Allocation problems 2D

1 $\begin{pmatrix} 48 & 34 & 140 \\ 140 & 37 & 67 \\ 53 & 43 & 56 \end{pmatrix}$ reducing rows $\begin{pmatrix} 14 & 0 & 106 \\ 103 & 0 & 30 \\ 10 & 0 & 13 \end{pmatrix}$

reducing columns $\begin{pmatrix} 4 & 0 & 93 \\ 93 & 0 & 17 \\ 0 & 0 & 0 \end{pmatrix}$

Minimum uncovered element is 4 $\begin{pmatrix} 0 & 0 & 89 \\ 89 & 0 & 13 \\ 0 & 4 & 0 \end{pmatrix}$

Solution: P – L (48)
Q – M (37) cost £141
R – N (56)

2 $\begin{pmatrix} 38 & 47 & 55 & 53 \\ 32 & 130 & 47 & 64 \\ 130 & 53 & 43 & 130 \\ 41 & 48 & 52 & 47 \end{pmatrix}$ reducing rows $\begin{pmatrix} 0 & 9 & 17 & 15 \\ 0 & 98 & 15 & 32 \\ 87 & 10 & 0 & 87 \\ 0 & 7 & 11 & 6 \end{pmatrix}$

reducing columns $\begin{pmatrix} 0 & 2 & 17 & 9 \\ 0 & 91 & 15 & 26 \\ 87 & 3 & 0 & 81 \\ 0 & 0 & 11 & 0 \end{pmatrix}$

Minimum uncovered element is 2 $\begin{pmatrix} 0 & 0 & 17 & 7 \\ 0 & 89 & 15 & 24 \\ 87 & 1 & 0 & 79 \\ 2 & 0 & 13 & 0 \end{pmatrix}$

Solution

R – E (47)
S – D (32)
T – F (43) cost £169
U – G (47)

3 $\begin{pmatrix} 46 & 53 & 67 & 75 \\ 48 & 150 & 61 & 78 \\ 42 & 46 & 53 & 62 \\ 39 & 50 & 150 & 73 \end{pmatrix}$ reducing rows $\begin{pmatrix} 0 & 7 & 21 & 29 \\ 0 & 102 & 13 & 30 \\ 0 & 4 & 11 & 20 \\ 0 & 11 & 111 & 34 \end{pmatrix}$

reducing columns $\begin{pmatrix} 0 & 3 & 10 & 9 \\ 0 & 98 & 2 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 7 & 100 & 14 \end{pmatrix}$

Minimum uncovered element is 2 $\begin{pmatrix} 0 & 1 & 8 & 7 \\ 0 & 96 & 0 & 8 \\ 2 & 0 & 0 & 0 \\ 0 & 5 & 98 & 12 \end{pmatrix}$

Minimum uncovered element is 1 $\begin{pmatrix} 0 & 0 & 8 & 6 \\ 0 & 95 & 0 & 7 \\ 3 & 0 & 1 & 0 \\ 0 & 4 & 98 & 11 \end{pmatrix}$

Solution: A – Q (53)
B – R (61)
C – S (62) cost £215
D – P (39)

4 a The initial cost matrix is shown below:

	Task *R*	**Task *S***	**Task *T***	**Task *U***	**Task *V***
Worker *J*	143	112	149	137	✗
Worker *K*	149	106	153	115	267
Worker *L*	137	109	143	121	✗
Worker *M*	157	✗	✗	134	290
Worker *N*	126	101	132	111	253

In the matrix an ✗ indicates that the worker is not qualified to carry out the task, these are known as forbidden allocations. Therefore, for all such entries we replace the ✗ with a large value (1000) to make the assignments ‘unattractive’. Therefore, the matrix becomes:

	Task *R*	**Task *S***	**Task *T***	**Task *U***	**Task *V***
Worker *J*	143	112	149	137	1000
Worker *K*	149	106	153	115	267
Worker *L*	137	109	143	121	1000
Worker *M*	157	1000	1000	134	290
Worker *N*	126	101	132	111	253

4 b As the table above is an n by n matrix (with $n = 5$), we do not need to add any additional dummy rows or columns.
The smallest numbers in rows 1, 2, 3, 4 and 5 are 112, 106, 109, 134 and 101. We reduce rows first by subtracting these numbers from each element in the row. The table becomes:

	Task R	Task S	Task T	Task U	Task V
Worker J	31	0	37	25	888
Worker K	43	0	47	9	161
Worker L	28	0	34	12	891
Worker M	23	866	866	0	156
Worker N	25	0	31	10	152

The smallest numbers in columns 1, 2, 3, 4 and 5 are 23, 0, 31, 0 and 152. We reduce columns by subtracting these numbers from each element in the column. Therefore, the reduced cost matrix is:

	Task R	Task S	Task T	Task U	Task V
Worker J	8	0	6	25	736
Worker K	20	0	16	9	9
Worker L	5	0	3	12	739
Worker M	0	866	835	0	4
Worker N	2	0	0	10	0

We now apply the Hungarian algorithm.

We can cover all the zeros in three lines, so the solution is not optimal:

	Task R	Task S	Task T	Task U	Task V
Worker J	8	0	6	25	736
Worker K	20	0	16	9	9
Worker L	5	0	3	12	739
Worker M	0	866	835	0	4
Worker N	2	0	0	10	0

	Task R	Task S	Task T	Task U	Task V
Worker J	8	0	6	25	736
Worker K	20	0	16	9	9
Worker L	5	0	3	12	739
Worker M	0	866	835	0	4
Worker N	2	0	0	10	0

The smallest uncovered element is 3, so we augment the matrix as follows:

- Add 3 to the elements covered by two lines circled in red
- Leave the elements covered by just one line unchanged
- Subtract 3 from the uncovered elements

This gives the following matrix:

4 b (continued)

	Task *R*	Task *S*	Task *T*	Task *U*	Task *V*
Worker *J*	5	0	3	22	733
Worker *K*	17	0	13	6	6
Worker *L*	2	0	0	9	736
Worker *M*	0	869	835	0	4
Worker *N*	2	3	0	10	0

We can cover all the zeros in four lines, so the solution is not optimal:

	Task *R*	Task *S*	Task *T*	Task *U*	Task *V*
Worker *J*	5	0	3	22	733
Worker *K*	17	0	13	6	6
Worker *L*	2	0	0	9	736
Worker *M*	0	869	835	0	4
Worker *N*	2	3	0	10	0

	Task *R*	Task *S*	Task *T*	Task *U*	Task *V*
Worker *J*	5	0	3	22	733
Worker *K*	17	0	13	6	6
Worker *L*	2	0	0	9	736
Worker *M*	0	869	835	0	4
Worker *N*	2	3	0	10	0

The smallest uncovered element is 3, so we augment the matrix as follows:

- Add 3 to the elements covered by two lines circled in red
- Leave the elements covered by just one line unchanged
- Subtract 3 from the uncovered elements

This gives the following matrix:

	Task *R*	Task *S*	Task *T*	Task *U*	Task *V*
Worker *J*	2	0	0	19	730
Worker *K*	14	0	10	3	3
Worker *L*	2	3	0	9	736
Worker *M*	0	872	835	0	4
Worker *N*	2	6	0	10	0

We can cover all the zeros in four lines, so the solution is not optimal:

	Task *R*	Task *S*	Task *T*	Task *U*	Task *V*
Worker *J*	2	0	0	19	730
Worker *K*	14	0	10	3	3
Worker *L*	2	3	0	9	736
Worker *M*	0	872	835	0	4
Worker *N*	2	6	0	10	0

4 b (continued)

	Task *R*	**Task *S***	**Task *T***	**Task *U***	**Task *V***
Worker *J*	2	0	0	19	730
Worker *K*	14	0	10	3	3
Worker *L*	2	3	0	9	736
Worker *M*	0	872	835	0	4
Worker *N*	2	6	0	10	0

The smallest uncovered element is 2, so we augment the matrix as follows:

- Add 2 to the elements covered by two lines
- Leave the elements covered by just one line unchanged
- Subtract 2 from the uncovered elements

This gives the following matrix:

	Task *R*	**Task *S***	**Task *T***	**Task *U***	**Task *V***
Worker *J*	0	0	0	17	728
Worker *K*	12	0	10	1	1
Worker *L*	0	3	0	7	734
Worker *M*	0	874	837	0	4
Worker *N*	2	8	2	10	0

We can cover all the zeros with five lines. There are two optimal solutions of:

Worker *J* – Task *R* (143)
Worker *K* – Task *S* (106)
Worker *L* – Task *T* (143)
Worker *M* – Task *U* (134)
Worker *N* – Task *V* (253)

or

Worker *J* – Task *T* (149)
Worker *K* – Task *S* (106)
Worker *L* – Task *R* (137)
Worker *M* – Task *U* (134)
Worker *N* – Task *V* (253)

Both solutions give a minimum time of 779 minutes.
Minimum time = 143 + 106 + 143 + 134 + 253 = 779 minutes, or
Minimum time = 149 + 106 + 137 + 134 + 253 = 779 minutes.

5 a The initial cost matrix is shown below:

	1	**2**	**3**	**4**	**5**
P	25	42	38	52	–
Q	43	37	29	46	55
R	30	26	44	35	47
S	36	41	–	40	53
T	39	45	37	46	49

As the table above is an n by n matrix (with $n = 5$), we do not need to add any additional dummy rows or columns.
To maximise the profit with incomplete data, first subtract every number from the largest value in the table. Then replace the '–' forbidden allocations with a large value (1000) to make the assignments 'unattractive'.
The largest value in the table is 55. Subtracting every value from 55 the table becomes:

	1	**2**	**3**	**4**	**5**
P	30	13	17	3	–
Q	12	18	26	9	0
R	25	29	11	20	8
S	19	14	–	15	2
T	16	10	18	9	6

Replacing the '–' entries with the value of 1000 – 55 = 945 gives:

	1	**2**	**3**	**4**	**5**
P	30	13	17	3	945
Q	12	18	26	9	0
R	25	29	11	20	8
S	19	14	945	15	2
T	16	10	18	9	6

The smallest numbers in rows 1, 2, 3, 4 and 5 are 3, 0, 8, 2 and 6. We reduce rows first by subtracting these numbers from each element in the row. The table becomes:

	1	**2**	**3**	**4**	**5**
P	27	10	14	0	942
Q	12	18	26	9	0
R	17	21	3	12	0
S	17	12	943	13	0
T	10	4	12	3	0

5 a (continued)

The smallest numbers in columns 1, 2, 3, 4 and 5 are 10, 4, 3, 0 and 0. We reduce columns by subtracting these numbers from each element in the column. Therefore, the reduced cost matrix is:

	1	**2**	**3**	**4**	**5**
P	17	6	11	0	942
Q	2	14	23	9	0
R	7	17	0	12	0
S	7	8	940	13	0
T	0	0	9	3	0

We now apply the Hungarian algorithm.
We can cover all the zeros in four lines, so the solution is not optimal:

	1	**2**	**3**	**4**	**5**
P	17	6	11	0	942
Q	2	14	23	9	0
R	7	17	0	12	0
S	7	8	940	13	0
T	0	0	9	3	0

	1	**2**	**3**	**4**	**5**
P	17	6	11	0	942
Q	2	14	23	9	0
R	7	17	0	12	0
S	7	8	940	13	0
T	0	0	9	3	0

The smallest uncovered element is 2, so we augment the matrix as follows:

- Add 2 to the elements covered by two lines circled in red
- Leave the elements covered by just one line unchanged
- Subtract 2 from the uncovered elements

This gives the following matrix:

	1	**2**	**3**	**4**	**5**
P	17	6	11	0	944
Q	0	12	21	7	0
R	7	17	0	12	2
S	5	6	938	11	0
T	0	0	9	3	2

We can cover all the zeros with five lines, so we have found our optimal solution of:

P – 4 (52)
Q – 1 (43)
R – 3 (44)
S – 5 (53)
T – 2 (45)

5 b Maximum profit = 52 + 43 + 44 + 53 + 45 = £237.

Challenge

a The initial cost matrix is shown below:

	Machine *A*	**Machine *B***	**Machine *C***	**Machine *D***
Worker 1	700	500	1800	150
Worker 2	600	450	1100	220
Worker 3	850	700	1300	300
Worker 4	500	450	1400	280
Worker 5	350	400	1000	200

If worker 2 must be trained to use machine *D* then we would reduce the cost for that cell (currently at £220) to 0 to make the allocation 'attractive'. Or we could delete the row for worker 2 and the column for machine *D*.

b In order to use the Hungarian algorithm to solve this problem the introduction of a dummy 5th machine (*E*) is required to make the matrix *n* by *n* as follows. If worker 2 must be assigned to use one of the machines (*A* to *D*) then we assign a cost of 10,000 for worker 2/Machine *E* to make the allocation 'unattractive'.

	Machine *A*	**Machine *B***	**Machine *C***	**Machine *D***	**Machine *E***
Worker 1	700	500	1800	150	0
Worker 2	600	450	1100	220	10000
Worker 3	850	700	1300	300	0
Worker 4	500	450	1400	280	0
Worker 5	350	400	1000	200	0

c Machine *D* should have two workers trained to it as all column *D*'s elements are smaller than all the other elements in the table. To allocate two workers, replicate Machine *D* as Machine *E*. To allocate two workers to machine *D* we replicate the costs associated with machine *D* for machine *E* as follows:

	Machine *A*	**Machine *B***	**Machine *C***	**Machine *D***	**Machine *E***
Worker 1	700	500	1800	150	150
Worker 2	600	450	1100	220	220
Worker 3	850	700	1300	300	300
Worker 4	500	450	1400	280	280
Worker 5	350	400	1000	200	200