International Journal of Mathematical Archive-2(12), 2011, Page: 2678-2684
Available online through www.ijma.info ISSN 2229 – 5046

REVISED MAXSMINT ROUTING TECHNIQUE FOR OPTIMIZATION

[*]Brindha G.R[1] and Anand.S[2]

[1]Assistant Professor, ICT Dept, School of Computing, SASTRA University, Thanjavur, Tamil Nadu, India
[2]B.Tech-Final Year student, Mechatronics Dept, SASTRA University,Thanjavur, Tamil Nadu, India

E-mail: grbssk@gmail.com, s.anand90@gmail.com

(Received on: 01-12-11; Accepted on: 19-12-11)

ABSTRACT

***O**ne of the concepts of networks, the shortest path technique has been analyzed extensively for many years. Though there are many techniques to find out the shortest path, only few of them handled the problem based on time. This paper introduces an innovative algorithm, Revised MAXSMINT (Maximizing Speed and Minimizing Time) for finding out the shortest route in a railway network. This new algorithm deals with determining the shortest time taking route which applies depth first search, pruning, backtracking, and also the basic shortest path technique on the time expanded diagraph to optimize time. Our technique is applied to get the optimum time based route on a directed graph. This approach is more comprehensive and is verified to provide greater efficiency and lesser process time than the basic methodologies.*

Keywords: *Shortest route, time optimization, time expanded digraph*

1. INTRODUCTION:

In the present world, time poses as a great constraint in all walks of life. Keeping this in mind, it is difficult to spend time in finding the shortest route by looking at the map and calculating the time required to reach the destination while travelling in trains. Due to this limitation, the passengers end up travelling in a route that takes a longer time or avoid the hassle by resorting to other means of transportation to reach their destination. In the recent years a number of works deal with the automation of this process. The problem statement is that the passenger has to travel from a particular station X to the destination station Y in the shortest possible time which is provided by the shortest path. The shortest path is calculated by considering the travelling time between two consecutive stations and thus the summation of the travelling time between the particular station and the destination will give the time period and the shortest path calculation continues in this manner.

Shortest path problems have considerable realistic implications in such areas as Information technology, operations research computer science and transportation engineering, etc. Standard shortest path problems have been studied intensively, resulting in the progress of a number of efficient algorithms, with single source linear time concept [1] [5] [15]. In one of the previous papers, besides Dijkstra's algorithm [15] and Dreyfus's algorithm [5], more connected issues and the difference of single-source problem have been intensively analyzed in the areas of shipping and networks [10] [11] [14]. The Bellman-ford based algorithm [12] considers edge weight concept which can be used on graphs with negative edge weights, till the graph contains no negative cycle accessible from the source node. A* algorithm is a generalization of basic shortest path technique that slash down on the size of the sub graph which must be explored. The extended A* algorithm focuses [9] priority queue for all paths to be expanded based on arc cost. In order to deal with a large network, the graph is divided into small parts and to form the shortest-paths. Such partitioning methods are studied such as disjoint edge-set partition [6][7] and disjoint node-set partition [8]. More number of authors had analyzed different variation of the least cost shortest path. These includes, multiple shortest paths in least cost [2] with single source [13] and turn constraints [16].Recently, Bolin et. al [3] presented an algorithm, denoted as Tdsp08. in this paper, to find a shortest path with a best departure time, it focuses on piecewise linear functions regarding implementations and performance studies. Compared to our previous work MAXSMINT shortest path technique [4] this Revised MAXSMINT takes lesser processing time, so that execution speed is increased.

The rest of this paper is ordered as follows. Section II analyzes the requirements by taking a sample railway map as an example. Section III describes the Revised MAXSMINT algorithm and explains it with examples. Section IV demonstrates the soundness of the proposed scheme with system management and transition diagram using Revised MAXSMINT. Section V depicts the simulation analysis with proper comparative charts. Conclusions and areas of future expansions are presented in Section VI.

Corresponding author: [*]Brindha G. R[1], *E-mail: grbssk@gmail.com

2. REQUIREMENT ANALYSIS:

The entire map of the railways network is required for the proper functioning of the system. The time schedule of the trains and also all the information pertaining to the various functionalities of the trains is constantly updated to the main server network. Thus, the system has all the information pertaining to the railways network, like the junctions, the departure and arrival times and the trains that are available. The starting station and start time is taken as input to the system from the user and the shortest route to the destination is determined by the system. Figure 1 shows the route map of a city. The assumptions are Each railway stations are nodes and the distance between the stations are arcs.
(Source: http//homepage.univie.ac.athorst.prillingermetroenglishnetwork_maps.html)

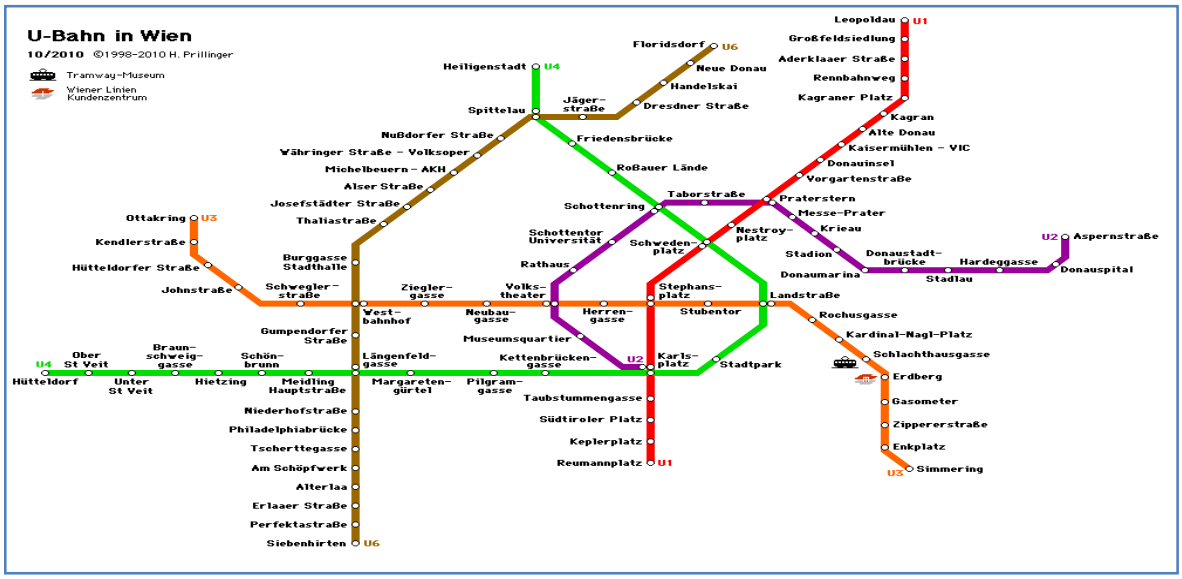

Fig.-1 Route Map of a sample city

2.2 Time- expanded digraph:

Time-expanded digraph method is adopted here where all the nodes are assumed to be consisting of clusters of arrival time and departure time. Since the passenger may change the train within a station, the arrival cluster should be connected to all the possible points in the departure cluster. For each station X(i), there are several arrival points A(x) and departure points D(y). The arrival and departure points correspond to the time of arrival and departure respectively.

2.3 Preliminaries and Node postulation:

Each node has a set of in-degree and out-degree nodes associated with them. The nodes which lead to or arrive at the current node are known as in-degree or arrival nodes. The nodes which have the path which are having the path which are departing from the current node are known as out-degree nodes or the departing nodes. These in-degree and out-degree nodes are used as the arrival and departure points of the node in the time-expanded digraph of the map.
A major deviation from the original MAXSMINT algorithm is that in the original work, each station is considered as a node and a search technique is used to determine the shortest route. This calculation constitutes a major portion of the processing time. The time required for processing is greatly reduced in the revised version of the algorithm by considering the stations which act as junctions as the nodes and all the stations that are in between the junctions as elements of an array which represents the particular line (eg. North-west line). Thus search technique is used for the nodes and the array elements are compared to determine if the final node has been reached. All the paths that lead from the source node to the destination node are determined through this process.

After this, the shortest time taking path has to be determined. This is done with the help of the time-expanded digraph which is constructed using the data from the schedule of the trains. The main motive of the system is to help the user to determine the shortest arrival time at the destination node. Thus, the shortest departure time at the current node (n^{th}) and the shortest arrival time at the next node ($n+1^{th}$) are considered for the calculation of the shortest time taking path. The arrival time of the $n+1^{th}$ node should be such that it is greater than the departure time of the current node. Once the time is determined for a path, pruning is done to eliminate the path that takes a longer time. This process is continued till the shortest time taking path is determined.

3. ALGORITHM:

```
Begin
Preliminaries:
For i=1 to total no. of juctions
Assign  Node id, Arrival_Time, Departure_Time  //Assign indegree, outdegree  for each node//
End for
Get arrays of nodes for the stations between junctions. //Arrays accessed by both junction node id and line id//
Get Source_Node , Destination_Node and  Start_Time
Assign
Destination_Node=Current_Node
Nodes{}=insert _first(indegree (current_node))
Declare
Fringe{},Non_Leading{};
Level[m][n]={} where m-=1 to possible levels, n=1 to no. of nodes ;

Level Extration:
While(not empty)Nodes{}
{ If (Current_Node is in Non_Leading{})
 remove from fringe
break
Else
Nodes{}=insert_first(indegree(current_node))
Endif
Check for source in array
If(source in array
Copy till source node to fringe{}
Break
Else
Copy array to fringe{}
If(Nodes{i} \*first node*\= fringe{})
remove node from Nodes{}
break
endif
//Backtracking with DFS and pruning:
 if (node{i}=Source_node)
        for level 1 to m
                for all nodes n in fringe{}
                        assign L[m][n]= fringe{}
Break
endif
if(indegree(current_node)=0)
        remove node from fringe{}
        remove node from nodes{}
        remove node from indegree
        add node to n{}
break
else
        current_node=first(nodes{})
        remove node from node{}
        assign node to first(fringe{})
endif
}

Time Based optimization with Prunning:
Copy L'[m][n]=L[m][n]
Current_Node_Time=start_time
For each level m in L'[m][n]
For all node n
If(n=Destination_node)
                Journey_time=Current_Node_time
                Break
If Departure_time(n)>Arrival_time(n) //Compare (schedule,Current_Node_time)//
        Current_node=n+1
```


If(Arrival_time≥Goal_time)
Remove m
Break
Display Journey_time
Display L'[m][n], the level m which leads to Journey_time (optimal)

First, the inputs are accepted and the various nodes and arrays are fetched from the database. The initial time, source and destination node are the inputs that are taken in by the system. Since the system employs search technique that starts from the destination, the destination node is taken as the current node and the in-degree nodes (the node which leads to the current node) of the current node is considered as the next junction that is closest to the current node. The array between the in-degree node and the current node is checked for the presence of the source node. If the source node is not present, then the search continues by taking the first in-degree node as the current node and the next junction as its in-degree node. The array between the two junctions is checked for the source node and the search proceeds in the same manner. The nodes or the junctions that do not lead to the source node are placed in the non-leading list and are constantly compared with the in-degree nodes so as to eliminate the nodes that do not lead to the source. Thus, by using the search technique, the various routes to the destination can be determined. The next step in this process is to determine the route which takes the shortest time. This is done by taking the first route and calculating the time taken by it. This is used as base and compared when calculating the time taken by the next route. If at any node, the cumulative time exceeds that of the base time, it is eliminated or pruned. If the time taken by another route is shorter, it is taken as the base. Finally, the base time and its route are displayed to the user.

In Figure 2 contains 4 paths in which the initial node and the final node is indicated in blue colour. Further it shows the achievable routes from the source node to the destination node. Table-1 explains the possible path in terms of nodes for above said graph and the travelling time for that corresponding path. From this resulting paths based on the optimum time calculated is selected, i.e. path 1 through which you can reach the destination within 232 minutes which is less than the other three paths.

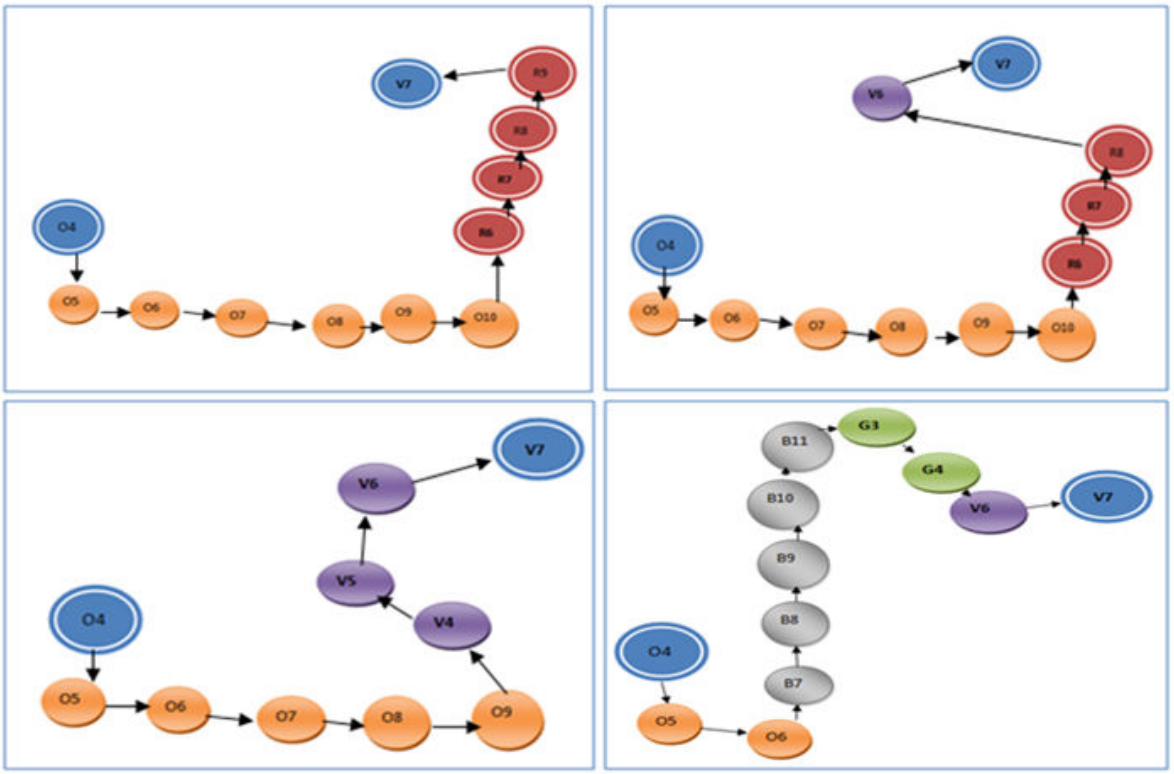

Fig.-2 Achievable routes from source to destination

Table-1 Achievable Routes Details with Time Duration

S.no	Nodes Travelled	Traveling Time -min
1	*O4-O5-O6-O7-O8-O9-R6-R7-R8-R9-V7*	*232*
2	O4-O5-O6-O7-O8-V3-V4-V5-V6-V7	265
3	O4-O5-O6-O7-O8-O9-V4-V5-V6-V7	270
4	O4-O5-O6-B7-B8-B9-B10-B11-G3-G4-V6-V7	284

4. SYSTEM ARCHITECTURE AND MANAGEMENT:

Figure 3 depicts the functional diagram of Revised MAXSMINT algorithm and Figure.4 describes the state transition diagram for the technique Revised MAXSMINT.

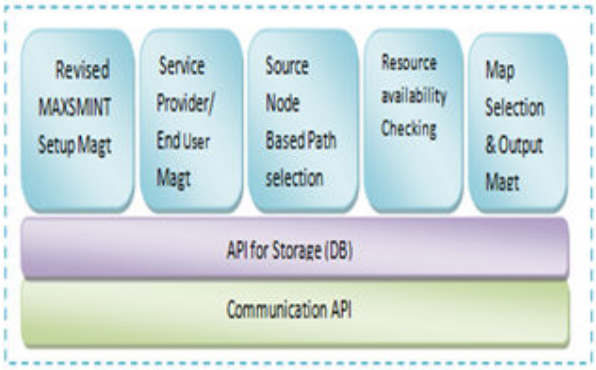

Fig. - 3 Functional Diagram of Revised MAXSMINT

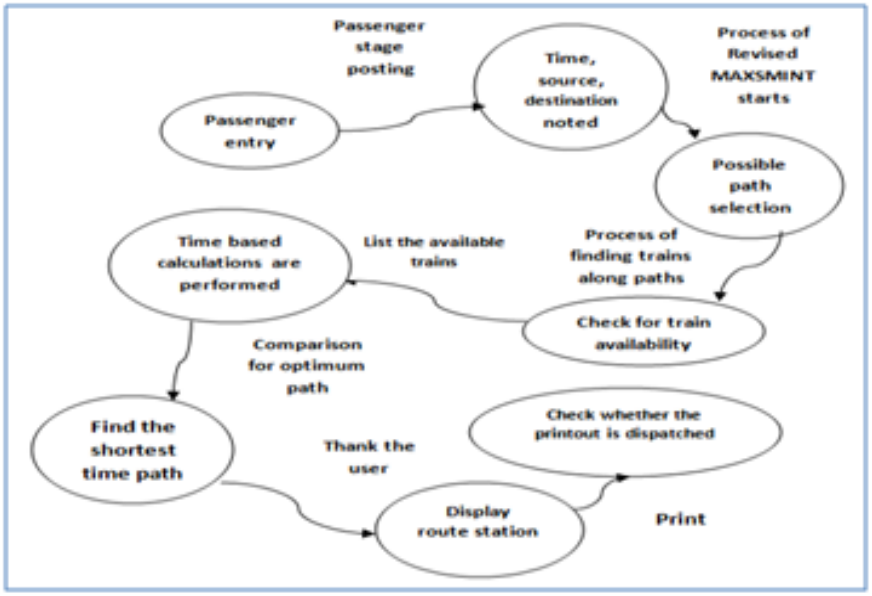

Fig. - 4 State Transition Diagram of Revised AXSMINT

5. SIMULATION ANALYSIS:

In the end user screen as shown in Figure.5 the user should enter the date, time, starting node and the destination node.

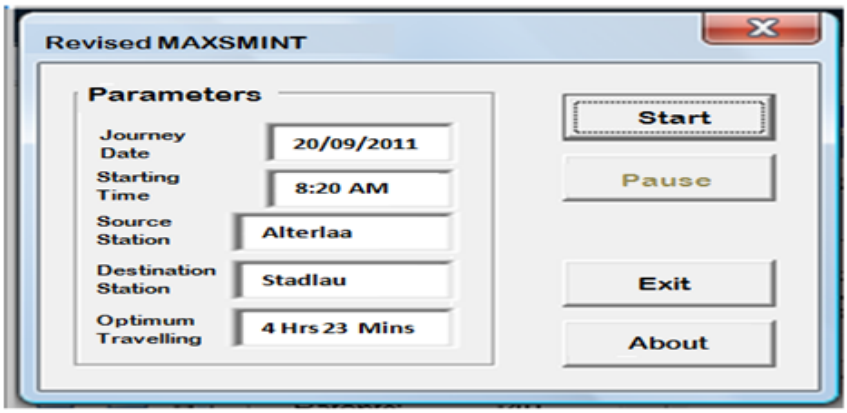

Fig.- 5. Revised MAXSMINT Simulation GUI

The algorithm will analyze the all achievable combination for the nodes then from the obtained possibilities, so that it will compute the less travelling time which can be achieved by the route.

The final map will be displayed as depicted in figure.6. Fig. 7 exhibits the comparative chart of the time taken by the users who used our Revised MAXSMINT, MAXSMINT and those who did not use.

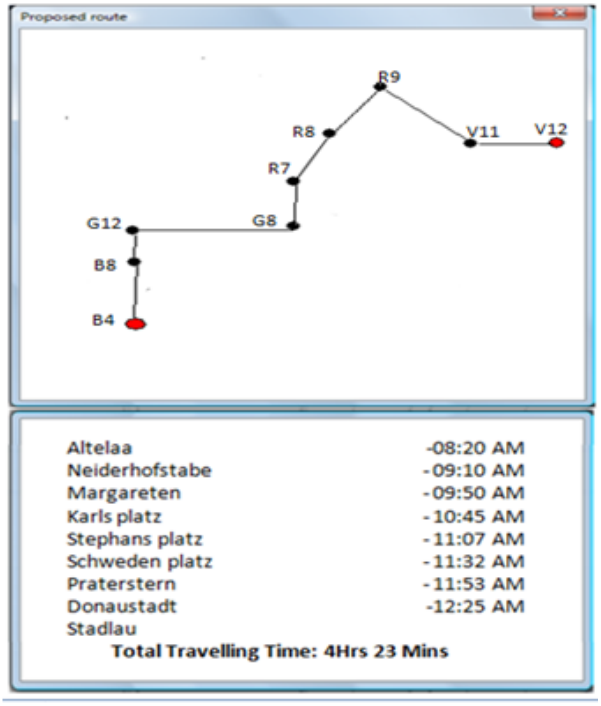

Fig.-6. Revised MAXSMINT Route Sheet

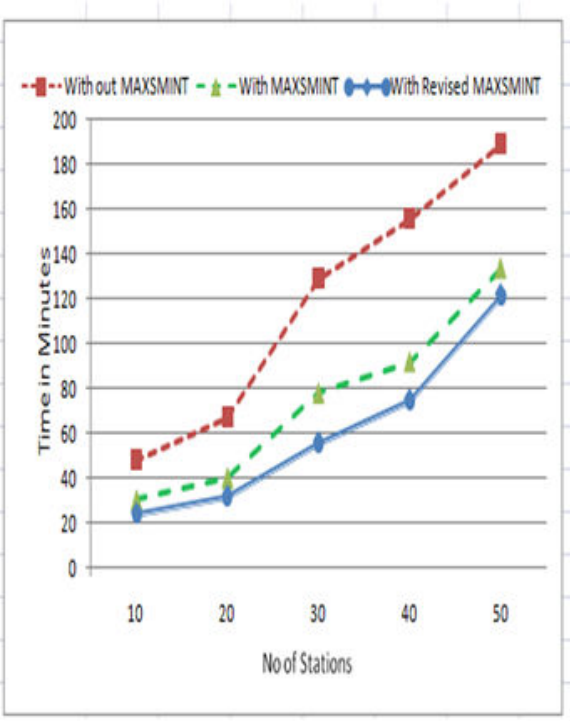

Fig.-7. Time consumption – Without, with MAXSMINT & with Revised MAXSMINT

6. CONCLUSION AND FUTURE EXPANSION:

With the current world giving more emphasis to time optimization, the novel technique that is presented in this paper serves as an effective application in the present day scenario. This technique allows increased optimization in travelling through trains and reduction the travelling time of the passengers. There is also inherent flexibility in the technique that allows it to be adapted in various other scenarios and applications. This technique can also be extended to bus services by incorporating that network into the system. Apart from time, cost also proves to be a major factor in the current scenario and many works have gone into the reduction of cost or in increasing the cost efficiency of the system. Thus, this technique can be modified to include the cost factor into it such that the route which has the lowest cost budget is determined.

REFERENCES:

[1] Ahuja.R.K, Mehlhorn.K, Orlin.J.B, and Tarjan.R.E, Faster algorithms for the shortest path problem, Journal of the ACM, 1990, 37(2), pp. 213-223.
[2] Bolin.D and Xu.Y.J and Lu.Q, Finding time-dependent shortest paths over large graphs, in Proceedings of the 11th international conference on Extending database technology, France, March 2008, pp. 205-216.
[3] Boroujerdi, A., Uhlmann, J.: An efficient algorithm for computing least cost paths with turn constraints. Information Processing Letters 67, 317–321 (1998); Elsevier Science (November 1997)
[4] Brindha.G.R., S.Anand,V.Gosakan, PM.Joeprathap: An Innovative routing Technique to optimize time and speed, The International Conference on communication Technology and System Design, ICCTSD 2011, Amritha School of Engineering Coimbatore.
[5] Dreyfus.S.E. An appraisal of some shortest-path algorithms. Operations Research, 1969, 17(3), pp. 395-412
[6] Jing.N, HuangY.W , and Rundensteiner.E.A, Hierarchical encoded path views for path query processing: An optimal model and its performance evaluation, IEEE Trans. Knowl. Data Eng., 1998, 10(3), pp. 409-432.
[7] Jing.N, HuangY.W , and Rundensteiner.E.A, Hierarchical optimization of optimal path finding for transportation applications, In Proc. Of ACM Conference on Information and Knowledge Management, 1996, pp. 261-268.

[8] Jung.S and Pramanik.S, Hiti graph model of topographical roadmaps in navigation systems, In Proc. of the 12th International Conference on Data Engineering 1996, pp. 76-84.
[9] Kanoulas.E, Du.Y, Xia.T, and Zhang.D, Finding fastest paths on a road network with speed patterns. In Proc. of the 22nd International Conference on Data Engineering, 2006, pp. 10-19.
[10] Narv¢ez.P, Siu.K.Y, and Tzeng.H.Y, New dynamic algorithms for shortest path tree computation, IEEE/ACM Trans. Netw., 2000, 8(6), pp. 734-746.
[11] Narv¢ez.P, Siu.K.Y, and Tzeng.H.Y, New dynamic SPT algorithm based on a ball-and-string model, IEEE/ACM Trans. Netw., 2001, 9(6), pp. 706-718.
[12] Orda.A and Rom.R, Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length, Journal of the ACM, 1990, 37(3), pp. 607-625.
[13] Rees, W.G.: Least-cost paths in mountainous terrain, Scott Polar esearch Institute, University of Cambridge, Lensfield Road, ambridge CB2 1ER, UK. Computers & Geosciences 30, 203–209 (2004)
[14] Sung.K, Bell.M.G, Seong.M, and Park.S, Shortest paths in a network with time-dependent flow speeds, European Journal of Operational Research, 2000, 121(12), pp. 32-39.
[15] Thorup.M, Undirected single-source shortest paths with positive integer weights in linear time, Journal of the ACM, 1999, 46(3), pp. 362-394.
[16] Trtiff, J.L.: An experimental comparison of two distributed single-source shortest path algorithms. Parallel Computing, 1505–1532 (April 1995); Elsevier Science

****

