Malaya Journal of Matematik, Vol. 6, No. 2, 321-325, 2018
https://doi.org/10.26637/MJM0602/0003

An amalgamated approach for solving unbalanced assignment problem

Anju Khandelwal[1]*

Abstract
Task assignment to any computing system is a most interesting and demandable research problem. Various methodologies and techniques are available in the literature to provide the solution of such problems. The assignment problems in distributed computing systems are one of major factor to determine the performance of such systems. The Assignment problem is one of the main problems while assigning jobs to the worker or machines to the worker to get an optimal solution. The problem mentioned in this paper suggests an exhaustive search approach to the unbalanced assignment problem through partial bound optimization. The bound is based on partial bound values. This paper contains the devised technique, computational algorithm of the approach and its implementation. This study is capable to deal all such real life situations where the Jobs (tasks) are more than the number of Machine (processors). The obtain solution is under consideration that all the jobs are allotted on the available machines in an optimum way. Finally numerical example and its algorithm have been given to show the efficiency of the proposed techniques.

Keywords
Unbalanced Assignment, Partial Bound Value, Hungarian Method, Distributed computing System

[1] *Department of Mathematics, Shri Ram Murti Smarak College of Engg. and Tech., Bareilly Affilated to Dr. APJ Abdul Kalam Technical University, Lucknow, India.*
***Corresponding author**: [1] dranju07khandelwal@gmail.com
Article History: Received **24** December **2017**; Accepted **09** March **2018**

Contents

1 **Introduction** 321
2 **Mathematical representation of assignment** 322
2.1 Standard Assignment Problem Formulation . . 322
3 **Logic for selection of partial bound value** 322
4 **Computational algorithm of proposed technique** 322
5 **Numerical implementation of proposed technique** 323
6 **Check proposed technique** 324
7 **Conclusion** 324
References 325

1. Introduction

An assignment problem is a particular case of transportation problem where the objective is to assign a number of resources to an equal number of activities so as to minimize total cost or maximize total profit of allocation. The problem of assignment arises because available resources such as men, machines, etc. have varying degrees of efficiency for performing different activities. Therefore, cost, profit or time of performing the different activities is different. Thus, the problem is how the assignments should be made so as to optimize the given objective. The assignment problem is one of the fundamental combinatorial optimization problems in the branch of optimization or operations research in Mathematics. It consists of finding a maximum weight matching in a weighted bipartite graph. In general, the assignment problem is of following type: There are a number of jobs and a number of machines. Any job can be assigned to perform any machine, incurring some cost that may vary depending on the job-machine assignment. In such problem, it is required to perform all jobs by assigning exactly one machine in such a way that the total cost of the assignment is minimized. If the number of jobs and machines are equal then the total cost of the assignment for all jobs is equal to the sum of the costs for each machine (or the sum of the costs for each Jobs, which is the same thing in this case), then the problem is called the linear assignment problem. Commonly, when speaking of the assignment problem without any additional qualification, then the linear assignment problem is meant. Otherwise, the problem is an unbalanced assignment. There are various ways to solve assignment problems. A well-known solution to the problem is the Hungarian method [1–3].

However, when we solve an unbalanced assignment problem, the Hungarian method requires adding dummy rows/columns to the machine-job assignment cost matrix so as the numbers of machines and jobs are equal. Hence, its space complexity is $O(n^2)$. To solve an unbalanced assignment problem [4, 13–15] proposes a new method with space complexity $O(nm)$. However, solution by mentioned does not always provide a minimal total cost. Therefore, this paper attempts to solve an unbalanced assignment problem by proposing a new method in order to improve assignment cost.

2. Mathematical representation of assignment

2.1 Standard Assignment Problem Formulation

Mathematically the Assignment problem can be expressed as follow:

$$Min.Z = \sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}x_{ij}$$

Subject to

$$\sum_{i=1}^{n} x_{ij} = 1, \quad 1,2,\cdots,n$$
$$\sum_{j=1}^{n} x_{ij} = 1, \quad 1,2,\cdots,n,$$
$$x_{ij} \in \{0,1\} \text{ for } i,j = 1,2,\cdots,n,$$

with an assumption that i^{th} job will be completely by j^{th} machine and

$$x_{ij} = \begin{cases} 1, & \text{if } i^{th} \text{ machines is assigned to } j^{th} \text{ job} \\ 0, & \text{otherwise} \end{cases}$$

3. Logic for selection of partial bound value

In this section the logic for selection of PBV is given which is coded and implemented in C.

```
# include <stdio.h>
int main(void) {
int n;
scanf("%d", &n);
int arr[100][100];
int i,j,k;
int val=0,ans=9999,min =9999;
for(i = 0;i < n;i++){
for(j = 0; j < n; j++){
scanf("%d",& arr[i][j]);
}
}
for(k = 0;k < n;k++){
val=arr[0][k];
for(i = 0;i < n;i++){
if(i==k)
continue;
for(j = 0; j < n; j++){
if(j==0)
continue;
else{
if(min>arr[j][i]){
min=arr[j][i];
}
}
}
val+=min;
min=9999;
}
printf("%d ",val);
}
return 0;
}
```

4. Computational algorithm of proposed technique

To give an algorithmic representation to the technique mentioned in previous section, let us consider a system in which a set $J = \{J_1, J_2, \cdots, J_m\}$ of m jobs is to be executed on a set $M = \{M_1, M_2, M_3, \cdots, M_n\}$ of n available machines.

Step-1: Input: $m, n, ECM(,)$

Step-2: Obtain the sum of each row of the $ECM(,)$ and store the result in one-dimensional array Sum_Row(,) of order m.

Step-3: Obtain the sum of each column of the $ECM(,)$ and store the result in one-dimensional array Sum_Column(,) of order n.

Step-4: Partitioned the execution cost matrix $ECM(,)$ of order $m \times n$ to sub matrices such that the order of these matrices become square i.e. number of row should be equal to number of column. Partitioning to be made as mentioned in the following steps.

- Step-4.1: Select the n job on the basis of Sum_Row(,) array i.e. select the n job corresponding to most minimum sum to next minimum sum, if there is a tie select arbitrarily.
- Step-4.2: Store the result in the two dimensional array $ECM(,,)$ to form the sub matrices of the sub problem.
- Step-4.3: If all the jobs are selected then go to step 4.7 else steps 4.4
- Step-4.4: Repeat the step 4.1 to 4.3 until the number of jobs become less than n.

Step-4.5: Select the remaining job say $r, r < n$, select the r machines on the basis of Sum_Column(,) array i.e. the machines corresponding to the most minimum sum to next minimum, if there is a tie select arbitrarily.

Step-4.6: Store the result in the two dimensional array $ECM(,,)$, which is the last sub problem.

Step-4.7: List of all sub problems formed through step 4.1 to 4.6 and repeat step 5 to step 9 to solve each of these sub problems.

Step-5: for each sub problem, calculate partial bound value (PBV) of each square sub matrices

Partial Upper Bound Value (PUBV)

$$= \sum_{i=1,j=1}^{n} a_{ij} \quad \text{or} \quad \sum_{i=1,j=n}^{n,1} a_{ij}$$

Partial Lower Bound Value (PLBV)

$$= \sum_{i=1}^{n} \min\left(\sum_{i=1,j=1}^{n} a_{ij}\right) \quad \text{or} \quad \sum_{j=1}^{n} \min\left(\sum_{i=1,j=1}^{m} a_{ij}\right)$$

Step-6: Calculate partial bound value (PBV) of job J_i; for $i = 1$ corresponding to each machine M_j; for $j = 1, 2, 3, \cdots n$..

Step-6.1: For $a_{ij}; i = 1; j = 1$, find the minimum value of remaining columns after neglecting the corresponding row and column elements and find sum of all minimum values corresponding to that element.

Step-6.2: Repeat this process for all $a_{ij}; i = 1; j = 1, 2, \cdots, n$.

Step-6.3: Find the Minimum value from all $a_{ij}; i = 1; j = 1, 2, \cdots, n$ and compare it with partial bound value (PBV).

Step-7: Compare the PBV.

Step-7.1: If PBV < PLBV then drop it.

Step-7.2: If PBV > PUBV then drop it.

Step-7.3: If PBV ($\geq$ PLBV , $\leq$ PUBV) then job J_i; for $i = 1$ assigned to corresponding to that Machine M_j; for $j = 1, 2, \cdots, n$ which having minimum partial bound value.

Step-7.4: If there if tie between two PBV, then select the machine for corresponding job having their minimum execution cost.

Step-8: repeat the process for remaining $m - 1$ jobs till all m jobs are assigned to n machines.

Step-9: repeat the process on each sub matrices and find the minimum cost of the problem by summing execution cost of each job to corresponding machines of each of the sub matrices.

Step-10: Stop.

5. Numerical implementation of proposed technique

Consider a system of a set $J = \{J_1, J_2, J_3, J_4, J_5, J_6\}$ of 6 jobs and a set $M = \{M_1, M_2, M_3, M_4\}$ of 4 machines.

Step-1: Input 6, 4

$ECM(,) =$

	M_1	M_2	M_3	M_4
J_1	6	5	1	6
J_2	2	5	3	7
J_3	3	7	2	8
J_4	7	7	5	9
J_5	12	8	8	6
J_6	6	9	5	10

Step-2: $Sum\_Row(,) = [18, 17, 20, 28, 34, 30]$

Step-3: $Sum\_Column(,) = [36, 41, 24, 36]$

Step-4: Partitioned the execution cost matrix $ECM(,)$ of order $m \times n$ to sub matrices such that the order of these matrices become square i.e. number of row should be equal to number of column. Partitioning to be made as:

$NECMI(,) =$

	M_1	M_2	M_3	M_4
J_2	2	5	3	7
J_1	6	5	1	6
J_3	3	7	2	8
J_4	7	7	5	9

$NECMII(,) =$

	M_3	M_1
J_6	5	6
J_5	8	12

Step-5: for each sub problem, calculate partial bound value (PBV) of each square sub matrices

$$\text{For } NECMI(,) = \begin{cases} \text{Partial Upper Bound Value} & = 22 \\ \text{Partial Lower Bound Value} & = 10 \end{cases}$$

$$\text{For } NECMII(,) = \begin{cases} \text{Partial Upper Bound Value} & = 17 \\ \text{Partial Lower Bound Value} & = 11 \end{cases}$$

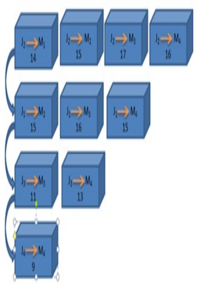
Step-6: For NECMI(,), Calculate partial bound value (PBV) of job J_2 corresponding to each machine M_j; for $j = 1, 2, \cdots, n$.

$\boxed{J_2 \to M_1 \; 14}$ $\boxed{J_2 \to M_2 \; 15}$ $\boxed{J_2 \to M_3 \; 17}$ $\boxed{J_2 \to M_4 \; 16}$

Step-7: Compare the PBV. Therefore job J_2 goes to machine M_1.

Step-8: Repeat the process for remaining $m-1$ jobs till all m jobs are assigned to n machines. Therefore

Jobs		Machine
J_1	$\to$	M_2
J_2	$\to$	M_1
J_3	$\to$	M_3
J_4	$\to$	M_4



Step-9: Repeat the process on NECMII(,)

$J_6 \to M_1$ 14 $J_5 \to M_3$ 17

Step-10: Stop.

The final optimal assignment for first sub matrices $NECMI(,)$ is as follows:

Jobs		Machine
J_1	$\to$	M_2
J_2	$\to$	M_1
J_3	$\to$	M_3
J_4	$\to$	M_4

The final optimal assignment for second sub matrices $NECMII(,)$ is as follows:

Jobs		Machine
J_5	$\to$	M_3
J_6	$\to$	M_1

The execution cost obtained for $NECMI(,) = 18$ and $NECMII(,) = 14$. Hence total optimum execution cost for $ECM(,) = 32$. After partitioning if we use Hungarian technique on sub problems $NECMI(,), NECMII(,)$ we get total optimum cost 33 which is higher than our proposed technique.

6. Check proposed technique

Using above result let we merge the Jobs as follow and then apply proposed technique.

$NECM(,) =$

	M_1	M_2	M_3	M_4
J_1	6	5	1	6
$J_2 * J_6$	8	14	8	17
$J_3 * J_5$	15	15	10	14
J_4	7	7	5	9

$$\text{For } NECM(,) = \begin{cases} \text{Partial Upper Bound Value } = 39 \\ \text{Partial Lower Bound Value } = 18 \end{cases}$$

For $NECM(,)$, Calculate partial bound value (PBV) of job J_1 corresponding to each machine M_j; for $j = 1, 2, \cdots, n$.

$J_1 \to M_1$ 27 $J_1 \to M_2$ 26 $J_1 \to M_3$ 24 $J_1 \to M_4$ 25

Taking minimum value we get job J_1 goes to machine M_3. Similarly repeat this process till all the jobs are not assigned on each machine. The optimum solution in this case is as follows:

Jobs		Machine
J_1	$\to$	M_3
$J_2 * J_6$	$\to$	M_1
$J_3 * J_5$	$\to$	M_4
J_4	$\to$	M_2

Optimum execution cost is 30 by our proposed technique. If we use Hungarian technique on this problem we get optimum cost 31 which is higher than our proposed technique.

7. Conclusion

By using Hungarian method on partitioned matrix $NECMI(,)$ and $NECMII(,)$ we get execution cost 19 and 14 unit whereas when we merge the jobs by our proposed techniques and then apply Hungarian method we get execution cost as 31 units. The table represents the comparative results.

	$NECMI(,)$	$NECMII(,)$	TotalEC	$NECM(,)$
Hungarian Technique	19	14	33	31
Proposed Technique	18	14	32	30

The proposed technique implemented on several sizes of the unbalanced assignment problems to test the effectiveness of the algorithm. Also the whole algorithm is run and coded in C to test the time complexity of technique. In Hungarian approach we use the dummy assignment which may be not possible in real life problems. In our proposed technique we cannot use dummy in getting optimum value. The time complexity of our approach is verified and found that we have the less optimum value as compared to Hungarian Technique. Also some time we have the same optimum solution by our proposed technique. This study is capable to deal all such real life situations where the jobs are more than the number of machines. The obtain solution is under consideration that all the jobs are allotted on the available machines in an optimum way.

References

[1] H.W. Kuhn, "The Hungarian Method for the Assignment Problem," *Naval Research Logistic Quarterly,* 2 (1955), 83–97.

[2] H.W. Kuhn, On the Origin of the Hungarian Method, in:History of Mathematical Programming – A Collection of Personal Reminescences (J.K. Lenstra, A.H.G. Rinnooy Kan, A. Schrijver, Eds) CWI Amsterdam and North-Holland, Amsterdam, 1991, 77–81.

[3] G. R. Burkard, M. Dell'Amico, and S. Martello, *"Assignment Problems",* Siam Society for Industrial and Applied Mathematics, Philadelphia, 2009.

[4] A. Kumar, "A Modified Method for Solving the Unbalanced Assignment Problems", *Journal of Applied Mathematics and Computation,* 176(2006), 76–82.

[5] S. Singh, G. C. Dubey and R. Rajesh Shrivastava, "A Comparative Analysis of Assignment Problem", *IOSR Journal of Engineering (IOSRJEN),* 2(2012), 1–15.

[6] H. A. Taha, *"Operation Research: An Introduction",* Prentice Hall, 8th Edition, 2007.

[7] A. Ahmed and A. Ahmad, "A New Method for finding and Optimal Solution of Assignment Problem", *IJMMS,* 12(1)(2014), 10–15.

[8] A. Khandelwal, "A Modified Approach for Assignment Method", *IJLRST,* 3(2)(2014), ISSN:2278-5299, 136–138.

[9] B. Nathan, F. J. Vasko, "Solving the Unbalanced Assignment Problem: Simpler Is Better", American Journal of Operations Research, 6(2016), 296–299.

[10] A. Khandelwal and A. Agarwal, "An Amalgamated Approach of Fuzzy Logic and Genetic Algorithm for Better Recruitment Process", TNC Transactions of Network and Communications, 3(3)(2015), 23–31.

[11] K. Kadhirvel, K. Balamurugan, "Method for Solving Unbalanced Assignment Problems Using Triangular Fuzzy Number", Int. Journal of Engineering Research and Application, 3(5)(2013), 359–363.

[12] Paul, Mausumi, Sanya Goutam, "Task-Scheduling in Cloud Computing using Credit Based Assignment Problem",*International Journal on Computer Science and Engineering (IJCSE),* 3(10)(2011), 3426–3430.

[13] V. Yedaiah and V. V. Haragopal, "A New Approach for Solving Single Objective Unbalanced Assignment Problem", *American Journal of Operations Research,* 6(2016), 81–89.

[14] Anupong Iampang, Veera Boonjing and Pisit Chanvarasuth, "A Cost and Space Efficient method for Unbalanced Assignment Problems", *IEEE*, 978-1-4244-8503-1/10, 2010.

[15] Kotwal, G. Jameer, S. Dhope, Tanuja, "Unbalanced Assignment Problem by Modified Approach", *International Journal of Advanced Research in Computer Science and Software Engineering,* 5(7)(2015), 451–456.

⋆⋆⋆⋆⋆⋆⋆⋆⋆
ISSN(P):2319 − 3786
Malaya Journal of Matematik
ISSN(O):2321 − 5666
⋆⋆⋆⋆⋆⋆⋆⋆⋆