Journal of Neonatal Surgery
ISSN(Online): 2226-0439
Vol. 14, Issue 15s (2025)
https://www.jneonatalsurg.com

OPEN ACCESS

Exact Lexicographic Scheduling For Sales Force Optimization With Facility Constraints

A. Prakash[a,b], U. Balakrishna[c]

[a]Research Scholar, JNTUA, Ananthapuramu, AP, India

[b] Professor, Department of Humanities and Sciences, Vemu Institute of Technology,P.Kothakota, Chittoor, AP, India

[c]Professor, Department of Science and Humanities, Sreenivasa Institute of Technology and management Studies, Chittoor, AP, India

Cite this paper as: A. Prakash, U. Balakrishna, (2025) Exact Lexicographic Scheduling For Sales Force Optimization With Facility Constraints. *Journal of Neonatal Surgery*, 14 (15s), 433-451.

ABSTRACT

There are I={1,2,3,…..,n} be set of n persons, J={1,2,3,…..,m} be set of m schedules and k={1,2,3,…….,p} be set of p facilities. S (t, j) is the number of calls made in jth schedule at time t. SB (t) is the minimum number of calls to be made by the persons at time t. C (i, j, k) is the cost of assigning jth schedule to ith person at facility k i.e., the cost depends on the facility k which is the third independent factor which influences the cost. The restriction is: If the same schedule is assigned to different persons then the facilities should be different.

The problem is to assign the schedules to n^1 (<n) persons with minimum cost with the above restriction and the total number of calls made in each time is greater than SB(t).

In the sequel we will develop a Lexi Search algorithm based on the 'Pattern Recognition Technique' to solve this problem which takes care of the simple combinatorial structure of the problem.

1. INTRODUCTION

This problem is a variant generalized assignment problem and the literature review of generalized assignment problem already discussed in chapter-1.Jacking Elzing and Vemuganti (1976), studied the above problem in two dimensional i.e., There are i persons and j schedules, C_{ij} is the cost of assigning jth schedule to ith person. We are given t time periods and in each period the total number of calls i.e., S_t made in jth schedule at time t is given. During each period at least one call can be made on an outlet. Aim is to assign call schedules to persons so as to minimize cost, subject to limits on the number of calls made in each period. But in the above attempt the simple combinatorial structure of the problem is not at all taken into consideration.

In the present problem out of the given n persons only n^1(<n) persons have assigned schedules. It is assumed that 1) We have considered the third dimension time (availing facility) which influences the cost 2) If different persons have given same schedules then the corresponding facilities are different 3) $S_t \geq 1$

1. Mathematical Formulation:

The solution X (i, j, k), (i, j, k)$\in I \times J \times K$ be defined as:

X (i, j, k) = 1, if the jth schedule is assigned to the ith person at facility k

= 0, otherwise

S (t, j) is the number of calls made in period t under schedule j

C (i, j, k) is the cost for assigning jth schedule to the ith person at facility k

SB (t) is the minimum number of calls to be made in period t

Then the problem can be defined as:

$$\text{MIN} \quad \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} C(i,j,k)X(i,j,k) \qquad (5\text{-}1)$$

$$\text{Subject to:} \quad \sum_{j=1}^{m}\sum_{k=1}^{p} s(t,j)X(i,j,k) \geq SB(t) \quad (t=1,2,3,\ldots..,T) \qquad (5\text{-}2)$$

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} X(i,j,k) = n^1, \quad n^1 = \text{number of persons} \qquad (5\text{-}3)$$

$$\sum_{j=1}^{m}\sum_{k=1}^{p} X(i,j,k) = 1 \quad \text{for } i=1,2,\ldots,n \qquad (5\text{-}4)$$

$$X(i,j,k) = 0(or)1, (i,j,k) \in I \times J \times K \qquad (5\text{-}5)$$

If $X(i_1,j,k_1)=1$ and $X(i_2,j,k_2)=1$ then $i_1 \neq i_2$ & $k_1 \neq k_2$ i.e., Same schedule j is assigned to different persons i_1 and i_2 at different facilities k_1 and k_2. (5-6)

Where i={1,2,3,…..,n}, j={1,2,3,…..,m} and k={1,2,3,…….,p} respectively are the sets of persons, schedules and facilities. It is to be noted that (5-2) to (5-6) defined the constraints of the problem, whose objective function is MIN

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} C(i,j,k)X(i,j,k)$$

,

X=X (i, j, k) is feasible Schedule if it satisfies all the above conditions

2. Numerical illustration:

The concepts and the algorithms developed will be illustrated by a numerical example for which n=6,m=4,p=2,n^1=4.The cost array C(i,j,k) is given in Table-1, The number of calls at every time period for each schedule i.e., S(t,j) is given in Table-1A and the minimum number of calls made in time period i.e., SB(t) is given in Table-1B.

TABLE-1

$$C(i,j,1) = \begin{vmatrix} 4 & 7 & 4 & 32 \\ 13 & 5 & 2 & 6 \\ 15 & 7 & 3 & 60 \\ 10 & 1 & 23 & 4 \end{vmatrix} \qquad C(i,j,2) = \begin{vmatrix} 35 & 4 & 12 & 3 \\ 5 & 16 & 3 & 9 \\ 16 & 5 & 72 & 13 \\ 4 & 19 & 5 & 8 \end{vmatrix}$$

TABLE-1A

S	1	2	3	4
1	2	8	7	3
2	7	6	3	8

TABLE-1B

	1	2
SB	25	25

A Feasible schedule of the sources can be represented by an appropriate $n \times m \times p$ indicator array X=[X(i, j, k), X(i, j, k) = 0 or 1 in which X(i, j, k)=1 indicates that the jth schedule is assigned to ith person at facility k and if there is no such schedule it is indicated by X(i, j, k)=0]. 'X' is called a 'solution'.

The indicator X given in Table-2 where $6 \times 4 \times 2$ of X is represented as two matrices for different values of k i.e., 1, 2 is a solution to the numerical example and is represented as follows:

TABLE-2

$$X(i,j,1) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad ; \quad X(i,j,2) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

The representation of the solution X to the problem is that 2nd person gets 3rd schedule using facility 1, 3rd person gets 4th schedule using facility 1, 5th person gets 2nd schedule using facility 1, 6th person gets 1st schedule using facility 2 and 6th person gets 2nd schedule using facility 2. This solution is a feasible solution. The scheduling corresponding to Table-2 can be represented as [(2,3,1), (3,4,1), (5,2,1), ,(6,1,2),(6,2,2)]. For this problem SXT, where SXT (j, i) = α indicates that jth schedule is given to person α at ith facility is given in TABLE-2A; and SV, where SV (i) =β, here β is the total calls at time i, is given in TABLE-2A.

TABLE-2A

SXT	1	2
1		6

2	5	6
3	2	
4	3	

	1	2
SV	28	30

2. CONCEPTS AND DEFINITIONS:

Definition of a pattern:

An indicator three-dimensional array which is associated with a schedule is called a 'pattern'. A Pattern is said to be feasible if X is a solution. The pattern represented in the table-2 is a feasible pattern.

Now V(X) the value of the pattern X is defined as

$$V(X) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} C(i,j,k)X(i,j,k)$$

The value V(X) gives the total cost of the schedule for the solution represented by X. Thus the value of the feasible pattern gives the total cost represented by it. In the algorithm, which is developed in the sequel, a search is made for a feasible pattern with the least value. Each pattern of the solution X is represented by the set of ordered triples [(i,j,k)] for which X(i,j,k)=1, with understanding that the other X(i,j,k)'s are zeros.

The ordered triple set [(2,3,1), (3,4,1), (5,2,1), (6,1,2), (6,2,2)] represents the pattern given in the table-2, which is a feasible solution. The set of ordered triples is [(3,2,1), (5,2,1), (6,3,1), (2,3,2), (6,3,2)] represents the pattern given in the table-2B, which is not feasible solution since 6th person gets same schedule 3, third person and fifth person gets same schedule 2 at same facility k=1 and minimum number of calls i.e., 25 are not reached in second time period.

TABLE-2B

$$X(i,j,1) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} ; \quad X(i,j,2) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

SXT	1	2
1		
2	3(5)	
3	6	2(6)
4		

	1	2
SV	37	21

There are M=$n \times m \times p$ ordered triples in the three-dimensional array X. For convenience these are arranged in ascending order of their corresponding costs and are indexed from 1 to M (Sundara Murthy-1979). Let SN= [1, 2, 3…M] be the set of M indices. Let D be the corresponding array of costs. If a,b $\in$ SN and a<b then D(a) $\leq$ D(b). Also let the arrays R, C, T be the array of row, column and facility indices of the ordered triples represented by SN and DC be the array of cumulative sum of the elements of D. The arrays SN, D, DC, R, C, T for the numerical example are given in the table-3. If p $\in$ SN then (R(p), C(p), T(p)) is the ordered triple and D(a)=C(R(a), C(a), T(a)) is the value of the ordered triple and DC(a)= $\sum_{i=1}^{a} D(i)$

TABLE-3

ALPHABET-TABLE

SN	D	DC	R	C	T
1	1	1	3	4	1
2	1	2	5	2	1
3	1	3	3	1	2
4	2	5	2	3	1
5	3	8	4	3	1
6	3	11	6	3	1
7	3	14	1	4	2
8	3	17	2	3	2
9	4	21	1	1	1
10	4	25	1	3	1
11	4	29	5	4	1
12	4	33	1	2	2
13	4	37	3	3	2
14	4	41	5	1	2
15	5	46	2	2	1
16	5	51	2	1	2
17	5	56	4	2	2
18	5	61	5	3	2
19	6	67	2	4	1
20	6	73	6	1	2
21	7	80	1	2	1
22	7	87	4	2	1
23	8	95	3	1	1
24	8	103	5	4	2
25	9	112	2	4	2

26	9	121	6	2	2
27	10	131	5	1	1
28	10	141	6	4	2
29	11	152	3	3	1
30	12	164	6	4	1
31	12	176	1	3	2
32	13	189	2	1	1
33	13	202	4	4	2
34	15	217	4	1	1
35	16	233	6	2	1
36	16	249	2	2	2
37	16	265	4	1	2
38	19	284	5	2	2
39	21	305	6	1	1
40	21	326	3	2	2
41	23	349	5	3	1
42	31	380	6	3	2
43	32	412	1	4	1
44	32	444	3	4	2
45	35	479	1	1	2
46	42	521	3	2	1
47	60	581	4	4	1
48	72	653	4	3	2

Let us consider $24 \in SN$. It represents the ordered triple (R(24),C(24),T(24))=(5,4,2).Then D(24)=C(5,4,2)=8 and DC(24)=103

Definition of an Alphabet – Table and a word:

Let $L_K = \{a_1,a_2,\ldots\ldots,a_K\}$, $a_i \in SN$ be an ordered sequence of k indices from SN. The pattern represented by the ordered triples whose indices are given by L_k is independent of the order of a_i in the sequence. Hence for uniqueness the indices are arranged in the increasing order such that $a_i \leq a_{i+1}$, i=1, 2… n-1. The set SN is defined as the "Alphabet-Table" with alphabetic order as (1, 2, 3…M) and the ordered sequence L_K is defined as a "word" of length k. A word L_k is called a "Sensible word" if $a_i<a_{i+1}$, for i=1, 2, 3…k-1 and if this condition is not met it is called a "insensible word". A word L_K is said to be feasible if the corresponding pattern X is feasible and same is with the case of infeasible and partial feasible. Therefore a partial feasible word is said to be feasible if $k=n^1$.

A partial word L_k is said to be feasible if the block of words represented by L_K has at least one feasible word or, equivalently the partial pattern represented by L_k should not have any inconsistency.

Any of the letters in SN can occupy the first place in the partial word L^k. Consider $L_{K-1} = (a_1,a_2,\ldots\ldots\ldots..,a_{k-1})$. The alphabet table for the kth position is $SNa_{k-1}=(a_{k-1}+1,a_{k-1}+2\ldots M)$, where SN_P is defined as $SN_P = (p+1,p+2,\ldots\ldots..,M)$. Thus for example consider a word with two letters as $(a_1,a_2)=(1,3)$. Then $SNa_2=SN_3=$ (4, 5, 6…48) is the alphabet for the third position. We concentrate on the set of words of length at most n^1 (for the numerical example it is 5). A leader L_k ($k<n^1$) is said to be

feasible, if the block of words defined by it contains at least one feasible word or equivalently there should not be inconsistency in the partial pattern defined by the partial word.

Lower Bound of a partial word LB (L^K):

A Lower bound LB (L^K) for the values of the block of words represented by L^K can be defined as follows:

$$L\square(L_k) = V(L_k) + \sum_{i=1}^{n^1-k} \square(a_k + \square)$$

$$= V(L_k) + \square\square(a_k + n^1 - \square) - \square\square(a_k)$$

For $L_3 = (1, 2, 4)$

$V(L_3) = 1+1+2=4$

$LB(L_3) = V(L_3) + DC(a_3+5-3) - DC(a_3)$

$= 4+DC(4+5-3) - DC(4)$

$= 4+DC(6) - DC(4)$

$= 4+11-5=10$

Feasibility criterion of a partial word:

A recursive algorithm is developed for checking the feasibility of a partial word $L_{K+1} = (a_1, a_2, \ldots, a_K, a_{K+1})$ given that L_K is a feasible partial word. We will introduce some more notations which will be useful in the sequel.

RI be an array where RI (i) =α, i∈I represents that ith person gets α schedules, here α >1 for VIP; α =1 for ordinary persons

L be an array where L (i) is the letter in the ith position of a word

SXT be an array where SXT (j, i) = α indicates that jth schedule is given to person α at ith facility

Then for a given partial word $L_K = (a_1, a_2 \ldots a_K)$ the values of the arrays RI,L,SXT as follows.

$RI(R(a_i)) = R(R(a_i)) + 1$ i=1, 2, 3……..K

$L(i) = a_i$, i=1, 2, 3……..K

$SXT(C(a_i), T(a_i)) = R(a_i)$ i=1, 2, 3…….K

For example consider a sensible partial word $L_4 = (1, 2, 4, 20)$ which is feasible. The array RI, L and SXT takes the values represented in the table-4 given below.

TABLE-4

	1	2	3	4	5	6
RI	0	1	1	0	1	1
L	1	2	4	20	-	-

SXT	1	2
1	0	6
2	5	0

3	2	0
4	3	0

The recursive algorithm for checking the feasibility of a partial word L_P is given as follows: In the algorithm first we equate IX=0. At the end if IX=1 then the partial word is feasible, otherwise it is infeasible. For this algorithm we have TR=R (a_{p+1}), TC=C (a_{P+1}) and TT=T (a_{P+1}).

ALGORITHM-1:

STEP1 : IX=0

STEP 2 : IS (VI (TR) = 1) IF YES GOTO 3

IF NO GOTO 4

STEP3 : RI (TR) =RI (TR) +1 GOTO 3A

STEP3A: IS (RI (TR) $\leq$ L) IF YES GOTO 3B

IF NO RI (TR) =RI (TR)-1 GOTO 7

STEP3B: IS VXI (TR, TC) =0 IF YES GOTO 3C

IF NO GOTO 7

STEP3C: IS SXT (TC, TT) =0 IF YES GOTO 6

IF NO GOTO 7

STEP4 : IS (RI (TR) =0) IF YES GOTO 5

IF NO GOTO 7

STEP5 : IS SXT (TC, TT) =0 IF YES GOTO 6

IF NO GOTO 7

STEP6 : IX=1

STEP 7 : STOP

This recursive algorithm will be used as a subroutine in the lexi-search algorithm. We start the algorithm with a very large value, say, 9999 as a trial value of VT. If the value of a feasible word is known, we can as well start with that value as VT. During the search the value of VT is improved. At the end of the search the current value of VT gives the optimal feasible word. We start with the partial word L_1= (a_1) = (1). A partial word $L_p=L_{p-1} * (a_p)$ where * indicates chain form or concatenation. We will calculate the values of V (L_p) and LB (L_p) simultaneously. Then two cases arises (one for branching and other for continuing the search).

1. LB (L_p) < VT. Then we check whether L_p is feasible or not. If it is feasible we proceed to consider a partial word of order (p+1), which represents a sub block of the block of words represented by L_p. If Lp is not feasible then consider the next partial word of order p by taking another letter which succeeds a_p in the p^{th} position. If all the words of order p are exhausted then we consider the next partial word of order (p-1).

2. LB (L_P) $\geq$ VT. In this case we reject the partial word meaning that the block of words with L_p as leader is rejected for not having an optimal word and we also reject all partial words of order p that succeeds L_p.

Now we are in a position to develop lexi search algorithm to find an optimal feasible word.

3. ALGORITHM-2: (LEXI-SEARCH ALGORITHM)

The following algorithm gives an optimal feasible word.

STEP 1 : (Initialization)

The arrays SN, D, DC , R, C, T and values of N, M,P,N^1 ,L,Q, V,I, SV and ST are made available RI, VXI, SXT are initialized to zero. The values I=1, J=0, VT=9999, NZ=N * M * P–I, MAX=NZ-1

STEP 2 : J=J+1

IS (J>MAX) IF YES GOTO 11

IF NO GOTO 3

STEP 3 : L (I) = J

JA=J+N^1-I

IS (I = 1) IF YES V (I) =D (J) GOTO 3B

IF NO GOTO 3A

STEP 3A : V (I) =V (I-1) +D (J)

GOTO 3B

STEP 3B : LB (I) =V (I) +DC (JA)-DC (J)

GOTO 4

STEP 4 : IS (LB (I) $\geq$ VT) IF YES GOTO 11

IF NO GOTO 5

STEP 5 : TR=R (J)

TC=C (J)

TT=T (J)

GOTO 6

STEP 6 : CHECK THE FEASIBILITY OF L (USING ALGORITHM-1)

IS (IX=0) IF YES GOTO 2

IF NO GOTO 7

STEP 7 : IS (I=N^1) IF YES GOTO 10A

IF NO GOTO 8

STEP 8 : L (I) = J

RI (TR) =1

SXT (TC, TT) =TR

SV(TA,I)=SV(TA,I-1)+S(TA,TC);TA=1,2

VXI (TR, TC) =1

GOTO 9

STEP9 : I=I+1

MAX=MAX+1

GOTO 2

STEP10A : IS [SV (TA, I) $\geq$SB (TA)); TA=1, 2] IF YES GOTO 10;

IF NO GOTO 2;

STEP10 : L (I) =J

L (I) IS FULL LENGTH WORD AND IS FEASIBLE.

VT=V (I), record L (I), VT

GOTO 13

STEP11 : IS (I=1) IF YES GOTO 14

IF NO GOTO 12

STEP12 : I=I-1

MAX=MAX+1

GO TO 13

STEP13 : J=L (I)

TR = R (J)

TC = C (J)

TT = T (J)

RI (TR) =0

SXT (TC, TT) =0

SV (TA, I-1) =SV (TA, I)-S (TA, TC); TA=1, 2

VXI (TR, TC) =0

GOTO 2

STEP14 : STOP

END

Now, let us construct a partial word, $L_3=L_2*(7)$, $L_2=(1, 5)$

At this stage, I=I+1=2+1=3, J=6

1) VT=9999, NZ=N*M*P-1=47, MAX=NZ-1=4; goto step 2.

2) Now J=J+1=6+1=7; j<max; goto step 3

3) L_3=7; JA=7+5-3=9

V (L_3) =V (L_2) +D (7) =4+3=7

LB (L_3) =V (L_3) +DC (9)-DC (7) =7+21-14=14; goto step 4

4) LB(L_3)<VT; goto step 5

5) TR=R(7)=1; TC=C(7)=4; TT=T(7)=2; goto step 6

6) Check the feasibility of the partial word L_3 using the recursive algorithm

IX=0; VI (TR) =0; RI (TR) =0; SXT (TC, TT) =0; IX=1; goto step 7

7) I=3 $\neq$ NA; goto step 8

8) RI(TR)=1; SXT(TC,TT)=1; SV(TA,I)=0+8=8; goto first stage

The current value of VT at the end of the search is the value of the optimal feasible word. At the end if VT = 9999 it indicates that there is no feasible solution.

4. SEARCH TABLE

The working details of getting an optimal word, using the above algorithm for the illustrative numerical example is given in the Table-5. The columns (1), (2), (3), (4) and (5) gives the letters in the first, second, third, fourth and fifth places respectively. The corresponding V (I) and LB (I) are indicated in the next two columns. The rows R,C and T gives the row, column and facility indices of the letter. The last column gives the remarks regarding the acceptability of the partial words. In the following table A indicates ACCEPT and R indicates REJECT.

TABLE-5

SEARCH TABLE

SN	1	2	3	4	5	V	LB	R	C	T	REM
1	1					1	8	3	4	1	A
2		2				2	8	5	2	1	A
3			3			3	8	3	1	2	R
4			4			4	10	2	3	1	A
5				5		7	10	4	3	1	R
6				6		7	10	6	3	1	R

7				7		7	10	1	4	2	R
8				8		7	11	2	3	2	R
9				9		8	12	1	1	1	R
10				10		8	12	1	3	1	R
11				11		8	12	5	4	1	R
12				12		8	12	1	2	2	R
13				13		8	12	3	3	2	R
14				14		8	13	5	1	2	R
15				15		9	14	2	2	1	R
16				16		9	14	2	1	2	R
17				17		9	14	4	2	2	R
18				18		9	15	5	3	2	R
19				19		10	16	2	4	1	R
20				20		10	17	6	1	2	A
21					21	17	17	1	2	1	R
22					22	17	17	4	2	1	R
23					23	18	18	3	1	1	R
24					24	18	18	5	4	2	R
25					25	19	19	2	4	2	R
26					26	19	19	6	2	2	A, VT=19
27				21		11	18	1	2	1	R
28				22		11	19	4	2	1	R, =VT
29			5			7	13	4	3	1	A
30				6		10	13	6	3	1	R
31				7		10	13	1	4	2	R
32				8		10	14	2	3	2	R
33				9		11	15	1	1	1	R
34				10		11	15	1	3	1	R
35				11		11	15	5	4	1	R
36				12		11	15	1	2	2	R
37				13		11	15	3	3	2	R
38				14		11	16	5	1	2	R
39				15		12	17	2	2	1	R
40				16		12	17	2	1	2	R
41				17		12	17	4	2	2	R

42				18		12	18	5	3	2	R
43				19		13	19	2	4	1	R, =VT
44			6			5	11	6	3	1	A
45				7		8	11	1	4	2	A
46					8	11	11	2	3	2	R
47					9	12	12	1	1	1	R
48					10	12	12	1	3	1	R
49					11	12	12	5	4	1	R
50					12	12	12	1	2	2	R
51					13	12	12	3	3	2	R
52					14	12	12	5	1	2	R
53					15	13	13	2	2	1	R
54					16	13	13	2	1	2	R
55					17	13	13	4	2	2	R
56					18	13	13	5	3	2	R
57					19	14	14	2	4	1	R
58					20	14	14	6	1	2	R
59					21	15	15	1	2	1	R
60					22	15	15	4	2	1	R
61					23	16	16	3	1	1	R
62					24	16	16	5	4	2	R
63					25	17	17	2	4	2	R
64					26	17	17	6	2	2	A,VT=17
65				8		8	12	2	3	2	A
66					9	12	12	1	1	1	R
67					10	12	12	1	3	1	R
68					11	12	12	5	4	1	R
69					12	12	12	1	2	2	R
70					13	12	12	3	3	2	R
71					14	12	12	5	1	2	R
72					15	13	13	2	2	1	R
73					16	13	13	2	1	2	R
74					17	13	13	4	2	2	R
75					18	13	13	5	3	2	R
76					19	14	14	2	4	1	R

77					20	14	14	6	1	2	A,VT=14
78				9		9	13	1	1	1	A
79					10	13	13	1	3	1	R
80					11	13	13	5	4	1	R
81					12	13	13	1	2	2	R
82					13	13	13	3	3	2	R
83					14	13	13	5	1	2	R
84					15	14	14	2	2	2	R, =VT
85				10		9	13	1	3	1	R
86				11		9	13	5	4	1	R
87				12		9	13	1	2	2	A
88					13	13	13	3	3	2	R
89					14	13	13	5	1	2	R
90					15	14	14	2	2	1	R, =VT
91				13		9	13	3	3	2	R
92				14		9	14	5	1	2	R, =VT
93			7			5	12	1	4	2	A
94				8		8	12	2	3	2	R
95				9		9	13	1	1	1	R
96				10		9	13	1	3	1	R
97				11		9	13	5	4	1	R
98				12		9	13	1	2	2	R
99				13		9	13	3	3	2	R
100				14		9	14	5	1	2	R, =VT
101			8			5	13	2	3	2	A
102				9		9	13	1	1	1	R
103				10		9	13	1	3	1	R
104				11		9	13	5	4	1	R
105				12		9	13	1	2	2	R
106				13		9	13	3	3	2	R
107				14		9	14	5	1	2	R, =VT
108			9			6	14	1	1	1	R, =VT
109		3				2	10	3	1	2	R
110		4				3	12	2	3	1	A
111			5			6	12	4	3	1	R

112			6			6	12	6	3	1	R
113			7			6	13	1	4	2	A
114				8		9	13	2	3	2	R
115				9		10	14	1	1	1	R, =VT
116			8			6	14	2	3	2	R, =VT
117		5				4	13	4	3	1	A
118			6			7	13	6	3	1	R
119			7			7	14	1	4	2	R, =VT
120		6				4	14	6	3	1	R, =VT
121	2					1	10	5	2	1	A
122		3				2	10	3	1	2	A
123			4			4	10	2	3	1	A
124				5		7	10	4	3	1	R
125				6		7	10	6	3	1	R
126				7		7	10	1	4	2	R
127				8		7	11	2	3	2	R
128				9		8	12	1	1	1	R
129				10		8	12	1	3	1	R
130				11		8	12	5	4	1	R
131				12		8	12	1	2	2	R
132				13		8	12	3	3	2	R
133				14		8	13	5	1	2	R
134				15		9	14	2	2	1	R, =VT
135			5			5	11	4	3	1	A
136				6		8	11	6	3	1	R
137				7		8	11	1	4	2	R
138				8		8	12	2	3	2	R
139				9		9	13	1	1	1	R
140				10		9	13	1	3	1	R
141				11		9	13	5	4	1	R
142				12		9	13	1	2	2	R
143				13		9	13	3	3	2	R
144				14		9	14	5	1	2	R, =VT
145			6			5	11	6	3	1	A
146				7		8	11	1	4	2	A

147					8	11	11	2	3	2	R
148					9	12	12	1	1	1	R
149					10	12	12	1	3	1	R
150					11	12	12	5	4	1	R
151					12	12	12	1	2	2	R
152					13	12	12	3	3	2	R
153					14	12	12	5	1	2	R
154					15	13	13	2	2	1	R
155					16	13	13	2	1	2	R
156					17	13	13	4	2	2	R
157					18	13	13	5	3	2	R
158					19	14	14	2	4	1	R, =VT
159				8		8	12	2	3	2	A
160					9	12	12	1	1	1	R
161					10	12	12	1	3	1	R
162					11	12	12	5	4	2	R
163					12	12	12	1	2	2	R
164					13	12	12	3	3	2	R
165					14	12	12	5	1	2	R
166					15	13	13	2	2	1	R
167					16	13	13	2	1	2	R
168					17	13	13	4	2	2	R
169					18	13	13	5	3	2	R
170					19	14	14	2	4	1	R, =VT
171				9		12	16	1	1	1	R, >VT
172			7			5	12	1	4	2	A
173				8		8	12	2	3	2	R
174				9		9	13	1	1	1	R
175				10		9	13	1	3	1	R
176				11		9	13	5	4	1	R
177				12		9	13	1	2	2	R
178				13		9	13	3	3	2	R
179				14		9	14	5	1	2	R, =VT
180			8			5	13	2	3	2	A
181				9		9	13	1	1	1	R

182				10		9	13	1	3	1	R
183				11		9	13	5	4	1	R
184				12		9	13	1	2	2	R
185				13		9	13	3	3	2	R
186				14		9	14	5	1	2	R, =VT
187			9			6	14	1	1	1	R, =VT
188		4				3	12	2	3	1	A
189			5			6	12	4	3	1	R
190			6			6	12	6	3	1	R
191			7			6	13	1	4	2	A
192				8		9	13	2	3	2	R
193				9		10	14	1	1	1	R, =VT
194			8			6	14	2	3	2	R, =VT
195		5				4	13	4	3	1	A
196			6			7	13	6	3	1	R
197			7			7	14	1	4	2	R, =VT
198		6				4	14	6	3	1	R, =VT
199	3					1	12	3	1	2	A
200		4				3	12	2	3	1	A
201			5			6	12	4	3	1	R
202			6			6	12	6	3	1	R
203			7			6	13	1	4	2	A
204				8		9	13	2	3	2	R
205				9		10	14	1	1	1	R, =VT
206			8			6	14	2	3	2	R, =VT
207		5				4	13	4	3	1	A
208			6			7	13	6	3	1	R
209			7			7	14	1	4	2	R, =VT
210		6				4	14	6	3	1	R, =VT
211	4					2	14	2	3	1	R, =VT

At the end of the search the current value of VT is 14 and it is the value of the optimal feasible word, $L^5 = (1, 2, 6, 8, 20)$.The array RI, L and SXT takes the values represented in the table-6 given below. It is given in the 77^{th} row of the search table. The pattern represented by the above optimal feasible word is represented in the following table-7.

TABLE-6

	1	2	3	4	5	6
RI		1	1		1	2
L	1	2	6	8	20	

SXT	1	2
1		6
2	5	
3	6	2
4	3	

TABLE-7

$$X(i,j,1)=\begin{bmatrix}0&0&0&0\\0&0&0&0\\0&0&0&1\\0&0&0&0\\0&1&0&0\\0&0&1&0\end{bmatrix} \quad ; \quad X(i,j,2)=\begin{bmatrix}0&0&0&0\\0&0&1&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\\1&0&0&0\end{bmatrix}$$

	1	2
SV	27	27

The schedule represented by the above pattern is[(3,4,1),(5,2,1), (6,3,1),),(2,3,2),(6,1,2)] where 3rd person gets 4th schedule using facility 1, 5th person gets 2nd schedule using facility 1, 6th person gets 3rd schedule using facility 1, 2nd person gets 3rd schedule using facility 2 and 6th person gets 1st schedule using facility 2 .This solution is a feasible solution

The Schedule is:

$$\begin{matrix}person\\schedule\\facility\end{matrix}\begin{pmatrix}2&3&5&6&6\\3&4&2&3&1\\2&1&1&1&2\end{pmatrix}$$

5. COMPUTATIONAL EXPERIENCE:

A Computer program for the above algorithm is written in C language and is tested on the system ACER. Random numbers are used to construct the cost matrix. The following table-8 gives the list of the problems tried along with the average CPU time in seconds required for solving them.

In the table AT represents the CPU time to construct the alphabet-table and ET represents the CPU time taken for the search of a feasible word. The time is represented in seconds. In the table-8 ‘n’ is the number of persons, m is the number of schedules, p is the number of facilities, n^1 is restricted number of persons .

Further experiments are carried out on a ACER system and by generating the three different classes of random data sets, where the three types of data sets are defined as follows:

Type 1: C (i, j, k) are uniformly random in [1,100]

Type 2: a) C (i, j, k) are uniformly random in [1,100]

b) VT=0.85VT

Type 3: a) C (i, j, k) are uniformly random in [1,100]

b) Max = (nxmxp)/3

And the results are tabulated in Table. For each type, three data sets are tested. It is seen that time required for the search (ET) of the optimal solution is fairly less.

TABLE-8

Problem dimensions				No. Of Prob's	AT	Total time taken(ET)								
						TYPE 1			TYPE 2			TYPE 3		
N	m	p	n^1			min	max	Avg	min	max	avg	min	max	avg
7	15	4	6	3	0.65	3.01	3.41	3.19	3.00	3.32	3.11	2.39	2.69	2.49
18	12	6	5	3	0.76	3.46	3.93	3.7	3.23	3.71	3.59	3.01	3.24	3.12
38	20	7	6	3	1.21	6.78	6.83	6.8	5.17	5.91	5.46	4.28	4.68	4.47

In the above table it can be notice that the average CPU times for Type 1, Type 2 and Type 3 are in decreasing order since in Type 2 the search is made around 0.85VT and in Type 3 the search is in 1/3 of the alphabet table. But in all the cases we are getting the same optimal solution which may be a coincidence.

Aggarwal,. V	1983	:	The assignment problem under categorized jobs, European Journal of Operational Research 14,193-195.
Barr, R.S., Glover, F. & Klingman, D	1977	:	The Alternating Basis Algorithm For Assignment Problem, Math. Prog.,13. 1-13.
Bertsekas, D.P.	1981	:	A New Algorithm For The Assignment Problem, Math. Prog. 21. 152-171.
Bhatia. H.L.	1977	:	Time minimizing assignment problem, Systems and Cybernatics in management 6, 75-83.
Demaio, A and Roveda, C	1971	:	An all 0-1 algorithm for a certain class of Transportation Problems, Opns. Res. 19, pp.1406-1418.
Geetha, S. & Nair, K. P.	1993	:	A Variation Of The Assignment Problem, Euro. J. Of Or, 68, 422-426.
Guignard, M. & Rosenwein, M.	1989	:	An Improved Dual-Based Algorithm For The Generalized Assignment Problem, Ops. Res., 37,658-663.
Haley, K. B.	1962	:	The Solid Transportation Problem, Ops. Res. 10, 448-463.
Hung, M.S. & Rom, W.O.	1980	:	Solving The Assignment Problem By Relaxation. Ops. Res., 18, 969-982.
Jack Elzinga and Vemuganti, R.R	1976	:	Optimal Scheduling of a sales force OPSEARCH 13(2), 79-92.
Jornsten, K . & Nasberg,	1986	:	A New Lagrangean Relaxation Approach To the Generalized

M.			Assignment Problem, Euro. J. Of Or, 27,313-323.
Klein, M & Takamori, H.	1972	:	Parallel Line Assignment Problem, Mgt. Sci., 19, 1-10.
Kuhn, H. W.	1955	:	The Hungarian Method For The Assignment Problem, Naval Res. Log. Qtly. 2, 83-97.
Martello, S. & Toth, P	1988	:	A New Algorithm For The 0-1 Knapsack Problem, Mgt. Sci., 34, 633-644.
Pierskalla, W.P.	1968	:	The Multi Dimensional Assignment problems. Opns. Res., 16, 442 – 50
Ravi Kumar, M	1994	:	Data Guided algorithms in Combinatorial Optimization, Ph.D. Thesis, Osmania University, Hyderabad, A.P., India
Ross, G.T and Soland, R.M	1975	:	A Branch and Bound algorithm for Generalized Assignment Problem - Mathematical Programming, pp 91-103.
Subramanyam, Y.V	1980	:	Scheduling Transportation and Allied Combinatorial programming problems, Ph.D., Thesis, REC, Kakatiya University, Warangal
Sundara Murthy, M	1979	:	Combinatorial Programming - A Pattern Recognition Approach. PhD, Thesis REC, Warangal, India