Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Networks 47 (2005) 923–937
Computer Networks
www.elsevier.com/locate/comnet

Precomputation for intra-domain QoS routing ☆

Yong Cui *, Jianping Wu, Ke Xu

Department of Computer Science, Tsinghua University, Beijing 100084, PR China

Received 10 January 2003; received in revised form 26 March 2004; accepted 1 November 2004
Available online 20 November 2004

Responsible Editor: I. Nikolaidis

Abstract

Quality-of-service routing (QoSR), seeking to find a feasible path with multiple constraints, is an NP-complete problem. We propose a novel precomputation approach to multi-constrained intra-domain QoS routing (PMCP). It is assumed that a router maintains the link state information of the entire domain. PMCP cares each QoS weight to several degrees, and computes a number of QoS coefficients uniformly distributed in the multi-dimensional QoS metric space. Based on each coefficient, a linear QoS function is constructed to convert the multiple QoS metrics to a single QoS value. We then create a shortest path tree with respect to the QoS value by Dijkstra's algorithm. Finally, according to the multiple coefficients, different shortest path trees are calculated to compose the QoS routing table. We analyze linear QoS functions in the QoS metric space, and give a mathematical model to determine the feasibility of a QoS request in the space. After PMCP is introduced, we analyze its computational complexity and present a method of QoS routing table lookup. Extensive simulations evaluate the performance of the proposed algorithm and present a comparative study.

Keywords: QoS routing; Precomputation; Multiple constraints

1. Introduction

Internet multimedia applications emerge continuously in recent years. With the increase of multimedia information and application classes, Internet turns to be an integrated multimedia-transmission network from the traditional simple data-transmission network. However, the network layer of Internet cannot distinguish the classes or

☆ Supported by: (1) the National Natural Science Foundation of China (No. 60403035); (2) the National Major Basic Research Program of China (No. 2003CB314801).
* Corresponding author.
E-mail addresses: cy@csnet1.cs.tsinghua.edu.cn (Y. Cui), jianping@cernet.edu.cn (J. Wu), xuke@mail.tsinghua.edu.cn (K. Xu).

1389-1286/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2004.11.002

requests of applications, and only fairly provide network resources to all kinds of applications (e.g. bandwidth, buffer and CPU resources on routers). Consequently, Internet takes best-effort forwarding as the only one object. Therefore, such a service of best-effort forwarding cannot satisfy different users and multimedia applications with the evolution of Internet.

It becomes a challenging issue to provide different quality-of-services (QoS) for different applications in the Internet [1]. QoS routing (QoSR) seeks to find a feasible path that satisfies multiple QoS constraints requested by an application. Since there are usually a lot of paths with different characteristics from a given node to another in the network, QoSR is a potential solution to this problem [2]. QoS constraints can be divided into link constraints and path constraints. A link constraint of a path is the constraint of the bottleneck link in the path, such as the bandwidth constraint. It can be easily dealt with in a preprocessing step by pruning all links that do not satisfy the constraint and computing a path from the rest sub-graph. The path constraint is the restriction of each link along the path, such as delay. Since finding a feasible path with multiple path constraints has the NP-complete computational complexity [3,4], we will focus on path constraints in the paper.

Many heuristics have been proposed for the problem. However, these algorithms have some or all of the following limitations [2]: (1) high-computational complexity prevents their practical applications; (2) low routing performance causes that a feasible path sometimes cannot be found even when it does exist; (3) some algorithms only work for an environment with a specific network.

Furthermore, most of them use an on-line computation scheme, which calculates the feasible path when a QoS request arrives. Even if we do not consider the long delay of a request induced by the on-line calculation, the computation for each flow may cause an insufferable computational overload in the high-speed next-generation networks. Contrarily, the scheme of routing precomputation uses an off-line procedure to calculate the routing table, and when a request arrives at a router, the router looks up a feasible path in the table and then forwards the request. In most cases, a considerable reduction in the overall computational load could be achieved by precomputation, especially when the rate of QoS request arrivals is much higher than that of significant changes in the network state [5]. Moreover, precomputation provides the possibility of packet-based hop-by-hop distributed routing.

This paper proposes a novel approach, precomputation for multi-constrained path (PMCP). We assume that each intra-domain router s maintains a consistent copy of link states in the entire domain with k different QoS metrics. The algorithm cares each QoS weight to b degrees. It then computes B ($B = C_{b+k-2}^{k-1}$) coefficients that are distributed uniformly in the k-dimensional QoS weight space. A linear QoS function (LQF) is constructed for each coefficient to convert multiple QoS metrics to a single QoS value. Node s then uses Dijkstra's algorithm to calculate a shortest path tree rooted by s with respect to each LQF, and a part of QoS routing table is created based on the shortest path tree. At last, s combines the B parts of the routing table to form an entire intra-domain QoS routing table it maintains. For distributed routing, the QoS routing table only needs to save the next hop of each path in addition to the destination and the weights of each path. For source routing, the end-to-end path along the least QoS-value tree should be saved in the routing table. Therefore, when a QoS connection request arrives, it can be routed by looking up a feasible path in the routing table that satisfies the QoS constraints.

Experimental results show that PMCP can be easily implemented with high performance and high scalability. There are two major contributions in this paper. (1) We present a mathematical model to partition the k-dimensional QoS constraint space, so that the feasibility of a QoS request can (cannot) be determined by the continuous change of k-dimensional LQF. (2) PMCP is proposed to precompute the QoS routing table for the multi-constrained QoSR problem.

The rest of the paper is organized as follows. Related work is discussed in Section 2. An overview of the proposed PMCP is given in Section 3. We analyze LQF to give the theoretical basis in Section 4 and PMCP is described in Section 5. Section 6 shows the performance evaluation and

Section 7 presents a comparative study. Finally conclusions appear in Section 8.

2. Related work

Finding a feasible path that satisfies multiple constraints is a NP-complete problem, for which many heuristic algorithms have been proposed. An extensive survey on QoSR can be found in [2,6]. If some scheduling schemes [7,8] (e.g. weighted fair queuing) are used, the queuing delay, jitter, and loss can be formulated as a function of bandwidth. The original NP-complete QoSR problem is then reduced to the standard shortest path problem based on the dependencies among QoS parameters [9,10]. With this approach Orda did an extensive study [11]. However, this is not the case for propagation delay, which needs to be taken into account for QoS routing in high-speed networks [12]. Furthermore, such algorithms can only be applied in networks with specific scheduling schemes.

In order to increase the applicability, pseudo-polynomial-time and approximate algorithms have been proposed. For 2-constrained problems, Jaffe proposed a distributed algorithm with the complexity $O(n^5\varphi \log n\varphi)$, where φ is the upper bound of the QoS metric on a link [13]. Because its complexity depends on the values of metrics (e.g., the maximum link metric) in addition to the size of network, it is called a pseudo-polynomial-time algorithm. Approximate algorithms have been proposed for the DCLC problem [14,15]. For an arbitrary $\varepsilon > 0$, these algorithms can find a path in polynomial time. Not only is the delay constraint of the path satisfied, but also its cost is less than $(1 + \varepsilon)$ times the optimal cost. For example, Lorenz proposed the algorithm with computational complexity $O(nm \log n \log\log n + (nm/\varepsilon))$ [15]. The complexity of such algorithms increases heavily for improving the performance.

In order to decrease the complexity, heuristics have been proposed based on the convergence of multiple metrics. Jaffe used the linear function $g(p) = a_1 w_1(p) + a_2 w_2(p)$ to solve 2-constrained problems first [13]. As his conclusion, for a given constraint vector (c_1,c_2) of a QoS request, when $a_2/a_1 = \sqrt{c_1/c_2}$, the path found by Dijkstra's algorithm with minimizing $g(p)$ can be feasible with maximum probability. Neve proposed TAMCRA [16] and its reformation SAMCRA [17] for multi-constrained problems based on nonlinear function $g_\lambda(p) = \sum_{l=1}^{k} (w_1(p)/c_1)^\lambda$. Because the path with minimum $g_\lambda(p)$ cannot be found in polynomial time for the nonlinear characteristics, the algorithm uses the variant of Dijkstra's algorithm to find and store K undominated paths on each node with the complexity of $O(Kn \log(Kn) + K^3 km)$.

Korkmaz proposed H_MCOP for the multi-constrained optimal-path problem by marking labels reversely [18]. This algorithm marks each node by running Dijkstra's algorithm reversely with $g_1(p)$. When it then runs Dijkstra's algorithm forward with $g_{\lambda>1}(p)$, it considers both the labels marked reversely and the tree partly constructed forwardly. Although such a forward process cannot guarantee to find the path with minimum $g_{\lambda>1}(p)$, the algorithm achieves a good performance.

However, most of the proposed QoSR algorithms use an on-line scheme, which is difficult to layout in the high-speed next-generation networks. To improve the scalability, the off-line (precomputation) scheme seems to be a good choice [5]. Yuan presented a limited granularity heuristic and a limited path heuristic [19]. The former limits the metric of each link and the latter limits the size of the routing table directly. The latter has a lower computational complexity $O(n^3 m \log n)$ with the routing table size $O(n^2 \log n)$. Pointing to the multi-constrained optimal problem, Orda proposed a precomputation algorithm by mapping the cost to a discrete space [5]. Its complexity is $O(1/\varepsilon Hm \log C)$, where H is the longest hop number and C is the upper bound of the cost. Some other early algorithms have higher complexity [20]. These precomputation algorithms are based on distance vectors and use the extended Bellman-Ford algorithm. Therefore, they have some inherent problems, e.g. the count-to-infinity problem, inevitable routing loops and a large quantity of updating information that may overload the network.

There are three major differences between PMCP proposed in this paper and other similar algorithms. (1) From the viewpoint of objectives and functions, PMCP is to solve the general multi-constrained routing problem, while some other

algorithms require some specific scheduling schemes [10,11] or a limited set of QoS metrics [13,16]. (2) Viewed from the methods, PMCP is a precomputation algorithm based on network link states while some others are on-line algorithms [13,16–18] or precomputation ones based on distance vectors [19]. (3) PMCP uses multiple QoS functions that are independent of QoS requests, and their linear characteristics guarantee to find the shortest paths with respect to the functions easily. Some similar algorithms use only one QoS function [13,18]. They either seek for a particular linear QoS function for a specific QoS request [13], or emphasize how to find the shortest path with regard to a non-linear QoS function by heuristics [18].

3. Overview of PMCP

3.1. Problem formulation

A directed graph $G(V,E)$ presents a network. V is the node set and the element $v \in V$ is called a node representing a router in the network. E is the set of edges representing links that connect the routers. The element $e_{ij} \in E$ represents the edge $e = v_i \rightarrow v_j$ in G. In QoSR, each link has a group of independent metrics $(w_0(e), w_1(e), \ldots, w_{k-1}(e))$, which is also called QoS metric $w(e)$.

Definition 1 (Multi-constrained path). For a given graph $G(V,E)$, source node s, destination node t, $k \geqslant 2$ and a constraint vector $c = (c_0, c_1, \ldots, c_{k-1})$, the path p from s to t is called a multi-constrained path, if $w_l(p) \leqslant c_l$ for any $l = 0, 1, \ldots, k-1$. We write $w(p) \leqslant c$ in brief.

Note: $w(e)$ and c are both k-dimensional vectors. For a given QoS request and its constraint c, QoSR seeks to find a feasible path p satisfying $w(p) \leqslant c$ based on the network state information.

The path constraints can be divided into additive constraints (e.g. cost, delay) and multiplicative constraints (e.g. loss rate) [2], while multiplicative constraints can be transformed into additive constraints by logarithm. Therefore, we only consider additive constraints in the paper. That is to say, for a path $p = (e_0, e_1, \ldots, e_n)$ and $l = 0, 1, \ldots, k-1$, the metric $w_l(e_i) \in R^+$ satisfies the additive characteristic $w_l(p) = \sum_{i=0}^{n} w_l(e_i)$ for $i = 0, 1, \ldots, n$.

3.2. Linear QoS function

Dijkstra's Shortest Path Tree (SPT) algorithm has a low computational complexity [21]. However, related to multiple metrics simultaneously, the QoSR problem turns to be NP-complete complexity, and the original Dijkstra's algorithm cannot solve it directly. In this case, one feasible method is to convert the multiple metrics to a single value.

Definition 2 (Linear QoS function (LQF)). The LQF of link e is defined as the linear function

$$g_a(e) = \sum_{l=0}^{k-1} a_l w_l(e), \tag{1}$$

where the coefficient $a_l \in [0,1]$ is independent of e for $l = 0, 1, \ldots, k-1$, and satisfies $\sum_{l=0}^{k-1} a_l = 1$. The coefficient $a = (a_0, a_1, \ldots, a_{k-1})$ that satisfies the above conditions is called a QoS coefficient, and $g_a(e)$ is called the QoS value of e.

Definition 3 (Least QoS-value path). For a given graph G, a source-destination pair (s,t) and a coefficient a, the path p_a from s to t is called the least QoS-value path, if

$$g_a(p_a) = \min_{p(s,t) \in G} g_a(p(s,t)). \tag{2}$$

Based on LQF, we convert the original multi-constrained path problem to a least QoS-value path problem. Each coefficient of LQF represents the important degree of the corresponding metric element in computing the SPT.

Theorem 1. *For a given graph G and a coefficient a, Dijkstra's SPT algorithm with respect to $g_a(e)$ can create a least QoS-value tree T_a rooted by s. The path p along the tree T_a from s to an arbitrary node t is a least QoS-value path.*

Proof. The original Dijkstra's algorithm guarantees that a path from s to an arbitrary node t along the tree has the least cost. Because $g_a(e)$ is a linear function, which satisfies

$$g_a(e_1+e_2)=\sum_{l=0}^{k-1}a_l w_l(e_1+e_2)$$
$$=\sum_{l=0}^{k-1}a_l w_l(e_1)+\sum_{l=0}^{k-1}a_l w_l(e_2)$$
$$=g_a(e_1)+g_a(e_2),$$

we can calculate $g_a(e)$ for each link e first, and then run the Dijkstra's SPT algorithm with respect to $g_a(e)$. Thus, a least QoS-value tree can be created. Therefore, a path p along the least QoS-value tree T_a from s to t must be a least QoS-value path for coefficient a, namely $g_a(p) = \min_{p(s,t)\in G} g_a(p(s,t))$. □

Theorem 2. *A least QoS-value path is a dominating path.*

Proof. If path p is a least QoS-value path from s to t, then a coefficient a exists for $aw(p) \leqslant aw(p')$ for any path p' from s to t. Since the coefficient $a > 0$, then we have $w(p) \leqslant w(p')$. Therefore, p is a dominating path. □

Because there are usually a great many paths from a given source to a given destination in a large-scale network, path reduction must be applied to improve the scalability. However, it may heavily lower the routing performance to ignore some important paths. Being a dominating path, the least QoS-value path has some special characteristics. It seems to be a good method to compose the QoS routing table with some least QoS-value paths.

3.3. Example of PMCP

Fig. 1 shows an example of the proposed algorithm. In the network there are two QoS metrics, i.e. (cost, delay). Node s is now going to construct its QoS routing table by computing the least QoS-value trees with respect to LQF. It first takes $a = (0,1)$ to find the least delay tree as shown in Fig. 1b, where only the delay is considered. Then it uses $a = (0.5, 0.5)$ to construct another tree in Fig. 1c, where the cost and the delay are both considered, and in Fig. 1d it finds the least cost tree with $a = (1,0)$. Thus, s finds different "good" trees, along which different "good" paths may exist for any destination on the tree. All of these three trees compose the final QoS routing table maintained by s at last. When a QoS request arrives, s looks up a feasible path in the routing table and then forwards the request. In this example, s cares each QoS weight in three different degrees (i.e. 0, 0.5, 1), respectively.

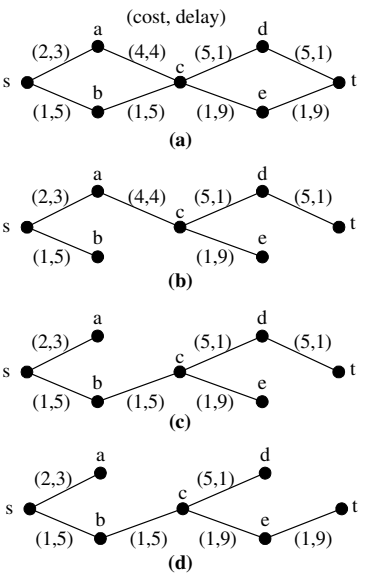


Fig. 1. An example of proposed algorithm. (a) Original network graph, (b) $a = (0,1)$, (c) $a = (0.5, 0.5)$, (d) $a = (1,0)$.

4. Linear QoS function analysis

If we take the QoS coefficient a as an independent variable, the question is changed: For a given G and source-destination pair (s,t), when QoS coefficient a reaches to all of the feasible values, what characteristics does the set $\{p_a|\forall a\}$ have? For example, how many elements does it have and how do they distribute? For convenience, we will first define the QoS metric space and then present theoretical basis of the proposed algorithm.

4.1. QoS metric space

Definition 4 (QoS metric space). $W^k = W_0 \times W_1 \times \ldots \times W_{k-1}$ is called the QoS metric space, if $w_l(p) \in W_l$ for any $p \in G$.

For the common condition as $w_l(e) \in R^+$, we take $W_l = R^+$, so $w_l(p) \in W_l$ for any path p. Thus, $w(p)$ is a point in the space W^k (i.e. $w(p) \in W^k$), and $\{w(p_a)|\forall a\}$ is a point set in W^k.

Theorem 3. *For a given G, a source-destination pair (s,t), a QoS coefficient a and taking $g_{opt} = \sum_{l=0}^{k-1} a_l w_l(p_a)$, $w(p)$ of an arbitrary path p from s to t must be on the upside of the hyperplane*

$$P = \left\{ w(p) | \sum_{l=0}^{k-1} a_l w_l(p) = g_{opt} \right\} \tag{3}$$

in the space W^k.

Proof. We use the reduction to absurdity. If there is a path p' with point $w(p')$ on the downside of hyperplane P, we have $g_{opt} = \sum_{l=0}^{k-1} a_l w_l(p_a) > \sum_{l=0}^{k-1} a_l w_l(p')$, which is contrary to $g(p_a) = \min_{p(s,t)\in G} g(p(s,t))$ in Definition 3. Thus, if there is any other path p' from s to t, $w(p')$ must be on the upside of the hyperplane P. □

For example, when $k = 2$ as shown in Fig. 2, for the given coefficient a, we use Dijkstra's algorithm with respect to g_a and create the least QoS-value path p_a from s to t. Drawing the perpendicular P of the coefficient vector a crossing the point $w(p_a)$, we get a partition of space W^2. The metric point $w(p')$ of any path p' from s to t must be on the upside of P as shown in Fig. 2a. We should note that, because of the discreteness of network topology graph, the corresponding p_a is changed discretely with the continuous change of coefficient

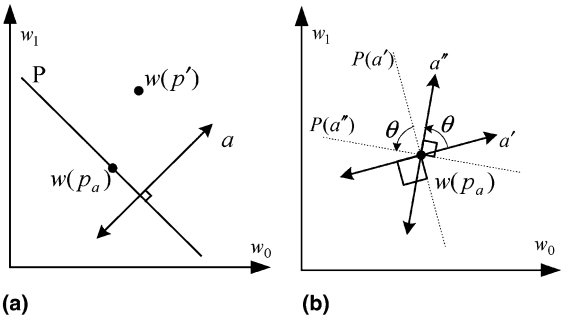


Fig. 2. Coefficient a and its hyperplane P. (a) $w(p')$ is on the upper side of P, (b) fixed p_a with different a.

a. Thus, we can see that the mapping $a \mapsto p_a$ is not an injection. For example in Fig. 2b, when the given coefficient changes from a' to a'', the least QoS-value path does not change, i.e. $p_b = p_a$ for any $b \in [a', a'']$ in. Therefore, the single path p_a corresponds to the continuous set $\{P(b)|b \in [a', a'']\}$ of hyperplanes.

4.2. Feasibility analysis for QoS constraint

We will give a partition of the QoS metric space W^k, including infeasible area M_{NOT}, feasible area M_{FEASIBLE} and unknown area M_{UNKNOWN} in this section. Accordingly, for a given QoS request with a constraint in W^k, we can judge its feasibility and find a feasible path for a feasible request.

Definition 5 (Infeasible area). The point set $M(a) = \{w|w \in W^k, w \text{ is in the lower side of } P(a)\}$ is called an infeasible area determined by a given coefficient a.

$$M_{\text{NOT}} = \bigcup_{\sum_{l=0}^{k-1} a_l = 1, a \geqslant 0} M(a) \tag{4}$$

is called the infeasible area.

Theorem 4. *For a constraint $c \in M_{NOT}$ of a request from s to t, there is NO feasible path p satisfying $w(p) \leqslant c$.*

Proof. According to the definition of M_{NOT}, M_{NOT} is the union of all $M(a)$ with the continuous change of coefficients a. For a given constraint $c \in M_{\text{NOT}}$, there must be a hyperplane $P(a)$ corresponding to the coefficient vector a, so that c is on the downside of $P(a)$. According to Theorem 3, any path p' from s to t must be on the upside of $P(a)$. Therefore, there is no feasible path p satisfying $w(p) \leqslant c$. □

Definition 6 (Available area). The point set

$$M_{\text{AVL}} = \overline{M_{\text{NOT}}} = \overline{\bigcup_{\sum_{l=0}^{k-1} a_l = 1, a \geqslant 0} M(a)} \tag{5}$$

is called the available area in space W^k, where $\overline{M_{\text{NOT}}} = W^k \setminus M_{\text{NOT}}$ is the complement of M_{NOT}. □

Fig. 3 shows the relation between M_{AVL} and M_{NOT}. For example, if there is only one least QoS-value path, Fig. 3a shows the infeasible area and the available area, where $M_{\mathrm{NOT}} = M_{\mathrm{NOT}}(a') \cup M_{\mathrm{NOT}}(a'')$. For the common sense where multiple least QoS-value paths exist, we should consider that (1) multiple coefficients a may map to a single point $w(p_a)$ and (2) a single coefficient may also map to multiple points. For example shown in Fig. 3b, when the coefficient changes continuously from a' to a'', the least QoS-value point keeps the same value, i.e. $w(p_a)$. It is the same case of the least QoS-value point $w(p_b)$ when the coefficient changes from b' to b''. However, the coefficient a'', equal to b', has two least QoS-value paths, i.e. p_a and p_b. In this case, a discrete change of least QoS-value paths occurs. Such a discrete change introduces a good characteristic: although the least QoS-value points are discrete, the infeasible area is continuous.

Theorem 5. *The available area M_{AVL} is a convex set, the metric point of an arbitrary least QoS-value path p_a is on the border of M_{AVL}, and there must be a coefficient a mapping to a vertex of M_{AVL} so that the point $w(p_a)$ is the vertex.*

Proof. We prove the above three parts in turn.

(1) To prove that M_{AVL} is a convex set (see [22] for the definition and the characteristics of a convex set): For a given coefficient a, the corresponding hyperplane divides the space W^k into two parts and either of them is a half-space. Thus, the available area $M_{\mathrm{NOT}}(a) = W^k \backslash M_{\mathrm{NOT}}(a)$ is a convex set for each give coefficient a. On the other hand, since $\overline{M_{\mathrm{AVL}}} = \overline{M_{\mathrm{NOT}}} = \bigcup_{|a|=1,a\geqslant 0} M_{\mathrm{NOT}}(a) = \bigcap_{|a|=1,a\geqslant 0} \overline{M_{\mathrm{NOT}}(a)}$ and the intersection of convex sets is still a convex set, M_{AVL} is a convex set.

(2) To prove that an arbitrary $w(p_a)$ is on the border of M_{AVL}: For an arbitrary $w(p_a)$, drawing the hyperplane $P(a)$ crossing the point $w(p_a)$ with a being the normal vector, M_{AVL} is on one side of $P(a)$ according to Theorem 3 and $P(a)$ is the border between $M(a)$ and $\overline{M(a)}$. Therefore, $w(p_a)$ is on the border of M_{AVL}.

(3) To prove that there must be a coefficient vector a mapping to a vertex of M_{AVL}: For any vertex $\forall x$ of M_{AVL}, because M_{AVL} is a convex set, there must be a hyperplane $P(a)$. Here, $P(a)$ crosses the point x and M_{AVL} is on one side of $P(a)$, where a is a normal vector of the hyperplane. Because M_{AVL} is an unlimited set, i.e. $(\infty,\infty,\ldots,\infty) \in M_{\mathrm{AVL}}$, M_{AVL} must be on the upside of $P(a)$. Therefore, for a given coefficient a, the point $w(p_a) = x$ must be able to be calculated. □

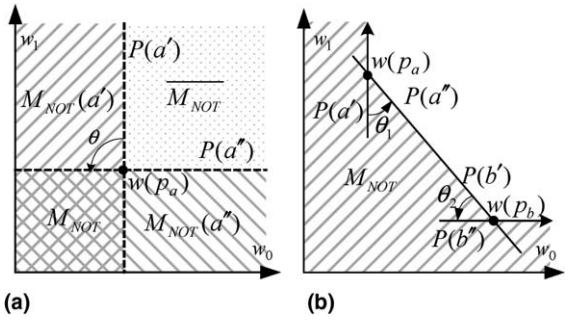


Fig. 3. Infeasible area M_{NOT} with least QoS-value paths. (a) $M_{\mathrm{NOT}} = M_{\mathrm{NOT}}(a') \cup M_{\mathrm{NOT}}(a'')$, (b) transition from p_a to p_b.

Fox example shown Fig. 4a, there are multiple least QoS-value paths with different QoS functions. Thus, the available area M_{AVL} is a convex set and each vertex of M_{AVL} is a least QoS-value path.

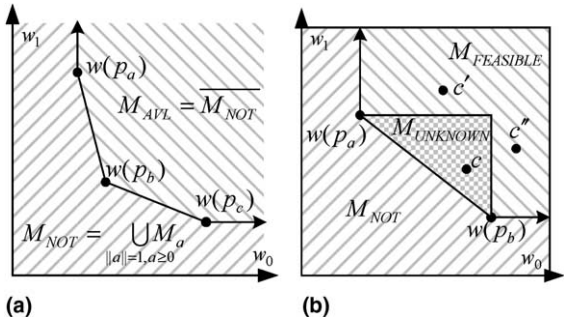


Fig. 4. Three partitions of W^k: infeasible, unknown and feasible areas. (a) The convexity of M_{AVL}, (b) feasible constraints: c' and c''.

We further divide M_{AVL} into two parts: the feasible area M_{FEASIBLE}, and the unknown area M_{UNKNOWN}.

Definition 7 (Feasible area M_{FEASIBLE}).

$$M_{\mathrm{FEASIBLE}} = \{w(p) | \exists a, w(p) \geqslant w(p_a)\} \tag{6}$$

is called the feasible area.

Theorem 6. *For an arbitrary constraint c of a QoS request, there must be a feasible path if* $c \in M_{FEASIBLE}$.

Proof. From the definition of M_{FEASIBLE}, $\exists a$ that satisfies $w(p_a) \leqslant c$, so p_a can be a feasible path for the constraint c. □

Definition 8 (Unknown area M_{UNKNOWN}).

$$M_{\mathrm{UNKNOWN}} = M_{\mathrm{AVL}} - M_{\mathrm{FEASIBLE}} \tag{7}$$

is called unknown area.

Fig. 4b shows an example of the three partitions in W^k. Path p_a is feasible for the constraint $c' \in M_{\mathrm{FEASIBLE}}$ because $w(p_a) < c'$, and p_b is a feasible path for $c'' \in M_{\mathrm{FEASIBLE}}$ because $w(p_a) < c''$. We do not know the existence of a feasible path for any constraint $c \in M_{\mathrm{UNKNOWN}}$. Thus, we divide the space W^k into three areas: an infeasible area M_{NOT}, an unknown area M_{UNKNOWN} and a feasible area M_{FEASIBLE}.

5. Algorithm description

5.1. The idea of PMCP

For a given QoS request with constraint c, there are three possibilities for the feasibility of the request according to the position of c in the space W^k. (1) For $c \in M_{\mathrm{FEASIBLE}}$, we know the feasibility and can find an element in the $\{p_a\}$ as the feasible path. (2) For $c \in M_{\mathrm{NOT}}$, we know that there is NO feasible path, so we refuse the request or start QoS negotiation. (3) For $c \in M_{\mathrm{UNKNOWN}}$, we do not know whether a feasible path exists. In the following performance evaluation section, simulations will show that the area M_{UNKNOWN} is small and most (95%) QoS requests will be in the other two areas. Furthermore, we will also demonstrate that a QoS request in M_{UNKNOWN} has a small opportunity to be feasible. Therefore, we take this area as the infeasible area without affecting the performance significantly, so we refuse this kind of QoS requests.

5.2. QoS coefficient distribution

A practical algorithm cannot change a continuously, so we discuss how to construct multiple QoS coefficient a. In order to make the discrete a independent of networks, we normalize the metrics of each link first. That is to say, the potential maximum metric, $\max_{e \in E} w_l(e)$, is a constant, which is independent of l. Then, we select b uniform numbers in $[0, 1]$, i.e.

$$D = \{0/(b-1), 1/(b-1), \ldots, 1\} \tag{8}$$

with totally b elements. Thus, we get the uniform coefficients

$$A\left\{ a | a \in D^k, \sum_{l=0}^{k-1} a_l = 1 \right\} \tag{9}$$

as the coefficients a in the subset $[0, 1]^k$ of QoS metric space W^k. At last, the node, carrying out PMCP, calculates the QoS routing table for all $a \in A$ based on the network link state it maintains.

Theorem 7. *The number of the elements in the QoS coefficient set* $A\{a | a \in D^k, \sum_{l=0}^{k-1} a_l = 1\}$ *is*

$$|A| = C_{b+k-2}^{k-1}. \tag{10}$$

Proof. There are b elements in set $D = \{0/(b-1), 1/(b-1), \ldots, 1\}$, and they are uniform in $[0, 1]$. We have $a \in D^k$, $\sum_{l=0}^{k-1} a_l = 1 = (b-1)/(b-1)$. Thus, if we take each $1/(b-1)$ as a ball, the set D can be considered as D={0 ball, 1 ball, 2 balls, …,$(b-1)$ balls}. The total number of the balls represented by the coefficient a is $(b-1)$. Additionally, the meaning of a_{l1} is different to that of a_{l2} for $l1 \neq l2$.

Therefore, we can equate $|A|$ with the number of the methods to put $(b-1)$ same balls into k different boxes, where each box can contain an arbitrary number of balls. According to the

combinatorics, there are $C_{b+k-2}^{k-1} = (b+k-2)!/(b-1)!(k-1)!$ different methods to put the balls [23], i.e. $|A| = C_{b+k-2}^{k-1}$. □

We denote B as the number of LQFs, i.e. $B = |A| = C_{b+k-2}^{k-1}$. For the k-constrained routing problem, node s first constructs B QoS coefficients. Then s computes the least QoS-value tree for each coefficient, respectively, and saves the path from s to each destination node t in the network along the tree to the QoS routing table. Thus, there are at most B different paths from s to each destination t. When a QoS request arrives at s, s only needs to look up a feasible path satisfying k constraints in the routing table.

5.3. Description of PMCP

We propose the precomputation algorithm, PMCP, for the k-constrained routing problem as shown in Fig. 5. G is the network graph with K metrics, s is the node running PMCP, and $(b-1)$ is the number of degrees to which each weight is cared. The algorithm includes two parts. (1) A number B of QoS coefficients $a = (a_0, a_1, \ldots, a_{k-1})$ are constructed according to the configuration of b (Line 1, 2, 8–11). (2) A part of QoS routing table is calculated with respect to each coefficient a (Line 3–7). This includes the following steps: (i) For the given G and each coefficient a, calculate the QoS-value $g_a(e)$ for each link (Line 3–4). (ii) Use Dijkstra's algorithm to compute a least QoS-value tree T_a rooted by node s regarding LQF g_a (Line 6). (iii) Save the path from s to each node along T_a to QoS routing table (Line 6–7).

```
PMCP (G, s, k, b)
1)    IF (k==K-1)
2)        a[k]=b-1
          // we have got the coefficient a[K]
3)        FOR EACH edge e IN G
4)            g_a(e) = Σ_{l=0}^{k-1} a_l w_l
5)        dijkstra (G,s)
6)        FOR EACH node t IN G
7)            store p_a(s,t)
8)    ELSE
9)        FOR(i=0; i<b; i++)
10)           a[k]=i
11)           PMCP(G, s, k+1, b-i)
```

Fig. 5. The proposed PMCP.

Because the linear function $g_a(e)$ satisfies the isotonicity, there are no routing loops in the routing table calculated by node s [24]. Therefore, in the source routing scheme, PMCP can avoid routing loops even when different nodes have the inconsistent copies of network state information. However, for distributed routing we need the consistence of the network state information in different nodes. If all nodes use the routing table entries generated with same $g_a(e)$ to forward a specific QoS request hop by hop, routing loops are avoided.

We analyze the computational complexity of PMCP to calculate QoS routing table. In a network graph G with k QoS metrics, the node number is $n = |V|$ and the edge number is $m = |E|$. Step (i) has the complexity of $O(m)$. Step (ii) is $O(n \log n + m)$ with the improved Dijkstra's algorithm and step (iii) is $O(n)$. The number of coefficients is $B = C_{b+k-2}^{k-1}$ (Theorem 7). As a result, including the recursive part, the overall computational complexity of PMCP is $O(C_{b+k-2}^{k-1}(m + n \log n + n))$, which is $B = C_{b+k-2}^{k-1}$ times the original Dijkstra's algorithm with a single metric. In a general situation, there are only a few kinds of path constraints, e.g. cost, delay, jitter, and loss rate. Therefore, k will not be very large and complexity of PMCP is acceptable.

Since the value $B = C_{b+k-2}^{k-1}$ is important to PMCP, we will further show the relation between B and the performance by extensive simulations, which show that PMCP performs well when B is small (e.g. when $k = 2$, we let $B = b = 7$).

5.4. QoS routing table lookup

We now analyze the computational complexity to look up a feasible path in the QoS routing table. As we analyzed above, there are totally B paths from a given source to any destination at most. Because the QoS routing table saves k metrics, a feasible path can be selected with the current packet classification technique in multiple dimensions in a constant computational complexity [25]. That is to say, it only needs to access the memory for

d times to find a route, where d is the reduction of ranges to prefix lookup.

Moreover, considering the staleness of the network state information and the routing table, we can select a path by maximizing the feasibility of the selected path from the routing table as the following method:

$$\left\{ p_{\text{selected}} \left| \max_{l=0}^{k-1} \frac{c_l}{w_l(p_{\text{selected}})} = \min_{i=0}^{B-1} \left(\max_{l=0}^{k-1} \frac{c_l}{w_l(p_i)} \right) \right. \right\}. \tag{11}$$

If multiple p_{selected} exist, a path may be selected randomly from $\{p_{\text{selected}}\}$.

We can first save all of the B paths to each destination together, and then use a traditional lookup method to address these B paths via the destination IP address of the request. The computational complexity of finding the selected feasible path in expression (11) is O(Bk). However, Routing table lookup is often implemented by hardware, especially in core routers. When a request with the constraint c arrives at a router, the current special hardware (e.g. TCAM [26]) can find these B paths via the destination IP address of the request with the computational complexity of O(1). For the simple comparison in expression (11), hardware can find the selected feasible path parallel in O(1) complexity with Bk pipelines.

PMCP can also be used to solve the multi-constrained optimal path routing problem with a proper objective function in routing table lookup procedure. Assuming that w_0 represents the cost and $w_1, w_2, \ldots, w_{k-1}$ represent $(k-1)$ different path constraints, we select the feasible optimal path from these B paths by minimizing the objective function

$$f(p_{\text{opt}}) = \min(w_0(p_i)), \tag{12}$$

where $w_l(p_i) \leqslant c_l$ for $l = 1, 2, \ldots, k-1$. The computational complexity of the procedure is O(Bk). Similar to the above method, using special hardware can also achieve O(1) complexity with Bk pipelines.

6. Performance evaluation

The routing performance of QoSR algorithms is often evaluated by two methods. (1) Competitive ratio, which indicates how well a heuristic algorithm performs, is defined as the ratio of the number of requests satisfied by using a heuristic algorithm and the number of requests satisfied by using an exhaustive algorithm. (2) Success ratio (SR) is defined as the ratio of the number of requests satisfied by using a heuristic algorithm and the total number of requests generated.

The difference between the two methods is the feasibility of the requests being the denominator in theory. Both have shortcomings because the evaluation depends heavily on the generated constraints of the requests, e.g. the distribution of constraints. For a large-scale network, it is difficult to judge the theoretical feasibility of a request, so SR is used widely in most cases. Because different distributions of the requests are used in different papers, the absolute value of SR are useless, while only the comparison with the same network makes sense. Therefore, in this section we present the proportion of the unknown-area, which is independent of QoS requests, to show the performance evaluation of PMCP.

Recalling the three partitions of the QoS metric space W^k by PMCP, if the size of area M_{NOT} is large, we cannot say that PMCP performs badly because the state of the network may cause the infeasibility for too strict requests. Similarly, a large area of M_{FEASIBLE} neither proves that PMCP performs well, since the network may have a good path p with $w(p) \approx 0$. Because PMCP cannot determine the feasibility of a request with the constraint in M_{UNKNOWN}, the size of M_{UNKNOWN} represents the performance of PMCP. The smaller M_{UNKNOWN} area is, the better PMCP performs.

Therefore, we take M_{UNKNOWN} in Fig. 3b as the inefficiency of PMCP and analyze the proportion of this area to the whole area. M_{UNKNOWN} is an triangle area in theory by changing coefficient a continuously, but for the limited number of discrete a, M_{UNKNOWN} extends to M'_{UNKNOWN} at most as shown in Fig. 6a because we cannot guarantee the nonexistence of least QoS-value paths in triangle A and B. M'_{UNKNOWN} is a limited space while M_{NOT} and M_{FEASIBLE} are unlimited spaces. For a given (s, t) pair, we use Dijkstra's algorithm to construct the shortest path p_l with respect to w_l. The unknown-area proportion is defined as

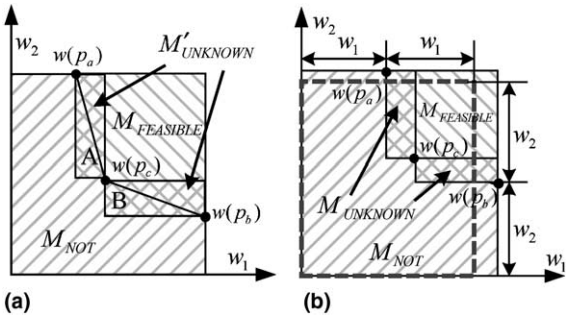


Fig. 6. Unknown area proportion. (a) Increase of unknown area, (b) the area we consider.

$$Pr = M'_{\mathrm{UNKNOWN}}/(M_{\mathrm{NOT}} + M_{\mathrm{FEASIBLE}} + M'_{\mathrm{UNKNOWN}}) \tag{13}$$

in the subset $\{x|0 \leqslant x_l \leqslant 2w_l(p_l),\ x \in W^k,\ l = 0,1,\ldots,k-1\}$ shown as the rectangle enclosed by the gray dashed line in Fig. 6b.

In each group of these experiments with a node number N being 50, 100, 200 and 500 respectively, we generate 10 pure random network graphs [27,28] with k metrics for each link, where $w_l(e) \sim$ uniform [1,1000] for $l = 1,2,\ldots,k$, and $w_l(e)$ have no correlation for different e or l. In each graph, we select source-destination node pair (s,t) 100 times (a particular node can be selected more than once), where we guarantee that the minimum hop is not less than two. Each source node s use our PMCP to calculate least QoS-value tree for $B = C_{b+k-2}^{k-1}$ times with B different QoS coefficients a.

Fig. 7 shows the average unknown-area proportion Pr with the 95% confidence interval for 10 random graphs. The X-axis represents the number of the degrees to care each weight and the Y-axis is Pr. As shown in this figure, (1) the proportion of unknown area is small; (2) Pr decreases rapidly to a constant value when b increases. This shows that in practice, to ensure high performance, we only need a few uniform coefficients a to find enough paths of different characteristics, e.g. when $k = 2$, $B = b = 7$. With larger b, most of the newly found paths are reduplications, which cannot decrease Pr furthermore. This is consistent with the conclusion in [19]. (3) In large-scale networks, Pr decreases when the number of metrics k increases, and the larger the k, the smaller the change that Pr decreases with b increasing. This shows that in multiple dimensions, when b is small, we use a number $(B = C_{b+k-2}^{k-1})$ of coefficients. Therefore, PMCP performs well with a small b in multiple dimensions. (4) With the increase of node number

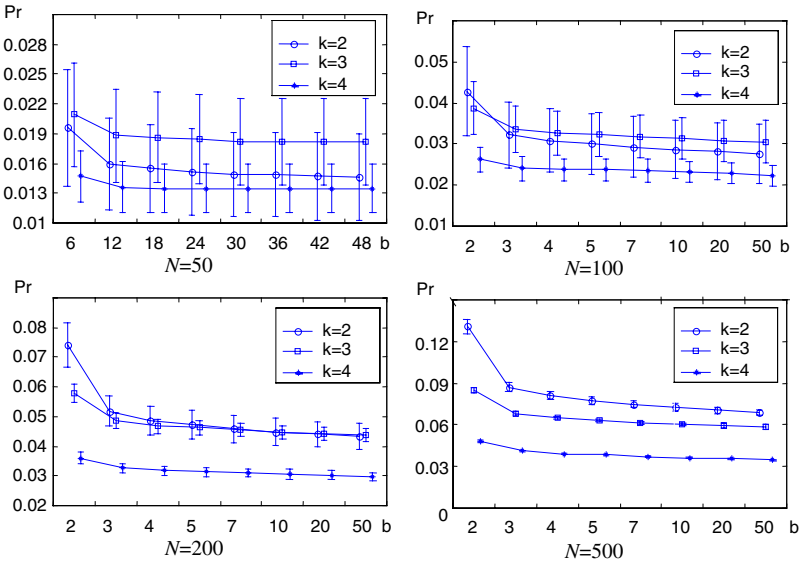


Fig. 7. The performance evaluation of proposed algorithm.

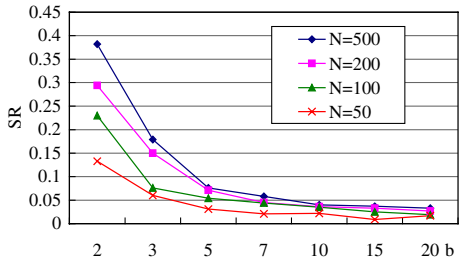


Fig. 8. The infeasibility of requests in the unknown area.

N, b plays a more significant role to the performance. The reason is that in larger networks, more paths are available between a particular (s,t) pair and the possibility to optimize a particular metric increases.

We have demonstrated that the unknown-area proportion Pr is small, and now we demonstrate that the feasibility for a request $c \in M_{\text{UNKNOWN}}$ is also small. First, we generate some constraints within the unknown area in Fig. 6 randomly. Then, we use H_MCOP [18] to seek feasible paths for each request. Fig. 8 shows the SR with $k = 2$, i.e. 2-constrained routing. When $b = 7$, the success ratio is less than 5% so that most requests may be inherently infeasible.

Having established that (1) the unknown-area proportion is small, and (2) a request within the unknown area has a low feasibility, PMCP can refuse the requests within the unknown area with a small probability of misjudgment (refuse feasible requests). As a result, PMCP performs well. Since the probability of misjudgment is small, the misjudgment is no longer the major factor that decreases the performance. Instead, the inherent staleness of network-state information based on which QoSR operates may be the major factor in practice [29].

7. Comparative study

Having showed the performance evaluation of PMCP, we then give a comparative study in this section. From the related work in Section 2, it is seen that there are only a few precomputation algorithms for QoSR at present. Some of them tend to have the prohibitive computational complexity or low performance, and some are based on distance vectors, so they are not fit for large-scale networks. In order to show the performance of PMCP, we compare PMCP with H_MCOP [18], which is also based on Dijkstra's algorithm.

7.1. Performance comparison with random constraints

We generate the constraints of requests to compare the two algorithms by the method in [18]. For a given (s,t) pair, we use Dijkstra's algorithm to calculate the shortest path p_l with respect to w_l, respectively. We then generate the constraints randomly for each (s,t) pair: $c_{l+1} \sim$ uniform $[0.8w_{l+1}(p_l), 1.2w_{l+1}(p_l)]$. The shadowed area in Fig. 9 shows the constraints generated in two dimensions.

Fig. 10 shows the success ratio in 2-constrained routing. When $b = 7$, the SR of PMCP is higher than that of H_MCOP. Table 1 shows the cases

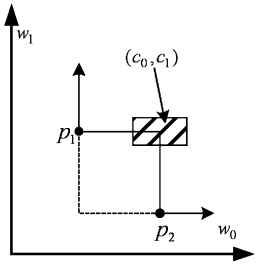


Fig. 9. Random constraints.

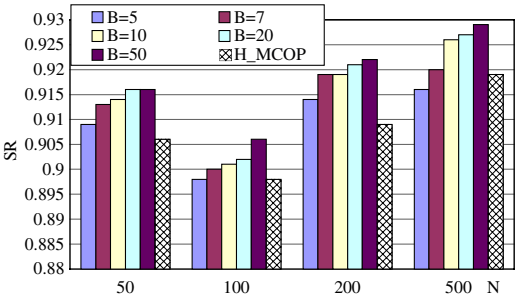


Fig. 10. Performance comparison with random constraints (two constraints).

Table 1
Performance comparison with random constraints (multiple constraints)

SR(%)		$k = 2$	$k = 3$	$k = 4$	$k = 5$
$N = 50$	$b = 3$	91.8	79.7	64.4	55.7
	$b = 7$	92.8	80.5	64.7	55.8
	H_MCOP	92.5	80.0	64.2	55.6
$N = 100$	$b = 3$	91.0	73.0	61.8	51.6
	$b = 7$	92.2	73.7	62.7	52.3
	H_MCOP	91.9	73.5	62.0	50.8
$N = 200$	$b = 3$	88.2	73.4	59.4	47.8
	$b = 7$	90.0	75.0	60.9	49.9
	H_MCOP	89.9	73.8	59.9	48.2

in multiple dimensions, i.e. k-constrained routing. With larger metric number k, the performance of both algorithms decreases rapidly since more requests generated by this random method are inherently infeasible. However, PMCP performs better than H_MCOP when b is small with a relatively large k (e.g. when $k = 5$, we choose $b = 3$). These results further confirm the conclusion of Section 6.

7.2. Performance comparison with simulated constraints

Because the routing performance depends on the distribution that the generated QoS constraints obey, experimental results in different papers cannot be compared directly with others. However, the Internet does not have a typical topology or traffic model [30], and we are even more short of knowledge about the constraints of the upcoming QoS applications. Therefore, it is difficult to give a reasonable model and a distribution of the constraints. Nevertheless, we know that most QoS applications care different weights to different degrees. For example, file transfer applications may care the loss rate to a much higher degree than delay since a packet loss cuts down the transmission speed heavily. On the other hand, multimedia applications may take delay as the most important parameter. The distribution of constraints in Fig. 9 does not consider this aspect, and it can generate constraints only round the middle between the w_0 axis and w_1 axis. For example, it cannot generate a constraint, whose w_0 is as important as three times w_1.

We use the method of weight ratio simulation to generate the constraint for a given pair (s, t). First, we assume that each QoS application has a coefficient a. The normalized $a_l/(a_0 + \cdots + a_{k-1})$ presents the degree, to which this application cares the weight w_l. Based on this assumption, we use the LQF g_a (Definition 2) to construct a constraint for a QoS request. For a given (s, t), we use Dijkstra's algorithm to calculate the least QoS-value path $p_a(s, t)$ and take its metric $w(p_a(s, t))$ as the QoS constraint of (s, t), i.e. $c(s, t) = w(p_a(s, t))$. Because a request with the simulated constraint must be feasible (e.g. $p_a(s, t)$ is a feasible path), SR, showing the absolute performance, is equal to the competitive ratio.

We simulate the network graphs and (s, t) pairs as we did in the experiments of Fig. 7, and take $a_l \sim \text{uniform}(0, 1)$. Fig. 11 shows the SR of these QoS requests we simulate. The experiment shows that PMCP overmatches H_MCOP, and PMCP has a good scalability since it is insensitive not only to the network scale, but also to the constraint number k.

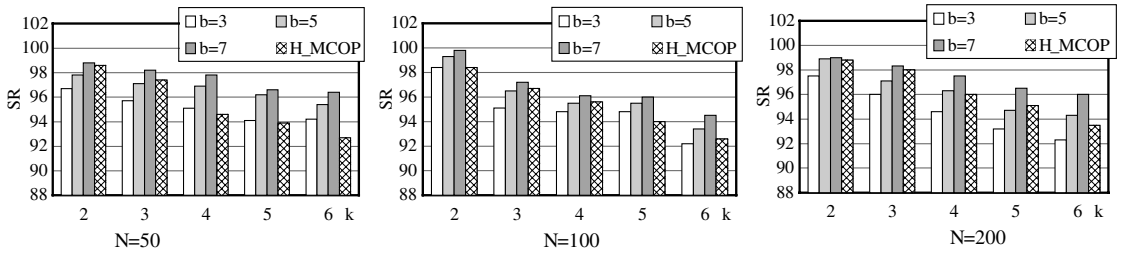


Fig. 11. Performance comparison with simulated constraints.

7.3. Computational complexity and running time comparison

On the aspect of computational complexity, H_MCOP is $O(km\log kn + n\log n + (k^2+1)m)$ with k-shortest path algorithm [31], while PMCP is $O(B(m + n\log n + n))$. For given $B = 7$ when $k = 2$, PMCP is better than H_MCOP.

On the running time in practical experiments with 500-node graphs, H_MCOP spends 15.3 ms in finding one path for a specific request averagely. PMCP only spends a comparable longer time as 51.8 ms in calculating the entire routing table to all destinations in the network, which can satisfy following different QoS requests. For example, if there are $(N-1)$ requests on a source node to the other $(N-1)$ nodes in the network, the running time of PMCP keeps fixed while that of H_MCOP increases $(N-1)$ times.

8. Conclusion

The multi-constrained routing problem has an NP-complete complexity. We propose a novel intra-domain precomputation algorithm PMCP based on linear QoS functions. With this algorithm, a router constructs a number (B) of uniform coefficients to construct B linear QoS functions. It then calculates B least QoS-value trees to compose the QoS routing table with computational complexity $O(B(m + n\log n + n))$. The size of the QoS routing table is less than or equal to B times that of the current routing table with a single cost. When a request arrives at the router, the router looks up a feasible path in the routing table and forwards the request accordingly.

In this paper, we analyze the linear QoS functions and partition the k-dimensional QoS metric space into three parts: an infeasible area, an unknown area and a feasible area. Based on the partition, a mathematical model is introduced to determine the feasibility of a request judged by the continuous change of k-dimensional LQF. After the algorithm is presented, we analyze the method to select the optimal feasible path from the routing table. An objective function of routing table lookup, which can be implemented in O(1) computational complexity by special hardware, is introduced to extend PMCP to solve the multi-constrained optimal-cost problem. The performance of PMCP is evaluated with the unknown-area proportion. Additionally, we show its comparative performance with the constraints of QoS requests generated by both the random method and metric-ratio simulation, respectively.

Since PMCP has a low computational complexity and uses precomputation scheme, it has a good scalability in the number of QoS constraints, the network scale and the packet arrival speed in next-generation networks. Extensive simulation results further confirm its scalability and the performance. Furthermore, PMCP is consistent with the routing architecture in the current Internet. In order to run PMCP, a router only needs to replace the traditional cost by the QoS value in SPT calculation.

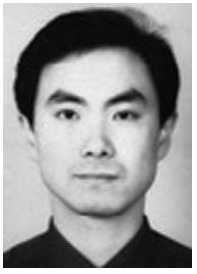
Acknowledgement

The authors would like to thank the editor of Computer Networks and the anonymous reviewers for their valuable comments.

References

[1] X. Xiao, L.M. Ni, Internet QoS: a big picture, IEEE Network 13 (2) (1999) 8–18.
[2] Y. Cui, J.P. Wu, K. Xu, et al., Research on internetwork QoS routing algorithms: a survey, Chinese Journal of Software 13 (11) (2002) 2065–2076.
[3] M.S. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-completeness, W.H. Freeman, New York, 1979.
[4] Z. Wang, J. Crowcroft, Quality-of-service routing for supporting multimedia applications, IEEE Journal on Selected Areas in Communications 14 (7) (1996) 1228–1234.
[5] A. Orda, A. Sprintson, QoS routing: the precomputation perspective, IEEE INFOCOM'00, vol. 1, 2000, pp. 128–136.
[6] S. Chen, K. Nahrstedt, An overview of quality-of-service routing for next-generation high-speed networks: problems and solutions, IEEE Network 12 (6) (1998) 64–79.
[7] J. Bennett, H. Zhang, Hierarchical packet fair queueing algorithms, ACM SIGCOMM'96, August 1996.
[8] L. Zhang, Virtual clock: A new traffic control algorithm for packet switching networks, ACM SIGCOMM'90, September 1990, pp. 19–29.

[9] Q. Ma, P. Steenkiste, Quality-of-service routing with performance guarantees. 4th International IFIP Workshop on Quality of Service, May 1997.
[10] C. Pornavalai, G. Chakraborty, N. Shiratori, QoS based routing algorithm in integrated services packet networks, IEEE ICNP'97, Atlanta, GA, 1997.
[11] A. Orda, Routing with end-to-end QoS guarantees in broadband networks, IEEE/ACM Transactions on Networking 7 (3) (1999) 365–374.
[12] H.F. Salama, D.S. Reeves, Y. Viniotis, A distributed algorithm for delay-constrained unicast routing, IEEE INFOCOM'97, Japan, April 1997.
[13] J.M. Jaffe, Algorithms for finding paths with multiple constraints, IEEE Networks 14 (1984) 95–116.
[14] D. Raz, Y. Shavitt, Optimal partition of QoS requirements with discrete cost functions, IEEE INFOCOM'00, vol. 2, 2000, pp. 613–622.
[15] H. Lorenz, A. Orda, D. Raz, Y. Shavitt, Efficient QoS partition and routing of unicast and multicast, IWQoS 2000, June 2000.
[16] H. de Neve, P. Van Mieghem, A multiple quality of service routing algorithm for PNNI, IEEE ATM Workshop Proceedings, 1998.
[17] P. Van Mieghem, H. de Neve, F. Kuipers, Hop-by-hop quality of service routing, Computer Netwroks 37 (2001) 407–423.
[18] T. Korkmaz, M. Krunz, Multi-constrained optimal path selection, IEEE INFOCOM'01, vol. 2, 2001, pp. 834–843.
[19] X. Yuan, X. Liu, Heuristic algorithms for multi-constrained quality of service routing, IEEE INFOCOM'01, vol. 2, 2001, pp. 844–853.
[20] A. Shaikh, J. Rexford, K.Shin, Efficient precomputation of quality-of- service routes, in: Proceedings of Workshop on Network and Operating Systems Support for Audio and Video (NOSSDAV'98), Cambridge, England, July 1998.
[21] E. Dijkstra, A note on two problems in connetion with graphs, Numerische Mathematik 1 (1959) 269–271.
[22] E. Kreyszig, Introductory Functional Analysis with Applications, Wiley, New York, 1978.
[23] R.A. Brualdi, Introductory Combinatorics, third ed., Prentice Hall, Upper Saddle River, NJ, 1999.
[24] J.L. Sobrinho, Algebra and algorithms for QoS path computation and hop-by-hop routing in the internet, IEEE INFOCOM'01, vol. 2, 2001, pp. 727–735.
[25] P. Gupta, N. Mckeown, Algorithms for packet classification, IEEE Network 15 (2) (2001) 24–32.
[26] M. Kobayashi, T. Murase, A. Kuriyama, A longest prefix match search engine for multi-gigabit IP, IEEE ICC'00 (2000).
[27] E.W. Zegura, K.L. Calvert, S. Bhattacharjee, How to model an internetwork, IEEE INFOCOM'96, vol. 2, 1996, pp. 594–602.
[28] E.W. Zegura, K.L. Calvert, M.J. Donahoo, A quantitative comparison of graph-based models for Internet topology, IEEE/ACM Transactions on Networking 5 (6) (1997) 770–783.
[29] A. Shaikh, J. Rexford, K.G. Shin, Evaluating the impact of stale link state on quality-of-service routing, IEEE/ACM Transactions on Networking 9 (2) (2001) 162–176.
[30] S. Floyd, V. Paxson, Difficulties in simulating the internet, IEEE/ACM Transactions on Networking 9 (4) (2001) 392–403.
[31] E.I. Chong, S.R. Sanjeev Rao Maddila, S.T. Morley, On finding single-source single-destination shortest paths, in the Seventh International Conference on Computing and Information (ICCI'95), July 1995, pp. 40–47.



Yong Cui, male, born in 1976, received the B.S., M.S. and Ph.D. degrees in computer science from Tsinghua University, P. R. China, in 1999, 2001 and 2004 respectively. He is now an assistant professor in the Department of Computer Science of Tsinghua University. During his Ph.D. study, he published 20 technical papers in journals and international conferences. He has also applied for several patents in China. His major research interests include computer network architecture, distributed routing protocols, QoS routing and core routers. He is an IEEE member.

Jianping Wu, male, born in 1953. He received master and doctor degrees in computer science from Tsinghua University. He is a full professor in the Department of Computer Science, Tsinghua University. In the research areas of the network architecture, high performance routing and switching, protocol testing and formal methods, he has published more than 200 technical papers in the academic journals and proceedings of international conferences.

Ke Xu, male, born in Jiangsu, P.R.China, in 1974. He received the B.S., M.S. and Ph.D. degrees in computer science from Tsinghua University, China in 1996, 1998 and 2001 respectively. Currently he is an Assistant Professor in the department of computer science of Tsinghua University. His researching interests include high-speed networks, switch and router architecture, QoS routing and congestion control. He is a member of IEEE and IEEE Communication Society.