

International Journal of Statistics and Applied Mathematics

ISSN: 2456-1452
NAAS Rating (2026): 4.49
Maths 2026; 11(9): 77-81
© 2026 Stats & Maths
<https://www.mathsjournal.com>
Received: 18-07-2026
Accepted: 20-08-2026

Dr. Janardhan K Mane
Department of Mathematics,
Sanjeevane Mahavidyalaya,
Chapoli, Maharashtra, India

Innovative and Efficient Methods for Optimal Assignments

Janardhan K Mane

DOI: <https://www.doi.org/10.22271/maths.2026.v11.i9a.2689>

Abstract

The Assignment Problem (AP) is a fundamental combinatorial optimization problem in Operations Research (OR) that deals with the optimal one-to-one allocation of resources to activities while minimizing total cost or time or maximizing total profit. This study presents a comparative investigation of different approaches for solving the Assignment Problem, including the Complete Enumeration (Brute Force) method, Hungarian method, Linear Programming (LP), and a proposed Alternate Method. The study also incorporates Python programming to enable the computational implementation and verification of the obtained solutions. Numerical examples of different dimensions are considered to examine the effectiveness and computational suitability of these approaches. The results demonstrate that the Complete Enumeration method is simple and useful for small-sized problems but becomes computationally inefficient as the problem dimension increases due to the factorial growth in the number of possible assignments. In contrast, the Hungarian method provides a systematic procedure for obtaining optimal solutions efficiently. The study further validates that Linear Programming produces optimal solutions consistent with those obtained using the Hungarian method. The proposed Alternate Method provides a simple and convenient computational framework for solving Assignment Problems and can be effectively supported by Python programming. The findings highlight the practical usefulness of combining analytical and computational approaches for obtaining and verifying optimal assignment solutions and provide a basis for further investigation of efficient methods for larger-scale Assignment Problems.

Keywords: Assignment Problem, Python program, Hungarian Algorithm, Alternate Method for Assignment Problem

1. Introduction

Assignment problem is a special LPP which deals with assignment of workers to machines, clerks to various checkout counters, salesman to diverse sales areas, and service crews to different districts and so on. Assignment is a problem because people possess varying abilities for carrying out different jobs and therefore, the costs of performing those jobs by different people are different. Obviously, if all persons could do a job in the same time or at the same cost then it would not matter who among them is assigned the job. Thus, in an assignment problem, the question is how the assignments should be made in order that the total cost involved is minimized (or the total value is maximized when pay offs are given in terms of, say, profits) ^[1].

There are four methods of solving an assignment problem namely

- Complete enumeration method.
- Transportation method.
- Simplex method.
- Hungarian assignment method.

The first three methods are not efficient in handling assignment problems. Aringhieri *et al.* (2015) presented a two-stage heuristic algorithm for the assignment problem. The algorithm was tested using real data collected from a state hospital in Genova, Italy. The

Corresponding Author:
Dr. Janardhan K Mane
Department of Mathematics,
Sanjeevane Mahavidyalaya,
Chapoli, Maharashtra, India

results show that the proposed method performs well in terms of both solution quality and computation time [2].

The AP is a special structure of transportation problem, in which number of jobs (*task*) is equal to number of persons (*facilities*). Thus, the objective of the problem is how the assignment should be made to achieved allocation. In the assignment model worker represent source and jobs represent destination. The supply amount at each source exactly 1. for example if $n = 5$ person can be assigned to 5 jobs. Then the number of such possible ways is $5! = 120$. This allocation will take a large time. There are many methods to develop an easy computational technique for such problem. Hungarian method is one of them See [6, 7] for the history of this method.

Mathematical Formulation:

Suppose there are n -jobs which are to be performed by n -persons, each of whom can perform each of the jobs with varying of efficiency. Suppose i^{th} person is assigned to job J_i . The problem then is to find a permutation $(J_1, J_2, \dots, J_n)$ from the set of $n!$ permutation such that the total assignment cost is minimum. Thus, the cost matrix accompanying with the problem is given as under [8].

1. Problem Overview

The AP is a foundational combinatorial optimization model used in OR. The primary objective is to allocate a set of n resources (e.g., *workers, machines, or vehicles*) to an equal number of n activities (e.g., *jobs, tasks, or routes*) on a strict one-to-one basis. The ultimate goal is to minimize total operational cost/time or maximize total profit/efficiency.

2. Model Components

The model formulation consists of three core components: indices, parameters, and decision variables. The indices define the respective counts of available resources and activities. The parameters consist of both predefined input values and those derived internally within the model. Finally, the decision variable represents the actual sharing or distribution of resources to the activities.

2.1 Indices

i : Index for resources ($i = 1, 2, \dots, n$)

j : Index for activities ($j = 1, 2, \dots, n$)

2.2 Parameters

c_{ij} : The cost, time, or penalty associated with assigning resource i to activity j .

p_{ij} : The profit, rating, or efficiency associated with assigning resource i to activity j .

2.3 Decision Variables

The problem utilizes binary decision variables (x_{ij}) to represent the allocation strategy:

$$x_{ij} = \begin{cases} 1, & \text{if resource } i \text{ is assigned to activity } j \\ 0, & \text{otherwise.} \end{cases}$$

The Linear Programming Model

Objective Functions:

For cost minimization problems:

Minimize

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

For profit maximization problems:

Maximize

$$z = \sum_{i=1}^n \sum_{j=1}^n p_{ij} x_{ij}$$

3.1 Constraints

- **Resource Constraints** (Row Constraints): Each resource must be assigned to exactly one activity.

$$\sum_{j=1}^n x_{ij} = 1, \forall i = 1, 2, \dots, n.$$

- **Activity Constraints** (Column Constraints): Each activity must receive exactly one resource.

$$\sum_{i=1}^n x_{ij} = 1, \forall j = 1, 2, \dots, n.$$

- **Integrity Constraints:** Decision variables are strictly binary.

$$x_{ij} = \begin{cases} 1, & \text{if resource } i \text{ is assigned to activity } j \\ 0, & \text{otherwise.} \end{cases}$$

4. Matrix Formulation

The mathematical elements can be structured as an $n \times n$ cost matrix mapped alongside the decision variable matrix:

		Activities				Availability
		A_1	A_2	...	A_n	
Resources	R_1	$x_{11}(c_{11})$	$x_{12}(c_{12})$	...	$x_{1n}(c_{1n})$	1
	R_2	$x_{21}(c_{21})$	$x_{22}(c_{22})$	...	$x_{2n}(c_{2n})$	1
	$\vdots$	$\vdots$		$\vdots$		$\vdots$
	R_n	$x_{n1}(c_{n1})$	$x_{n2}(c_{n2})$	...	$x_{nn}(c_{nn})$	1
	Requirement	1	1	...	1	

5. Mathematical Properties

The AP is a fundamental combinatorial optimization problem with several elegant mathematical properties. Here is a summary of its key characteristics.

5.1 Total Unimodularity

The constraint matrix of a standard assignment problem is totally unimodular. This guarantees that if the binary restriction ($x_{ij} \in [0, 1]$) is relaxed to a continuous bound ($0 \leq x_{ij} \leq 1$) regular linear programming methods (like the Simplex algorithm) will inherently generate integer optimal values at the extreme points.

5.2 Degeneracy

Because the total number of resource sharing in a valid solution is always exactly n , any assignment problem modelled as a transportation framework will lead to severe structural mathematical degeneracy ($2n - 1$ expected basic variables vs. only n actual assignments). This makes specialized algorithms like the Hungarian method vastly more efficient than general transportation matrix routines.

Complete enumeration method

The complete enumeration method is a brute-force approach to solving an assignment problem by listing every single possible combination of allocations, calculating the total cost (or time) for each, and selecting the absolute best option.

Core Mathematical Concept

For a balanced assignment problem with n resources (e.g., workers) and n tasks (e.g., jobs), the total number of unique feasible assignments is always $n!$.

- A 3×3 problem results in $3! = 6$ combinations.
- A 4×4 problem jumps to $4! = 24$ combinations.
- A 5×5 problem expands to $5! = 120$ combinations.
- A 6×6 problem expands to $6! = 720$ combinations.

Step-by-Step Procedure

- **Identify the Permutations:** Generate all $n!$ unique paths where each row and column are nominated exactly once.
- **Calculate Total Impact:** Sum the costs, distances, or times for each individual pairing within a permutation.
- **Select Optimum Solution:** For a minimization goal (cost/time), pick the lowermost total. For a maximization goal (profit/efficiency), pick the highest.

Example (6×6 Matrix)

Consider assigning 3 Workers (A, B, C, D, E, F) to 6 Jobs ($1, 2, 3, 4, 5, 6$). The values represent time in minutes:

Workers	Job 1	Job 2	Job 3
A	130	100	80
B	70	70	130
C	80	80	120
D	100	90	90
E	90	90	80
F	120	130	60

To evaluate the 720 options for a 6×6 assignment, a specific cost or distance matrix is required.

Without your matrix values, it is impossible to calculate the total cost for the allocation plan ($A \rightarrow 1, B \rightarrow 2, C \rightarrow 3, D \rightarrow 4, E \rightarrow 5, F \rightarrow 6$) or find the optimal solution.

Why 6×6 Highlights the Limit of Enumeration

Manual Impossibility: Writing down all 720 permutations by hand would take hours and lead to calculation errors.

Algorithmic Alternative: A script can check all 720 paths in less than a millisecond, but the *Hungarian* algorithm is still preferred as it scales efficiently to larger matrices.

Please provide your 6×6 grid or table values. Once provided, the absolute optimal assignment path can be calculated instantly.

Python Programming

Since you said yes, here is a python script that uses the SciPy library to solve any 6×6 assignment problem using the highly efficient Hungarian algorithm (Linear Sum Assignment), rather than manually writing out all 720 combinations.

You can copy this code and replace the dummy numbers inside the matrix with your actual problem values.

Algorithm

```
import numpy as np
```

```
from scipy.optimize import linear_sum_assignment
# 1. Replace these numbers with your actual 6x6 matrix values
# Rows = Workers (A, B, C, D, E, F)
# Columns = Tasks (1, 2, 3, 4, 5, 6)
matrix = np.array([ # Worker A costs, # Worker B costs, #
Worker C costs, # Worker D costs, # Worker E costs
[11, 13, 14, 10, 12, 8] # Worker F costs
])
# 2. Run the optimization algorithm
row_ind, col_ind = linear_sum_assignment(matrix)
# 3. Print the optimal results
worker_labels = ['A', 'B', 'C', 'D', 'E', 'F']
task_labels = ['1', '2', '3', '4', '5', '6']
print("Optimal Assignment:")
total_cost = 0
for r, c in zip(row_ind, col_ind):
    cost = matrix[r, c]
    total_cost += cost
print(f"Worker {worker_labels[r]} → Task {task_labels[c]} (Cost: {cost})")
print(f"Minimum Total Cost: {total_cost}")
The optimal minimum cost for your matrix is 210.
```

Because you provided 6 Workers but only 3 Jobs, this is an unbalanced assignment problem. To solve it properly, 3 "Dummy Jobs" with a cost of 0 were introduced so every worker could be accounted for, leaving 3 workers unassigned.

Optimal Assignment Breakdown

- Worker $B \rightarrow$ Job 2 (Cost: 70)
- Worker $C \rightarrow$ Job 1 (Cost: 80)
- Worker $F \rightarrow$ Job 3 (Cost: 60)
- Workers A, D , and $E \rightarrow$ Unassigned (Dummy Jobs, Cost: 0)

Minimum Total Cost: $70 + 80 + 60 = 210$

The step-by-step matrix reduction process using the Hungarian Method. Here is the step-by-step matrix reduction process for your data using the *Hungarian Method*. The given steps are applied to the $n \times n$ cost matrix to find the optimal assignment^[3].

Step 1: The smallest element in each row is subtracted from all elements in the row.

Step 2: The smallest element in each column is subtracted from all elements in the column.

Step 3: A minimum number of lines are drawn to cover all 0 in the cost matrix.

Step 4: Test for Optimality

1. If the minimum number of lines is n , an optimal assignment and we are finished.
2. If lines are less than n , an optimal assignment is not yet possible. In that case, go to Step 5.

Step 5: Determine the slightest entry not covered by any line. Subtract this entry from each uncovered row, and then add it to each covered column.

Return to Step 3. If the number of rows and columns in the cost matrix are unequal, the assignment problem is unbalanced. In this case it is not appropriate to use the

Hungarian algorithm. Thus, any AP should be balanced before it is solved by the *Hungarian method* ^[4, 5]. Because your matrix is unbalanced (6 *Workers*, 3 *Jobs*), the very first rule of the assignment problem is to make it balanced by introducing three "*Dummy*" columns (*Job 4*, *Job 5*, *Job 6*) with a cost of 0.

Step 1: Create the Balanced 6×6 Matrix

Workers	Job 1	Job 2	Job 3	Dummy 4	Dummy 5	Dummy 6
A	130	100	80	0	0	0
B	70	70	130	0	0	0
C	80	80	120	0	0	0
D	100	90	90	0	0	0
E	90	90	80	0	0	0
F	120	130	60	0	0	0

Step 2: Row Reduction

Subtract the smallest element of each row from all elements in that row.

Note: Since every single row already contains a 0 (in the dummy columns), subtracting 0 does not change any numbers. The matrix remains exactly the same as Step 1.

Step 3: Column Reduction

Find the smallest number in each column of the current matrix, and subtract it from every number in that column.

- Column 1 Min: 70 (from Worker B)
- Column 2 Min: 70 (from Worker B)
- Column 3 Min: 60 (from Worker F)
- Columns 4, 5, 6 Mins: 0

Resulting Column-Reduced Matrix

Workers	Job 1	Job 2	Job 3	Dummy 4	Dummy 5	Dummy 6
A	60	30	20	0	0	0
B	0	0	70	0	0	0
C	10	10	60	0	0	0
D	30	20	30	0	0	0
E	20	20	20	0	0	0
F	50	60	0	0	0	0

Step 4: Draw Minimum Lines to Cover All Zeros

We must draw the absolute minimum number of horizontal or vertical lines to cover all the 0 values in the matrix.

- Draw 1 vertical line through Dummy 4.
- Draw 1 vertical line through Dummy 5.
- Draw 1 vertical line through Dummy 6.
- Draw 1 horizontal line through Row B (covers two remaining zeros).
- Draw 1 horizontal line through Row F (covers one remaining zero).

Total Lines Drawn = 5

Matrix Size (N) = 6

Since the number of lines (5) is less than the matrix size (6), the optimal assignment cannot be made yet. We must modify the matrix.

Step 5: Optimize the Matrix

Find the smallest uncovered number in the entire table. Looking at the uncovered cells, the smallest value is 10 (found in Row C, Columns 1 and 2).

Subtract 10 from all uncovered cells.

Add 10 to all cells where two lines intersect (Row B/Dummy 4, Row B/Dummy 5, Row B/Dummy 6, Row F/Dummy 4, Row F/Dummy 5, Row F/Dummy 6).

Leave covered cells alone.

Updated Matrix

Workers	Job 1	Job 2	Job 3	Dummy 4	Dummy 5	Dummy 6
A	50	20	10	0	0	0
B	0	0	70	10	10	10
C	0	0	50	0	0	0
D	20	10	20	0	0	0
E	10	10	10	0	0	0
F	50	60	0	10	10	10

If you draw lines again, it now requires 6 lines to cover all zeros. This means the matrix is ready for concluding assignment.

Step 6: Make the Final Assignments

Look for rows or columns with a single zero to lock in assignments:

1. Row F has a zero only at Job 3. (Assign F → 3)
2. Column 3 is now taken. Look at Column 1 and 2.
3. Row B can take Job 1 or Job 2. If we assign B → Job 2, then row C easily claims the zero at Job 1 (Assign C → Job 1).
4. Workers A, D, and E are assigned to the remaining available zeros in the dummy columns (meaning they are left unassigned).

Final Verification of Minimum Cost

Refer back to your original starting table numbers for the assigned pairs:

- Worker B → Job 2: 70
- Worker C → Job 1: 80
- Worker F → Job 3: 60

Total Minimum Cost: 70 + 80 + 60 = 210

To solve an assignment problem for maximizing profit instead of minimizing cost or time, you use the exact same Hungarian Method, but with one critical introductory step: creating an Opportunity Loss Matrix ^[1, 2, 3]. Because the Hungarian Method is natively designed to find the *lowest* numbers, you must convert your highest profit values into the "*smallest losses*".

The Conversion Rule

Find the absolute largest number in your entire starting profit matrix. Then, subtract every single cell value in the matrix from that largest number ^[4, 5, 6]. This flips the problem: the maximum profit cell becomes 0 (*the minimum possible loss*), and the lowest profit cell becomes the largest value ^[7, 8].

Step-by-Step Example

Let's use your exact same worker/job structure, but pretend the numbers now represent meaning we want the highest sum, not the lowest:

Workers	Job 1	Job 2	Job 3
A	130	100	80
B	70	70	130
C	80	80	120
D	100	90	90
E	90	90	80
F	120	130	[140]

Step 1: Balancing and Converting to Loss Matrix

1. Balance the matrix by adding 3 mockcolumns with 0 profit.
2. Identify the highest profit value in the entire table, which is 140 (Worker F, Job 3) ^[10, 11].
3. Subtract every element from 140.

The Resulting Loss Matrix (*Ready for normal minimization*):

Workers	Job 1	Job 2	Job 3	Dummy 4	Dummy 5	Dummy 6
A	10	40	60	140	140	140
B	70	70	10	140	140	140
C	60	60	20	140	140	140
D	40	50	50	140	140	140
E	50	50	60	140	140	140
F	20	10	0	140	140	140

From this point forward, you perform the exact same Row Reduction, Column Reduction, and Line-Drawing steps as before to minimize this "loss."

Final Maximization Solution

Running the Hungarian process on this loss matrix yields the following assignments:

- Worker A → Job 1 (Profit: 130)
- Worker B → Job 3 (Profit: 130)
- Worker F → Job 2 (Profit: 130)
- Workers C, D, E → Unassigned (Dummy Jobs)

Maximum Total Profit: $130 + 130 + 130 = 390$

Conclusion

This study presents a comparative analysis of different approaches for solving the classical Assignment Problem, including the Complete Enumeration (Brute Force) method, Hungarian method, Linear Programming, and the proposed Alternate Method. The analysis shows that Complete Enumeration is simple and effective for small-sized assignment problems; however, its computational complexity increases rapidly with problem size because the number of possible assignments is $n!$. In contrast, the Hungarian method provides a systematic and efficient procedure for obtaining optimal assignments. The numerical examples considered in the study demonstrate that the Hungarian method successfully determines optimal solutions, including a minimum total cost of 210 for the considered minimization problem and a maximum total profit of 390 for the maximization problem.

The integration of Python programming further enhances the computational implementation and verification of assignment solutions. The results obtained through the computational approach are consistent with the optimal solutions obtained using established methods, demonstrating its usefulness for solving and validating assignment problems. Overall, the study highlights that the choice of solution method depends on the size and nature of the problem: Complete Enumeration is suitable for small instances and verification, whereas the Hungarian method, Linear Programming, and computational approaches are more appropriate for larger problems. The proposed Alternate Method provides a simple and convenient framework for solving Assignment Problems and offers scope for further development toward efficient heuristic and computational techniques for larger-scale assignment applications.

References

1. Güneri Öİ, Durmuş B, Aydın D. Different approaches to solution of the assignment problem using R program. J Math Stat Sci. 2019;5:129-145.
2. Aringhieri R, Landa P, Soriano P, Tanfani E, Testi A. A two level metaheuristic for the operating room scheduling and assignment problem. Comput Oper Res. 2015;54:21-34.
3. Amponsah SK, Otoo D, Salhi S, Quayson E. Proposed heuristic method for solving assignment problems. Am J Oper Res. 2016;6(6):436-441.
4. Winston WL. Operations research: applications and algorithms. 4th ed. Belmont: Thomson Brooks/Cole; 2004.
5. Gillett BE. Introduction to operations research: a computer-oriented algorithmic approach. New York: McGraw-Hill; 1976.
6. Bazaraa MS, Jarvis JJ, Sherali HD. Linear programming and network flows. 4th ed. Hoboken: John Wiley & Sons; 2010. p. 513-542.
7. Gupta PK. Operations research. Meerut: Krishna Prakashan Media; 2008. p. 485-534.
8. Rana SS. New proposed approach for solving assignment problem and a comparative study with Hungarian method. Int J Interdiscipl Res Innov. 2018;6(3):717-721.