is not necessary to generate all three cuts before selecting one.

Arbitrarily selecting the cut generated from the $x_2$-row, we can write it in equation form as

$$-\tfrac{7}{22}x_3 - \tfrac{1}{22}x_4 + s_1 = -\tfrac{1}{2},\ s_1 \ge 0 \qquad \text{(Cut I)}$$

This constraint is added as a secondary constraint to the LP optimum tableau as follows:

| Basic | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $s_1$ | Solution |
|---|---|---|---|---|---|---|
| $z$ | 0 | 0 | $\frac{63}{22}$ | $\frac{31}{22}$ | 0 | $66\frac{1}{2}$ |
| $x_2$ | 0 | 1 | $\frac{7}{22}$ | $\frac{1}{22}$ | 0 | $3\frac{1}{2}$ |
| $x_1$ | 1 | 0 | $-\frac{1}{22}$ | $\frac{3}{22}$ | 0 | $4\frac{1}{2}$ |
| $s_1$ | 0 | 0 | $-\frac{7}{22}$ | $-\frac{1}{22}$ | 1 | $-\frac{1}{2}$ |

The tableau is optimal but infeasible. We apply the dual simplex method (Section 4.4) to recover feasibility, which yields

| Basic | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $s_1$ | Solution |
|---|---|---|---|---|---|---|
| $z$ | 0 | 0 | 0 | 1 | 9 | 62 |
| $x_2$ | 0 | 1 | 0 | 0 | 1 | 3 |
| $x_1$ | 1 | 0 | 0 | $\frac{1}{7}$ | $-\frac{1}{7}$ | $4\frac{4}{7}$ |
| $x_3$ | 0 | 0 | 1 | $\frac{1}{7}$ | $-\frac{22}{7}$ | $1\frac{4}{7}$ |

The last solution is still noninteger in $x_1$ and $x_3$. Let us arbitrarily select $x_1$ as the next source row—that is,

$$x_1 + (0 + \tfrac{1}{7})x_4 + (-1 + \tfrac{6}{7})s_1 = 4 + \tfrac{4}{7}$$

The associated cut is

$$-\tfrac{1}{7}x_4 - \tfrac{6}{7}s_1 + s_2 = -\tfrac{4}{7},\ s_2 \geq 0 \qquad \text{(Cut II)}$$

| Basic | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $s_1$ | $s_2$ | Solution |
|---|---|---|---|---|---|---|---|
| $z$ | 0 | 0 | 0 | 1 | 9 | 0 | 62 |
| $x_2$ | 0 | 1 | 0 | 0 | 1 | 0 | 3 |
| $x_1$ | 1 | 0 | 0 | $\frac{1}{7}$ | $-\frac{1}{7}$ | 0 | $4\frac{4}{7}$ |
| $x_3$ | 0 | 0 | 1 | $\frac{1}{7}$ | $-\frac{22}{7}$ | 0 | $1\frac{4}{7}$ |
| $s_2$ | 0 | 0 | 0 | $-\frac{1}{7}$ | $-\frac{6}{7}$ | 1 | $-\frac{4}{7}$ |

The dual simplex method yields the following tableau:

| Basic | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $s_1$ | $s_2$ | Solution |
|---|---|---|---|---|---|---|---|
| $z$ | 0 | 0 | 0 | 0 | 3 | 7 | 58 |
| $x_2$ | 0 | 1 | 0 | 0 | 1 | 0 | 3 |
| $x_1$ | 1 | 0 | 0 | 0 | −1 | 1 | 4 |
| $x_3$ | 0 | 0 | 1 | 0 | −4 | 1 | 1 |
| $x_4$ | 0 | 0 | 0 | 1 | 6 | −7 | 4 |

The optimum solution ($x_1 = 4$, $x_2 = 3$, $z = 58$) is all integer. It is not accidental that all the coefficients of the last tableau are integers. This is a typical property of the implementation of the fractional cut.

It is important to point out that the fractional cut assumes that *all* the variables, *including slack and surplus,* are integer. This means that the cut deals with pure integer problems only. The importance of this assumption is illustrated by an example.

Consider the constraint

$$x_1 + \tfrac{1}{3}x_2 \leq \tfrac{13}{2}$$

$$x_1, x_2 \geq 0 \text{ and integer}$$

From the standpoint of solving the associated ILP, the constraint is treated as an equation by using the nonnegative slack $s_1$—that is,

$$x_1 + \tfrac{1}{3}x_2 + s_1 = \tfrac{13}{2}$$

The application of the fractional cut assumes that the constraint has a feasible integer solution in all $x_1$, $x_2$ and $s_1$. However, the equation above will have a feasible integer solution in $x_1$ and $x_2$ *only if* $s_1$ *is noninteger.* This means that cutting-plane algorithm will show that the problem has no feasible integer solution, even though the variables of concern, $x_1$ and $x_2$, can assume integer feasible values.

There are two ways to remedy this situation.

**1.** Multiply the entire constraint by a proper constant to remove all the fractions. For example, multiplying the constraint above by 6, we get

$$6x_1 + 2x_2 \leq 39$$

Any integer solution of $x_1$ and $x_2$ automatically yields integer slack. However, this type of conversion is appropriate for only simple constraints because the magnitudes of the integer coefficients may become excessively large in some cases.

**2.** Use a special cut, called the **mixed cut**, which allows only a subset of variables to assume integer values, with all the other variables (including slack and surplus) remaining continuous. The details of this cut will not be presented in this chapter (see Taha, 1975, pp. 198–202).

## PROBLEM SET 9.2C

9.2

1. In Example 9.2-2, show graphically whether or not each of the following constraints can form a legitimate cut:
   **(a)** $x_1 + 2x_2 \leq 10$
   **(b)** $2x_1 + x_2 \leq 10$
   **(c)** $3x_2 \leq 10$
   **(d)** $3x_1 + x_2 \leq 15$
2. In Example 9.2-2, show graphically how the following two (legitimate) cuts can lead to the optimum integer solution:

$$x_1 + 2x_2 \leq 10 \quad \text{(Cut I)}$$
$$3x_1 + x_2 \leq 15 \quad \text{(Cut II)}$$

3. Express cuts I and II of Example 9.2-2 in terms of $x_1$ and $x_2$, and show that they are the same ones used graphically in Figure 9.12.
4. In Example 9.2-2, derive cut II from the $x_3$-row of the tableau resulting from the application of cut I. Use the new cut to complete the solution of the example.
5. Show that, even though the following problem has a feasible integer solution in $x_1$ and $x_2$, the fractional cut would not yield a feasible solution unless all the fractions in the constraint have been eliminated.

$$\text{Maximize } z = x_1 + 2x_2$$

subject to

$$x_1 + \tfrac{1}{2}x_2 \leq \tfrac{13}{4}$$
$$x_1, x_2 \geq 0 \text{ and integer}$$

6. Solve the following problems by the fractional cut, and compare the true optimum integer solution with the solution obtained by rounding the continuous optimum.

**(a)** Maximize $z = 4x_1 + 6x_2 + 2x_3$

subject to

$$\begin{aligned} 4x_1 - 4x_2 \quad\quad &\leq 5 \\ -x_1 + 6x_2 \quad\quad &\leq 5 \\ -x_1 + x_2 + x_3 &\leq 5 \\ x_1, x_2, x_3 \geq 0 &\text{ and integer} \end{aligned}$$

**(b)** Maximize $z = 3x_1 + x_2 + 3x_3$

subject to

$$\begin{aligned} -x_1 + 2x_2 + x_3 &\leq 4 \\ 4x_2 - 3x_3 &\leq 2 \\ x_1 - 3x_2 + 2x_3 &\leq 3 \\ x_1, x_2, x_3 \geq 0 &\text{ and integer} \end{aligned}$$

### 9.2.4 Computational Considerations in ILP

To date, and despite over 40 years of research, there does not exist a computer code that can solve ILP consistently. Nevertheless, of the two solution algorithms presented in this chapter, B&B is more reliable. Indeed, practically all commercial ILP codes are B&B-based. Cutting plane methods are generally difficult and uncertain, and the roundoff error presents a serious problem. Though attempts have been made to improve the cutting plane computational efficacy, the end results are not encouraging. In most cases, the cutting plane method is used in a secondary capacity to improve B&B performance at each subproblem.

The most important factor affecting computations in integer programming is the number of integer variables and the feasible range in which they apply. Because available algorithms are not consistent in producing a numeric ILP solution, it may be advantageous computationally to reduce the number of integer variables in the ILP model as much as possible. The following suggestions may prove helpful:

1. Approximate integer variables by continuous ones wherever possible.
2. For the integer variables, restrict their feasible ranges as much as possible.
3. Avoid the use of nonlinearity in the model.

The importance of the integer problem in practice is not yet matched by reliable solution algorithms. It is unlikely that a new theoretical breakthrough will be achieved in the area of integer programming. Instead, new technological advances in computers (such as parallel processing) may offer the best hope for improving the efficiency of ILP codes.

## 9.3 SOLUTION OF THE TRAVELING SALESPERSON PROBLEM

In the obvious sense, the traveling salesperson problem deals with finding the shortest (closed) tour in an $n$-city situation where each city is visited exactly once. The problem, in essence, is an assignment model with additional restrictions that guarantee the exclusion of subtours in the optimum solution. Specifically, in an $n$-city situation, define

$$x_{ij} = \begin{cases} 1, & \text{if city } j \text{ is reached from city } i \\ 0, & \text{otherwise} \end{cases}$$

Given $d_{ij}$ is the distance from city $i$ to city $j$, the traveling salesperson model is given as

$$\text{Minimize } z = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij},\ d_{ij} = \infty \text{ for } i = j$$

subject to

$$\sum_{j=1}^{n} x_{ij} = 1,\ i = 1, 2, \ldots, n \qquad (1)$$

$$\sum_{i=1}^{n} x_{ij} = 1,\ j = 1, 2, \ldots, n \qquad (2)$$

$$x_{ij} = (0, 1) \qquad (3)$$

$$\text{Solution forms a tour} \qquad (4)$$

Constraints (1), (2), and (3) define a regular assignment model (Section 5.4). In general, the assignment problem will produce subtour solutions rather than a complete tour that encompasses all $n$ cities. Figure 9.13 demonstrates a 5-city problem. The arcs represent two-way routes. The figure also illustrates a tour and a subtour solution of the associated assignment model. If the assignments form a tour solution, then it is optimum. Otherwise, additional restrictions are added to the assignment model to remove the subtours. The use of these restrictions is given later in this section.

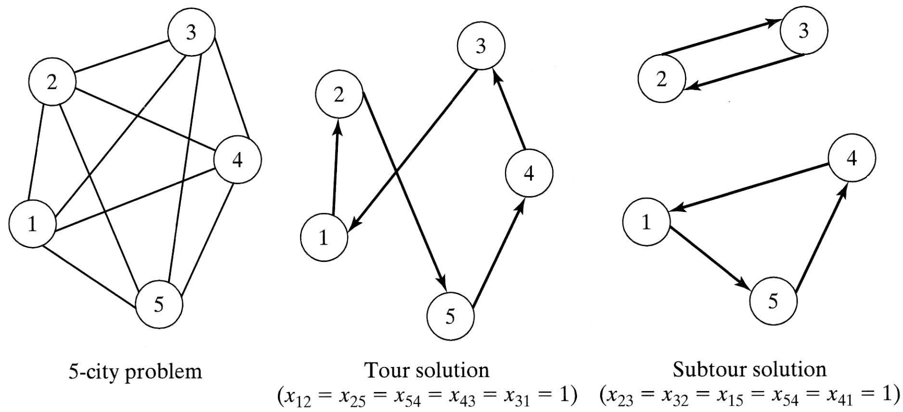



FIGURE 9.13

A 5-city traveling salesperson example with tour and subtour solutions of the associated assignment model

Available solution methods for the traveling salesperson problem are rooted in the ideas of the general B&B or cutting plane algorithms presented in Section 9.2. Before presenting these algorithms, we give an example that demonstrates the versatility of the traveling salesperson model in representing other practical situations (see also Problem Set 9.3a).

### Example 9.3-1

The daily production schedule at the Rainbow Company includes batches of white ($W$), yellow ($Y$), red ($R$), and black ($B$) paints. Because Rainbow uses the same facilities for all four types of paint, proper cleaning between batches is necessary. The following table summarizes the cleanup time in minutes where the row-designated color is followed by the column-designated color. For example, when white is followed by yellow, the cleanup time is 10 minutes. Because a color cannot follow itself, the corresponding entries are assigned infinite setup time. Determine the optimal sequencing for the daily production of the four colors that will minimize the associated total cleanup time.

| | Cleanup min given next paint is | | | |
|---|---|---|---|---|
| *Current paint* | *White* | *Yellow* | *Black* | *Red* |
| White | $\infty$ | 10 | 17 | 15 |
| Yellow | 20 | $\infty$ | 19 | 18 |
| Black | 50 | 44 | $\infty$ | 25 |
| Red | 45 | 40 | 20 | $\infty$ |

Each paint is thought of as a "city" where the "distances" represent the cleanup time needed to switch from one paint batch to the next. The situation thus reduces to determining the *shortest loop* that starts with one paint batch and passes through each of the remaining three paint batches exactly once before returning back to the starting paint.

We can solve this problem by exhaustively enumerating the six $[(4 - 1)! = 3! = 6]$ possible loops of the network. The following table shows that $W \rightarrow Y \rightarrow R \rightarrow B \rightarrow W$ is the optimum loop.

| Production loop | Total cleanup time |
|---|---|
| $W \rightarrow Y \rightarrow B \rightarrow R \rightarrow W$ | $10 + 19 + 25 + 45 = 99$ |
| $W \rightarrow Y \rightarrow R \rightarrow B \rightarrow W$ | $10 + 18 + 20 + 50 = 98$ |
| $W \rightarrow B \rightarrow Y \rightarrow R \rightarrow W$ | $17 + 44 + 18 + 45 = 124$ |
| $W \rightarrow B \rightarrow R \rightarrow Y \rightarrow W$ | $17 + 25 + 40 + 20 = 102$ |
| $W \rightarrow R \rightarrow B \rightarrow Y \rightarrow W$ | $15 + 20 + 44 + 20 = 99$ |
| $W \rightarrow R \rightarrow Y \rightarrow B \rightarrow W$ | $15 + 40 + 19 + 50 = 124$ |

Exhaustive enumeration of the loops is not practical in general. Even a modest-sized 11-city problem will require enumerating $10! = 3{,}628{,}800$ tours, a demanding task indeed. For this reason, the problem must be formulated and solved in a different manner, as we will show later in this section.

To develop the assignment-based formulation for the paint problem, define

$$x_{ij} = 1 \text{ if paint } j \text{ follows paint } i \text{ and zero otherwise}$$

Letting $M$ be a sufficiently large positive value, we can formulate the Rainbow problem as

$$\text{Minimize } z = Mx_{WW} + 10x_{WY} + 17x_{WB} + 15x_{WR} + 20x_{YW} + Mx_{YY} + 19x_{YB} + 18x_{YR}$$
$$+50x_{BW} + 44x_{BY} + Mx_{BB} + 25x_{BR} + 45x_{RW} + 40x_{RY} + 20x_{RB} + Mx_{RR}$$

subject to

$$x_{WW} + x_{WY} + x_{WB} + x_{WR} = 1$$
$$x_{YW} + x_{YY} + x_{YB} + x_{YR} = 1$$
$$x_{BW} + x_{BY} + x_{BB} + x_{BR} = 1$$
$$x_{RW} + x_{RY} + x_{RB} + x_{RR} = 1$$
$$x_{WW} + x_{YW} + x_{BW} + x_{RW} = 1$$
$$x_{WY} + x_{YY} + x_{BY} + x_{RY} = 1$$
$$x_{WB} + x_{YB} + x_{BB} + x_{RB} = 1$$
$$x_{WR} + x_{YR} + x_{BR} + x_{RR} = 1$$
$$x_{ij} = (0, 1) \qquad \text{for all } i \text{ and } j$$

Solution is a tour (loop)

The use of $M$ in the objective function guarantees that a paint job cannot follow itself.

## PROBLEM SET 9.3A

**1.** A manager has a total of 10 employees working on six projects. There are overlaps among the assignments as the following table shows:

| | | Project | | | | | |
|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 |
| | 1 | | x | | x | x | |
| | 2 | x | | x | | x | |
| | 3 | | x | x | x | | x |
| | 4 | | | x | x | x | |
| Employee | 5 | x | x | x | | | |
| | 6 | x | x | x | x | | x |
| | 7 | x | x | | | x | x |
| | 8 | x | | x | x | | |
| | 9 | | | | | x | x |
| | 10 | x | x | | x | x | x |

The manager must meet all 10 employees once a week to discuss their progress. Currently, the meeting with each employee lasts about 20 minutes—that is, a total of 3 hours and 20 minutes for all 10 employees. A suggestion is made to reduce the total time by holding group meetings, depending on the projects the employees share. The manager wants to schedule the projects in a way that will reduce the traffic (number of employees) in and out of the meeting room. How should the projects be scheduled?

**2.** A book salesperson who lives in Basin must call once a month on four customers located in Wald, Bon, Mena, and Kiln. The following table gives the distances in miles among the different cities.

| | Basin | Wald | Bon | Mena | Kiln |
|---|---|---|---|---|---|
| Basin | 0 | 120 | 220 | 150 | 210 |
| Wald | 120 | 0 | 80 | 110 | 130 |
| Bon | 220 | 110 | 0 | 160 | 185 |
| Mena | 150 | 110 | 160 | 0 | 190 |
| Kiln | 210 | 130 | 185 | 190 | 0 |

The objective is to minimize the total distance traveled by the salesperson. Formulate the problem as an assignment-based ILP.

**3.** Circuit boards (such as those used with PCs) are fitted with holes to allow mounting different electronic components. The holes are drilled with a movable drill. The following table provides the distances (in centimeters) between pairs of 10 holes of a specific circuit board. The objective is to determine the optimum sequence for drilling all the holes.

$$\|d_{ij}\| = \begin{pmatrix} — & 1.2 & .5 & 2.6 & 4.1 & 3.2 \\ 1.2 & — & 3.4 & 4.6 & 2.9 & 5.2 \\ .5 & 3.4 & — & 3.5 & 4.6 & 6.2 \\ 2.6 & 4.6 & 3.5 & — & 3.8 & .9 \\ 4.1 & 2.9 & 4.6 & 3.8 & — & 1.9 \\ 3.2 & 5.2 & 6.2 & .9 & 1.9 & — \end{pmatrix}$$

Formulate the problem as an assignment-based ILP.

### 9.3.1 B&B Solution Algorithm

The idea of the B&B algorithm is to start with the solution of the associated assignment problem. If the solution is a tour, then there is nothing more to be done and the process ends. Otherwise, we need to introduce restrictions that remove the subtours. This can be achieved by creating as many branches as the number of $x_{ij}$-variables associated with one of the subtours. Each branch will then correspond to setting one of the variables of the subtour to zero (recall that all the variables associated with a subtour equal 1). The solution of the resulting assignment problem may or may not produce a tour. If it does, we use its objective value as an upper bound on the true minimum tour length. If it does not, further branching will be necessary, again creating as many branches as the number of variables in one of the subtours. The process continues until all unexplored subproblems have been fathomed, either by producing a better (smaller) upper bound or because there is evidence that the subproblem cannot produce a better solution. The optimum tour is the one associated with the best upper bound.

The following example provides the details of the traveling salesperson B&B algorithm.

## Example 9.3-2

The matrix below summarizes the distances in a 5-city traveling salesperson problem.

$$\|d_{ij}\| = \begin{pmatrix} \infty & 10 & 3 & 6 & 9 \\ 5 & \infty & 5 & 4 & 2 \\ 4 & 9 & \infty & 7 & 8 \\ 7 & 1 & 3 & \infty & 4 \\ 3 & 2 & 6 & 5 & \infty \end{pmatrix}$$

We start by solving the associated assignment (using TORA), which yields the following solution:

$$z = 15, (x_{13} = x_{31} = 1), (x_{25} = x_{54} = x_{42} = 1), \text{ all others} = 0$$

This solution yields two subtours: (1-3-1) and (2-5-4-2) as shown at node 1 in Figure 9.14. The associated total distance is $z = 15$, which provides a lower bound on the optimal length of the 5-city tour.

A straightforward way to determine an upper bound is to select any tour and then sum its respective distances to obtain an upper bound estimate. For example, the tour 1-2-3-4-5-1 (selected totally arbitrarily) has a total length of $10 + 5 + 7 + 4 + 3 = 29$. (You may be able to find a better upper bound by inspection. Remember that the smaller the upper bound, the more efficient the B&B search.)

The calculation of the lower and upper bounds now tells us that the optimum length of the tour must lie in range (15, 29). A solution that yields a tour length larger than 29 is discarded as nonpromising.

To eliminate the subtours at node 1, we need to "disrupt" its loop by forcing its member variables, $x_{ij}$, to zero level. Subtour 1-3-1 is broken if we impose $x_{13} = 0$ or $x_{31} = 0$ (i.e., one at a time) on the assignment problem at node 1. Similarly, subtour

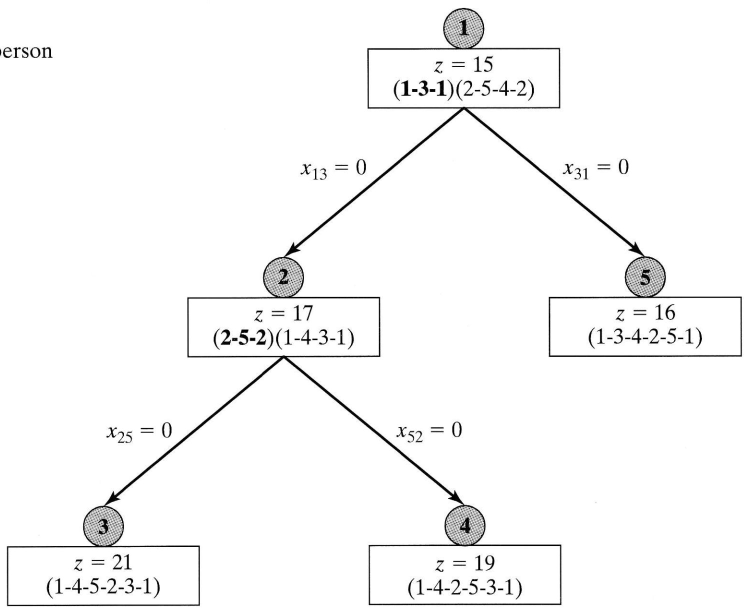



FIGURE 9.14

B&B solution of the traveling salesperson problem of Example 9.3-2

2-5-4-2 is eliminated by imposing one of the restrictions $x_{25} = 0$, $x_{54} = 0$, or $x_{42} = 0$. In terms of the B&B tree, each of these restrictions gives rise to a branch and hence a new subproblem. It is important to notice that branching *both* subtours at node 0 is *not* necessary. Instead, only *one* subtour needs to be disrupted at any one node. The idea is that a breakup of one subtour automatically alters the member variables of the other subtour and hence produces conditions that are favorable to creating a tour. Under this argument, from the computational standpoint, preference is given to the shortest subtour because it creates the smallest number of branches.

Targeting the shorter subtour (1-3-1), two branches $x_{13} = 0$ and $x_{31} = 0$ are created at node 1. The associated assignment problems are constructed by removing the row and column associated with the zero variable, which will make the assignment problem smaller. Another way of achieving the same result is to leave the size of the assignment problem unchanged and simply assign an infinite distance to the branching variable. For example, the assignment problem associated with $x_{13} = 0$ requires substituting $d_{13} = \infty$ in the assignment model at node 0. Similarly, for $x_{31} = 0$, we substitute $d_{31} = \infty$.

In Figure 9.14, we arbitrarily solve the subproblem associated with $x_{31} = 0$. Node 2 gives the solution $z = 17$ but continues to produce the subtours (2-5-2) and (1-4-3-1). Repeating the procedure we made at node 1 gives rise to two branches: $x_{25} = 0$ and $x_{52} = 0$.

We now have three unexplored subproblems, one from node 1 and two from node 2, and we are free to investigate any of them at this point. Arbitrarily exploring the subproblem associated with $x_{25} = 0$ from node 2, we set $d_{13} = \infty$ and $d_{25} = \infty$ in the original assignment problem, which yields the solution $z = 21$ and the tour solution 1-4-5-2-3-1 at node 3. Node 3 need not be investigated any further and hence is fathomed.

The solution at node 3 provides an improved upper bound, $z = 21$, on the optimal length of the tour. This means that any unexplored subproblem that can be shown to yield a tour length larger than (or equal to) 21 must be discarded as nonpromising.

We now have two unexplored subproblems. Selecting subproblem 4 for exploration, we set $d_{13} = \infty$ and $d_{52} = \infty$ in the original assignment, which yields the tour solution 1-4-2-5-3-1 with $z = 19$. The new tour solution provides the better upper bound $z = 19$.

Only subproblem 5 remains unexplored. Substituting $d_{31} = \infty$ in the original assignment problem at node 1, we get the tour solution 1-3-4-2-5-1 with $z = 16$, at node 5. Once again, this is a better solution than the one associated with node 3 and thus requires updating the upper bound to $z = 16$.

There are no remaining unfathomed nodes, which completes the search tree. The optimal tour is the one associated with the current upper bound: 1-3-4-2-5-1 with length 16 miles.

One remark is in order: The search sequence $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$ for exploring the nodes demonstrates once again one of the difficulties associated with the B&B algorithm. We have no way of predicting in advance which sequence we should follow to explore subproblems in the B&B tree. For example, had we started with node 5, we would have obtained the tight upper bound $z = 16$, which will automatically fathom subproblem 2 and hence eliminate the need to create subproblems 4 and 5.

Of course, there are heuristics that can be of help in "foreseeing" which sequence could lead to a more efficient tree. For example, after specifying all the branches from a given node, we can start with the branch associated with the *largest* $d_{ij}$ among all the created branches. This heuristic calls for exploring branch $x_{31} = 0$ at node 0. Had this been done, the upper bound $z = 16$ would have been encountered at the first subproblem.

### PROBLEM SET 9.3B

1. Solve Example 9.3-2 using subtour 2-5-4-2 to start the branching process at node 0 and the following sequences for exploring the nodes.
   **(a)** Explore all the subproblems horizontally from left to right in each tier before proceeding to the next tier.
   **(b)** Follow each path vertically from node 0 until it ends with a fathomed node.
2. Solve Problem 1, Set 9.3a using B&B.
3. Solve Problem 2, Set 9.3a using B&B.
4. Solve Problem 3, Set 9.3a using B&B.

## 9.3.2 Cutting Plane Algorithm

The idea of the cutting plane algorithm is to add a set of constraints, which when added to the assignment problem are guaranteed to prevent the formation of a subtour. The additional constraints are defined as follows. In an $n$-city, associate a continuous variable $u_j(\geq 0)$ with cities 2, 3, ... , and $n$. Next, define the required set of additional constraints as

$$u_i - u_j + nx_{ij} \leq n - 1,\ i = 2, 3, \ldots, n;\quad j = 2, 3, \ldots, n;\quad i \neq j$$

These constraints, when added to the assignment model, will automatically remove all subtour solutions but will not eliminate any tour solution.

### Example 9.3-3

Consider the following distance matrix of a 4-city traveling salesperson problem.

$$\|d_{ij}\| = \begin{pmatrix} \infty & 13 & 21 & 26 \\ 10 & \infty & 29 & 20 \\ 30 & 20 & \infty & 5 \\ 12 & 30 & 7 & \infty \end{pmatrix}$$

The associated LP consists of the assignment model constraints plus the following additional constraints that prevent the formation of subtour solutions. All $x_{ij} = (0, 1)$ and all $u_j \geq 0$. The problem is solved as a mixed integer linear program.

| $x_{11}$ | $x_{12}$ | $x_{13}$ | $x_{14}$ | $x_{21}$ | $x_{22}$ | $x_{23}$ | $x_{24}$ | $x_{31}$ | $x_{32}$ | $x_{33}$ | $x_{34}$ | $x_{41}$ | $x_{42}$ | $x_{43}$ | $x_{44}$ | $u_2$ | $u_3$ | $u_4$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | 4 | | | | | | | | | | 1 | −1 | | ≤3 |
| | | | | | | | 4 | | | | | | | | | 1 | | −1 | ≤3 |
| | | | | | | | | | 4 | | | | | | | −1 | 1 | | ≤3 |
| | | | | | | | | | | | 4 | | | | | | 1 | −1 | ≤3 |
| | | | | | | | | | | | | | 4 | | | −1 | | 1 | ≤3 |
| | | | | | | | | | | | | | | 4 | | | −1 | 1 | ≤3 |

The optimum solution, obtained by TORA's ILP module (file Ch9ToraTraveling SalespersonEx9-3-3.txt), is given as

$$u_2 = 0,\ u_3 = 1,\ u_4 = 2,\ x_{12} = x_{23} = x_{34} = x_{41} = 1,\ \text{tour length} = 59$$

This corresponds to the tour solution 1-2-3-4-1. The solution satisfies all the additional constraints in $u_j$ (verify!).

To show that subtour solutions do not satisfy the additional constraints, consider the subtour solution (1-2-1, 3-4-3). This solution corresponds to $x_{12} = x_{21} = 1$, $x_{34} = x_{43} = 1$. Now, consider constraints 4 and 6 in the tableau above—namely,

$$4x_{34} + u_3 - u_4 \leq 3$$

$$4x_{43} - u_3 + u_4 \leq 3$$

Substituting $x_{34} = x_{43} = 1$ and summing the two inequalities yields $8 \leq 6$, which is impossible, thus disallowing the formation of the subtour.

The main disadvantage of the cutting plane model is that its size grows exponentially with the number of cities. For this reason, the B&B algorithm offers a more efficient way for solving the problem.

## PROBLEM SET 9.3C

**1.** Solve the following traveling salesperson problem by the cutting plane algorithm.

**(a)** $\|d_{ij}\| = \begin{pmatrix} \infty & 43 & 21 & 20 \\ 10 & \infty & 9 & 22 \\ 20 & 10 & \infty & 5 \\ 42 & 50 & 27 & \infty \end{pmatrix}$

**(b)** Problem 2, Set 9.3a.

**(c)** Problem 3, Set 9.3a.

## SELECTED REFERENCES


Nemhauser, G., and L. Wolsey, *Integer and Combinatorial Optimization,* Wiley, New York, 1988.

Salkin, H., and K. Mathur, *Foundations of Integer Programming,* North-Holland, New York, 1989.

Taha, H., *Integer Programming: Theory, Applications, and Computations,* Academic Press, Orlando, FL, 1975.

Wolsey, L., *Integer Programming,* Wiley, New York, 1998.


## COMPREHENSIVE PROBLEMS

**9.1** A development company owns 90 acres of land in a growing metropolitan area, where it intends to construct office buildings and a shopping center. The developed property is rented for 7 years and then sold. The sales price for each building is estimated at 10 times its operating net income in the last year of rental. The company estimates that the project will include a 4.5-million-square-foot shopping center. The master plan calls for constructing three high-rise and four garden office buildings.

The company is faced with a scheduling problem. If a building is completed too early, it may stay vacant; if it is completed too late, potential tenants may be lost to other projects. The demand for office space over the next 7 years based on appropriate market studies is

| | Demand (thousands of $ft^2$) | |
|---|---|---|
| *Year* | *High-rise space* | *Garden space* |
| 1 | 200 | 100 |
| 2 | 220 | 110 |
| 3 | 242 | 121 |
| 4 | 266 | 133 |
| 5 | 293 | 146 |
| 6 | 322 | 161 |
| 7 | 354 | 177 |

The following table lists the proposed capacities of the seven buildings:

| Garden building | Capacity ($ft^2$) | High-rise building | Capacity ($ft^2$) |
|---|---|---|---|
| 1 | 60,000 | 1 | 350,000 |
| 2 | 60,000 | 2 | 450,000 |
| 3 | 75,000 | 3 | 350,000 |
| 4 | 75,000 | — | — |

The gross rental income is estimated at $25 per square foot. The operating expenses are $5.75 and $9.75 per square foot for the garden and high-rise buildings, respectively. The associated construction costs are $70 and $105 per square foot, respectively. Both the construction cost and the rental income are estimated to increase at roughly the inflation rate of 4%.

How should the company schedule the construction of the seven buildings?

**9.2**[5] In a National Collegiate Athletic Association women's gymnastics meet, competition includes four events: vault, uneven bars, balance beam, and floor exercises. Each team may enter the competition with six gymnasts per event. A gymnast is evaluated on a scale of 1 to 10. Past statistics for the U of A team produce the following scores:

| | U of A Scores for Gymnast | | | | | |
|---|---|---|---|---|---|---|
| *Event* | *1* | *2* | *3* | *4* | *5* | *6* |
| Vault | 6 | 9 | 8 | 8 | 4 | 10 |
| Bars | 7 | 9 | 7 | 8 | 9 | 5 |
| Beam | 9 | 8 | 10 | 9 | 9 | 8 |
| Floor | 6 | 6 | 5 | 9 | 10 | 9 |

The total score for a team is determined by summarizing the top five individual scores for each event. An entrant may participate as a specialist in one event or an "all-rounder" in all four events, but not both. A specialist is allowed to compete in at most

[5]Based on P. Ellis and R. Corn, "Using Bivalent Integer Programming to Select Teams of Intercollegiate Women's Gymnastic Competition," *Interfaces*, Vol. 14, No. 3, pp. 41–46, 1984.

three events, and at least four of the team participants must be all-rounders. Set up an ILP model that can be used to select the competing team, and find the optimum solution using TORA.

**9.3**[6] In 1990, approximately 180,000 telemarketing centers employing 2 million individuals were in operation in the United States. In the year 2000, more than 700,000 companies employed approximately 8 million people in telemarketing their products. The questions of how many telemarketing centers to employ and where to locate them are of paramount importance.

The ABC company is in the process of deciding on the number of telemarketing centers to employ and their locations. A center may be located in one of several candidate areas selected by the company and may serve (partially or completely) one or more geographical areas. A geographical area is usually identified by one or more (telephone) area codes. ABC's telemarketing concentrates on eight area codes: 501, 918, 316, 417, 314, 816, 502, and 606. The following table provides the candidate locations, their served areas, and the cost of establishing the center.

| Center location | Served area codes | Cost ($) |
|---|---|---|
| Dallas, TX | 501, 918, 316, 417 | 500,000 |
| Atlanta, GA | 314, 816, 502, 606 | 800,000 |
| Louisville, KY | 918, 316, 417, 314, 816 | 400,000 |
| Denver, CO | 501, 502, 606 | 900,000 |
| Little Rock, AR | 417, 314, 816, 502 | 300,000 |
| Memphis, TN | 606, 501, 316, 417 | 450,000 |
| St. Louis, MO | 816, 502, 606, 314 | 550,000 |

Customers in all area codes can access any of the centers 24 hours a day.

The communication costs per hour between the centers and the area codes are given in the following table.

| | From area code | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *To* | *501* | *918* | *316* | *417* | *314* | *816* | *502* | *606* |
| Dallas, TX | $14 | $35 | $29 | $32 | $25 | $13 | $14 | $20 |
| Atlanta, GA | $18 | $18 | $22 | $18 | $26 | $23 | $12 | $15 |
| Louisville, KY | $22 | $25 | $12 | $19 | $30 | $17 | $26 | $25 |
| Denver, CO | $24 | $30 | $19 | $14 | $12 | $16 | $18 | $30 |
| Little Rock, AR | $19 | $20 | $23 | $16 | $23 | $11 | $28 | $12 |
| Memphis, TN | $23 | $21 | $17 | $21 | $20 | $23 | $20 | $10 |
| St. Louis, MO | $17 | $18 | $12 | $10 | $19 | $22 | $16 | $22 |

ABC would like to select between three and four centers. Where should they be located?

[6]Based on T. Spencer, A. Brigandi, D. Dargon, and M. Sheehan, "AT&T's Telemarketing Site Selection System Offers Customer Support," *Interfaces,* Vol. 20, No. 1, pp. 83–96, 1990.

**9.4**[7] An electric utility company serving a wide rural area wants to decide on the number and location of Customer-Service Linemen (CSL) centers that will provide responsive service regarding repairs and connections. The company groups its customer base in five clusters according to the following data:

| Cluster | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Number of customers | 400 | 500 | 300 | 600 | 700 |

The company has selected five potential locations for its CSL centers. The following table summarizes the average travel distance in miles from the CSLs to the different clusters. The average speed of the service truck is approximately 45 miles per hour.

| | CSL center | | | | |
|---|---|---|---|---|---|
| *Cluster* | *1* | *2* | *3* | *4* | *5* |
| 1 | 40 | 100 | 20 | 50 | 30 |
| 2 | 120 | 90 | 80 | 30 | 70 |
| 3 | 40 | 50 | 90 | 80 | 40 |
| 4 | 80 | 70 | 110 | 60 | 120 |
| 5 | 90 | 100 | 40 | 110 | 90 |

The company would like to keep the response time to a customer request to around 90 minutes. How many CSL centers should be in operation?

[7]Based on E. Erkut, T. Myrdon, and K. Strangway, "Transatlanta Redesigns Its Service Delivery Network," *Interfaces,* Vol. 30, No. 2, pp. 54–69, 2000.

10