Original Article

Bulletin of Pure and Applied Sciences.
Vol. 37E (Math & Stat.), No.1, 2018.P.63-81
Print version ISSN 0970 6577
Online version ISSN 2320 3226
DOI: 10.5958/2320-3226.2018.00008.5

Restricted Constraints in a Max-Flow and Min-Cost problem

Nazimuddin Ahmed*

Author Affiliation:
Assistant Professor, Department of Statistics, D.H.S.K. College, Dibrugarh, Assam, India-786001.
E-mail: mdnahmed@yahoo.com

***Corresponding Author:**
Nazimuddin Ahmed, Assistant Professor, Department of Statistics, D.H.S.K. College, Dibrugarh, Assam, India-786001.

E-mail: mdnahmed@yahoo.com

Received on 24.02.2018, **Accepted on 08.03.2018**

Abstract

The present paper defines a combination of two most important and old flow problems such as maximum flow and minimum cost flow. In the first one a flow with the maximum value from source node to sink node is sought. The second one seeks a flow with the minimum total transferring costs from source nodes to demand nodes. Here an attempt is made to obtain an optimal route of a more realistic situation as to scheduling some restricted constraint, in which its transferring cost is minimized and its value is maximized. The proposed algorithm is formulated and solved by the lexicographic search approach. It is seen that the time required for the search of the optimal solution is fairly less. Also the algorithm is applied to different order matrices with number of stations = 5, 8, 10, … , 50 and in the dimensions of 6 to exhibit its effectiveness.

Keywords: Distance and capacity matrix, lexicographic search approach, restricted constraints and algorithm.

1. INTRODUCTION

A network consists of a set of nodes (vertices) and a set of arcs (edges). Each node represents a location (city) and each arc represents the connection (road link) between two different locations (cities). The number(s) on each arc represents the distance (cost, time) between the two locations (cities), capacity of the link between the cities, etc.

Let us consider the situation of a pipe network used to transfer fluid (oil, water, etc.) from one location to another. The maximum flow of fluid in each pipe- segment will be limited because of a particular value (capacity of the arc) depending on the diameter of the pipe in that segment or the slope of the pipe in the segment. The arc $i-j$, i.e., pipe segment between any two locations i and j of the network is associated with the following data:

(i) The maximum permitted flow of fluid per unit time from node *i* to the node *j* and ***l*** or
(ii) The maximum permitted flow of fluid per unit time from node *j* to the node *i*.

The problem we have considered here is:
Let the undirected network consists of nodes which for convenience are denoted in any order of sequence *S*, 1, 2, . . . , (*D* –1), *D*. It is assumed that certain capacity restriction and unit-shipping cost is associated with each link of the network. The problem is to schedule maximum possible flows in minimum cost from some specified node, say, source denoted by *S*, to another specified node, say, destination denoted by *D*. We have considered a 11×11 matrix (chosen arbitrarily) with some restricted constraints (also chosen arbitrarily) and solved the same by lexicographic search approach [2, 3], gives the exact solution of the problem.

The importance of restricted constraints like precedence constraints, fixed position constraints and mixed constraints, etc. in a routing problem has been very well discussed by Scrogg and Tharp [24]. It occurs frequently in many realistic situations. For example, a salesman has a quota of products to sell on a tour and he may want to visit the better prospects early. An early supply to a station may be made to the flood prone areas than the other stations to avoid the difficult situation it might fall in; a CEO of a company may plan a tour and need to be in a certain station on a particular date, etc.
By precedence constraints one means that the stations are visited in such a way that a particular station is to be preceded by another specific station. Precedence relation need not be immediate. Let us assume restricted relations are of the type $(a<b)$ or $(a<b<c)$ or $(a,b,c<d)$ etc. and these lead to $(b \to a = \infty)$, $(c \to b \to a = \infty)$, $(d \to a,b,c = \infty)$, etc.

2. A BRIEF REVIEW OF LITERATURE

Routing problems pertain to the search for a shortest route (minimum cost or minimum distance) or maximum flow, etc. connecting two specified stations or nodes described as 'source' and 'sink'. The problem of determining the shortest route under some constraints has been solved by many authors [4, 5, 17, 19].

In all these problems it has been assumed that the link capacities are infinite. Ford and Fulkerson [12, 13] introduced the idea of imposing capacity restriction on the links. Some references along this line can be made as Fathabadi and Shirdel [11], Sedeno-Noda and Gonzalez-Martin [25]. Purusotham and Sundara Murthy [22, 23] proposed a Pandit's maxi-min algorithm for solving the maximum flow with minimum cost problem and showed the efficiency of the algorithm in their paper. Ahmed, et. al. [1] applied a lexicographic search algorithm in their paper for the problem of maximum flow with minimum attainable cost in a network and find an optimum solution.

Many solution algorithms have appeared for the problem(s) to find a maximum capacity route [8] or maximum capacity flow using as many routes as possible [9, 14]. Also, many situations have appeared to search for a shortest route connecting two specified nodes [6, 7, 20, 21, 22]. Saksena and Kumar have solved a variant of this problem when the route must pass through some specified nodes [26]. Other flow problems are of the type to find the minimal cost flow where each arc is assigned with (cost, capacity) values [15]. Das and Borah [10] showed a lexi-search approach to a precedence constrained *m*-TSP.

3. NOTATION AND STATEMENT OF THE PROBLEM

The problem we have considered here is:
Let the undirected network consist of nodes which for convenience are denoted in any order of sequence *S*, 1, 2, . . . , (*D* –1), *D*. It is assumed that certain capacity restriction and unit-shipping cost is associated with each link of the network. The problem is to schedule maximum flow in minimum possible cost from some specified node, say source denoted by *S*, to another specified node, say

destination denoted by *D*, using the restricted constraints, i.e., the stations are visited in such a way that a particular station is preceded by another given station(s).
Let us assume that the precedence relations are of type, $a<b,\ c<e,\ b<d,\ e<f,\ g<h$ etc.

The precedence relation being transitive, we can form precedence chains from the above relations as, $(S<a<b<d<D)$, $(S<c<e<f<D)$, $(S<g<h<D)$ and $(S<a<b<d<D)$; implies, $(S\to b=\infty)$, $(S\to d=\infty)$, $(b\to a=\infty)$, $(d\to b=\infty)$, $(d\to a=\infty)$, $(a\to D=\infty)$ and $(b\to D=\infty)$.

It is, of course, not necessary that the nodes (*a*, *b*, *d*), (*c*, *e*, *f*) etc. are to be visited in a string, i.e., '*a*' need not be immediately followed by '*b*', '*d*' or '*c*' need not be immediately followed by '*e*', '*f*' etc. However, when (*a*, *b*, *d*), (*c*, *e*, *f*) or (*g*, *p*) are in a string, a merger of nodes into one and deletion of appropriate rows and columns in the distance (cost) matrix reduces it to a problem virtually without constraints.

Similarly, for other relations, the cost matrix can be suitably modified.

In particular, in the present study, we consider the precedence relations as $(2,\ 5<7<9)$ and $(1<3<5,\ 8)$. From these relations we construct a table of predecessors for the different nodes as shown in following table 1. However, in the table, we have only entered the predecessors which are directly prescribed by the restrictions. We shall denote the set of predecessor of a node '*a*' by *P*(*a*).

Table 1: List of predecessors for the different nodes

Nodes	*P*(*a*) (other than the depot)
S	-
1	-
2	-
3	1
4	-
5	1, 3
6	-
7	2, 5
8	1, 3
9	2, 5, 7

4. MATHEMATICAL FORMULATION

Let the set *W* contains all feasible flow in the network *G* = (*V*, *E*) satisfying the described restricted constraint. In a general mathematical form, the problem is written as

$$\underset{X\in W}{Maximize}\quad v_x$$

Such that,

$$Z(X)=Min\sum_{x}CX\ ,$$

$$\sum_{j}x_{ij}-\sum_{j}x_{ji}=\begin{cases} v_x & \text{for } i=1\\ 0 & \text{for } i\neq 1,\ n\\ -v_x & \text{for } i=n\end{cases}$$

and $$0\le x_{ij}\le k_{ij}\ ,\quad (i,j)\in G\,.$$

where, $Z(y)$ is the value of the objective function of the inner linear programming for a given flow, *y* say.

The precedence constraints are satisfied while searching for the minimum cost flow.

5. NUMERICAL ILLUSTRATION

The problem is to schedule maximum flow, which costs, minimum for a situation given in the following diagram (fig. 1)

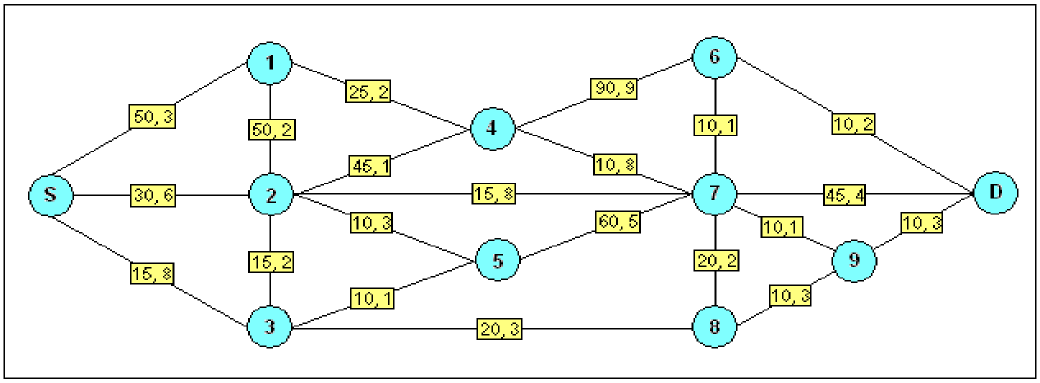

Fig. 1: Given network

In this graph the first element in the ordered pair (in a square) on a link indicates its capacity and second element in the ordered pair gives the unit shipping cost. The given network hold for both directions, i.e., the network under study is an undirected one. Here, the restricted constraints are $(A_2 < A_7)$, $(A_5 < A_7)$, $(A_3 < A_5)$, $(A_3 < A_8)$, $(A_7 < A_9)$ and $(A_1 < A_3)$.

For simplicity, we take (2 < 7), (5 < 7), (3 < 5), (3 < 8), (7 < 9) and (1 < 3) .

All these constraints can be simply written as $(2,\ 5 < 7 < 9)$ and $(1 < 3 < 5,\ 8)$. The precedence constraints are to be satisfied when the concerned nodes occur in the same route.

6. COMPUTATIONAL PROCEDURE

We solve the problem of finding the restricted constraints in the problem of maximum attainable flow in minimum cost in a network in 4 steps.

In the first step, modify the cost matrix by assigning $d_{ij} = \infty$ for which the tour between node '*i*' and node '*j*' is not possible due to the precedence constraints.

In the second step, using the cost matrix, we construct an alphabet table.

In the third step, we apply the lexicographic search approach to find the valid routes. During the lexicographic search procedure, a computationally simple bound can be obtained as $B(L_k) = V(L_k) + d'(S_k)$ where, $V(L_k)$ is the value of the leader L_k. Evaluation of $d'(S_k)$ are facilitated by an alphabet table with nodes $0,1,2,\cdots,(n-1)$ listed in increasing order of their distances to the home station. The lower bound setting as above helps converging to optimum solution rapidly.

In the fourth step, we select the first minimum cost route and search for maximum admissible flow through this route. Delete all those link(s) which are completely busy.

Next we select the second minimum cost and search for maximum admissible flow through this route, deleting all those link(s) which become completely busy in the process.

The process is continued till any one of the following cases arises:
(i) All the links terminating in destination (*D*) are full.
(ii) All the links starting from source (*S*) are full.
(iii) Neither the links which leave the source are full nor those which terminate in destination are full but the (*S*) and (*D*) do not remain connected.

It may be pointed out that if the given situation enters (i) or (ii) above, we shall terminate the process, as no further flow will be possible. However, if the situation happens to be in (iii), scheduling of more flow might be possible by diverting the original scheduled flow due to the fact that links starting from source have more capacity for outflow and links terminating in the destination can allow more inflow. It may be noted that any diversion will require more cost of shipping, even for those items which are originally scheduled to be at a lesser cost of shipping.

Also care has to be taken throughout the process that if alternative routes (in the sense of shipping cost) between any two nodes are available, the choice will be for a path whose capacity is more, or if possible, engage all such alternative routes.

The detailed algorithm and flow chart of the problem are described in section 6.1.

6.1. PART-I: Formation of cost matrix and alphabet table(s)

Step1: Modify the cost matrix by assigning $d_{ij} = \infty$ for which the tour between node '*i*' and node '*j*' is not possible following the precedence constraints.

Step2: Using the modified cost matrix, list under column *i*, $(i = 0,1,2,\cdots,(n-1))$ having links from the column *i* with the nodes $\{0,1,2,\cdots,(n-1)\}$ in order $J_0^{(i)}, J_1^{(i)},\cdots, J_{n-1}^{(i)}$ such that the cost values are in ascending order of that column. The ordering $(J_0, J_1,\cdots, J_{n-1})$ so obtained for a given node *j*, is defined as the alphabetical order for column *j* and the table, thus formed containing all the columns $0,1,2,\cdots,(n-1)$ is called the alphabet table 4. In alphabet table along the ascending cost value the respective link position value is also accommodated for each column.

Step 3: Construct another table using the last column of the cost matrix, arranging the nodes $J_i^{(i)}$, $i = 0,1,2,\cdots,(n-1)$ such that $r < s$ if $d_{rD} \le d_{sD}$, the ordering thus obtained is an alphabet table (cf. Table 4). This arrangement of column matrix is used for lower bound setting. The alphabet table (cf. Table 4) thus enables us to list the tours in a systematic way such that the values of 'incomplete words' (leaders) at different stages also present a useful hierarchical structure. We can set 'lower bounds' to this incomplete word for quick convergence to the optimal solution.

Setting up of lower bound: A computationally simple bound can be obtained as $B(L_k) = V(L_k) + d'(S_k)$ where, $V(L_k)$ is the value of the leader L_k. Evaluation of $d'(S_k)$ are facilitated by alphabet table (cf. Table 4) with nodes $0,1,2,\cdots,(n-1)$ listed in increasing order of their distances to *D*, the destination.

Suppose at some stage *k*, the leader of order *k* is $L_k = \{a_0,\ a_1,\ a_2,\ \cdots,\ a_k\}$ and $S_k = \{a_0,\ a_1,\ a_2,\ \cdots,\ a_k;\ D\}$, the corresponding node set

Since the flow has to reach to *D*, we can examine at stage *k*, at least how much the flow would take to go to *D* from any of the unvisited stations, i.e.,

$$d'(S_k) = \min_{j \notin S_k} d(j,\ D) \qquad \ldots \qquad \ldots \qquad \ldots \qquad (1)$$

If we have a trial solution L^* of value V_t and if $V_t < B(L_k)$, it is obvious that L_k cannot contain any complete tour, which is even as good as the solution L^* at hand and we jump over the block and search in the next entry of order *k* – 1. On the other hand, if $B(L_k) < V_t$, it may give a better solution than at hand and so we proceed to complete the route and go to the next block and examine the possibility of a better solution. If $B(L_k) = V_t$ on completion, we might get an alternate solution of value V_t. If $B(L_k) > V_t$, we reject the current leader and go back to $(k-1)^{th}$ step and examine the next entry on that stage and so on.

6.2. PART-II: Lexicographic search

We start with a trial solution V_t with a sequence *T*.

Step 1: Let $L_k = \{a_0, a_1, a_2, \cdots, a_{k-1}\}$ and $F(L_k) = \{f_1, f_2, \cdots, f_k\}$ be the corresponding flow sequence of L_k. $L_k = a_1$ with *k* =1. The first entry of the block 1, go to step 2.

Step 2: Is $k \le n$? If yes go to 3. If no go to 5c.

Step 3: Compute the cost from $V(L_k)$.
$V(L_k) = V(L_{k-1}) + C(a_{k-1}, a_k)$ with $V(L_0) = 0$. Go to 4.

Step 4: Is $V(L_k) > V_t$? If yes go to 6. If no go to 5.

Step 5: Is $a_k = D$ (sink node)? If yes go to 8. If no go to 5a.

Step 5a: Is the elements in the respective column of a_k is exhausted?
If yes go to 5c. If no go to 5b.

Step 5b: $L_{k+1} = (L_k, a_{k+1})$; where, a_{k+1} is the first sub-block of L_k.
Put *k* = *k* +1, go to 2.

Step 5c: Is *k* =1? If yes go to 7. If no go to 6.

Step 6: Move out of the present L_k and go to next block of order *k* – 1, select the next variable entry. Go to 2.

Step 6a: Check for fulfillment of restricted constraints
Is $P(a_k) \subset S_k$? If yes, go to 7. If no go to 6.

Step 7: Is the list of column 1 exhausted?
If yes go to 12. If no go to 11.

Step 8: Compute the flow from $F(L_k)$.
$F(L_k) = \text{Min } \{f_1, f_2, \cdots, f_k\} = f_i$ (say), Go to 9.

Step 9: Is $f_i = 0$? If yes go to 9a. If no go to 10.

Step 9a: Block the corresponding arc. Go to 6.

Step 10: $V(L_k) = V_t$, $L_k^* = T$ and $F(L_k) = f_i$. Go to 10a.

Step 10a: Set *k* = *k* –1. Go to 2.

Step 11: Take the next available entry in L_k. Go to 1.

Step 12: Search is over; current *T* gives minimum cost maximum flow (MCMF) in a single channel with a value V_t. Go to 13.

Step 13: Record final value $V(L_k) = V_t$ and $F(L_k)$. Go to 14.

Step 14: $F(L_k) = F = f_k - f_i;\ i = 1,\ 2,\ \cdots,\ k$. Go to 14a.

Step14a: Reconstruct the revised alphabet table with the latest available arcs. Go to15.

Step 15: Repeat the algorithm until the flow is zero from a source node to sink node. Go to 16.

Step 16: Cumulate the cost obtained from $V(L_k)$ and flow obtained from $F(L_k)$ for all the different sequences. Go to 17.

Step 17: Stop and end.

Flow charts of the solution procedure have been given in Fig. 2. & Fig. 3.

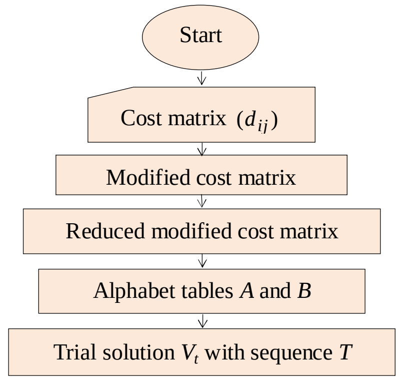

Fig. 2: Part I: Formation of alphabet table and a trial solution.

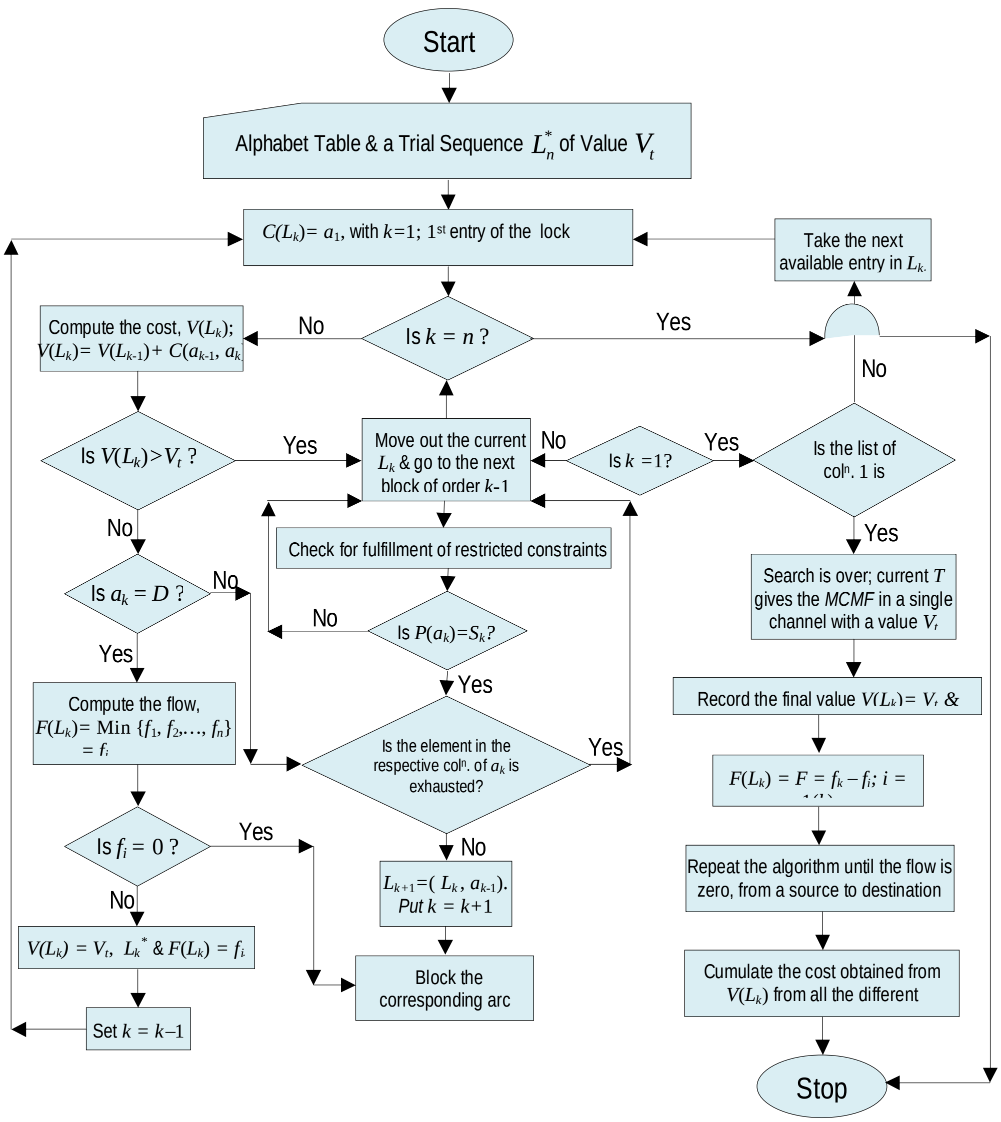

Fig. 3: Part II: Lexicographic search

7. SOLUTIONS

To apply the lexicographic search approach for obtaining the optimum solution of minimum cost and the maximum flow problem, first we construct a cost (distance) and flow (capacity) matrix with the help of the given network (fig.1). The matrix is in symmetric form as the network is an undirected one, otherwise it is asymmetric. The set of nodes is taken as row and column indices. The cost and flow associated with a pair of nodes are represented in the respective position of the cost matrix. If there is no direct link between a pair of nodes than the corresponding cost will be taken as infinity in the case cost minimization problems. Conveniently, the bold numbers in a cell of table 2 indicates the unit shipping cost of the respective flow. The cost-flow matrix is shown in table 2.

Table 2: Cost-flow matrix

Stations	*S*	1	2	3	4	5	6	7	8	9	*D*
S	-	3^{50}	6^{30}	8^{15}	×	×	×	×	×	×	×
1	3^{50}	-	2^{50}	×	2^{25}	×	×	×	×	×	×
2	6^{30}	2^{50}	-	2^{15}	1^{45}	3^{10}	×	8^{15}	×	×	×
3	8^{15}	×	2^{15}	-	×	1^{10}	×	×	3^{20}	×	×
4	×	2^{25}	1^{45}	×	-	×	9^{90}	8^{10}	×	×	×
5	×	×	3^{10}	1^{10}	×	-	×	5^{60}	×	×	×
6	×	×	×	×	9^{90}	×	-	1^{10}	×	×	2^{10}
7	×	×	8^{15}	×	8^{10}	5^{60}	1^{10}	-	2^{20}	1^{10}	4^{45}
8	×	×	×	3^{20}	×	×	×	2^{20}	-	3^{10}	×
9	×	×	×	×	×	×	×	1^{10}	3^{10}	-	3^{10}
D	×	×	×	×	×	×	2^{10}	4^{45}	×	3^{10}	-

Here the restricted and unrestricted stations are:

Restricted stations:	1, 2, 3, 5, 7, 8, 9
Unrestricted stations:	4, 6

All the given precedence constraints can be simply written as $(2,\ 5 < 7 < 9)$ and $(1 < 3 < 5,\ 8)$ these constraints lead to the following implications:

$1\to5=\infty$ $1\to8=\infty$	$2\to9=\infty$	$3\to1=\infty$	$5\to1=\infty$ $5\to3=\infty$ $5\to9=\infty$	$7\to2=\infty$ $7\to5=\infty$	$8\to1=\infty$ $8\to3=\infty$	$9\to2=\infty$ $9\to5=\infty$ $9\to7=\infty$

The links not existing due to the structure of the problem:
$9\to2=\infty$, $9\to5=\infty$, $3\to1=\infty$, $5\to1=\infty$ and $8\to1=\infty$.

Hence the reduced distance matrix is

Table 3: Modified cost matrix

Stations	*S*	1	2	3	4	5	6	7	8	9	*D*
S	-	3^{50}	6^{30}	8^{15}	×	×	×	×	×	×	×
1	3^{50}	-	2^{50}	×	2^{25}	×	×	×	×	×	×
2	6^{30}	2^{50}	-	2^{15}	1^{45}	3^{10}	×	8^{15}	×	×	×
3	8^{15}	×	2^{15}	-	×	1^{10}	×	×	3^{20}	×	×
4	×	2^{25}	1^{45}	×	-	×	9^{90}	8^{10}	×	×	×
5	×	×	3^{10}	∞^{10}	×	-	×	5^{60}	×	×	×
6	×	×	×	×	9^{90}	×	-	1^{10}	×	×	2^{10}
7	×	×	∞^{15}	×	8^{10}	∞^{60}	1^{10}	-	2^{20}	1^{10}	4^{45}
8	×	×	×	∞^{20}	×	×	×	2^{20}	-	3^{10}	×
9	×	×	×	×	×	×	×	∞^{10}	3^{10}	-	3^{10}
D	×	×	×	×	×	×	2^{10}	4^{45}	×	3^{10}	-

(i) Finding the first minimum cost route(s):

(a) Construction of the alphabet table:

The alphabet table (table 4) is obtained from the cost-flow matrix. The entries in each column of the alphabet table are the cost values in ascending order with its connected link from the respective column of cost-flow matrix. The flow is moving from source to sink, so that the connections 1 - *S*, 2 - *S* and 3 - *S* are neglected. The first number indicates the link from the respective node; the second number indicates the cost on the same link.

Table 4: Alphabet table

S↓	1	2	3	4	5	6	7	8	9	For *LB* →*D*
1 – 3 2 – 6 3 – 8	2 – 2 4 – 2	4 – 1 1 – 2 3 – 2 5 – 3 7 – 8	5 – 1 2 – 2 8 – 3	2 – 1 1 – 2 7 – 8 6 – 9	2 – 3 7 – 5 3 – ∞	7 – 1 *D*– 2 4 – 9	6 – 1 9 – 1 8 – 2 *D*– 4 4 – 8 2 – ∞ 5 – ∞	7 – 2 9 – 3 3 – ∞	8 – 3 *D*– 3 7 – ∞	6 – 2 9 – 3 7 – 4

(b) Searching mechanism for the optimal solution: The systematic version of the lexicographic search method is presented here to find the optimum solution of the problem (fig.1). The search mechanism is based, on the alphabet table, i.e. it starts from source point and moves towards the sink point via the minimum allowable cost arcs so as to get the maximum flow. The lexicographic search table is shown in table 5.

Table 5: Lexicographic search table (partial) from table 4

Route No.	S↓										Cost
1	$1\underset{(3)}{\overline{\quad}}3$	$2\underset{(5)}{\overline{\quad}}2$	$4\underset{(6)}{\overline{\quad}}1$	$7\underset{(14)}{\overline{\quad}}8$	$6\underset{(15)}{\overline{\quad}}1$	$D\underset{(17)}{\overline{\quad}}2$	23				$17=V_t$
						×					
2					$D\underset{(18)}{\overline{\quad}}4$						$>V_t$
					×						
3				$6\underset{(15)+2}{\overline{\quad}}9$	$7\underset{(16)+2}{\overline{\quad}}1$						$>V_t$
4					$D\underset{(17)}{\overline{\quad}}2$						17
					×						
				×							
5			$3\underset{(7)+2}{\overline{\quad}}2$	$5\underset{(8)+2}{\overline{\quad}}1$	$7\underset{(13)+2}{\overline{\quad}}5$	$6\underset{(14)+2}{\overline{\quad}}1$	$D\underset{(16)}{\overline{\quad}}2$				$16=V_t$
6							$4\underset{(23)}{\overline{\quad}}9$				$>V_t$
							×				
7						$9\underset{(14)+2}{\overline{\quad}}1$	$8\underset{(17)}{\overline{\quad}}3$				$>V_t$
							×				
8						$8\underset{(15)+2}{\overline{\quad}}2$	$9\underset{(18)+2}{\overline{\quad}}3$				$>V_t$
							×				
9						$D\underset{(17)}{\overline{\quad}}4$					$>V_t$
						×					
					×						
10				$8\underset{(10)+2}{\overline{\quad}}3$	$7\underset{(12)+2}{\overline{\quad}}2$	$6\underset{(13)+2}{\overline{\quad}}1$	$D\underset{(15)}{\overline{\quad}}2$				$15=V_t$
(Continued)											
29	$3\underset{(8)+2}{\overline{\quad}}8$	$2\underset{(10)+2}{\overline{\quad}}2$	$5\underset{(13)+2}{\overline{\quad}}3$								$>V_t$
			×								
30		$8\underset{(11)+2}{\overline{\quad}}3$	$7\underset{(13)+2}{\overline{\quad}}2$								$>V_t$
			×								
		×									
	×										

The search is complete.

Note 1: Once the nodes 6, 7 and 9 are completed and the destination (i.e. *D*) yet to reach, then no need to proceed further, because there is no other link connected to *D*.

Note 2: 'NF' indicates 'not feasible'. The feasible routes are those, which start from the source (*S*) and end at the destination (*D*) and obey the capacity restrictions and precedence constraints of the routes.

The first minimum cost is 15 and the available route is:

Table 6: Valid route(s) having minimum cost 15

Route No.	The valid route having minimum cost 15	Max. Flow
10	$(S)\underset{3}{\overset{50}{\rightarrow}}(1)\underset{2}{\overset{50}{\rightarrow}}(2)\underset{2}{\overset{15}{\rightarrow}}(3)\underset{3}{\overset{20}{\rightarrow}}(8)\underset{2}{\overset{20}{\rightarrow}}(7)\underset{1}{\overset{10}{\rightarrow}}(6)\underset{2}{\overset{10}{\rightarrow}}(D)$	10

With minimum cost 15 and maximum flow available in the route number 11 is 10. The following network diagram shows the path:

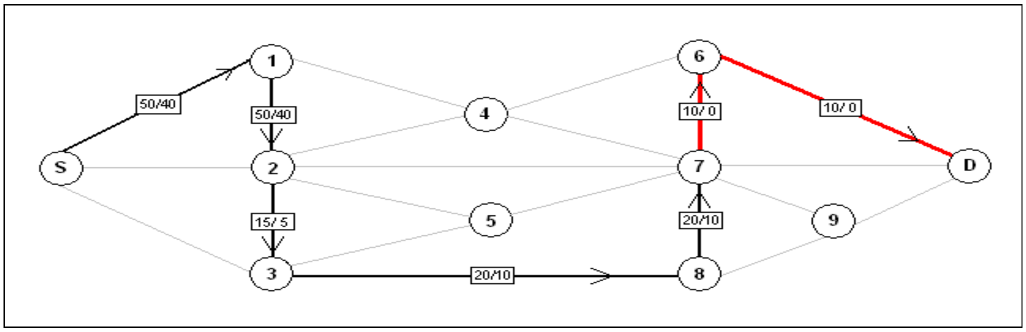

Fig. 4: Network shows the route of cost 15 and flow 10

Thus the links $(7) \leftrightarrow (6)$ and $(6) \leftrightarrow (D)$ indicated by the bold line from Fig.4 are completely busy, and are not used in the subsequent process. Since (6) is deleted, then the link $(4) \leftrightarrow (6)$, is not usable. Also, the routes having minimum costs 16, 17 cannot be utilized. Since, the routes 1, 5 with cost 17 and 16, respectively, contain the link $(6) \rightarrow (D)$, we need not consider those two routes for associated maximum possible flow. The resulting residuals are shown in fig. 5:

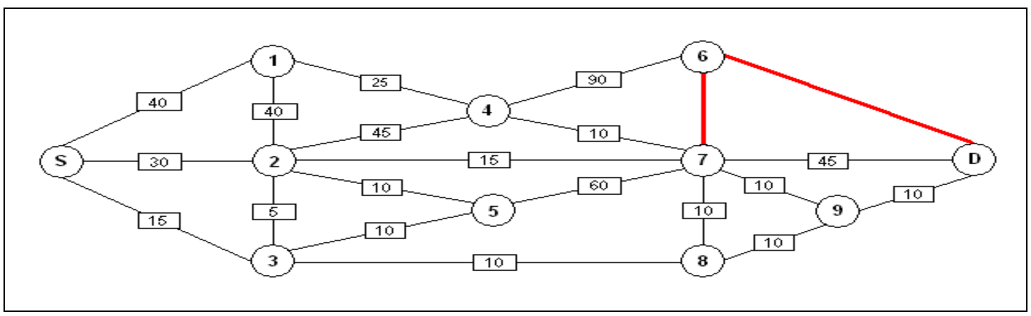

Fig. 5: First modified residual network

(ii) Finding the second minimum cost route(s): Since the links $(7) \leftrightarrow (6)$ and $(6) \leftrightarrow (D)$ are completely busy, therefore, neglecting these links from the further search. This leads to the deletion of node (6) from the next reduced alphabet table. The first revised one can be observed in table 7.

Table 7: 1st Modified alphabet table

$S\downarrow$	1	2	3	4	5	7	8	9	For LB $\rightarrow D$
Predecessor$\rightarrow$	$(1<3)$	$(2<7)$	$(3<5,8)$		$(5<7)$	$(7<9)$			
1 – 3 2 – 6 3 – 8	2 – 2 4 – 2	4 – 1 1 – 2 3 – 2 5 – 3 7 – 8	5 – 1 2 – 2 8 – 3	2 – 1 1 – 2 7 – 8	2 – 3 7 – 5	9 – 1 8 – 2 D – 4 4 – 8	7 – 2 9 – 3	8 – 3 D – 3	9 – 3 7 – 4

Table 8: Lexicographic search table (partial) from table 7

Route No.	S↓										Cost
1	$1\xrightarrow[(3)]{}3$	$2\xrightarrow[(5)]{}2$	$4\xrightarrow[(6)]{}1$	$7\xrightarrow[(14)]{}8$	$9\xrightarrow[(15)]{}1$	$8\xrightarrow[(18)]{}3$					NF
2						$D\xrightarrow[(18)]{}3$					$18=V_t$
						×					
3					$8\xrightarrow[(16)+3]{}2$						$>V_t$
4					$D\xrightarrow[(18)]{}4$						18
					×						
				×							
5			$3\xrightarrow[(7)+3]{}2$	$5\xrightarrow[(8)+3]{}1$	$7\xrightarrow[(13)+3]{}5$	$9\xrightarrow[(14)+3]{}1$	$8\xrightarrow[(17)]{}3$				NF
6							$D\xrightarrow[(17)]{}3$				$17=V_t$
							×				
7						$8\xrightarrow[(15)+3]{}2$					NF
8	$1\xrightarrow[(3)]{}3$	$2\xrightarrow[(5)]{}2$	$3\xrightarrow[(7)+3]{}2$	$5\xrightarrow[(8)+3]{}1$	$7\xrightarrow[(13)+3]{}5$	$D\xrightarrow[(17)]{}4$					17
						×					
					×						
9				$8\xrightarrow[(10)+3]{}3$	$7\xrightarrow[(12)+3]{}2$	$9\xrightarrow[(13)+3]{}1$	$D\xrightarrow[(16)]{}3$				$16=V_t$
							×				
10						$D\xrightarrow[(16)]{}4$					16
						×					
11					$9\xrightarrow[(13)+3]{}3$	$D\xrightarrow[(16)]{}3$					16
						×					
(Continued)											
32	$3\xrightarrow[(8)+3]{}8$	$2\xrightarrow[(10)+3]{}2$	$5\xrightarrow[(13)+3]{}3$	$7\xrightarrow[(18)+3]{}5$							$>V_t$
				×							
33			$7\xrightarrow[(18)+3]{}8$								$>V_t$
			×								
34		$8\xrightarrow[(11)+3]{}3$	$7\xrightarrow[(13)+3]{}2$	$9\xrightarrow[(14)+3]{}1$							$>V_t$
35				$D\xrightarrow[(17)]{}4$							$>V_t$
				×							
36			$9\xrightarrow[(14)+3]{}3$								$>V_t$
			×								
		×									
	×										

The search is complete.

The available routes of second minimum cost 16 and are shown in table 9:

Table 9: valid routes having minimum cost 16

Route No.	The valid routes having minimum cost 16	Max. Flow
9	$(S)\xrightarrow[3]{40}(1)\xrightarrow[2]{40}(2)\xrightarrow[2]{5}(3)\xrightarrow[3]{10}(8)\xrightarrow[2]{10}(7)\xrightarrow[1]{10}(9)\xrightarrow[3]{10}(D)$	5
10	$(S)\xrightarrow[3]{40}(1)\xrightarrow[2]{40}(2)\xrightarrow[2]{5}(3)\xrightarrow[3]{10}(8)\xrightarrow[2]{10}(7)\xrightarrow[4]{45}(D)$	5
11	$(S)\xrightarrow[3]{40}(1)\xrightarrow[2]{40}(2)\xrightarrow[2]{5}(3)\xrightarrow[3]{10}(8)\xrightarrow[3]{10}(9)\xrightarrow[3]{10}(D)$	5

Since all the routes have the same maximum flow (i.e., 5) and using $(2)\leftrightarrow(3)$ with capacity 5, only one route is possible, therefore, we can take any one route from this stage, say, the route (10) and the network diagram of the maximum flow 5 with minimum cost 16 is shown in fig.6:

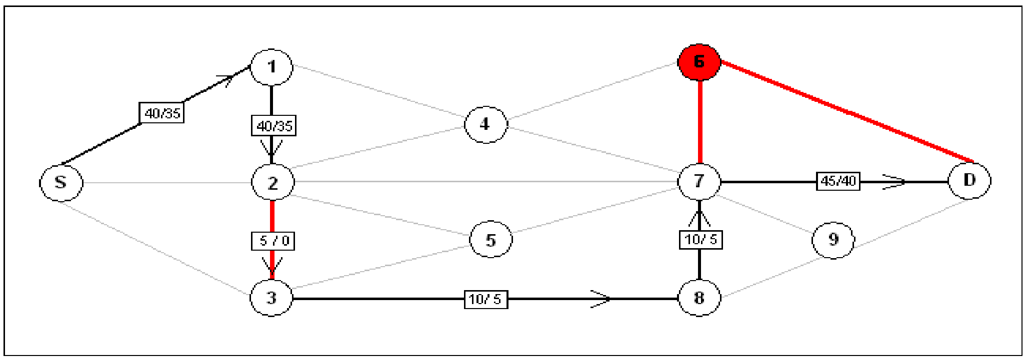

Fig. 6: Network shows the route of cost 16 and flow 5

Since the link $(2)\rightarrow(3)$ is completely busy from fig.4 and we cannot pass any further flow through this link, therefore, the two other valid routes (9) and (11) are being neglected. Because of the same reason routes (6) and (8) with minimum cost 17 are not usable. The remaining available capacities are shown in the network diagram of fig.7:

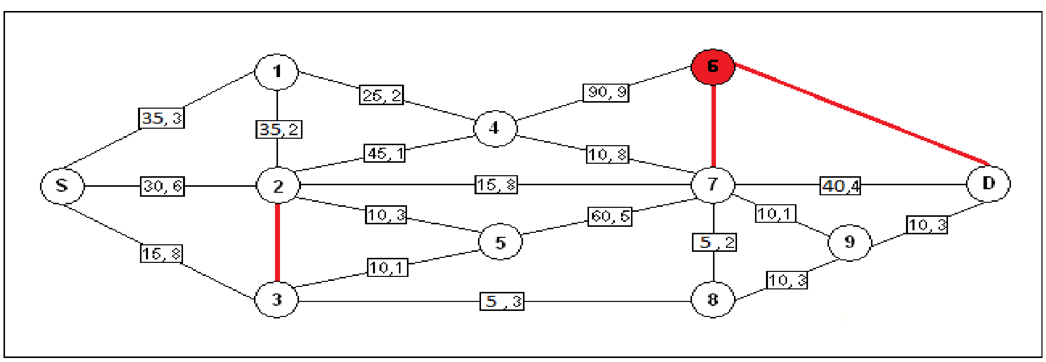

Fig. 7: Second modified residual network

(iii) Finding the third minimum cost route(s): Since the link $(2)\leftrightarrow(3)$ is completely busy and we cannot pass any further flow from this link, therefore, we delete the link $(2)\leftrightarrow(3)$ from the preceding alphabet table and revised alphabet table is shown in table 10.

Table 10: Second modified alphabet table

S↓	1	2	3	4	5	7	8	9	For LB →*D*
Predecessor→	$(1<3)$	$(2<7)$	$(3<5,8)$		$(5<7)$	$(7<9)$			
1 – 3 2 – 6 3 – 8	2 – 2 4 – 2	4 – 1 1 – 2 5 – 3 7 – 8	5 – 1 8 – 3	2 – 1 1 – 2 7 – 8	2 – 3 7 – 5	9 – 1 8 – 2 *D*– 4 4 – 8	7 – 2 9 – 3	8 – 3 *D*– 3	9 – 3 7 – 4

Table 11: Lexicographic search table (partial) from table 10

Route No.	S↓										Cost
1	$1\underset{(3)}{—}3$	$2\underset{(5)}{—}2$	$4\underset{(6)}{—}1$	$7\underset{(14)}{—}8$	$9\underset{(15)}{—}1$	$8\underset{(18)}{—}3$					NF
2						$D\underset{(18)}{—}3$					$18=V_t$
						×					
3					$8\underset{(16)+3}{—}2$						$>V_t$
4					$D\underset{(18)}{—}4$						18
					×						
				×							
5			$5\underset{(8)+3}{—}3$	$7\underset{(13)+3}{—}5$	$9\underset{(14)+3}{—}1$	$8\underset{(17)}{—}3$					NF
6						$D\underset{(17)}{—}3$					$17=V_t$
						×					
7					$8\underset{(15)+3}{—}2$						$>V_t$
8					$D\underset{(17)}{—}4$						17
				×							
9			$7\underset{(13)+3}{—}8$	$9\underset{(14)+3}{—}1$	$8\underset{(17)+3}{—}3$						$>V_t$
10					$D\underset{(17)}{—}3$						17
					×						
11				$D\underset{(17)}{—}4$							17
(Continued)											
17	$1\underset{(3)}{—}3$	$2\underset{(5)}{—}2$	$7\underset{(13)+3}{—}8$	$9\underset{(14)+3}{—}1$	$D\underset{(17)}{—}3$						17
					×						
				×							
18				$D\underset{(17)}{—}4$							17
				×							
(Continued)											
27	$1\underset{(3)}{—}3$	$8\underset{(11)+3}{—}3$	$7\underset{(13)+3}{—}2$	$9\underset{(14)+3}{—}1$	$D\underset{(17)}{—}3$						17
					×						
28				$D\underset{(17)}{—}4$							17
				×							
29			$9\underset{(14)+3}{—}3$	$D\underset{(17)}{—}3$							17
			×								
		×									
	×										

The search is complete.

The available valid routes of cost 17 with the different maximum flow for the same are shown in table 12:

Table 12: Valid route having minimum cost 17

Route No.	The valid route having minimum cost 17	Max. flow
6	$(S)\xrightarrow[3]{35}(1)\xrightarrow[2]{35}(2)\xrightarrow[3]{10}(5)\xrightarrow[5]{60}(7)\xrightarrow[1]{10}(9)\xrightarrow[3]{10}(D)$	5
8	$(S)\xrightarrow[3]{35}(1)\xrightarrow[2]{35}(2)\xrightarrow[3]{10}(5)\xrightarrow[5]{60}(7)\xrightarrow[4]{40}(D)$	10
10	$(S)\xrightarrow[3]{35}(1)\xrightarrow[2]{35}(2)\xrightarrow[8]{15}(7)\xrightarrow[1]{10}(9)\xrightarrow[3]{10}(D)$	10
11	$(S)\xrightarrow[3]{35}(1)\xrightarrow[2]{35}(2)\xrightarrow[8]{15}(7)\xrightarrow[4]{40}(D)$	15
17	$(S)\xrightarrow[3]{35}(1)\xrightarrow[2]{25}(4)\xrightarrow[8]{10}(7)\xrightarrow[1]{10}(9)\xrightarrow[3]{10}(D)$	10
18	$(S)\xrightarrow[3]{35}(1)\xrightarrow[2]{25}(4)\xrightarrow[8]{10}(7)\xrightarrow[4]{40}(D)$	10
27	$(S)\xrightarrow[8]{15}(3)\xrightarrow[3]{5}(8)\xrightarrow[2]{5}(7)\xrightarrow[1]{10}(9)\xrightarrow[3]{10}(D)$	5
28	$(S)\xrightarrow[8]{15}(3)\xrightarrow[3]{5}(8)\xrightarrow[2]{5}(7)\xrightarrow[4]{40}(D)$	5
29	$(S)\xrightarrow[8]{15}(3)\xrightarrow[3]{5}(8)\xrightarrow[3]{10}(9)\xrightarrow[3]{10}(D)$	5

Since, with minimum cost 17, the maximum flow is 15 and is available on the route number (11), therefore we first include the route number (11) in our solution and $(2)\leftrightarrow(7)$ become full, we reject route (10). Similarly, we select arbitrarily from among routes with capacity 10, the route (8) and since $(2)\leftrightarrow(5)$ becomes full we reject route (6). Similarly, we select route (18) and reject route (17). Lastly, we select route (28) and it rejects route numbers (27) and (29). We get the resulting flow network as in fig. 8. The resulting available flows are:

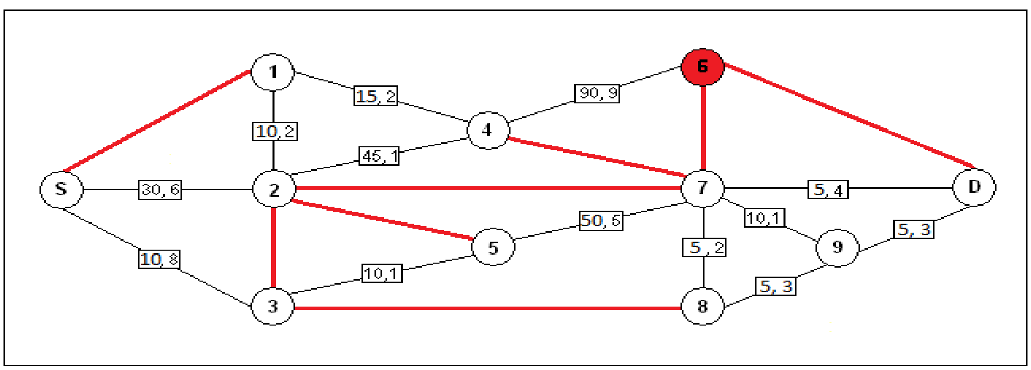

Fig. 8: Third modified residual network

In this step link $(3)\leftrightarrow(8)$ is completely busy. Neglecting the routes having the busy links $(S)\rightarrow(1)$, $(2)\leftrightarrow(3)$, $(2)\leftrightarrow(5)$, $(2)\leftrightarrow(7)$, $(3)\leftrightarrow(8)$, $(4)\leftrightarrow(7)$, $(6)\rightarrow(D)$ & $(7)\leftrightarrow(6)$, then no other valid routes of cost 17 are available.

(iv) Finding the fourth minimum cost route(s): Neglect, all the busy routes of cost 17 from the previous network. The further modified alphabet table is shown in table 13.

Table 13: Third modified alphabet table

$S\downarrow$	1	2	3	4	5	7	8	9	For LB $\rightarrow D$
Predecessor$\rightarrow$	$(1<3)$	$(2<7)$	$(3<5,8)$		$(5<7)$	$(7<9)$			
2 – 6 3 – 8	2 – 2 4 – 2	4 – 1 1 – 2	5 – 1	2 – 1 1 – 2	2 – 3 7 – 5	9 – 1 8 – 2 *D*– 4 4 – 8	7 – 2 9 – 3	8 – 3 *D*– 3	9 – 3 7 – 4

Table 14: Lexicographic search table from table 13

Route No.	$S\downarrow$										Cost
1	$2\underset{(6)}{-}6$	$4\underset{(7)}{-}1$									NF
2		$1\underset{(8)}{-}2$									NF
		×									
3	$3\underset{(8)}{-}8$	$5\underset{(9)}{-}1$	$2\underset{(12)}{-}3$	$4\underset{(13)}{-}1$							NF
				×							
4			$7\underset{(14)}{-}5$	$9\underset{(15)}{-}1$	$8\underset{(18)}{-}3$						NF
5					$D\underset{(18)}{-}3$						$18=V_t$
					×						
6				$8\underset{(18)+3}{-}3$							$>V_t$
7				$D\underset{(18)}{-}4$							18
				×							
			×								
		×									
	×										

The Search is complete.

Table 15: Valid routes having minimum cost 18

Route No.	The valid routes having minimum cost 18	Max. Flow
5	$(S)\overset{10}{\underset{8}{\rightarrow}}(3)\overset{10}{\underset{1}{\rightarrow}}(5)\overset{50}{\underset{5}{\rightarrow}}(7)\overset{10}{\underset{1}{\rightarrow}}(9)\overset{5}{\underset{3}{\rightarrow}}(D)$	5
7	$(S)\overset{10}{\underset{8}{\rightarrow}}(3)\overset{10}{\underset{1}{\rightarrow}}(5)\overset{50}{\underset{5}{\rightarrow}}(7)\overset{5}{\underset{4}{\rightarrow}}(D)$	5

In table 15, the maximum flow is 5 (with minimum cost 18) is available on the route number (5) and (7). Therefore, we can take any one of them; say (5) first, and then we check whether route (7) is possible along with route (5). The allowable flow up to the 7th node is 10 in the specified routes. Now the flow can send from $(7)\rightarrow(D)$ directly and also via the 9th node. So that both the routes can be combined then we get flows of value 5+5=10 and the resulting network is given fig.9:

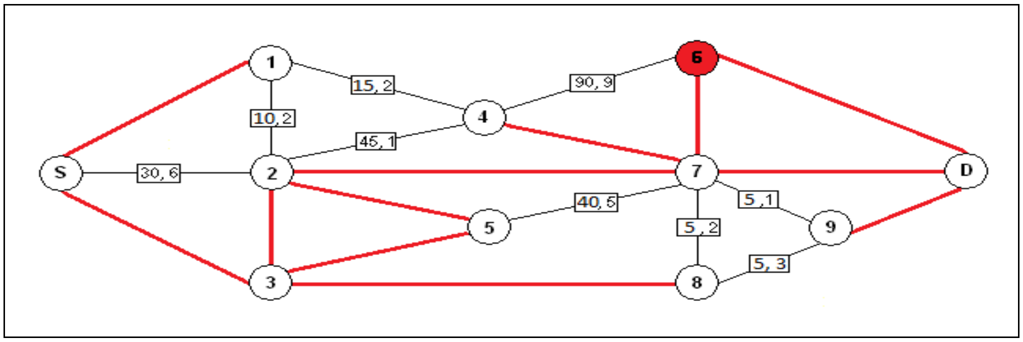

Fig. 9: Fourth modified residual network

It is now observed that from fig.9 all the direct links terminating in destination (*D*) become full, and no further flow will be possible. Therefore, the further search is stopped and we reached at optimal solution. The optimum solution is given in table 16.

Table 16: Routes in the optimum solution

LS table no.	Rout e no.	Routes in the optimum solution	Min cost	Max flow
5	10	$(S)\xrightarrow[3]{50}(1)\xrightarrow[2]{50}(2)\xrightarrow[2]{15}(3)\xrightarrow[3]{20}(8)\xrightarrow[2]{20}(7)\xrightarrow[1]{10}(6)\xrightarrow[2]{10}(D)$	15	10
8	10	$(S)\xrightarrow[3]{40}(1)\xrightarrow[2]{40}(2)\xrightarrow[2]{5}(3)\xrightarrow[3]{10}(8)\xrightarrow[2]{10}(7)\xrightarrow[4]{45}(D)$	16	5
11	11	$(S)\xrightarrow[3]{35}(1)\xrightarrow[2]{35}(2)\xrightarrow[8]{15}(7)\xrightarrow[4]{40}(D)$	17	10
	8	$(S)\xrightarrow[3]{35}(1)\xrightarrow[2]{35}(2)\xrightarrow[3]{10}(5)\xrightarrow[5]{60}(7)\xrightarrow[4]{40}(D)$	17	15
	27	$(S)\xrightarrow[3]{35}(1)\xrightarrow[2]{25}(4)\xrightarrow[8]{10}(7)\xrightarrow[4]{40}(D)$	17	10
	29	$(S)\xrightarrow[8]{15}(3)\xrightarrow[3]{5}(8)\xrightarrow[3]{10}(9)\xrightarrow[3]{10}(D)$	17	5
14	5	$(S)\xrightarrow[8]{10}(3)\xrightarrow[1]{10}(5)\xrightarrow[5]{50}(7)\xrightarrow[1]{10}(9)\xrightarrow[3]{5}(D)$	18	5
	7	$(S)\xrightarrow[8]{10}(3)\xrightarrow[1]{10}(5)\xrightarrow[5]{50}(7)\xrightarrow[4]{5}(D)$	18	5
Total			135	65

Therefore, the maximum possible flow will be 65 with minimum cost 135 units.

8. COMPUTATIONAL EXPERIENCE

A computer program of the algorithm has been developed in C languages and worked out on the system HP COMPAQ dx2280 and Intel Pentium D processor. Random numbers are used to construct the cost matrix. The following table (table: 17) gives the list of the problems tried along with the average CPU run time (in seconds) for solving them.

Table 17: CPU run time (in sec.)

Serial number	Number of stations	No. of problems tried in the respective dimensions	Average CPU runs time (in Sec.)	
			Alphabet table	Search table
1	5	6	0.00000	0.0000
2	8	6	0.00000	0.0000
3	10	6	0.05494	0.0000
4	12	6	0.05494	0.0349
5	15	6	0.08932	0.5812
6	18	6	0.08932	1.3841
8	20	6	0.10989	1.9794
9	25	6	0.10989	2.2649
10	30	6	0.16483	3.0366
11	40	6	0.85389	3.4738
12	50	6	1.54920	4.8643

9. CONCLUSION

It is observed that when there are restricted constraints the flow can be maximized and cost can be minimized by adopting the proposed algorithm and it has immediate application in energy supply and the time required for the search of the optimality is fairly less.

Further, for the efficiency of the proposed algorithm, a large number of problems are tested and it is found that the algorithm is workable in all the cases.

REFERENCES

[1]. Ahmed, N., Das, S. and Purusotham, S. "The problem of maximum flow with minimum attainable cost in a network", OPSEARCH, Vol. 50, No. 2, 197 – 214, 2013.

[2]. Ahmed, N., Das, S. and Purusotham, S. "The oil tankers dispatching problem", OPSEARCH, Vol. 49, No. 4, 366 – 385, 2012.

[3]. Ahmed, N. and Das, S. "A Travelling Salesman Problem with Multiple Job Facilities", OPSEARCH, Vol. 38, No. 4, 394 – 406, 2001.

[4]. Bansal, S. P. and Kumar, S: "Shortest-Route Subject to Improvements through Dynamic Programming" (Privately Circulated), 1970.

[5]. Bansal, S. P. and Kumar, S: "Optimal Tour with Multiple Job Facilities at each Station", Indian Journal of Mathematics, Vol.13, No. 1, PP. 45-49, 1971.

[6]. Bellman: "On a Routing Problem". Quarterly Applied Mathematics, Vol. XVI, No. 1, pp. 87-90, 1958.

[7]. Dantzig, G. B: "Discrete Variable Extremum Problems", Operations Research, Vol. 5, PP. 226-277, 1957c.

[8]. Das, S: "Routing and Allied Combinatorial Programming Problems (A lexicographic Search Approach)", Ph. D. Thesis, Dibrugarh University, 1976.

[9]. Das, S. and Ahmed, N.: "A Travelling Salesman Problem (TSP) with Multiple Job Facilities". OPSEARCH, Vol. 38, No. 4, pp. 394-406, 2001.

[10]. Das, S. and Borah, P. C: "A Lexicographic Search Approach to a Precedence Constrained M-TSP". Paper presented in the 3rd International Conference on Lattices Path Combinatorial & Applications, held at Delhi, 1993.

[11]. Fathabadi, H.S. and Shirdel, G.H: "An O(nm^2) Time Algorithm for Solving Minimal Cost Network Flow Problems", Asia-Pacific Journal of Operational Research (APORS), Vol. 20, No. 2, 2002.

[12]. Ford, L.R., Jr. and Fulkerson, D.R: "Maximal Flow through a Network", Canadian Journal of Mathematics, Vol. 8, PP. 399-404, 1956.

[13]. Ford, L.R.,Jr. and Fulkerson, D.R: "Flows in Network", Princeton University Press, Princeton, N. J., 1962.

[14]. Fulkerson, D.R.: "An Out-of-Kilter Method for Minimal Cost Flow Problems", J. Soc. Indust. Appl. Math., 9, PP: 18-27, 1961.

[15]. Klein, M: "A Primal Method for Minimal Cost Flows with Application to the Assignment and Transportation Problems", Management Science, Vol. 14, No. 3, PP. 205-220, 1967.

[16]. Kumar, S: "Minimal Cost Flow Problem in a Network". Advancing Frontier in Operational Research (Proceeding of the International Seminar), held at New Delhi, 7-10 August, 1967.

[17]. Pandit, S. N. N: "Some Observations on a Routing Problem", Operations Research, Vol. 10(5), PP. 726-727, 1962.

[18]. Pandit, S. N. N: "The Loading Problem", Operations Research, Vol. 10(5), PP. 639-646, 1962.

[19]. Pandit, S. N. N: "Some Quantitative Combinatorial Search Problem". Ph. D. Thesis, IIT, Kharagpur, 1963.

[20]. Peart, R. M., *et al*: "The Shortest Route Problem", Operations Research, Vol. 8, PP. 866-868, 1960.

[21]. Pollack, M. and Walter, W: "Solutions of the Shortest Route Problem – A Review", Operations Research, Vol. 8, PP. 224-230, 1960.

[22]. Purusotham, S and Sundara Murthy, M: "An Exact Algorithm for Multi-Product Bulk Transportation Problem", International Journal on Computer Science and Engineering, Vol. 3, No. 9, PP: 3222-3236, 2011.

[23]. Purusotham, S and Sundara Murthy, M: "A New Approach for solving the Network problems", OPSEARCH, Vol. 49, No. 1, PP: 1-21, 2012.

[24]. Scroggs, R.E. and Tharp, A.L.: "An Algorithm for Solving Travelling Salesmen Problems in Restricted Contexts". (Privately Circulated)..

[25]. Sedeno-Noda, A and Gonzalez-Martin, C: "An Alternative Method to Solve the Bi-objective Minimum Cost Flow Problem", Asia-Pacific Journal of Operational Research (APORS), Vol. 20, No. 2, 2003.

[26]. Saksena, J. P. and Kumar, S: "The Routing Problem with k-specified Nodes", Operations Research, Vol. 14, No. 5, pp. 909-913, 1966.