ISSN: 0974-5823 Vol. 7 No. 1 January, 2022

International Journal of Mechanical Engineering

AN EXACT PATTERN RECOGNITION BASED LEXI SEARCH ALGORITHM FORRESTRICTEDUNBALANCED ASSIGNMENT PROBLEM

Sk.Mastan[1] U.Balakrishna[2] G.SankarSekhar Raju[3] T. Jayanth Kumar[4]

[1]Research Scholar, JNTUA, Ananthapuramu, India-515002

[2]Professor, SITAMS, Chittoor, India-517127

[3]Professor, JNTUA, Pulivendula,, India-516390

[4]Assistant Professor, Jain (Deemed to be University), Kanakapura Rd, Bengaluru, Karnataka, India-562112

ABSTRACT

In this paper, a practical variant of unbalanced assignment problem called restricted unbalanced Assignment problem(RUAP). The RUAP isa usual unbalanced Assignment Problem with the restrictions that a job cannot be assigned to more than one person, each person is restricted to do atleast certain jobs and specific jobs can only be assigned to specific persons with the objective that the overall time on performing the given jobs by the persons is minimum. This problem is formulated using 0-1 integer Linear Programming (0-1 ILP). To deal withthe RUAP optimally, an exact Lexi-search algorithm (LSA) witha pattern recognition technique is developed.The efficiency of the proposed LS approach to the RUAP as against different approacheshas been examined for a variety of randomly generated test problems.

KEYWORDS

Restricted unbalanced Assignment problem(RUAP), Lexi-searchalgorithm, Pattern recognition technique, 0-1 Integer linear programming

1. INTRODUCTION

The assignment models exist in many ways, such as healthcare, transport, Sports and Education. Indeed, It's a good analysissubject in Combinatorial optimization models of operational analysis or optimization. Besides, the assignment problem is an important topic used for solving several issues globally[1]. In many educational programmes around the world, this issue has been encountered.

The problem with the assignment relates to the study of how to better assign items (optimal)[2, 3]. The assignments and objective function are two assignment problem components. The assignment describes a combinational framework underlying the function, while the goal role represents the preferences to be optimised. The question, however, is, "What should be done to execute the task optimally while meeting all the relevant requirements at the same time?".Because of the problem, different methodologies [1,2] have already been proposed, including exact techniques [4], heuristic approaches [5,6], population search[7], and hybrid algorithms[8].

The goal of the analysis is to define a task between two or more element sets that can reduce the total cost of every matching pair.By the nature of the matching sets and the cost functional form, the assignment problem can be classified into a quadratic, bottleneck, linear, or multidimensional group [9].So in every assignment problem, there is a table or matrix. The rows and columns typically take persons/machines and jobs/tasks which are to be assigned. The elements in the matrix represent the cost/time required for performing each job.All the algorithms developed are intendedto provide an efficient solution for the assignment problems. The solution uses a specially designed algorithm called the Lexi Search[10,11] because of its complexity.

This paper is structuredas follows: The problem statement and the mathematical model of the RUAP are described in Section 2. Section 3 outlines the proposed algorithm for the resolution of the RUAP. An example is presented in Section 4. Section 5 offers comparable data. Finally, concluding remarks are given in Section 6.

2. PROBLEM STATEMENT AND FORMULATION

The RUAP can be described accordingly:

Consider m persons i.e., I={1,2,3,…..,m}, n jobs i.e., J={1,2,3,…..,n} and a non negative integer time T(i,j) indicates ith person takes to complete the jth job.Let n^1 (<n) jobs out of n jobsare to be assigned to m personssuch thatat least $\left\lfloor \frac{n^1}{m} \right\rfloor$ number of jobs are to be assigned to each person.The problem RUAP determines n^1 jobs are assigned to m persons such that (n^1/m)jobs are done by every person and specific jobs can only be assigned to specific persons with the objective that the overall time on performing the given jobs by the persons is minimum.

The RUAP model is formulated using the following assumptions:

- n^1>m
- All of 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/cost matrix take arbitrary values

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

$$MIN \sum_{j\in J} T(i,j)X(i,j) \quad (1)$$

Subject to:

$$\sum_{i\in I} X(i,j)=1,\ j\in J \quad (2)$$

$$\left\lfloor \frac{n^1}{m} \right\rfloor \le \sum_{j\in J} X(i,j) \le \left\lceil \frac{n^1}{m} \right\rceil, i\in I \quad (3)$$

$$\sum_{i\in I}\sum_{j\in J} X(i,j)=n^1 \quad (4)$$

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

{ }{ } (5)

$$X(i,j)=0 \text{ or } 1, (i,j)\in IXJ \quad (6)$$

In this modal, (1) is an objective function which minimises the time of n^1 jobs for m people. Constraint (2) represents each job can be assigned to only one person. Constraint (3) represents atleast $\left\lfloor \frac{n^1}{m} \right\rfloor$ jobs are assigned to every person. Constraint (4) denotes n^1 jobs are assigned to m persons. Constraint (5) represents the Specific jobs to specific persons. Finally, In (6) X(i,j)=1, j^{th} job will be allocated to the i^{th} personand otherwise X(i,j)=0

3. PRELIMINARIES OF LSA

3.1 Feasible Solution

A solution to the RUAP fulfils all the limitations of (2) to (6), then it is called a feasible solution.

3.2 Pattern

A two-dimensional arrayis referred to as a pattern associated with an assignment. A pattern is called feasible if X is a feasible solution and itsvalueis calculated using (7), gives the total assignment cost and this is same as that of the value of the objective function.

$$V(X)=\sum_{j\in J} T(i,j)X(i,j) \quad \text{……(7)}$$

3.3 Alphabet Table

The Time Matrix elements T(i, j) in an alphabet table are arranged in non-decreasing order andnamed from 1 to $m\times n$. Let SN = {1,2,..., $m\times n$ } be the set of $m\times n$ ordered indices, arrays.The time and its cumulative sums for T elements respectively are expressed by T and CT. Let arrays R, C be the row, column indices of SN's ordered pairs.

This table includes the ordered index set as the table of alphabets, such as SN, T, CT, R&C. Ls={S_1,S_2,..... S_r} is a commanded SN index string where S_i is an SN member. The pattern Ls shown in orders and these indexes are independent of the order S_i in sequence.To be unique, the indexes of the SN are ordered in the order of $S_i < S_{i+1}$, i=1,2,3… r-1.

3.4 Word and partial word

Asystematic sequence Ls= {S_1, S_2 ,..... S_r} is said to be a word of length *r*. If $r < n^I$, then the word *Ls* is called a partial feasible word and it is the full-length word when $r = n^I$. Any index in SN can be the prime position in Ls. A partial word Ls describes a set of words as the leader, with Ls. If the word block defined by it has at least one feasible word then it is said that the leader is feasible and infeasible, otherwise.

3.5 Value of a word(V(Ls))

The value of the word Lsis denoted by V(Ls) and is calculated using$V(L_{s-1}) + D(S_r)$ with $V(L_0)=0$ and obviously, $T(S_r)$ be the time array that is arranged in a way such that $T(S_r) \leq T(S_{r-1})$ for r=1,2,3,….m×n. V(Ls) is similar to V(X) value (Sundara Murthy).

3.6 Calculation of bounds

The powerful lower and upper bound configuration is harder to handle the NP-hard problem search field. Initially, as a trial solution, the upper bound of Ls is believed to be a high value (UB= VT= 9999) (for objective functions of minimisation). The lower bound LB (Ls) can be described as follows for block values of words represented by Ls:
$LB(L_s) = V(L_s)+CT(S_k+n^I-k)-CT(S_k)$

3.7 Lexi Search Method:

Optimum solutions achieved using exact search methods have become more desirable in dealingcombinatorial optimization models. The same approaches as the full and implicit search methods can be observed. One of the most common implicate searching techniques is the Branch and Bound approach (B & B).

One such implicated method of enumeration is the LSA, where only a fractional part of thesolution space is examinedand provides optimal solution systematically (Pandit, 1962).

In reality, B & Bcan be viewed as a special LSAcase. AllB & Belements are protected by the LSA, including the design of feasible solutions, the test of viability and the determination of limits to a partially viable solution.

In a specific way, the entire search process is carried out and looks analogousto a dictionary search for the meaning of a word, so it is called "Lexi-search". Also, this systemic quest defend stack overflow and search time

The main issue with implicit enumeration methods is (i) to verify the feasibility (ii) to set effective limits. For a few problems, checking the feasibility is a great deal. To deal withthis problem effectively, a Lexi-search method was developed and defined following pattern-recognition technique (Murthy, 1976):
"Each problem solution has to do with a specific pattern. A partial pattern is a partial solution.An alphabet table is described by the words describing the pattern in a lexicographic or dictionary order. The first limits are defined when a partial word is used, and then the sections for which the value is lower than the meaning of the trail are checked for viability when choosing the ideal word".

3.8 Proposed LSA

The steps involved in LSAare discussed as follows:

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: Use the given Time matrix T to construct an alphabet table, and then transfer to Step 3.
Step 3**: Setting the Bounds**
The procedure begins with a partial word Ls = (Ss) = 1, Ssϵ*SN*, which is a single word length, i.e. *s=1*.
ComputeLB (Ls). If the lower limit of an LB (Ls) is strictly less than *VT*, then move to step 5, else go to Step 4.
Step 4: If the lower limit of an LB (Ls) is greater than or equals to *VT*, then delete the partial word *Ls* and suspend the block of words with *Ls* as a leader and thus all the partial words of the order s that succeeds Ls will be discarded and go to Step 7.
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**

If Ls is a full-length feasible word s (i.e.s= n^l), then substitute VT with the value LB(Ls) and go to Phase 8. If Ls is a partial term, it can be concatenated by using Ls+1= Ls * (Ss+1), where * indicates the operation of concatenation and goes to step 3.

Step 7: If all the terms of order s are overand the length of the word Ls is 1, then the search process is completedand moveto Step 9.

Step 8: **Backtracking**

To explore the search space, backtracking is adopted; the current VT is presumed to be an upper bound and the search continues with the next letter of the partial word of order s-1, go to Step 3. Repeat steps 3 to 8 until there is no further change in VT and disregard feasible/infeasible alternatives that are not in the optimal solution. Go to step 9

Step 9: Storethe current VT and the word Ls and Go to Step 10

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. (Remove this line)

4. Numerical illustration

To explain the concepts and meanings that are part of the problem we considered n=8, m=2, n^1=6, SJ*= {(2, 2)}.The Time matrix $T\ (i,\ j)$ of RUAPis given as follows.

Table-1

$$T\ (i,\ j) = \begin{pmatrix} 3 & 3 & 7 & 13 & 5 & 3 & 15 & 5 \\ 7 & 2 & 3 & 4 & 1 & 16 & 30 & 20 \end{pmatrix}$$

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 i^{th}person j^{th}job, or X(i,j)=0.*X*is called asolution.

4.1 Alphabet – Table

Table 2 relates to alphabet table construction for time matrix T. In Table 2, the notations SN, T, CT, R and C respectively denotes the serial number, time, cumulative time, row and column of the indices.

Table - 2: ALPHABET TABLE (AT)

SN	T	CT	R	C
1	1	1	2	6
2	2	3	2	2
3	2	5	2	7
4	3	8	1	6
5	3	11	2	3
6	3	14	2	8
7	5	19	1	5
8	6	25	1	1
9	7	32	1	3
10	7	39	2	1
11	8	47	1	2
12	13	60	1	4
13	15	75	1	7
14	20	95	2	4
15	24	119	1	8
16	30	149	2	5

4.2 Search Table

A numerical example in Table 3 shows the logical flow of the developedLSA.

Table –3: SEARCH TABLE (ST)

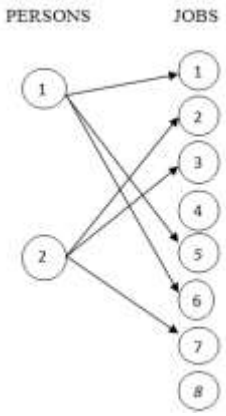
SN	1	2	3	4	5	6	V(I)	LB(I)	R	C	REM
1	1						1	14	2	6	A
2		2					3	14	2	2	A
3			3				5	14	2	7	A
4				4			8	14	1	6	R
5				5			8	16	2	3	R
6				6			8	19	2	8	R
7				7			10	23	1	5	A
8					8		16	23	1	1	A
9						9	23	23	1	3	A=VT=23
10					9		27	24	1	3	R>VT
11				8			11	25	1	1	R>VT
12			4				6	17	1	6	R
13			5				6	20	2	3	A
14				6			9	20	2	8	R
15				7			11	24	1	5	R>VT
16			6				6	24	2	8	R>VT
17		3					3	17	2	7	A
18			4				6	17	1	6	R
19			5				6	20	2	3	R
20			6				6	24	2	8	R>VT
21		4					4	21	1	6	R
22		5					4	25	2	3	R>VT
23	2						2	18	2	2	A
24		3					4	18	2	7	A
25			4				7	18	1	6	A
26				5			10	18	2	3	A
27					6		13	18	2	8	R
28					7		15	21	1	5	A
29						8	21	21	1	1	A=VT=21
30					8		16	23	1	1	R>VT
31				6			10	21	2	8	R=VT
32			5				7	21	2	3	R=VT
33		4					5	22	1	6	R>VT
34	3						2	22	2	7	R>VT

In the end, the value of VT is 21lies in 29th row of the tableand its corresponding word is given byL_6 = (2,3,4,5,7,8), which is also shown as apattern in Table 4.

TABLE-4

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

The assignment represented by the above pattern is [(2,2), (2,7), (1,6), (2,3), (1,5), (1,1)], where, the 1st person completed 1st, 5th and 6th jobs; 2nd person completed 2nd, 3rd and 7th jobs.It is notedthat 2nd job is assigned to 2ndperson which is pre-assigned.It can also be graphically representedas:



5. EXPERIMENTAL RESULTS

A computer programme is written and inspected in MATLAB for the proposed LSA. The experiments are performed by uniformly generating the time values (T_{ij}) between [1, 1000]. For different sizes, we tried out a series of problems. The Time Matrix is built using random numbers. The results are tabulated in Table-5. It has been shown that there is comparatively less time needed to find the optimal solution. The following table demonstrates that the CPU takes time to find the best solutions for various hard instances with the proposed LSA.

Table: 5

SN	*m*	*n*	n^l	*SJ**	The proposed LSA CPU Run Time in seconds		Total time Avg. (AT+ ST) Sec
					Avg. AT	Avg. ST	
1	2	10	8	(2,2)	0.0000	0.0000	0.0000
2	3	12	9	(3,7)	0.0000	0.0000	0.0000
3	7	21	15	(4,6),(4,8)	0.0547	0.0000	0.0547
4	10	32	24	(7,9)(7,10)	0.1088	0.0000	0.1088
5	15	45	40	(10,9)(10,12)(10,42)	0.2193	0.2751	0.4944
6	22	100	79	(20,22)(20,55)(20,69)(20,97)	0.2752	0.2192	0.4944

** In the table-5 SN = serial number, m = number of persons, n=number of jobs, n^l=number of jobs to be done from n, SJ* is specific jobs to the specified persons, AT = CPU run time for alphabet table printing, ST=CPU run time for the optimum search table solution.

6. CONCLUSION

In this paper, we built **anLSA**for solving the RUAP based on a pattern recognition technique to get an optimal solution. This problem is formulated using 0-1 ILP. For a better understanding of the principles and steps involved in the algorithm, an appropriate numerical illustration is presented. Randomly produced test calculation using our Lexi method has shown comparable results. However, the run time and convergence rate of the proposed LSA exhibitedless computational expensive than that of the other methods. Besides, in many combinatorial problems, Lexi-search algorithms are proved to be more effective and faster than other algorithms.Based on this experience, we strongly feel that this algorithm can perform problems of a larger scale and that more is very effective.

7. REFERENCES

1. H. Basirzadeh, "Ones assignment method for solving assignment problems," *Applied Mathematical Sciences*, vol. 6, no. 45-48, pp. 2345–2355, 2012.
2. S. Singh, "A Comparative Analysis of Assignment Problem," *IOSR Journal Of Engineering*, vol. 2, no. 8, pp. 1–15, 2012.
3. E. Çela, "Assignment Problems," in *Handbook of Applied Optimization, Part II-Applications*, vol. 6, pp. 667–678, 2002.
4. 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.
5. 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.
6. 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.

7. E. Ersoy, E. Özcan, and A. S. Uyar, "Memetic algorithms and hyperhill-climbers," i*n Proceedings of the 3rd Multidisciplinary International Scheduling Conference: Theory and Applications (MISTA07)*, pp. 159–166, 2007.
8. H. Turabieh and S. Abdullah, "An integrated hybrid approach to the examination timetabling problem," *Omega* , vol. 39, no. 6, pp. 598–607, 2011.
9. 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.
10. 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.
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.