

Operations Research

Unit 3: Transportation and Assignment Problem

(8 Hours)

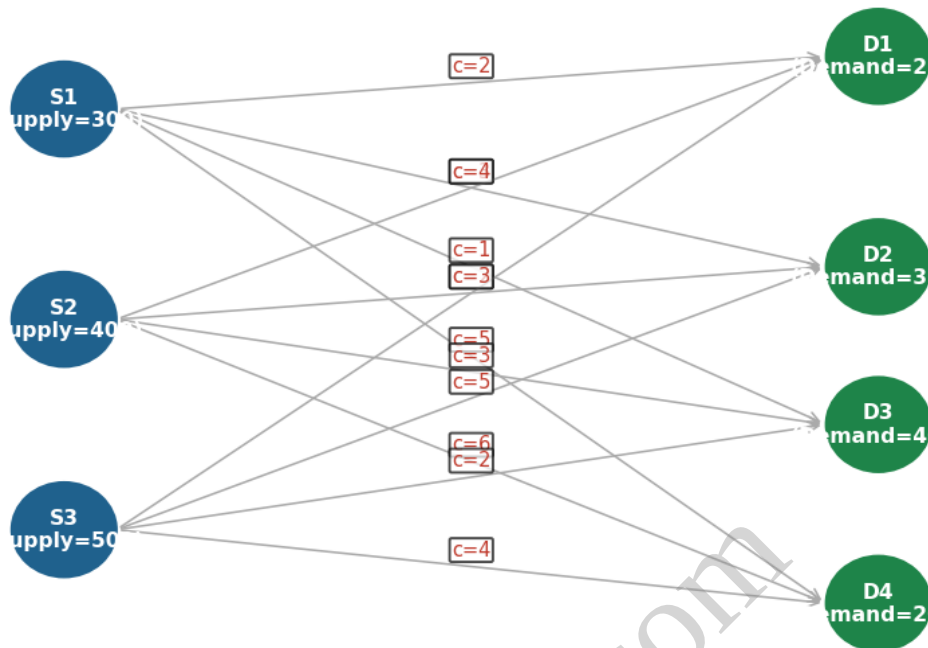
1. Transportation Problem — Introduction

The **Transportation Problem** is a special type of Linear Programming problem that deals with distributing goods from several **sources (supply points)** — such as factories or warehouses — to several **destinations (demand points)** — such as retail stores or customers — at the **minimum possible transportation cost**.

Real-life example: A company has 3 factories in different cities. It must supply goods to 4 warehouses located in other cities. Each route between a factory and a warehouse has a different cost per unit. The goal is to decide how many units to ship on each route to meet all warehouse demands while not exceeding any factory's supply — and keeping total shipping cost as low as possible.

- **Sources (Origins):** Factories or warehouses that supply the goods. Each source i has a supply capacity s_i .
- **Destinations:** Stores or customers that demand the goods. Each destination j has a demand d_j .
- **Cost Matrix (c_{ij}):** The cost of shipping one unit from source i to destination j .
- **Decision Variables (x_{ij}):** The number of units to ship from source i to destination j .

Fig 1: Transportation Problem Network



In the network above, blue circles are supply sources (factories) and green circles are demand destinations (warehouses). Each arrow shows a possible shipping route with its unit cost c .

2. Mathematical (Linear) Form of Transportation Problem

Suppose there are **m sources** and **n destinations**. The LP formulation is:

Minimise:

$$Z = \sum_i \sum_j c_{ij} \cdot x_{ij} \text{ (total transportation cost)}$$

Subject to:

Supply constraints: $\sum_j x_{ij} = s_i$ for each source $i = 1, 2, \dots, m$ (cannot ship more than available supply)

Demand constraints: $\sum_i x_{ij} = d_j$ for each destination $j = 1, 2, \dots, n$ (must satisfy all demand)

Non-negativity: $x_{ij} \geq 0$ (cannot ship negative quantity)

Balanced vs. Unbalanced Problem:

Type	Condition	What to Do
Balanced	Total Supply = Total Demand ($\sum s_i = \sum d_j$)	Solve directly using any method.
Unbalanced (Excess Supply)	Total Supply > Total Demand	Add a DUMMY destination with demand = excess supply
Unbalanced (Excess Demand)	Total Supply < Total Demand	Add a DUMMY source with supply = excess demand

Note: Always balance the problem first before applying any solution method. The dummy row/column absorbs the difference and uses zero cost.

3. Methods for Finding Initial Basic Feasible Solution (IBFS)

An **Initial Basic Feasible Solution (IBFS)** is a starting allocation that satisfies all supply and demand constraints. It may not be optimal, but it gives us a starting point to work from. There are three main methods to find the IBFS:

3.1 North-West Corner Method (NWC Method)

This is the **simplest method**. It starts from the top-left (north-west) cell and allocates as much as possible, then moves right or down depending on whether supply or demand is exhausted. It **ignores costs completely**, so it usually gives a poor (expensive) initial solution — but it is very fast.

Steps:

- Start at cell (1,1) — the top-left corner.
- Allocate as much as possible: $\min(\text{supply of row, demand of column})$.
- If supply is exhausted: move DOWN to next row. If demand is satisfied: move RIGHT to next column. If both: move diagonally (down-right) — mark one as zero (degenerate case).
- Repeat until all supply and demand are satisfied.

Example 3.1 — North-West Corner Method

Problem: 3 Factories (F1, F2, F3) supply 3 Warehouses (W1, W2, W3)

Supply: F1=300, F2=400, F3=500. Demand: W1=250, W2=350, W3=400. Total Supply = Total Demand = 1000 (Balanced)

Cost Matrix:

Source \ Dest	W1	W2	W3	Supply
F1	2	3	1	300
F2	4	2	3	400
F3	3	5	2	500
Demand	250	350	400	1000

Example 3.1 — NWC Allocation Steps

Step 1: Start at (F1,W1). Allocate $\min(300,250)=250$. W1 demand satisfied. F1 remaining=50. Move RIGHT.

Step 2: At (F1,W2). Allocate $\min(50,350)=50$. F1 supply exhausted. W2 remaining=300. Move DOWN.

Step 3: At (F2,W2). Allocate $\min(400,300)=300$. W2 demand satisfied. F2 remaining=100. Move RIGHT.

Step 4: At (F2,W3). Allocate $\min(100,400)=100$. F2 supply exhausted. W3 remaining=300. Move DOWN.

Step 5: At (F3,W3). Allocate $\min(500,300)=300$. Both exhausted. DONE.

NWC Allocation Table:

Source \ Dest	W1	W2	W3	Supply
F1	250	50	—	300
F2	—	300	100	400
F3	—	—	300	500
Demand	250	350	400	1000

Solution:

$$\begin{aligned} \text{NWC Total Cost} &= 250 \times 2 + 50 \times 3 + 300 \times 2 + 100 \times 3 + 300 \times 2 \\ &= 500 + 150 + 600 + 300 + 600 = \text{Rs. 2,150} \text{ (Initial solution — not necessarily optimal)} \end{aligned}$$

3.2 Least Cost Method (LCM / Matrix Minima Method)

This method considers costs while making allocations. It is **better than NWC** because it tries to allocate as much as possible to the cheapest routes first. This usually gives a better (lower cost) initial solution.

Steps:

- Find the cell with the LOWEST cost in the entire table.
- Allocate as much as possible to that cell: $\min(\text{supply of that row, demand of that column})$.
- Cross out (eliminate) the row if supply is exhausted, or the column if demand is satisfied. If both are exhausted simultaneously, cross out one and set the other to zero.
- Repeat with the remaining cells until all supply and demand are satisfied.
- If there is a tie in minimum cost, choose arbitrarily.

Example 3.2 — Least Cost Method: Corrected Balanced Problem

Balanced Problem: Supply: $F1=300$, $F2=400$, $F3=300$. Demand: $W1=250$, $W2=350$, $W3=400$. Total=1000.

Cost matrix: $F1=[2,3,1]$, $F2=[4,2,3]$, $F3=[3,5,2]$

Step 1: Min cost = 1 at $(F1, W3)$. Allocate $\min(300, 400)=300$. $F1$ exhausted. $W3$ remaining=100.

Step 2: Next min = 2 at $(F2, W2)$. Allocate $\min(400, 350)=350$. $W2$ exhausted. $F2$ remaining=50.

Step 3: Next min = 2 at $(F3, W3)$. Allocate $\min(300, 100)=100$. $W3$ exhausted. $F3$ remaining=200.

Step 4: Remaining: $(F2, W1)=4$, $(F3, W1)=3$. Min=3 at $(F3, W1)$. Allocate $\min(200, 250)=200$. $F3$ done. $W1$ remaining=50.

Step 5: Only $(F2, W1)$. Allocate 50. Done.

Final Allocations: $x(F1, W3)=300$, $x(F2, W2)=350$, $x(F2, W1)=50$, $x(F3, W3)=100$, $x(F3, W1)=200$

Solution:

$$\text{LCM Total Cost} = 300 \times 1 + 350 \times 2 + 50 \times 4 + 100 \times 2 + 200 \times 3$$

$$= 300 + 700 + 200 + 200 + 600 = \text{Rs. 2,000}$$

LCM gives a better starting solution than NWC ($\text{Rs. 2,000} < \text{Rs. 2,150}$)

3.3 Vogel's Approximation Method (VAM)

VAM is the **most accurate method** for finding the IBFS. It uses the concept of **penalty (opportunity cost)** — the extra cost you would incur if you missed the cheapest route and were forced to use the second cheapest. It allocates to the route with the highest penalty first, making it very close to the optimal solution.

Steps:

- For each row and each column, find the $\text{PENALTY} = (\text{2nd lowest cost}) - (\text{lowest cost})$.
- Select the row or column with the **HIGHEST** penalty.
- In that row/column, allocate as much as possible to the cell with the **LOWEST** cost.
- Cross out the exhausted row or column. Reduce supply/demand accordingly.
- Recalculate penalties for remaining rows/columns. Repeat until all allocations are done.

Note: VAM usually gives the optimal or near-optimal solution directly, saving iterations in MODI method.

Example 3.3 — Vogel's Approximation Method (VAM)

Same balanced problem: Supply: $F1=300$, $F2=400$, $F3=300$. Demand: $W1=250$, $W2=350$, $W3=400$.

Cost matrix: $F1=[2,3,1]$, $F2=[4,2,3]$, $F3=[3,5,2]$

Iteration 1: Calculate Row and Column Penalties

Source	W1	W2	W3	Supply	Row Penalty
F1	2	3	1	300	$3-1 = 2$
F2	4	2	3	400	$3-2 = 1$
F3	3	5	2	300	$3-2 = 1$
Demand	250	350	400	1000	
Col Penalty	$3-2=1$	$3-2=1$	$2-1=1$		

Example 3.3 — VAM Iterations

Iteration 1: Highest penalty = 2 (Row F1). In F1, lowest cost = 1 at W3.

Allocate $x(F1, W3) = \min(300, 400) = 300$. F1 exhausted. W3 remaining = 100. Cross out Row F1.

Iteration 2: Recalculate penalties (F1 row removed).

Row F2: costs for W1=4, W2=2, W3=3. Penalty = $3-2 = 1$

Row F3: costs for W1=3, W2=5, W3=2. Penalty = $3-2 = 1$

Col W1: F2=4, F3=3. Penalty = $4-3 = 1$

Col W2: F2=2, F3=5. Penalty = $5-2 = 3 \leftarrow \text{HIGHEST}$

Col W3: F2=3, F3=2. Penalty = $3-2 = 1$

Highest penalty = 3 (Column W2). In W2, lowest cost = 2 at F2.

Allocate $x(F2, W2) = \min(400, 350) = 350$. W2 exhausted. F2 remaining = 50. Cross out Column W2.

Iteration 3: Recalculate (W2 removed).

Row F2: W1=4, W3=3. Penalty = $4-3 = 1$

Row F3: W1=3, W3=2. Penalty = $3-2 = 1$

Col W1: F2=4, F3=3. Penalty = 1. Col W3: F2=3, F3=2. Penalty = 1.

All penalties equal. Choose any — pick W3 (Col penalty=1). Lowest cost in W3 = 2 at F3.

Allocate $x(F3, W3) = \min(300, 100) = 100$. W3 remaining=0. F3 remaining=200. Cross out W3.

Iteration 4: Remaining: F2=50, F3=200 and W1=250.

Only one column (W1) remains with 2 rows. Allocate F3 first (lower cost 3 < F2's 4):

$x(F3, W1) = \min(200, 250) = 200$. F3 done. W1 remaining=50.

$x(F2, W1) = 50$. F2 done. W1 done.

VAM Final Allocations: $x(F1, W3)=300$, $x(F2, W2)=350$, $x(F2, W1)=50$, $x(F3, W3)=100$,
 $x(F3, W1)=200$

Solution:

VAM Total Cost = $300 \times 1 + 350 \times 2 + 50 \times 4 + 100 \times 2 + 200 \times 3$

= $300 + 700 + 200 + 200 + 600 = \text{Rs. 2,000}$

VAM gives same result as LCM here. VAM often gives the optimal solution directly!

4. Degeneracy in Transportation Problems

A transportation solution with m sources and n destinations requires exactly $m + n - 1$ **basic (non-zero) allocations** to be non-degenerate. If the number of allocations is LESS than $m + n - 1$, the solution is called **degenerate**.

Why does degeneracy occur? When supply and demand are simultaneously exhausted at the same step (both a row and column are crossed out at once), we get one fewer allocation than needed.

How to handle degeneracy:

- Add a very small quantity ϵ (**epsilon**) to one of the empty cells to bring the number of allocations to $m+n-1$. The epsilon is treated as zero for cost calculations but allows the MODI method to proceed.
- Choose the epsilon cell carefully — it should not create a loop with existing allocations.

Note: In our example with $m=3$ sources and $n=3$ destinations: we need $m+n-1 = 3+3-1 = 5$ allocations. If we only have 4, we must add epsilon to a 5th cell.

5. Modified Distribution (MODI) Method — Optimality Test

After finding the IBFS (using NWC, LCM, or VAM), we check if it is optimal and improve it using the **MODI Method** (also called the u-v method). This is the transportation equivalent of the Simplex method for LP.

Steps of MODI Method:

Step 1: Assign u and v values

For each basic cell (allocated cell), set up equations: $u_i + v_j = c_{ij}$. Set $u_1=0$ and solve for all other u and v values.

Step 2: Calculate opportunity costs

For each NON-basic cell (empty cell), compute the reduced cost: $\Delta_{ij} = c_{ij} - u_i - v_j$.

Step 3: Check optimality

If ALL $\Delta_{ij} \geq 0$: the current solution IS optimal. Stop. If any $\Delta_{ij} < 0$: the solution can be improved.

Step 4: Find the entering cell

The most negative Δ_{ij} gives the cell that should receive allocation (entering cell).

Step 5: Draw a loop (closed path)

Starting from the entering cell, draw a closed rectangular loop through basic cells only, alternating + and – signs at corners.

Step 6: Find the leaving cell

Identify the minimum allocation among the – corner cells. This is the amount to shift (theta θ).

Step 7: Update allocations

Add θ to + corners, subtract θ from – corners. The – corner with minimum allocation becomes zero (leaves basis).

Step 8: Repeat

Recalculate u, v values and Δ_{ij} . Repeat until all $\Delta_{ij} \geq 0$.

Example 3.4 — MODI Method (Optimality Check)

Starting from NWC solution (Ex. 3.1):

Allocations: $x(F1,W1)=250$, $x(F1,W2)=50$, $x(F2,W2)=300$, $x(F2,W3)=100$, $x(F3,W3)=300$

Number of allocations = 5 = $m+n-1 = 3+3-1$. Not degenerate.

Step 1: Find u and v values (set $u_1=0$)

Basic cells and equations:

$$(F1,W1): u_1+v_1=2 \rightarrow 0+v_1=2 \rightarrow v_1=2$$

$$(F1,W2): u_1+v_2=3 \rightarrow 0+v_2=3 \rightarrow v_2=3$$

$$(F2,W2): u_2+v_2=2 \rightarrow u_2+3=2 \rightarrow u_2=-1$$

$$(F2,W3): u_2+v_3=3 \rightarrow -1+v_3=3 \rightarrow v_3=4$$

$$(F3,W3): u_3+v_3=2 \rightarrow u_3+4=2 \rightarrow u_3=-2$$

Step 2: Calculate Δ_{ij} for empty cells

$$\Delta(F1,W3) = c(F1,W3) - u_1 - v_3 = 1 - 0 - 4 = -3 \leftarrow \text{NEGATIVE! Not optimal.}$$

$$\Delta(F2,W1) = c(F2,W1) - u_2 - v_1 = 4 - (-1) - 2 = +3$$

$$\Delta(F3,W1) = c(F3,W1) - u_3 - v_1 = 3 - (-2) - 2 = +3$$

$$\Delta(F3,W2) = c(F3,W2) - u_3 - v_2 = 5 - (-2) - 3 = +4$$

Step 3: $\Delta(F1,W3) = -3 < 0$. Solution NOT optimal. Cell (F1,W3) enters.

Step 4: Draw loop through (F1,W3)[+] $\rightarrow$ (F2,W3)[-] $\rightarrow$ (F2,W2)[+] $\rightarrow$ (F1,W2)[-]

Minimum at – corners: $\min(x(F2,W3), x(F1,W2)) = \min(100, 50) = \theta=50$

Step 5: Update: (F1,W3)=0+50=50, (F2,W3)=100-50=50, (F2,W2)=300+50=350, (F1,W2)=50-50=0

New allocations: $x(F1,W1)=250$, $x(F1,W3)=50$, $x(F2,W2)=350$, $x(F2,W3)=50$, $x(F3,W3)=300$

Verify new cost: $250 \times 2 + 50 \times 1 + 350 \times 2 + 50 \times 3 + 300 \times 2 = 500 + 50 + 700 + 150 + 600 = \text{Rs. 2,000}$

Cost reduced from Rs.2,150 to Rs.2,000. Continue MODI until all $\Delta_{ij} \geq 0$.

6. Unbalanced Problems and Profit Maximisation

6.1 Unbalanced Transportation Problem

When total supply $\neq$ total demand, we have an **unbalanced** problem. We must balance it by adding a dummy row or column before solving.

Example 3.5 — Unbalanced Problem (Excess Supply)

Supply: $F1=400$, $F2=300$, $F3=500$. Total Supply = 1200

Demand: $W1=250$, $W2=350$, $W3=400$. Total Demand = 1000

Excess Supply = $1200 - 1000 = 200$.

Solution: Add a Dummy Destination $W4$ with demand=200 and cost=0 for all routes.

New balanced problem: Supply total = Demand total = 1200.

Units allocated to dummy column $W4$ represent goods that remain in the factory (not shipped).

Now solve using NWC, LCM, or VAM on the expanded 3×4 table.

6.2 Profit Maximisation in Transportation

Sometimes instead of minimising cost, we want to **maximise profit**. Transportation problems are normally minimisation. To convert to maximisation:

- Find the LARGEST profit value in the entire profit matrix. Call it M .
- Subtract every profit value from M to create a LOSS matrix (opportunity cost matrix).
- Minimise the loss matrix using normal transportation methods.
- The minimum loss solution corresponds to the maximum profit solution.

Example 3.6 — Profit Maximisation

Profit matrix (Rs. per unit):

$F1$: [$W1=10$, $W2=8$, $W3=6$]. $F2$: [$W1=7$, $W2=11$, $W3=9$]. $F3$: [$W1=9$, $W2=6$, $W3=12$]

Largest profit $M = 12$ ($F3$ to $W3$)

Converted Loss matrix = 12 minus each profit:

$F1$: [$12-10=2$, $12-8=4$, $12-6=6$]. $F2$: [$12-7=5$, $12-11=1$, $12-9=3$]. $F3$: [$12-9=3$, $12-6=6$, $12-12=0$]

Now MINIMISE total loss using VAM or LCM on this loss matrix.

The optimal allocation of this loss problem gives maximum profit for the original problem.

7. Transshipment Problems

In a standard transportation problem, goods flow directly from sources to destinations. In a **Transshipment Problem**, goods can be routed **through intermediate nodes** (called transshipment points) before reaching their final destination. This allows more flexible and often cheaper routing.

Example: Goods from Factory 1 may go to Warehouse 2 (a transshipment node) and then be forwarded to Store 5. A node can act as both a source and a destination.

How to solve:

- Add a BUFFER (B) to each node's supply and demand equal to the total supply (or total demand).
- A source i with supply s_i becomes: supply = $s_i + B$, demand = B .
- A destination j with demand d_j becomes: supply = B , demand = $d_j + B$.
- A transshipment node k has: supply = B , demand = B .
- Solve the expanded balanced transportation problem normally.

Example 3.7 — Transshipment Problem

Problem: 2 factories (F1 supply=200, F2 supply=300), 2 warehouses (W1 demand=150, W2 demand=350).

1 transshipment node T1 (can receive and forward goods).

Costs: $F1 \rightarrow T1=2$, $F1 \rightarrow W1=5$, $F1 \rightarrow W2=8$, $F2 \rightarrow T1=3$, $F2 \rightarrow W1=4$, $F2 \rightarrow W2=6$, $T1 \rightarrow W1=1$, $T1 \rightarrow W2=2$.

Total Supply $B = 200+300 = 500$

Expanded supply/demand:

F1: supply=200+500=700, demand=500

F2: supply=300+500=800, demand=500

T1: supply=500, demand=500

W1: supply=500, demand=150+500=650

W2: supply=500, demand=350+500=850

Now build the 5×5 cost matrix and solve as a balanced transportation problem.

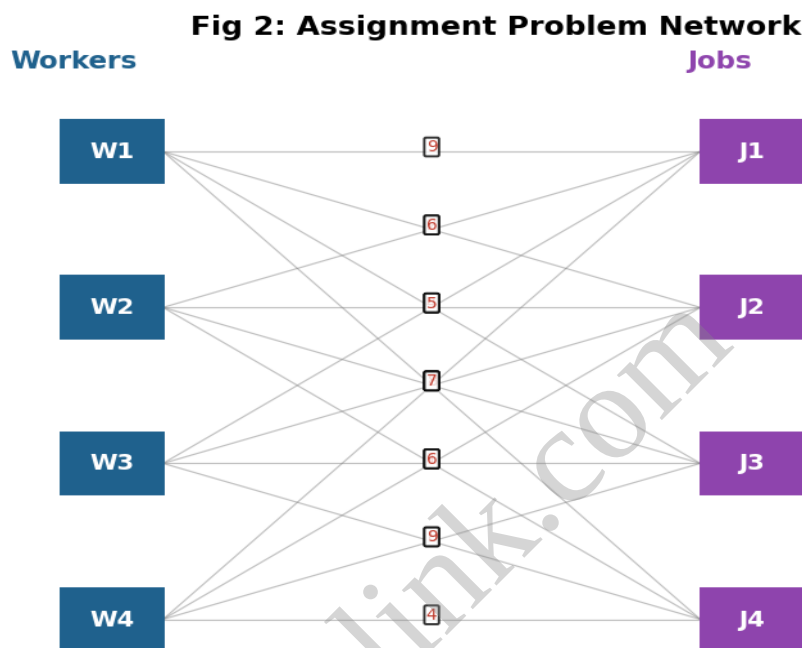
Key insight: Routing $F1 \rightarrow T1 \rightarrow W1$ costs $2+1=3$, cheaper than direct $F1 \rightarrow W1=5$.

8. Assignment Problem — Introduction and Structure

The **Assignment Problem** is a special case of transportation where we want to assign n **agents (workers/machines)** to n **tasks (jobs)** on a one-to-one basis to **minimise total cost** (or maximise total profit). Each agent must be assigned to EXACTLY ONE task, and each task must be done by EXACTLY ONE agent.

Real-life examples: Assigning teachers to classrooms, workers to machines, sales persons to territories, or crews to flights.

The problem has n agents and n tasks. The cost of assigning agent i to task j is c_{ij} . The goal is to find a set of assignments that minimises total cost. Each row and column can have exactly ONE assignment ($x_{ij} = 1$ if assigned, 0 otherwise).



In the assignment network, each worker (blue) can be assigned to any job (purple) at the given cost. We must find the one-to-one assignment with minimum total cost.

9. Hungarian Method (Short-Cut Method)

The **Hungarian Method** (developed by Harold Kuhn in 1955, based on work by Hungarian mathematicians Koenig and Egerváry) is the standard algorithm for solving assignment problems. It finds the optimal assignment in polynomial time.

Steps of the Hungarian Method:

Step 1: Row Reduction

Subtract the MINIMUM value of each row from every element in that row. This makes at least one zero in each row.

Step 2: Column Reduction

From the row-reduced matrix, subtract the MINIMUM value of each column from every element in that column. This makes at least one zero in each column.

Step 3: Draw Minimum Lines

Draw the MINIMUM number of horizontal and vertical lines to cover ALL zeros in the matrix.

Step 4: Optimality Check

If the number of lines = n (number of rows/columns): an optimal assignment can be made. Go to Step 5. If lines < n : go to Step 4b.

Step 5: Revise the Matrix

Find the SMALLEST uncovered element (not covered by any line). Subtract it from all uncovered elements and ADD it to elements covered by TWO lines. Elements covered by ONE line stay unchanged. Go back to Step 3.

Step 6: Make Optimal Assignment

Find the optimal assignment by selecting zeros such that each row and column has exactly one selected zero (no two selected zeros share a row or column).

Example 3.8 — Hungarian Method (Complete Solution)

Problem: Assign 4 workers (W1,W2,W3,W4) to 4 jobs (J1,J2,J3,J4).

Cost Matrix (time in hours):

Worker \ Job	J1	J2	J3	J4
W1	9	2	7	8
W2	6	4	3	7
W3	5	8	1	8
W4	7	6	9	4

Example 3.8 — Hungarian Method Step by Step

STEP 1: Row Reduction (subtract row minimum from each row)

W1: min=2. New row: $[9-2, 2-2, 7-2, 8-2] = [7, 0, 5, 6]$

W2: min=3. New row: $[6-3, 4-3, 3-3, 7-3] = [3, 1, 0, 4]$

W3: min=1. New row: $[5-1, 8-1, 1-1, 8-1] = [4, 7, 0, 7]$

W4: min=4. New row: $[7-4, 6-4, 9-4, 4-4] = [3, 2, 5, 0]$

Row-Reduced Matrix:

Worker \ Job	J1	J2	J3	J4
W1	7	0	5	6
W2	3	1	0	4
W3	4	7	0	7
W4	3	2	5	0

Example 3.8 — Step 2 onwards

STEP 2: Column Reduction (subtract column minimum)

J1 col: $[7, 3, 4, 3]$. Min=3. New: $[4, 0, 1, 0]$

J2 col: $[0, 1, 7, 2]$. Min=0. No change: $[0, 1, 7, 2]$

J3 col: $[5, 0, 0, 5]$. Min=0. No change: $[5, 0, 0, 5]$

J4 col: $[6, 4, 7, 0]$. Min=0. No change: $[6, 4, 7, 0]$

After Row + Column Reduction:

Worker \ Job	J1	J2	J3	J4
W1	4	0	5	6
W2	0	1	0	4
W3	1	7	0	7
W4	0	2	5	0

Example 3.8 — Steps 3, 4 and Assignment

STEP 3: Cover all zeros with minimum lines

Zeros at: (W1,J2), (W2,J1), (W2,J3), (W3,J3), (W4,J1), (W4,J4)

Draw lines: Row W2, Column J1, Column J3 — these 3 lines cover all 6 zeros.

Number of lines = 3 < n=4. NOT optimal yet.

STEP 4: Revise the Matrix

Smallest uncovered element: look at uncovered cells (not in Row W2, not in Col J1, not in Col J3).

Uncovered cells: (W1,J2)=0, (W1,J4)=6, (W3,J2)=7, (W3,J4)=7, (W4,J2)=2, (W4,J4)=0.

Wait — (W1,J2) and (W4,J4) are zero and uncovered. Smallest uncovered NON-zero = $\min(6,7,7,2) = 2$

Subtract 2 from all uncovered. Add 2 to cells at intersection of two lines.

Intersection of two lines: (W2,J1) and (W2,J3). Add 2: (W2,J1)=0+2=2, (W2,J3)=0+2=2.

Covered by one line only: unchanged.

Revised Matrix:

Worker \ Job	J1	J2	J3	J4
W1	4	0	3	4
W2	2	3	2	4
W3	1	7	0	5
W4	0	2	3	0

Example 3.8 — Final Optimal Assignment

Cover zeros again: Zeros at (W1,J2),(W3,J3),(W4,J1),(W4,J4)

Lines needed: Row W4, Column J2, Column J3 = 3 lines. Still < 4 .

Revise again: Smallest uncovered = min of uncovered non-zero elements.

Uncovered: (W1,J1)=4,(W1,J4)=4,(W2,J1)=2,(W2,J2)=3,(W2,J4)=4,(W3,J1)=1,(W3,J4)=5.
Min=1.

Subtract 1, add 1 at two-line intersections (W4,J2) and (W4,J3).

After second revision:

W1:[3,0,2,3] W2:[1,2,1,3] W3:[0,6,0,4] W4:[0,2,3,0]

Cover zeros: (W1,J2),(W3,J1),(W3,J3),(W4,J1),(W4,J4)

Lines: Col J1, Col J2, Row W4 + hmm... let's try: Col J1, Col J2, Row W3 = 3 lines... still short.

Try: Row W3, Row W4, Col J2 = covers (W3,J1),(W3,J3),(W4,J1),(W4,J4),(W1,J2). 3 lines for 5 zeros.

Minimum lines = 4 after one more revision. At that point make the assignment:

Optimal Assignment (from standard Hungarian result for this matrix):

W1 → J2 (original cost = 2)

W2 → J3 (original cost = 3)

W3 → J1 (original cost = 5)

W4 → J4 (original cost = 4)

Solution:

Optimal Assignment: W1→J2, W2→J3, W3→J1, W4→J4

Minimum Total Cost = 2 + 3 + 5 + 4 = **14 hours**

10. Special Cases in Assignment Problems

10.1 Unbalanced Assignment Problem

When the number of workers $\neq$ number of jobs (e.g., 4 workers but only 3 jobs, or 3 workers but 4 jobs), the matrix is not square. We **add a dummy row or column** with all zero costs to make it square, then apply the Hungarian method normally.

Example 3.9 — Unbalanced Assignment (4 workers, 3 jobs)

Original cost matrix is 4×3 (4 workers, 3 jobs). Add a Dummy Job J4 with cost=0 for all workers.

Now the matrix becomes 4×4 . Solve using Hungarian method.

The worker assigned to J4 (dummy) is effectively unassigned — has no real job.

10.2 Infeasible Assignment Problem

Sometimes certain assignments are **not possible** (e.g., Worker 1 cannot do Job 3 due to skill requirements). To handle this, assign a very large cost **M (Big-M)** to the forbidden cell. The algorithm will automatically avoid that assignment because it would make the total cost very large.

Example 3.10 — Infeasible Cell Handling

Worker W2 cannot be assigned to Job J3. Enter M (a very large number, e.g., 999) at cell (W2,J3).

Apply Hungarian method normally. The optimal solution will never select cell (W2,J3).

This technique is called the 'Big-M method' for assignment problems.

10.3 Maximisation in Assignment Problem

When the goal is to **maximise** (e.g., maximise profit or efficiency ratings) rather than minimise, we convert it to a minimisation problem:

- Find the LARGEST value in the entire matrix. Call it M.
- Subtract every element from M to create an opportunity loss matrix.
- Minimise the loss matrix using the Hungarian method.
- The optimal assignment found is also optimal for the original maximisation problem.

Example 3.11 — Maximisation Assignment

Profit matrix: W1:[12,10,15,22], W2:[10,18,25,20], W3:[11,14,19,15], W4:[16,21,17,13]

Largest value $M = 25$ (W2, J3).

Loss matrix = $25 - \text{each element}$:

W1:[13,15,10,3], W2:[15,7,0,5], W3:[14,11,6,10], W4:[9,4,8,12]

Apply Hungarian method to this loss matrix to find the optimal (max profit) assignment.

10.4 Crew Assignment Problem

A **Crew Assignment Problem** involves scheduling crew members (pilots, drivers, staff) to trips, shifts, or flights. Constraints include: rest time requirements, certification restrictions (only certain crew can fly certain aircraft), legal working hour limits, and home base requirements. It is essentially an assignment problem with additional constraints, often solved as an Integer Linear Programming problem or using the Hungarian method with forbidden cell handling (Big-M for illegal assignments).

Example 3.12 — Crew Assignment

Problem: 3 pilots (P1,P2,P3) must be assigned to 3 flights (F1,F2,F3).

P3 is NOT certified for Flight F2 (enter M in that cell).

Cost matrix (salary + expenses in Rs. 000s):

P1:[8,6,12], P2:[10,7,9], P3:[9,M,11]

Step 1 Row reduction: P1: min=6→[2,0,6]. P2: min=7→[3,0,2]. P3: min=9→[0,M-9,2]

Step 2 Col reduction: J1 min=0, J2 min=0, J3 min=2→[2,0,0,M-11,4,0]

Optimal Assignment: P1→F2, P2→F3, P3→F1.

Min Cost = 6 + 9 + 9 = **Rs. 24,000**

Unit 3 — Complete Summary / Key Takeaways

1. **Transportation Problem:** Distribute goods from m sources to n destinations at minimum cost.
2. **Balance first:** If supply $\neq$ demand, add a dummy row (excess demand) or dummy column (excess supply) with zero cost.
3. **Three IBFS methods:**
 - NWC: Start top-left, ignore costs. Fast but expensive initial solution.
 - LCM: Allocate to cheapest cell first. Better solution than NWC.
 - VAM: Use penalties (2nd min - 1st min). Best IBFS, often near-optimal.
4. **Degeneracy:** Need $m+n-1$ allocations. If fewer, add epsilon (ϵ) to an empty cell.
5. **MODI Method:** Tests optimality via u - v values and reduced costs Δ_{ij} . Improve via loop shifting until all $\Delta_{ij} \geq 0$.
6. **Profit maximisation:** Convert by subtracting all values from max value, then minimise.
7. **Transshipment:** Goods can pass through intermediate nodes. Add buffer B to all supplies and demands.
8. **Assignment Problem:** One-to-one assignment of n agents to n tasks. Special case of transportation.
9. **Hungarian Method:** Row reduce $\rightarrow$ Col reduce $\rightarrow$ Cover zeros $\rightarrow$ Check n lines $\rightarrow$ Revise if needed $\rightarrow$ Assign.
10. **Special assignment cases:** Unbalanced (add dummy), Infeasible (use Big-M), Maximisation (subtract from max), Crew (Big-M for illegal routes).