Assignment Problems and the Hungarian Method

Endre Boros
26:711:653: Discrete Optimization

Spring, 2019

Salesmen on the GO

- You work as a manager for a company, and currently have three of your salesmen meeting buyers at different cities: Austin, Boston and Chicago. You want them to meet new buyers tomorrow in Denver, Edmonton, and Fargo.
- These are the airfares between these cities:

From\To	Denver	Edmonton	Fargo
Austin	250	400	350
Boston	400	600	350
Chicago	200	400	250

- **Where should you send each of your salespeople in order to minimize airfare?**

Salesmen on the GO

- You work as a manager for a company, and currently have three of your salesmen meeting buyers at different cities: Austin, Boston and Chicago. You want them to meet new buyers tomorrow in Denver, Edmonton, and Fargo.

- These are the airfares between these cities:

From\To	Denver	Edmonton	Fargo
Austin	250	400	350
Boston	400	600	350
Chicago	200	400	250

- Where should you send each of your salespeople in order to minimize airfare?

Salesmen on the GO

- You work as a manager for a company, and currently have three of your salesmen meeting buyers at different cities: Austin, Boston and Chicago. You want them to meet new buyers tomorrow in Denver, Edmonton, and Fargo.

- These are the airfares between these cities:

From\To	Denver	Edmonton	Fargo
Austin	250	400	350
Boston	400	600	350
Chicago	200	400	250

- **Where should you send each of your salespeople in order to minimize airfare?**

Salesmen on the GO

- Here is one possible assignment:

From\To	Denver	Edmonton	Fargo
Austin	**250**	400	350
Boston	400	**600**	350
Chicago	200	400	**250**

- The cost of this assignment is

$$\$250 \quad + \$600 \quad + \$250 \quad = \quad \$1,100$$

Salesmen on the GO

- Here is one possible assignment:

From\To	Denver	Edmonton	Fargo
Austin	**250**	400	350
Boston	400	**600**	350
Chicago	200	400	**250**

- The cost of this assignment is

$$\$250 \quad + \$600 \quad + \$250 \quad = \quad \$1,100$$

Salesmen on the GO

- Here is another possible assignment:

From\To	Denver	Edmonton	Fargo
Austin	**250**	**400**	350
Boston	400	**600**	**350**
Chicago	**200**	400	**250**

- The cost of this assignment is

$$\begin{aligned} \$250 \quad &+ \$600 \quad + \$250 \quad = \quad \$1,100 \\ \$200 \quad &+ \$400 \quad + \$350 \quad = \quad \$950 \end{aligned}$$

- Brute force: $n!$ assignments!

$$\mathbf{n! >> 2^n >> n^2}$$

Salesmen on the GO

- Here is another possible assignment:

From\To	Denver	Edmonton	Fargo
Austin	**250**	**400**	350
Boston	400	**600**	**350**
Chicago	**200**	400	**250**

- The cost of this assignment is

$$\begin{array}{lllll} \$250 & + \$600 & + \$250 & = & \$1,100 \\ \$200 & + \$400 & + \$350 & = & \$950 \end{array}$$

- Brute force: $n!$ assignments!

$$\mathbf{n! >> 2^n >> n^2}$$

Salesmen on the GO

- Here is another possible assignment:

From\To	Denver	Edmonton	Fargo
Austin	**250**	**400**	350
Boston	400	**600**	**350**
Chicago	**200**	400	**250**

- The cost of this assignment is

$$\begin{array}{lllll} \$250 & + \$600 & + \$250 & = & \$1,100 \\ \$200 & + \$400 & + \$350 & = & \$950 \end{array}$$

- Brute force: $n!$ assignments!

$$\mathbf{n! >> 2^n >> n^2}$$

An Observation

- If a number a is added or subtracted from all entries in a row (or in a column) of the cost matrix, then all assignments change value by $\pm a$.
- **Consequently, these two cost matrices have the same optimal assignments.**

An Observation

- If a number a is added or subtracted from all entries in a row (or in a column) of the cost matrix, then all assignments change value by $\pm a$.
- **Consequently, these two cost matrices have the same optimal assignments.**

The Hungarian Method

- H. Kuhn (1955) created this algorithm using results from Hungarian mathematicians D. König (1916) and J. Egerváry (1931).
- This method reduces the assignment problem to a series of maximum matching (or equivalently, minimum vertex cover) problems in bipartite graphs – recall labeling algorithm.
- Munkres (1957) realized that this can be implemented in strongly polynomial $O(n^4)$ time.

The Hungarian Method

- H. Kuhn (1955) created this algorithm using results from Hungarian mathematicians D. König (1916) and J. Egerváry (1931).
- This method reduces the assignment problem to a series of maximum matching (or equivalently, minimum vertex cover) problems in bipartite graphs – recall labeling algorithm.
- Munkres (1957) realized that this can be implemented in strongly polynomial $O(n^4)$ time.

The Hungarian Method

- H. Kuhn (1955) created this algorithm using results from Hungarian mathematicians D. König (1916) and J. Egerváry (1931).
- This method reduces the assignment problem to a series of maximum matching (or equivalently, minimum vertex cover) problems in bipartite graphs – recall labeling algorithm.
- Munkres (1957) realized that this can be implemented in strongly polynomial $O(n^4)$ time.

The Hungarian Method

1 Input and Initialization: An $n \times n$ nonnegative cost matrix.

- Subtract the smallest entry in each row from all entries of its row.
- Subtract the smallest entry in each column from all entries of its column.

2 Find MinVertexCover = MaxMatching: Cover the zero entries of the cost matrix by the minimum number μ of lines (rows and/or columns). *In other words, solve a maximum matching/minimum vertex cover problem in the bipartite graph defined by the zeros of the cost matrix.*

3 Test of Optimality: If $\mu = n$, then **STOP**. There is an optimal assignment using the zero entries of the current matrix - the maximum cardinality/perfect matching found in the previous step.

4 Adjust: Let c be the smallest entry not covered by any of the chosen lines. ($c > 0$!) Subtract c from each entry of each uncovered row, and add c to each entry of each covered column. **RETURN to STEP 2**.

The Hungarian Method

1 Input and Initialization: An $n \times n$ nonnegative cost matrix.

- Subtract the smallest entry in each row from all entries of its row.
- Subtract the smallest entry in each column from all entries of its column.

2 Find MinVertexCover = MaxMatching: Cover the zero entries of the cost matrix by the minimum number μ of lines (rows and/or columns). *In other words, solve a maximum matching/minimum vertex cover problem in the bipartite graph defined by the zeros of the cost matrix.*

3 Test of Optimality: If $\mu = n$, then **STOP**. There is an optimal assignment using the zero entries of the current matrix - the maximum cardinality/perfect matching found in the previous step.

4 Adjust: Let c be the smallest entry not covered by any of the chosen lines. ($c > 0$!) Subtract c from each entry of each uncovered row, and add c to each entry of each covered column. **RETURN to STEP 2**.

The Hungarian Method

1 Input and Initialization: An $n \times n$ nonnegative cost matrix.

- Subtract the smallest entry in each row from all entries of its row.
- Subtract the smallest entry in each column from all entries of its column.

2 Find MinVertexCover = MaxMatching: Cover the zero entries of the cost matrix by the minimum number μ of lines (rows and/or columns). *In other words, solve a maximum matching/minimum vertex cover problem in the bipartite graph defined by the zeros of the cost matrix.*

3 Test of Optimality: If $\mu = n$, then **STOP**. There is an optimal assignment using the zero entries of the current matrix - the maximum cardinality/perfect matching found in the previous step.

4 Adjust: Let c be the smallest entry not covered by any of the chosen lines. ($c > 0$!) Subtract c from each entry of each uncovered row, and add c to each entry of each covered column. **RETURN to STEP 2**.

The Hungarian Method

1 Input and Initialization: An $n \times n$ nonnegative cost matrix.

- Subtract the smallest entry in each row from all entries of its row.
- Subtract the smallest entry in each column from all entries of its column.

2 Find MinVertexCover = MaxMatching: Cover the zero entries of the cost matrix by the minimum number μ of lines (rows and/or columns). *In other words, solve a maximum matching/minimum vertex cover problem in the bipartite graph defined by the zeros of the cost matrix.*

3 Test of Optimality: If $\mu = n$, then **STOP**. There is an optimal assignment using the zero entries of the current matrix - the maximum cardinality/perfect matching found in the previous step.

4 Adjust: Let c be the smallest entry not covered by any of the chosen lines. ($c > 0$!) Subtract c from each entry of each uncovered row, and add c to each entry of each covered column. **RETURN to STEP 2**.

The Hungarian Method

1 Input and Initialization: An $n \times n$ nonnegative cost matrix.

- Subtract the smallest entry in each row from all entries of its row.
- Subtract the smallest entry in each column from all entries of its column.

2 Find MinVertexCover = MaxMatching: Cover the zero entries of the cost matrix by the minimum number μ of lines (rows and/or columns). *In other words, solve a maximum matching/minimum vertex cover problem in the bipartite graph defined by the zeros of the cost matrix.*

3 Test of Optimality: If $\mu = n$, then **STOP**. There is an optimal assignment using the zero entries of the current matrix - the maximum cardinality/perfect matching found in the previous step.

4 Adjust: Let c be the smallest entry not covered by any of the chosen lines. ($c > 0$!) Subtract c from each entry of each uncovered row, and add c to each entry of each covered column. **RETURN to STEP 2**.

The Hungarian Method

1 Input and Initialization: An $n \times n$ nonnegative cost matrix.

- Subtract the smallest entry in each row from all entries of its row.
- Subtract the smallest entry in each column from all entries of its column.

2 Find MinVertexCover = MaxMatching: Cover the zero entries of the cost matrix by the minimum number μ of lines (rows and/or columns). *In other words, solve a maximum matching/minimum vertex cover problem in the bipartite graph defined by the zeros of the cost matrix.*

3 Test of Optimality: If $\mu = n$, then **STOP**. There is an optimal assignment using the zero entries of the current matrix - the maximum cardinality/perfect matching found in the previous step.

4 Adjust: Let c be the smallest entry not covered by any of the chosen lines. ($c > 0$!) Subtract c from each entry of each uncovered row, and add c to each entry of each covered column. **RETURN to STEP 2**.

Example 2

A 4×4 cost matrix:

$$\begin{bmatrix} 90 & 75 & 75 & 80 \\ 35 & 85 & 55 & 65 \\ 125 & 95 & 90 & 105 \\ 45 & 110 & 95 & 115 \end{bmatrix}$$

Example 2: Step 1

- Decrease by row minima

$$\begin{bmatrix} 90 & 75 & 75 & 80 \\ 35 & 85 & 55 & 65 \\ 125 & 95 & 90 & 105 \\ 45 & 110 & 95 & 115 \end{bmatrix} \longrightarrow \begin{bmatrix} 15 & 0 & 0 & 5 \\ 0 & 50 & 20 & 30 \\ 35 & 5 & 0 & 15 \\ 0 & 65 & 50 & 70 \end{bmatrix}$$

- Decrease by column minima

$$\begin{bmatrix} 15 & 0 & 0 & 5 \\ 0 & 50 & 20 & 30 \\ 35 & 5 & 0 & 15 \\ 0 & 65 & 50 & 70 \end{bmatrix} \longrightarrow \begin{bmatrix} 15 & 0 & 0 & 0 \\ 0 & 50 & 20 & 25 \\ 35 & 5 & 0 & 10 \\ 0 & 65 & 50 & 65 \end{bmatrix}$$

Example 2: Step 1

- Decrease by row minima

$$\begin{bmatrix} 90 & \color{red}{75} & 75 & 80 \\ \color{red}{35} & 85 & 55 & 65 \\ 125 & 95 & \color{red}{90} & 105 \\ \color{red}{45} & 110 & 95 & 115 \end{bmatrix} \longrightarrow \begin{bmatrix} 15 & 0 & 0 & 5 \\ 0 & 50 & 20 & 30 \\ 35 & 5 & 0 & 15 \\ 0 & 65 & 50 & 70 \end{bmatrix}$$

- Decrease by column minima

$$\begin{bmatrix} 15 & \color{blue}{0} & \color{blue}{0} & \color{blue}{5} \\ \color{blue}{0} & 50 & 20 & 30 \\ 35 & 5 & 0 & 15 \\ 0 & 65 & 50 & 70 \end{bmatrix} \longrightarrow \begin{bmatrix} 15 & 0 & 0 & 0 \\ 0 & 50 & 20 & 25 \\ 35 & 5 & 0 & 10 \\ 0 & 65 & 50 & 65 \end{bmatrix}$$

Example 2: Step 2

- Find minimum line-cover:

$$\begin{array}{cc} & \begin{matrix}\downarrow & & \downarrow & \end{matrix} \\ \begin{matrix}\rightarrow \\ \\ \\ \end{matrix} & \begin{bmatrix} 15 & 0 & 0 & 0 \\ 0 & 50 & 20 & 25 \\ 35 & 5 & 0 & 10 \\ 0 & 65 & 50 & 65 \end{bmatrix}\end{array}$$

- $\mu = 3 \; < \; n = 4$. GOTO STEP 3.

Example 2: Step 2

- Find minimum line-cover:

$$\begin{array}{cc} & \begin{matrix}\downarrow & & \downarrow & \end{matrix} \\ \begin{matrix}\rightarrow \\ \\ \\ \\ \end{matrix} & \begin{bmatrix}15 & 0 & 0 & 0\\ 0 & 50 & 20 & 25\\ 35 & 5 & 0 & 10\\ 0 & 65 & 50 & 65\end{bmatrix}\end{array}$$

- $\mu = 3 \; < \; n = 4$. GOTO STEP 3.

Example 2: Step 3

- Find minimum uncovered entry

$$\begin{array}{cc} & \begin{matrix} \downarrow & & \downarrow & \end{matrix} \\ \begin{matrix} \rightarrow \\ \\ \\ \\ \end{matrix} & \begin{bmatrix} 15 & 0 & 0 & 0 \\ 0 & 50 & 20 & 25 \\ 35 & 5 & 0 & 10 \\ 0 & 65 & 50 & 65 \end{bmatrix} \end{array}$$

- $c = 5$. Subtract c from uncovered rows and add it to covered columns:

$$\begin{array}{cc} & \begin{matrix} \downarrow & & \downarrow & \end{matrix} \\ \begin{matrix} \rightarrow \\ \\ \\ \\ \end{matrix} & \begin{bmatrix} 20 & 0 & 5 & 0 \\ 0 & 45 & 20 & 20 \\ 35 & 0 & 0 & 5 \\ 0 & 60 & 50 & 60 \end{bmatrix} \end{array}$$

- GOTO STEP 2.

Example 2: Step 3

- Find minimum uncovered entry

$$\begin{array}{cc} & \begin{array}{cccc} \downarrow & & \downarrow & \end{array} \\ \begin{array}{c} \rightarrow \\ \\ \\ \\ \end{array} & \begin{bmatrix} 15 & 0 & 0 & 0 \\ 0 & 50 & 20 & 25 \\ 35 & 5 & 0 & 10 \\ 0 & 65 & 50 & 65 \end{bmatrix} \end{array}$$

- $c = 5$. Subtract c from uncovered rows and add it to covered columns:

$$\begin{array}{cc} & \begin{array}{cccc} \downarrow & & \downarrow & \end{array} \\ \begin{array}{c} \rightarrow \\ \\ \\ \\ \end{array} & \begin{bmatrix} 20 & 0 & 5 & 0 \\ 0 & 45 & 20 & 20 \\ 35 & 0 & 0 & 5 \\ 0 & 60 & 50 & 60 \end{bmatrix} \end{array}$$

- GOTO STEP 2.

Example 2: Step 3

- Find minimum uncovered entry

$$\begin{array}{cc} & \begin{matrix} \downarrow & & \downarrow & \end{matrix} \\ \begin{matrix} \rightarrow \\ \\ \\ \\ \end{matrix} & \begin{bmatrix} 15 & 0 & 0 & 0 \\ 0 & 50 & 20 & 25 \\ 35 & 5 & 0 & 10 \\ 0 & 65 & 50 & 65 \end{bmatrix} \end{array}$$

- $c = 5$. Subtract c from uncovered rows and add it to covered columns:

$$\begin{array}{cc} & \begin{matrix} \downarrow & & \downarrow & \end{matrix} \\ \begin{matrix} \rightarrow \\ \\ \\ \\ \end{matrix} & \begin{bmatrix} 20 & 0 & 5 & 0 \\ 0 & 45 & 20 & 20 \\ 35 & 0 & 0 & 5 \\ 0 & 60 & 50 & 60 \end{bmatrix} \end{array}$$

- GOTO STEP 2.

Example 2: Step 2

- Find minimum line-cover:

$$\begin{array}{cc} & \begin{array}{cccc} \downarrow & & & \end{array} \\ \begin{array}{c} \rightarrow \\ \\ \rightarrow \\ \\ \end{array} & \begin{bmatrix} 20 & 0 & 5 & 0 \\ 0 & 45 & 20 & 20 \\ 35 & 0 & 0 & 5 \\ 0 & 60 & 50 & 60 \end{bmatrix} \end{array}$$

- $\mu = 3 \; < \; n = 4$. GOTO STEP 3.

Example 2: Step 2

- Find minimum line-cover:

$$\begin{array}{cc} & \begin{array}{cccc}\downarrow & & & \end{array} \\ \begin{array}{c}\rightarrow \\ \\ \rightarrow \\ \end{array} & \begin{bmatrix} 20 & 0 & 5 & 0 \\ 0 & 45 & 20 & 20 \\ 35 & 0 & 0 & 5 \\ 0 & 60 & 50 & 60 \end{bmatrix}\end{array}$$

- $\mu = 3 \; < \; n = 4$. GOTO STEP 3.

Example 2: Step 3

- Find minimum uncovered entry

$$\begin{array}{cc} & \begin{array}{cccc} \downarrow & & & \end{array} \\ \begin{array}{c} \rightarrow \\ \\ \rightarrow \\ \\ \end{array} & \begin{bmatrix} 20 & 0 & 5 & 0 \\ 0 & 45 & 20 & 20 \\ 35 & 0 & 0 & 5 \\ 0 & 60 & 50 & 60 \end{bmatrix} \end{array}$$

- $c = 20$. Subtract c from uncovered rows and add it to covered columns:

$$\begin{array}{cc} & \begin{array}{cccc} \downarrow & & & \end{array} \\ \begin{array}{c} \rightarrow \\ \\ \rightarrow \\ \\ \end{array} & \begin{bmatrix} 40 & 0 & 5 & 0 \\ 0 & 25 & 0 & 0 \\ 35 & 0 & 0 & 5 \\ 0 & 40 & 30 & 40 \end{bmatrix} \end{array}$$

- GOTO STEP 2.

Example 2: Step 3

- Find minimum uncovered entry

$$\begin{array}{c c} & \begin{matrix} \downarrow & & & \end{matrix} \\ \begin{matrix} \rightarrow \\ \\ \rightarrow \\ \\ \end{matrix} & \begin{bmatrix} 20 & 0 & 5 & 0 \\ 0 & 45 & 20 & 20 \\ 35 & 0 & 0 & 5 \\ 0 & 60 & 50 & 60 \end{bmatrix} \end{array}$$

- $c = 20$. Subtract c from uncovered rows and add it to covered columns:

$$\begin{array}{c c} & \begin{matrix} \downarrow & & & \end{matrix} \\ \begin{matrix} \rightarrow \\ \\ \rightarrow \\ \\ \end{matrix} & \begin{bmatrix} 40 & 0 & 5 & 0 \\ 0 & 25 & 0 & 0 \\ 35 & 0 & 0 & 5 \\ 0 & 40 & 30 & 40 \end{bmatrix} \end{array}$$

- GOTO STEP 2.

Example 2: Step 3

- Find minimum uncovered entry

$$\begin{array}{cc} & \begin{array}{cccc} \downarrow & & & \end{array} \\ \begin{array}{c} \rightarrow \\ \\ \rightarrow \\ \\ \end{array} & \begin{bmatrix} 20 & 0 & 5 & 0 \\ 0 & 45 & 20 & 20 \\ 35 & 0 & 0 & 5 \\ 0 & 60 & 50 & 60 \end{bmatrix} \end{array}$$

- $c = 20$. Subtract c from uncovered rows and add it to covered columns:

$$\begin{array}{cc} & \begin{array}{cccc} \downarrow & & & \end{array} \\ \begin{array}{c} \rightarrow \\ \\ \rightarrow \\ \\ \end{array} & \begin{bmatrix} 40 & 0 & 5 & 0 \\ 0 & 25 & 0 & 0 \\ 35 & 0 & 0 & 5 \\ 0 & 40 & 30 & 40 \end{bmatrix} \end{array}$$

- GOTO STEP 2.

Example 2: Step 2

- Find minimum line-cover:

$$\begin{array}{c} \rightarrow \\ \rightarrow \\ \rightarrow \\ \rightarrow \end{array} \begin{bmatrix} 40 & 0 & 5 & 0 \\ 0 & 25 & 0 & 0 \\ 35 & 0 & 0 & 5 \\ 0 & 40 & 30 & 40 \end{bmatrix}$$

- $\mu = 4 \;=\; n = 4$. STOP. We have an optimal matching using the zero entries:

$$\begin{bmatrix} 40 & 0 & 5 & \mathbf{0} \\ 0 & 25 & \mathbf{0} & 0 \\ 35 & \mathbf{0} & 0 & 5 \\ \mathbf{0} & 40 & 30 & 40 \end{bmatrix} \qquad \begin{bmatrix} 90 & 75 & 75 & \mathbf{80} \\ 35 & 85 & \mathbf{55} & 65 \\ 125 & \mathbf{95} & 90 & 105 \\ \mathbf{45} & 110 & 95 & 115 \end{bmatrix}$$

- Optimal assignment value $= 80 + 55 + 95 + 45 = 275$.

Example 2: Step 2

- Find minimum line-cover:

$$\begin{bmatrix} \to & 40 & 0 & 5 & 0 \\ \to & 0 & 25 & 0 & 0 \\ \to & 35 & 0 & 0 & 5 \\ \to & 0 & 40 & 30 & 40 \end{bmatrix}$$

- $\mu = 4 \ = \ n = 4$. STOP. We have an optimal matching using the zero entries:

$$\begin{bmatrix} 40 & 0 & 5 & \mathbf{0} \\ 0 & 25 & \mathbf{0} & 0 \\ 35 & \mathbf{0} & 0 & 5 \\ \mathbf{0} & 40 & 30 & 40 \end{bmatrix} \qquad \begin{bmatrix} 90 & 75 & 75 & \mathbf{80} \\ 35 & 85 & \mathbf{55} & 65 \\ 125 & \mathbf{95} & 90 & 105 \\ \mathbf{45} & 110 & 95 & 115 \end{bmatrix}$$

- Optimal assignment value $= 80 + 55 + 95 + 45 = 275$.

Example 2: Step 2

- Find minimum line-cover:

$$\begin{array}{c} \rightarrow \\ \rightarrow \\ \rightarrow \\ \rightarrow \end{array} \begin{bmatrix} 40 & 0 & 5 & 0 \\ 0 & 25 & 0 & 0 \\ 35 & 0 & 0 & 5 \\ 0 & 40 & 30 & 40 \end{bmatrix}$$

- $\mu = 4 \; = \; n = 4$. STOP. We have an optimal matching using the zero entries:

$$\begin{bmatrix} 40 & 0 & 5 & \mathbf{0} \\ 0 & 25 & \mathbf{0} & 0 \\ 35 & \mathbf{0} & 0 & 5 \\ \mathbf{0} & 40 & 30 & 40 \end{bmatrix} \qquad \begin{bmatrix} 90 & 75 & 75 & \mathbf{80} \\ 35 & 85 & \mathbf{55} & 65 \\ 125 & \mathbf{95} & 90 & 105 \\ \mathbf{45} & 110 & 95 & 115 \end{bmatrix}$$

- Optimal assignment value $= 80 + 55 + 95 + 45 = 275$.