THE ASSIGNMENT PROBLEM

In many business situations, management finds it necessary to assign personnel to jobs, jobs to machines, machines to job locations within a plant, or salespersons to territories within the distribution area of the business. In each of these cases, management would like to make the most effective or cost-efficient assignment of a set of *workers*(or objects) to a set of *jobs* (or assignments) .

Typically, we have a group of n "**applicants**" applying for n **"jobs**", and the non-negative cost c_{ij} of assigning the i th applicant to j th job is known. The objective is to assign one job to each applicant in such a way as to achieve the minimum possible total cost.

Define binary variables x_{ij} with value of either 0 or 1.

When x_{ij} =1, it indicates that we should assign applicant i to job j. Otherwise (x_{ij} = 0), we should not assign applicant i to job j.

A special case of the transportation problem is the assignment problem, which occurs when each supply is 1 and each demand is 1 and m equals to n. In this case, the integrality implies that every supplier will be assigned one destination and every destination will have one supplier. The costs give the charge for assigning a supplier and destination to each other.

The criteria used to measure the effectiveness of a particular set of assignments may be

- Total cost,
- Total profit, or
- Total time

to perform a set of operations.

General linear model for balanced transportation problem

$$\min \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$s.t. \sum_{j=1}^{n} x_{ij} = s_i \, (i=1,2,\ldots m)$$

$$\sum_{i=1}^{m} x_{ij} = d_j \, (j=1,2,\ldots,n)$$

$$x_{ij} \geq 0 (i=1,\ldots,m;\ j=1,\ldots,n)$$

General linear model for assignment problem

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$s.t. \sum_{j=1}^{n} x_{ij} = 1 (i=1,2,\ldots n)$$

$$\sum_{i=1}^{n} x_{ij} = 1 (j=1,2,\ldots,n)$$

$$x_{ij} = 0,1 (i=1,\ldots,n;\ j=1,\ldots,n)$$

THE HUNGARIAN ALGORITHM

The following description of the Hungarian algorithm assumes that:

1. There is a cost assignment matrix for n "**applicants**" to be assigned to n **"jobs**". If necessary we add dummy rows or columns consisting of all 0's so that the numbers of applicant and jobs are the same.)
2. All costs are nonnegative.
3. The problem is a minimization problem.

The Hungarian Algorithm

Initialization

1. For each row, subtract the minimum number from all numbers in that row.(Row reduction)
2. In the resulting matrix, subtract the minimum number in each column from all numbers in the column.(Column reduction)

Iterative Steps

1. Make as many 0 cost assignments as possible. If all applicants are assigned, STOP; this is the minimum cost assignment. Otherwise draw the minimum number of horizontal and vertical lines to cover all 0'a in the matrix.
2. Find he smallest value not covered by the lines; this number is the reduction value.
3. Subtract the reduction value from all numbers not covered by any lines. Add he reduction value to any number covered by both a horizontal and vertical line. GO TO STEP 1.

Drawing the Minimum Number of Lines to Cover All 0's

1. ***Mark all rows with no assignments (with a "!").***
2. ***For each row just marked, mark each column that has a 0 in that row (with a "!").***
3. ***For each column just marked, mark each row that has an assignment in that column (with a "!").***
4. ***Repeat steps 2 and 3 until no more marks can be made.***
5. ***Draw lines through unmarked rows and marked columns.***

Example:

	A	B	C	D	E
1	10	4	6	10	12
2	11	7	7	9	14
3	13	8	12	14	15
4	14	16	13	17	17
5	17	11	17	20	19

$\downarrow$

Row Subtraction (Row Reduction)

	A	B	C	D	E
1	6	0	2	6	8
2	4	0	0	2	7
3	5	0	4	6	7
4	1	3	0	4	4
5	6	0	6	9	8

$\downarrow$

Column Subtraction (Column Reduction)

	A	B*	C	D	E
*1	5	**0**	2	4	4
2	3	0	**0**	0	3
*3	4	0	4	4	3
4	**0**	3	0	2	0
*5	5	0	6	7	4

Minimum Uncovered Value (2)

	A	B*	C	D	E
1	3	0	**0**	2	2
2	3	2	0	**0**	3
*3	2	**0**	2	2	1
4	**0**	5	0	2	0
*5	3	0	4	5	2

Minimum uncovered value

	A	B	C	D	E
1	3	1	**0**	2	2
2	3	3	0	**0**	3
3	1	0	1	1	**0**
4	**0**	6	0	2	0
5	2	**0**	3	4	1

Optimal Solution

1→C	6
2→D	9
3→E	15
4→A	14
5→B	11

Total 65

Check total reduction value

Example :

	A	B	C	D	E
1	6	2	7	4	8
2	8	5	4	3	1
3	10	6	4	6	2
4	6	8	6	5	3
5	7	5	5	1	4

$\downarrow$

Row Subtraction (Row Reduction)

	A	B	C	D	E
1	4	0	5	2	6
2	7	4	3	2	0
3	8	4	2	4	0
4	3	5	3	2	0
5	6	4	4	0	3

$\downarrow$

Column Subtraction (Column Reduction)

	A	B	C	D	E
1	1	**0**	3	2	6
2	4	4	1	2	**0**
3	5	4	**0**	4	0
4	**0**	5	1	2	0
5	3	4	2	**0**	3

Optimal Solution

1→B	2
2→E	1
3→C	4
4→A	6
5→D	1
Total	**14**

Check total reduction value

Example:

	A	B	C	D	E
1	10	5	9	18	11
2	13	19	6	12	14
3	3	2	4	4	5
4	18	9	12	17	15
5	11	6	14	19	7

$\downarrow$

Row Subtraction (Row Reduction)

	A	B	C	D	E
1	5	0	4	13	6
2	7	13	0	6	8
3	1	0	2	2	3
4	9	0	3	8	6
5	5	0	8	13	1

$\downarrow$

Column Subtraction (Column Reduction)

	A	B*	C	D	E
*1	4	**0**	4	11	5
2	6	13	**0**	4	7
3	0	0	2	**0**	2
*4	8	0	3	6	5
5	4	0	8	11	**0**

Minimum uncovered value

	A	B*	C*	D	E
*1	1	**0**	1	8	2
*2	6	16	**0**	4	7
3	0	3	2	**0**	2
*4	5	0	0	3	2
5	4	3	8	11	**0**

Minimum uncovered value

	A	B	C	D	E
1	**0**	0	1	7	1
2	5	16	**0**	3	6
3	0	4	3	**0**	2
4	5	**0**	0	2	1
5	4	4	9	11	**0**

Original Data

	A	B	C	D	E
1	**10**	5	9	18	11
2	13	19	**6**	12	14
3	3	2	4	**4**	5
4	18	**9**	12	17	15
5	11	6	14	19	**7**

Optimal Solution

1→A	10
2→C	6
3→D	4
4→B	9
5→E	7
Total	**37**

Check total reduction value

Unbalanced Problem and Degeneration

	A	B	C	D
1	10	5	9	18
2	13	19	6	12
3	3	2	4	4
4	18	9	12	17
5	11	6	14	19

We add a dummy column with 0 cost

	A	B	C	D	E
1	10	5	9	18	0
2	13	19	6	12	0
3	3	2	4	4	0
4	18	9	12	17	0
5	11	6	14	19	0

$\downarrow$

Row Subtraction does not required
Column Subtraction (Column Reduction)

	A	B	C	D	E
1	7	3	5	14	0
2	10	17	2	8	0
3	0	0	0	0	0
4	15	7	8	13	0
5	8	4	10	15	0

	A	B	C	D	E
1	7	3	5	14	0
2	10	17	2	8	0
3	0	0	0	0	0
4	15	7	8	13	0
5	8	4	10	15	0

Minimum uncovered value (2)

	A	B	C	D	E*
1	5	1	3	12	**0**
2	8	15	**0**	6	0
3	**0**	0	0	0	2
4*	13	5	6	11	0
5*	6	2	8	13	0

	A	B	C	D	E*
1*	5	1	3	12	**0**
2	8	15	**0**	6	0
3	**0**	0	0	0	2
4*	13	5	6	11	0
5*	6	2	8	13	0

Minimum uncovered value (1)

	A	B	C	D	E*
1	4	**0**	2	11	0
2	8	15	**0**	6	1
3	0	0	0	**0**	3
4*	12	4	5	10	0
5*	5	1	7	12	**0**

	A	B	C	D	E*
1	4	0	2	11	0
2	8	15	0	6	1
3	0	0	0	0	3
4*	12	4	5	10	0
5*	5	1	7	12	0

Minimum uncovered value (1)

	A	B*	C	D	E*
1*	4	**0**	2	11	0
2	8	15	**0**	6	2
3	**0**	0	0	0	4
4*	11	3	4	9	**0**
5*	4	0	6	11	0

	A	B*	C	D	E*
1*	4	**0**	2	11	0
2	8	15	**0**	6	2
3	**0**	0	0	0	4
4*	11	3	4	9	**0**
5*	4	0	6	11	0

Minimum uncovered value (2)

	A	B*	C	D	E*
1*	2	**0**	0	9	1
2	8	17	**0**	6	1
3	**0**	2	0	0	6
4*	9	3	2	7	**0**
5*	2	0	4	9	0

	A	B*	C	D	E*
1*	2	**0**	0	9	1
2	8	17	**0**	6	1
3	**0**	2	0	0	6
4*	9	3	2	7	**0**
5*	2	0	4	9	0

DEGENARATION (Uncovered 0 value)
(Table cannot be updated)

New Assignment Plan (by columns)

start here

	A*	B	C	D*	E
1	2	**0**	0	9	1
2	8	17	**0**	6	1
3*	**0**	2	0	0	6
4	9	3	2	7	**0**
5	2	0	4	9	0

	A*	B	C	D*	E
1	2	**0**	0	9	1
2	8	17	**0**	6	1
3*	**0**	2	0	0	6
4	9	3	2	7	**0**
5	2	0	4	9	0

No degeneration

Minimum uncovered value (2)

	A	B	C	D	E
1	**0**	0	0	7	1
2	6	17	**0**	4	4
3	0	4	2	**0**	8
4	7	3	2	5	**0**
5	0	**0**	4	7	0

Solution found

	A	B	C	D	E
1	**0**	0	0	7	1
2	6	17	**0**	4	4
3	0	4	2	**0**	8
4	7	3	2	5	**0**
5	0	**0**	4	7	0

Alternative solution

	A	B	C	D	E
1	0	**0**	0	7	1
2	6	17	**0**	4	4
3	0	4	2	**0**	8
4	7	3	2	5	**0**
5	**0**	0	4	7	0

Original Table Solution

	A	B	C	D
1	10	5	9	18
2	13	19	**6**	12
3	3	2	4	**4**
4	18	9	12	17
5	11	**6**	14	19

T_{min}=26

	A	B	C	D
1	10	5	9	18
2	13	19	**6**	12
3	3	2	4	**4**
4	18	9	12	17
5	**11**	6	14	19

Alternative solution
T_{min}=26