ASSIGNMENT PROBLEM

There are n people who need to be assigned to execute n jobs, one person per job. (That is, each person is assigned to exactly one job and each job is assigned to exactly one person.)

The cost that would accrue if the i^{th} person is assigned to the j^{th} job is a known quantity [i, j] for each pairi, j = 1, 2, . . . , n. The problem is to find an assignment with the minimum total cost.

Assignment problem solved by exhaustive search is illustrated with an example as shown in figure 2.8. A small instance of this problem follows, with the table entries representing the assignment costs *C*[*i, j*].

	Job 1	Job 2	Job 3	Job 4
Person 1	9	2	7	8
Person 2	6	4	3	7
Person 3	5	8	1	8
Person 4	7	6	9	4

FIGURE 2.7 Instance of an Assignment problem.

An instance of the assignment problem is completely specified by its cost matrix *C*.

$$= \begin{bmatrix} 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 8 \\ 7 & 6 & 9 & 4 \end{bmatrix}$$

The problem is to select one element in each row of the matrix so that all selected elements are in different columns and the total sum of the selected elements is the smallest possible.

We can describe feasible solutions to the assignment problem as *n*-tuples

$<j_1, \ldots, j_n>$ in which the i^{th} component, i=1,...,n, indicates the column of the element selected in the i^{th} row (i.e., the job number assigned to the i^{th} person). For example, for the cost matrix above, <2, 3, 4, 1> indicatestheassignmentofPerson1toJob2, Person2 to Job3, Person3toJob4, and Person 4to Job 1. Similarly, we can have 4! = 4 · 3 · 2 · 1 = 24, i. e., 24 permutations.

The requirements of the assignment problem imply that there is a one-to-one correspondence between feasible assignments and permutations of the first *n* integers.

Therefore, the exhaustive-search approach to the assignment problem would require generating all the permutationsofintegers1, 2,,computing the total cost of each assignment by summing up the corresponding elements of the cost matrix, and finally selecting the one with the smallest sum.

A few first iterations of applying this algorithm to the instance given above are given below.

<1, 2, 3,4>	cost = 9 + 4 + 1 + 4 =**18**	**<2, 1, 3, 4>**	**cost = 2 + 6 + 1 + 4 = 13 (Min)**
<1, 2, 4,3>	cost = 9 + 4 + 8 + 9 =30	<2, 1, 4,3>	cost = 2 + 6 + 8 + 9 =25
<1, 3, 2,4>	cost = 9 + 3 + 8 + 4 =24	<2, 3, 1,4>	cost = 2 + 3 + 5 + 4 =14
<1, 3, 4,2>	cost = 9 + 3 + 8 + 6 =26	<2, 3, 4,1>	cost = 2 + 3 + 8 + 7 =20
<1, 4, 2,3>	cost = 9 + 7 + 8 + 9 =33	<2, 4, 1,3>	cost = 2 + 7 + 5 + 9 =23
<1, 4, 3, 2>	cost = 9 + 7 + 1 + 6 = 23	<2, 4, 3, 1>	cost = 2 + 7 + 1 + 7 = 17, etc

FIGURE 2.8 First few iterations of solving a small instance of the assignment problem by exhaustive search.

Since the number of permutations to be considered for the general case of the assignment problem is ***n!,*** exhaustive search is impractical for all but very small instances of the problem. Fortunately, there is a much more efficient algorithm for this problem called the ***Hungarian method.***