Page – 1

The network model may be easily converted to the matrix model of Table 14-2(a) or 14-2(b). Table 14-2(a) is the familiar transportation matrix, which can be used since the assignment problem is but a special class of transportation problem. However, much of the information shown in the transportation type of matrix is unnecessary. Consider, for example, the rim conditions. A 1 will always appear for each availability and demand, and thus the rims may be dropped. Another simplification is concerned with each internal cell. An allocation to a cell will always be either a zero (no assignment) or a 1 (an assignment); consequently, we simply list the cost of each cell as shown in Table 14-2(b). The latter matrix is the simplified or condensed matrix that will normally be used. Note also that, in Table 14-2(b), the optimal assignment has been indicated by simply placing a box about the cost elements of the cells to which an assignment has been made. This, too, will be our normal practice.

Table 14-2. MATRIX MODELS

From \ *To*	1	2	3	4	*Available*
A	$\boxed{31}$	$\boxed{32}$	$\boxed{34}$	$\boxed{29}$	1
B	$\boxed{36}$	$\boxed{30}$	$\boxed{37}$	$\boxed{33}$	1
C	$\boxed{34}$	$\boxed{34}$	$\boxed{37}$	$\boxed{35}$	1
D	$\boxed{30}$	$\boxed{33}$	$\boxed{31}$	$\boxed{28}$	1
Demand	1	1	1	1	4 \ 4

(a) ~~Transportation~~-Type Matrix

	1	2	3	4
A	31	32	34	$\boxed{29}$
B	36	$\boxed{30}$	37	33
C	$\boxed{34}$	34	37	35
D	30	33	$\boxed{31}$	28

(b) Condensed (Assignment) Matrix

The final model to be presented is the mathematical model, which is derived in the same way as was the mathematical model of the transportation problem 13. This model is, for our specific example:

Find $x_{i,j}$ ($i = 1, 2, 3, 4$ and $j = 1, 2, 3, 4$) to

$$\text{minimize} \quad z = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{i,j} x_{i,j} \tag{14.1}$$

subject to

$$\left.\begin{aligned} \sum_{j=1}^{n} x_{i,j} &= 1 \qquad \text{all } i \\ \sum_{i=1}^{m} x_{i,j} &= 1 \qquad \text{all } j \end{aligned}\right\} \tag{14.2}$$

$$x_{i,j} = 0, 1 \qquad \text{all } i \text{ and } j \tag{14.3}$$

where m = number of source nodes (= 4 in example)
n = number of sink nodes (= 4 in example)
$c_{i,j}$ = "cost" (time in the example) of assigning source node i to sink node j

Notice in particular the restrictions implied in (14.3). In the assignment problem

$$x_{i,j} = \begin{cases} 1 & \text{if source node } i \text{ is assigned to sink node } j \\ 0 & \text{otherwise} \end{cases}$$

Thus, the decision variables within the assignment problem are all zero–one-valued and the problem itself is a linear, zero–one programming problem. This contrasts with the more general linear, integer programming model associated with the transportation problem.

Recall in Chapter 13 that the transportation model possessed the property of unimodularity. This is also true of the assignment model, which may be seen if the structure of the constraint matrix is examined. This means, then, that if we start with any integer feasible solution, the simplex method will converge to the integer-valued optimal solution (wherein the integer values themselves will be restricted to either zero or 1). However, just as we did with the transportation problem, we consider more efficient, special solution methods.

Selected Solution Techniques

There are numerous ways to solve the conventional assignment problem, ranging from the general simplex algorithm to very specific techniques. We restrict our attention to just a few of these approaches: (1) VAM/MODI, (2) the Hungarian algorithm, and (3) branch-and-bound.

VAM/MODI approach. Since the assignment problem is but a special subclass of the transportation problem, we can form the transportation matrix model and solve via the VAM/MODI algorithm presented in Chapter 13.

Page - 2

Unfortunately, the assignment problem will always have a degenerate solution, and thus this approach turns out to not be particularly efficient. Consider, for example, the assignment problem represented in Table 14-3.

Table 14-3. ASSIGNMENT PROBLEM IN TRANSPORTATION MATRIX FORMAT

From \ To	*I*	*II*	*III*	*IV*	a_i
A	$\boxed{2}$	$\boxed{10}$	$\boxed{3}$ 1	$\boxed{17}$	1
B	$\boxed{5}$	$\boxed{3}$	$\boxed{9}$	$\boxed{10}$ 1	1
C	$\boxed{8}$	$\boxed{2}$ 1	$\boxed{5}$	$\boxed{14}$	1
D	$\boxed{3}$ 1	$\boxed{5}$	$\boxed{10}$	$\boxed{16}$	1
b_j	1	1	1	1	4 \ 4

The solution shown in Table 14-3 is found via VAM, based on the assumption that the $c_{i,j}$'s are costs (i.e., that we are minimizing). Note two things:

1. There must always be a *single* allocation to any row or column (this is known as finding an independent set of assignments).
2. The solution is always degenerate.

Given an assignment problem with $m = n$ (i.e., equal number of rows and columns), a nondegenerate solution must have (see Chapter 13) exactly $m + n - 1$ assignments. However, since only a single assignment can appear in any row or column, the assignment problem *always* will have exactly m $(= n)$ assignments. This means that the number of ϵ allocations necessary for phase 2 (MODI) is: $(m + n - 1) - m = n - 1 = m - 1$. For example, an assignment problem with 100 rows (and columns) will require *99 ϵ allocations*. Consequently, we find VAM/MODI to not be an attractive approach to the assignment problem, particularly for problems of modest to large size.

Hungarian algorithm. The so-called Hungarian algorithm [3] provides us with a relatively effective and rather novel approach to the assignment problem. It is particularly appealing when solving by hand problems of small to modest size because it involves the use of visual pattern recognition, an ability in which the human being can often outperform the computer.

Consider an assignment problem in which we are minimizing and *all cell elements are nonnegative*. The lowest possible cost in any cell is thus zero. Further,

the lowest *possible* total cost is also zero, which could occur only if it we possible to make a selection of independent assignments to cells for which tl individual costs were zero. Now, it is extremely unlikely that we would ev encounter such a problem in actual practice. However, the basic concept implic above is the foundation of the Hungarian algorithm.

Any assignment problem may be transformed into an equivalent proble in which the smallest cost value in any row or column is zero. This is done l means of the following procedure:

Step 1. Select the smallest element in each row (of the assignment matrix) and su tract that element from each element in that row. This will generate at lea one zero element in every row of the matrix.

Step 2. Select the smallest element in each column and subtract that element from eac element in that column. This step will generate at least one zero element every column of the matrix.

The solution to the converted matrix must correspond to that of the origina Applying these two steps to the assignment matrix for the problem given Table 14-3 [or Table 14-4(a)] will result in Table 14-4(b). Notice that the assig ment given through VAM for Table 14-3 is also depicted in Table 14-4(b wherein each assigned cell is indicated by a box about the (converted) cos Since this assignment is to cells that each have a cost of zero, no better assig ment is possible. Consequently, our solution previously arrived at by VAM wa actually optimal, with a total cost of $3 + 10 + 2 + 3 = 18$ units.

Normally, it will not be nearly so easy to solve the assignment problem ar we must establish a systematic process (the Hungarian algorithm in this particula

Table 14-4. CONVERTED ASSIGNMENT MATRIX

2	10	3	17
5	3	9	10
8	2	5	14
3	5	10	16

(a) Original Matrix

0	8	$\boxed{0}$	8
2	0	5	$\boxed{0}$
6	$\boxed{0}$	2	5
$\boxed{0}$	2	6	6

(b) Converted Matrix

Page-3

case) to identify the final, optimal solution. Given a matrix that has been converted as above, the steps of the Hungarian algorithm begin by first determining if there is a feasible assignment that may be made by using only cells with zero elements. If so, the problem is solved. If not, we incorporate a process that will, in essence, generate additional zeros in the matrix. The steps of the Hungarian algorithm are given below and are demonstrated on the example problem shown in Table 14-5 (wherein the previous conversion process has already taken place).

Step 1. Place a "box" around a zero in any row that has only one zero in it and cross off all other zeros in the column containing the box. If all rows contain a box, you are finished. Otherwise, go to step 2. (See Table 14-6.)

Step 2. Place a box around a zero in any column that has only one zero in it and cross out all other zeros in the row containing the box. If every column has one box in it, you are finished. Otherwise, go to step 3. (See Table 14-7.)

Step 3. If steps 1 and 2 have not led to a feasible solution, it is necessary to attempt to generate additional zeros in the matrix by drawing the *minimum* number of lines (either horizontal or vertical, or some mixture) that will cover (pass through) *all* the zeros in the matrix. A procedure to aid in drawing such lines follows:

(a) Check the rows having no boxed zeros (row *C* in the example).
(b) Check the columns having a zero in a checked row. (Check column I in the example.)
(c) Check the rows having a boxed zero in a checked column. (Check the second row.)
(d) Repeat steps (b) and (c) until no further rows or columns can be checked.
(e) Draw lines through all unchecked rows and all checked columns. (See Table 14-8.)

Step 4. All elements with lines through them are termed "covered." Elements without lines passing through them are "uncovered." Select the *smallest uncovered element* in the matrix (the number 1 in row *B*, column II) and subtract this element from *all uncovered elements*. Add this element to all *covered cells that occur at the intersection of two covering lines*. (The resultant matrix is shown in Table 14-9.)

Step 5. Repeat steps 1 through 4 until an optimal assignment is found. (See Table 14-10.)

Table 14-5. EXAMPLE FOR HUNGARIAN ALGORITHM

From \ To	*I*	*II*	*III*
A	2	0	1
B	0	1	4
C	0	5	5

Table 14-6

	I	*II*	*III*
A	2	0	0
B	$\boxed{0}$	1	3
C	$\cancel{0}$	5	4

Table 14-7

	I	*II*	*III*
A	2	$\boxed{0}$	$\cancel{0}$
B	$\boxed{0}$	1	3
C	$\cancel{0}$	5	4

Table 14-8

	I	*II*	*III*	
A	2	$\boxed{0}$	0	
B	$\boxed{0}$	1	3	✓
C	0	5	4	✓
	✓			

Table 14-9

	I	*II*	*III*
A	3	0	0
B	0	0	2
C	0	4	3

Table 14-10

	I	*II*	*III*
A	3	$\cancel{0}$	$\boxed{0}$
B	$\cancel{0}$	$\boxed{0}$	2
C	$\boxed{0}$	4	3

The final solution to our example is shown in Table 14-10, wherein v assign:

A to III

B to II

C to I