www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 2 Issue 7 (July 2013), Page No. 2177-2185

NETWORK OF P- PATH MINIMUM CONSTRAINED CONNECTIVITY FROM CITIES TO HEAD QUARTER

Soma Sekhar Srinivas V.K[1] Suresh Babu C[2] Purusotham S[3] Sundara Murthy M[4]
1, 2, 4 Dept. of Mathematics, S. V. University, Tirupati, Andhra Pradesh, India.
3. Asst. Professor, Statistics and Operations Research Division,VIT University, Vellore, Tamil Nadu, India
Email:somasekharsrinivas@gmail.com, suresh8044@gmail.com, drpurusotham.or@gmail.com, profmurthy@gmail.com

Abstract:
Many Combinatorial programming problems are NP-hard (Non Linear Polynomial), and we consider one of them called P-path minimum distance connectivity from head quarter to the cities. Let there be n cities and the distance matrix D (i, j, k) is given from i^{th} city to j^{th} city using k^{th} facility. There can be an individual factor which influences the distances/cost and that factor is represented as a facility k. We consider m<n cities are in cluster and to connect all the cities in subgroup (cluster) from others by using same facility k. The problem is to find minimum distance to connect all the cities from head quarter (say 1) threw p-paths subject to the above considerations. For this problem we developed a Pattern Recognition Technique based Lexi Search Algorithm, we programmed the proposed algorithm using C. we compared with the existing models and conclude that it suggested for solving the higher dimensional problems.
Keywords: Lexi Search Algorithm, Pattern Recognition Technique, Partial word, Pattern.

1. Introduction:

In recent years the development of networks in the area of telecommunication and computer has gained much importance. One of the main goals in the design process is to reach total connectivity at minimum distance/cost. The total connections are in some paths (say P). Similar problems arise in the planning of road maps, integrated circuits. The technical restriction that the number of connections at a node is bounded is modelled by introducing constraints that bound the node degrees. Garey al all [2] proved that the resulting degree-constrained minimum. In this paper we study a variation of Minimum spanning models. For this we developed a Lexi- algorithm based on the "Pattern Recognition Technique" to solve this problem which takes care of simple combinatorial structure of the problem and computational results are reported.

Some of the researchers studied variations in the Minimum Spanning Tree (MST) problems. They are Pop, P.C [6], Karger [3] found a linear time randomized algorithm based on a combination of Boruvka"s algorithm and the reverse-delete algorithm. The problem can be solved deterministically in linear by Chazelle [1]. Its running time is o (m, α (m, n)) where function α grows extremely slowly. Thus Chazelle's algorithm takes very close to linear time. Seth Pette [4, 5] has found a probably optimal deterministic comparison-based minimum spanning tree algorithm. Sobhan Babu [8] studied a variation of spanning models using pattern recognition technique [9]. Let there be N cities to be connected to the Headquarter City {1}. There is an individual factor which influences the distances/cost and that factor is represented as a facility K. If the city j_1 and j_2 different cities are connected from city i_1 then k_1 and k_2 should be same. Suresh Babu [10] studied another variation of spanning models, which is reverse case of [11]. The problem is to find optimal solution for all the cities connected to head quarter {1} with minimum distance by using k facilities.

2. Pattern Recognition Technique based Lexicographic Search:

Lexicographic Search Approach is a systematized Branch and Bound approach, developed by Pandit in the context of solving of loading problem in 1962. In principle, it is essentially similar to the Branch and Bound method as adopted by little et.al-1963. This approach has been found to be productive in many of the Combinatorial Programming Problems. It is significance mentioning that. Branch and Bound can be viewed as a particular case of Lexicographic Search approach {Pandit-1965]. The name Lexicographic Search itself suggests that, the search for an optimal solution is done in a systematic manner, just as one searches for the meaning of a word in a dictionary and it is derived from Lexicography the science of effective storage and retrieval of information. This approach is based on the following grounds [Pandit -1963].

(i) It is possible to list all the solutions or related configurations in a structural hierarchy which also reflects a hierarchical ordering of the corresponding values of these configurations.

(ii) Effective bounds can be set to the values of the objective function, when structural combinatorial restraintsareplaced on the Allowable configurations.

.The Pattern-recognition technique can be described as follows.

"A unique pattern is associated with each solution of a problem. Partial pattern defines a partial solution. An alphabet-table is defined with the help of which the words, representing the pattern are listed in a Lexicographic order. During the search for an optimal word, when a partial word is considered, first bounds are calculated and then the partial words for which the value is less than the trail value are checked for the feasibility"

3. Problem Description:

In this problem we have studied a variation of minimum spanning models called P-path minimum distance/cost connectivity from n-1 cities to head quarter city(1). Let there be n cities and the distance matrix D (i, j) is given from i^{th} city to j^{th} city. There can be an individual factor which influences the distances/cost and that factor is represented as a facility k. Let there are set of n cities in N = {1, 2. . . n} and the set of k facilities in K = {1,2, . . ., q}. Then the three dimensional problem is: Let D (i, j, k) be the distance/cost from i^{th} city to j^{th} city using k^{th} facility where i, j Є N and k Є K. Let there be m cities, M= (1, 2…m), M<N. We consider these m cities are in cluster. The restriction of the problem is to use the same facility for connecting all the cities in subgroup (cluster) to others.

Let '{1}' be head quarter city. We want to connection all the (n-1) cities to head quarter city by P-paths. Each city connected to head quarter city {1} either directly or indirectly. The objective of the problem is to find minimum total distance to connecting all the n-1 cities to head quarter {1} under the considerations. For this we developed an algorithm called as Lexi-Search algorithm based on the pattern recognition Technique and it is illustrated with a suitable numerical example for three paths.

4. Mathematical Formulation:

Minimize $Z(X) = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{q} D(i,j,k), X(i,j,k) \quad ---1$

Subjected to the constraints

$\sum_{s=1}^{p} X(1, m_{s1}) = P$ ---------- (2)

$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{q} X(i,j,k) = n-1$ ------- (3)

Let $\alpha_{s1},\alpha_{s2},\alpha_{s3}\ldots\alpha_{sns}$ are n_s cities in s^{th} path ---------(4)

then $\sum_{s=1}^{ns-1} X(\alpha_{sj}\ \alpha_{sj+1}) = n_{s-1}$

(s=1,2,3…p)

$\sum_{i=1}^{p} n_i = n-1$ ------- (5)

Let $i_1, i_2 Є M$,

If $X(i_1,j_1,k_1)=X(i_2,j_2,k_2)=1$. Then $k_1=k_2$ --------- (6)

X(I,j,k)= 0 or 1 ------- (7)

7. Numerical Illustration:

The concept and algorithm developed will be illustrated by a numerical example for which total number of cities N = {1, 2, 3, 4, 5, 6, 7}. Among them the cities 3, 4&6 are taken as separate cluster says **b** = {3, 4, 6}. All the cities in M are connected to other cities should use same facility. Then the distance matrices D (i, j, k) are as follows.

TABLE-1

D(i,j,1)=

∞	∞	∞	∞	∞	∞	∞
28	∞	01	15	07	25	35
27	03	∞	13	03	19	16
01	10	20	∞	12	26	32
29	14	05	22	∞	34	17
24	02	18	09	21	∞	33
08	06	23	04	28	30	∞

TABLE-2

D(i,j,2)=

∞	∞	∞	∞	∞	∞	∞
01	∞	05	18	02	31	09
22	19	∞	13	21	26	16
28	15	17	∞	32	04	33
02	25	08	11	∞	34	23
29	06	27	14	20	∞	07
23	24	03	05	30	36	∞

In the above tables (Table 1&2) the value ∞ indicates the non-connectivity of the cities directly and along with the elements in the diagonal. Here the entire D (i, j, k)'s taken as non-negative integers it can be easily seen that this is not a necessary condition and the distances can be any positive quantities. In table 2, suppose D (3, 4, 2) =13 means the distance of the connecting the city 3 to 4 by using facility 2 is 13.

TABLE-3

B=

1	2	3	4	5	6	7
0	0	1	1	0	1	0

In the above numerical example given in Table – 3, B (i) = 1 represent that the the city i belongs to cluster M. Otherwise B (i) = 0. Suppose B (3) =1 means city 3 is in cluster **b**.

8. CONCEPTS AND DEFINITION:

8.1. **Definition of Pattern**: An indicator three-dimensional array which is associated the assignment is called a "pattern". A pattern is said to be feasible if X is solution.

$$V(X) = \sum_{i\in N}\sum_{j\in N}\sum_{k\in K} D(i,j,k)\ X(i,j,k)$$

The pattern represented is a feasible pattern. The value V(X) gives the total time 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 X(i, j, k)=1, which understanding that the other X(i, j, k) 's are zeros.

8.2. Alphabet Table:

There is M= n×n×k ordered triples in the three-dimensional array X. For convenience these are arranged in ascending order of their corresponding cost 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 distance. If a, b ∈ SN and a < b then D (a) $\leq$ D (b). Also let the arrays R, C, K be the array of row, column and facility indices of the ordered triples represented by SN and CD be the array of cumulative sum of the elements of V. The arrays SN, D, CD, R, C, and K for the numerical example are given in the table-4. If p ∈ SN then (R(p),C(p),K(p)) is the ordered triple and D(a) = T(R(a), C(a), K(a)) is the value of the ordered triple and CD (a) $= \sum_{i=1}^{a} D(i)$

TABLE-4

ALPHABET TABLE

SN	D	CD	R	C	K
1	1	1	2	3	1
2	1	2	4	1	1
3	1	3	2	1	2
4	2	5	6	2	1
5	2	7	2	5	2
6	2	9	5	1	2
7	3	12	3	2	1
8	3	15	3	5	1
9	3	18	7	3	2
10	4	22	7	4	1
11	4	26	4	6	2
12	5	31	5	3	1
13	5	36	2	3	2
14	5	41	7	4	2
15	6	47	7	2	1
16	6	53	6	2	2
17	7	60	2	5	1
18	7	67	6	7	2
19	8	75	7	1	1
20	8	83	5	3	2
21	9	92	6	4	1
22	9	101	2	7	2
23	10	111	4	2	1
24	11	122	5	4	2
25	12	134	4	5	1
26	13	147	3	4	1
27	13	160	3	4	2
28	*14*	*174*	*5*	*2*	*1*
29	14	188	6	4	2
30	15	203	2	4	1
31	15	218	4	2	2
32	16	234	3	7	1
33	16	250	3	7	2
34	17	267	5	7	1
35	17	284	4	3	2
36	18	302	6	3	1
37	18	320	2	4	2
38	19	339	3	6	1
39	19	358	3	2	2
40	20	378	4	3	1
41	20	398	6	5	2
42	21	419	6	5	1
43	21	440	3	5	2
44	22	462	5	4	1
45	22	484	3	1	2
46	23	507	7	3	1
47	23	530	5	7	2
48	23	553	7	1	2
49	24	577	6	1	1
50	24	601	7	2	2
51	25	626	2	6	1
52	25	651	5	2	2
53	26	677	4	6	1
54	26	703	3	6	2
55	27	730	3	1	1
56	27	757	6	3	2
57	28	785	2	1	1
58	28	813	7	5	1
59	28	841	4	1	2
60	29	870	5	1	1
61	29	899	6	1	2
62	30	929	7	6	1
63	30	959	7	5	2
64	31	990	2	6	2
65	32	1022	4	7	1
66	32	1054	4	5	2
67	33	1087	6	7	1
68	33	1120	4	7	2
69	34	1154	5	6	1
70	34	1188	5	6	2
71	35	1223	2	7	1
72	36	1259	7	6	2

From the above table – 8, Let us consider 11 ∈ SN. It represents the ordered pair D ((R (11), C (11), and K (11)) = (4, 6, 2). Then D (11) = D (4, 6, 2) = 4 and CD (11) = 26.

8.3. Definition of Alphabet Table and a Word:

Let SN= (1, 2...) be the set of indices; let D be an array of distances in between respective cities. CD is an array of cumulative sum of elements in D. Let arrays R, C and K be respectively, the row, column and facility indices of the ordered triples. Let $L_k = \{a_1, a_2, \ldots a_k\}$, a_i €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 ...k-1. The set SN is defined as the "Alphabet-Table" with alphabet order as (1, 2... n) and the ordered sequence L_k is defined as a "word"of length k. A word L_k is called a "sensible word". If $a_i \leq a_{i+1}$, for i=1, 2 ...k-1 and if this condition is not met it is called a "insensible word". A word L_k has at least one feasible word or, equivalently the partial pattern represented by L_k has at least one feasible word or equivalently the partial pattern represented by 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. Our interest is only in set of words of length at most equation. Since the words of length greater then n are necessarily infeasible, as any feasible pattern can have only n unit entries in it. If k≤n, L_k is called a partial word and if k=n, it is a full length word or simply a word. A partial word L_k represents, a block of words with L_k as a leader i.e., as it's first k letters. A partial word L_k as a leader

i.e. as its first k letters. A leader is said to be feasible, if the block of word, defined by it has at least one feasible word.

8.4. Value of the Word:

The value of the partial word L_k , $V(L_k)$ is defined recursively as $V(L_k)=V(L_{k-1})+D(a_i)$ with $V(L_0)=0$, where $D(a_i)$ is the distance array arranged such that $D(a_i)\leq D(ai_{+1})$.The values of V (L_k) and V(X) of the pattern X(represented by L_k, (Sundara Murthy – 1979)).will be the same.

Consider the partial word L_4 = (1, 2, 4, 6). Then V (L_4) =1+1+2+2=6.

8.5. 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= $(a_i, a_2 ...a_k)$ can be defined as follows.

$$LB(L_k) = V(L_K) + \sum_{j=1}^{n-1-k} D(a_{k+1})$$

$$= V(L_k) + CD(a_k + n - 1 - k) - CD(a_k)$$

Consider the partial word L_4 = (1, 2, 4, 6)

and V (L_4) =1+1+2+2=6

Then LB (L_k) =V (L_k) +CD $(a_4+n-1-k)$-CD (a_k)

= 6+CD (6+6-4)-CD (6)

= 6+CD (8)-CD (6)

=6+15-09 = 12

8.6Feasibility criterion of partial word:

An algorithm was developed, in order to check the feasibility of a partial word L_{k+1} = {$a_1, a_2 ...ak, ak_{+1}$} given that L_k is a feasible word. We will introduce some more notations which will be useful in the sequel.

- **IR** be an array where IR(i)=1, i ЄN indicates that the i^{th} city is connected to some city j otherwise IR(i)=0
- **IC** be an array where IC (i) = 1, iЄN indicates that the i^{th} city is connected from some city j otherwise IC(i)=0
- **SW** be an array where SW (i) = j indicates that the i^{th} city is connected to city j otherwise, SW (i) = 0
- **L** be an array where L[i] = α_i, i∈ N is the letter in the i^{th} position of a word
- **B** be an array, where B (i) =1, indicates that the i^{th} city is belongs to M cluster otherwise B(i)=0.

Then the values of the arrays IR, IC, SW, L and B are as follows

- IC (C (a_i)) = 1, i = 1, 2, - - - - - , k and IC (j) = 0 for other elements of j
- IR (R (a_i)) = 1, i = 1, 2, - - - - - , k and IR (j) = 0 for other elements of j
- SW(R (a_i)) = C (a_i), i = 1, 2, - - - , k and SW (j) = 0 for other elements of j
- L (i) = a_i, i = 1, 2, - - - - -, k, and L(j) = 0, for other elements of j
- For example consider a sensible partial word L_4 = (1, 3, 4, 6) which is feasible. The array IR, IC, L, SW takes the values represented in table - 5 given below.

TABLE-5

	1	2	3	4	5	6	7
L	1	2	4	6			
IR		1		1	1	1	
IC	2	1	1				
SW		3		1	1	2	

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_i), TC=C(a_i),TK=K(a_i)

9. ALGORITHMS:

ALGORITHM 1: (Algorithm for feasible checking)

STEP-0 IX=0 GO TO 1
STEP-1 IS (TC) =HC, IF YES {NP=NP+1} GO TO 2;
IF NO GO TO 3;
STEP-2 IS(IC [HC] <P) , IF YES GO TO 4;
IF NO GO TO 16;
STEP-3 IS IC (TC) =1 IF YES GO TO 16;
IF NO GO TO 4;
STEP-4 IS IR (TR) =1 IF YES GO TO 16;
IF NO GO TO 5;
STEP-5 Z_1=P-NP;
RP=N-1-I; IF YES GO TO 6;
IS RP>=Z_1 IF NO GO TO 16;
STEP-6 W = TC GOTO 7;
STEP-7 IS SW (W) = 0 IF YES GO TO10;
IF NOGO TO 8;
STEP-8 IS W=TR IF YES GO TO 16;
IF NO GO TO 9;
STEP-9 W=SW (W) GO TO 7;
STEP-10 IS b (TR) =1 IF YES GOTO11;
IF NO GOTO 15;
STEP: 11 IS (NB) =O IF YES GOTO 12;
IF NO GOTO 13;
STEP: 12 B=b (TR),
F (B) =TK GOTO14;
STEP: 13 IS F (B) =TK IF YES GO TO 14;
IF NO GO TO 16;
STEP: 14 NB=NB+1 GO TO 15;
STEP: 15 IX=1
STEP: 16 STOP

$L_k = L_{k-1}$ * Where * indicates chain formulation. We will calculate the values of V (Lp) and LB (Lp) simultaneously. Then two situations arises one for branching and other for continuing the search.

LB (L_p) VT. Then we check whether L_p is feasible or not. If it is feasible we processed to consider a partial word of under (P+1). Which represents a sub-block of the block of Words represented by L_p

1. LB (Lp) ≥VT. In this case we reject the partial word L_p. We reject the block of word with L_p as leader as not having optimum feasible solution and also reject all partial words of order p that succeeds L_p.

ALGORITHM 2: (Lexi -Search Algorithm)

STEP-1: (initialization):

The arrays SN, D, CD, R, C, K, P, M, N and B are made available IR, IC, F, L, V, LB and SW are initialized to zero. The values I=1, J=0, VT =999, MAX=NZ-1.

STEP-2 : J=J+1
NP=IC (HC)
MAX=n*n IF YES GO TO 14;
IS (J≥MAX) IF NO GO TO 3;

STEP-3

: L (I) =J
TR=R (J)
TC=C (J)
TK=K (J) GOTO 4;

STEP-4 : V (I) =V (I-1) +D (J)
LB (I) =V (I+CD (J+N-1-I)-CD (J) GOTO 5;

STEP-5 : IS (LB (I)>=VT) IF YES GOTO 17
IF NO GOTO 6;

STEP-6 :(CHECK THE FEASIBILLITY OF USING AGORITHM-1)
IS (IX=0) IF YES GO TO 2;
IF NO GO TO 7;

STEP 7 : IS (I = N-1) IF YES GO TO 10;
IF NO GO TO 8;

STEP 8 : L (I) = J
IR (TR) = 1
SW (TR) =TC
IS TC = HC IF YES {IC (TC) =IC (TC) +1}, GO TO9;
IF NO {IC (TC) =1}, GOTO 9;

STEP: 9 I=I+1, GO TO 2;

STEP: 10 IS (H(C) =T(C)) IF YES GO TO 11;
IF NO GO TO 12;

STEP: 11 IS (IC (HC) =P-1) IF YES GO TO 13;
IF NO GO TO 2;

STEP: 12 IS(IC (HC) =P) IF YES GO TO 13;
IF NO GO TO 2;

STEP 13 :VT = V (I),
L (I) = J;
IS (b (TR) =1 IF YES {NB=NB-1}, GO TO14
IF NO GO TO 14;

STEP-14: I=I-1 GO TO 15;

STEP-15: J=L (I)
TR=R (J)
TC=C (J)
IR (TR) =0
SW (TR) =0
L (I+1) =0
IS TC = HC IF YES {IC (HC) =IC (HC)-1}, GOTO16;
IF NO {IC (TC) =0}, IF NO GOTO 16;

STEP: 16 IS (b (TR) =1) IF YES {NB=NB-1}, GO TO 2;
IF NO GO TO 2;

STEP 17 : IS (I = 1) IF YES GO TO 18;
IF NO GO TO 14;

STEP-18 STOP

10. SEARCH TABLE:

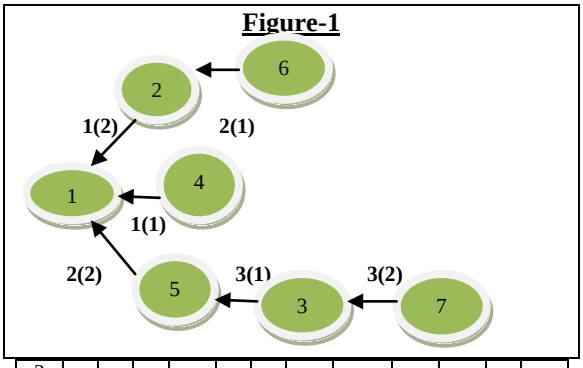
The working detail of getting an optimal word using the above algorithm for the illustrative numerical example is given in the following table-6. The columns named (1), (2), (3).....give the letters in the first second thirds and so on places respectively. The columns R, C and K give the row, column and facility indicates the letter. The last columns give the remarks regarding the acceptability of the partial word. In the following table the letter "A" indicates accept and "R" indicates reject.

TABLE-6

S	1	2	3	4	5	6	V	LB	R	C	K	R
N												EM
1	1						1	09	2	3	1	A
2		2					2	09	4	1	1	A
3			3				3	09	2	1	2	R
4			4				4	11	6	2	1	A
5				5			6	11	2	5	2	R
6				6			6	12	5	1	2	A
7					7		9	12	3	2	1	R
8					8		9	12	3	5	1	A
9						9	12	12	7	3	2	R
10						10	13	13	7	4	1	R
11						11	13	13	4	6	2	R
12						12	14	14	5	3	1	R
13						13	14	14	2	3	2	R
14						14	14	14	7	4	2	R
15						15	15	15	7	2	1	R
16						16	15	15	6	2	2	R
17						17	16	16	2	5	1	R
18						18	16	16	6	7	2	R
19						19	17	17 VT	7	1	1	A
20					9		9	13	7	3	2	R
21					10		10	14	7	4	1	A
22						11	14	14	4	6	2	R
23						12	15	15	5	3	1	R
24						13	15	15	2	3	2	R
25						14	15	15	7	4	2	R
26						15	16	16	7	2	1	R
27						16	16	16	6	2	2	R
28						17	17	17	2	5	1	R,=VT
29					11		10	15	4	6	2	R
30					12		11	16	5	3	1	R
31					13		11	16	2	3	2	R
32					14		11	17	7	4	2	R,=VT
33				7			07	13	3	2	1	R
34				8			07	14	3	5	1	A
35					9		10	14	7	3	2	R

36					10		11	15	7	4	1	R
37					11		11	16	4	6	2	R
38					12		12	17	5	3	1	R,=VT
39				9			07	15	7	3	2	R
40				10			08	17	7	4	1	R,=VT
41			5				04	12	2	5	2	R
42			6				04	13	5	1	2	A
43				7			07	13	3	2	1	A
44					8		10	13	3	5	1	R
45					9		10	14	7	3	2	R
46					10		11	15	7	4	1	A
47						11	15	15	4	6	2	R
48						12	16	16	5	3	1	R
49						13	16	16	2	3	2	R
50						14	16	16	7	4	2	R
51						15	17	17	7	2	1	R,=VT
52					11		11	16	4	6	2	R
53					12		12	17	5	3	1	R=VT
54				8			07	14	3	5	1	A
55					9		10	14	7	3	2	R
56					10		11	15	7	4	1	A
57						11	15	15	4	6	2	R
58						12	16	16	5	3	1	R
59						13	16	16	2	3	2	R
60						14	16	16	7	4	2	R
61						15	17	17	7	2	1	R,=VT
62					11		11	16	4	6	2	R
63					12		12	17	2	3	2	R,=VT
64				9			7	15	7	3	2	R
65				10			8	17	7	4	1	R,=VT
66			7				5	15	3	2	1	A
67				8			8	15	3	5	1	R
68				9			8	16	7	3	2	R
69				10			9	18	7	4	1	R,>VT
70			8				5	16	3	5	1	A
71				9			8	16	7	3	2	R
72				10			9	18	7	4	1	R,>VT
73			9				5	18	7	3	2	R,>VT
74		3					2	11	2	1	2	R
75		4					3	13	6	2	1	A
76			5				5	13	2	5	2	R
77			6				5	14	5	1	2	A
78				7			8	14	3	2	1	R
79				8			8	15	3	5	1	A
80					9		11	15	7	3	2	R
81					10		12	16	7	4	1	R
82					11		12	17	4	6	2	R,=VT
83				9			8	16	7	3	2	R
84				10			9	18	7	4	1	R,>VT
85			7				6	16	3	2	1	R
86			8				6	17	3	5	1	R,=VT
87		5					3	14	2	5	2	R
88		6					3	16	5	1	2	A
89			7				6	16	3	2	1	A
90				8			9	16	3	5	1	R
91				9			9	17	7	3	2	R,=VT
9			8				6	17	3	5	1	R

Figure-1



2												,= VT
93		7					4	18	3	2	1	R ,> VT
94	2						1	11	4	1	1	A
95		3					2	11	2	1	2	A
96			4				4	11	6	2	1	A
97				5			6	11	2	5	2	R
98				6			6	12	5	1	2	A
99					7		9	12	3	2	1	R
100					8		9	12	3	5	1	A
101						9	12	12 VT	7	3	2	A
102					9		9	13	7	3	2	R ,> VT
103				7			7	13	3	2	1	R ,> VT
104			5				4	12	2	5	2	R
105			6				4	13	5	1	2	R ,> VT
106		4					3	13	6	2	1	R ,> VT
107	3						1	13	2	1	2	R ,> VT

The above table – 6, gives optimal solution of the taken numerical example and the partial word is L_6 = (2,3,4,6,8,9) is a feasible partial word. The end of search table optimal solution VT is 12. It is in the 101th row of the search table. For this partial word the array IR, IC, SW, L are given in the following Table – 7.

Table – 7

	1	2	3	4	5	6	7
L	2	3	4	6	8	9	
IR		1	1	1	1	1	1
IC	3	1	1		1		
SW		1	5	1	1	2	3

At the end of the search table the optimum solution value of VT is **12** and is the value of optimal feasible word L_6 = (2, 3, 4, 6, 8, 9). Then the following figure – 1 represents the optimal solution to the Network.

From the above figure-1, the city 1 connected by city 2 at fecility 2, city 2connected by city 6 at facility 1,city 1 is connected by city 4 by fecility 1,city 1 is connected by city 5 by facility 2. In similar way all the cities are connected to headquarter 1 directly or indirectly. More over all the cities in **b** are connected by same (1) facility. Hence the solution of the above figure-1as follows.

Distance=Z=D(2,1,2)+D(6,2,1)+D(4,1,1)+D(7,3,2)+ D(5,1,2)+D(3,5,1)

=1+2+1+3+2+3=12

12. Experimental Results:

The following table-8 shows that the computational results for proposed algorithm called pattern recognition technique based lexi-search algorithm. We write Computer program for this algorithm in C language and it is verified by the system COMPAQ dx2280 MT. We ensure this algorithm by trying a set of problems for different sizes. We take the different random numbers as values in distance matrix. The distance Matrix D (i, j, k) takes the values uniformly random in [0. 1000]. We tried a set of problems by giving different values to N, K and CL. The results are tabulated in the Table - 8 are given below. For each instance, five to eight data sets are tested. It is seen that the time required for the search of the optimal solution is fairly less.

Table-8

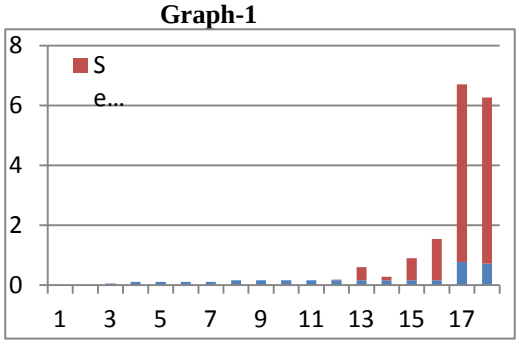
SN	Problem dimension			NPT	CPU runtime in seconds	
	N	K	CL		Avg AT	Avg ST
1	4	2	1	6	0	0
2	4	3	1	6	0	0
3	8	2	2	6	0.054945	0
4	8	3	1	6	0.054945	0
5	8	3	2	6	0.109890	0
6	8	4	2	6	0.109890	0
7	12	2	1	8	0.109890	0
8	12	2	2	8	0.109890	0
9	12	3	2	8	0.164835	0
10	12	4	2	8	0.164835	0
11	12	4	3	8	0.164835	0
12	16	2	1	8	0.164835	0
13	16	3	1	8	0.164835	0.021978
14	16	3	2	8	0.164835	0.439560
15	16	3	3	8	0.164835	0.109890
16	16	4	2	8	0.164835	0.5369231
17	20	2	1	8	0.164835	1.173626
18	20	2	2	8	0.769231	5.934060

19	20	3	2	8	0.714286	5.549451

In the above table-8, SN = serial number, N = number of cities, K = number of facilities, CL = number of clusters to be taken, NPT = number of problems tried. In the next columns Avg. AT = average CPU run time to form an alphabet table. Avg. ST = average CPU run time to form search table for obtaining the optimal solution. It is seen that time required for the search of the optimal solution is moderately less.

The graphical representation of the above instances is given below. In the following Graph – 1, X axes taken the SN and Y axes taken the values of CPU run time for forming alphabet table and getting optimal solution. The bars of different colours indicate CPU run time for formation of alphabet table and search time for getting optimal solution respectively. i.e., series1 represent that CPU run time for formation of alphabet table and series 2 represent that CPU run time for searching the optimal solution.

Graph-1



13. Comparison Details:

We implement the Pattern Recognition Technique based Lexi Search Algorithm (LSA) with C language for this model. We tested the proposed algorithm by different set of problems and compared the computational results with the published minimum spanning tree model by C. Suresh Babu Volume 3, No. 1, Jan-Feb 2012 International Journal of Advanced Research in Computer Science. Then Table-9 shows that the comparative results of different sizes.

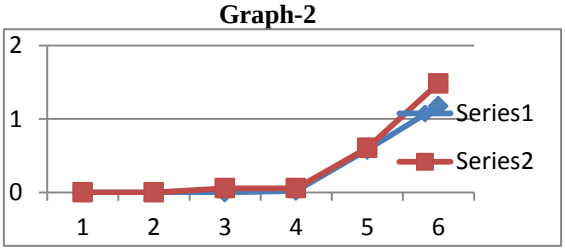
Table-9

SN	N	M	CL	NPT	CPU Run Time in seconds	
					Published model	Proposed model
1	4	2	1	5	0.000	0.000
2	8	3	1	5	0.000	0.000
3	12	3	3	5	0.05490	0.000
4	16	2	1	5	0.054945	0.000
5	18	3	2	5	0.604396	0.580312
6	20	2	1	5	1.483516	1.173626

In the above table- 9. The last two columns show the CPU run time of published model and proposed model. As compared the two models of size N=20. The runtime of this instance with the existing model is 1.483516 sec., and the proposed model took 1.173626 Sec., it is reasonably less time. The present model takes very less computational time for finding the optimal solution. Hence, suggested the present model for solving the higher dimensional problems also.

The graphical representation of the CPU run time for the two models presented in the above 6 instances is given below. In the Graph-2, X axes taken the SN and Y axes taken the values of CPU run time for the published and proposed models.

Graph-2



From the above Graph-2, series2 represent that CPU run time for getting optimal solution by published model and series 1 represent that CPU run time for searching the optimal solution by the proposed model. Also the proposed model takes less time than published model for giving the solution.

14. Conclusion:

In this paper, we have presented an exact algorithm called lexi-search algorithm based on pattern recognition technique. First the model is formulated into a zero-one programming problem. A Lexi-Search Algorithm based on Pattern Recognition Technique is developed for getting an optimal solution. The problem is discussed with suitable numerical illustration. We have programmed the proposed algorithm using C-language. The computational details are reported. As an observation the CPU run time is fairly less for higher values to the parameters of the problem to obtain an optimal solution.

15. References:

[1]. B. Chazelle, ACM 47 (2000), Minimum Spanning Tree Algorithm with Inverse-Ackermann Type Complexity, Journal), 1028-1047. Prelim. Version in FOCS 1997.

[2]. Garey, Michael R.; Johnson, David S. (1979), Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, ISBN 0-7167-1045-5.

[3]. Karger, David R.; Klein, Philip N.TarjanRobert E. (1995), "A randomized linear-time algorithm to find minimum spanning trees", *Journal of the Association for Computing Machinery* **42** (2): 321–328, doi:10.1145/201019.201022, MR 1409738

[4]. Pette,Seth,Ramachandran,Vijaya(2002),"A randomized time-work optimal parallel algorithm for finding a minimum spanning forest",*SIAM Journal on Computing***31**(6):1879–1895, doi:10.1137/S0097539700371065, MR 1954882.

[5]. Pettie, Seth; Ramachandran, Vijaya (2002), "Minimizing randomness in minimum spanning tree, parallel connectivity, and set maxima algorithms", *Proc. 13th ACM-SIAM Symposium on Discrete Algorithms (SODA '02)*, San Francisco, California, pp. 713–722.

[6]. Pop, P.C.,"New models of the Generalized Minimum Spanning Tree Problem", Journal of Mathematical

Modelling and Algorithms, Volume 3, issue 2, 2004, 153-166.

[7]. Reeves, C.R., "Modern Metaheuristics Techniques for Combinatorial Problems", Blackwell, Oxford, 1993.

[8]. SobhanBabu,K.,Chandra Kala, K., Purusotham, S. and Sundara Murthy, M. "A New Approach for Variant Multi Assignment Problem", International Journal on Computer Science and Engineering, Vol.02, No.5, 2010, 1633-1640.

[9]. SundaraMurthy,M."CombinatorialProgramming: A Pattern Recognition Approach," A Ph.D. Thesis, REC, Warangal. 1979.

[10]. Suresh Babu C, Sobhan Babu K and Sundara Murthy M, 2011, published a paper entitled "Variant Minimum Spanning Network Connectivity Problem" by the International Journal of Engineering Science and Technology for publication. Volume 3, No. 1, Jan-Feb 2012.