International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:3, 2008

A Characterized and Optimized Approach for End-to-End Delay Constrained QoS Routing

P.S.Prakash, S.Selvan

***Abstract*—**QoS Routing aims to find paths between senders and receivers satisfying the QoS requirements of the application which efficiently using the network resources and underlying routing algorithm to be able to find low-cost paths that satisfy given QoS constraints. The problem of finding least-cost routing is known to be NP hard or complete and some algorithms have been proposed to find a near optimal solution. But these heuristics or algorithms either impose relationships among the link metrics to reduce the complexity of the problem which may limit the general applicability of the heuristic, or are too costly in terms of execution time to be applicable to large networks. In this paper, we analyzed two algorithms namely Characterized Delay Constrained Routing (CDCR) and Optimized Delay Constrained Routing (ODCR). The CDCR algorithm dealt an approach for delay constrained routing that captures the trade-off between cost minimization and risk level regarding the delay constraint. The ODCR which uses an adaptive path weight function together with an additional constraint imposed on the path cost, to restrict search space and hence ODCR finds near optimal solution in much quicker time.

***Keywords*—**QoS, Delay, Routing, Optimization

I. Introduction

DELAY sensitive application such as video conferencing, video streaming VoIP etc. require packets to be delivered to the destination within the stipulated time period. At the same time, it is highly desirable to reduce the path cost as much as possible; this may be monetary cost or cost of utilizing network resources. QoS requirements are diverse subject to demands of different applications. Bandwidth, delay, delay jitter and loss ratio are the commonly required QoS metrics. These requirements can be classified into three types[10] viz; concave, additive, and multiplicative .Since the concave type can be converted into the additive constraints by the logarithmic operation. For an additive parameter (e.g., delay, delay-jitter, administrative weight), the cost of an end-to-end path is given by the sum of the individual link values along that path. In contrast, the cost of a path with respect to a non-additive parameter such as bandwidth, is determined by the value of that constraint at the bottleneck link For the mere purpose of determining a feasible path, constraints associated with non-additive parameters can be easily handled via a preprocessing step by pruning all links that do not satisfy these constraints [11].

Performance of operational networks can be improved by engineering Internet traffic so as it is routed over resource-efficient constrained based paths [1].However constrained shortest path problem or multi constrained optimization path selection problem is highly challenging and has been proved to be NP complete [2].In this paper, we analyze the problem of finding a least-cost path subject to an end-to-end delay constraint. It is called as a Delay-constrained Least Cost (DCLC) unicast routing problem, or broadly, a constrained optimization problem. An optimal solution proposed in [3] which performs breath-first search to find the optimum path, thus its running time might grow exponentially. Algorithm proposed by H.Salama [4] try to compute the path distributively in order to overcome the centralized computation overhead, but paths returned by these algorithms may be costly and the path set up time may be too long. Some earlier studies mainly focus on a simpler problem; the multiple-constraints path(MCP), which does not optimize the value of any of the metrics, instead, it only seeks a feasible path that satisfies all the constraints and this problem is NP hard if more than one metric is additive and takes real values[5].In [6] a non- linear function of link cost and delay is proposed to convert the problem into the much simpler single-metric routing problem, and so as to efficiently find a Path that is far away from all the metric bounds.

Since heuristic of MCP problem is easier in terms of execution time than DCLC problem and it appears attractive to convert DCLC into a MCP problem. Based on this, we propose Delay Cost Constrained Routing (DCCR) to rapidly generate a near optimal delay-constrained path in large networks with asymmetric link metrics namely delay and cost. This algorithm first introduces a cost bound according to the network state then it employs the shortest path algorithm [7] with a new non-linear weight function of path delay and cost to search for a path subject to both the requested delay and cost constraint. The search space is reduced as paths that do not satisfy both constraints are pruned-off. In our algorithm, weight function is designed to give more priority to lowest cost paths, and this algorithm is more suitable to solve DCLC problem. As an improvement, we employ an algorithm ODCR to refine the search space. The complexity of this algorithm is asymptotically in the same order as a regular single metric

Manuscript received 14 June 2008.
P.S.Prakash is with Computer Science and Engineering (PG) Department, Sri Ramakrishna Engineering College, Coimbatore, TamilNadu, India. (Phone:+91 422 2312021, 99945 25625; e-mail: prakashpsrajan@rediffmail.com)
Dr. S. Selvan is with St.Peter's Engineering College,Chennai, India.

shortest-path algorithm. We observe by simulation that the cost of the path found by ODCR algorithm is very near to that of the optimal path generated by the much more computationally expensive CBF Algorithm.

II. CDCR NETWORK MODEL AND ROUTING INFORMATION

A. Network Model

A network is modeled by a connected directed graph G(V,E) where V is the set of nodes representing network routers. To each link *l* we associate two parameters c(l) and d(l), where c(l) denotes the communication cost of link l, which can be monetary cost, hop count or any other cost function and d(l) measures the delay that a message experience on link l, this includes the queuing, transmission and propagation delays. A path P(s,d) from a source node s to a destination node d is composed of a set of links $l_1=(s,v_2),l_2=(v_2,v_3),\ldots l_k=(v_k,d)$.The cost of delay of the path P(s,d), denoted by C(P(s,d))and D(P(s,d)) are defined as follows:

$$C(P(s,d)) = \sum_{l \in p(s,d)} c(l) \quad (1)$$

$$D(P(s,d)) = \sum_{l \in p(s,d)} d(l) \quad (2)$$

B. Routing Information

Each node $V_i \in V$ maintains 'cost' and 'delay' information related to its outgoing links. It also maintain two routing table 'Cost Routing Table' and 'Delay Routing Table' that we denote 'CostRTab' and 'DelayRTab' respectively. CostRTab at node V_i contains information about 'Least cost Paths'(i.e; shortest path in terms of cost) from V_i to all destination in the network. DelayRTab at node V_i contains information about the Least Delay Paths (i.e; the shortest path in terms of delay) from V_i to all destinations in the network. These paths can easily be computed by running any distance-vector or link-state algorithms on both cost and delay metrics. CostRTab consists of (v-1) entries, one entry for each other node V_j and V. The entry for node $V_j \in V$. The entry for node $V_j \in V(V_j \neq V_i)$ contain the following items:

- The destination node identity: V_j
- The total cost of the Least cost path $LCP(V_i,V_j)$ from V_i to V_j : $C(LCP(V_i,V_j))$
- The end-to-end delay of the Least cost path $LCP(V_i,V_j)$ from V_i to V_j: $D(LCP(V_i,V_j))$

The next hop on the Least cost path $LCP(V_i,V_j)$ fromV_i to V_j: nh $(LCP(V_i,V_j))$

Delay RTAB consists of (v-1) entries, one entry for each other node V_j in V. The entry for node $V_j \in V(V_j \neq V_i)$ contain following items:

- The destination node identity: V_j
- The total cost of the least delay path LDP (V_i,V_j) from V_i to V_j:$C(LDP(V_i,V_j))$
- The end-to-end delay of the least delay path $LDP(V_i,V_j)$ from V_i to V_j: $D(LDP(V_i,V_j))$
- The next hop on the least delay path $LDP(V_i,V_j)$ from V_i to V_j:nh $(LDP(V_i,V_j))$

C. Algorithm

We present a delay constrained algorithm called Characterized Delay Constrained Routing Algorithm (CDCR).With CDCR a delay constrained path is computed hop-by-hop between a source and a destination node. At each hop, the current node has the choice to continue the path construction either least delay or least cost. The CDCR compares the 'delay-gain' and the 'cost gain' of the two possible paths and only one node may change its routing mode. Once this change is done by an intermediate router all the following nodes should follow the same selected path.

The algorithm CDCR constructs a path P (s,d) from the source node s to the destination node d. Paths constructed by CDCR should satisfy two conditions:

(i) Each path satisfies the delay constraint Δ

(ii) Each path presents a good compromise between end-to-end delay and cost minimization.

Total cost is minimized in order to optimize the overall utilization of the network resources. Average end-to-end delay is minimized in order to offer the best possible services for the underlying real-time application. The path from a source node 's' to a destination 'd' is set –up hop by hop.At each step a set up message is sent to the next hop to inform it about the constrained portion of the path and to help it choosing its own next hop.Path set up process begin when the source node executes the following procedure:

```
If D(LDP(s,d)) < Δ
 then
  if [D(LCP(s,d)) < Δ and Selection_Function
                          (LDP(s,d),LCP(s,d), α )=2]
then next_node ← nh(LCP(s,d))
     routing_mode ← cost
else
     next_node ← nh(LDP(s,d))
     routing-mode ← Delay
cost_sofar ← C(s,next_node)
delay_sofar ← D(s,next_node)
switch_flag ← 0
setup_message ← (d, routing_mode, Δ ,cost_sofar,
                          delay_sofar, switch_flag)
send the setup_message to next node
else
exit (no suitable path is available)
```

The source node 's' starts by checking if the Least Delay Path to 'd' satisfies Δ ($D(LDP(s,d)) < \Delta$), else CDCR stops. otherwise the source continue by checking if the Least Cost Path to 'd' satisfies Δ .If it is the case, source has two possible paths towards destination d, LDP(s,d)and LCP(s,d).If the 'delay_gain' is not greater than α times the 'cost_gain' LCP(s,d)is selected and the source sets the different

parameters as follows:

Next_node ← nh (LCP(s,d))
Routing_mode ← cost

In all other cases LDP(s,d) is selected and source sets the different parameters as follows:

Next_node ← nh (LDP(s,d))
Routing_mode ← delay

After setting the next_mode and the routing_mode, the source sets the rest of CDCR parameters as follows

cost_sofar ← C (s,next_node)
delay_sofar ← D (s,next_node)
switch_flag ← 0

The first intermediate node which changes the routing mode from cost to delay or vice versa will set the switch flag to 1 and no other switching is allowed, all the following nodes in the constructed path will use the same routing mode. The source node 's' then constructs a setup message that contains all the parameters and sends it to next_node.

When an intermediate node $v \neq d$ receives a setup_message it checks first if it has the permission to change the routing mode, if yes then there are two cases:

Consider the following definition atan intermediate node v:

(i) LDCP(s,v,d)= LDP (s,v) + LCP (v,d). LDLP (s,v,d) is the path composed by two subpaths, the least delay path from s to v and the least cost path from v to d.

(ii) LCDP (s,v,d) = LCP(s,v) + LDP(v,d).LDLP(s,v,d) is the path composed by two subpaths, the least cost path from s to v and the least delay path from v to d.Intermediate node v execute the following procedure when $v \neq d$ at the reception of a set-up message

```
if switch_flag=0
then if routing_mode=Delay
then if [D(LDCP(s,v,d))< Δ and Selection_Function (LDP(s,d),LDCP(s,v,d), α ) = 2]
then next_node ← nh(LCP(v,d))
      routing_mode←cost
      switch_flag←1
else
   next_node←nh(LDP(v,d))
else
   if Selection_Function(LCDP(s,v,d),LCP(s,d), α ) =1
then
   next_node ← nh (LDP(v,d))
   routing_mode←Delay
   switch_flag←nh(LCP(v,d))
else
   if routing_mode = Cost then next_node ← nh(LCP(v,d))
   if routing _mode = Delay then next_node ← nh(LDP(v,d))

   cost_so_far ← cost_so_far +C (v, next_node)
   delay_so_far← delay _so_far +D (v, next_node)
   setup_message←(d,routing_mode, Δ ,cost_so_far,delay_
                                    so_far,switch_flag)
send the setup_message to next_node
```

D. Path Comparison

We present the comparison function in terms of delay and cost that we use to select paths in our algorithm. Assume that we have two different path $P_1(s,d)$ and $P_2(s,d)$ from a source node 's' to a destination node 'd' such that $C(P_1) \geq C(P_2)$ and $D(P_1) \leq D(P_2)$.This means that P_1 is better than P_2 in terms of delay but more expensive in terms of cost.If both paths satisfy a given delay constraint Δ, so source node 's' has the choice to set up path P_1 or P_2 as its real path toward node 'd'. As $D(P_1) \leq D(P_2)$, delay-gain of P_1 compared to path P_2 as follows:

$$\text{Delay-Gain}(P_1, P_2) = D(P_2) - D(P_1)/ D(P_2) = 1 - D(P_1)/ D(P_2) \quad (3)$$

As $C(P_2) \leq C(P_1)$, cost gain of P_2 compared to path P_1 as follows:

$$\text{Cost-Gain}(P_2, P_1) = C(P_1) - C(P_2)/C(P_1) = 1 - C(P_2)/C(P_1) \quad (4)$$

The source 's' chooses path P_1 (i.e; better delay) rather than P_2 (i.e; better cost) if the following inequality is verified.

$$\text{Delay-Gain}(P_1, P_2) \geq \alpha \text{ .Cost\_Gain}(P_2, P_1) \text{ where } \alpha \geq 1$$

If the inequality is not satisfied, the source 's' chooses path P_2 (i.e; better cost).

E. Constructed Paths

During the path construction process only one intermediate node is authorized to change the routing mode and this leads to the construction of a path which has one of the following four forms:

(i)The path is the Least cost path between s and (LCP(s,d)) The source has chosen the cost as routing mode, and no intermediate node has switched to Delay routing mode.

(ii)The path is the Least Delay Path between s and d LDP(s,d):

The source has chosen the delay as routing mode, and no intermediate node has switched to cost routing mode.

(iii)The path is composed by two sub paths, the least cost path from 's' to an intermediate node 'v' and the least delay path from this intermediate 'v' to 'd', this is known as LCDP(s,v,d).This means that the source has chosen the cost as routing mode, and intermediate node 'v' has switched to the Delay routing mode.i.e; LCDP(s,v,d)=LCP(s,v) + LDP(v,d)

(iv) same as (iii) but delay and cost are interchanged

III. CDCR RESULT ANALYSIS

In this section, we provide an overview of our simulation model and some of the results we obtained by comparing our algorithm with other algorithms

In these simulations we look for the effect of the delay constraint on the end-to-end delay and on the total cost. Simulations are carried over a set of Random Euclidean graphs generated using a modified version of Waxman Algorithm [12].

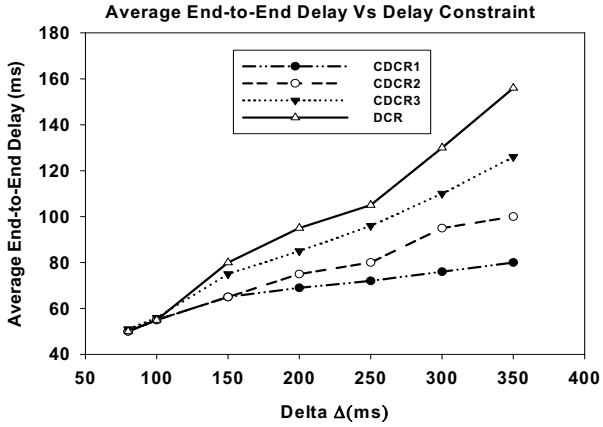

Fig. 1 Average End-to End Delay Vs Delay Constraint

A random Euclidean graph is generated by distributing nodes on an Euclidean plane uniformly and adding edges between nodes on a probabilistic basis. We used graphs with 100 nodes and average degree of 3.We assign high delay values uniformly distributed within [90ms,100ms] to 20% of links and to all the rest we assign delay values uniformly distributed between [1ms,10ms].Since Internet traffic load is skewed , with most links under utilized and a few links heavily congested, it seems logic that this can be expressed in terms of link delay, since in a highly loaded link we possibly experience a high delay. Link cost is uniformly distributed within [1,100] interval.

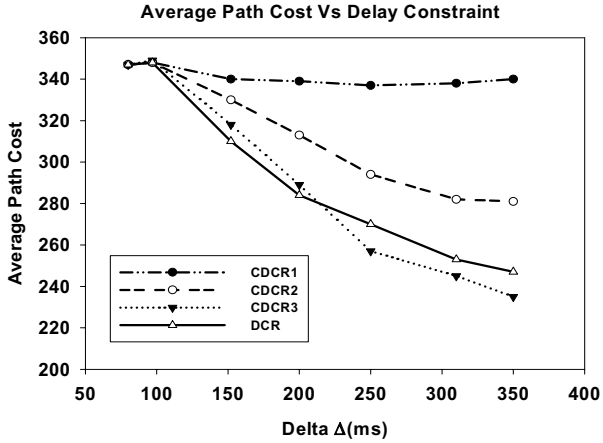

Fig. 2 Average path Cost Vs Delay constraint

In our algorithm we consider three values of α for CDCR α =1 (CDCR1), α =2(CDCR2) and α =3 (CDCR3). We compare end-to-end delay and the average path cost while varying the delay- constraint from 50 to 350 ms. We also simulate both the Least cost path algorithm (LCP) and Least Delay Path Algorithm (LDP) because they represent the two limits for PDCR and DCR. We found that when Delay-constraint is tight, the delay constrained routing algorithm produces paths close to the Least Delay Paths and as the delay-constraint becomes wider, the Delay Constrained Routing Algorithm produces paths close to least cost paths. However, the CDCR produces paths that converge more or less rapidly to the Least Cost Path as the delay constrain increases.

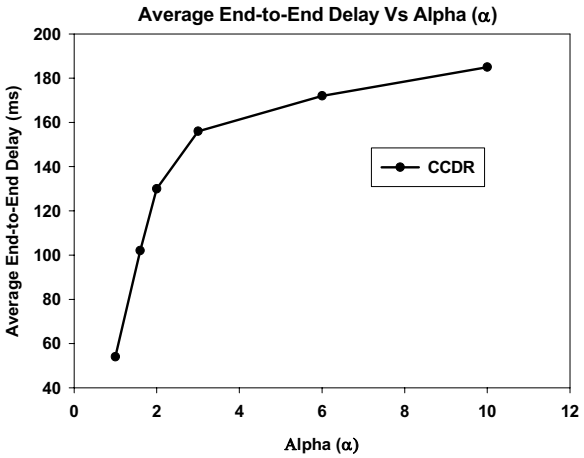

Fig. 3 Average End-to-End Delay Vs Alpha (α)

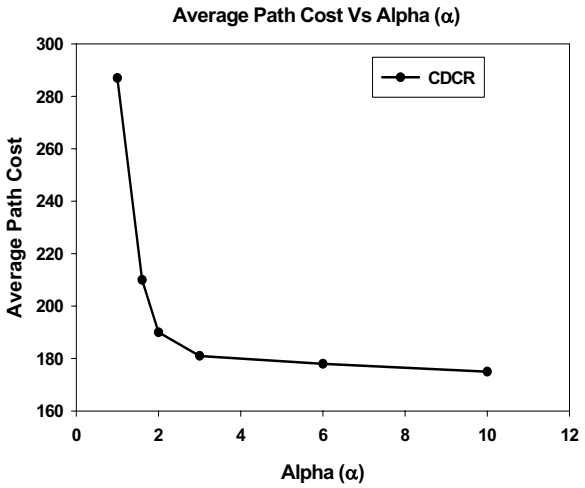

Fig. 4 Average Path Cost Vs Alpha (α)

CDCR maintains a balance between delay and cost minimization for constructed paths. These paths are close to or far from LCP and LDP depending on the values of α .When α =1, CDCR produce paths close to least delay and when α increases paths becomes closer to least cost path.

Paths in the middle way between LDP and LCP are obtained by values of α between 1.5 and 4, so that this leaves some freedom for application to choose the best value of α in order to produce the best suitable paths

IV. PROBLEM DESCRIPTION OF ODCR

We represent the network by directed graph G = (V,E) where V is the set of all vertices (nodes), representing routers, E is the set of edges (links) representing physical or logical connectivity between nodes. All links are bidirectional which means that the existence of a link e=(u,v) from node ‘u’ to

node 'v' implies the existence of another link e' =(u,v) for any u,v $\in$ V. Any link e $\in$ E has a cost c(e):E $\rightarrow R^+$ and a delay d(e): E$\rightarrow R^+$ associated with it, where R^+ is set of non-negative real numbers. The function c (.) defines the measure we want to constrain. Since computer networks are asymmetrical, it is possible that c (e) $\neq$ c (e') and d (e) $\neq$ d (e').

For a given source node s $\in$ V and destination node d $\in$ V, P(s,d) = P_1 ……..P_m is the set of all possible paths from s to d. The cost and delay of a path P_i is defined as :

$C(P_i)= \sum_{e \in Pi} c(e)$ and $D(P_i) = \sum_{e \in Pi} d(e)$ respectively. A delay constraint Δ_d is specified by the application as a performance guarantee.

A. Delay Constrained Least Cost Problem (DCLC)

Given a directed network G, a source node s, a destination node d, a non-negative link delay function d(.), a non-negative link cost function c(.).For each link e $\in$ E and a positive delay constraint Δ_d, the constrained minimization problem is to find a path satisfies $P_i \in P'$ (s,d) if and only if $D(P_i) \leq \Delta_d$ where P' (s,d) $\subseteq$ P(s,d) is the set of paths from s to d for which the end-to-end delay is bounded by Δ_d .

The DCLC problem involves optimizing one or more variables and imposing constraints on other variables. A variant called Multi Constraint Path (MCP) problem only searches for a feasible solution for which all variables are bounded by the constraints. A variant of MCP is Delay Cost Constrained (DCC) which can be stated as the DCLC problem except the objective is to find a path $P_i \in P'(s,d)$ if and only if $D(P_i) <$ and $C(P_i) \leq \Delta_c$ where Δ_c is the application specified cost bound.

B. Description of ODCR Algorithm

We transform DCLC into a DCC problem by defining a sufficiently loose cost bound so that original DCLC could be easily solved. Now we solve a DCC problem by taking least - delay path is selected as the cost-bound. Now we search for a feasible path of possibly highly delay and lower cost. If such a path exists the algorithm returns that path, else it returns the least-delay path itself. Thus we can convert the original DCLC problem into the problem of searching for near-optimal path in the solution space of this new DCC problem. Now we need to examine the paths that satisfy both the requested delay and introduced cost bound. For this, we define a weight function which combines all features of the link metrics so that by optimizing the weight, a solution arrived that optimizes all link metrics simultaneously.

Since linear weight functions are slow convergence especially if numbers of paths are more, we now define a non-linear weight function to overcome this difficulty. By using a path weight function $\max[C(P)/\Delta_c, D(P)/\Delta_d]$, the algorithm finds the shortest path with both cost and delay are far from their bounds. With a non-linear weight function, it is now the weight of a path is no longer the sum of the weight of all links on this path. i.e; $W(P) \neq \sum_{e \in p} w(e)$ .But since it is easy to record the cumulative delay and cumulative cost of a path, we can easily solve this problem by computing the path weight as a function of F(.) of the delay and cost of the path. i.e; W(P)=F[C(P),D(P)].In non-linear functions sub sections of least-weight paths are not necessarily shortest path themselves. This is called 'optimal sub-structure property'. This may result in a shortest path algorithm, may sometimes fail to find least-weight (shortest path).

The weight function used in this algorithm is

$$W(P_i^u) = \begin{cases} D(P_i^u)/1 - C(P_i^u/\Delta c) & \text{if } D(P_i^u) \leq \Delta d \text{ and } C(P_i^u) \leq \Delta d \\ \infty & \text{otherwise} \end{cases}$$

$$\text{i.e; path weight} = \frac{PathDelay}{1 - \frac{PathCost}{CostBound}} \quad (5)$$

Where $P_i^u \in P(s,u)$ is the i^{th} path from source node s to node u found by the algorithm. Since our objective is to find a path with least cost, we are using this weight function that gives priority to low-cost paths. With our definition the path weight has an exponential growth with the path cost, and is only linearly proportional to path delay.

Our algorithm adopts greedy strategy and uses a non-linear weight function in searching for best solution. Since non-linear weight function does not have 'optimal-substructure property' we first employ k-shortest path algorithm[8] which finds shortest path in increasing weight order, for each node and we can choose the path with lowest cost in the final stage as the best feasible solution.

Our algorithm restricts the search space by only examining paths that satisfy the requested delay bound and cost bound. Here the cost bound is taken to be the cost of the least delay path. This is justifiable since if there is no path with lower cost than that of the least-delay path, then the least-delay path itself is the optimal path and is returned by our algorithm. However this cost bound may be too loose especially when the relationship between cost and delay is inversely proportional to each other. Since the weight of all infeasible paths to be infinity, it is easy to see that if we use a tighter cost bound, the number of possible feasible solution decreases and the opportunity that this algorithm finds the optimal least-cost solution increases.

We use another heuristic, to search for a tighter cost bound, proposed by handler. It uses a linear function of the link delay and cost to compute link weight but it adjusts the weights given to cost and delay in the weight function according to the quality of the current path, thus it iteratively approaches the optimal solution.

The algorithm has two parts namely Least-Delay Path (LDP) and the Least Cost Path (LCP) computed using any shortest-path algorithm with the weight function being link delay and cost respectively. If LDP is a feasible path, then the algorithm returns this feasible path. If it is not feasible path at each iteration, the algorithm maintains two paths, the current best infeasible path LDP and the current best infeasible path LDP. It then defines two parameters α and β to construct a new linear path weight function w(p)= α x D(p) + β x C(p) for each path p. Using this new linear function of link cost and delay, the algorithm tries to find a new path Least Weight Path (LWP) with least weight so as to reduce both path cost and delay. When W(LWP) < ρ where ρ is current least path weight and LWP is feasible (i.e; D(LWP) ≤ Δ_d), LWP replaces LDP to become the best feasible path, thus the weight given to link cost increases in the next round which means that lower cost paths are given most preference. If LWP is infeasible, LWP replaces LCP in the next iteration, thus the weight given to link delay increases, which means that lower delay paths are given more preference. The algorithm stops when W(LWP) = ρ and returns the best feasible path out of LWP and LDP as the near-optimal solution.

The path found by Handler algorithm is still not the optimal path due to this inherent weakness of the linear weight function. But its cost is close enough to the optimal cost to be efficiently used as a tight cost-bound for DCCR and this enhanced algorithm named Optimized Delay Constrained Routing (ODCR) since using a tighter cost bound is a mechanism to reduce the search space.

C. Validity of ODCR

(i) There exists k such that ODCR always returns a delay constrained path for a given source 's' and destination 'd', if such a path exists.

If no feasible path exists, i.e; the delay of each path that connects 's' and 'd' is greater than the delay bound, then the minimum path weight computed at node 'd' will have a weight of infinity and ODCR returns no path when the search is completed.

We prove by contradiction that ODCR return a path if one or more feasible paths exist. If there are one or more feasible paths, the possible reason for ODCR to return no paths is if the algorithm finds no feasible path leading to an intermediate node along the feasible path from 's' to 'd'. In other words, let $P_i^d = \{ s, v_1, v_2, \ldots v_m, d\} \in P(s,d)$ be a feasible path from 's' to 'd' and v_1 to v_m are intermediate nodes, we would have the following two conditions satisfied:

$$\exists\ P_i^d \text{ such that } D(P_i^d) \le \Delta_d \quad (6)$$

$$\exists\ V_j \in P_i^d \text{ such that } \forall\ P_i^{Vj} \in P\ (s, v_j),\ D(P_i^{Vj}) > \Delta_d \quad (7)$$

(ii) Since delay is an additive metric and non-negative it is not possible that the sub-path of a feasible path is not feasible. This shows that ODCR can always find a feasible solution if it exists. The final path returned by ODCR for a given source's' and destination'd' is loop-free.

Since the algorithm does not visit dominated paths, a path that contains a loop is never recorded and thus the final k-shortest paths recorded at node'd' are loop-free and so is the final path returned.

V.SIMULATION RESULTS OF ODCR

A discrete-event C++ simulator is used to investigate the performance of different algorithm in a realistic communication environment. We used the graph generation process as in [9] where the positions of the nodes are lie in a stipulated area. We fixed the position of the source node 's' and the destination node 'd' such that 'Manhattan distance' between 's' and 'd' is the longest possible distance in the graph. The average node degree is 3 and approximately what the situation is in current networks.

The link delay function consists of the propagation delay function T_p, the transmission delay T_t and the queuing delay T_q. Since the network is high speed in nature the transmission delay is negligible $=T_q/T_p$,the ratio between the queuing delay and propagation delay and this parameter shows the traffic condition in the network. The link delay is defined as d(e) = (1+ τ) x T_p. We let τ be uniformly distributed in [0,T] where T is maximum queuing delay allowed at each switch. Larger the value of T, the more likely the generated network is asymmetric. Assigning link cost is a challenging job since it affects the difficulty in finding the optimal path. If link and cost are directly proportional to each other, then it is enough to just use a single metric shortest path algorithm.

In our simulation model we consider the negative correlation between cost and delay and we define link cost as $c(e)=\frac{M}{c+d(e)}$ .We choose m=500 and c=1 in our simulation and d(e) varies from 0.1 to 15. Since tightness of the delay bound might affect the performance of the algorithms under investigation, we choose the delay bound based on the configuration of the graph. Each time a new graph is generated, a shortest path algorithm is used to find least-delay path and least- cost path, then compute the delay of these two paths, denoted by D(LDP) and D(LCP) respectively. We define the delay bound Δ_d as Δ_d = D(LDP) + ψ[D(LCP)-D(LDP)] where ψ is called delay bound ratio and lies between 0 and 1.In our simulation we chose ψ = 0.5.

We selected k=3 and M=5 for the network size starting from M=100. K is the number of shortest paths maintained from the source to each node and M is the number of iterations executed to compute a tight cost bound for our algorithm. Here K and M are much smaller than the network size and such a short value is enough to produce good performance. To measure the inefficiency and speed of the algorithm, CBF is used since it provides the optimal solution in terms of path cost. Thus we define the inaccuracy (i.e; inefficiency) of an algorithm as the cost different relative to the ratio of the CBF path.

Inefficiency (A)=$[C(P_A)-C(P_{CBF})]/C(P_{CBF})$. We also measure actual execution time of each investigated algorithm.

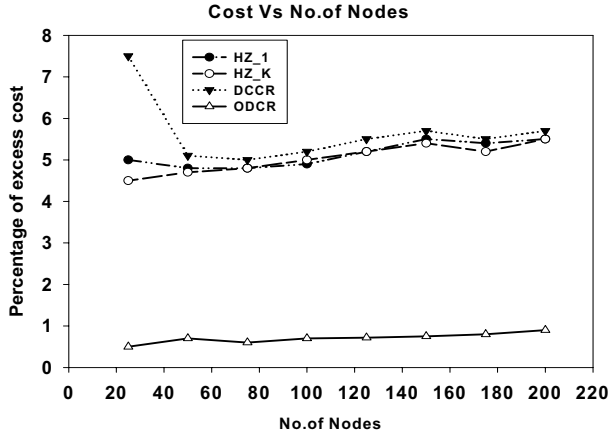

Fig. 5 Percentage of excess cost w.r.t CBF Vs Number of Nodes

Fig. 5 and Fig. 6 show the performance measures of different heuristics. Since link cost and delay are inversely proportional the least-delay path can cost as high as three times that of the optimal path. Our improved ODCR algorithm shows an attractive cost performance; the relative excess cost of ODCR always remain under 1%.This percentage of excessive cost with reference to CBF is shown in Y-axis. We can also see that the relative order and scale of cost difference does not change much with the network size. In our algorithm value of k can be kept small even for a large network.

Fig. 6 shows the data for all algorithms except the CBF algorithm.HZ_1 can converge very fast to the final solution even though an analytical bound does not exist. The speed of DCCR is slightly slower than HZ_1 since DCCR uses a non-linear path weight function and requires a k-shortest path algorithm. The proposed ODCR algorithm runs in almost the same speed as the original DCCR algorithm, which implies a more efficient search under ODCR.

We also compared the speed of the optimal CBF solution and ODCR algorithm in Fig.7. It is clear that the CBF algorithm has an exponential growth with the network size in terms of execution time, as opposed to the polynomial growth of ODCR algorithm. Fig. 8. shows the effect of delay bound on the performance .We can see that the relative excess cost of HZ_1 and DCCR is increasing on the delay bound gets looser.

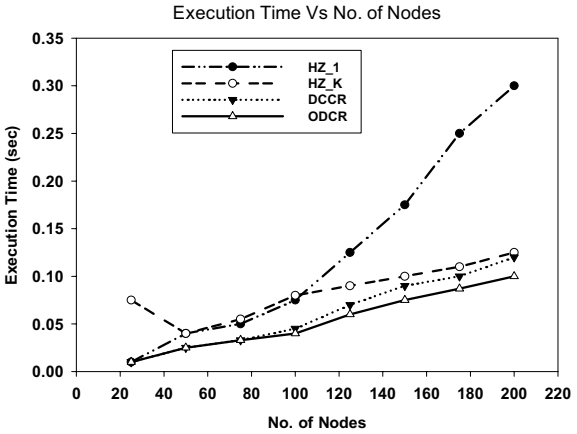

Fig. 6 Execution time Vs Number of Nodes

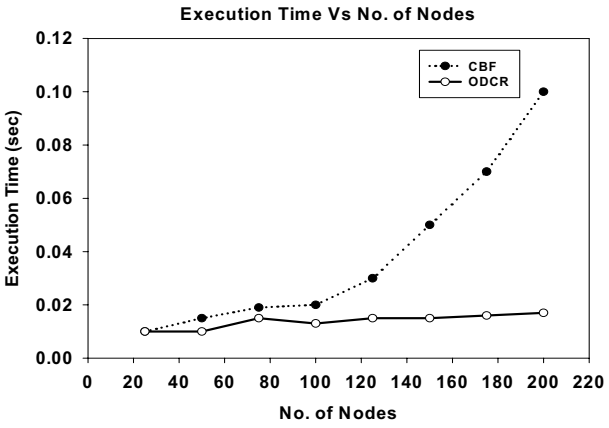

Fig. 7 Execution Time Vs Number of Nodes

This is because a looser bound will enlarge the solution space, thus the capability of these algorithms becomes limited by either the weakness of linear weight function or fixed value of k. However, the performance of ODCR is less sensitive to the delay bound since the cost bound given by HZ_1 heuristic is tight enough to restrict the number of feasible paths. All the above simulations assume that the link cost and link delay are inversely proportional to each other and increases the difficulty in finding the optimal cost.

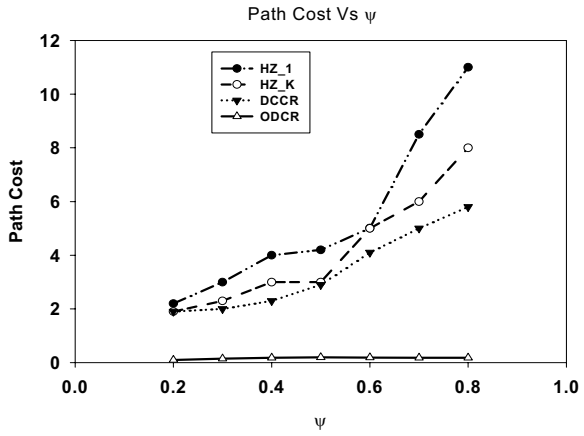

Fig. 8 Percentage of excess cost relative to CBF Vs ψ

VI. Conclusion

We analyzed an algorithm CDCR based on efficient characteristic selection function that provides cost minimization and meeting delay constraint. We proposed a quick convergence algorithm ODCR is presented in this paper. This algorithm uses a non-linear path weight function to make the path search more accurate and quicker. The results from extensive simulation show that even under the most difficult situation where the link cost and delay are inversely proportional to each other, our improved ODCR algorithm always return very quickly a feasible path whose cost is very close to that of the optimal one.

References

[1] Internet traffic Engineering IETF Working Group ftp://ftpext.eng.us.uu.net/tewg
[2] Z.Wang and J Crowcroff "Quality-of-Service routing for supporting Multimedia Applications. IEEE Journal on Select Areas in communication14(7):1188-1234 September 1996..
[3] R.Widyono, "The Design and Evolution of Routing Algorithm for Real-Time channels" Technical Report ICSI,International Computer Science Institute V.C.Berkeley, June 1994.
[4] H.F salama,D.S.Reeves and Y.Viniotis. "A Distributed Algorithm for Delay –Constrained Unicast Routing.In Proc IEEE INFOCOMM '97,April 97.
[5] J.Chen, "New approaches to routing for large scale data networks", Ph.D Thesis, Department of Computer Science, Rice University, 2000.
[6] H.De Nere and Van Mieghan, "A multiple Quality of Service Routing Algorithm for PNNI. IEEE ATM '98, PP 306-314, May 1998.
[7] E.I Chong,S.Maddila and S. Morley " On finding Single source single Destination k shortest paths, International Conference on Computing and information '98 pp40-47 July 1998.
[8] Y.Lenug, G.Li, Z.B.Xu, " A generic algorithm for multiple destination routing Problem" IEEE transaction of Evolutionary Computation, 2(4),2001.
[9] Inet Toopology Generator http://topology.eecs.unich.edu.
[10] S.Chen, K.nahsted, "An overview of quality of service routing for next-generation high-speed networks:problems and solution", IEEE network 12(6), pp64-79,1998
[11] Z.Wang, J.Crowcroft, "Quality-of-Service routing for supporting multimedia application, IEEE Journal on selected Areas in communications pp1228-1234,14(7),1996.
[12] Bernard M.Waxman, "Routing of Multipoint Connections", IEEE Journal on Selected areas in Communications,Vol 6, No.9, Dec 1988,pp 1617-1622.

Dr. S.Selvan is serving as Principal at St.Peter's Engineering College, Chennai, India. He has more than 3 decades of teaching experience and published about 75 papers in international, national journals and conference proceedings. His areas of research include computer networking and soft computing.

P.S.Prakash received B.E(EEE) and M.E(Electrical Machines) from P.S.G College of Technology, Coimbatore, India and also received M.E in Computer Science and Engineering from Bharathiar University, India. His research interests include QoS Routing, Network Security and Management. He is currently pursuing Ph.D in Computer Science and Engineering and serving as Assistant Professor at Sri Ramakrishna Engg. College, Coimbatore, India.