ECE 307 - Techniques for Engineering Decisions

Hungarian Method

George Gross

Department of Electrical and Computer Engineering

University of Illinois at Urbana-Champaign

ASSIGNMENT PROBLEM

- We are given

  n machines $\quad M_1, M_2, \ldots, M_n \leftrightarrow i$

  n jobs $\quad J_1, J_2, \ldots, J_n \leftrightarrow j$

$$c_{ij} = \begin{cases} \text{cost of doing job } j \text{ on machine } i \\ Q \quad \text{if job } j \text{ cannot be done on machine } i \end{cases}$$

- Each machine can only do one job and each job requires one machine

ASSIGNMENT PROBLEM

- We wish to determine the optimal match, i.e., the assignment with the lowest total costs of doing the jobs on the n machines

- The brute force approach is simply enumeration: consider $n = 10$ and there are 3,628,800 possible choices!

SOLUTION APPROACH

- We can, however, introduce *categorical* decision variables

$$x_{ij} = \begin{cases} 1 & \text{job } j \text{ is assigned to machine } i \\ 0 & \text{otherwise} \end{cases}$$

- And the constraints can be stated as

$$\sum_{j=1}^{n} x_{ij} = 1 \quad \forall i \quad \text{each machine does exactly 1 job}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j \quad \text{each job is assigned to only 1 machine}$$

SOLUTION APPROACH

- The assignment problem, then, is formulated as

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

$$s.t.$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad \forall i$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j$$

$$x_{ij} \in \{ 0 , 1 \} \quad \forall i , j$$

- Thus, the assignment problem can be viewed as a special case of transportation problem

COST MATRIX

mach i \ *job j*	J_1	J_2	$\cdots$	J_n	*supplies*
M_1	x_{11} c_{11}	x_{12} c_{12}	$\cdots$	x_{1n} c_{1n}	1
M_2	x_{21} c_{21}	x_{22} c_{22}	$\cdots$	x_{2n} c_{2n}	1
$\vdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\vdots$
M_n	x_{n1} c_{n1}	x_{n2} c_{n2}	$\cdots$	x_{nn} c_{nn}	1
demands	1	1		1	

SIMPLIFIED COST MATRIX

- Since demands and supplies are 1 for all assignment problems, we represent the assignment problem by the cost matrix below

mach i \ *job j*	J_1	J_2	$\dots$	J_n
M_1	c_{11}	c_{12}	$\dots$	c_{1n}
M_2	c_{21}	c_{22}	$\dots$	c_{2n}
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
M_n	c_{n1}	c_{n2}	$\dots$	c_{n1}

HISTORY OF HUNGARIAN METHOD

- First published by Harold Kuhn in 1955

- Based on earlier works of two Hungarian

  mathematicians, Dénes König and Jenő Egerváry

FACT

$$\left.\begin{aligned} & \min Z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\, x_{ij} \\ s.t.\quad & \\ & \sum_{j=1}^{n} x_{ij} = 1 \quad \forall i \\ & \sum_{i=1}^{n} x_{ij} = 1 \quad \forall j \\ & x_{ij} \in \{0, 1\} \quad \forall i, j \end{aligned}\right\} (i)$$

$$\left.\begin{aligned} & \min \tilde{Z} = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}\, x_{ij} - k \\ s.t.\quad & \\ & \sum_{j=1}^{n} x_{ij} = 1 \quad \forall i \\ & \sum_{i=1}^{n} x_{ij} = 1 \quad \forall j \\ & x_{ij} \in \{0, 1\} \quad \forall i, j \end{aligned}\right\} (ii)$$

- **If** $x^*_{ij} \leq 1 \;\; for \;\; i, j \leq n$ **optimizes problem (i), then**

  $x^*_{ij} \leq 1 \;\; for \;\; i, j \leq n$ **also optimizes problem (ii)**

BASIC IDEA

- The optimal assignment is not affected by a constant added or subtracted from any row of the original assignment cost matrix by the fact in the previous slide and

$$\tilde{Z} = \sum_{j=1}^{n} \left(c_{qj} - k \right) x_{qj} + \sum_{\substack{i=1 \\ i \neq q}}^{n} \sum_{j=1}^{n} c_{ij} \, x_{ij}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} \, x_{ij} - k \sum_{j=1}^{n} x_{qj}$$

$$= Z - k$$

- A similar statement holds with respect to the column of the cost matrix

BASIC IDEA

- If all elements of the cost matrix are nonnegative, then the objective is nonnegative

- If the objective is nonnegative, and there exists a feasible solution such that the total cost is zero, then the feasible solution is the optimal solution

THE HUNGARIAN METHOD

- For each i , we consider the elements i and compute

$$\underline{c}_i = min\left(c_{ij} , 1 \leq j \leq n \right)$$

and subtract $\underline{c}_i$ from each element in row i to get

$$\tilde{c}_{ij} = c_{ij} - \underline{c}_i , \; 1 \leq j \leq n$$

- Then, we do the same procedure to each column
- We try to assign jobs only using the machines with zero costs since such an assignment, if found, is optimal

EXAMPLE 1

job j / *mach i*	J_1	J_2	J_3	J_4
M_1	**10**	**9**	**8**	**7**
M_2	**3**	**4**	**5**	**6**
M_3	**2**	**1**	**1**	**2**
M_4	**4**	**3**	**5**	**6**

EXAMPLE 1

job j / *mach i*	J_1	J_2	J_3	J_4
M_1	3	2	1	*0*
M_2	3	4	5	6
M_3	2	1	1	2
M_4	4	3	5	6

EXAMPLE 1

mach i \ *job j*	J_1	J_2	J_3	J_4
M_1	3	2	1	*0*
M_2	*0*	1	2	3
M_3	1	*0*	*0*	1
M_4	1	*0*	2	3

EXAMPLE 1

job j / *mach i*	J_1	J_2	J_3	J_4
M_1	**3**	**2**	**1**	*0*
M_2	*0*	**1**	**2**	**3**
M_3	**1**	*0*	*0*	**1**
M_4	**1**	*0*	**2**	**3**

HUNGARIAN METHOD

- In general, feasible assignment only using cells with zero costs may not exist after single row and column subtraction
- In such cases, we need to draw a minimum number of lines through certain rows and columns to cover all the cells with zero cost
- The minimum number of lines needed is the maximum number of jobs that can be assigned to the zero cells subject to all the constraints, a result that was proved by König

HUNGARIAN METHOD

- Then we look up the submatrix that is not covered by the lines to identify the smallest element

- Subtract from each element of the submatrix the value of the smallest element and add the value to all elements at the intersection of two lines

HUNGARIAN METHOD

- The rationale for this operation is that we subtract the smallest value from each element in a row including any element that is covered by a line; to compensate we also need to add an equal value to the element which is covered by the intersection of two lines and therefore the operation keeps the value of the elements not at an intersection unchanged

EXAMPLE 2

mach i \ job j	J_1	J_2	J_3	J_4
M_1	10	9	7	8
M_2	5	8	7	8
M_3	5	4	6	5
M_4	2	3	4	5

EXAMPLE 2

mach i \ *job j*	J_1	J_2	J_3	J_4
M_1	10	9	7	8
M_2	5	8	7	8
M_3	5	4	6	5
M_4	2	3	4	5

EXAMPLE 2

job j / *mach i*	J_1	J_2	J_3	J_4
M_1	3	2	*0*	1
M_2	*0*	3	2	3
M_3	1	*0*	2	1
M_4	*0*	1	2	3

EXAMPLE 2

job j / *mach i*	J_1	J_2	J_3	J_4
M_1	3	2	*0*	1
M_2	*0*	3	2	3
M_3	1	*0*	2	1
M_4	*0*	1	2	3

EXAMPLE 2

job j / *mach i*	J_1	J_2	J_3	J_4
M_1	3	2	*0*	*0*
M_2	*0*	3	2	2
M_3	1	*0*	2	*0*
M_4	*0*	1	2	2

EXAMPLE 2

job j / mach i	J_1	J_2	J_3	J_4
M_1	3	2	*0*	*0*
M_2	*0*	3	2	2
M_3	1	*0*	2	*0*
M_4	*0*	1	2	2

EXAMPLE 2

job j / mach i	J_1	J_2	J_3	J_4
M_1	4	2	*0*	*0*
M_2	*0*	2	1	1
M_3	2	*0*	2	*0*
M_4	*0*	*0*	1	1

EXAMPLE 2

	J_1	J_2	J_3	J_4
M_1	4	2	*0*	*0*
M_2	*0*	2	1	1
M_3	2	*0*	2	*0*
M_4	*0*	*0*	1	1

PROBLEM 3-13

- We cast the problem as an assignment with the days being the machines and the courses being the jobs

- In order for the assignment problem to be balanced, we introduce an additional course whose costs are zero for each day

PROBLEM 3-13

job j / *mach i*	J_1	J_2	J_3	J_4	J_5
M_1	50	40	60	20	*0*
M_2	40	30	40	30	*0*
M_3	60	20	30	20	*0*
M_4	30	30	20	30	*0*
M_5	10	20	10	30	*0*

PROBLEM 3-13

mach i \ *job j*	J_1	J_2	J_3	J_4	J_5
M_1	**40**	**20**	**50**	**0**	***0***
M_2	**30**	**10**	**30**	**10**	***0***
M_3	**50**	**0**	**20**	**0**	***0***
M_4	**20**	**10**	**10**	**10**	***0***
M_5	***0***	***0***	***0***	**10**	***0***

PROBLEM 3-13

job j / mach i	J_1	J_2	J_3	J_4	J_5
M_1	40	20	50	*0*	*0*
M_2	30	10	30	10	*0*
M_3	50	0	20	*0*	*0*
M_4	20	10	10	10	*0*
M_5	*0*	*0*	*0*	10	*0*

PROBLEM 3-13

	J_1	J_2	J_3	J_4	J_5
M_1	40	20	50	*0*	10
M_2	20	*0*	20	*0*	*0*
M_3	50	*0*	20	*0*	10
M_4	10	*0*	*0*	*0*	*0*
M_5	*0*	*0*	*0*	10	10

PROBLEM 3-13

	J_1	J_2	J_3	J_4	J_5
M_1	40	20	50	*0*	10
M_2	20	*0*	20	*0*	*0*
M_3	50	*0*	20	*0*	10
M_4	10	*0*	*0*	*0*	*0*
M_5	*0*	*0*	*0*	10	10

