Journal of Applied Sciences 11 (16): 2916-2923, 2011
ISSN 1812-5654 / DOI: 10.3923/jas.2011.2916.2923
© 2011 Asian Network for Scientific Information

A Framework to Recover Single Link Failure on Shortest Path in Shortest Path Tree

Muhammad Aasim Qureshi, Mohd Fadzil Hassan, Sohail Safdar and Rehan Akbar
Department of Computer and Information Science, Universiti Teknologi PETRONAS, Perak, Malaysia

Abstract: Risk management in many applications become very important for the uninterrupted continuation of the process. Shortest path and many related path planning problems play a vital role in many applications like robot navigation, games, transportation and communication routing. In problems like these and many other, efficient and reliable recovery from the adverse situation with minimum additional healing cost and delay is always required. This study presents a framework to construct shortest path tree with embedded backups for single link failure on any location in shortest path from source to destination. This tree entertain one link failure at one point of time. The cost of the presented algorithms is as low as any shortest path tree algorithm. So it is providing additional feature in the same cost.

Key words: Single link failure recovery, shortest path, embedded backups, risk management

INTRODUCTION

Risk is a factor that always remain there when we start any execution whether simulation or a real life travel. To overcome the risks of failures and the situations leading to failure it is often suggested to have backup plans. These plans, if prepared before the execution, will help in overcoming the risk situations and will provide alternate execution route for the system. Shortest path and recovery algorithms are being considered as another methodology to tailor a software process (Akbar *et al*., 2011a).

Many important areas of computer science like robot navigation, network communication and vehicle routing need robust and reliable processing and hardly afford failure or collapse of the plan. Path planning is the backbone of these systems. In these applications, most optimal path that leads to safe and reliable accomplishment of job (packet transfer in network communication and object's movement to destinations in the other two cases) is always desired. Theoretical aspect for the solutions of these and many other practical problems lie under the research area of shortest path in theoretical computer science (Qureshi *et al*., 2010, 2011; Qureshi and Hassan, 2010).

In graph theory, the shortest path problem is the problem of finding an optimized path between two vertices (or nodes) such that the sum of the weights of its constituent edges is minimized (Kwon *et al*., 2007). An example is finding the quickest way from one location to another in a road network like Ibrahim (2007) presented his technique of finding shortest paths in traversing some locations within the Sokoto metropolis, in such applications vertices represent locations and edges represent segments of road that are weighted by the time required to travel on that road segment.

Formally, given a weighted graph G (V, E) with $V = \{v_1, v_2, v_3, v_4, \ldots, v_n\}$ and $E = \{e_1, e_2, e_3, e_4, \ldots, e_m\}$ such that $|V| = n$ and $|E| = m$. A shortest path tree can be defined as a union of all paths, connected, starting from source s, to all other vertices including destination, t with minimum most total edge cost at each path.

Shortest Path Tree (SPT) Problem plays a very vital role in solving a wide range of applications like robot navigation, games, vehicle routing and networks. Even in intrusion threats it has its application (Safdar *et al*., 2011). In project management the shortest cost providing schemes are selected (Akbar *et al*., 2011b). Its robust and effective solution plays an important role in the solution of the problem (s) and performance of its application. In fact, the shortest path problem is extensively applied in communication, computer systems, transportation networks and many other practical problems (Zhang *et al*., 2006). Shortest path algorithms also support the claim of the shortest processes for development. Its solution leads to the management control of flow in workflow processes efficiently. In short Shortest Path Problem helps us in finding optimal solutions for problems in many diverse areas (Khan *et al*., 2009).

Single Source Shortest Path (SSSP) is one of the hard areas of theoretical computer science (Qureshi and Maqbool, 2007a, b) but at the same time being a classic problem. It is one of the most attempted and famous problems among researchers. Most of the study is based on the Dijkstra (1959) which was presented in 1959. There are many variations of shortest path problem; Hershberger *et al*. (2007) discussed the difficulty of

Corresponding Author: Muhammad Aasim Qureshi, Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, Perak, Malaysia

different algorithms in 2007. SPT is also used in Energy-Aware Cluster-Based Routing (EACBR) in Wireless Sensors and Networks (Dai *et al.*, 2009). In short Pettie *et al.* (2002), Thorup (2000, 1998) and Raman (1997) and many other researchers (Alivand *et al.*, 2008; Zaferanieh and Fathali, 2009) contributed in one way or the other to the solution(s) of this problem.

Risks can be seen everywhere around us and we always handle those risks with our planning. "Risk is a concept that denotes a potential negative impact to some characteristic of value that may arise from a future event or we can say that reisks are events or conditions that may occur and whose occurance, if it does take place, has a harmful or negative effect" (Mojtahedi *et al.*, 2009). Risks and risk management have gained high important functioning of many modern societies and contemporary organizations. In accademic discourse and desciplines as diverse as business management, finance and economics, public administration, energy policy, psycology, socilogy, antropology, health care, education, people are really looking onto it to manage and avoid risks and risk questions (Attar, 2010, 2011).

Link failure is a common problem in communication and vehicle routing. A link can fail which may lead to system hang or system crash. In order to recover from such collapses systems are equipped with capabilities to handle such vulnerable situations and adopt new path for continuation of the execution. Recovery from such link failure(s) at run time is one of the enormous SSSP based problems. Its applications can be seen in transportation networks as well as in communication networks (Baker and Gokhale, 2007). This problem can formally be defined as follows:

Given a graph G = (V, E) where V = $\{v_1, v_2, v_3, v_4,..., v_n\}$ and E = $\{e_1, e_2, e_3, e_4,..., e_m\}$ such that $|V|$ = n and $|E|$ = m. G is an undirected non-negative weighted connected graph, with pre-specified source vertex s and destination vertex t such that s, tϵV. Let Cost (PG (s, t)) be the cost of the path P in graph G from s and t. We have to find a tree T = $\{t_1, t_2, t_3,..., t_k\}$, where $t_i \epsilon E$ for all i = 1 to k, rooted at s, that have alternate shortest paths on the failure of any link on shortest path (from the very point of failure on the shortest path from source to destination).

Path detection is widely used in the area of navigation systems and path tracking system. Jeong *et al.* (2009) presented a heuristic algorithm for searching the alternative paths instead of finding one optimal paths. Their algorithm is based on the renewed path partition detection method and used reversed (Dijkstra, 1959) algorithm to find the cost of the paths. This algorithm overcome the limitation of K-shortest path algorithm which provides very similar alternative paths and gives a solution to find dissimilar alternative paths based on user specified conditions.

Roditty and Wick (2005) presented a O(m$\sqrt{}$n) time algorithm to find the replacement paths from the unweighted directed graphs. The work presented by Roditty and Wick (2005) takes two variations of the problem into account, restricted replacement path problem and edge replacement path problem. The idea is to find the best short detour and best long detour for every edge in the path and determine the most optimal replacement paths by using these detours. In order to find the shortest detour (Dijkstra, 1959) algorithm is used. The study also provides an improvement to k-shortest path problem and provides a new randomize running time O (km$\sqrt{}$nlogn) for k-shortest path problem. The limitation of this study is that it only works with unweighted graphs or small weight values.

Medard *et al.* (1999) presented an algorithm which was referred as MFBG algorithm (after the names of authors) (Xue *et al.*, 2003). In MFBG it was required to traverse back to the root node and restart execution with a different path in order to recover from the link failure. Chen *et al.* (2008) introduced a distributive approach to decrease the delay in recovering the network after a single link failure. They introduced an improved algorithm based on MFBG algorithm (presented by Medard *et al.* (1999) in which execution did not have to restart the from the root node.

Krasikov and Noble (2004) proposed a proof to find out the best available path next to the shortest path for undirected weighted graphs in O (n^3) time. This is the path having a length strictly greater than the shortest path in the set of all shortest paths.

Bhosle and Gonzalez (2003) addressed the single link failure recovery problem and presented an O(m+n logn) time algorithm for undirected weigthed 2-connected graphs and proposed that the complexity of this algorithm could be reduced to O(m+n) if the all the edge weights are equal. This study has also shown that the solution to a single link failure recovery can be used in an alternate path routing problem. Bhosle and Gonzalez (2008) presented a distributed algorithm for finding an alternate path in an edge weighted 2-connected graphs.

Nardelli *et al.* (2003) addressed another problem related to shortest path which is finding a most vital node in 2-connented undirected graphs which have positive weights and presents an O(m+n logn) time algorithm.

A O(n) times algorithm was presented by Bhosle (2005) but that was only for planar undirected graphs while the same paper presented a O (m+n. α (2n, n)) times algorithm for undirected graphs with integral edge weights. Bhosle and Gonzalez (2003) suggested, that a solution to single link recovery problem can be used for alternate path routing mechanism that is applied in ATM networks. Li *et al.* (2008) presented an

O (mn+mn^2 log n) time optimal shortest path tree that plays an important role in single link failure recovery.

Proposed scheme of Zhou *et al*. (2011) was claimed to provide protection for at least three types of failures, i.e., the ONU link failure, the OLT link failureand the OLT failure.

FRAME WORK

Framework shown in Fig. 1 has three components out of which first and second can performe in parallel or in other words we can say that first and second are independent of each other while third step depends upon the results of the former two.

First component-SPT Generator: In this component shortest path tree is needed to be calculated. Any comparison based shortest path tree algorithm can be used to get this tree. The inputs for this component will be the graph and the output will be a shortest path tree from source to all other vertices including destination. The output shortest path tree will be stored in an array of size |V|. Figure 2 shows the resultant tree where the red nodes and red-Thick lines show the intermediate vertices and links of shortest path, from s to t while black-thick-dashed links show the intermediate links between the shortest path source and other vertices. To keep it easy to understand weights and other details are not shown.

Second component--Reverse SPT generator: As any shortest path tree algorithm calculates a tree of all shortest paths starting from source to destination. But here tree from all other vertices to the destination is required. If source and destination can be replaced then this tree can be found easily. The only thing that is required is to swap the two vertices, source and destination.

```
Temp  =  s
s     =  t
t     =  Temp
```

After swapping the two any single source multiple destination shortest path tree algorithm can be applied. The resultant SPT will be providing the tree but with a difference. In normal SPT parents are accessed through children but in this tree children would be accessible from the vertex in hand.

Figure 3 shows the resultant tree in which the red nodes and red-Thick lines show the intermediate vertices and links of shortest path, from s to t while green-thick-dashed-dot links show the intermediate links between the shortest path from destination and other vertices. To keep it easy to understand weights and other details are not shown.

Third component--Merger: This component is dependent on the results of the two previous components. In this component resultant trees of first and second component are linked and merged together in such a way that the backup paths are calculated and embedded in the tree. Following subroutine named Merger() will perform the said operation and generate the shortest path tree with embedded backs for single link failure recover for all collapses that take place on the shortest path.

Two trees, lets call them T_1 and T_2, will be convolved/merged on graph and the edges will be marked on the basis of the following rules:

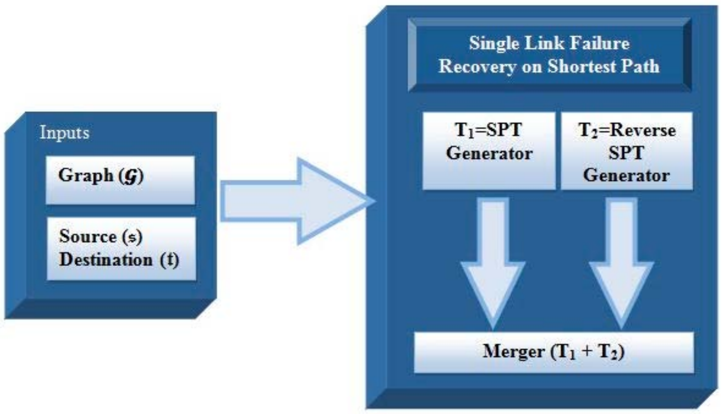

Fig. 1: Framework for single link failure recovery at shortest path from source to destination

- **Rule 1:** Follow vertex by vertex from P
- **Rule 2:** Ignore the red and green edge from the neighbors
- **Rule 3:** Find a neighbor with the least cost leading to a black vertex in T_1, if there
- **Rule 4:** Neighbor should not be predecessor but can be successor
- **Rule 5:** If no other edge then pick green edge, if there and repeat Rule 1 to Rule 5

Figure 4a and b show, graphically, the merging/ convolution process and Fig. 4c shows the final resultant tree in which doted blue line depict backup edges that lead to best back up path available at that point while the green-dashed lines show the links that lead onwards from the first backup edge.

This subroutine will generate a tree of backup paths i.e., B. B combined with T of first step will single source shortest path tree with embedded backups.

Expected outcome: It is expected that the algorithms designed on the basis of this framework will be able to generate a tree that will have all paths starting from source and ending at all vertices with minimum possible total cost of each path. At the same time it will also be able to have all the alternate paths from every vertex on shortest path and ending at destination. Figure 5 shows the details of the outcome of the framework. If in case L_{AD}

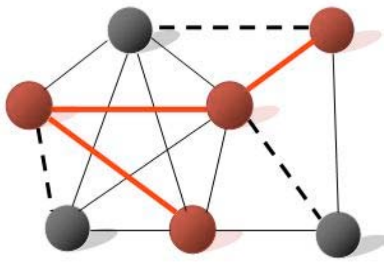

Fig. 2: Resultant tree of SPT generator. Red-Thick Lines shows the links that on the shortest path from s to t. Black-dashed-thick lines show other shortest path tree links

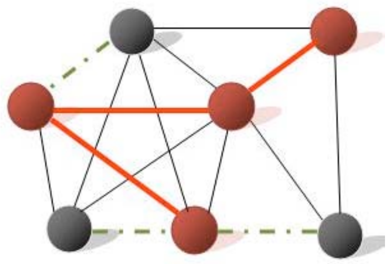

Fig. 3: Resultant tree of Reverse generator. Red-Thick Lines shows the links that are on the shortest path from s to t. Green-dashed-dot-thick lines show other shortest path tree links

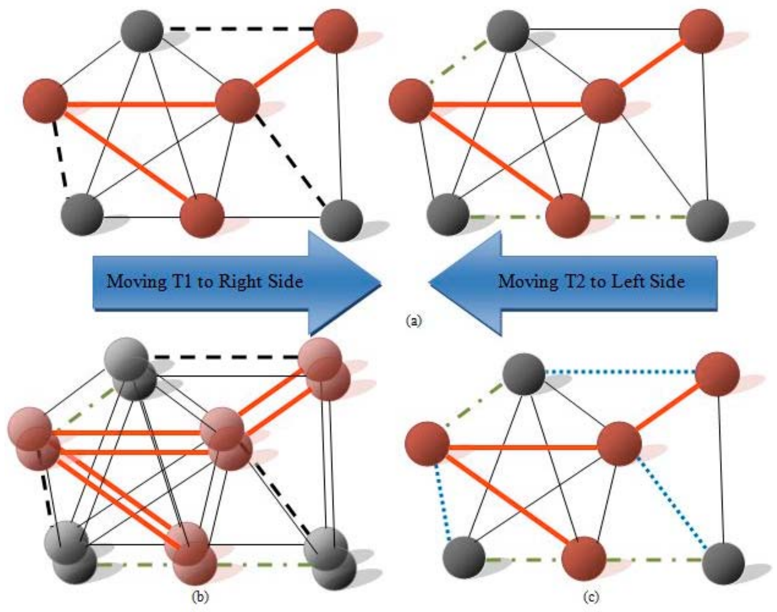

Fig. 4 (a-c): (a) Process of merging the two graphs figuratively to make it easy to understand, (b) Overlapping the two trees. Based on the rules new back up edges can be calculated now and (c) Based upon the rules the new deted-blue links are marked as backup edges and are embedded in the tree to have back up of every available link on the shortest path from s to t fails then L_{AB} can be opted without wasting time and then from there L_{BD} can be used to get back to the shortest path. If L_{DG} fails then L_{DI} can be opted from where the L_{IH} and L_{HJ} will lead to the destination. But if L_{GJ} collapses then traversal can be continued through links L_{GE}, L_{EH} and L_{HJ}.

Algorithm based on framework: On the basis of the framework an algorithm is devised that will accept a graph and two vertices-source and destination. Any shortest path tree routine can be used in the algorithm that accepts a graph and two vertices-source and destination and return a single source multiple destination shortest path. Three components now become three steps. Complete algorithm can be seen in Fig. 6.

VERIFICATION AND VALIDATION

To verify the framework and expected outcome in the form of tree an example is being solved. In this example a connected undirected positive edge weight graph with 12 vertices and 22 links is taken. To depict a natural picture links and their weights are generated randomly.

Figure 7a presents the initial pictorial representation of the graph with the said properties. Figure 7b shows the results of first step of the algorithm just designed based on the framework under discussion. A single source multiple destination shortest path tree is generated and is shown with all thick lines. The red-solid-thick lines show the links that lead from source to destination with minimum most cost. Black-dashed-thick lines presents links that lead to the shortest paths from source to other vertices.

Figure 7c shows the results of second step of the algorithm just designed based on the framework under discussion. A single source multiple destination shortest path tree is generated and is shown with all thick lines where source and destination vertices are swapped. The red-solid-thick lines show the links that lead from source (originally destination) to destination (originally source) with minimum most cost. Green-dashed-thick lines presents links that lead to the shortest paths from source (originally destination) to other vertices.

Figure 7d presents the final resultant tree with embedded back up links; red-thick-solid links show the shortest path from source to destination while doted-thick links present first link that lead to the next best available path to continue traversal on the green links. Green lines present the links making shortest paths from other vertices to destination vertex.

To verify and validate the results on the basis of what we were expecting. Here we can see that if L_{AC} is not available then it can traverse through L_{AG} and then can continue its traversal on the path marked with green-dashed lines. If at vertex link L_{CD} fails or no more in

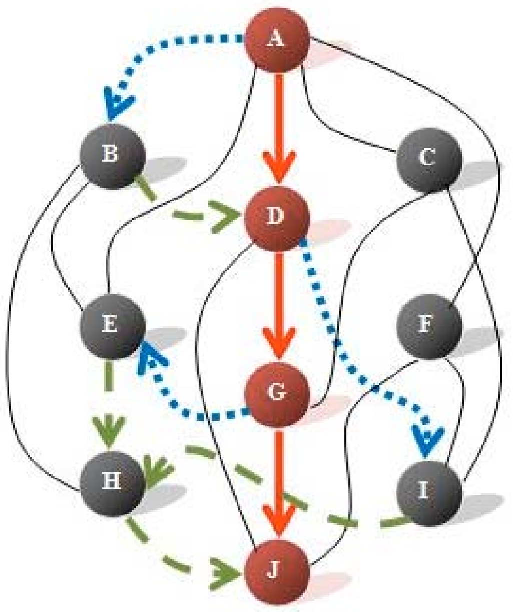

Fig. 5: Graph shows shortest path in thick-solid lines while doted-thick lines show the first link that lead to the alternate path that will guarantee the shortest path

```
Tree $T_1$ = SPT (Graph G, vertex s, vertex t)
Tree $T_2$ = SPT (Grsph G, Vertex t, vertex s)
min→ -1
min-Node →$\phi$
B ← $\phi$
B ← B $\cup$ s
I ← 0
u ← s
while u $\neq$ t
        For each v $\in$ Adj [u]
          If v $\notin$ $Pred_u$
          then
            If mmin > $\Delta_1$ [u] + $\Delta_2$ [v] + $e_{u,v}$
            then
                min ← $\Delta_1$ [u] + $\Delta_2$ [v] + $e_{u,v}$
                min-Node ← v
            Else
                next-Node ← v
      B ← B $\cup$ min-node
      i ← i + 1
      u ← $v_i$
```

Fig. 6: Complete algorithm designed on the basis of the framework where B is the tree with backup links

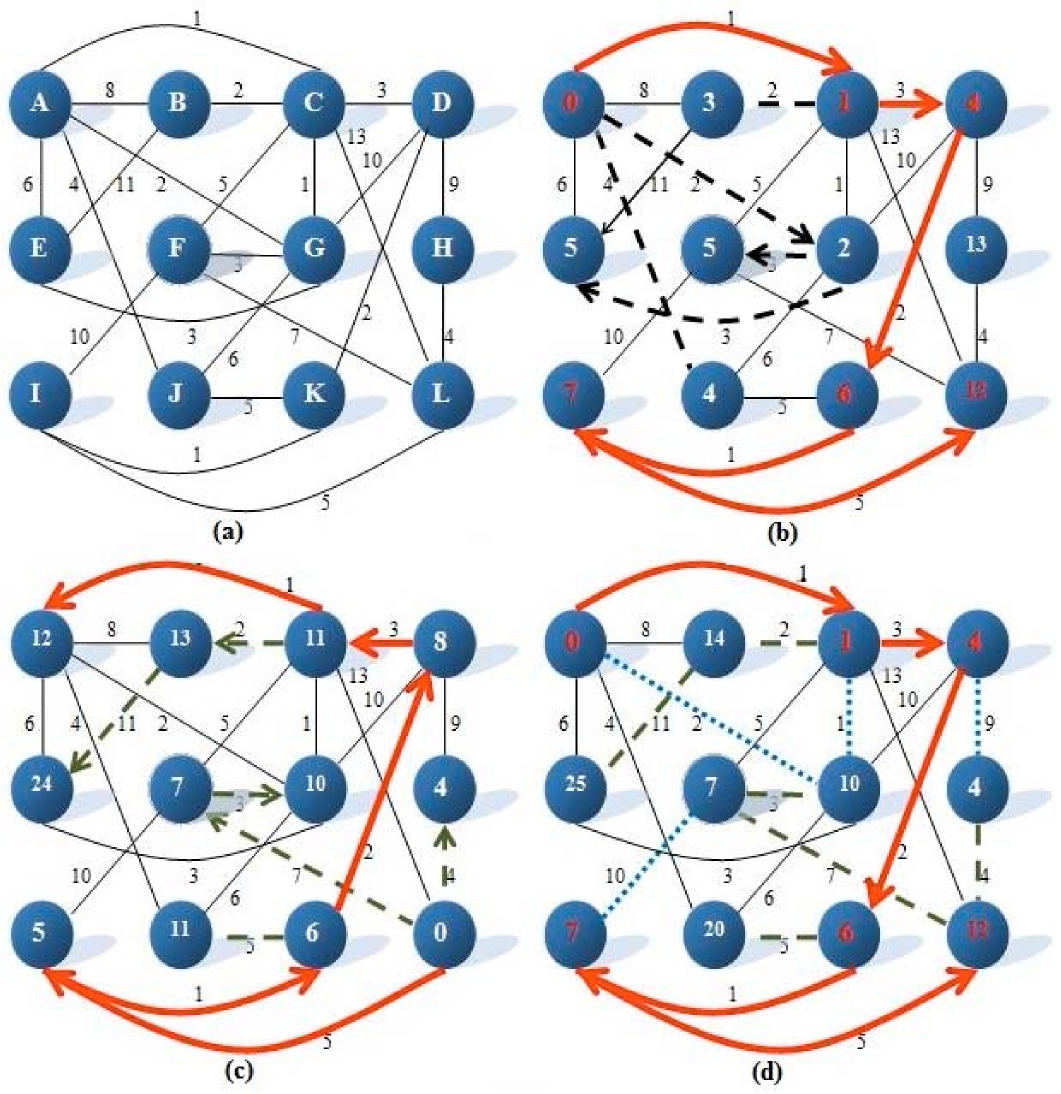

Fig. 7 (a-d): (a) original graph with 12 vertices and 22 links, (b) final result of step-I; thick-solid links show the shortest path links on the shortest path from source to destination while thick dashed lines show the other tree links contributing in the shortest paths of other vertices, (c) final result of step-II; thick-solid links show the shortest path from destination to source while thick dashed lines show the other tree links contributing in the shortest paths of other vertices and (d) final resultant tree with back up links; thick-solid links show the shortest path from source to destination while doted-thick links presets first link that lead to the next best available path to continue traversal on the green links

working condition then traversal can be continued through L_{CG} and then the traversal can be continued on the path marked with green-dashed lines to destination successfully. Onwards for the backup link for L_{DK} is L_{DH} and for L_{KI},L_{KJ} is providing backup. For L_{IL}'s failure the backup path starts with L_{IF}. So it verifies the framework and validates its practical application in the provision of backup paths.

T (n) = T (step-I) + T (step-II) + T (step-III)
T (step-1) = O (E + V log V)
time required to complete a SPT alogrithm
(T (step-II) = O (E) + O (E + V log V)
O (E) time is required to built G' by deleting edges.
O (E + V logV) is time to complete a SPT algorithm
T (step -III) = O (V + E)
a BFS like traversal required to mark all levels
T (n) = T (step-I) + T (step + T (step-II) + T (step-III)
T (n) = O (E + V log V) + O (E) + O (E + V log V) + O (V + E)
T (n) = O (E + V log V) + O (E) + (O (E + V log V) + O (V + E)
T (n) = O (E + V log V + E + E + V log V) + V + E)
=O (4 E + V + 2 V log V)
≃ O (E + V log V)

Fig. 8: Step by step calculation of the complexity of the algorithm

TIME COMPLEXITY

Total time complexity will be the sum of the independent time required to complete each step. The algorithm complete its execution in O (E + V log V) time. Step by step calculation of the complexity can be seen in Fig. 8.

CONCLUSION

This study has presented a framework through which recovery from single link failures is made possible. It overcomes the problems like delay in calculating alternate shortest path on runtime. In case if some link failure takes place, on runtime, not only the traversal can be continued without any delay but at the same time, the time required on all backups can also be seen which may be advantageous in applications for respective decision making.

Though algorithm needs little extra space but it is inevitable to hold all information regarding backup paths.

Its time complexity proven above shows that it is not only asymptotically efficient but also practically suitable in applications like transportation networks and robot navigation. It does not have high constant factor which normally slows down the speed of algorithms when it comes to practical situation. There exist no such work available in such detail on theoretical level.

REFERENCES

Akbar, R., M.F. Hassan, A. Abdullah, M.A. Qureshi and S. Safdar, 2011a. Directions and advancements in global software development a summarized review of GSD and agile methods. Res. J. Inform. Technol., (In Press).

Akbar, R., M.F. Hassan, M.A. Qureshi and S. Safdar, 2011b. Structured role based interaction model for agile based outsourced IT projects: Client's composite structure. Inform. Technol. J., 10: 1009-1016.

Alivand, M., A.A. Alesheikh and M.R. Malek, 2008. New method for finding optimal path in dynamic networks. World Applied Sci. J., 3: 25-33.

Attar, H., 2010. Product innovation and the games of uncertainty and risk. J. Applied Sci., 10: 801-812.

Attar. H., 2011. The scientific approaches to risk and risk management: A critical review. Trends Applied Sci. Res., 6: 386-393.

Baker, Z.K. and M. Gokhale, 2007. On the acceleration of shortest path calculations in transportation networks. Proceedings of the 15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, April 23-25, IEEE, pp: 23-34.

Bhosle, A.M. and T.F. Gonzalez, 2003. Efficient algorithms for single link failure recovery and its application to ATM networks. Proceedings of the 15th IASTED International Conference on PDCS, (IICP'03), Marina del Rey, CA, USA., pp: 87-92.

Bhosle, A.M. and T.F. Gonzalez, 2008. Distributed algorithms for computing alternate paths avoiding failed nodes and links. Eprint arXiv:0811.1301, abs/0811.1. http://arxiv.org/PS-cache/arxiv/pdf/0811/0811.1301v1.pdf.

Bhosle, A.M., 2005. Improved algorithms for replacement paths problems in restricted graphs. Operat. Res. Lett., 33: 459-466.

Chen, W., L. Kong, D. Jin and L. Zeng, 2008. Reducing average delay after link failure in single-link recovery trees using distributed switching mechanism. AEU Int. J. Elect. Commun., 62: 235-238.

Dai, Z., Z. Li, B. Wang and Q. Tang, 2009. An energy-aware cluster-based routing protocol for wireless sensor and actor network. Inform. Technol. J., 8: 1044-1048.

Dijkstra, E.W., 1959. A note on two problems in connection with graphs. Numerische Math., 1: 269-271.

Hershberger, J., S. Suri and A.M. Bhosle, 2007. On the difficulty of some shortest path problems. ACM Trans. Algorithms, 3: 1-15.

Ibrahim, A.A., 2007. Graph algorithms and shortest path problems: A case of djikstra`s algorithm and the dual carriage ways in Sokoto metropolis. Trends Applied Sci. Res., 2: 348-353.

Jeong, Y.J., T.J. Kim, C.H. Park and D.K. Kim, 2009. A dissimilar alternative paths-search algorithm for navigation services: A heuristic approach. KSCE. J. Civil Eng., 14: 41-49.

Khan, L., N. Ayub and A. Saeed, 2009. Anycast based routing in vehicular adhoc networks (VANETS) using vanetmobisim. World Applied Sci. J., 7: 1341-1352.

Krasikov, I. and S.D. Noble, 2004. Finding next-to-shortest paths in a graph. Inf. Process. Lett., 92: 117-119.

Kwon, W., I.H. Suh, S. Lee and Y. Cho, 2007. Fast reinforcement learning using stochastic shortest paths for a mobile robot. Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, Oct. 29-Nov. 2, San Diego CA., pp: 82-87.

Li, Y., Z. Nie and Z. Xiaohong, 2008. Finding the optimal shortest path tree with respect to single link failure recovery. Proceeding of the 4th International Conference on Networked Computing and Advanced Information Management, Sept. 2-4, IEEE Computer Society, Washington, DC. USA., pp: 412-415.

Medard, M., S.G. Finn, R.A. Barry and R.G. Gallager, 1999. Redundant trees for preplanned recovery in arbitrary vertex-redundant or edge-redundant graphs. IEEE/ACM Trans. Network., 7: 641-652.

Mojtahedi, S.M.H., S.M. Mousavi and A. Aminian, 2009. A non-parametric statistical approach for analyzing risk factor data in risk management process. J. Applied Sci., 9: 113-120.

Nardelli, E., G. Proietti and P. Widmayer, 2003. Finding the most vital node of a shortest path. Theoret. Comp. Sci., 296: 167-177.

Pettie, S., V. Ramachandran and S. Sridhar, 2002. Experimental evaluation of a new shortest path algorithm. Proceedings of the 4th Workshop on Algorithm Engineering and Experiments, (ALENEX'02), Springer-Verlag, Berlin, Heidelberg, pp: 126-142.

Qureshi, M.A. and O. Maqbool, 2007a. The complexity of teaching: Computability and complexity. Proceedings of the International Conference on Teaching and Learning, (ALT'10), Springer-Verlag Berlin, Heidelberg, pp: 209-223.

Qureshi, M.A. and O. Maqbool, 2007b. The complexity of teaching: Computability and complexity. INTI J. Special Issue Teaching Learnn., 1: 171-182.

Qureshi, M.A. and M.F. Hassan, 2010. Improvements over two phase shortest path algorithm. Proceedings of the 4th International Symposium on Information Technology, (ITSIM'10), Bandar Seri Iskandar, Malaysia, pp: 683-688.

Qureshi, M.A., M.F. Hassan, S. Safdar and R. Akbar, 2010. Two phase shortest path algorithm for non-negative weighted undirected graphs. Proceedings of the International Conference on Communication Software and Networks, Feb. 26-28, Singapore, pp: 223-227.

Qureshi, M.A., M.F. Hassan, S. Safdar and R. Akbar, 2011. A near linear shortest path algorithm for weighted undirected graphs. IEEE Symposium on Computers and Informatics KL. Malaysia

Raman, R., 1997. Recent results on the single-source shortest paths problem. SIGACT News, 28: 81-87.

Roditty, L. and U.Z. Wick, 2005. Replacement paths and k simple shortest paths in unweighted directed graphs. Proceedings of the Liam Roditty, Uri Zwick Replacement Paths and Simple Shortest Paths in Unweighted Directed Graphs, (ICALP'05), ACM, Transsections, pp: 249-260.

Safdar, S., M.F. Hassan, M.A. Qureshi and R. Akbar, 2011. Effective methods for secure authentication in workflows under intrusion threat. Inform. Technol. J., (In Press).

Thorup, M., 1998. Floats Integers and Single Source Shortest Paths. Symposium on Theoretical Aspects of Computer Science Paris, France.

Thorup, M., 2000. Floats, integers and single source shortest paths. J. Algorithms, 35: 189-201.

Xue, G., S. Member, L. Chen and K. Thulasiraman, 2003. Quality-of-service and quality-of-protection issues in preplanned recovery schemes using redundant trees. Selected Areas Commun. IEEE J., 21: 1332-1345.

Zaferanieh, M. and J. Fathali, 2009. Ant colony and simulated annealing algorithms for finding the core of a graph. World Applied Sci. J., 7: 1335-1341.

Zhang, B., J. Zhang and L. Qi, 2006. The shortest path improvement problems under Hamming distance. J. Combinatorial Optim., 12: 351-361.

Zhou, X.L., F.X. Yu, Y.C. Wen and Z.M. Lu, 2011. A novel protection architecture scheme for EPON. Inform. Technol. J., 10: 591-596.