Optimized Multi Constrained Path Quality of Service Routing Protocol

PRAKASH P.S
Department of Computer Science and Engineering
AnnaUniversity of Technology, Coimbatore
Sri Ramakrishna Engineering College, Coimbatore 641 022
INDIA
prakashpsrajan@rediffmail.com

SELVAN.S
Principal
Anna University, Chennai 600 025
Alpha College of Engineering, Chennai 602 107
INDIA

Abstract: - One of the key issues in providing Quality-of-Service (QoS) guarantees in packet networks is how to determine a feasible path that satisfies a set of additive constraint. Finding a path subject to multiple additive constraints is intractable. In this paper a distributed Optimized Multi Constrained Routing (OMCR) protocol is proposed that computes feasible paths at each node and routes the packets based on their destinations address. OMCR performs connectionless, hop-by-hop Internet Protocol routing and need not remember the global state of the network. It makes routing decision independently and therefore has the ability to respond to network dynamics swiftly. The volume of update messages are reduced significantly by using vector transform method and formation of loops reduced by addressing count-to-infinity issues effectively. OMCR protocol outperforms shortest path routing protocols both link-state and distance vector in terms of convergence time, success ratio and message overhead.

Key-Words: Quality-of-Service, Constrained path, Additive, Routing, Distance vector, Intractable.

1 Introduction

The objective of QoS routing is to find feasible paths from source to destination that satisfy multiple constraints and routing the traffic in such a way that network resources are utilized effectively [30]. Optimized Multi-Constrained Routing (OMCR) is presented to compute feasible paths in packet switched networks. OMCR applies distance vector to construct a shortest path for each destination with reference to a given optimization metric, from which a set of feasible paths are derived at each node. OMCR is able to find feasible paths, optimize the utilization of network resources and operates with the hop-by-hop, connectionless routing protocol. OMCR finds path without loops subject to multiple constraints and nodes need not maintain the global network state. Some tight constraint applications require packets to be delivered to the destination within the stipulated time frame. The problem of finding a path subject to two or more independent additive and/or multiplicative constraints in any possible combination is intractable [13,6,20]. In general, QoS routing focuses on how to find feasible and optimal paths that satisfy QoS requirements of various multimedia applications [11]. Mieghem and Kuipers [16] and Smith Bradley and Garcia-Luna-Aceves [22] have shown that maintaining non-dominated paths for each destination is sufficient to compute constrained feasible paths.

QoS routing approaches have many bottlenecks such as (i) Update of global network state at appropriate time at each node. Some approaches proposed earlier have assumed that the dissemination of routing constraints is known [17], but it is difficult to achieve in practice. This is due to dynamic nature of network and parameters such as residual bandwidth, queue lengths and propagation delay. (ii) Addressing QoS routing subject to only a single constraint such as delay or

bandwidth [29]. (iii) Computing only shortest paths without satisfying the multiple constraints simultaneously [7] and (iv) Considering either multi-constrained path computation or optimization even though they are very much related to each other.All centralized schemes suffer from high convergence time, excessive overhead and slow response to network dynamics [10]. De Neve and Van Mieghem [2] proposed a hop-by-hop destination based QoS routing. In this work, the global network state is required at each node and routing constraints must also be known a priori. Sobrinho adopted an algebraic approach and dealt with path optimization problem in hop-by-hop QoS routing. It is shown that maintaining only non-dominated paths for each destination are sufficient to compute constrained feasible paths [18,22]

The proposed distributed algorithm in [8] suffers from high overhead, duplication of many calculations on each router [8]. Li [14] achieved satisfactory performance only at the cost of high computation complexities; normally it is not acceptable especially network size is growing exponentially. Shortest paths are calculated with reference to an optimization metric and may not satisfy multiple constraints simultaneously [15]. Packets overhead are high due to the nature of vector used [31]. The proposed OMCR is compared with the DMR, MPOR which are given in [15,30] since these algorithms are belong to the same category. The computational complexity is the time it consumes to converge after a single change in the network. The complexity of the proposed scheme is $O(N^i)$ where N^i is the number of neighbours for the node 'i' when a single shortest path is maintained.It is observed that existing distributed MCP algorithms require a global view of the network state at each node and some algorithms assumed that the distribution of routing constraint is known which may not be possible in practice. Hence, it is important that a solution is required for distributed multi constrained routing that does not require global network state to be made available at each and every node.

In this paper, to address the limitations of the above algorithms a distributed Optimized Multi Constrained Routing (OMCR) is proposed which computes feasible paths locally at each node and forward packets on hop-by-hop basis. The proposed algorithm finds multi-constrained paths while optimizing the overall routing performance according to the given optimization metric. The proposed algorithm presented in this paper is an extension of [21]. In this paper a comparison to distributed schemes are added. Performance of packet overhead is analysed for various number of nodes. Convergence time analysis is carried out for different number of hops.

The remaining part of this paper is organised as follows. Section 2 describes OMCR network model. Section 3 explains the description of algorithm. Section 4 deals with flow diagram of the proposed algorithm. In section 5 the pseudo code of proposed algorithm is presented. In section 6 a case study is discussed. Section 7 present simulation results and finally section 8 concludes this paper.

2 OMCR Network Model

A network is modeled by a connected directed graphG= {V, E} , where V is the set of nodes and E is the set of links interconnecting the nodes. Assuming that each link $l_{u,v}$ is associated with a link weight vector W(u,v)={W_1, W_2,…, W_k,} in which W_i is an individual weight component like single routing metric. Accordingly, any path 'p' from a source 's' to a destination 'd' can be assigned a path weight vector $W^p = \{W_1^p, W_2^p, ..., W_k^p\}$, where $W_i^p = \sum_{l_{u,v}} W(u,v)$, if W_i is an additive metric such as delay or $W_i^p = \min(W_i(u,v)), l_{u,v} \in p$, if W_i is a concave metric such as bandwidth.

3 Algorithm Description

Let D_{ij} denote the distance between nodes 'i' and 'j' as known by the node 'i'. D^i_{jk} denotes the shortest distance D_j^k from node 'k', which is a neighbor of node 'i', to destination 'j' as reported to node 'i' by node 'k'. SFD_{ij} denotes shortest feasible distance of node 'i' for destination 'j' which is an estimate of minimum shortest distance maintained for destination 'j' by node 'i'. A node 'i' maintaining a routing entry for each destination 'j' which includes SFD_{ij}, D_{ij} and the successor set chosen for 'j' and denoted by S_{ij}. Node 'i' maintains a neighbor table that records the shortest distance D^i_{jk} reported by each node 'k' in its neighbor set N^i for each destination 'j'; and a link table that reflects the link state w(i,k) for each adjacent link $l_{i,k}$, $k \in N^i$. Each node must run OMCR for each destination and focused on the operation of node i's computation of the set of feasible paths for destination 'j'. Each

node maintains 'x' feasible paths for destination 'j', node 'i' may receive and record 'x' values of D^i_{jk} from each neighbor 'k'; node 'i' also reports to its neighbors the shortest distance of 'x' feasible paths from itself to destination 'j', of which the minimum value is also used as the shortest feasible distance SFD_{ij} of node 'i'. $D_{jj} = 0$, $SFD_{jj} = 0$ and $D^k_{jj} = 0$ for destination 'j'. When a node is activated, SFD is set to infinity which is defined by function of p, and all the other entries are set to empty. When node 'i' receives D_{kj} from neighbor 'k', either updates the estimates D^i_{jk} and without affecting other estimates or node 'i' updates $S_{ij}(d)$ and $SFD_{ij}(d)$ for destination 'j' based on the equation,

$$S_{ij}(t) = \left\{ k \middle| D^i_{jk}(t) \prec SFD_{ij}(t), \quad k \in N^i \right\} \tag{1}$$

and updates its shortest feasible distances as follows:

$$SFD_{ij}(t) = \min\left(D^i_{jk}(t) \middle\| sd(i,k)(t) \right) \tag{2}$$

for all D_{kj} reported by each neighbor 'k' and overall neighbors in N^i or node 'i' remains idle. 'sd' is shortest distance of the adjacent link $l_{i,k}$. The two links are combined by the optimization function and compute the shortest distance of the resulting combined path. Apart from this, node 'i' refreshes the shortest distance of each feasible path maintained for 'j' and sends neighbors updates if any change occurs. Equations (1) and (2) make OMCR a loop-free QoS routing algorithm.

3.1 Optimization

The total number of routing entries for node 'j' maintained at each node forms a directed graph rooted at 'j', which is a sub graph of network G and denoted by SG_j. If routing converges correctly, SG_j is a directed acyclic graph in which each node can have multiple successors for node 'j'. At any point of time, multiple SG_j can exist for the destination 'j', to achieve routing optimization, OMCR constructs SG_j in such a way that path with shortest distance for destination 'j', is always maintained according to Equations (1) and (2). The paths computed between node 'i' and 'j' are called 'feasible shortest path', denoted by FSP_{ij} and at least one of the path has the minimum shortest distance for 'j'. OMCR sends shortest distance only amongst neighboring nodes, like distributed Bellman-Ford (DBF) algorithm, and eliminating expensive routing overhead by distributing link-state information throughout the network [8]. The optimization function used is a combination that considers each link-weight component equally, which is defined by $f(p) = \sum \frac{W_i}{k}$, where 'k' is number of constraints.

Vector transform method transforms the multi dimensional vector to single one by integrating multi-constrained parameter to single one.

$$f_x = \max\left[\frac{w_1 e_x}{c_1}, \frac{w_2 e_x}{c_2} \cdots \frac{w_k e_k}{c_k} \right]$$

$$F_x = f_x \sqrt{k} \tag{3}$$

where $\frac{w_k(e_k)}{c_k}$ is the normalized sub-component weight parameter of multiple vectors and F_x represents the modulus of weight parameter after converting as single vector.

3.2 Network State Information

The underlying network is dynamic and it is essential to update the dynamics of network. It may be the case that node 'i' is unable to find neighbor node 'k' that has reported a shortest distance which is better than the shortest feasible distance maintained at node 'i'. When this happens node 'i' to choose a new set of successors and all nodes whose shortest distance for 'j' involve node 'i' have to incorporate the update in their computation of feasible shortest distance. This can be achieved by coordinating node 'i' with all upstream nodes that use node 'i' in their feasible shortest path calculation for destination 'j'. OMCR does spread-out the calculation to achieve this and two state of operation involved namely ON and OFF state [4]. Nodes in OFF state behave much like a distributed Bellman-Ford algorithm, which means that nodes will simply calculate the shortest distance without coordinating with other nodes.

If none of the nodes resulted in an optimal path for the destination 'j' the node may switch to ON state and it increase the feasible shortest distance so that it can coordinate and synchronize to get a new set of successors. By sending 'query' to neighbors that reports the desired shortest distance for destination 'j'. Node '*i*' returns to OFF state if at least one of the newly elected successor by Equation (1), provides the feasible shortest distance for destination 'j'.

OMCR does not create any closed paths/loops while determining the feasible path. OMCR stays in 'OFF' state as long as the shortest distance to destination remain unchanged or getting reduced. But when the shortest distance increases to a particular destinations, for which nodes send out queries and transit into 'ON' state i.e. a node that runs OMCR synchronizes with upstream nodes and raises its feasible shortest distance up to a sufficient value such that another set of successors can be obtained and return to 'OFF' state. It can be seen that Feasible Shortest Distance (FSD) used by node 'i' to 'j' is lesser than shortest path distance for 'j' that node 'i' reports to all its neighbors $\forall t$. At any point of time, query and reply are not getting overlapped and hence OMCR does not form any loop.

4 FlowDiagram of OMCR

Flow diagram of OMCR algorithm is shown in fig.1. If a node unable to find its neighbour it reports that Shortest Distance (SD) is less than Feasible Shortest Distance (FSD). Node "i" must increase FSD_{ij} to find a new successor set S and node "i" has to synchronize with all upstream nodes. The algorithm operates in two different states namely ON and OFF state. A node is in OFF state if its successor set S_{ij} includes nodes that can provide the optimal path with the shortest distance for "j". Nodes in OFF state may use any conventional algorithm to compute shortest distance for destination "j" and no need to establish any co-ordination with their neighbours.

In the ON state, it requires to find a new set of paths from source node "i" to "j" due to network dynamics i.e. a case where a node unable to find its neighbour reported that shortest distance less than feasible shortest distance. If shortest distance is grater than feasible shortest distance, the node to send query to each neighbour and get reply from all neighbours. If shortest distance with new successor satisfying shortest distance dominates feasible shortest distance then this algorithm returns to OFF state i.e. the new successor set may replace the existing one. Otherwise this operation continues as shown in fig. 1.

In OFF state, in case of a node or link failure the node finds its feasible successor independently and reports it to all neighbour nodes. If the newly derived successor set does not succeed then it may invoke ON state, otherwise the successor set proceeds further. In the derived successor set, minimum shortest distance reported by neighbour nodes and minimum shortest distance of adjacent links are to be linked to find paths with feasible shortest distance. Then these changes need to reflect in routing table of a node. After updating the entries, OFF state finds the paths between any given source and destination node pair.

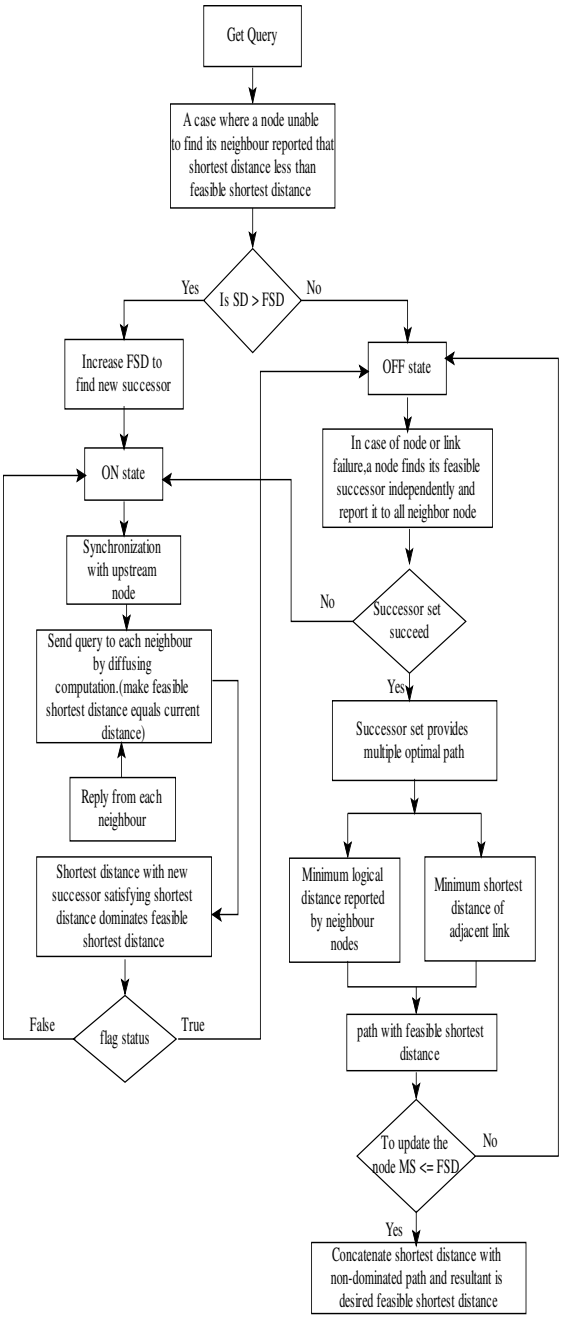


Fig. 1. Flow Diagram

5 Pseudo Code of OMCR

Pseudo code of OMCR algorithm is presented in Alg.1.

integer attributes

SD-Shortest Distance

FSD- Feasible Shortest Distance

x-non-dominated path

k-neighbor, N- set of neighbor

sd-shortest distance

n-node, d-distance, w-weight of link

rsd-reported shortest distance

string attributes

mt-query

p-path

S-successor set, s- single successor

l-link, E-edge

```
1.  void main( ) {
/* initialize Shortest Distance, Feasible Shortest Distance, successor set, each successor neighbor, link cost, for each node, distance, set of neighbor, reported shortest distance, link weight, path, non-dominated path */
2.  int SD[], FSD[], S[], k[], sd[], lc[],n, s, d[], N[], rsd, w[];
3.  string mt, p, x[], l[], E;
4.  SD[i,j]=∞; FSD[i,j]= ∞;
5.  SD[n]=0; FSD[n,j]= 0;
6.  S=NULL; S[i,j]=1;
7.      for(n=1;n≤E;n++){
8.        SD[n,j]= SD[n,j]+w[n,j];
9.        FSD[n,j]= FSD[n,j]+w[n,j];
10.       S[n,j]= S[i,i]+k;}
/* update successor set using shortest distance and feasible shortest distance */
11. update_successor (int j){
12.     if(SD[n,j]< FSD[n,j]){
13.       S[i,j]=n;
14.       for(s=1; s≤ S[n,j]; s++){
15.           for(p=i;p<x[n,j];p++){
/* non-dominated path is calculated from the successor */
16.     p=p+l[i,s];
17.     x[i,j]=x[i,j]+p;
18. printf("Feasible shortest distance is %s",
        x[i,j]);
19.     }}}} //end update_successor()
/* start the computation to stay in ON state */
20. computation(mt, k,d[],rsd[]){
21.     for(k=1;k≤N[i,j];k++){
22.         mt= "query";
23.         k=N[i,j];
24.         d[i,j]=w[i,j];
25.         rsd=SD[i,j]; }
26.         x[k,j]=x[k,j]+p;
27.     SD[k,j]=rsd;}//endcomputation()
/* update the successor using Shortest Distance */
28. update successor ();
29. SD[i,j]=min(SD[n,j]+lc[n]);
30.     if(SD[i,j] < FSD[i,j]){
31.         stateON();}
32.     else{
33.         stateOFF();}
/* calculate new set of successor and update feasible shortest distance */
34. stateON() {
35.     if(mt== "query"){
36.       for(n=1;n≤N[i,j];n++){
37.         mt= "reply";
38.         k=N[i,j];
39.         d[i,j]=FSD[i,j];
40.         rsd=SD[i,j];}
41. computation (mt, k,d[],rsd[]);
42.     else{
43.  if((mt== "reply")||if(SD[i,j] < FSD[i,j])||S[i,j]==NULL){
44.     stateOFF();}}}}//endstateON()
/* perform the operation with successor set */
45. stateOFF() {
46. FSD[i,j]=min(SD[i,j];
47. update successor();} //end stateOFF()
48. } //end main()
```

Alg. 1 OMCR Algorithm

Line 1-6: Various parameters are defined to compute the feasible paths. Feasible shortest distance is defined in array to record the changes in the new feasible shortest distance and recorded to calculate the shortest distance. The successor set is defined to update the new set of successor each time the feasible shortest distance calculated to find the shortest distance. Shortest distance and link cost is defined and used to calculate the feasible shortest distance, successor set and 'n' denotes the number of nodes in the topology. Reported shortest distance (RSD) is the distance reported from the neighbour to the source node. Initially the successor set is empty,

all distance are treated as infinity and the distance to the self node is zero.

Line 7-10: The shortest distance from source 'i' to destination 'j' is calculated along with weight associated with each link. The feasible shortest distance is calculated as that of shortest distance, it is done by calculating each node from source 'i' to destination 'j'.

Line 11-19: This segment shows the process of update successor set to new successor set whenever topology changes. The shortest distance (SD) is computed to see whether or not it dominates the FSD. Once it is satisfied, the feasible distance is found from the available successor set and the desired path is achieved with less convergence time and communication complexity. Node is in OFF state if its successor set S_j^i given in Equations (1) and (2). It includes nodes that can provide the optimal path with the shortest distance for 'j'. Nodes in OFF state behave much like Distributed Bellman–Ford algorithm i.e. they simply use Equation (2) to compute the shortest distance for destination 'j', without having to establish any co-ordination with their neighbour.

Line 20-27: Computation process is shown in this segment. Node 'i' transits into ON state by sending queries to its neighbours and remains in ON state until all neighbours send back reply. A continuous ON state happens only when some of the neighbours do not reply to outstanding query. However, this is not true due to the following reasons. First, a node replies to a query immediately and remains in the OFF state as long as the shortest distance provides by its current successor set. Second, only the upstream nodes of node 'i' in the successor graph SG_{ij} can be affected by the transition of node 'i' i.e. they also transit into ON state if shortest path for 'j' is not found by the local computation.

Line 28-33: From the reported shortest distance from the process of computation, the shortest distance is updated using the minimal of shortest distance and minimal link cost. The updated shortest distance is compared with feasible shortest distance; it is highly necessary because states converge based on the results. From the algorithm it is clear that it has two mutually exclusive states namely ON and OFF. OFF state is one in which there is no hindrance in calculating the shortest distance and providing the feasible solution for the path based on the available successor set. Then it calls the *update successor()* from OFF state, where it compares the returned feasible shortest distance with shortest distance.

Line 34-45: When there is a change in topology such as link failure or reported shortest distance dominates the feasible shortest distance, it starts sending query to each neighbour and receives the reply from all neighbours. The ON state is one in which the reported shortest distance is greater than the feasible shortest distance, it first set a synchronizing stream along the all upcoming nodes and increment the feasible shortest distance to find the new set of successor. The node starts computation i.e. it starts with a single node and return the parameter obtained from the ON state where it compares the path along with all the nodes and finally call the successor set. The new successor set along with feasible shortest distance is updated with non-dominated path and the final result is the desired feasible shortest distance. It is shown that it can have only finite number of ON states. An ON state can be caused by a sequence of network changes or queries sent from neighbours. An infinite number of ON state occurs when a node 'i' has back-to-back ON state without replying pending queries from its neighbour nodes, or experiences infinite number of changes. However, this cannot be true because of the following reasons. First, a query may be blocked by adding its sender 'k' into set of neighbour, and therefore node 'i' may not receive the same query from 'k' endlessly. Second, in practice one can only have finite number of network changes in a sequence, and each node can only have finite number of neighbours from which it can receive queries or updates.

Line 45-47: Once new successor set is found it moves to OFF state from there it updates the feasible shortest distance. In OFF state feasible shortest distance from source to destination could be found by using available successor set. It kept trying to find the successor set till succeeds. If no successor set is found then the algorithm may stop the process of finding the feasible path for the given source-destination pair.

6 Case Study

An example topology is shown in fig.2. Feasible path *'p'* computed for destination *'k'* and weight W_p should also be maintained, because we need to verify W_p whether feasible path can be obtained within the delay bounds. Nodes are labeled with feasible path p(x,y) for the destination *'y'* and edges

are labelled with their link weights that consist of cost and delay.

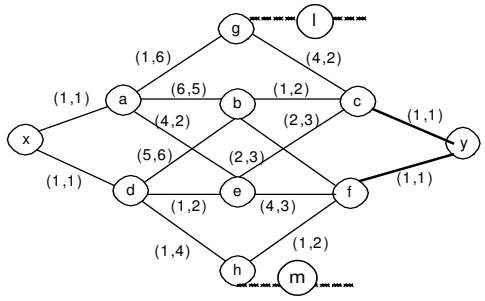
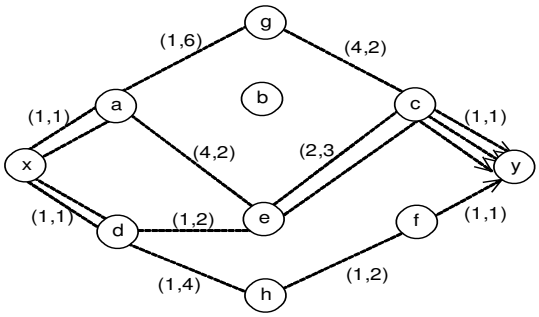


Fig 2. Example Topology

Distance vectors are propagated from destination and propagate to source through upstream nodes. At node *'e'*, two paths (c,y) and (f,y) are propagated from neighbor *'c'* and *'f'* respectively and both have path weight (1,1). $pw_{ij}(m) = pw_{kj}(m) + w_m$ for additive metric where m=1,2,…, number of constraints. Fig. 3 shows the feasible paths between node 'x' and 'y'.



x →p(7,10)	a→p(6,9)	g→p(5,3)	c→p(1,1)
p(8,7)	p(8,8)		
p(4,8)	p(7,6)	b→p(2,3)	f→p(1,1)
p(5,7)	p(9,6)	p(3,6)	
d→p(3,7)	e→p(3,4)		
p(6,6)	p(5,4)		
p(4,6)	h→p(2,3)		
p(7,9)			

Fig.3. Feasible paths between nodes 'x' and 'y'

7 Simulation Results

In this section, experimental results, which are carried out in a network simulator are presented [9, 19, 32]. A random graph used by many researches [23, 25] is used. Each link weight component is uniformly distributed between 1 and 10 [28]. Each source-destination pair is randomly chosen from the networks. Additive routing metrics are only considered so that it can be compared the performance with other multi-constrained path algorithms. The source and destination pairs are randomly generated such a way that minimum hop-count between them is at least two. All the simulations were carried out in Core2Duo, 2.4 GHz, 1GB memory Linux OS Computer.

Optimized Multi Constrained Routing (OMCR) is compared with Distributed Multi Constrained Routing protocol (DMR) [31], Multi Constrained Path Optimization Routing Protocol (MPOR) [16], Distributed Bellman-Ford (DBF) [8] algorithm which is based on Distance Vector (DV) routing and also with Link State (LS) routing [1]. The process of establishing routes for all destinations is determined at each node for the first time since nodes are *'up'*. Single link failure, node failure and finally a sequence of link failures are analyzed. Link or node failures are simulated since these are worst case scenario that may cause either network partition or shortest distance to increase.

'Number of operation' (NM_OP) which denotes iterations of the main loop in its implementation i.e. sum over all nodes in the network are analyzed. Then the 'number of messages' (NM_MSG), which is the total number of messages sent over all links, in case of link or/and node failure. Convergence time is comparatively less in OMCR. This is due to less messages are being exchanged while establishing the routes. In MPOR and DBF the number of messages exchanged is relatively high because each distance vector carries only one message. However in OMCR the multiple vectors reduce overhead by using vector transform that in turn reduce the convergence time. Table 1 represents the node configuration considered for the simulations.

Table 1 Node Configuration

S.No	Number of Nodes	Area in Sq.m
1	20-60	500×500
2	80-120	1000×1000
3	140-160	1500×1500

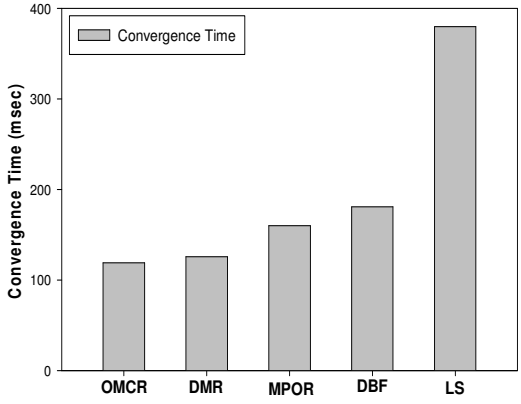
Table 2 describes the various simulation parameters

Table 2 Simulation Parameters Details

S.No	Simulation Parameter Details	Area in Sq.m
1	Type of Topology	Arbitrary
2	Placement of Node	Random
3	Distance between the Nodes	Euclidean
4	Value of Link Weight Component	Between 1 and 10
5	Nature of Constraints	Additive
6	Packet Size	512 bytes

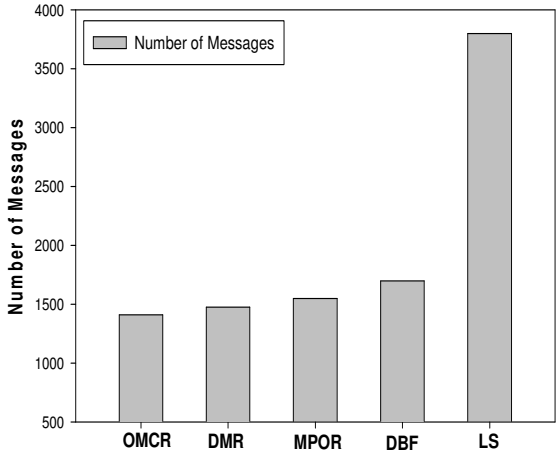
7.1 Message Overhead in Establishment of Routes

Computing the multi-constrained QoS routing by multi dimensional vector will cause a lot of overhead and inconsistent with optimal theoretical route and QoS requirements could not be satisfied. But, by vector transform method i.e. the multi dimensional vector space is first converted into single dimension vector space and then all routes from source to destination, which meet QoS requirements that can be searched by OMCR. By using this, non-dominated path problem is solved.



(a) Convergence Time in Establishment of Routes

Fig.4 (a) and (b) represents effect of convergence time and number of messages exchanged in establishing the routes respectively. In establishment of routes, OMCR slightly outperform DMR (Distributed Multi Constrained Routing). Even though both algorithms adopt distance vector concepts, feasible paths are found efficiently in OMCR by using vector transformation concept.



(b) Number of Messages Exchanged in Establishment of Routes

Fig. 4 Number of Messages Exchanged and Convergence Time in Establishment of Routes

Non-dominated routes are determined with multiple constraints by vector transformation i.e. transformation of multi dimension to single one. Moreover, OMCR effectively prunes the links that are not satisfying the QoS constraints according to concave parameter.

It is observed that the 'Number of Messages' in OMCR is comparatively less. This is due to information exchanged in fewer amounts of packets. But in LS protocol (Link State) the entire network has to be updated and hence both 'Number of Messages' exchanged and 'Convergence Time' is significantly high. Even though DBF and MPOR do not reflect the entire state of the network, Number of Messages and Convergence time is higher when comparing to OMCR. This is due to more number of vectors exchanged i.e. only single vectors are implemented to exchange data, whereas in OMCR vectors transform method is used so that overhead is reduced significantly.

'Number of operations' (NM_OP) is the measure of number of times the main loop of the protocol is executed. In other words, it is the summation of iteration of loop over all the vertices in the network. OMCR maintains 'sub optimal property' which means that the shortest path of any two sub nodes is also the shortest path of the two nodes taken into consideration at any point of time. In other words, OMCR does not form any loops while computing the feasible paths. 'Number of Operations' is marginally higher in DMR. But OMCR converges quickly comparing to other algorithms; this is due to

the messages exchanged efficiently by a single transformed vector. Hence, OMCR outperforms other protocols. Every node is running OMCR to select its successor in the process of finding the feasible paths; the number of operations required is almost the same as that of MPOR and DBF. In case of LS the number of operations is less since it maintains the overall state of the network as shown in Fig. 5.

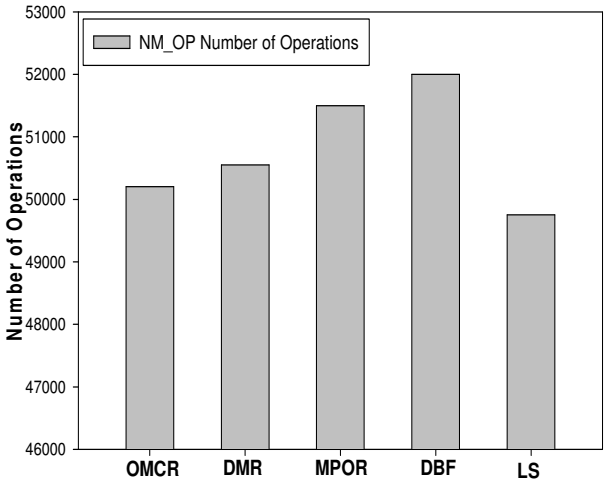
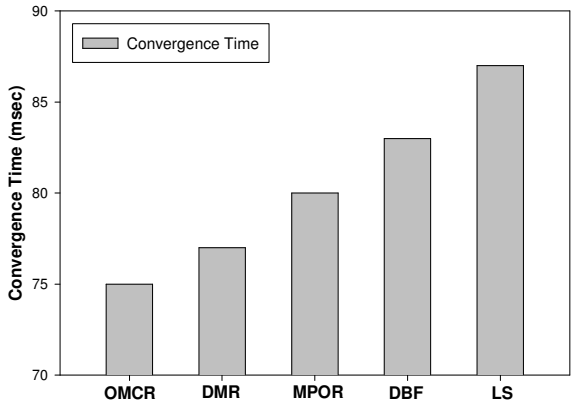


Fig.5. Number of Operations to Establish Routes

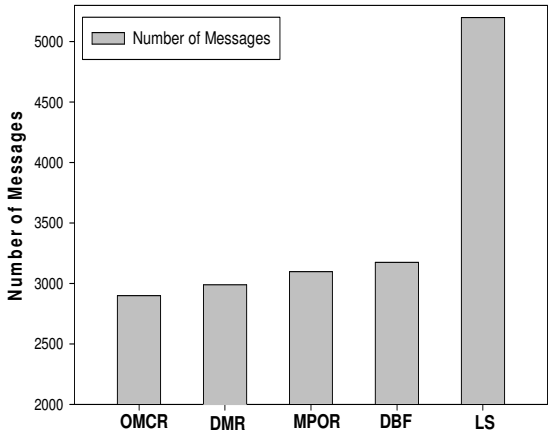
Generally, in distance vector routing each node talks only to its directly connected neighbours, but it tells them everything it has learned. In link state routing each node talks to all other nodes but it tells them only what it knows i.e. only the state of its directly connected links[12].

7.2 Issues of Single Link Failure

Fig.6 (a) and (b) depicts the effect of convergence time and number of messages in single link failure respectively. MPOR and DBF function in the same way in case of single link failure. But LS needs to perform the algorithm on the whole topology for every newly received link-state, since the topology of the network is kept changing. In case of OMCR and DMR they behave almost in similar way because there is no network partition if a single link fails.



(a) Convergence Time in Single Link Failure



(b) Number of Messages in Single Link Failure

Fig.6.Convergence Time and Number of Messages in Single Link Failure

OMCR performs well almost in all network load conditions i.e. both in light and heavy load conditions. This is due to effective update information exchanges in case of a topology change. Neighbour-alive messages are being sent whenever an event occurs such as link or node failure. Due to this, the convergence time is less whereas in other protocols information is exchanged relatively larger packets that increase the overhead.

Owing to these reasons, OMCR converges quickly. So, the convergence time and 'Number of Messages' are almost similar or even little less in OMCR whereas due to high overhead and longer update process it is on upper side in a centralized protocol such as Link State. Similarly in case of single link failure, Link State takes longer time than any other protocol because of update overhead. NM_OP and NM_MSG metrics are measured since

the start of the simulation until all nodes stop updating their tables.

OMCR has less number of operations required to update a node in case of a single link failure comparing with DMR, MPOR and DBF. Since OMCR keep updating the network whenever a event occurs or at regular time interval, the convergence time required is always less than its counterparts. Moreover, these update messages are small in size and hence update process of the network is also relatively simpler comparing to DMR and MPOR. Finally the time to converge the network is analyzed. Convergence time is the time difference between the start of an event and completion of update process of all routing tables of nodes. The results are depicted in Fig.7.

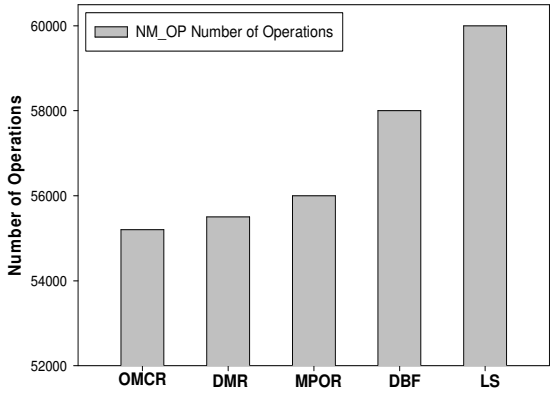


Fig. 7. Single Link Failure Vs Number of Operations

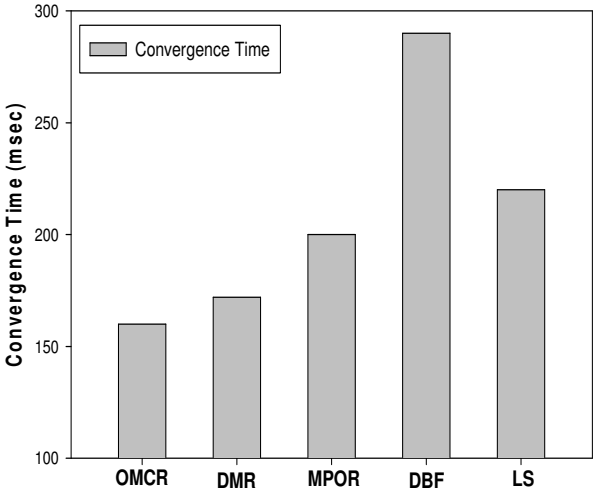
7.3 Count-to-Infinity Problem

OMCR can avoid count-to-infinity (CTI) problem and reduces the update time by sending neighbour-alive-messages. In distance vector routing the count-to-infinity problem reflects a routing loop. Distance vector protocols may get into an unstable state as soon as a count-to-infinity situation occurs [3]. OMCR protocol avoids CTI routing loops. Because in OMCR algorithm update message traffic is reduced by sending small neighbour-alive messages instead of periodic routing updates. An incoming neighbour-alive message confirms all routes received from this neighbour. The neighbour-alive message has an invariably small size in contrast to the ordinary periodic routing update whose size depends on the amount of subnets in the network. Moreover these smaller messages checks either the entire routing update traffic can be reduced or the sending interval and thus convergence time of the network can be reduced.

The impact of CTI situation becomes apparent by examining their duration. As long as CTI situation lasts, all packets for the corrupted network is cycling in the loop. This is a burden to all routers and subnets located in the loop. The average CTI duration increases with increasing hop-count. OMCR converges faster than the normal DBF from the time where the topology change happens. It is possible to avoid CTI situation and to converge faster than distance vector algorithms in case of any topology change.

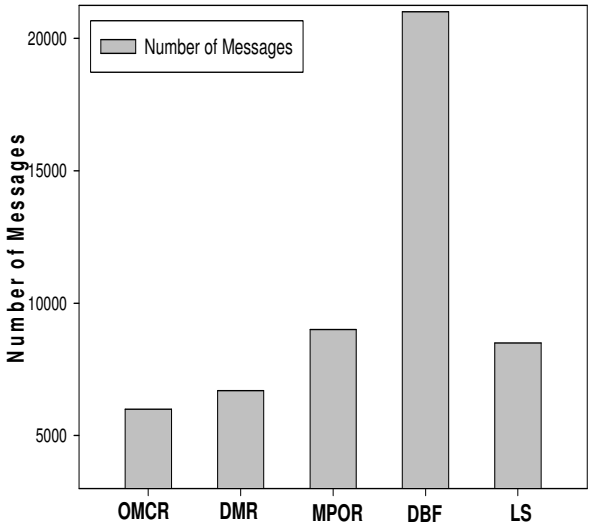
When a node failure occurs, DBF and MPOR take longer time to converge, since 'count-to-infinity' problem is not addressed in MPOR. Because of this, the message exchanged is enormously high. But in OMCR this problem is addressed efficiently i.e. the maximum number of hop-count is limited. So when the update message exceeds this value for a particular node, OMCR understands that the particular node is unreachable and relinquishes the update process. This is the reason why the convergence time of OMCR is less. In case of LS, it finds alternative path since it knows the state of entire network, but it is very expensive, especially if the network size is large. Fig. 8 (a) and (b) represents convergence time and 'number of messages' in single node failure respectively.

Similarly the number of messages to bring back the network to normal, in case of single node failure is very high in DBF i.e. many order of magnitude higher than OMCR. But in the case of OMCR, it has been brought to significantly lower by efficiently addressing 'count-to-infinity' problem as shown in the Fig. 8.



(a) Convergence Time in Single Node Failure

OMCR behaves much like DMR while a link fails and no network disastrous events are occurred. Since OMCR protocol maintains the degree of the node more than two, the possibilities of network isolation is drastically reduced. Fig. 10 (a) and (b) represents effect of convergence time and 'number of messages' in multiple link failure respectively. Due to 'count-to-infinity' phenomenon, the number of messages exchanged is



(b) Number of Messages in Single Node Failure

Fig. 8 Convergence Time and Number of Messages in Single Node Failure

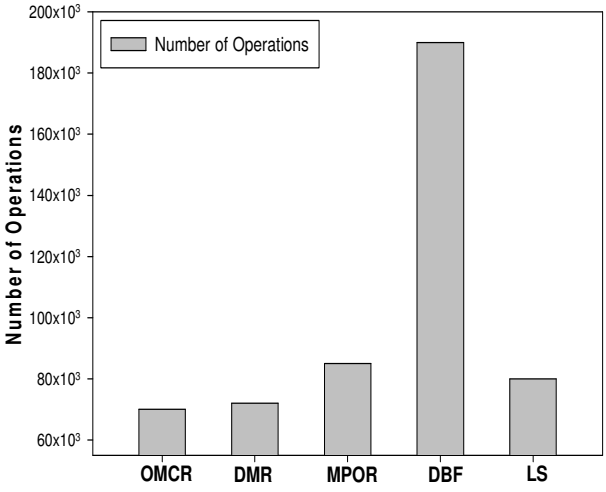
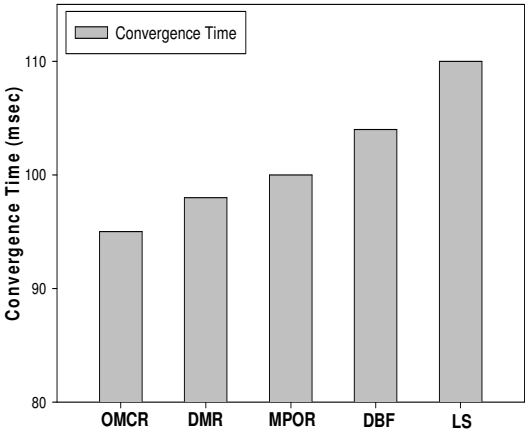


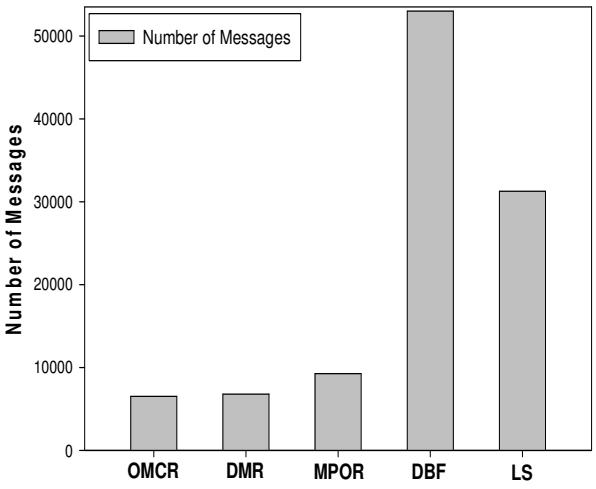
Fig.9. Single Node Failure Vs Number of Operations

quite high in case of a node failure and multiple link failure in MPOR compared to OMCR or link state routing. Fig. 9 shows the effect of Number of Operation in the event of single node failure.

If all adjacent links are failed to a particular node in a given point of time, again MPOR behaves poorly. Also it is observed that NM_OP, NM_MSG and convergence time of Link State routing are much higher than OMCR. This is due to more link updates need to be sent out to keep nodes updated about the network state. Routing tables must be refreshed to track the latest changes made in the network. If more link failures happen it further degrades the performance of Link State routing protocol as shown in Fig. 9. and Fig.10.



(a) Convergence Time in Multiple Link Failure



(b) Number of Messages in Multiple Link Failure

Fig. 10. Multiple Link failure Vs Convergence Time and Number of Messages

'Number of Operation' including the number of times the loop should be iterated is comparatively less as shown in Fig. 11. Unlike DMR and MPOR, the proposed algorithm converges quickly since it efficiently handles count-to-infinity problem and update network nodes. Multiple links results in CTI situation and ignoring this, the network may not converge. OMCR effectively treating count-to-infinity situation and hence the number of operations required to converge the network is relatively small. In DMR and MPOR this issue is not dealt with effectively. CTI situation is not taken into consideration and hence time to converge the network in terms of 'Number of Operations' is comparatively high.

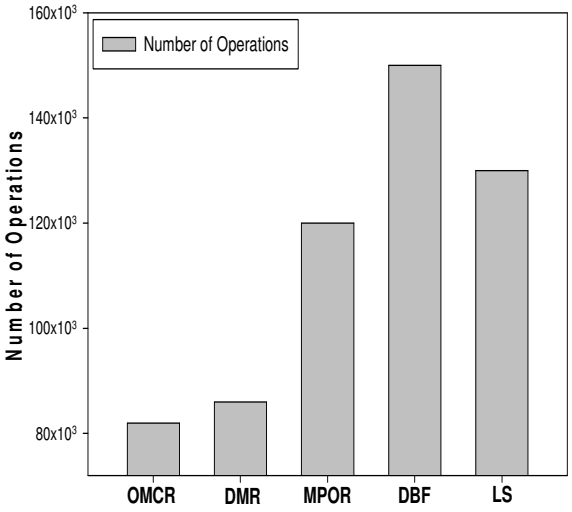


Fig. 11 Multiple Link Failure Vs Number of Operations

7.4 Analysis of Convergence Time

Fig.12 shows the impact of convergence time when number of hops is increasing. When the number of hops increase, the update process of the network should be done in tandem; otherwise it may lead to a situation called 'inaccurate state information' which is a predominant limitation in Link State protocols. OMCR outperforms protocols such as MPOR and DBF in terms of convergence time. This is due to update information of OMCR reaches quickly to neighbouring links in smaller packets whereas MPOR and DBF are following the update at periodic time interval. DMR closely follows OMCR in terms of convergence time.

OMCR outperforms other protocols in converging the network when the 'number of hops' is increasing i.e. when the size of the network grows, link state protocols perform poorly because it has to update the entire network. The time taken to update the network is kept increasing when the size of the network is high and it may also lead to inaccurate state information. It reveals that having global network state at each node is not always a better approach in practice. It is further observed that when the state of network changes rapidly it is very hard to update the information at router.

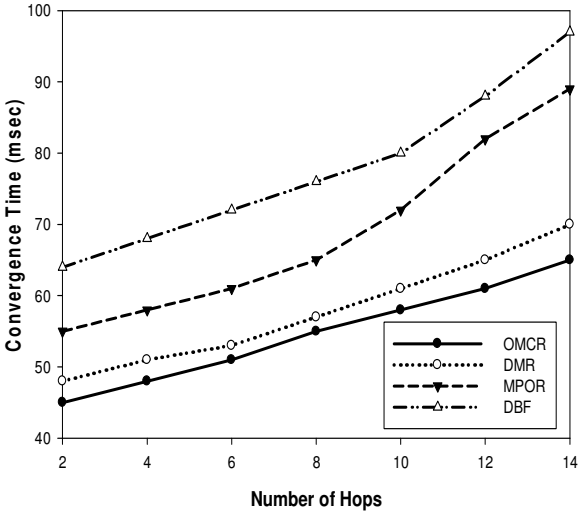


Fig.12. Convergence Time Vs Number of Hops

Fig.13. shows the effect of a packet overhead when the number of nodes is increasing. Networks are generated using Waxman graph algorithm [25]. Here 'n' nodes are randomly distributed and each node is placed at a location with integer coordinates. The Euclidean metric is then used to determine the distance between each pair of nodes. On the other hand edges are introduced between pairs of nodes u,v with a probability that depends on the distance between them.

The edge probability is given by $p(u,v) = \beta\exp(-d(u,v)/\alpha L)$ where $d(u,v)$ is the distance from node 'u' to 'v', L is the maximum distance between two nodes and α, β are parameters in the range between 0 and 1. The value of α is selected as 0.15 and β as 0.2.

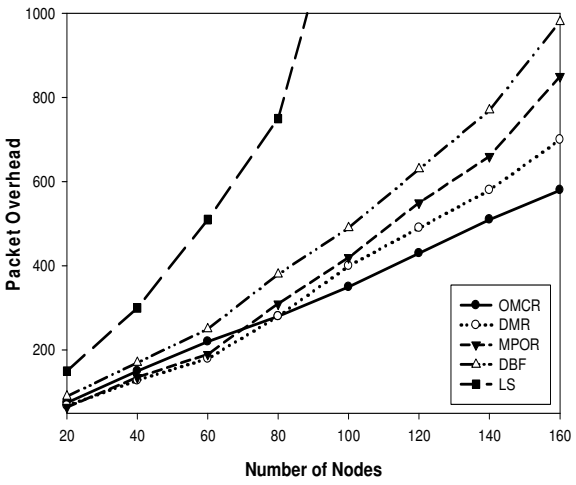


Fig. 13. Packet Overhead Vs Number of Nodes

Here, the number of packets required to update the network are considered. OMCR scores better in amount of packet sent to update the network. Moreover, OMCR occupies less bandwidth comparing to other protocols due to its smaller update packet size. So, even when the size of the network is large, OMCR maintains the nodes intact and finds a feasible non-dominated path subject to many additive constraints i.e. OMCR handles scalability issues effectively. Size and volume of overhead packets is larger in DMR and DBF, since both of these protocols adopt conventional techniques in updating the information. DMR closely follows OMCR in terms of packet overhead.

7.5 Success Ratio Analysis

Even though DMR deals with CTI situation, the volume of the packet is slightly higher. This may be due to many vectors sent in updating the network state information. But in OMCR, this is reduced by vector transformation and hence it is performing better than DMR and other algorithms. In case of Link State, the number of overhead packets is high and also it may consume longer time to converge the network. Moreover, there is a possibility that the network state may further change before refreshing an earlier update and this situation leads to inaccurate state information.

Success Ratio (SR) of OMCR is compared with other centralized schemes in Table 3. Centralized algorithms require the complete state information. Feasible Path Algorithm with Two Constraints (FPATC) [10] and variation of Jaffe algorithm (JA)[5], where a shortest path algorithm with reference to the aggregated link-cost function are compared with OMCR. In this calculation, Success Ratio is defined as the ratio of routing requests routed to total routing requests received at any given point of time [24, 26, 27]. It is found that performance of Jaffe's algorithm performs poorly as compared with other algorithms in all circumstances. It derives single aggregated metric that may not be the better approach for constrained path computation. The performance of FPATC is good in terms of success rate, but handles only two additive constraints and running time cannot be ascertained if the number of constraints is more.

Table 3 Comparison of Percentage Success Ratio of OMCR with Centralized Schemes

Total Routing Request	OMCR		FPTAC		JA	
	Routing Request Routed	% SR	Routing Request Routed	% SR	Routing Request Routed	% SR
100	95	95.00	91	91.00	81	81.00
200	194	97.00	186	93.00	170	85.00
300	292	97.33	281	93.66	261	87.00
400	391	97.75	376	94.00	358	89.50
500	488	97.60	468	93.60	449	89.80
600	584	97.33	565	94.16	546	91.00
700	683	97.57	665	95.00	641	91.57
800	775	96.87	758	94.75	736	92.00
900	872	96.88	851	94.55	828	92.00
1000	969	96.90	949	94.90	918	91.80

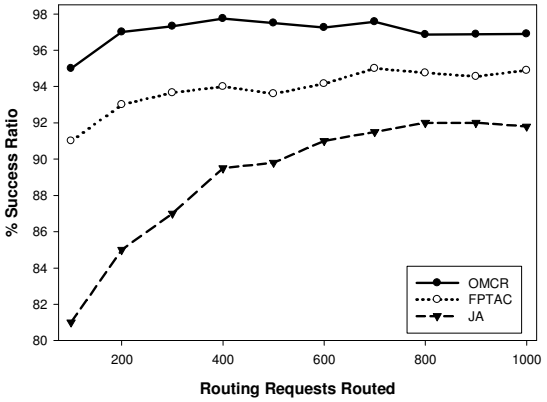


Fig.14 Percentage of Success Ratio Vs Routing Requests Routed in Centralized Schemes

OMCR solves general K additive constrained MCP problems. Comparison of Success Ratio of OMCR with other centralized schemes is depicted in Fig.14.

In Table 4 success ratio of OMCR is compared with distributed schemes. OMCR is compared with DMR, MPOR and DBF in terms of success ratio as shown in Fig.13. OMCR is marginally outperforms DMR and it delivers significantly better results compared to MPOR and DBF. OMCR deals with vector transform method to reduce the overhead messages and non-dominated feasible paths were determined.

In DMR no such scheme is implemented and hence its performance is marginally poor. The algorithms such as MPOR and DBF are using only conventional schemes in sending update message and due to this they under perform when compared to OMCR. Moreover, the size of the update message is significantly smaller in OMCR. OMCR success rate is in the range between 95% and 98% by maintaining 4 to 5 non-dominated paths. DMR performs closely to OMCR in terms of success ratio whereas the other two algorithms are showing marginally less percentage of success ratios. OMCR performs better by solving non-dominated path problem of multi constrained routing and by effectively handling CTI situation as depicted in Fig.15.

Table 4 Comparison of Percentage of Success Ratio of OMCR with Distributed Schemes

Total Routing Request	**OMCR**		**DMR**		**MPOR**		**DBF**	
	Routing Request Routed	**% SR**	**Routing Request Routed**	**% SR**	**Routing Request Routed**	**% SR**	**Routing Request Routed**	**% SR**
100	95	95.00	95	95.00	94	94.00	92	92.00
200	194	97.00	194	97.00	192	96.00	190	95.00
300	292	97.33	291	97.00	289	96.33	285	94.90
400	391	97.75	389	97.25	387	96.75	382	95.50
500	488	97.60	485	97.00	482	96.4	476	95.40
600	584	97.33	581	96.8	578	96.33	569	94.83
700	683	97.57	677	96.75	672	96.00	663	94.70
800	775	96.87	773	96.65	671	95.86	756	94.50
900	872	96.88	870	96.70	862	95.77	851	94.55
1000	969	96.90	966	96.60	958	95.80	943	94.30

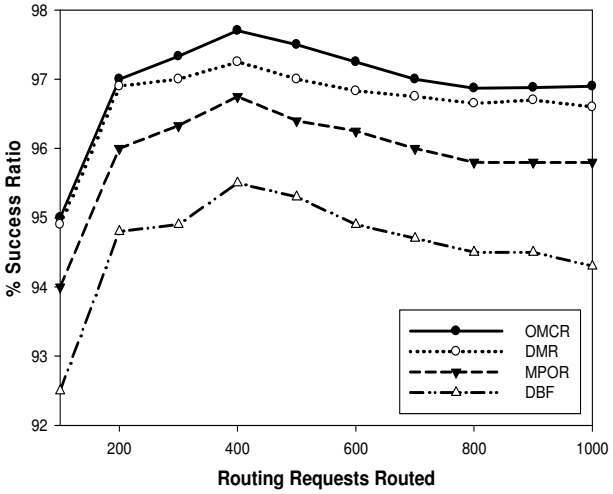


Fig.15 Percentage of Success Ratio Vs Routing Requests Routed in Distributed Schemes

8 Conclusion

Optimal path selection subject to multiple constraints is an intractable problem. Optimized Multi Constrained Routing is a QoS routing protocol that uses distance vectors to solve multi constrained path problem with loop-free paths. OMCR does not require storing the global network state and also addresses optimization issues in routing. Experimental results show that OMCR outperforms the shortest path routing algorithm that are being used in current Internet environment with regard to network overhead and routing complexity. Further, it is revealed that having global network state at each node is not always a good approach in practice. Convergence time of the proposed algorithm is comparatively less and this algorithm achieves a better routing success ratio while comparing with many other routing schemes both in distance vector and link state segments. In the future direction of research OMCR may be extended to a situation where packets for the same destination are distributed through many paths simultaneously. The performance of OMCR could be experimented in wireless and adhoc network environment after having incorporated the necessary changes to cope up with mobility issues.

References:

[1] Cormen, T.H., Leiserson, C.E., Rivest, R.L. and Stein, C., "Introduction to Algorithms", 2nd Edition, New York: McGraw Hill, 2001

[2] De Neve, H. and Van Mieghem, P., "TAMCRA: A Tunable Accuracy Multiple Constraints Routing Algorithm", Journal on Computer Communications, Vol. 23, No. 7, 2000, pp. 667-679.

[3] Frank Bohdanowicz, Marcel Jakobs and Christoph Steigner, "Statistical Convergence Analysis of Routing Algorithms", In Proc. of Ninth International Conference on Networks (ICN), 2010, pp. 365-370.

[4] Garcia-Lunes-Aceves, J.J., "Loop-free routing using diffusing computations", IEEE/ACM Transaction on Computer Networking, Vol. 1, No. 1, pp. 130-141, 1993.

[5] Jaffe, J.M., "Algorithms for finding paths with multiple constraints", Networks, Vol. 14, 1996, pp. 1228-1234.

[6] Jalel Ben-Othmanand Bashir Yahya, "Energy efficient and QoS based routing protocol for wireless sensor networks", Journal of Parallel and Distributed Computing, Elsevier Publications, Vol. 70, No. 8, 2010, pp. 849-857.

[7] Jun Wang and Klara Nahrstedt, "Hop-by-hop routing algorithms for premium-class traffic in DiffServ networks", In Proc. of 21st Annual Joint Conference of IEEE Computer and Communications Societies, (IEEE INFOCOM "02), Hilton, New York, Vol. 2, No. 1, 2002, pp. 705-714.

[8] Kevin, R.H., Terri, L.S. and Douglas, R.S., "On the distributed Bellman-ford Algorithm and the Looping problem", Journal of Computing, Vol. 19, No. 4, 2007, pp. 542-551.

[9] Kevin Fall and Kannan Varathan, "The ns Manuals, The Vint Project", University of California, Berkeley, USA, 2007, pp. 28-130.

[10] Korkmaz, T., Krunz, M. and Tragoudas, S., "An efficient algorithm for finding a path subject to two additive constraints", In Proc. of International Conference on Measurements and Modeling of Computer Systems, ACM SIGMETRICS 2000, 2000, pp. 318-327.

[11] Kuipers, F.A. and Mieghem, P.V., "Conditions That Impact the Complexity of QoS Routing", IEEE/ACM Transactions on Networking, Vol. 13, No. 4, 2005, pp. 717-730.

[12] Larry, L.P. and Bruce, S.D., "Computer Networks: A Systems Approach", Morgan Kaufman Publishers, Fourth Edition, p. 283, 2007.

[13] Line Blander Reinhardt and David Pisinge, "Multi-objective and multi-constrained non-additive shortest path problems", Computers and Operations Research, Elsevier

Publications, Vol. 38, No. 3, 2011, pp. 605-616.

[14] Li, Z., "Finding multi-constrained feasible paths by using depth-first search", Journal of Wireless Networks, Springer Science Publications, Vol. 13, 2007, pp. 323-334.

[15] Li, Z., "A distributed approach for multi-constrained path selection and routing optimization", In Proc. of ACM International Conference Proceeding Series, 3rd International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks2006(QSHINE"06), Vol. 191, 2006, Article No. 36.

[16] Mieghem, P.V. and Kuipers, F.A., "Concepts of exact QoS routing algorithms", IEEE/ACM Transaction on Networking (TON), Vol. 12, No. 5, 2004, pp. 851-864.

[17] Mieghem, P.V., Neve, H.D. and Kuipers, F. "Hop-by-hop quality of service routing", Computer Networks: The International Journal of Computer and Telecommunications Networking, Vol. 37, Nos. 3-4, 2001, pp. 407-423.

[18] Mieghem, P.V. and Kuipers, F.A., "Concepts of exact QoS routing algorithms", IEEE/ACM Transaction on Networking (TON), Vol. 12, No. 5, 2004, pp. 851-864.

[19] Mie Mie Thaw, "Fuzzy-based multi-constrained quality of service distance vector routing protocol in mobile ad-hoc networks", In Proc. Second International Conference on Computer and Automation Engineering (ICCAE), Vol. 3, , 2010, pp. 429-433.

[20] Navrati Saxena, Abhishek Roy and Jitae Shin, "QuESt: a QoS-based energy efficient sensor routing protocol", Wireless Communications and Mobile Computing, Vol. 9, No. 3, 2009, pp. 417-426.

[21] P.S.Prakash, S.Selvan, "An efficient and Optimized Multi Constrained Path Computation for Real Time Interactive Applications in Packet Switched Networks" International Journal of Computer Science Vol.3, No.4, 2008, pp.293-300.

[22] Smith Bradley, R. and Garcia-Luna-Aceves, J.J., "Efficient policy-based routing without virtual circuits", In Proc. of First International Conference on Quality of Service in Heterogeneous wired/wireless Networks QSHINE"04, Dallas, Texas, Vol. 1, 2004, pp. 242-251.

[23] Shigang Chen and Nahrstedt, K., "Distributed QoS routing with imprecise state information", In Proc. of 7th International Conference on Computer Communications and Networks 1998 (ICCCN 1998), Lafayette, LA, USA, Vol. 1, 1998, pp. 614-623.

[24] Tam, W.Y., Lui, K.S., Uludag, S. and Nahrstedt, K., "Quality-of-Service routing with path information aggregation", Computer Networks, Elsevier Publications, Vol. 51, 2007, pp. 3574-3594.

[25] Waxman, B.M., "Routing of multipoint connection", IEEE Journal on Selected Areas in Communications, Vol. 6, No. 9, 1988, pp. 1617-1622.

[26] Wang Xueshun, Yu Shao-hua, Dai Jinyou and Luo Ting, "A Multiple Constraint quality of service routing algorithm base on Dominating Tree", In Proc. International Conference on Computational Intelligence and Software Engineering (CiSe 2009), 2009, pp. 1-4.

[27] Wen-Lin Yang, "Optimal and heuristic algorithms for quality-of-service routing with multiple constraints", Journal on Performance Evaluation, Elsevier Publications, Vol. 57, 2004, pp. 261-278.

[28] Yuan Xin, "Heuristic algorithm for multi-constrained quality-of-service routing", IEEE/ACM Transaction on Networking (TON), Vol. 10, No. 2, 2002, pp. 244-256

[29] Zhang Zhi-Li, Nelakuditi, S., Tsang, R.P. and Du, D.H.C., "Adaptive proportional routing: A Localized QoS Routing Approach", IEEE/ACM Transactions on Networking (TON), Vol. 10, No. 6, 2002, pp. 790-804.

[30] Zheng Wang and Crowcroft, J., "Quality-of-service routing for supporting multimedia applications", IEEE Journal on Selected Areas in Communication", Vol. 14, No. 7, 1996, pp. 1228-1234.

[31] Zhenjiang Li and Garcia-Luna-Aceves, J.J. "Loop-free constrained path Computation for hop-by-hop QoS Routing", Computer Networks: The International Journal of Computer and Telecommunications Networking, Elsevier Publications, Vol. 51, No. 11, 2007. pp. 3278-3293.

[32] Zhu Daxin and Cai Danlin, "Research and simulation of distributed QoS routing algorithm", In Proc. Second IEEE International Conference on Broad Band Network and Multimedia Technology, 2009, pp. 24-27.