ijecs open access www.ijecs.in

International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 2 Issue 8 August, 2013 Page No. 2415-2423

New Algorithm for Minimum Spanning Network Connectivity Problem

Madhu Mohan Reddy P[1] Suresh Babu C[2] Purusotham S[3] Sundara Murthy M[4]

1. Research Scholar, Dept. of Mathematics, S. V. University, Tirupati, Andhra Pradesh, India.

2. Academic Consultant, Dept. of Mathematics, S. V. University, Tirupati, Andhra Pradesh, India.

3. Asst. Professor, Statistics and Operations Research Division,VIT University, Vellore, Tamil Nadu, India

4. Professor, Dept. of Mathematics, S. V. University, Tirupati, Andhra Pradesh, India.

Email: mmrphdsv@gmail.com, suresh8044@gmail.com, drpurusotham.or@gmail.com, profmurthy@gmail.com

Abstract:

Minimum Spanning Network Connectivity possesses self adaptcity. In order to construct telecommunications, wireless communications, routing is one of the key challenging issues to be addressed. In this paper, A new Algorithm for Minimum Spanning Network Connectivity is proposed. This algorithm is proposed based on the Pandit's approach. Proposed new algorithm incorporates with drainage system in a city, Vehicle routing, Linked with different rivers etc. Thus new algorithm aims to choose optimal minimum spanning network connectivity from nodes to the destination. This algorithm is compared with previously suggested algorithms to performance differences. Extension experimental evaluations are performed from suitable numerical example. It is shows that the new algorithm for minimum spanning network connectivity provides better than the previously suggested algorithms.

Key words: Minimum Spanning, Pandit's approach.

1.1 Introduction:

Now days the development of networks in the area of telecommunication, wireless sensor networks, in the directional sensor networks and cognitive radio technology has gained much importance. One of the main goals in the design process is to reach total connectivity at minimum cost/distance. 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 modeled by introducing constraints that bound the node degrees. Garey et all [2] proved that the resulting degree-constrained minimum. In this paper we studied a variation of Minimum spanning models. For this we developed a new algorithm by Pandit's approach to solve this problem.

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 et al [4, 5] have found a probably optimal deterministic comparison-based minimum spanning tree algorithm. Sobhan Babu [8] studied a variation of minimum spanning models using pattern recognition technique. i.e The problem is to find optimal solution for all the cities connected from head quarter {1} with minimum distance/cost. Pattern recognition technique was first used by Sundara Murthy [9]. Suresh Babu [10] studied another variation of spanning models, which is reverse case of Sobhan Babu. i.e The problem is to find optimal solution for all the cities connected to head quarter {1} with minimum distance/cost. Research has also considered parallel algorithms for the minimum spanning tree problem. With a linear number of processors it is possible to solve the problem in $O(\log n)$ time. Chong et al[11], (2001), demonstrate an algorithm that can compute MSTs 5 times faster on 8 processors than an optimized sequential algorithm. Bader et al [12] (2006), Later Nobari et al [13] (2012), proposed a novel, scalable,

parallel Minimum Spanning Forest (MSF) algorithm for undirected weighted graphs.

1.4 Problem Description:

Many people had tried on minimum spanning network connectivity problems. For this paper i studied the previous research problems on minimum spanning network connectivity. From those problems i took two problems which are of Suresh Babu and Sobhan Babu. Which are minimum spanning network connectivity from head quarter to the cities with least cost/distance and minimum spanning network connectivity from cities to the head quarter with least cost/distance which are related to the problems of Sobhan Babu and Suresh Babu. i.e. Sobhan Babu had tried the problem of minimum spanning network connectivity from head quarter to the cities with least cost/distance, which is taken as the first case and Suresh Babu had tried problem minimum spanning network connectivity from cities to the head quarter with least cost/distance is the second case. In both cases they used third facility which influences the cost/distance. They proposed algorithms of their problems. Those algorithms are very close. They are used, for solving the problem by lexi search approach using pattern recognition technique.

This paper deals the above two cases. *i.e*. minimum spanning network connectivity from head quarter to the cities and minimum spanning network connectivity from cities to the head quarter. I modified the algorithm of minimum spanning network connectivity from cities to the head quarter (second case) from three dimensional to two dimensions. The modified algorithm is used to the numerical example and got the solutions. We got the same solution by the new algorithm and this approach is called as Pandit's approach. For this we proposed an exact algorithm which takes less time for higher values also and easy to understand. This algorithm appears more effective than Lexi-Search algorithm using pattern recognition Technique and it is illustrated with a suitable numerical example for comparison. This algorithm is for minimum spanning network connectivity from n-1cities to the head quarter (second case). After some modifications in our algorithm we get the solution of minimum spanning network connectivity from head quarter to the cities also. This problem is named as "A New Algorithm for Minimum Spanning Network Connectivity Problem".

Symbolically consider that there are n cities and the distance array d (i, j) is given, the cost / distance from i^{th} city to j^{th} city, where i, j Є N; N = {1,2, . . ., n} i.e., D is given distance matrix and take 1 as head quarter city. Each city is to be connected to head quarter city either directly or indirectly. Our objective is to get the minimum spanning network connectivity from head quarter{1} to the {n-1}cities and minimum spanning network connectivity from {n-1} cities to the head quarter{1} by Pandit's Approach and we proposed a new algorithm.

2. Mathematical Formulation:

Let N= {1, 2… n} be the set of n cities. The distance matrix D is given. Let α^1 be the city connected from α, where α^1 is either city 1 or a city connected to head quarter.

$$Minimize Z = \sum_{\alpha_i} D(\alpha, \alpha^1) X(\alpha, \alpha^1) \text{ -----}$$

(1)

$\alpha \in N - \{1\}$ and $\alpha^1 \in N$

Subject to constraints

$$\sum_i \sum_j X(i, j) = n - 1 \text{ ----------} \quad (2)$$

X (i, j) = 0 or 1 ---------- (3)

Equation (1) represents that the objective function of the problem. i.e. to find total minimum distance to connect from all the cities to the head quarter. Equation (2) represent that the total number of arcs n-1 of the solution in the network. Equation (3) represents the i^{th} city is connected to the j^{th} city if it is 1 otherwise 0.

3. Numerical Illustration:

The algorithm and concepts are developed by illustrated a numerical example. For which the total number of cities N = {1, 2, 3, 4, 5, 6, 7, 8}. For the following example, first we find the optimal solution in Lexi-search approach using the "Pattern Recognition Technique" from cities to the head quarter. Here city "1"is taken as the head quarter. In the table the distance (i, i) is taken as 00 and "-"indicates dis-connectivity to the corresponding cities, which can be taken as ∞. Here the entries D (i, j)'s taken as non-negative integers, it can be easily seen that this is not a necessary condition and the distance can be positive quantity.

The distance matrix D (i, j) is given in the following **table-1**.

Table – 1

D(i,j)								
00	**1**	**2**	**3**	**12**	**5**	**16**	**11**
25	**00**	**13**	**22**	**-**	**4**	**1**	**-**
14	**1**	**00**	**4**	**7**	**-**	**6**	**10**
7	**3**	**4**	**00**	**2**	**8**	**17**	**9**
7	**8**	**22**	**2**	**00**	**5**	**3**	**3**
21	**6**	**23**	**1**	**11**	**00**	**-**	**9**
8	**18**	**19**	**3**	**-**	**4**	**00**	**4**
5	**-**	**9**	**7**	**21**	**4**	**8**	**00**

From the above table D (3, 2) =1 means the distance of the connecting the city 3 to 2 is 1. The objective is that the 7 cities are to be connected to the city 1 with minimum distance.

4. Concepts and Definitions:

4.1 Definition of a pattern:

An indicator two-dimensional array which is associated with the spanning is called a 'pattern'. A Pattern is said to be feasible if X is a solution.

V (x) = ΣΣ D (i, j) X (i, j)

The pattern represented in the **table-2** is a feasible pattern. The value V(X) gives the total cost/distance of the minimum spanning for the solution represented by X. Thus the value of the feasible pattern gives the total cost/distance 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 pairs [(i,j)] for which X(i,j)=1, with understanding that the other X(i,j)'s are zeros.

4.2 Feasible solution:

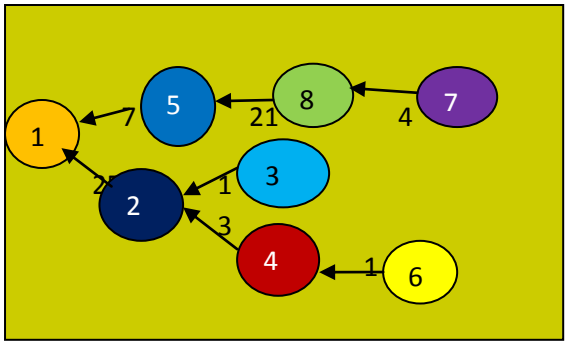
Consider an ordered pair set{(2,1),(3,2),(4,2),(5,1),(6,4)(7,8),(8,5)} represents the pattern given in the **table-2**, which is a feasible solution for minimum spanning network connectivity from cities to the head quarter.

Table-2

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

The following **figure-1** represents a feasible solution. The circles represent cities and the values in circles indicates name of the cities Also values at each arc represent distance between the respective two nodes.

FIGURE- 1



From the above **figure-1** the City 2 connected to 1, City 3and4 connected to 2, City 5 connected to 1, City 6 connected to 4, City 7 connected to 8 and City 8 connected to 5. Here the solution

Z= D(5,1)+ D(8,5)+ D(7,8)+D(2,1)+ D(3,2)+ D(4,2)+D(6,4)

=

7+21+4+25+1+3+1=62

4.3 Infeasible Solution:

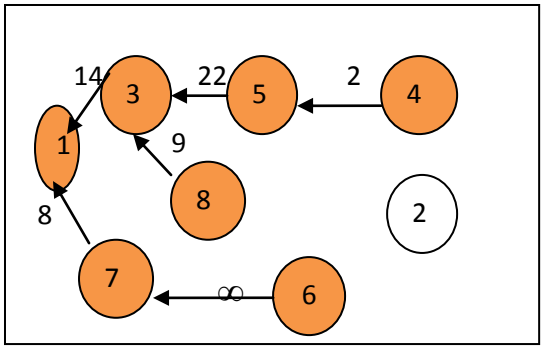
Consider a ordered pairs set {(3,1),(5,3),(4,5),(8,3),(7,1),(6,7)} represents the pattern given in the **table-3**, which is a infeasible solution for minimum spanning network connectivity from cities to the head quarter.

TABLE-3

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

The following **figure-2** represents an infeasible solution. The circles represent cities and the values in circles indicates name of the cities Also values at each arc represent distance between the respective two nodes.

Figure-2



From the above **figure-2** the City 2 is not connected to any city and ∞ indicates city 6 to city7 have no connectivity, for these two reasons the above ordered pairs gives infeasible solution.

4.4 Definition of Pattern:

An indicator two-dimensional array which is associated with spanning is called a 'pattern'. A Pattern is said to be feasible if X is a solution.

$$V(X)=\sum_{i\epsilon N}\sum_{j\epsilon N} D(i,j)\,.\,X(i,j)$$

The value V(x) is gives the total distance of the minimum spanning network connectivity for the solution represented by X. The pattern represented in the table-2 is a feasible pattern. The value V(X) gives the total distance of the network for the solution represented by X. Thus X is the feasible pattern gives the total distance 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 pairs (i, j) for which X (i, j) =1, with understanding that the other X(i,j) are zeros. The ordered pairs set {(2,1),(3,2),(4,2),(5,1),(6,4)(7,8),(8,5)} represents the pattern given in table – 2, which is feasible solution.

4.5 Alphabet Table:

There is M= n x n ordered pairs in the two-dimensional array X. For convenience these are arranged in ascending order of their corresponding distance 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 distances. For our convenience we use the same notation D. If a, b ∈ SN and a < b then, D (a) ≤ D (b). Also let the arrays R, C be the array of row and column indices of the ordered pairs represented by SN and DC be the array of cumulative sum of the elements of D. The arrays SN, D, DC, R, C, for the numerical example are given in the **table-4**. If p ∈ SN then (R(p),C(p)) is the ordered pairs and D(a)=D(R(a),C(a)) is the value of the ordered pairs and DC (a) = $\sum_{i=1}^{a} D(i)$.

Alphabet – Table (Table-4)

SN	D	DC	R	C
1	1	1	1	2
2	1	2	2	7
3	1	3	3	2
4	1	4	6	4
5	2	6	1	3
6	2	8	4	5
7	2	10	5	4
8	3	13	1	4
9	3	16	4	2
10	3	19	5	7
11	3	22	5	8
12	3	25	7	4
13	4	29	2	6
14	4	33	3	4
15	4	37	4	3
16	4	41	7	6
17	4	45	7	8
18	4	49	8	6
19	5	54	1	6
20	5	59	5	6
21	5	64	8	1
22	6	70	3	7
23	6	76	6	2
24	7	83	3	5
25	7	90	4	1
26	7	97	5	1
27	7	104	8	4
28	8	112	4	6
29	8	120	5	7
30	8	128	7	1
31	8	136	8	7
32	9	145	4	8
33	9	154	6	8
34	9	163	8	3
35	10	173	3	8
36	11	184	1	8
37	11	195	6	5
38	12	207	1	5
39	13	220	2	3
40	14	234	3	1
41	16	250	1	7
42	17	267	4	7
43	18	285	7	2
44	19	304	7	3
45	21	325	6	1
46	21	346	8	5
47	22	368	2	4
48	22	390	5	3
49	23	413	6	3
50	25	438	2	1

Let us consider 31∈SN. It represents the ordered pair (R (31), C (31)) = (8, 7). Then D (31) =D (8, 7) =8 and DC (31) =136.

4.6 Definition of an Alphabet - Table and a word:

Let SN = (1,2,…) be the set of indices, D be an array of corresponding distances of the ordered pairs and Cumulative sums of elements in D is represented as an array DC. Let arrays R, C be respectively, the row, column indices of the ordered pairs. Let $L_k = \{a_1, a_2, - - -- - , a_k\}$, $a_i \in$ SN be an ordered sequence of k indices from SN. The pattern represented by the ordered pairs 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 alphabetic order as (1, 2, - - - -, n^2) 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, - - - -, 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 pattern. 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.

In the partial word L_k any of the letters in SN can occupy the first place. Since the words of length greater than n-1 are necessarily infeasible, as any feasible pattern can have only n unit entries in it. L_k is called a partial word if k < n-1, and it is a full length word if k = n-1, or simply a word. A partial word L_k represents, a block of words with 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.

4.7. 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_k) with V (L_o) = 0 where D (a_k) is the distance/cost array arranged such that D (a_k) < D (a_{k+1}). V (L_k) and V(x) the values of the pattern X will be the same. Since X is the (partial) pattern represented by L_k, (Sandara Murthy – 1979).

4.8. 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
$L_k = (a_1, a_2, - - - - , a_k)$ can be defined as follows.

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

Consider the partial word L_4 = (2, 3, 4, 6)

V (L_4) = 1+1+1+2 = 5

LB (L_4) = V (L_4) + DC (a_4 + n - k) - DC (a_4)

= 5+ DC (6 + 7 - 4) - DC (6)

= 5+ DC (9) - DC (6)

= 5 + 16 - 8 = 13

Where DC (a_k) = $\sum_{i=1}^{k} D(a_i)$. It can be seen that LB (L_k) is the value of the complete word, which is obtained by concatenating the first (n-k) letters of SN (a_k) to the partial word L_k

.

4.9. Feasibility criterion of a partial word:

An algorithm was developed, in order to check the feasibility of a partial word L_{k+1} = (a_1, a_2, - - - -- a_k, a_{k+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 ∈ N indicates that the i^{th}city is connected to city j. Otherwise IC = 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, α_i ЄSN is the letter in the i^{th} position of word.

The values of the arrays IR, SW, L are as follows

- **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) = α_i, i = 1, 2, - - - - -, k, and L (j) = 0, for other elements of j.

For example consider a sensible partial word L_7 = (2, 3, 4, 6, 11, 12, 21) which is feasible. The array IR, IC, L takes the values represented in **table - 5** given below.

Table - 5

	1	2	3	4	5	6	7	8
L	2	3	4	6	11	12	21	-
IR	-	1	1	1	1	1	1	1
IC	-	1	1	1	1	1	1	1
SW	-	7	2	5	8	4	4	1

4.10 Algorithm-1:

STEP1 :IX=0

STEP 2 :IS (IR (TR) = 1) IF YES GOTO 8
IF NO GOTO 3

STEP3 :W = TC GOTO 4

STEP 4 :IS W = TR IF YES GOTO 8

IF NO GOTO 5

STEP5 :IS SW (W) = 0 IF YES GOTO 7
IF NO GOTO 6

STEP 6 ;W = SW (W) GOTO 4

STEP7 : IX=1 IF YES GOTO 8

IF NO GOTO 1

STEP 8 : STOP

We start with the partial word L_1 = (a_1) = (1). A partial word L_p is constructed as $L_p = L_{p-1} * (\alpha_p)$. Where * indicates chain formulation. We will calculate the values of V (L_p) and LB (L_p) simultaneously. Then two situations 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 under (p+1). Which represents a sub-block of the block of? Words represented by L_p. If L_p is not feasible then consider the next partial word p by taking another letter which succeeds a_p in the 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) ≥ 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.

4.11 Algorithm- 2: (Lexi - Search algorithm)

STEP 1 : (Initialization)

The arrays SN, D, DC, R, C and the values of n, High, MAX are made available IR, IC, L, V, LB are initialized to zero. The values I=1, J=0, IR (TR) =0, VT = High

STEP 2 :J=J+1

IS (J>MAX) IF YES GOTO 11
IF NO GOTO 3

STEP 3 :L (I) = J TR=R (J) TC=C (J)
GOTO 4

STEP 4 : V (I) =V (I-1) +D (J)

LB (I) =V (I) +DC (J+N-I)-DC (J)
GOTO 5

STEP 5 :IS (LB (I) ≥ VT) IF YES GOTO 10
IF NO 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) IF YES GOTO 8
IF NO GOTO 9

STEP 8 : L (I) = J

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

VT=V (I), RECORD L, VT GOTO 11

STEP 9 :IR (TR) =1

SW (TR) = TC

I = I + 1 GOTO 2

STEP10 : IS (I=1) IF YES GOTO 13
IF NO GOTO 11

STEP11 : I=I GO TO 12

STEP12 : J=L (I) ,TR = R (J),
TC = T (J) IR
(TR) =0 SW(TR)=0
GOTO2

STEP13 : STOP & END.

4.12 Search Table:

The working details of getting an optimal word using the above algorithm for the illustrative numerical example is given in the **Table-6** The columns named (1), (2), (3),…, gives the letters in the first, second, third and so on places respectively. The columns R, C give the row, column 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.

Search Table (table-6)

SN	1	2	3	4	5	6	7	V	LB	R	C	Re
1	1							1	10	1*	2	R
2	2							1	11	2	7	A
3		3						2	12	3	2	A
4			4					3	12	6	4	A
5				5				5	12	1*	3	R
6				6				5	13	4	5	A
7					7			7	13	5	4	R
8					8			8	14	1*	4	R
9					9			8	14	4*	2	R
10					10			8	14	5	7	A
11						11		11	14	5*	8	R
12						12		11	15	7	4	R
13						13		12	16	2*	6	R
14						14		12	16	3*	4	R
15						15		12	16	4*	3	R
16						1		1	16	7	6	R
						6		2				
17						17		12	16	7	8	A
18							18	16	16	8	6	R
19							19	17	17	1*	6	R
20							20	17	17	5*	6	R
21							21	17	17=vt	8	1	A
22						18		12	17	8	6	R
23					11			8	15	5	8	A
24						12		11	15	7	4	A
25							13	15	15	2*	6	R
26							14	15	15	3*	4	R
27							15	15	15	4*	3	R
28							16	15	15	7*	6	R
29							17	15	15	7*	8	R
30							18	15	15	8	6	R
31							19	16	16	1*	6	R
32							20	16	16	5*	6	R
33							21	16	16=vt	8	1	A
34						13		12	16*	2*	6	R
35					12			11	19*	7	4	R
36				7				7	16*	5	4	R
37			5					4	14	1*	3	R
38			6					4	15	4	5	A
39				7				6	15	5	4	R
40				8				7	16*	1	4	R
41			7					4	16*	5	4	R
42		4						2	17*	6	4	R
43	3							1	14	3	2	A
44		4						2	14	6	4	A
45			5					4	14	1*	3	R
46			6					4	15	4	5	A
47				7				6	15	5	4	R
48				8				7	16*	1	4	R
49			7					4	16*	5	4	R
50		5						3	16*	1	3	R
51	4							1	16*	6	4	R

The partial word is L_7 = (2, 3, 4, 6, 11, 12, 21) is a feasible partial word. For this partial word the array IR, IC, L are given in the following

Table –7.

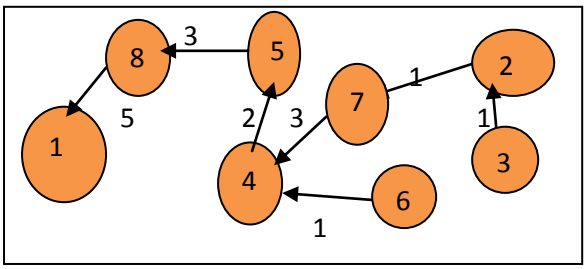
	1	2	3	4	5	6	7	8

L W	**2**	**3**	**4**	**6**	**11**	**12**	**21**	**-**
IR	**-**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
IC	**1**	**1**	**-**	**1,1**	**1**	**-**	**1**	**1**
S W	**-**	**7**	**2**	**5**	**8**	**4**	**4**	**1**

At the end of the search the current value of the **VT is 16** and it is the value of optimal feasible word.L_7= (2, 3, 4, 6, 11, 12, 21) it given in the **33rd row** of the search table.

The following **figure-3** represents a feasible solution. The values at each arc represent distance between the respective two nodes.

FIGURE-3



From the above figure-3 the City 8connected to 1, City 5 connected to 8, City 4 connected to 5, City 6 and 7 connected to 4, City 2 connected to 7 and City 3 connected to 2. Here the solution Z=D(8,1)+ D(5,8)+ D(4,5)+ D(7,4)+ D(2,7)+ D(3,2)+ D(6,4) = 5+3+2+3+1+1+1=16.

For the above numerical example (**table-1**) the optimal solution is 16 from cities to the head quarter, based on lexi search approach using pattern recognition technique. The optimal solution from cities to the headquarter city is 16. For the same numerical example, we develop the new algorithm which solves and gives the same solution 16. The algorithm is efficient, easy and it is more convenient to solve. The distance matrix is as follows, which is in table-1.

Table-1

D(i,j)

00	**1**	**2**	**3**	**12**	**5**	**16**	**11**
25	**00**	**13**	**22**	**-**	**4**	**1**	**-**
14	**1**	**00**	**4**	**7**	**-**	**6**	**10**
7	**3**	**4**	**00**	**2**	**8**	**17**	**9**
7	**8**	**22**	**2**	**00**	**5**	**3**	**3**
21	**6**	**23**	**1**	**11**	**00**	**-**	**9**
8	**18**	**19**	**3**	**-**	**4**	**00**	**4**
5	**-**	**9**	**7**	**21**	**4**	**8**	**00**

The distances in the above matrix D are arranged in a convenient form in two matrices DV and DA. In each column the distances of D are arranged in increasing order and the corresponding row indices are also noted. The distances of j^{th} column and are taken as the j^{th} column elements of DV and the corresponding row indices are taken as the j^{th} column of DA. To illustrate D(1,5)=12. It is 4^{th} value in the 5^{th} column. So DV (4, 5) =12 and it row index is 1. So DA (4, 5) =1. i.e. D(DA(4,5),5)=DV(4,5)=12.

DV Matrix (TABLE-8)

DV(i,j)=

5	**1**	**2**	**1**	**2**	**4**	**1**	**3**
7	**1**	**4**	**2**	**7**	**4**	**3**	**4**
7	**3**	**9**	**3**	**11**	**4**	**6**	**9**
8	**6**	**13**	**3**	**12**	**5**	**8**	**9**
14	**8**	**19**	**4**	**21**	**5**	**16**	**10**
21	**18**	**22**	**7**	**-**	**8**	**17**	**11**
25	**-**	**23**	**22**	**-**	**-**	**-**	**-**
-	**-**	**-**	**-**	**-**	**-**	**-**	**-**

DA Matrix (TABLE-9)

DA(i,j)=

8	**1**	**1**	**6**	**4**	**2**	**2**	**5**
4	**3**	**4**	**5**	**3**	**7**	**2**	**7**
5	**4**	**8**	**1**	**6**	**8**	**3**	**4**
1	**6**	**2**	**7**	**5**	**1**	**8**	**6**
3	**5**	**7**	**2**	**8**	**5**	**1**	**3**
6	**7**	**5**	**8**	**-**	**4**	**4**	**1**
7	**-**	**6**	**2**	**1**	**3**	**-**	**-**
-	**-**	**-**	**-**	**-**	**-**	**-**	**-**

From the tables 1, 8&9 we develop the new algorithm and it gives the solution for the minimum spanning network connectivity between cites to the head quarter city 1 which is equal to the optimal solution16.

5. Algorithm:

STEP0: DV,DA, MAX, n are made available and IA,JA, INX,V are initialized to zero.

STEP2: I=1
INX (1) =1
IZ=DA (1, 1)
INX (IZ) =1
V (1) =DV (1, 1) GO TO 4

STEP4: I=I+1
IS (I<N) IF YES GO TO 6,
IF NO GO TO 20

STEP6: IA =0
MIN=MAX
GO TO 8

STEP8: IA=IA+1 , JA=0
IS (IA<N)
IFYES GOTO10,
IF NO GO TO 18

STEP10: IS (INX (IA) =0)
IFYES GOTO8,
IF NO GO TO 12

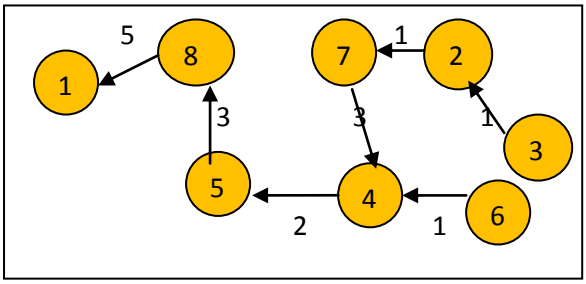
STEP12: JA=JA+1
IZ=DA (IA, JA)

IS (JA≤N)
IFYESGOTO14,
IF NO GO TO 8
STEP 14: IS (MIN<DV (IZ, IA))
IF YES GO TO 16,
IF NO GO TO 8
STEP16: MIN=DV (IZ, JA)
IK=IZ
IJ=JA
GO TO 8
STEP18: V (I) =V (I) +DV (IK, IJ)
WRITE V (I), (IK, JA)
INX(IK) =1
GOTO 4
STEP 20: END

Using the algorithm the solution for given numerical example is

Z=D(8,1)+D(5,8)+D(4,5)+D(6,4)+D(7,4)+D(2,7)+D(3,2)=5+3+2+1+3+1+1=16.

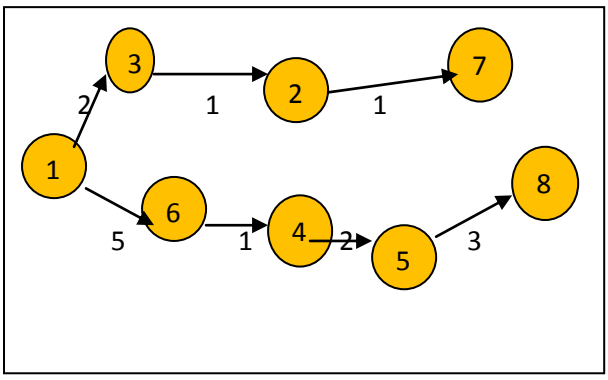
The following **figure-4** represents a feasible solution, which is also optimal.



Therefore, the solutions of lexi search algorithm using pattern recognition technique and the new algorithm are same. The above new algorithm gives the optimal solution for higher dimensions also.

In the above lexi search algorithm with some modifications we can get the minimum spanning network connectivity from head quarter to the n-1 cities. From this algorithm, we got the optimal solution. The optimal solution from head quarter to the cities n-1=7 is 15. Which is shown in the following figure-5.

FIGURE-5



From the above **figure-5** the City 1 connected to 3&6, City 3 connected to 2, City 2 connected to 7, City 6 connected to 4, City 4 connected to 5 and City 5 connected to 8. Here the solution Z=D(1,3)+ D(3,2)+ D(2,7)+ D(1,6)+ D(6,4)+ D(4,5)+ D(5,8) = 2+1+1+5+1+2+3=15. For the above numerical example (**table-1**) the optimal solution is 15 from head quarter to the cities, based on lexi search approach using pattern recognition technique. For the same numerical example, we modified the new algorithm apply to get and the same solution 15. The distance matrix is as follows, which is in table-1.

D(i,j)

00	1	2	3	12	5	16	11
25	00	13	22	-	4	1	-
14	1	00	4	7	-	6	10
7	3	4	00	2	8	17	9
7	8	22	2	00	5	3	3
21	6	23	1	11	00	-	9
8	18	19	3	-	4	00	4
5	-	9	7	21	4	8	00

The distances in the above matrix D are arranged in a convenient form in two matrices DV and DA. In each row the distances of D are arranged in increasing order and the corresponding column indices are also noted. The distances of i^{th} row and are taken as the i^{th} row elements of DV and the corresponding indices are taken as the i^{th} row of DA. The following **table -10** is DV matrix and **table-11** is DA matrix. Again, Agai for convenience we use the same nota V and D

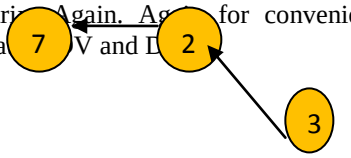


TABLE-10

DV(i,j)=

1	*2*	*3*	*5*	*11*	*12*	*16*	-
1	*4*	*13*	*22*	*25*	-	-	-
1	*4*	*6*	*7*	*10*	*14*	-	-
2	*3*	*4*	*7*	*8*	*9*	*17*	-
2	*3*	*3*	*5*	*7*	*8*	*22*	-
1	*6*	*9*	*11*	*21*	*23*	-	-
3	*4*	*4*	*8*	*18*	*19*	-	-
4	*5*	*7*	*8*	*9*	*21*	-	-

TABLE-11

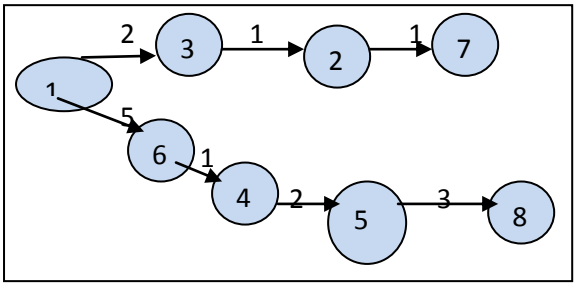
DA(i,j)=

2	*3*	*4*	*6*	*8*	*5*	*7*	-
7	*6*	*3*	*4*	*1*	-	-	-
2	*4*	*7*	*5*	*8*	*1*	-	-
5	*2*	*3*	*1*	*6*	*8*	*7*	-
4	*7*	*8*	*6*	*1*	*2*	*3*	-
4	*2*	*8*	*5*	*1*	*3*	-	-
4	*6*	*8*	*1*	*2*	*3*		
6	*1*	*4*	*7*	*3*	*5*	-	-

The solution is
Z=D(1,2)+D(2,7)+D(7,4)+D(4,5)+D(5,8)+D(7,6)+D(1,3)=1+1+3+2+3+4+1=15 The following **figure-6** represents an optimal solution. The values at each arc represent distance between the

respective two nodes.

FIGURE-6



From the above **figure-6** the City 1 connected to 3&6, City 3 connected to 2, City 2 connected to 7, City 6 connected to 4, City 4 connected to 5 and City 5 connected to 8. Here the solution Z=D(1,3)+ D(3,2)+ D(2,7)+ D(1,6)+ D(6,4)+ D(4,5)+ D(5,8) = 2+1+1+5+1+2+3=15.

6. Conclusion:

In this paper, we have studied a model namely "A New Algorithm for Minimum Spanning Network Connectivity Problem". We have developed a new algorithm which is efficient, accurate and easy to understand than the algorithm of lexi search approach using pattern recognition technique. First the model is formulated in to a zero one programming problem. The problem is discussed in detail with help of numerical illustration.

7. References:

[1]. B. Chazelle, A Minimum Spanning Tree Algorithm with Inverse-Ackermann Type Complexity, Journal ACM 47 (2000), 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.; Tarjan, Robert 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]. Pettie, 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., "Moderen Metaheuristics Techniques for Combinatorial Problems", Blackwell, Oxford, 1993.

[8]. Sobhan Babu, 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]. Sundara Murthy, M. "Combinatorial Programming: 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.

[11] Chong, Ka Wong; Han, Yijie; Lam, Tak Wah (2001), "Concurrent threads and optimal parallel minimum spanning trees algorithm", *Journal of the Association for Computing Machinery* **48** (2): 297–323, doi:10.1145/375827.375847, MR 1868718.

[12] Bader, David A.; Cong, Guojing (2006), "Fast shared-memory algorithms for computing the minimum spanning forest of sparse graphs", *Journal of Parallel and Distributed Computing* **66** (11): 1366–1378, doi:10.1016/j.jpdc.2006.06.001.

[13] Nobari, Sadegh; Cao, Thanh-Tung; Karras, Panagiotis; Bressan, Stéphane (2012), "Scalable parallel minimum spanning forest computation", *Proceedings of the 17th ACM SIGPLAN symposium on Principles and Practice of Parallel Programming*