3

Transportation Problems

3.1 Introduction

In Chapter 2 we have discussed linear programming in general. There are some problems which give rise to special forms of linear programming. In this chapter, and the next, we will consider two such classes—transportation and assignment problems.

In order to illustrate the class of transportation problems, we will consider a simple example related very much to the pioneering developments in this area.

An electricity utility has four power stations, which we label $j = 1, 2, 3, 4$. They are supplied by three collieries, which we label $i = 1, 2, 3$. The total supply of coal at the collieries is equal to the total requirement by the power stations in appropriate units. There is a cost of transporting one unit of coal from each colliery to each power station. Table 3.1 gives the available supplies, the requirements, and the unit transportation costs.

Table 3.1

		Power station j				
		1	2	3	4	Available supplies
Colliery i	1	2	3	4	5	10
	2	5	4	3	1	15
	3	1	3	3	2	21
Requirements		6	11	17	12	

The transportation problem is to determine how many units to transport from each colliery to each power station in order to minimise the total cost in meeting all requirements.

Readers may wish to see if they can solve this problem themselves before looking at the standard way of solving the problem. The minimal total cost is 102 and the unique, optimal, solution is to transport 10, 3, 12, 6, 1, 14 units from collieries 1, 2, 2, 3, 3, 3 to power stations 2, 3, 4, 1, 2, 3 respectively.

As has been indicated, the transportation problem may be considered to be a special class of linear program, and we will now look at this.

3.2 A linear programming formulation

We will proceed to identify the decision variables, constraints, and objective functions as we do in general linear programming.

We will initially formulate the problem as one in which *no more than* the available supplies may be used, and *at least* the specified requirements must be met, so that the problem is in *inequality* form.

Let x_{ij} be the amount to be transported from colliery i to power station j.

The constraints are as follows.

$$\begin{aligned}
x_{11}+x_{12}+x_{13}+x_{14} &\leqslant 10 &&\text{(colliery 1)}\\
x_{21}+x_{22}+x_{23}+x_{24} &\leqslant 15 &&\text{(colliery 2)}\\
x_{31}+x_{32}+x_{33}+x_{34} &\leqslant 21 &&\text{(colliery 3)}\\
x_{11}+x_{21}+x_{31} &\geqslant 6 &&\text{(power station 1)}\\
x_{12}+x_{22}+x_{32} &\geqslant 11 &&\text{(power station 2)}\\
x_{13}+x_{23}+x_{33} &\geqslant 17 &&\text{(power station 3)}\\
x_{14}+x_{24}+x_{34} &\geqslant 12 &&\text{(power station 4)}
\end{aligned}$$

The problem has been formulated as one of at least meeting the requirements, and hence we use $\geqslant$ rather than $=$ for the moment. For general linear programming problems it is not necessarily true that optimal solutions require that all constraints be satisfied as *equalities*.

$$x_{ij} \geqslant 0 \text{ for all } i \text{ and } j.$$

The objective function is as follows.

$$\begin{aligned}\text{Total cost} = M = {} & 2x_{11}+3x_{12}+4x_{13}+5x_{14}+5x_{21}+4x_{22}+3x_{23}+\\ & 1x_{24}+1x_{31}+3x_{32}+3x_{33}+2x_{34}.\end{aligned}$$

The problem is to minimise M subject to the specified constraints.

In terms of the above notation, the optimal solution is given by

$$\begin{aligned}&M = 102, x_{12} = 10, x_{23} = 3, x_{24} = 12, x_{31} = 6, x_{32} = 1, x_{33} = 14,\\ &x_{11} = x_{13} = x_{14} = x_{21} = x_{22} = x_{34} = 0.\end{aligned}$$

This linear programming problem may be solved using the simplex method of Chapter 2. If the availability levels are whole numbers, then the simplex method always gives whole number solutions for every iteration (i.e. for all basic feasible solutions). However, there is a simpler method of solving this problem, related to the *simplex* algorithm and to *duality* as discussed in Chapter 2. This method is known as the *transportation algorithm*, and we will restrict ourselves to this in this text. It is the simplex method done more efficiently than for general linear programming problems. Before doing so, we must convert our *inequality* problem to an *equality* problem in which the inequalities for the availability and requirement constraints are replaced by equalities. The reader should check that this is valid for our problem in which total availability equals total requirements.

3.3 The transportation algorithm

There are two essential steps in the transportation algorithm: (a) *finding starting solutions*; and (b) *improving on solutions*, if possible, until no further improvement is possible.

Before proceeding with the method, it is to be noted that all approaches involve essentially the same operations in that they

(1) select a cell (i, j)
(2) make x_{ij} as large as possible
(3) delete a row or column
(4) return to (1).

The differences in the three approaches which we will use lie solely in the mechanism for selecting (i, j) in (1) for the starting solution only. Thereafter, the operations are identical.

This feature of finding a starting solution, improving on it in some way, and repeating the procedure, is a common one in other chapters in this text. At each stage, there will be checks to determine whether or not an improvement is possible (e.g. see the next chapter on the assignment problem).

Finding a starting solution

By a *starting solution* we mean a set of transportation quantities which will satisfy all the constraints. There are several ways of finding starting solutions. We will deal with three: the north-west corner rule method, minimal unit cost method, and Vogel's method. The first method is the more easily applied method, but the latter methods, although involving extra work, seek to begin with *good* starting solutions in an attempt to speed up the algorithm. They work loosely on a principle of the kind *smallest in a row, or column, is best.*

Throughout the application of the method, each solution will have $m+n-1$ positive values of x_{ij}, if the problem has m supply sources and n requirements points (i.e. m rows and n columns in the formulation), with the exception of degenerate situations, which we will briefly discuss later on. For our example, this means that solutions will have $6(=3+4-1)$ positive transportation quantities.

North-west corner rule

We begin in the top left-hand corner (i.e. north-west position) and make x_{11} as large as possible compatible with the constraints. This gives $x_{11} = 6$

(= minimum of available supply of 10 and requirement of 6). This exhausts the requirement in column 1, which we delete, with a residual supply left, for row 1, of 4 units. The new sub-problem is then given in Table 3.2.

Table 3.2

		Power station j 1	2	3	4	Available supplies
Colliery i	1	6				4
	2			sub-problem		15
	3					21
Requirements			11	17	12	

We now repeat the procedure with the sub-problem of Table 3.2, and make x_{12} as large as possible compatible with the new constraints. This gives $x_{12} = 4$ (= minimum of available supply of 4 and requirement of 11). This exhausts the supply in row 1, which we delete, with a residual requirement of 7 for column 2.

The procedure is repeated until we arrive at the north-west corner rule solution of Table 3.3. Each time a row or column is exhausted it is deleted, and the procedure applied to the new sub-problem arising from this.

Table 3.3

		Power station j 1	2	3	4	Available supplies
Colliery i	1	6	4			10
	2		7	8		15 Total cost = 127
	3			9	12	21
Requirements		6	11	17	12	

Although we have explained the step-by-step procedure in terms of sub-problems, it is quite easy to do the calculations in one go on a single table.

For some problems we may simultaneously exhaust a row *and* a column when the availability and requirements are equal for that row and column. This will result in a *degenerate* situation to which we will return later on.

The minimal unit cost method

This method proceeds by allocating as much as possible to the cell (i, j) which has the smallest unit cost, deleting the row or column whose availability or requirement is exhausted, and repeating the procedure until all rows and columns have been deleted. The calculations proceed as in Tables 3.4–3.8.

Table 3.4

		Power station j 1	2	3	4	Available supplies
Colliery i	1	2	3	4	5	10
	2	5	4	3	1	15
	3	1	3	3	2	21
Requirements		6	11	17	12	

We may make x_{24} or x_{31} as large as possible. We choose $x_{24} = 12$, and column 4 is deleted (Table 3.5)

Table 3.5

		Power station j 1	2	3	4	Available supplies
Colliery i	1	2	3	4		10
	2	5	4	3	12	3
	3	1	3	3		21
Requirements		6	11	17		

We make x_{31} as large as possible, i.e. $x_{31} = 6$, and column 1 is deleted (Table 3.6).

We may make x_{12}, x_{23}, x_{32} or x_{33} as large as possible. We choose $x_{12} = 10$, and row 1 is deleted to give Table 3.7.

We may make x_{23}, x_{32}, or x_{33} as large as possible. We choose $x_{23} = 3$. Then $x_{32} = 1$, $x_{33} = 14$, and we obtain the starting solution in Table 3.8.

This happens to be the optimal solution, but the method need not give optimal solutions, even for the case $m = n = 2$.

Table 3.6

		Power station j 1	2	3	4	Available supplies
Colliery i	1		3	4		10
	2		4	3	12	3
	3	6	3	3		15
Requirements			11	17		

Table 3.7

		Power station j 1	2	3	4	Available supplies
Colliery i	1		10			
	2		4	3	12	3
	3	6	3	3		15
Requirements			1	17		

Table 3.8

		Power station j 1	2	3	4	Available supplies
Colliery i	1		10			10
	2			3	12	15 Total cost = 102
	3	6	1	14		21
Requirements		6	11	17	12	

For the example of Table 3.9, it may be shown that for $m = n = 2$, the minimal unit cost rule will not give an optimal starting solution.

Table 3.9

		Destination j 1	2	Available supplies
Source i	1	1	2	2
	2	2	4	1
Requirements		1	2	

The appeal of the method derives from the intuitively desirable feature of using up as much of the available supplies or requirements at as small a cost as possible.

Vogel's method

As has been indicated, it seems prudent to seek *good* starting solutions. Vogel's method does this by looking, in effect, at *relative* unit costs. More specifically, at each step the procedure determines, for the sub-problem being considered, the two smallest unit costs in each row and the two smallest unit costs in each column. If several unit costs are equally the smallest, then any two of these may be used. The procedure then takes the difference between the pair so determined for each row and column. Again, note that the smallest and second smallest may be equal, so this difference may be zero. For the row, or column, where this difference is largest, the minimal unit cost is determined and, as for the north-west corner rule, the maximal transportation amount, x_{ij}, compatible with the availabilities and requirements is determined for the cell (i, j) where this minimum is located. The row or column which is thereby exhausted is then deleted, and the procedure is repeated until all rows and columns have been deleted and a starting solution is obtained.

For our specimen problem, we obtain Tables 3.10–3.14.

Table 3.10

		Power station j 1	2	3	4	Unit cost difference
Colliery i	1	2	3	4	5	1
	2	5	4	3	1	2 ← Largest
	3	1	3	3	2	1
Unit cost difference		1	0	0	1	

The maximal difference in Table 3.10 is in row 2, and the smallest unit cost in this row is 1 in cell (2, 4) (i.e. row 2, column 4). We make x_{24} as large as possible compatible with availability and requirement levels, to give $x_{24} = 12$, and column 4 is deleted to give Table 3.11.

Table 3.11

		Power station j 1	2	3	4	Available supplies	Unit cost difference
	1	2	3	4		10	1
Colliery i	2	5	4	3	12	3	1
	3	1	3	3		21	2 ← largest
Requirements		6	11	17			
Unit cost difference		1	0	0			

We make x_{31} as large as possible, i.e. $x_{31} = 6$, and column 1 is deleted to give Table 3.12.

Table 3.12

		Power station j 1	2	3	4	Available supplies	Unit cost difference
	1		3	4		10	1 ← largest
Colliery i	2		4	3	12	3	1
	3	6	3	3		15	0
Requirements			11	17			
Unit cost difference			0	0			

We make x_{12} as large as possible, i.e. $x_{12} = 10$, and row 1 is deleted to give Table 3.13.

We make x_{23} as large as possible, i.e. $x_{23} = 3$. We must then have $x_{32} = 1$ and $x_{33} = 14$, to give the Vogel starting solution in Table 3.14.

Table 3.13

		Power station j					
		1	2	3	4	Available supplies	Unit cost difference
Colliery i	1		10				
	2		4	3	12	3	1 ← largest
	3	6	3	3		15	0
Requirements			1	17			
Unit cost difference			1	0			

Table 3.14

		Power station j				
		1	2	3	4	Available supplies
Colliery i	1		10			10
	2			3	12	15 Total cost = 102
	3	6	1	14		21
Requirements		6	11	17	12	

For this problem it so happens that the optimal solution is obtained, but this is not generally so unless $m = n = 2$.

For the problem of Table 3.15, neither the minimal unit cost method nor Vogel's method will give optimal solutions, as the reader should check once he has learnt how to get optimal solutions.

The question arises of why Vogel's method works so well. Let us look at Table 3.10. The method requires that we allocate as much as possible to the cell (2, 4). If we did not do so, then there would be subsequent allocations to row 2 with a minimal increase in unit cost of 2 units per unit allocation, and this is the largest increase possible if we had chosen to allocate the maximal amount to any cell other than (2, 4). By choosing (2, 4) we try to avoid this maximal penalty of 2 units which might be incurred by later allocations.

Table 3.15

		Destination j			
		1	2	3	Available supplies
Source i	1	4	4	6	10
	2	10	8	8	15
	3	9	12	10	21.
Requirements		6	16	24	

Improving the solution

We will use the north-west corner rule to find our starting solution (see Table 3.3), although the method now to be described for improving the solution also applies to any other starting solution method.

The method we will describe is called the *stepping-stone method* since it involves an operation which looks like stepping from one stone to another. Let us first of all illustrate this stepping-stone operation beginning with Table 3.3, which we now restate as Table 3.16 for convenience.

Table 3.16

		Power station j				
		1	2	3	4	Available supplies
Colliery i	1	6	4			10
	2		7	8		15
	3			9	12	21
Requirements		6	11	17	12	

Let us suppose that we wish to consider what change in the cost arises if we allocate a quantity θ to the cell (1, 3), i.e. make $x_{13} = \theta$. We must now adjust the other allocations to maintain row and column availability and requirement levels. In doing so we will allow changes in only the positive entries in Table 3.16. This corresponds to the simplex procedure in linear programming, where a vertex (basic feasible solution) has at most $m+n-1$ positive entries if our transportation problem has m rows and n columns, a point to which we will return later on.

The positive entry cells are the *stones* in the stepping stone operation. We will seek a circuit of such stones, including the new stone in cell (1, 3), with exactly two stones in each row and column. The circuit, which is unique in any problem, is given in Table 3.17.

Table 3.17

		Power station j				
		1	2	3	4	Available supplies
Colliery i	1	6	$4-\theta\leftarrow$	θ		10
	2		$7+\theta\rightarrow$	$8-\theta$		15
	3			9	12	21
Requirements		6	11	17	12	

From the unit cost Table 3.1, the additional cost arising from Table 3.17 is

$$\theta(4-3+4-3) = 2\theta > 0$$

if $\theta > 0$. Hence we would not allocate θ to cell (1, 3). If we, for example, allocate θ to cell (2, 4) we have Table 3.18.

Table 3.18

		Power station j				
		1	2	3	4	Available supplies
Colliery i	1	6	4			10
	2		7	$8-\theta\leftarrow$	θ	15
	3			$9+\theta\rightarrow$	$12-\theta$	21
Requirements		6	11	17	12	

The additional cost is

$$\theta(1-3+3-2) = -\theta$$

and if $\theta > 0$ we reduce the cost. We must keep all allocations non-negative, and, in Table 3.17, the maximal value of θ consistent with this is $\theta = 4$, whereas in

Table 3.18 the maximal value of θ is 8. This is, in effect, the standard *pivot row* selection rule used in the simplex method. Putting $\theta = 8$ in Table 3.18 we obtain an improved solution given in Table 3.19.

Table 3.19

		Power station j					
		1	2	3	4	Available supplies	
Colliery i	1	6	4			10	
	2		7		8	15	Total cost = 119
	3			17	4	21	
Requirements		6	11	17	21		

It would be possible to treat each location, with a non-positive allocation, in a similar manner and terminate the calculations if all the associated changes in cost were positive. For a problem with m rows and n columns this would involve $mn-m-n+1$ cells to be checked, with the corresponding stepping-stone operation, and could be quite time consuming if m or n were large. Fortunately, we can avoid having to do this by making use of a little of our linear programming knowledge. Let us now look at the full transportation method. Before doing so, however, let us examine the general balanced transportation problem in a linear programming context so that we will be able to see why the method works.

If we have m supply sources, n requirement points, and the total supply equals the total requirement, we have a *balanced problem*. If $\{a_i\}$ are the supply levels, and $\{b_j\}$ are the requirement levels, the linear programming formulation is as follows (check against the linear programming formulation of the colliery–power station example, noting that the inequalities for that problem may be reduced to equalities since it is a balanced problem):

$$\text{minimise} \quad M = \sum_i \sum_j c_{ij} x_{ij}$$

subject to

$$\sum_j x_{ij} = a_i,\ i = 1, 2, \ldots, m \qquad \text{(availabilities)}$$

$$\sum_i x_{ij} = b_j,\ j = 1, 2, \ldots, n \qquad \text{(requirements)}$$

$$x_{ij} \geqslant 0 \text{ for all } i \text{ and } j,$$

where x_{ij} is the amount transported from supply point i to requirement point j.

We have $m+n$ equations in mn variables. The equations are dependent in that any one may be deduced from the remaining $m+n-1$. From our linear programming results we seek $m+n-1$ basic variables, and this is why we keep $m+n-1$ allocations in each solution at each stage. The dual problem is easily stated:

$$\text{maximise } M = \sum_i a_i u_i + \sum_j b_j v_j$$

subject to

$$u_i + v_j \leqslant c_{ij} \qquad \text{for all } i, j$$

with $\{u_i\}$, $\{v_j\}$ unsigned because the primal problem is in equality form. The complementary slackness conditions require that, if $x_{ij} \neq 0$, then the corresponding dual constraint should be an equality (i.e. the slack variable of the dual constraint is 0) and, as will be seen in step (2) of the stepping-stone method, we require $u_i + v_j = c_{ij}$ whenever $x_{ij} > 0$. In the simplex method, the numbers $\{c_{ij} - u_i - v_j\}$ are the coefficients, in the objective function, of the variable $\{x_{ij}\}$ and, as with the linear programming method, we can increase x_{ij}, where $c_{ij} - u_i - v_j < 0$.

It should be noted that, whereas in the linear programming chapter, the complementary slackness conditions were used only at an *optimal* solution, for the transportation problem they are used at each step of the algorithm, and this is a special feature of the transportation problem and not of *all* linear programming problems.

Before proceeding with the algorithm, let us recall, from the linear programming chapter, that the dual problem can be given physical interpretations in terms of market prices for constraint levels. Thus, in this problem, suppose that the manager responsible for the transportation problem was given a price u_i for each unit loaded at supply point i, and price v_j for unloading a unit at destination j. Then the total price paid to the market is exactly the objective function given for all the operations required to move the goods from the supply points to the destination points. The constraints placed by the manager would be that his unit costs should be no higher than they were before, and the market would seek to maximise its income.

Let us now look at the steps we follow in the transportation algorithm once we have a starting solution.

(1) Identify the unit costs associated with the positive allocations of the starting solution. These are given in Table 3.20.

(2) With each row i associate a dual variable u_i, which we will call a row *shadow cost*, and with each column j associate a dual variable v_j, which we will call a column *shadow cost*, in such a way that the sum of u_i and v_j equals the unit cost in cell (i, j) for all positive cells. Thus we require

$$u_1 + v_1 = 2$$
$$u_1 + v_2 = 3$$

Table 3.20

		Power station j 1	2	3	4	Available supplies
Colliery i	1	2	3			10
	2		4	3		15
	3			3	2	21
Requirements		6	11	17	12	

$$u_2 + v_2 = 4$$
$$u_2 + v_3 = 3$$
$$u_3 + v_3 = 3$$
$$u_3 + v_4 = 2$$

We have 6 ($= m+n-1$) equations in 7 ($= m+n$) variables, and can set any one of the 7 variables equal to any number (we choose the number 0) and solve for the rest.

It is easily seen, for example, that in the above equations we can add any constant k to the values of $\{u_i\}$, and subtract the same constant k from the values of $\{v_j\}$, and still get a solution to the equations.

An alternative way of seeing that we have a degree of freedom in our solution for the $\{u_i\}$ and $\{v_j\}$ is to note, as has already been indicated, that there is no loss in removing any single constraint, and that hence we merely drop the corresponding dual variable by setting it equal to 0 (or, indeed, to any value). The calculations can be done quite easily on Table 3.20 directly to give Table 3.21.

Table 3.21

			Power station j $v_1 = 0$ 1	$v_2 = 1$ 2	$v_3 = 0$ 3	$v_4 = -1$ 4	Available supplies
Colliery i	$u_1 = 2$	1	2	3			10
	$u_2 = 3$	2		4	3		15
	$u_3 = 3$	3			3	2	21
Requirements			6	11	17	12	

(3) For each cell (i, j) subtract the sum of the row and column shadow costs from the unit cost for that cell. Cells corresponding to the positive allocations will, because of step (2), always have a zero value for this. Table 3.22 gives the results of this step.

Table 3.22

		Power station j				
		1	2	3	4	Available supplies
	1	0	0	2	4	10
Colliery i	2	2	0	0	−1	15
	3	$\boxed{-2}$	−1	0	0	21
Requirements		6	11	17	12	

From our knowledge of linear programming, it may be shown that the entries in Table 3.22 are the coefficients of the objective function we would obtain if our basic variables had been chosen to correspond to the positive allocation cells, the entries being zero for the latter. Any extra allocation θ to any cell, followed by an adjustment using the stepping stone operation, will change the total cost by an amount $\theta \times$ the corresponding entry in Table 3.22, just as in linear programming. This may also be checked a different way using the derivation of $\{u_i\}, \{v_j\}$ in step (3). Thus suppose we allocate θ to cell (3, 1), and use the stepping stone operation. We obtain Table 3.23.

Table 3.23

		Power station j				
		1	2	3	4	Available supplies
	1	$6-\theta$	$4+\theta$			10
Colliery i	2		$7-\theta$	$8+\theta$		15
	3	θ		$9-\theta$	12	21
Requirements		6	11	17	12	

The addition to the total cost is

$$\begin{aligned}
&\theta(1-2+3-4+3-3)\\
&=\theta((1-u_3-v_1)-(2-u_1-v_1)+(3-u_1-v_2)\\
&\quad -(4-u_2-v_2)+(3-u_2-v_3)-(3-u_3-v_3)) \qquad \text{(as is easily checked)}\\
&=\theta(1-u_3-v_1) \qquad \text{(since the remaining brackets are all zero by virtue of step (3))}\\
&=-2\theta.
\end{aligned}$$

(4) We make θ as large as possible (i.e. $\theta = 6$) to derive Table 3.24, which becomes the starting solution for a repetition of all the steps (1)–(4).

Table 3.24

		Power station j					
		1	2	3	4	Available supplies	
	1		10			10	
Colliery i	2		1	14		15	Total cost = 115
	3	6		3	12	21	
Requirements		6	11	17	12		

The steps are repeated until the table corresponding to Table 3.22 has all entries non-negative, and the corresponding allocation solution will be optimal. Again note that this is the simplex method at work. In carrying out the calculation it is much simpler to put some calculations on the same table. We will use the convention shown in Table 3.25 in constructing Table 3.26 for a given cell (i, j).

Table 3.25

	Power station j
Colliery i	c_{ij} $x_{ij} \pm \theta$ $\boxed{c_{ij}-u_i-v_j}$

Here c_{ij} is the unit cost and we put the most negative value of $c_{ij}-u_i-v_j$ inside a square, and encircle the $\{c_{ij}\}$ corresponding to the positive allocations x_{ij} at each stage. Tables 3.27 and 3.28 complete the calculations.

Table 3.26

Power station j

			$v_1 = 0$ 1	$v_2 = 3$ 2	$v_3 = 2$ 3	$v_4 = 1$ 4	Available supplies
	$u_1 = 0$	1	2; 2	3; 10; 0	4; 0	5; 4	10
Colliery i	$u_2 = 1$	2	5; 4	4; $1-\theta$; 0	3; $14+\theta$; 0	1; -1	15
	$u_3 = 1$	3	1; 6; 0	3; θ; $\boxed{-1}$	3; $3-\theta$; 0	2; 12; 0	21 Total cost $= 115$
Requirements			6	11	17	12	$\theta_{max} = 1$

Table 3.27

Power station j

			$v_1 = 0$ 1	$v_2 = 2$ 2	$v_3 = 2$ 3	$v_4 = 1$ 4	Available supplies
	$u_1 = 1$	1	2; 1	3; 10; 0	4; 1	5; 3	10
Colliery i	$u_2 = 1$	2	5; 4	4; 1	3; $15-\theta$; 0	1; θ; $\boxed{-1}$	15
	$u_3 = 1$	3	1; 6; 0	3; 1; 0	3; $2+\theta$; 0	2; $12-\theta$; 0	21 Total cost $= 114$
Requirements			6	11	17	12	$\theta_{max} = 12$

Table 3.28

Power station j

			$v_1 = 0$ 1	$v_2 = 2$ 2	$v_3 = 2$ 3	$v_4 = 0$ 4	Available supplies
	$u_1 = 1$	1	2; 1	3; 10; 0	4; 1	5; 4	10
Colliery i	$u_2 = 1$	2	5; 4	4; 1	3; 3; 0	1; 12; 0	15
	$u_3 = 1$	3	1; 6; 0	3; 1; 0	3; 14; 0	2; 1	21 Total cost $= 102$
Requirements			6	11	17	12	

Note that we have two equally negative values of $c_{ij} - u_i - v_j$ equal to -1 and we choose cell (3, 3). We may equally well choose cell (2, 4) for the next step. Either choice will work.

In Table 3.28 all the entries $c_{ij} - u_i - v_j$ in the bottom right-hand quarters of each square are non-negative, and hence the solution is optimal:

$$x_{12} = 10,\ x_{23} = 3,\ x_{24} = 12,\ x_{31} = 6,\ x_{32} = 1,\ x_{33} = 14,$$
$$x_{11} = x_{13} = x_{14} = x_{21} = x_{22} = x_{34} = 0,\ M = 102.$$

Let us now consider some special points relating to the transportation problem.

3.4 Special aspects

(i) Unbalanced transportation problems

For some problems we may have total supply not equal to total requirement. Consider the example in Table 3.29.

Table 3.29

		Power station j 1	2	3	4	Available supplies
Colliery i	1	2	3	4	5	11
	2	5	4	3	1	16
	3	1	3	3	2	22
Requirements		6	11	17	12	

The total available supplies (49) are greater than the total requirement (46). Hence, some supplies will not be used. We add a dummy column to obtain a balanced transportation problem in Table 3.30, where $\{p_i\}$ are the penalties for each unit of supply i not being used.

Table 3.30

		Power station j 1	2	3	4	Dummy 5	Available supplies
Colliery i	1	2	3	4	5	p_1	11
	2	5	4	3	1	p_2	16
	3	1	3	3	2	p_3	22
Requirements		6	11	17	12	3	

Unless stated otherwise the $\{p_i\}$ are usually equal to 0.

Likewise if total supplies are less than total requirements, we construct a standard balanced transportation problem with a dummy row.

In general, whether the problem is balanced or not, we may not need to use all the supplies or to meet all the requirements. We then add a dummy row and a dummy column to obtain Table 3.31, where $\{p_i\}$, $\{q_j\}$ are specified penalties for units of supply not used or requirements not met respectively. Here a_4 is the dummy supply level and b_5 the dummy requirement level. We require $a_1 + a_2 + a_3 + a_4 = b_1 + b_2 + b_3 + b_4 + b_5$. The values of a_4, b_5 merely have to be big enough to allow no requirements to be met or no supplies to be used in the original problem; e.g.

$$a_4 = b_1 + b_2 + b_3 + b_4, \quad b_5 = a_1 + a_2 + a_3$$

will do.

Table 3.31

		Power station j 1	2	3	4	Dummy 5	Available supplies
Colliery i	1	2	3	4	5	p_1	a_1
	2	5	4	3	1	p_2	a_2
	3	1	3	3	2	p_3	a_3
Dummy	4	q_1	q_2	q_3	q_4	0	a_4
Requirements		b_1	b_2	b_3	b_4	b_5	

(ii) Non-unique optimal solutions

Suppose that in Table 3.1 we had $c_{13} = 3$ instead of $c_{13} = 4$. Then, following the stepping-stone procedure we would end up with Table 3.32 instead of Table 3.28.

Table 3.32

			Power station j $v_1 = 0$ 1	$v_2 = 2$ 2	$v_3 = 2$ 3	$v_4 = 0$ 4	Available supplies
Colliery i	$u_1 = 1$	1	2; 1	3; $10-\theta$; 0	3; θ; 0	5; 4	10
	$u_2 = 1$	2	5; .; 4	4; 1	3; 3; 0	1; 12; 0	15
	$u_3 = 1$	3	1; 6; 0	3; $1+\theta$; 0	3; $14-\theta$; 0	2; 1	21
Requirements			6	11	17	12	Total cost = 102

We can see that $c_{13} - u_1 - v_3 = 0$. If we allocate a quantity θ to the cell (1, 3), and use the stepping-stone operation, the change in cost is $\theta.0 = 0$ for all θ such that $0 \leqslant \theta \leqslant 10$. Thus for all such θ we will obtain solutions each with a cost of 102, and the solution arrived at with $\theta = 0$ is not uniquely optimal. Again this is the simplex method at work (see Chapter 1 for non-uniqueness in general linear programming).

(iii) Degeneracy

As we have seen in linear programming, degeneracy can arise when one, at least, of the basic variables is zero. In the transportation problem this corresponds to one of the $m+n-1$ allocations of our solutions being zero. Let us see how this may arise. Suppose that instead of Table 3.1 we have Table 3.33.

Table 3.33

		Power station j				
		1	2	3	4	Available supplies
	1	2	3	4	5	10
Colliery i	2	5	4	3	1	15
	3	1	3	3	2	21
Requirements		10	11	11	14	

If we use the north-west corner rule we obtain $x_{11} = 10$ and row 1 *and* column 1 would be exhausted. If we deleted both and continued with the north-west corner rule we would obtain Table 3.34. This has only 5, instead of 6, positive allocations and gives a degenerate solution.

Table 3.34

		Power station j				
		1	2	3	4	Available supplies
	1	10				10
Colliery i	2		11	4		15
	3			7	14	21
Requirements		10	11	11	14	

A look at the stepping-stone method will show that each allocation x_{ij} will be equal to $\pm$ (the sum (may be zero) of some of the available source levels $\{a_i\}$ minus the sum (may be zero) of some of the requirement levels $\{b_j\}$). Thus in Table 3.16

$$x_{11} = b_1 - 0 = 6, \quad x_{12} = a_1 - b_1 = 10-6 = 4, \quad x_{13} = 0-0 = 0,$$
$$x_{14} = 0-0 = 0, \quad x_{21} = 0-0 = 0,$$
$$x_{22} = b_2 - (a_1 - b_1) = b_1 + b_2 - a_1 = 7;$$

and so on. Thus we can avoid any of the stepping-stone entries being zero if we can avoid any non-null differences of the sums of available supply levels and requirements levels being zero. We do this by changing Table 3.33 to Table 3.35.

Table 3.35

		Power station j				
		1	2	3	4	Available supplies
	1	2	3	4	5	$10+\varepsilon$
Colliery i	2	5	4	3	1	$15+\varepsilon$
	3	1	3	3	2	$21+\varepsilon$
Requirements		$10+3\varepsilon$	11	11	14	

It is easily checked that the above property regarding the differences of the sums of sub-sets of the available supply levels and requirement levels is satisfied if ε is small enough, e.g. $\varepsilon = \frac{1}{4}$. For a general $m \times n$ problem, with $n \geqslant m$, with whole number supply and requirement levels, we change a_i to $a_i + \varepsilon$, b_1 to $b_1 + m\varepsilon$, and leave the remaining levels unaltered. The calculations then proceed as before and ε is set equal to 0 when they terminate. For computer applications it is sufficient to put $\varepsilon = 1/2n$. The modification is usually done at the beginning, even though degeneracy may not have arisen without it. Alternatively, one can wait until degeneracy arises and modify the solution at that stage, although this may require some searching of the empty allocation cells before a solution is found. Thus if we arrive at Table 3.34, then Table 3.36 will suffice.

(iv) Post-optimality sensitivity analysis

We have dealt with sensitivity analysis in linear programming. We do so with respect to variations in the unit costs only. Suppose that Table 3.1 is replaced by Table 3.37.

Table 3.36

		Power station j				
		1	2	3	4	Available supplies
Colliery i	1	$10+\varepsilon$				$10+\varepsilon$
	2	2ε	11	$4-\varepsilon$		$15+\varepsilon$
	3			$7+\varepsilon$	14	$21+\varepsilon$
Requirements		$10+3\varepsilon$	11	11	14	

Table 3.37

		Power station j				
		1	2	3	4	Available supplies
Colliery i	1	$2+\delta_{11}$	3	$4+\delta_{13}$	$5+\delta_{14}$	10
	2	$5+\delta_{21}$	$4+\delta_{22}$	3	1	15
	3	1	3	3	$2+\delta_{34}$	21
Requirements		6	11	17	12	

In order to keep matters simple, we have kept the unit costs associated with the positive solution allocations unaltered, and made small perturbations $\{\delta_{ij}\}$ in the remaining unit cost entries. We stress that this is merely to keep computations simple and that, in practice, variations in *all* the unit costs would be considered.

If we take the solution given in Table 3.28 for the original problem, the $\{c_{ij}-u_i-v_j\}$ entries for the new cost table are given in Table 3.38, noting that the $\{u_i\}, \{v_j\}$ values remain the same since we have calculated these for the same final solution of Table 3.28.

Then the original solution given in Table 3.28 is still optimal for the new unit cost matrix if all the entries in Table 3.38 are non-negative, i.e.

$$\delta_{11} \geqslant -1, \quad \delta_{13} \geqslant -1, \quad \delta_{14} \geqslant -4, \quad \delta_{21} \geqslant -4,$$
$$\delta_{22} \geqslant -1, \quad \delta_{34} \geqslant -1.$$

This is equivalent to

$$c_{11} \geqslant 1, \quad c_{13} \geqslant 3, \quad c_{14} \geqslant 4, \quad c_{21} \geqslant 1, \quad c_{22} \geqslant 3, \quad c_{34} \geqslant 1.$$

Table 3.38

			Power station j				
			$v_1=0$	$v_2=2$	$v_3=2$	$v_4=0$	
			1	2	3	4	Available supplies
	$u_1=1$	1	$1+\delta_{11}$	0	$1+\delta_{13}$	$4+\delta_{14}$	10
Colliery i	$u_2=1$	2	$4+\delta_{21}$	$1+\delta_{22}$	0	0	15
	$u_3=1$	3	0	0	0	$1+\delta_{34}$	21
Requirements			6	11	17	12	

If we allow the unit costs associated with the positive allocation solutions in Table 3.28 to change also, then a similar analysis is possible, but the $\{u_i\}, \{v_j\}$ must be recalculated, i.e.

$$v_1 = 0, \; u_3 = 1+\delta_{31}, \quad v_2 = 2+\delta_{32}-\delta_{31}, \quad v_3 = 2+\delta_{33}-\delta_{31},$$
$$u_2 = 1+\delta_{23}-\delta_{33}+\delta_{31}, \quad v_4 = \delta_{24}-\delta_{23}+\delta_{33}+\delta_{31},$$
$$u_1 = 1+\delta_{12}-\delta_{32}+\delta_{31}.$$

(v) Non-transportation problems formulated as transportation problems

There are problems which are not physically transportation problems but which, by appropriate identification of sources and destinations, may be profitably formulated as such. Consider the following problems.

A caterer problem

A caterer undertakes to organise garden parties for a week and decides that he will need a supply of new napkins, each of which costs *a* pence. He will buy as few as possible, and make use of a laundry service by sending used napkins for laundering. There is a standard laundry service, costing *b* pence per napkin, which returns napkins within 4 days. There is an express service, costing *c* pence per napkin, which returns napkins within 2 days. The caterer estimates that, during the seven days, he requires 130, 70, 60, 100, 80, 90 and 120 napkins respectively. How many new napkins should he buy and how should he use the laundry service to keep the total costs as low as possible?

Let us identify *sources*. Each napkin used on a given day must either be a new napkin (source 1) or a used napkin from a previous day, which has been laundered in time (sources 2–8).

The *destinations* (1–7) are then the days on which napkins are used, together with a final inventory of napkins when the activities have been completed (8). We then derive a transportation problem of the form shown in Table 3.39.

Table 3.39

		Days on which used j							Final inventory	
		1	2	3	4	5	6	7	8	
New napkins	1	a	a	a	a	a	a	a	0	M
i Used napkins	2			c	c	b	b	b	0	130
	3				c	c	b	b	0	70
	4					c	c	b	0	60
	5						c	c	0	100
	6							c	0	80
	7								0	90
	8								0	120
		130	70	60	100	80	90	120	M	

This labelling is chosen to avoid unnecessary problems which, for example, the north-west corner rule would generate.

Let us assume that $a > c > b$. In Table 3.39, M has to be large enough to give a feasible schedule. We could make M = total requirement, 650. However, it is clear that no optimal solution requires more than the maximal total over any 4 successive days (i.e. slow service), which is 390, and hence we set $M = 390$.

The unit costs in the empty squares are made large enough to ensure they are never used, since feasible solutions, using any of the given starting methods, will exist.

A transhipment problem

Let us suppose we have a standard transportation problem with unit costs, supplies, and requirements as given in Table 3.40, where we use primes to denote the initial destinations.

The optimal solution is $x_{11'} = 1$, $x_{12'} = 1$, $x_{22'} = 2$, $x_{23'} = 4$, $x_{31'} = 2$, $x_{34'} = 5$, with a minimal cost of 196.

Now let us suppose that we *tranship* items so that, for example, we can send items from 1 to 2′ via 2, or via 3′, and so on. Let the unit transportation costs *within* $\{1, 2, 3\}$, and *within* $\{1', 2', 3', 4'\}$ be as shown in Tables 3.41 and 3.42.

To complete the picture we need the reverse costs of Table 3.40, which we assume are not symmetric. These are given in Table 3.43.

We may now formulate the problem, allowing for transhipment, as one large transportation problem as follows. The maximal amount one can tranship into

Table 3.40

		Destinations j'				
		1′	2′	3′	4′	Available supplies
Sources i	1	13	11	15	20	2
	2	17	14	12	13	6
	3	18	18	15	12	7
Requirements		3	3	4	5	

Table 3.41

		Destinations j		
		1	2	3
Sources i	1	0	6	5
	2	5	0	4
	3	4	4	0

Table 3.42

		Destinations j'			
		1′	2′	3′	4′
Sources i'	1′	0	5	4	4
	2′	5	0	2	8
	3′	4	2	0	9
	4′	5	8	9	0

Table 3.43

		Destinations j		
		1	2	3
Sources i'	1′	14	19	20
	2′	10	12	17
	3′	17	13	14
	4′	18	12	14

and out of any location is 15 (the total available supplies at $\{1, 2, 3\}$). Hence we add 15 to all initial supplies and requirements, noticing that the original requirements for $\{1, 2, 3\}$ are all 0, and the original supplies for $\{1', 2', 3', 4'\}$ are 0. Table 3.44 gives the final table.

Table 3.44

		New destinations j			j'				
		1	2	3	1′	2′	3′	4′	Available supplies
New sources i	1	0	6	5	13	11	15	20	17
	2	5	0	4	17	14	12	13	21
	3	4	4	0	18	18	15	12	22
i'	1′	14	19	20	0	5	4	5	15
	2′	10	12	17	5	0	2	8	15
	3′	17	13	14	4	2	0	9	15
	4′	18	12	14	5	8	9	0	15
Requirements		15	15	15	18	18	19	20	

It is to be noted that the addition to supplies and requirements (which we have chosen to be 15) is arbitrary, providing it is large enough. Any excess is channelled to the diagonal, i.e. unused. The solution is given in Table 3.45.

The diagonal entries may be removed since they merely indicate unused possible transhipments.

Table 3.45

		New destinations j			j'				
		1	2	3	1′	2′	3′	4′	Available supplies
New sources i	1	13			3	1			17
	2		15			2	4		21
	3	2		15				5	22
i'	1′				15				15
	2′					15			15
	3′						15		15
	4′							15	15
Requirements		15	15	15	18	18	19	20	Total cost = 194

There is an alternative method, related to optimal routing, which we will discuss later on in Exercise 3 of Chapter 7. For each (i, j') we calculate the minimal cost for each unit sent from i to j', via intermediate points if need be. Let $c^*_{ij'}$ be this minimal cost. The problem is then reformulated in exactly the same way as in Table 3.40, using the new costs. From the solution in this form we can calculate all the requisite transhipment quantities.

Exercises

1 Solve the following cost minimisation transportation problems, commenting on uniqueness in each case, where i, j, A, R denote supply points, destinations, availabilities and requirements.

(a)

		j 1	2	3	4	A
i	1	1	2	3	4	7
	2	4	3	2	0	8
	3	0	2	2	1	11
	R	5	6	9	6	

(b)

		j 1	2	3	4	A
i	1	10	5	6	10	15
	2	8	2	7	6	26
	3	12	3	4	8	50
	R	16	10	30	35	

(c)

i \ j	1	2	3	A
1	6	8	4	6
2	4	9	3	10
3	1	2	6	15
R	14	12	5	

(d)

i \ j	1	2	3	4	A
1	2	1	4	5	20
2	5	4	3	5	10
3	6	2	1	3	35
4	3	6	2	8	16
R	14	20	20	27	

(e)

i \ j	1	2	3	A
1	10	40	20	1
2	16	34	28	3
3	20	30	40	3
R	2	1	3	

2 (a) In Exercise 1(a), suppose the requirement at destination 1 increases by two units, and that the supplies may be increased to meet this. What changes in the supply levels give an optimal solution? What would the answer be if the requirement for destination 4 increased by two units?

(b) In Exercise 1(a), suppose that a fifth destination point is added, and a fourth source added to supply exactly the requirement for that destination. If $\{c_{4j}\}$, $\{c_{i5}\}$ are the unit costs for the additional parts of the first problem, determine inequalities between these unit costs for which the original solution will remain optimal.

(c) In Exercise 1(b), suppose that the requirement at destination 1 becomes 15. What is the solution to the problem then, and is it unique?

(d) In Exercise 1(c), suppose that there are penalties q_1, q_2, q_3 respectively for each unit not supplied to destinations 1, 2, 3, and penalties p_1, p_2, p_3 respectively for not using a unit of each supply. Find inequalities between the $\{q_j\}, \{p_i\}$ for which it is optimal not to incur any penalties.

3 (a) A company has to supply a product in each of three months. Table 3.46 gives the details. There is also a cost of 1 for each month a unit is kept in stock. Find a supply schedule which will meet the requirements at a minimal total cost. Is your solution unique?

(This problem will be treated in a different manner in Exercise 4 of Chapter 12 on dynamic programming.)

(b) The company knows that the stockholding cost is only an estimate. For what range of the stockholding cost would your solution in (a) still remain optimal?

Table 3.46

Month	1	2	3
Requirements	7	8	10
Maximal amount supplied by normal time operation	6	6	6
Maximal amount supplied by overtime operation	4	4	4
Cost per unit during normal time	2	6	4
Cost per unit during overtime	3	9	6

4 Consider the transportation problem data given in Table 3.47.

Table 3.47

i \ j	1	2	3	4	A
1	2	3	4	5	10
2	5	4	3	1	15
3	1	3	3	2	21
R	6	11	17	12	

(a) Formulate this problem as a linear program in six constraints.

(b) Use the linear programming method of Chapter 2 to show that the solution of Table 3.28 is a unique optimal solution.

5 (a) A company has four warehouses and supplies four customers. The distances involved are small, and the company charges its customers in terms of the charges per unit involved in loading at the warehouses and unloading at the destinations according to Table 3.48. Use the transportation algorithm to find the optimal solution. Is it unique?

Table 3.48

Warehouse	Unit loading charge	Available supplies	Customer	Unit unloading charge	Requirements
1	1	20	1	1	15
2	2	20	2	2	35
3	3	30	3	3	15
4	4	10	4	4	10

(b) Explain
  (i) using a linear programming formulation
  (ii) using the transportation algorithm

  why the set of all optimal solutions is the set of all solutions which use up completely the supplies from the first three supply points.

6 Explain what modifications you would make to the transportation algorithm to convert maximisation problems to minimisation problems, giving reasons.

4

Assignment Problems

4.1 Introduction

Another special linear programming problem is the *assignment problem*. Indeed it is also a very special form of the transportation problem. Although, as such, it may be solved in principle using linear programming, or the transportation stepping-stone algorithm, there is a simpler way of solving the problem, unless it is very large and needs to be computerised, known as the *Hungarian algorithm*, being a development by H. W. Kuhn of some work of the Hungarian E. Egerváry.

As we shall see later on, the linear programming formulation is a very degenerate one which leads to some difficulties, but these can be overcome.

As an example let us suppose that we have a set I of people and a set J of jobs, each set having an equal number of members. Each person in I has to be assigned to exactly one job in J.

If element i in I is assigned to element j in J, there is a cost (or some other measure of performance) c_{ij}. The problem is to determine a total assignment which minimises the sum of the $\{c_{ij}\}$ in the assignment.

The standard assignment problem is in *minimisation* form, in which case c_{ij} must be non-negative. *Maximisation* problems can be handled by using the fact that the maximal value of a function is equal to the negative of the minimal value of its negative, in which case we can cater for negative c_{ij} values.

Let us consider a few examples.

4.2 Some examples

(i) *Medley relay*

A swimming team has 5 members and has to decide which member will swim which leg of a 5 leg relay race. The times in seconds for each swimmer, in excess of a base time for each leg (the base time does not matter since it will be incurred anyhow for any assignment) are given in Table 4.1.

The reader may like to try to solve this. The minimal total time and assignments are given later in the text where this example is needed to illustrate the Hungarian algorithm. Without the aid of the algorithm one might try to solve it using similar procedures to those used to obtain a good starting solution for the transportation problem. Thus one might use the minimal unit cost method: find the smallest c_{ij}, and assign i to j where this occurs, then delete the ith row and jth column,