International Journal of scientific research and management (IJSRM)
||Volume||2||Issue||12||Pages||1801-1806||2014||
Website: www.ijsrm.in ISSN (e): 2321-3418

Avoid Link Failure using Multipath Routing Technique

V.Yashwanth[1], M.Appa Rao[2]

[1]Aditya Institute of Technology and Management,
Tekkali
yashwanth.valasala@gmail.com

[2] Aditya Institute of Technology and Management,
Tekkali
apparaomukalla@gmail.com

Abstract: We present a noval method to find an alternative path, after a link failure between source node to destination node. In this project we are using "Multipath routing scheme" & Dijkstra's algorithm which is a promising routing scheme to accommodate these requirements by using multiple pairs of routes between a source and a destination. Multipath routing is the routing technique of using multiple alternative paths through a network, which can yield a variety of benefits such as increased bandwidth, or improved security. We are also using the Dijkstra's algorithm for the calculation of the shortest distance between the nodes for improving the performance of the routing path.

Keywords: Dijkstra's algorithm, routing scheme

1.Introduction

1.1 Routing in the Network

Routing is the process of selecting best paths in a network. In the past, the term routing was also used to mean forwarding network traffic among networks. However this latter function is much better described as simply forwarding. Routing is performed for many kinds of networks, including the telephone network (circuit switching), electronic data networks (such as the Internet), and transportation networks. This article is concerned primarily with routing in electronic data networks usingpacket switching technology.In packet switching networks, routing directs packet through intermediate nodes.

Intermediate nodes are typically network hardware devices such as routers, bridges, gateways, firewalls, or switches. General-purpose computers can also forward packets and perform routing, though they are not specialized hardware and may suffer from limited performance. The routing process usually directs forwarding on the basis of routing tables which maintain a record of the routes to various network destinations. Thus, constructing routing tables, which are held in the router's memory, is very important for efficient routing. Most routing algorithms use only one network path at a time. Multipath routing techniques enable the use of multiple alternative paths.In case of overlapping/equal routes, the following elements are considered in order to decide which routes get installed into the routing table

1.Prefix-Length: where longer subnet masks are preferred

2.Metric: where a lower metric/cost is preferred (only valid within one and the same routing protocol)

3.Administrative distance: where a lower distance is preferred (only valid between different routing protocols)

Routing, in a more narrow sense of the term, is often contrasted with bridging in its assumption that network addresses are structured and that similar addresses imply proximity within the network. Structured addresses allow a single routing table entry to represent the route to a group of devices. In large networks, structured addressing (routing, in

the narrow sense) outperforms unstructured addressing (bridging). Routing has become the dominant form of addressing on the Internet. Bridging is still widely used within localized environments.

1.2 Minimum cost Path Routing

Finding the minimum-cost path from an origin to a destination in a network is a well-studied problem. A slight variation to the basic Dijkstra's approach is the symmetrical or bi-directional Dijkstra algorithm (Cherkassky et al 1993)[1].It performs a forward search from the origin and a backward search from the destination simultaneously, in an attempt to reduce the search complexity. When the underlying network is Euclidean, another search technique called the A* algorithm is often used (Sedgewick and Vitter 1986). The key idea is to integrate the inherent geometric information in order to bias a more directed search towards the destination. Although the A* algorithm on average improves the run time over the Dijkastra's algorithm, the solution issub-optimum, i.e., it does not always find the minimum-costpath.The minimum-cost path problems have significant applications in many fields of Intelligent Transportation Systems (ITS), especially in Advanced Traffic Management Systems (ATMS) and Advanced Traveler Information Systems (ATIS) (Ziliaskopoulos and Mahmassani 1993). For example, an essential part of most in-vehicle Route Guidance Systems (RGS) currently under development ison-line calculation of the optimum route between a designated origin-destination (O-D) pair (Bekhor et al 2001).

Herein the optimum route is usually the one with least expected travel time between the given O-D pair. The RGS consists of two sub-systems:the centralized system and the decentralized systems. The centralized system updates the link travel time regularly and calculates the optimum route in real-time upon receiving drivers' requests[2]. The optimum route is then relayed to the decentralized systems for the drivers' use.

However, there are many situations that finding the minimum-cost path is not sufficient[3]. Instead, it is necessary to identify some alternative paths for a given O-D pair. For example, currently most drivers are not equipped with the RGS and hence do not have complete information of the traffic network. Studies have shown that in such cases the route decisions are stochastic (Fu and Rilett 1998). In other words, the actual route taken by a particular driver is not necessarily the minimum-cost path. As another example, assume the perfect case where all drivers are guided by the RGS. In reality, some drivers may prefer one particular route to another for various reasons such as route familiarity, toll charges, etc. Therefore, it is more desirable that the RGS suggests several routes for each O-D pair and lets the driver make the choice. In addition, for transportation management purposes there are needs to take alternatives to the minimum-cost path. One particular example is that under emergency situations there are often excessive traffic demands from the affected areas to the safe areas. Part of the traffic should be diverted off the minimum-cost route in order to reduce congestions and minimize the total evacuation time. Finally, finding alternative paths has important applications in transportation planning and analysis processes. In particular, stochastic traffic assignment models typically encompass a route generation phase, in which the set of efficient routes is generated for each O-D pair (Dial 1969). Then atthe next phase traffic demand is assigned appropriately among those paths.

1.3 Current Approaches

Current approaches for finding alternative paths fall into two main categories. The first is known as the K shortest paths method, where the goal is to identify the first, second, …, and Kth best path for a given O-D pair (Yen 1971). It was first proposed in the 1950' (Bellman 1958), and ever since then numerous algorithms have been developed as improvement. The biggest disadvantage of the K shortest paths method is its computational complexity (Yen 1971). It tends to run fairly slow, especially when large network is involved. This impedes its potential applications in ATIS where real-time route calculation is essential. Yet another drawback of it is that the generated paths tend to be quite similar, meaning they havea high percentage of sharedlinks (Scott 1997). This is often undesirable because congestions may develop at the shared links if all traffic flow between the O-D pair is routed via the K shortest paths.

The second strategy for finding alternative paths involves a penalty concept (Scott 1997). Given an O-D pair, the algorithm

first calculates the minimum-cost path using standard Dijkstra's approach[4]. Then one or more links comprising the best path are chosen to be penalized by some factor λthat is greater than one, meaning that the cost of each link is multiplied by λ. The newly calculated minimum-cost path for the updated network is regarded as an alternative to the best path. It has been shown that the penalty method is able to run as fast as the standard Dijkstra's algorithm, and it tends to avoid the similarity problem that the K shortest path method suffers (Scott 1997). As an extreme variation of the penalty method is the link elimination method (Azevedo et al 1993). In this case every chosen link is removed from the network rather than increasing its cost. Both the penalty method and the elimination method have disadvantages. First the algorithm must explicitly specify which particular links should be chosen for penalization or elimination. This choice usually affects the quality of the generated path. In the case of link elimination, it is even possible that deleting some links may cause the network to lose the necessary connectivity for having any path between the O-D pair. Another issue is that only the links comprising the initial best path are modified while the rest of the whole network remains unaffected. Therefore, the method tends to find only a very limited number of alternative paths. In other words, if the method is applied many times in an attempt to generate different paths, presumably it will keep finding a small set of paths over and over again. In this paper, a new technique is proposed for generating alternative paths in traffic networks. The new approach is designed to achieve two main objectives. The first is that it must be computationally efficient so as to be applicable for large real traffic networks. The second goal is that the generated paths should be diverse and efficient. The paths are considered diverse if they do not overlap with one another significantly. The paths are considered efficient if none is associated with an unacceptable high cost. Both criterion are carefully defined and evaluated in this study.

2.Module Description :

2.1 Novel Routing Technique:

We now present the details of the method. Let $G = (N, A)$ be an undirected connected graph with node set N and arc set A. For $x \in N$, let $N(x)$ is the set of neighbors of x, where a neighbor of x is a node one arc away from x. We associate with each undirected arc $(i, j) \in A$ a cost $c(i, j)$, and require each $c(i, j)$ to be a positive integer. (The integer valued restriction can always be met by approximating, to the desired accuracy, each arc cost by an improper fraction, and then multiplying all the fractions by the least common multiple of the fraction denominators.) For $i, j \in N$, let $c\_(i, j)$ be the cost of the shortest path in G between i and j. When using *Route*(s, d) for fast re-route in the event of an arc failure, which is the target application, $c\_(i, j)$ represents the shortest path cost *before* the IGP has reconverged in response to the link failure[8].

4.2 Multipath Routing:

Multipath routing is a promising routing scheme to accommodate these requirements by using multiple pairs of routes between a source and a destination. Multipath routing is the routing technique of using multiple alternative paths through a network, which can yield a variety of benefits such as increased bandwidth, or improved security. The multiple paths computed might be overlapped, edge-disjointed or node-disjointed with each other. Extensive research has been done on multipath routing techniques[10].

2.2 Failure Recovery

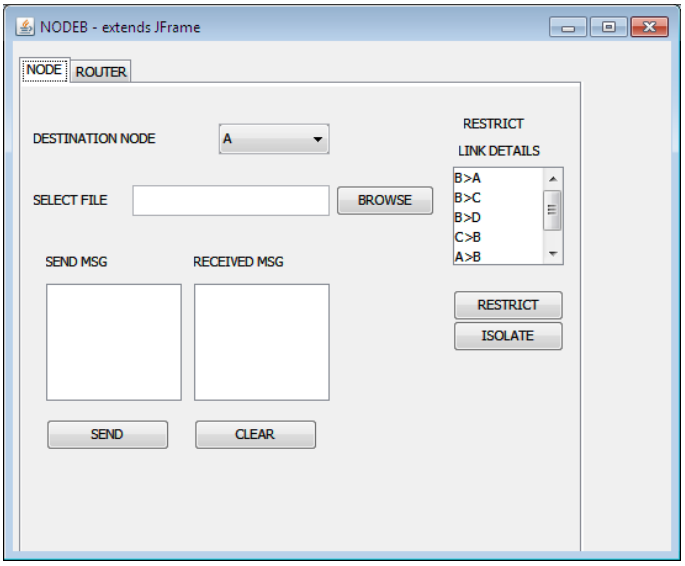
Techniques developed for fast recovery from single-link failures provide more than one forwarding edge to route a packet to a destination. Whenever the default forwarding edge fails or a packet is received from the node attached to the default forwarding edge for the destination, the packets are rerouted on the backup ports. In the authors present a framework for IP fast reroute detailing three candidate solutions for IP fast reroute that have all gained considerable attention. when a forwarding link on a tree fails, the packet may be switched to the other tree.

2.3 Dijkstra's algorithm:

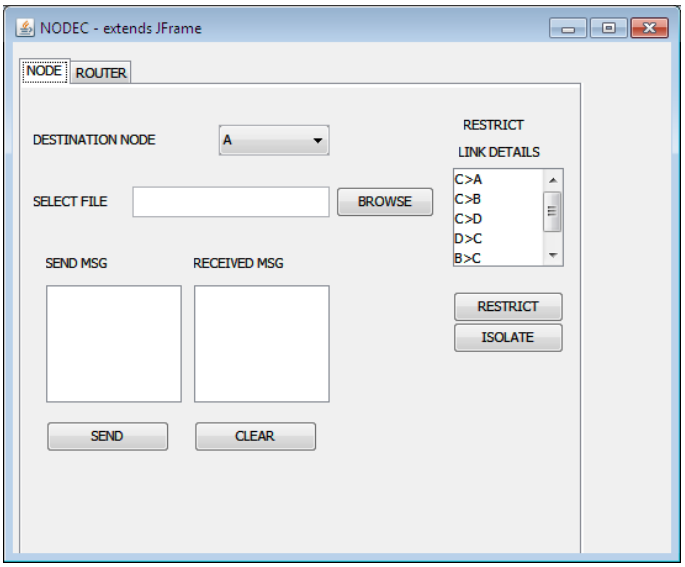
Dijkstra's algorithm is a graph search algorithm that solves the single-source shortest path problem for a graph with non-negative edge path costs, producing a shortest path tree. This algorithm is often used in routing and as a subroutine in other graph algorithms[9].

3.Screen Shots

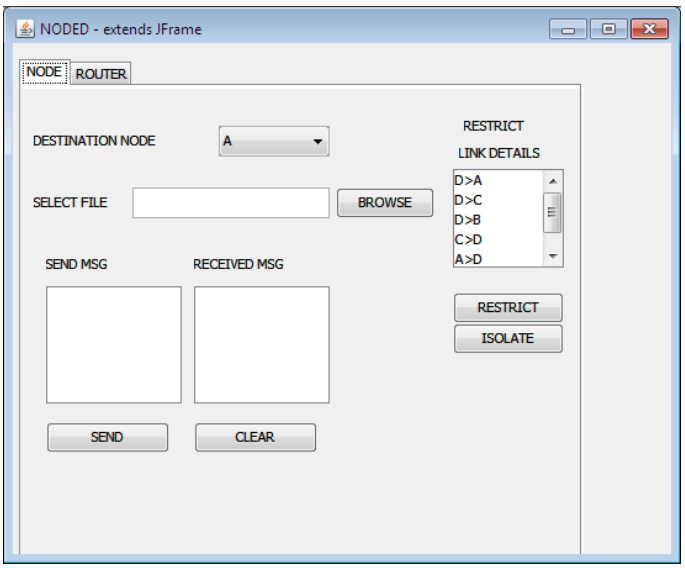
The following are the screen shots that shows the routing process and that avoids the routing part with a link.

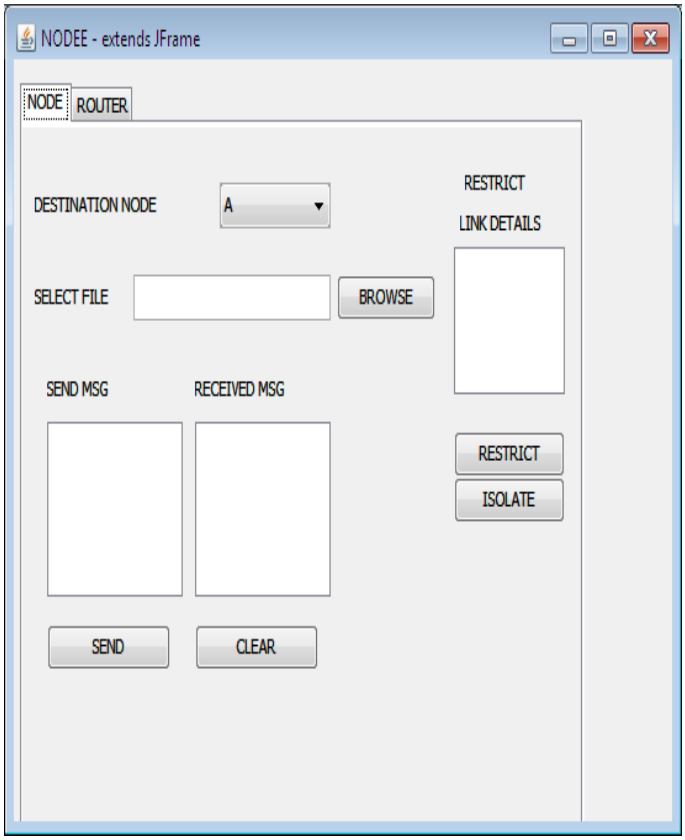


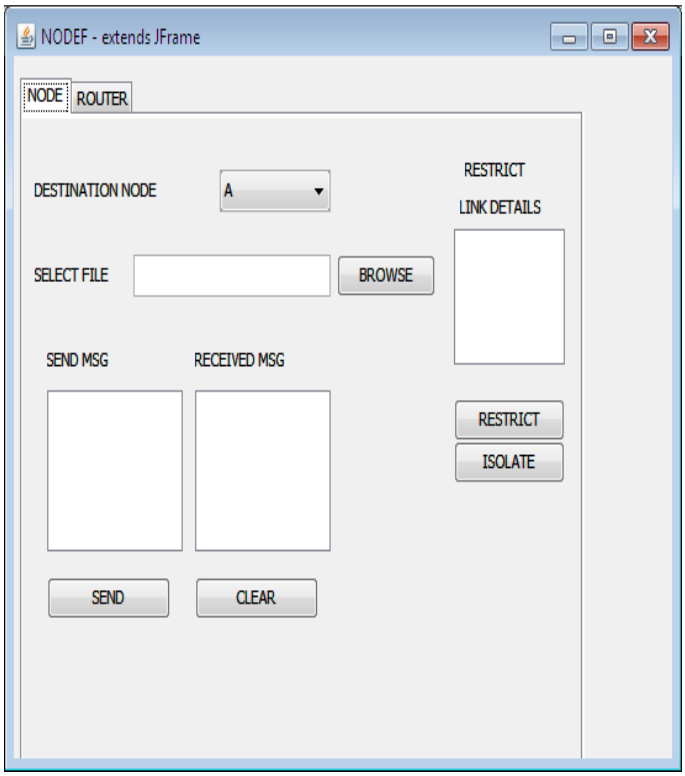
Here we are going to start routing for source A

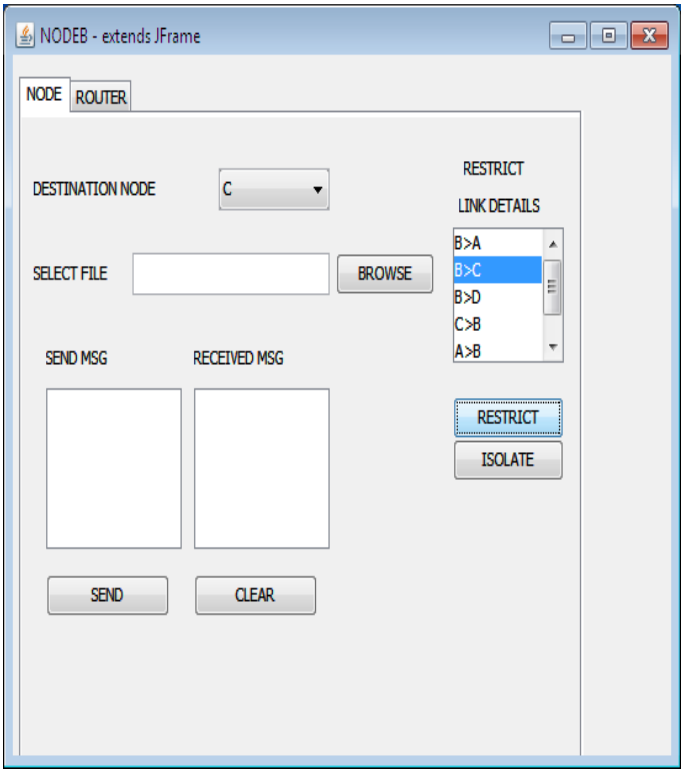


Here the source A travels travels through the paths C ,B and D



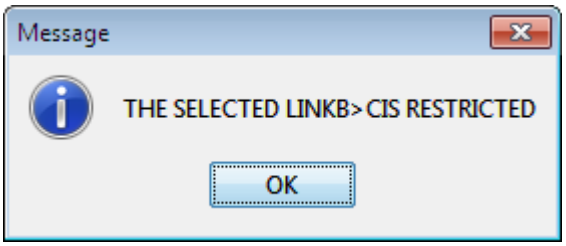






Here the Source A has reached its destination C and thereby when the link fails it displays a message

as follows



4.conclusion

The new algorithm finds an alternative path in traffic network at two stages. At the first stage, the program internally determines the new locations for the designated source and destination pair. At the second stage, the program uses a Dijkstra's Algorithm for searching a path between the new source and destination pair. As a result, the complexity is equal to the Dijkstra's algorithm. In order to get a different path than the minimum-cost one, the program assigns a random factor to the cost of each link as it is scanned and choose a best alternative path for the destination. We show the different multiple paths which is not having the link failure node. The use of randomized link cost is shown to meet both the path diversity and efficiency requirements.

The new algorithm is readily scalable to more sophisticated applications. For example, intersection turning delays can be added to the link cost function to better represent the travel time. As another example, if redundant paths are not allowed, an extra path-checking procedure following the algorithm would well solve the problem. If a certain node or link must be included in any path, a straightforward solution is dividing the target path into two parts at the desired node or link and using the algorithm to find a path for each part before connecting them together. On contrary, if a certain node or link must be excluded from any path, one solution is first using the algorithm to generate a set of initial candidate paths, and then eliminating those which include the node or link.

5.References

[1] A. Atlas, Ed., "U-turn alternates for IP/LDP fast-reroute," IETF draft-atlas-ip-local-protect-uturn-03, Feb. 2006.

[2] A. Atlas and A. Zinin, Eds., "Basic specification for IP fast reroute: loop-free alternative," IETF RFC 5286, Sept. 2008.

[3] S. Bryant, C. Filsfils, and M. Shand, "Remote LFA FRR," IETF Internet Draft draft-shand-remote-lfa-00, Oct. 2011.

[4] C. Filsfils and P. Francois, Eds., "Loop-free alternative (LFA) applicabil-ity in service provider (SP) networks," IETF RFC 6571, June 2012.

[5] E. M. Gafni and D. P. Bertsekas, " Distributed algorithms for generating loop-free routes in networks with frequently changing topology," IEEE Trans. Commun., vol. COM-29, pp. 11–18, 1981.

[6] I. Hussain, Fault-Tolerant IP and MPLS Networks. Cisco Press, 2005.

[7] A. Kvalbein, A. F. Hansen, T.Ciˇci´c, S. Gjessing, and O. Lysne, "Multiple routing configurations for fast IP network recovery," IEEE/ACM Trans. Netw., vol. 17, pp. 473–486, 2009.

[8] K. W. Kwong, L. Gao, R. A. Gu´ erin, and Z. Zhang, "On the feasibility and efficacy of protection routing in IP networks," IEEE/ACM Trans. Netw., vol. 19, pp. 1543–1556, 2011.

[9] S. S. Lor, R. Ali, and M. Rio, "Recursive loop-free alternates for full protection against transient link failures," inProc. 2010 IEEE Symp. on Comput.andCommun., pp. 44–49.

[10] G. R´ etv´ari, J. Tapolcai, G. Enyedi, and A. Cs´asz´ar, "IP Fast ReRoute: loop free alternates revisited," in Proc. 2011 IEEE Int. Conf. on Comput.Commun., pp. 2948–2956.