Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Multi Constrained Unbalanced Assignment Problem: Lexi-Search Algorithm Using Pattern Recognition Technique

A. Prakash[1,2], U. Balakrishna[3]

[1]Research Scholar, JNTUA, Ananthapuramu, Andhra Pradesh, India.
[2]Department of Humanities and Sciences, Vemu Institute of Technology, Chittoor, Andhra Pradesh, India.
[3]Professor, Department of Science and Humanities, Sreenivasa Institute of Technology and Management Studies, Chittoor, Andhra Pradesh, India.

Assume that there are m people and n jobs, such that the number of people should be smaller than the number of jobs and the cost of those occupations to people. With the following restrictions, it is a typical unbalanced assignment problem: n*(≤n) jobs out of n jobs are assigned to m* (≤m) persons out of m person & a job cannot be assigned to more than one person; each of the m* persons can perform at least one job, $[\frac{n^1}{m}]$ number of jobs are to be assigned to a special person, and specific jobs can only be assigned to specific persons with the goal that the total cost of performing the jobs by the persons with the above restrictions is as low as possible.
0-1 integer linear programming (0-1 ILP) is used to formulate this problem. A Lexi-search technique (LSA) based on pattern recognition is created to address the issue as well as possible. For a range of randomly generated test issues, the efficacy of the suggested LS method to the CLBSUAP in comparison to alternative approaches has been investigated.
Keywords: Multi Constrained Unbalanced Assignment Problem[MCUAP], Lexi-search algorithm, Pattern recognition technique, 0-1 Integer linear programming.

1. Introduction

A wide range of industries, including healthcare, education, sports, and transportation, adopt assignment models. Actually, it's a good analytical problem for models of optimization that

use combinatorial optimization or for operational analysis. Furthermore, the assignment problem is an important topic that is used to solve a lot of issues all over the world [3,5,12,14].The assignment outlines the combinational architecture that supports the function, while the aim role represents the preferences that must be optimized.The true question is, "What should be done to execute the task optimally while meeting all the relevant requirements at the same time?" .The problem has already been addressed by a variety of methods [4], such as population search [8], heuristic approaches [6], and precise procedures [13].The goal of the study is to identify a job between two or more element sets that can reduce the total cost of each matching pair.Depending on the properties of the matching sets and the cost functional form, the assignment problem can be classified as a quadratic, bottleneck, linear, or multidimensional group [9].Because of this, each assignment problem has a table or matrix. People, machines, and assigned responsibilities and tasks are typically found in the rows and columns.The elements in the matrix represent the time and expense required to finish each task.Any algorithm's objective is to provide a workable solution to the given problems. The approach relies on a specially designed algorithm called the Lexi Search due to its intricacy [1,2,9,10,11,15].

An outline of the structure of this document is provided below: Section 2 provides specifics on the problem statement and mathematical model of the MCUAP. In Section 3, the suggested algorithm for solving the MCUAP is explained. An example can be found in Section 4. Section 5 has comparable data, while Section 6 presents the findings.

2. PROBLEM STATEMENT AND FORMULATION

Think of m people, or I={1,2,3,…..,m}, n jobs, or J={1,2,3,…..,n}, and a nonnegative integer time T(i,j), which shows how long it takes the ith person to finish the jth job. Assume that m people will be assigned

$n^1(< n)$ jobs out of n jobs so that each person can perform at least one job, that a certain number of jobs will be assigned to a special person, and that certain $[\frac{n^1}{m}]$ jobs can only be assigned to specific people. The issue According to MCUAP, n^1 jobs are assigned to m people so that each person can complete at least one work. $[\frac{n^1}{m}]$ Jobs are completed by a special person, and specific jobs can only be assigned to certain people with the goal that the people's total time spent on the assigned tasks is as little as possible.

The MCUAP model is formulated using the following assumptions:

- $n^1>m$
- All the persons start to work at the same time
- The same job cannot be done by more than one person
- A person is allowed to do more than one job, but one after another in any order
- The elements in the time matrix take arbitrary values

Using the above assumptions, the MCUAP model is developed as 0-1 ILP:

$$MIN \sum_{i\in I,\ j\in J,} T(i,j)X(i,j)$$

MIN

(3.2.1)

Subject to: $$\sum_{i\in I} X(i,j)=1,\ j\in J \quad (3.2.2)$$

$$\left\lfloor \frac{n^1}{m} \right\rfloor \le \sum_{j\in J} X(i,j) \le \left\lceil \frac{n^1}{m} \right\rceil, i\in I \quad (3.2.3)$$

$$\sum_{i\in I}\sum_{j\in J} X(i,j)=n^1 \quad (3.2.4)$$

$$SJ^* = \{(i,j)/\, i\in I, j\in J \ \ and\ \ i\ \ denotes\ \ \ specified\ \ \ persons\ \ \ and\ \ \ j\ \ \ denotes\ \ \ specified\ \ \ jobs\ \} \quad (3.2.5)$$

$$X(i,j)=0\, or\, 1, (i,j)\in IXJ \quad (3.2.6)$$

(3.2.1) is an objective function in this mode that reduces the time required for n1 jobs for m individuals. Constraint (3.2.2) states that only one worker may be allocated to each task.

At least [n1m] jobs are allocated to each individual, according to constraint (3.2.3). The constraint (3.2.4) indicates that m people are assigned to n1 occupations. The individual jobs to specific people are represented by constraint (3.2.5). Lastly, if X (i, j) = 1 in (3.2.6), the jth job will be assigned to the ith person; if not, X (i, j) = 0.

3. PRELIMINARIES OF LSA

3.1 Feasible Solution

An MCUAP solution that meets all the requirements in (3.2.1-3.2.6) is considered a feasible solution.

3.2 Pattern

A pattern linked to a function is a two-dimensional array. The model is deemed practicable if X is a feasible solution and its value is established using (3.3.2.1), which explains the overall cost allocation, and is equal to the value of the objective function.

$$V(X)= \sum_{j\in J} T(i,j)X(i,j) \qquad (3.3.2.1)$$

3.3 Alphabet Table

The alphabetic table T(i, j)'s time matrix's elements are arranged in non-decreasing order and have names ranging from 1 to m×n. Assume that SN = {1,2,..., m×n } is a set of ordered m×n matrices and indices. Assign the row and column symbols of the pairings marked with SN to the R and C sets, assuming that T and CT stand for time and its cumulative sum for the components of T.

A list of designated variables and a table of letters (SN, T, CT, R, C, etc.) are included in this table. Si is an SN member, and Ls={S1,S2,.....,Sr} is an ordered SN index string. The pattern of Ls displayed in the commands and these symbols is unrelated to the sequence of Si in the series. The order of SN symbols is Si i=1,2,3…r-1 for uniqueness.

3.4 Word and partial word

A systematic progression Ls= {S1, S2,..., Sr} is a word of length r, according to our definition. If $r < n^1$the word Ls is considered partial feasible, and when $r = n^1$, it is the full-length word. Any index in SN can serve as the principal position in Ls. A set of words is identified as the leader by a partial term, Ls. The leader is said to be both feasible and infeasible if there is at least one feasible word in the word block it specifies.

3.5 Value of a word (V (Ls))

V(Ls) is the value of the word Ls, and it is computed using V(Ls-1) + D(Sr) = V(L0)=0. It is clear that T(Sr) is the time array set up so that, for every r = 1, 2, 3,... n, T(Sr)≤T(Sr-1). V(Ls) and V(X) have comparable values (Sundara Murthy).

3.6 Calculation of bounds

In the search sector for NP-hard problems, the strong lower and upper bound configuration is more difficult to manage. For objective functions of minimization, the upper bound of Ls is first thought to have a huge value (UB= VT= 9999) as a trial solution. When considering the block values of words represented by Ls, the lower limit LB (Ls) can be understood as follows:

$$LB(Ls) = V(Ls) + CT(S_k+n^1-k)-CT(S_k)$$

3.7 Lexi Search Method

When using combinatorial optimization models, the best results found through exact search methods have grown in value. The full and implicit search methods are obviously similar to these algorithms. The Branch and Bound method is one of the most widely used implicate search strategies (B & B).The LSA is one such sophisticated enumeration method that frequently yields the best result while merely examining a subset of the solution space (Pandit, 1962).

In actuality, B & B might be regarded as a special LSA circumstance. All B&B elements are protected by the LSA, including the systematic development of feasible solutions, viability testing, and the identification of restrictions for partially viable solutions.

The phrase "Lexi-search" describes the manner in which the entire search process is carried out and seems to be analogous to a dictionary search for a word's definition. Additionally, this methodical quest protection includes search time and stack overflow.

Determining the viability and defining effective boundaries are the two main issues with implicit enumeration techniques. For a few issues, it is crucial to confirm the viability. A Lexi-search strategy was developed and implemented with a pattern-recognition technology to address this problem (Murthy, 1976):

Every solution to an issue has a connection to a specific pattern. Partial patterns can be found in partially solved problems. An alphabet table is defined by the words that, in lexicographic or dictionary order, reflect the pattern. The first boundaries are established when a partial word is used; the locations where the value falls short of the trail's meaning are then assessed for viability when choosing the best word.

3.8 Proposed LSA

The following is a discussion of the steps that go into LSA:

Step 1: Input

Time matrix T=[T(i, j)],the required parameters n, m, n^I, $SJ^* = \{(i,j)/i \in I, j \in J \text{ and } i \text{ denotes specified persons and } j \text{ denotes specified jobs}\}$ and UB = VT = 9999 (large value) and then Step 2.

Step 2: Create an alphabet table using the provided Time matrix T, then move on to Step 3.

Step 3: Setting the Bounds

Ls = (Ss) = 1, Ss ЄSN, a partial word with a single word length (s=1), is used to start the process.

Compute LB (Ls). Proceed to Step 5 if the lower limit of an LB (Ls) is strictly less than VT; otherwise, proceed to Step 4.

Step 4: All partial words of orders that follow Ls will be disregarded and move on to Step 7 if the lower limit of an LB (Ls) is larger than or equal to VT. Afterward, delete the partial word Ls and suspend the block of words with Ls as a leader.

Step 5: Checking the Feasibility

If the partial word Ls follows the constraints then it is said to be feasible, otherwise, it would be infeasible. If Ls is feasible, then accept it and continue for next partial word of order s+1 and go to Step 6, else proceed with the next partial word of order s by considering another letter that succeeds Ss in its s^{th} position and go to Step 3.

Step 6: Concatenation

Step 8 will be reached if Ls is a full-length feasible word (such that s= n^1) and you replace VT with the value LB(Ls). $L_{s+1}= L_s * (S_{s+1})$, where * denotes the concatenation operation and advances to step 3, can be used to concatenate a partial term, Ls.

Step 7: The search is finished, and Step 9 is reached if all order terms have expired and the word Ls has a length of one.

Step 8: Backtracking

Backtracking is used to examine the search space; the current VT is taken to be an upper bound, and the search is carried out using the subsequent letter of the partial word of order s-1; proceed to Step 3. Steps 3 through 8 should be repeated until VT stops changing, and any viable or impractical options that are not part of the best solution should be ignored. Proceed to step 9.

Step 9: Proceed to Step 10 after storing the current VT and the word Ls.

Step 10: Stop

Finally, VT provides the optimal solution at the end of the search and the word Ls gives the position of the letters and with the aid of Ls one can find the optimal schedule for the connectivity of the given cities.

4. NUMERICAL ILLUSTRATION

To explain the concepts and meanings that are part of the problem we considered n=9, m=2, n^1=6, 2nd person can do $[\frac{n^1}{m}]=[\frac{6}{2}] = 3$ jobs, SJ*= {(1,5),(2, 7)}.The Time matrix T(i, j)of MCUAP is given as follows.

Table-3.4.1

$$T(i,j) = \begin{pmatrix} 7 & 9 & 8 & 14 & 6 & 4 & 16 & 25 & 39 \\ 8 & 3 & 4 & 21 & 31 & 2 & 3 & 4 & 70 \end{pmatrix}$$

T(i j) is taken in the numerical illustration in Table-1 as non-negative. In Matrix X = [X(i,j)/X(i,j)= 0 or1] indicators in which X(i,j)=1 means the ith person j^{th} job, or X(i,j)=0.Xis called a solution.

4.1 Alphabet – Table

Table 3.4.2 relates to alphabet table construction for time matrix T. In Table 3.4.2, the notations SN, T, CT, R and C respectively denote the serial number, time, cumulative time, row and column of the indices.

Table –3.4.2 Alphabet Table (AT)

SN	T	CT	R	C
1	2	2	2	6
2	3	5	2	2
3	3	8	2	7
4	4	12	1	6

SN	T	CT	R	C
5	4	16	2	3
6	4	20	2	8
7	6	26	1	5
8	7	33	1	1
9	8	41	1	3
10	8	49	2	1
11	9	58	1	2
12	14	72	1	4
13	16	88	1	7
14	21	109	2	4
15	25	134	1	8
16	31	165	2	5
17	39	204	1	9
18	70	274	2	9

4.2 Search Table

A numerical example in Table 3.4.2 shows the logical flow of the developed LSA.

Table–3.4.3 Search Table (ST)

SN	1	2	3	4	5	6	V(I)	LB(I)	R	C	REM
1	1						2	20	2	6	A
2		2					5	20	2	2	A
3			3				8	20	2	7	A
4				4			12	20	1	6	R
5				5			12	20	2	3	R
6				6			12	25	2	8	R
7				7			14	29	1	5	A
8					8		21	29	1	1	A
9						9	29	29	1	3	A=VT=29
10					9		22	30	1	3	R>VT
11				8			15	31	1	1	R>VT
12			4				9	23	1	6	R
13			5				9	26	2	3	A
14				6			13	26	2	8	R
15				7			15	30	1	5	R>VT
16			6				9	30	2	8	R>VT
17		3					5	23	2	7	A

18			4				9	23	1	6	R
19			5				9	20	2	3	R
20			6				9	30	2	8	R>VT
21		4					6	27	1	6	R
22		5					6	31	2	3	R>VT
23	2						3	24	2	2	A
24		3					6	24	2	7	A
25			4				10	24	1	6	A
26				5			14	24	2	3	A
27					6		18	24	2	8	R
28					7		20	27	1	5	A
29						8	27	27	1	1	A=VT=27
30					8		21	29	1	1	R>VT
31				6			14	27	2	8	R=VT
32			5				10	27	2	3	R=VT
33		4					7	28	1	6	R>VT
34	3						3	28	2	7	R>VT

In the end, the value of VT is 27 lies in 29th row of the table and its corresponding word is given by $L_6 = (2, 3, 4, 5, 7, 8)$, which is also shown as a pattern in Table 3.4.3.

TABLE-3.4.4

$$X(i, j) = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

5. EXPERIMENTAL RESULTS

In MATLAB, a computer program is created and tested for the intended LSA. The experiments are performed by uniformly generating the cost values (C_{ij}) between [1, 1000]. We experimented with a variety of challenges in various sizes. Cost random integers are used to build the matrix. The results are summarized in Table-5. It has been demonstrated that finding the best answer takes significantly less time. With the suggested LSA, the CPU takes time to identify the optimal solutions for various hard situations, as seen in the table below.

Table-5: Computational ResultsTable

S.No.	Number of Persons(m)	Number of Jobs(n)	Number of jobs to be done from n jobs(n1)	Specific jobs to specific person	Average time for alphabet table generation (AT)	Average time for getting optimal solution (OS)	Avg. (AT+ OS) Seconds
1.	2	12	10	(1,4),(2,10)	0.0000	0.0000	0.0000

2.	4	28	24	(2,25),(3,8)	0.0000	0.0000	0.0000
3.	5	35	26	(1,22),(4,27),(5, 33)	0.0000	0.0547	0.0547
4.	6	46	38	(1,6),(3,19), (6,32)	0.0000	0.1088	0.1088

References

1. A Prakash, Uruturu Balakrishna and Jayanth Kumar Thenepalle, ”An exact algorithm for constrained k-cardinality unbalanced assignment problem”,International Journal of Industrial Engineering Computations,13(2),pp.267-276,2022
2. Balakrishna. U, Sk. Mastan, G.S.S.Raju and Jayanth Kumar T,” Constrained Load Balancing solid unbalanced assignment problem:An exact enumerative algorithm”, International Journal of Mechanical Engineering,6(special issue 2022),pp.780-794,2022
3. E. Çela, “Assignment Problems,” in Handbook of Applied Optimization, Part II-Applications, vol. 6, pp. 667–678, 2002.
4. E. Ersoy, E. Özcan, and A. S. Uyar, “Memetic algorithms and hyperhill-climbers,” in Proceedings of the 3rd Multidisciplinary International Scheduling Conference: Theory and Applications (MISTA07), pp. 159–166, 2007.
5. H. Basirzadeh, “Ones assignment method for solving assignment problems,” Applied Mathematical Sciences, vol. 6, no. 45-48, pp. 2345–2355, 2012.
6. H. Turabieh and S. Abdullah, “An integrated hybrid approach to the examination timetabling problem,” Omega , vol. 39, no. 6, pp. 598–607, 2011.
7. M. Ayob, S. Abdullah, and A. M. A. Malik, “A practical examination timetabling problem at the Universiti Kebangsaan Malaysia,” International Journal of Computer Science and Network Security, vol. 7, no. 9, pp. 198–204, 2007.
8. M. Caramia, P. Dell'Olmo, and G. F. Italiano, “Novel local-search-based approaches to university examination timetabling”, INFORMS Journal on Computing, vol. 20, no. 1, pp. 86–99, 2008.
9. Masthan SK, Balakrishna U, Sankar Sekhar Raju and Jayanth Kumar T,”An exact Pattern Recognition based Lexi Search Algorithm for restricted Unbalanced Assignment Problem”, International Journal of Mechanical Engineering 7(1),pp.6672-6678,2022
10. Masthan SK, Balakrishna U, Sankar Sekhar Raju and Jayanth Kumar T,”Solving an open k-city travelling salesman problem with ordered constraint: An exact Leaxi-search algorithm”, Turkish Online Journal of Qualitative Inquiry(TOJQI), 12(7),4520-4528,2021
11. N Vasantha kumar, Balakrishna. U,“Constrained Asymmetric Three Dimensional Generalized Travelling Salesman Problem::Pattern Recognition Lexi Search Approach”, Journal of Emerging Technologies and Innovative Research(JETIR),6(3),pp.172-176,2019.
12. R. E. Burkard, “Selected topics on assignment problems,” Discrete Applied Mathematics: The Journal of Combinatorial Algorithms, Informatics and Computational Sciences, vol. 123, no. 1-3, pp. 257–302, 2002.
13. R. Qu, E. K. Burke, B. McCollum, L. T. G. Merlot, and S. Y. Lee, “A survey of search methodologies and automated system development for examination timetabling,” Journal of Scheduling, vol. 12, no. 1, pp. 55–89, 2009.
14. S. Singh, “A Comparative Analysis of Assignment Problem,” IOSR Journal Of Engineering, vol. 2, no. 8, pp. 1–15, 2012.
15. U. Balakrishna, and Sundaramurthy. M,“A Pattern Recognition Lexi Search Approach to Generalized Time Dependent Travelling Salesman Problem”, Journal of Operational Research society of India(Springer Publications), 49(3),pp.191- 208,2012.