

A Limited Path Unicast QoS Routing Algorithm

¹YanXing Zheng, ²Jing Tian, ¹ZhiFeng Liu, ¹WenHua Dou

¹School of Computer, National University of Defense Technology, Chang Sha, P.R.China, 410073

²School of Mechanical Engineering & Automation Control, NUDT, Chang Sha, China 410073

mlinux@163.com

Abstract

In this paper, We propose a novel limited path algorithm called LPUA to deal with two constrained unicast QoS routing problems, which is known as NP-complete. By combining two weights linearly into a single one, Dijkstra algorithm based on the single weight can be used directly to return a least cost path between the source node and the destination node. When a routing request comes, LPUA only needs to compute a limited number of paths. If any of these paths cannot meet the routing constraints, the routing request will be refused. Extensive simulations have show that LPUA is very efficient.

1. Introduction

Today there are more and more applications that need service guarantees in order to function properly. Most multimedia applications have QoS requirements, such as delay, cost, bandwidth etc. The metrics of QoS constraints can be classified into three categories: additive, (e.g., delay), concave, (e.g., bandwidth), and multiplicative, (e.g., loss rate). Concave metrics can be dealt with easily by doing some reduction operation before running the routing algorithm. Multiplicative metrics can be turned into additive metrics. So the challenge that QoS routing must face up to comes mainly from additive metrics.

Definition 1. Multi-constrained path (MCP) problems Consider a network $G(N, E)$, N is the node set. E is the link between nodes. Each link is specified by a metric vector ω , which is m -dimensional. Each component of ω is an additive QoS metric: $\omega_i(u, v) \geq 0, i = 1, \dots, m$. Given m constraints $L_i, i = 1, \dots, m$, the problem is to find a path P from a source node s to a destination node d such that:

$$w_i(P) \triangleq \sum_{(u,v) \in P} w_i(u,v) \leq L_i \text{ for } i = 1, \dots, m. \quad (1)$$

A path that satisfies all m constraints is often referred to as a feasible path. There may be many different paths in the graph $G(N, E)$ that satisfy the constraints, i.e., the solution of MCP is a set of paths. When $m=2$, MCP becomes two constrained QoS routing problems.

Multiple additive constrained QoS routing problems are known to be NP-complete, resolving which is much time consuming. Lots of heuristic algorithms have been presented to address it [1~12]. Orda [2] and Lorenz *et al.* [3] proposed a modified ϵ -Optimal heuristic algorithm to scale better in hierarchical networks. Reeves and Salama [4] presented a distributed heuristic algorithm, DCUR, for the RSP problems. The algorithm explores the network topology based on two segments: (1) the so-far explored path from the node to the destination node u ; and (2) the least-delay path from the node u to the destination node. Juttner [5] presented a Lagrange relaxation based method, which combines the delay and cost linearly, the key issue is how to choose appropriate combination weights. Jaff [6] presented an approximate algorithm to deal with two constrained problems. Iwata *et al.* [7] proposed a polynomial-time algorithm to solve the MCP problems. H.De. Neve, *et al.* [8] proposed a tunable multiple constraints routing algorithm TAMCRA. Korkmaz and Krunz [9] presented a heuristic algorithm H_MCOP, which tries to find a path within the constraints by using a nonlinear path length function. A thorough evaluation of the most important constrained based routing algorithms presented before 2002 can be found in [10].

Yong Cui [11] *et al.* proposed a pre-computation algorithm, MEPPA, for multi-constrained QoS routing problems. MEPPA convert the multiple weights to a single value by combining them linearly. Then Dijkstra algorithm can be used based on a single weight.

The main limitation of these algorithms is that to keep the computation complexity under what we can afford, the algorithms sometimes cannot find a feasible path even when it does exist.

The rest of paper is organized as follows. Section II discusses the related works. The main idea of 2EFPA is discussed in section III. The idea of LPUA and its theory foundation are discussed in part IV. The performance of LPUA is evaluated by simulations in part V. Part VI concludes this paper.

2. Related work

The straightforward idea to deal with multiple constrained problems is combining the weights linearly as proposed by Jaffe [6].

$$l(u \rightarrow v) = \sum_{i=1}^k d_i w_i(u \rightarrow v) \quad (1)$$

Where $l(u \rightarrow v)$ is the cost of link $u \rightarrow v$, $w_i(u \rightarrow v)$ is i 'th weight of link $u \rightarrow v$, d_i is the weight coefficient. If the weight metric is additive, then the cost of path P is:

$$w(P) = \sum_{i=1}^k d_i w_i(P) \quad (2)$$

Where $w_i(P)$ is the sum of i 'th weight of the path P . Then the problem is transformed to a single parameter problem enabling the use of Dijkstra's shortest path algorithm. Running the Dijkstra algorithm on the network topology with link weights condensed in this way will return a path between the source-destination pair (s, d) that minimizes the cost defined by (2). Neve [8] defined the following path cost function:

$$w(P) = \sum_{i=1}^k \left\{ \frac{w_i(P)}{c_i} \right\}^q \quad (3)$$

The cost of the whole path is a non-linear combination of the individual path costs. When $q \rightarrow \infty$, Neve pointed out that the cost function defined in (3) is equivalence with that defined in (4):

$$w(P) = \max \left\{ \frac{w_1(P)}{c_1}, \frac{w_2(P)}{c_2}, \dots, \frac{w_k(P)}{c_k} \right\} \quad (4)$$

Based on path cost function defined in (4), Neve proposed TAMCRA algorithm. TAMCRA uses Dijkstra algorithm to return the least cost path. For the reason that according to the path function defined in (4), subsections of least cost paths are not necessarily least cost paths themselves, so TAMCRA stores k paths per node via a k -shortest path implementation of Dijkstra algorithm. The time-complexity of the

TAMCRA is $O(kn \log(kn) + k^3 m)$. If we take k a large value, the probability of missing the least cost path is very small. But a large k will lead to too much computation complexity.

Korkmaz [9] has shown that the cost function defined in (3) is equivalence with that defined in (4). Based on the same path cost function with [8], [9] proposed H_MCOP algorithm. H_MCOP tries to find a path within the constraints and minimize simultaneously the cost of the single measure along the path. To achieve both objectives at the same time, H_MCOP executes two modified versions of Dijkstra algorithm in backward and forward directions. In the backward direction, H_MCOP uses the Reverse_Dijkstra algorithm for computing the shortest paths from every node to the destination node. In the forward direction, H_MCOP uses a modified version of Dijkstra algorithm called Look_Ahead_Dijkstra. Look_Ahead_Dijkstra algorithm discover a middle node u , it uses the information that the already traveled sub-path from s to u and the estimated information of the remaining path from u to d . Since H_MCOP considers complete paths before reaching the destination, it can foresee several infeasible paths during the search process. The complexity of the H_MCOP is $O(N \log N + mE)$. If one deals only with the MCP problems, then the H_MCOP should be stopped whenever a feasible path is found during the search in the backward direction, reducing the computational complexity.

Xin Yuan [12] proposed two heuristic algorithms, the limited granularity heuristic and the limited path heuristic. The idea of the two algorithms is to limit the number of optimal QoS paths maintained in each node. Thus each node is not able to record all optimal QoS path from s to d and the heuristics can only find approximate solutions. Hence the challenge of the heuristic is how to limit the number of paths maintained by each node while keeping the effectiveness in finding paths that satisfy Routing request.

Yong cui [11] proposed a pre-computation algorithm called MEPPA, for multi-constrained QoS routing problems. MEPPA uses the path cost function defined in (2) and tries to satisfy all preferences one has, namely MEPPA tries to use Dijkstra algorithm based on all linear combinations of coefficient to compute the least cost path. For the discreteness of the network topology, MEPPA cares each weight into b degrees. Hence the number of weight coefficient combinations is a limit value. Yong Cui has shown by simulations that MEPPA is better than H_MCOP in most cases. The limits of MEPPA are that (a) the path

returned by MEFPFA is redundant which means that MEFPFA perform some extra-computation and the routing table is large. (b) As a pre-computation algorithm, it is definitely affected by the inaccurate routing information.

As reviewed above, previously proposed algorithms will fail to find the least cost path in some sense sometimes. If they try to reduce the probability of losing the least cost path, a high computation complexity will be introduced. For the reason that delay and cost are the most common two additive weights that are researched in MCP problems, we proposed a limited paths unicast routing algorithm LPUA to deal with two-constrained QoS routing problems. The main contribution of this paper are: (a) we propose LPUA which has a higher ratio success than MEFPFA; (b) LPUA uses the same space partition with that of MEFPFA, but it avoid extra computation to some extent and the routing table maintained is small. Later we care only about two constrained QoS routing problems and call MEFPFA 2EFPA.

3. Idea of LPUA and Its Theory Foundation

Because of the discreteness of the path in real network topology, the path found is discrete when the weight coefficients change continuously. So we do not need all possible values of (α, β) , $\alpha + \beta = 1$. Yong Cui cares each QoS weight to b degrees, i.e., each weight coefficient can be $0/b, 1/b, \dots, (b-1)/b$. Then for k weights, there are $B = C_{b+k-2}^{k-1}$ different combinations of weight coefficient. Each combination will be used for Dijkstra algorithm to compute a least cost path. So there are B paths that are stored in routing table between a source node and a destination node. The size of routing table is B times the original routing table with a single weight. When a routing request comes, the router looks up a feasible path in the routing table and forwards the request. If there is no path in the routing table that can meet the request, the request is refused. Yong Cui compared MEFPFA with H_MCOP and drew the conclusion that for two constraints problems, when we take $b=7$, MEFPFA is better than H_MCOP not only on computation complexity, but also on the running time in practical experiments.

Of course we can let b be a large number, then there will be more combinations of weight coefficient and we can find more paths which means that the

unknown area will be small and more routing request will be met. While large number of B means running Dijkstra algorithm B times, which will lead to too much computation complexity. So there must be a tradeoff between performance and computation cost. Fortunately, extensive simulations have shown that if we select several different combinations of weight coefficient properly, we can get most of least cost paths. Thus the problem becomes how to select weight coefficients, such that, most of least cost paths will be found while the computation complexity is under what we can afford.

2EFPA behaves very well as a pre-computation algorithm, but there are still some limits on it. The most important one is 2EFPA brings itself with some redundant paths, which directly means extra computation cost and a large routing table.

We have done extensive experiments to verify that the paths returned by 2EFPA are redundant. For each experiment, we generate randomly a Waxman network topology, select randomly a source node, and 20 destination nodes and stat the average redundant paths between the source node and 20 destination nodes.

We do such experiment over a network with a node number N being 50, 100, 200, 500. For each node number, we carry out experiment 100 times. Each link in the randomly generated topology has two weights, and each weight $w \sim \text{uniform}[1, 300]$.

Figure 1 shows the results of the experiments. X-coordinate is the ID of experiment (EID). Y-coordinate is the number of average redundant paths (ARP).

From the simulations, we can draw the following conclusions: (1) 2EFPA performs some extra computation especially when b is large, e.g., $b=20$. (2) The average of ARP is no less than 3 paths when $b=7$; the average of ARP is no less than 6 paths when $b=10$; the average of ARP is no less than 15 paths when $b=20$. Simulations tell us that MEFPFA performed some extra computation. Then whether it means we can let b be a small value? When b is small, we find few least cost paths, which leads to a large unknown area. Then a request, which can be accepted in fact, will more easily be refused than ever. So only by reducing b to avoid path redundancy is not a good idea.

We propose an efficient QoS routing algorithm based on dynamic weight coefficient to deal with two-constrained routing problems.

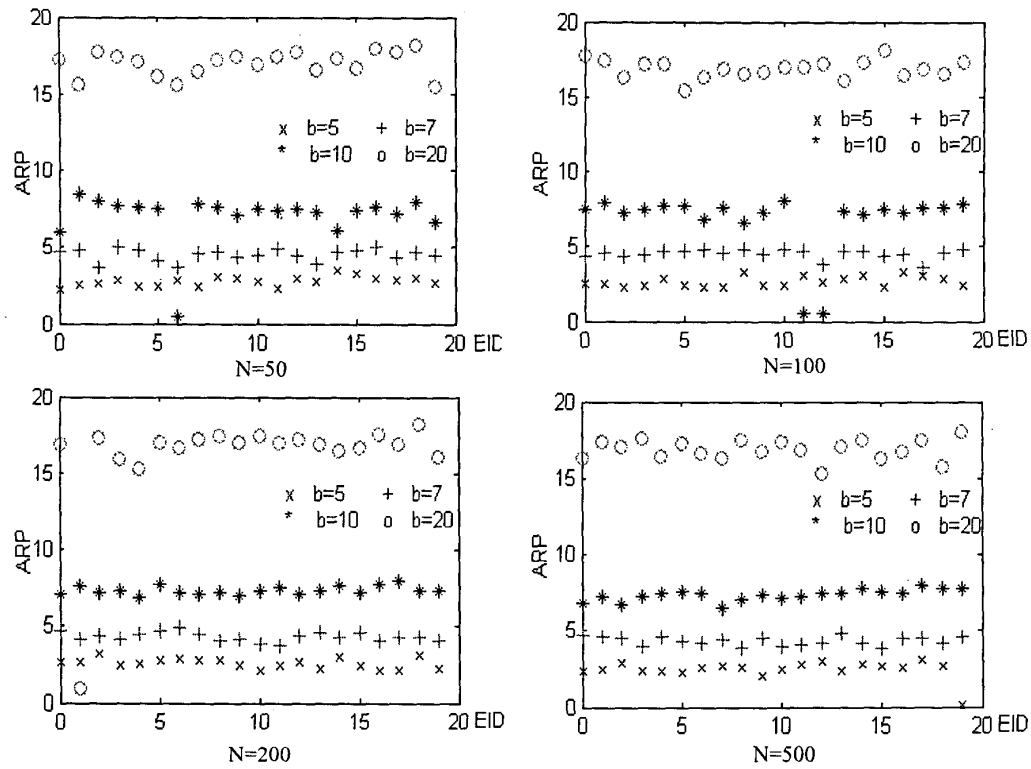


Figure 1. Paths found by 2EFPA are redundant

In [13], we showed that If two different weight coefficient combinations: (α_1, β_1) , (α_2, β_2) correspond to the same path, then we need not calculate any (α, β) that α lies between α_1 and α_2 . Thus we can avoid extra computation to some extent. Based on this idea, we propose a two-constrained dynamic weight coefficient unicast QoS routing algorithm, LPUA.

At first, LPUA computes two least cost path P1 and P2 by running Dijkstra(1,0) and Dijkstra(0,1). If P1 and P2 are the same paths, LPUA returns P1 and ends. Otherwise, LPUA stores P1 and P2, and performs a search in a direction that is perpendicular line P1 and P2. Then LPUA starts dealing with an interval operation until the termination conditions are met.

Figure 2. demonstrates the main steps of LPUA. We can see that LPUA use a more intelligent mechanism to decide the search directions.

```

LPUA(G, s, d)
1)Search in (1,0) direction and find P1
2)If P1 satisfies the quest, return;
3)Search in (0,1) direction and find P2
4)If P2 satisfies the quest, return;
5)If P1= P2
   Return P1
6)search in a direction that is perpendicular
   to line P1P2 and find P3
7)If P3 satisfies the quest, or P3 is the same
   with one of P1 and P2
   return;
8)Else
   If routing request lies between P1 and P3,
     Replace P2 with P3
   Else Replace P1 with P3
9) Goto step 6
  
```

Figure2. LPUA algorithm

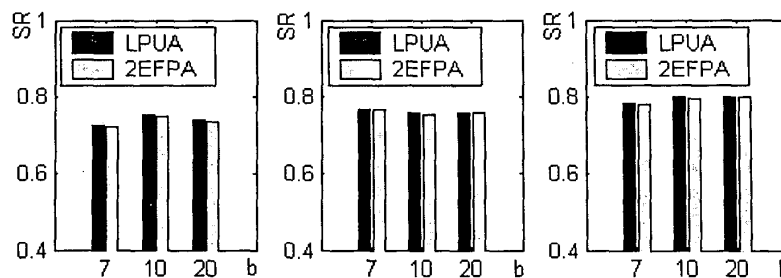


Figure 3. Comparable performance of LPUA and 2EFPA

We have Stat. the running times of Dijkstra algorithm when LPUA can no longer find a different path from one of those have been found. The result is promising: to find most or all of the least cost paths between s and d , LPUA only needs run 2~3 times Dijkstra algorithm averagely. So LPUA has a very low computation complexity.

4. Performance Evaluation

We compare LPUA with 2EFMA to verify the validation of our algorithm. The experiment environment is as follows: network topology is randomly generated Waxman network graph with two weights, and each weight $w \sim \text{uniform}[1, 300]$.

Notice should be paid that as an on-command algorithm, LPUA has to make a tradeoff between computation cost and performance. Yong Cui has shown that when $b=7$, 2EFPA outperforms H_MCOF. So we only consider at what computation cost that LPUA can attain or outperform the performance of 2EFPA with $b=7$. If the termination conditions is when the largest length of the interval in the interval list less than $1/6$, the performance of LPUA is comparable with that of 2EFPA with $b=7$.

We used the same method of generating random constraints for requests in [11].

To attain the performance of 2EFPA, we set one of the termination conditions of LPUA to that when the length of the largest interval is less than $1/(b-1)$. Thus the two algorithms have the comparable performance.

Figure 3 shows the relative performance of LPUA. Y-coordinate is average SR, the ratio of the number of requests satisfied using heuristics and the total number of request generated. And X-coordinate is experiment ID. As shown in the Figure 3, LPUA and 2EFPA have the comparable performance.

Figure 4 shows the average cost of LPUA when it attains the performance in Figure 3. X-coordinate is to which degrees that 2EFPA care each weight coefficient. Y-coordinate is the average times (AT) LPUA need to run Dijkstra algorithm to attain or outperform the performance of 2EFPA with b .

As shown in this Figure 4, (1) LPUA only needs to run Dijkstra several times (2~2.5) to attain the performance of 2EFPA with $b=7, 10, 20$. This is mainly due to the following reasons: (1) The dynamic weight coefficient mechanism that makes LPUA reach to a fine granularity of weight coefficient quickly. (2) When the newly found path by LPUA dominates the constraint, LPUA ends successfully. And when the newly found path is dominated by the constraint, LPUA knows there is no a path that can satisfy the constraint, so it ends. As an on-command routing algorithm, LPUA is very efficient. In fact, LPUA can easily attain the performance of 2EFPA with a large b .

5. Conclusion

We propose an efficient limited path unicast QoS routing algorithm, LPUA, for the most common two-constrained QoS routing problems.

By combining two weights linearly into a single one, we run Dijkstra algorithm based on the single weight to return a least cost path between s and d . According to the non-dominated characteristic of the paths returned by Dijkstra based on different combinations of weight coefficients, we present LPUA. When a routing request comes, LPUA at most needs to compute a limited number of paths by running Dijkstra algorithm based on different weight coefficient combinations properly selected.

Extensive of simulations have show that (1) if there is a feasible path, then the probability of satisfying the request is very large; and (2) if the paths found by LPUA cannot meet the request, then the

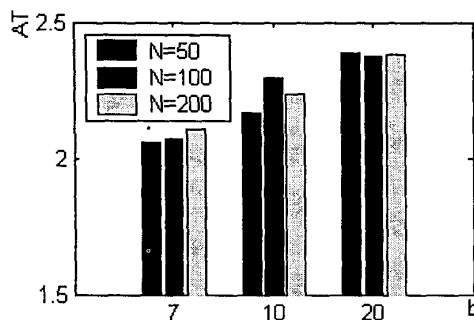


Figure 4. The average cost when LPUA attains the performance of 2EFPA with b

probability of that there is no feasible path between the source node and the destination node is very large too, so LPUA will refuse the request. The probability of misjudgment is small. Furthermore, LPUA implements a dynamic weight coefficient mechanism, which assures that LPUA can reach a fine granularity of weight coefficient quickly. A small granularity of weight coefficient means that LPUA can find more paths that can meet the routing request while the computation complexity is kept under what we can afford.

Routing information is inherently inaccurate [14]. The dominated areas of different least cost paths are largely overlapped, which compensates for the inaccurate routing information to some extent.

REFERENCES

- [1] YanXing Zheng, Jing Tian, Wenhua Dou, "GA for Vector Constraint Multicast Routing Problem", Chinese Journal of Computers, 2003, No.6, pp. 746-752
- [2] A.Orda, Routing with End-to-End QoS Guarantees in Broadband Networks, IEEE/ACM Transactions on Networking, Vol.7, No.3, pp. 365-374, 1999.
- [3] D.H. Lorenz, A. Orda, D.Raz and Y. Shavitt, Efficient QoS partition and Routing of Unicast and Multicast, Proceeding of IWQoS, pp. 75-83, June, 2000.D.S.
- [4] D.S Reeves and H. F. Salama, A distributed algorithm for delay-constrained unicast routing, IEEE/ACM Transactions on Networking, 8(2): 239-250, April 2000.
- [5] A. Juttner, B. Szviatovszki, I. Mecs and Z. Rajko, Lagrange relaxation based method for the QoS routing problem, in Proceedings of the INFOCOM 2001 Conference, volume 2, pages 859-868. IEEE, April 2001.
- [6] A. Iwata, R. Izmailov, D.-S. Lee, B. Sengupta, G. Ramamurthy and H. Suzuki, ATM Routing Algorithms with Multiple QoS Requirements for Multimedia Internetworking, IEICE Transactions and Communications E79-B, no.8, pp. 999-1006, 1996.
- [7] P. van, Mieghem, H.De. Neve and F.A. Kuipers, Hop-by-Hop Quality of Service Routing, Computer Network, vol. 37/3-4, pp. 407-423, October 2001.
- [8] H.D.e Neve, and P.Van Mieghem, "TAMCRA: A Tunable Accuracy Multiple Constraints Routing Algorithm", Computer Communications, Vol. 23, pp. 667-679, 2000
- [9] T. Korkmaz and M. Krunz, Multi-Constrained Optimal Path Selection, IEEE INFOCOM, 2001.
- [10] F.A. Kuipers, T. Korkmaz, M. Krunz and P. Van Mieghem, *An Overview of Constraint-Based Path Selection Algorithms for QoS Routing*, IEEE Communications Magazine, vol. 40, no. 12, December 2002.
- [11] Yong Cui, Ke Xu, Jiangping Wu, Precomputation for Multi-constrained QoS Routing in High-speed Networks, INFOCOM, 2003.
- [12] Xin Yuan, "Heuristic Algorithms for Multiconstrained Quality-of-service Routing", ACM/IEEE Transactions on Networking, Vol. 10, No.2, April 2002.
- [13] Yanxing Zheng, Jing Tian, ZhiFeng Liu, Wenhua Dou, "An Efficient Dynamic weight coefficient QoS Routing algorithm", International network conference(INC), 2004
- [14] YanXing Zheng, Wenhua Dou, Jing Tian, and Mingyan Xiao, An Overview of Research on QoS Routing. In Springer LNCS, 2834. pp. 387-397 2003.