Heuristic Algorithms for Multi–Constrained Quality of Service Routing

Xin Yuan Xingming Liu
Department of Computer Science, Florida State University, Tallahassee, FL 32306.
{xyuan, liu}@cs.fsu.edu

Abstract— **Multi–constrained Quality of Service (QoS) routing finds a route in the network that satisfies multiple independent quality of service constraints. This problem is NP–hard and a number of heuristic algorithms have been proposed to solve the problem. This paper studies two heuristics, the limited granularity heuristic and the limited path heuristic, for solving general k–constrained problems. Analytical and simulation studies are conducted to compare the time/space requirements of the heuristics and the effectiveness of the heuristics in finding the paths that satisfy the QoS constraints. We prove analytically that for an N nodes and E edges network with k (a small constant) independent QoS constraints, the limited granularity heuristic must maintain a table of size $O(|N|^{k-1})$ in each node to be effective, which results in a time complexity of $O(|N|^k|E|)$. We also prove that the limited path heuristic can achieve very high performance by maintaining $O(|N|^2 lg(|N|))$ entries in each node, which indicates that the performance of the limited path heuristic is not sensitive to the number of constraints. We conclude that although both the limited granularity heuristic and the limited path heuristic can efficiently solve 2–constrained QoS routing problems, the limited path heuristic is superior to the limited granularity heuristic in solving k–constrained QoS routing problems when $k > 3$. Our simulation study further confirms this conclusion.**

Keywords— **Quality of service routing, multi–constrained, limited granularity heuristic, limited path heuristic**

I. Introduction

The Quality of Service (QoS) requirement of a point–to–point connection is typically specified as a set of constraints, which can be *link constraints* or *path constraints* [2]. A link constraint, such as the bandwidth constraint, specifies the restriction on the use of links. For example, the bandwidth constraint requires that each link along the path must be able to support certain bandwidth. A path constraint, such as the delay constraint, specifies the end–to–end QoS requirement for the entire path. For example, the delay constraint requires that the aggregate delay of all links along the path must be less than the delay requirement.

Multi-constrained QoS routing finds a path that satisfies multiple independent *path* constraints. One example is the delay-cost-constrained routing, i.e., finding a route in the network with bounded end–to–end delay and bounded end–to–end cost. We will use the notion *k–constrained routing* to refer to multi–constrained QoS routing problems with exactly k path constraints. The delay-cost-constrained routing is an example of a 2–constrained routing problem. Multi–constrained QoS routing is known to be NP–hard[4], [8]. Previous work [1], [9], [10] has focused on developing heuristic algorithms to solve 2-constrained problems. The general k–constrained routing problem receives little attention. In practice, however, effective heuristics to solve general k–constrained QoS routing problems, such as the delay–jitter–cost–constrained problem, are needed. Whether (and how) the existing heuristics that effectively solve 2–constrained problems can be extended to handle general k–constrained problems is unclear. The performance of the heuristics in solving general k–constrained problems also needs further investigation.

In this paper, we study two heuristics, the limited path heuristic and the limited granularity heuristic, to solve general k–constrained QoS routing problems. The two heuristics were proposed in [10] to solve 2–constrained QoS routing problems using variations of the extended Bellman–Ford shortest path algorithm [9]. We extend the heuristics to deal with k–constrained problems, investigate the performance of the heuristics in solving k–constrained problems, and identify the conditions for the heuristics to be effective. We prove analytically that for an N nodes and E edges network with k independent path constraints (k is a small constant), the limited granularity heuristic must maintain a table of size $O(|N|^{k-1})$ in each node to achieve high probability of finding a path that satisfies the QoS constraints when such a path exists. By maintaining a table of size $O(|N|^{k-1})$, the time complexity of the limited granularity heuristic is $O(|N|^k|E|)$. Our analysis also shows that the performance of the limited path heuristic is rather insensitive to k and that the limited path heuristic can achieve very high performance by maintaining $O(|N|^2 lg(|N|))$ entries in each node. These results indicate that although both heuristics can efficiently solve 2–constrained routing problems [10] and solve k–constrained routing problems with very high probability in polynomial time, the limited granularity heuristic is inefficient when $k > 3$ since the time/space requirement of the limited granularity heuristic increases drastically when k increases. We conclude that the limited path heuristic is more effective than the limited granularity heuristic in solving k–constrained QoS routing problems when $k > 3$. Our simulation study further confirms this conclusion.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 describes the multi-constrained QoS routing problem and introduces the extended Bellman–Ford algorithm that can solve this problem. Section 4 studies the limited granularity heuristic for k–constrained problems. Section 5 analyzes the limited path heuristic. Section 6 presents the simulation study. Section 7 concludes the paper.

II. Related Work

Much work has been done in QoS routing recently, an extensive survey can be found in [2]. Among the proposed QoS routing schemes, the ones that deal with multi-constrained QoS

This project was supported in part by NSF grants: CCR-9904943 and CCR-0073482

routing are more related to the work in this paper. In [7], a distributed algorithm is proposed to find paths that satisfy the end–to–end delay constraint while minimizing the cost. Although this algorithm considers two path constraints, it does not solve the 2–constrained problem because the cost metric is not bounded. Ma [5] showed that when the weighted fair queuing algorithm is used, the metrics of delay, delay–jitter and buffer space are not independent and all of them become functions of the bandwidth. Orda [6] proposed the quantization of QoS metrics for efficient QoS routing in networks with a rate–based scheduler at each router. Although the idea of quantization of QoS metrics is similar to the limited granularity heuristic, the technique was proposed to improve the performance of a polynomial time QoS routing algorithm that solves the bandwidth–delay bound problem. Jaffe [4] proposed a distributed algorithm that solves 2–constrained problems with a time complexity of $O(|N|^5 blog(|N|b))$, where b is the largest number of the weights. This algorithm is pseudo-polynomial in that the execution time depends on the value of the weights (not just the size of the network). Widyono [9] proposed an algorithm that performs exhaustive search on the QoS paths in exponential time. Chen [1] proposed a heuristic algorithm that effectively solves 2–constrained problems. Yuan [10] studied the limited granularity heuristic and the limited path heuristic for 2–constrained problems. Our paper differs from the previous work in that it studies heuristic algorithms that efficiently solve the general k–constrained QoS routing problem. Some of the results for 2–constrained QoS routing [1], [10] are special cases of the results in this paper. To the best of our knowledge, this is the first paper that reports the study on the general k–constrained QoS routing problem.

III. BACKGROUND

A. Assumptions and notations

The network is modeled as a directed graph $G(N, E)$, where N is the set of nodes representing routers and E is the set of edges representing links that connect the routers. Each edge $e = u \rightarrow v$ is associated with k independent weights, $w_1(e)$, $w_2(e)$, ..., $w_k(e)$, where $w_l(e)$ is a positive real number ($w_l(e) \in R^+$ and $w_l(e) > 0$) for all $1 \leq l \leq k$. The notation $w(e) = w(u \rightarrow v) = (w_1(e), w_2(e), ..., w_k(e))$ is used to represent the weights of a link. It is assumed that all the constraints are path constraints and that the weight functions are additive [8], that is, the weight of a path is equal to the summation of the weights of all edges on the path. Thus, for a path $p = v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow ... \rightarrow v_n$, $w_l(p) = \sum_{i=1}^{n} w_l(v_{i-1} \rightarrow v_i)$. Notation $w(p) \leq w(q)$ denotes $w_l(p) \leq w_l(q)$ for all $1 \leq l \leq k$. Other comparative operators $<, =, >, \geq$ and arithmetic operators $+, -$ on the weight vectors are defined similarly. Let a path $p = v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow ... \rightarrow v_n$ and a link $e = v_n \rightarrow v_{n+1}$. The notation $p + e$ or $p + v_n \rightarrow v_{n+1}$ denotes the path $v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow ... \rightarrow v_n \rightarrow v_{n+1}$. This paper considers centralized algorithms and assumes that the global network state information is known.

Given a set S, the notation $|S|$ denotes the size of the set S. We will use the following notations: binary logarithm function $lg(n) = log_2(n)$, natural logarithm function $ln(n) = log_e(n)$, exponentiation of the logarithm function $lg^k(n) = (lg(n))^k$, and factorial function $n! = n * (n-1) * ... * 1$. We define $0! = 1$.

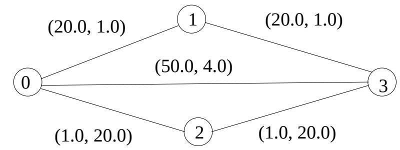


Fig. 1. Optimal QoS paths

B. Multi–Constrained QoS Routing

`Definition 1`: Given a directed graph $G(N, E)$, a source node src, a destination dst, $k \geq 2$ weight functions $w_1 : E \rightarrow R^+$, $w_2 : E \rightarrow R^+$, ..., $w_k : E \rightarrow R^+$, and k constants c_1, c_2, ... c_k represented by a vector $c = (c_1, c_2, ..., c_k)$, *multi–constrained QoS routing* is to find a path p from src to dst such that $w(p) \leq c$, that is, $w_1(p) \leq c_1$, $w_2(p) \leq c_2$, ..., $w_k(p) \leq c_k$.

We will call a multi–constrained routing problem with k weight functions a *k–constrained* problem. Since the number of weight functions in a network is small, we will assume that k is a small constant.

`Definition 2`: Given a directed graph $G(N, E)$ with $k \geq 2$ weight functions $w_1 : E \rightarrow R^+$, $w_2 : E \rightarrow R^+$, ..., $w_k : E \rightarrow R^+$, a path $p = src \rightarrow v_1 \rightarrow v_2 \rightarrow ... \rightarrow dst$ is said to be an *optimal QoS path* from src to dst if there does not exist another path q from src to dst such that $w(q) < w(p)$.

When $k = 1$, the optimal QoS path is the same as the shortest path. When $k > 1$, however, there can be multiple optimal QoS paths between two nodes. For example, in Figure 1, both path $p_1 = 0 \rightarrow 1 \rightarrow 3$ ($w(p_1) = (40.0, 2.0)$) and path $p_2 = 0 \rightarrow 2 \rightarrow 3$ ($w(p_2) = (2.0, 40.0)$) are optimal QoS paths from node 0 to node 3. Path $p_3 = 0 \rightarrow 3$ is not an optimal QoS path since $w(p_3) = (50.0, 4.0) > w(p_1)$. Optimal QoS paths are interesting because each optimal QoS path can potentially satisfy particular QoS constraints that no other path can satisfy. On the other hand, when there exists a path that satisfies the QoS requirement, there always exists an optimal QoS path that satisfies the same QoS requirement. Thus, a QoS routing algorithm can guarantee finding a path that satisfies the QoS constraints when such a path exists if the algorithm considers all optimal QoS paths. Notice that the number of optimal QoS paths can be exponential with respect to the network size as shown in Figure 2. In Figure 2, the number of optimal QoS paths from node $src = 0$ to node $dst = 3k$ is equal to 2^k because from each node $3i$ where $0 \leq i < k$, taking the link $3i \rightarrow 3i + 1$ or $3i \rightarrow 3i + 2$ will result in different optimal QoS paths.

C. Extended Bellman–Ford Algorithm

Since the heuristics that we consider are variations of the extended Bellman–Ford algorithm, we will describe a version of the extended Bellman–Ford algorithm in this section for the completeness of the paper. Figure 3 shows the algorithm, which

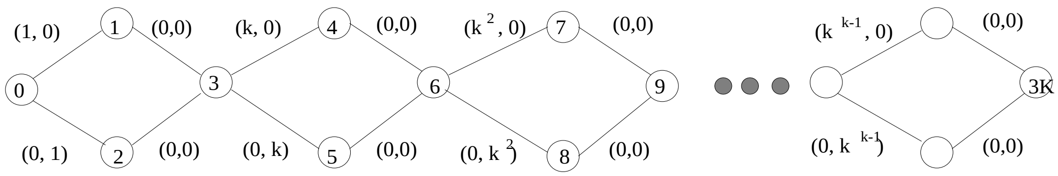


Fig. 2. The number of optimal QoS paths between two nodes

is a variation of the Constrained Bellman–Ford algorithm in [9]. For simplicity, the algorithm only checks whether there exists a path that satisfies the QoS constraints. The algorithm can easily be modified to find the exact path. We will call the algorithm $EBFA$.

$EBFA$ extends the original Bellman-Ford shortest path algorithm [3] by having each node u to maintain a set $PATH(u)$ that records all optimal QoS paths found so far from src to u. The first three lines in the main routine (BELLMAN_FORD) initialize the variables. Lines (4) to (6) perform the relax operations. After the relax operations all optimal QoS paths from node src to node dst are stored in the set $PATH(dst)$. Lines (7) and (8) check whether there exists an optimal QoS path that satisfies the QoS constraints. The $RELAX(u, v, w)$ operation is a little more complicated since all the elements in $PATH(u)$ and $PATH(v)$ must be considered. For each element $w(p)$ in $PATH(u)$, line (4) in the RELAX routine checks whether there exists an old path q from src to v that is better than path $p + (u \rightarrow v)$. If such a path exists, then $p + (u \rightarrow v)$ is not an optimal QoS path. Line (6) checks whether path $p + (u \rightarrow v)$ is better than any old path from src to v. If such an old path q exists, then path q is not an optimal QoS path and is removed from the set $PATH(v)$. Line (8) adds the newly found optimal QoS path to $PATH(v)$.

$EBFA$ guarantees to find a path that satisfies the QoS constraints when such a path exists by recording all optimal QoS paths in each node. Given a network $G(N, E)$, the algorithm executes the $RELAX$ operation $O(|N||E|)$ times. The time and space needed to execute $RELAX(u, v, w)$ depend on the sizes of $PATH(u)$ and $PATH(v)$, which are the number of optimal QoS paths from node src to nodes u and v respectively. Since the number of optimal QoS paths from src to u or v can be exponential with respect to $|N|$ and $|E|$, the time and space requirement of $EBFA$ may also grow exponentially. Thus, heuristics must be developed to reduce the time and space complexity.

The idea of both the limited granularity heuristic and the limited path heuristic is to limit the number of optimal QoS paths maintained in each node, that is, the size of $PATH$, to reduce the time and space complexity of the $RELAX$ operation. By limiting the size of $PATH$, each node is not able to record all optimal QoS paths from the source and the heuristics can only find approximate solutions. Thus, the challenge of the heuristics is how to limit the size of $PATH$ in each node while maintaining the effectiveness in finding paths that satisfy QoS constraints. In the next few sections, we will discuss two different methods to limit the size of $PATH$ and study their performance when solving general k–constrained QoS routing problems.

RELAX(u, v, w)
(1) For each $w(p)$ in PATH(u)
(2) $\quad flag = 1$
(3) $\quad$ For each $w(q)$ in Path(v)
(4) $\qquad$ if $(w(p) + w(u, v) \geq w(q))$ then
(5) $\qquad\quad flag = 0$
(6) $\qquad$ if $(w(p) + w(u, v) < w(q))$ then
(7) $\qquad\quad$ remove $w(q)$ from $PATH(v)$
(8) $\quad$ if $(flag = 1)$ then
(9) $\qquad$ add $w(p) + w(u, v)$ to $PATH(v)$

BELLMAN-FORD(G, w, c, src, dst)
(1) For $i = 0$ to $|N(G)| - 1$
(2) $\quad PATH(i) = \phi$
(3) $PATH(src) = \{\vec{0}\}$
(4) For $i = 1$ to $|N(G)| - 1$
(5) $\quad$ For each edge $(u, v) \in E(G)$
(6) $\qquad$ RELAX(u, v, w)
(7) For each $w(p)$ in $PATH(dst)$
(8) $\quad$ if $(w(p) < c)$ then return "yes"
(9) return "no"

Fig. 3. The extended Bellman–Ford algorithm (EBFA) for multi–constrained QoS routing

IV. The limited granularity heuristic

When all QoS metrics except one take bounded integer values, the multi-constrained QoS routing problem is solvable in polynomial time. The idea of the limited granularity heuristic is to use bounded finite ranges to approximate QoS metrics, which reduces the original NP–hard problem to a simpler problem that can be solved in polynomial time. This algorithm is a generalization of the algorithms in [1], [10]. To solve the k–constrained problem defined in Section 3.2, the limited granularity heuristic approximates $k - 1$ metrics with $k - 1$ bounded finite ranges. Let w_2, ..., w_k be the $k - 1$ metrics to be approximated, that is, for $2 \leq i \leq k$, the range $(0, c_i]$ is mapped into X_i elements, r_1^i, r_2^i, ..., $r_{X_i}^i$, where $0 < r_1^i < r_2^i < ... < r_{X_i}^i = c_i$. The $w_i(e) \in (0, c_i]$ is approximated by r_j^i if and only if $r_{j-1}^i < w_i(e) \leq r_j^i$. In the rest of the section, we will use the notation $aw_i(p)$, $2 \leq i \leq k$, to denote the approximated $w_i(p)$ in the bounded finite domain $\{r_1^i, r_2^i, ..., r_{X_i}^i\}$.

Figure 4 shows the limited granularity heuristic that solves k–constrained problems. In this heuristic, each node u maintains a table $d^u[1 : X_2, 1 : X_3, ..., 1 : X_k]$. An entry $d^u[i_2, i_3, ..., i_k]$ in the table records the path that has the smallest w_1 weight among all paths p from the source to node u that satisfy $w_2(p) \leq r_{i_2}^2$, $w_3(p) \leq r_{i_3}^3$, ..., $w_k(p) \leq r_{i_k}^k$. In the $RELAX(u, v, w)$ opera-

RELAX(u, v, w)
for each $d^v[i_2, i_3, ..., i_k]$
    Here, $1 \leq i_2 \leq X_2$, ..., $1 \leq i_k \leq X_k$
    Let $d^v[\vec{i}] = d^v[i_2, i_3, ..., i_k]$
    Let j_l be the largest j_l such that
    $r_{j_l} < r_{i_l} - w_l(u, v)$, $2 \leq l \leq k$
    Let $d^v[\vec{j}] = d^v[j_2, j_3, ..., j_k]$
    if ($j_l \geq 1$, for all $2 \leq l \leq k$) then
      if ($d^v[\vec{i}] > d^u[\vec{j}] + w_1(u, v)$) then
        $d^v[\vec{i}] = d^u[\vec{j}] + w_1(u, v)$

Limited_Granularity_Heuristic(G, w, c, src, dst)
For $i = 0$ to $|N(G)| - 1$
  For each $d^i[i_2, i_3, ..., i_k]$
    Here, $1 \leq i_2 \leq X_2$, ..., $1 \leq i_k \leq X_k$
    if ($i = src$) then $d^{src}[i_2, i_3, ..., i_k] = 0$
    else $d^i[i_2, i_3, ... i_k] = \infty$
For $i = 1$ to $|N(G)| - 1$
  For each edge $(u, v) \in E(G)$
    RELAX(u, v, w)
if ($d^{dst}[X_2, X_3, ..., X_k] < c_1$) then return TRUE
return FALSE

Fig. 4. The limited granularity heuristic for k–constrained routing problems

tion, to compute $d^v[i_2, i_3, ..., i_k]$, only $d^u[j_2, j_3, ..., j_k]$ where j_l is the largest j_l such that $r^l_{j_l} \leq r^l_{i_l} - w_l(u, v)$, for $2 \leq l \leq k$, needs to be considered. The $RELAX$ routine has a time complexity of $O(X_2X_3...X_k)$. Notice that the approximation of the weights is carried out implicitly in the $RELAX$ operation. For example, if, for each path p from src to dst, there exists an i, $2 \leq i \leq k$, such that $aw_i(p = src \to v_0 \to v_1 \to ... \to dst) > c_i$, then $d^{dst}[X_2, X_3, ..., X_k] = \infty$ at the end of the algorithm after all RELAX operations are done.

Let $X = X_2X_3...X_k$ be the size of the table maintained in each node. By limiting the granularity of the QoS metrics, the limited granularity heuristic has a time complexity of $O(X|N||E|)$. The most important issue of this heuristic is to determine the relation between the size of the table (which, in turn, determines the time complexity of the heuristic) and the effectiveness of the heuristic in finding paths that satisfy the k QoS constraints. The following lemmas attempt to answer this question.

`Lemma 1`: In order for the limited granularity heuristic to find any path of length L that satisfies the QoS constraints, the size of the table in each node must be at least L^{k-1}. That is, $X = X_2X_3...X_k \geq L^{k-1}$.

`Proof`: Assuming $X = X_2X_3...X_k < L^{k-1}$, there exists an i, $2 \leq i \leq k$, such that $X_i < L$. Let $p = v_0 \to v_1 \to v_2 \to ... \to v_L$ be a path that satisfies the QoS constraints $w(p) \leq c$. Let the range $(0, c_i]$ be approximated by X_i discrete elements, $r^i_1, r^i_2, ..., r^i_{X_i}$, where $0 < r^i_1 < r^i_2 < ... < r^i_{X_i} = c_i$.

Let $p(n)$ denote the path $v_0 \to v_1 \to v_2 \to ... \to v_n$. By induction, it can be shown that $aw_i(p(n)) \geq r^i_n$. Base case, when $n = 1$, since r^i_1 is the smallest value that can be used to approximate, $aw_i(v_0 \to v_1) \geq r^i_1$. Assuming that $aw_i(p(n-1)) \geq r^i_{n-1}$, $aw_i(p(n)) = aw_i(p(n-1)) + aw_i(v_{n-1} \to v_n) \geq r^i_{n-1} + w_i(v_{n-1} \to v_n) > r^i_{n-1} \geq r^i_n$. Thus, $aw_i(p(X_i)) \geq r^i_X = c_i$. When $L > X_i$, $aw_i(p(L)) > c_i$. That is, the approximation value for the $w_i(p)$ weight is larger than c_i. Thus, the heuristic does not recognize the path as a path that satisfies $w(p) \leq c$. □

Lemma 1 shows that in order for the limited granularity heuristic to be effective in finding paths of length L that satisfy k independent path constraints, the number of entries in each node should be at least L^{k-1}. For an N–node network, paths can potentially be of length N. Thus, the limited granularity heuristic should at least maintain a table of size $O(|N|^{k-1})$ in each node to be effective. This result indicates that the limited granularity heuristic is quite sensitive to the number of constraints, k. Notice that this lemma does not make any assumptions about the values of X_2, ..., X_k and the values of r^i_j, where $2 \leq i \leq k$ and $1 \leq j \leq X_i$. Thus, it applies to all variations of the limited granularity heuristic.

`Lemma 2`: Let n be a constant, $X_2 = X_3 = ... = X_k = nL$ so that $X = X_2X_3...X_k = n^{k-1}L^{k-1}$. For all $2 \leq i \leq k$, let the range $(0, c_i]$ be approximated with equally spaced values $\{r^i_1 = \frac{c_i}{X_i}, r^i_2 = \frac{c_i}{X_i} * 2, ..., r^i_{X_i} = c_i\}$. The limited granularity heuristic guarantees finding a path q that satisfies $w(q) \leq c$ if there exists a path p of length L that satisfies

$$w_1(p) \leq c_1 \text{ and } w_i(p) \leq c_i - \frac{c_i}{n} \text{ for all } 2 \leq i \leq k.$$

`Proof:` Consider the approximation of any ith weight of path p, $2 \leq i \leq k$,

$$\begin{aligned} aw_i(p) &= \textstyle\sum_{(u \to v) \text{ on } p} aw_i(u \to v) \\ &< \textstyle\sum_{(u \to v) \text{ on } p} (w_i(u \to v) + \frac{c_i}{X_i}) \\ &= \textstyle\sum_{(u \to v) \text{ on } p} w_i(u \to v) + \sum_{(u \to v) \text{ on } p} \frac{c_i}{X_i} \\ &\leq c_i - \frac{c_i}{n} + \frac{L}{X_i} * c_i = c_i \end{aligned}$$

Thus, the approximation of all w_i weights, $2 \leq i \leq k$, will satisfy the condition $w_i(p) \leq c_i$. Since the heuristic does not approximate the w_1 weight, the heuristic can guarantee finding that path p satisfies $w(p) \leq c$. □

Lemma 2 shows that when each node maintains a table of size $n^{k-1}|N|^{k-1} = O(|N|^{k-1})$ and when n is a reasonably large constant, the limited granularity heuristic can find most of the paths that satisfy the QoS constraints. Furthermore, by maintaining a table of size $n^{k-1}N^{k-1}$, the heuristic guarantees finding a solution when there exists a path whose QoS metrics are better than $(1 - \frac{1}{n}) * \vec{c}$, where $\vec{c}$ is the required QoS metrics of the connection. This guarantee will be called finding an $(1 - \frac{1}{n})$-*approximate solution*. For example, if $n = 100$, the heuristic guarantees finding a path p that satisfies $w(p) \leq c$ when there exists a path q that satisfies $w(q) \leq 0.99 * c$, that is, it guarantees finding an 0.99-*approximate solution*.

V. THE LIMITED PATH HEURISTIC

The limited path heuristic ensures the worst case polynomial time complexity by maintaining a limited number of optimal QoS paths, say X optimal QoS paths, in each node. Here, X corresponds to the size of the table maintained in each node in the limited granularity heuristic. The limited path heuristic is basically the same as the extended Bellman–Ford algorithm in Figure 3 except that before a path is inserted into $PATH$, the size of $PATH$ is checked. When $PATH$ already contains X elements, the new path will not be inserted. By limiting the size of $PATH$ to X, the time complexity of the $RELAX$ operation is reduced to $O(X^2)$. The time complexity of the heuristic is $O(X^2|N||E|)$.

We must choose the value X carefully for the heuristic to be both efficient and effective. If X is sufficiently large such that each node actually records all optimal QoS paths, the heuristic is as effective as $EBFA$. However, large X results in an inefficient heuristic in terms of the time/space complexity. In [10], it was established that for a network of N nodes, the limited path heuristic can solve 2–constrained problems with very high probability when $X = O(|N|^2 lg(|N|))$. In this section, we extend this result and show that the limited path heuristic can solve general k–constrained problems with very high probability when $X = O(|N|^2 lg(|N|))$, assuming that k is a small constant. This result is significant in the sense that it indicates that unlike the limited granularity heuristic, the limited path heuristic is insensitive to the number of QoS constraints in the network.

Let us assume that the weights of the links in a graph are randomly generated and are independent of one another. For a set S of $|S|$ paths of the same length, we derive the probability $prob_i$ that set S contains i optimal QoS paths. We then show that when $X = O(|N|^2 lg(|N|))$, $\sum_{i=1}^{X} prob_i$ is very large (or $\sum_{i=X+1}^{|S|} prob_i$ is very small), which indicates that when each node maintains $O(|N|^2 lg(|N|))$ entries, the limited path heuristic will have very high probability to record all optimal QoS paths in each node and thus, will have very high probability to find the QoS paths when such paths exist.

We use the following process to derive the probability $prob_i$ that the set S contains i optimal QoS paths. First, the path, p, that has the smallest w_1 weight is chosen from S. The path p is an optimal QoS path because $w_1(p)$ is the smallest among all the paths. All paths whose w_j weights, $2 \le j \le k$, are larger than $w_j(p)$ are not optimal QoS paths. Let the set T include all such non–optimal QoS paths. The set $S - T$ contains all paths q where there exists at least one j, $2 \le j \le k$, such that $w_j(q) < w_j(p)$. Thus, a path in the set $S - T$ may potentially be an optimal QoS path. The process is then repeated on the set $S - T$. If S contains m optimal QoS paths, the process can be repeated m times.

Let us use the notion $P_k^{i,j}$ to represent the probability of the remaining set size equal to j when the process is applied to a set of i paths and the number of QoS metrics is k. We will always assume $0 \le j \le i-1$, when the notion $P_k^{i,j}$ is used. The process can be modeled as a Markov process as shown in Figure 5. The Markov chain contains $|S|+1$ states, each state i in the Markov chain represents a set of i paths. The transition matrix for the Markov chain is

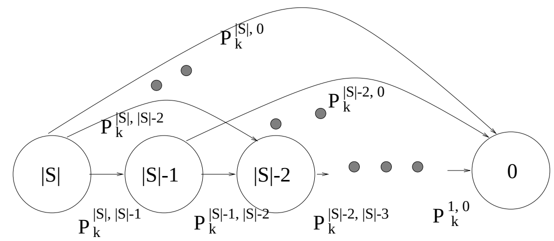


Fig. 5. The Markov Chain

$$A_k = \begin{pmatrix} 0 & 0 & & 0 & 0 \\ P_k^{1,0} & 0 & \cdots & 0 & 0 \\ P_k^{2,0} & P_k^{2,1} & \cdots & 0 & 0 \\ & & \cdots & & \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ P_k^{|S|-1,0} & P_k^{|S|-1,1} & \cdots & 0 & 0 \\ P_k^{|S|,0} & P_k^{|S|,1} & \cdots & P_k^{|S|,|S|-1} & 0 \end{pmatrix}$$

.

Let us define $A_k^1 = A_k$ and $A_k^m = A_k^{m-1} A_k$ for $m > 1$. $A_k^m(i,j)$ represents the probability of the state transferring from node i to node j in exactly m steps. For example, $A_k^1(|S|, 0)$ represents the probability of a set of size $|S|$ became empty after one optimal QoS path is chosen. $A_k^m(|S|, 0)$ is the probability that the set of size $|S|$ becomes empty after selecting exactly m optimal QoS paths, that is, $A_k^m(|S|, 0)$ is the probability that the set S contains exactly m optimal QoS paths. Our goal is to determine the value X such that $\sum_{m=X}^{|S|} A_k^m(|S|, 0)$ is very small.

A special case, when $k = 2$, was studied in [10]. When $k = 2$, each link has two weights w_1 and w_2. In the path selection process, we choose from the set of i paths a path whose w_1 weight is the smallest. Since w_2 and w_1 weights are independent and the length of the paths are the same, the probability of the size of the remaining set may be 0, 1, ..., $i-1$, each with probability $\frac{1}{i}$, since $w_2(p)$ can be ranked 1, ..., i among the i paths in the set with equal probability $\frac{1}{i}$, $P_2^{i,j} = \frac{1}{i}$. By manipulating the transition matrix, A_2, it was shown that when each node maintains $O(|N|^2 lg(|N|))$ paths, the heuristic can solve the 2-constrained problem with very high probability. Next, we will first derive the formula for computing the general $P_k^{i,j}$, then prove that maintaining $O(|N|^2 lg(|N|))$ paths enables the heuristic to solve the general k–constrained problem with very high probability. Lemma 3 shows how to compute $P_k^{i,j}$.

`Lemma 3:` $P_k^{i,j} = \frac{1}{i} \sum_{l=0}^{j} P_{k-1}^{i-l,j-l}$.

`Proof:` Let S be the set of i paths. Let p be the path with the smallest $w_1(p)$. Consider $w_2(p)$, since the weights are randomly generated and are independent, $w_2(p)$ can be ranked 1, 2, ..., i among all the paths with equal probability $\frac{1}{i}$. In other words, the probability that there are l, $0 \le l \le i-1$, paths whose w_2 weights are smaller than $w_2(p)$ is $\frac{1}{i}$. When $l = 0$, all the $i-1$ paths are potential candidates to be considered for the rest $k-2$ weights in the remaining set. In this case, the probability that the remaining set size equal to j is equivalent to the case to choose from i paths the path with the smallest w_1 weight and the remaining set size is equal to j with $k-1$ weights. Thus, the probability is $P_{k-1}^{i,j}$. When $l=1$, there exists one path q where $w_2(q) < w_2(p)$, thus, path q belongs to the remaining set. In this case, the probability that the remaining set size equal to j is equivalent to the case to choose from $i-1$ paths (all paths but path q) the path with the smallest w_1 weight and the remaining set size is equal to $j-1$ with $k-1$ weights (since path q is already in the remaining set by considering w_2). Thus, the probability is $P_{k-1}^{i-1,j-1}$. Similar arguments apply for all cases from $l=0$ to $l=j$. When $l>j$, there will be at least l paths in the remaining set, thus, the probability that the remaining set size equal to j is 0. Combining all these cases, we obtain

$$\begin{aligned}P_k^{i,j} &= \tfrac{1}{i}P_{k-1}^{i,j} + \tfrac{1}{i}P_{k-1}^{i-1,j-1} + ... + \tfrac{1}{i}P_{k-1}^{i-j,0}\\ &= \tfrac{1}{i}\sum_{l=0}^{j} P_{k-1}^{i-l,j-l}. \square\end{aligned}$$

Next, we will introduce a number of lemmas (Lemmas 4, 5, and 6) that summarize some property of $P_k^{i,j}$ and A_k.

`Lemma 4:` $P_k^{i,j} > P_k^{i+1,j}$.

`Proof:` Base case, $k=2$, $P_2^{i,j} = \frac{1}{i} > \frac{1}{i+1} = P_2^{i+1,j}$.

Induction case, assuming that for any i, j and k, $P_k^{i,j} > P_k^{i+1,j}$, $P_{k+1}^{i,j} = \frac{1}{i}\sum_{l=0}^{j} P_k^{i-l,j-l} > \frac{1}{i+1}\sum_{l=0}^{j} P_k^{i+1-l,j-l} = P_{k+1}^{i+1,j}$ $\square$

`Lemma 5:` For $k \geq 3$ and $0 \leq j \leq |S|$,

$$\sum_{i=0}^{|S|} A_k(i,j) = \sum_{i=j+1}^{|S|} P_k^{i,j} < 2.$$

`Proof:` Base case, $k=3$. From Lemma 3, $P_3^{i,j} = \frac{1}{i}(\sum_{l=0}^{j} P_2^{i-l,j-l}) = \frac{1}{i}(\frac{1}{i}+\frac{1}{i-1}+...+\frac{1}{i-j})$. Hence, for any j,

$$\begin{aligned}&\sum_{i=0}^{|S|} A_3(i,j) - \sum_{i=0}^{|S|} A_3(i,j+1)\\ &= \sum_{i=j+1}^{|S|} P_3^{i,j} - \sum_{i=j+2}^{|S|} P_3^{i,j+1}\\ &= \tfrac{1}{j}(\tfrac{1}{j}+\tfrac{1}{j-1}+...+\tfrac{1}{1}) - (\tfrac{1}{j+1}\tfrac{1}{1}+\tfrac{1}{j+2}\tfrac{1}{2}+...+\tfrac{1}{i}\tfrac{1}{i-j-1})\\ &> 0\end{aligned}$$

Thus, $2 > \sum_{i=1}^{|S|}\frac{1}{i^2} = \sum_{i=0}^{|S|} A_3(i,0) > \sum_{i=0}^{|S|} A_3(i,1) > ... > \sum_{i=0}^{|S|} A_3(i,|S|)$.

Induction case, for any j and k, assuming $\sum_{i=0}^{|S|} A_k(i,j) = \sum_{i=j+1}^{|S|} P_k^{i,j} < 2$,

$$\begin{aligned}\sum_{i=0}^{|S|} A_{k+1}(i,j) &= \sum_{i=j+1}^{|S|} P_{k+1}^{i,j}\\ &= \sum_{i=j+1}^{|S|}\tfrac{1}{i}\sum_{l=0}^{j} P_k^{i-l,j-l}\\ &< \tfrac{1}{j+1}\sum_{l=0}^{j}\sum_{i=l+1}^{|S|} P_k^{i,l}\\ &\leq \tfrac{1}{j+1}*(2*(j+1))\\ &= 2 \square\end{aligned}$$

`Lemma 6:` $P_k^{i,j} \leq \frac{1}{i}(\frac{1}{i}+\frac{1}{i-1}+...+\frac{1}{i-j})^{k-2}$.

`Proof:` Base case, $k=2$,

$$P_2^{i,j} = \tfrac{1}{i} \leq \tfrac{1}{i}(\tfrac{1}{i}+\tfrac{1}{i-1}+...+\tfrac{1}{i-j})^{2-2}.$$

Induction case, assuming that for any i, j and k,

$$P_k^{i,j} \leq \tfrac{1}{i}(\tfrac{1}{i}+\tfrac{1}{i-1}+...+\tfrac{1}{i-j})^{k-2}.$$

$$\begin{aligned}P_{k+1}^{i,j} &= \tfrac{1}{i}(P_k^{i,j} + P_k^{i-1,j-1} + ... + P_k^{i-j,0})\\ &\leq \tfrac{1}{i}(\tfrac{1}{i}(\tfrac{1}{i}+\tfrac{1}{i-1}+...+\tfrac{1}{i-j})^{k-2}\\ &\quad+\tfrac{1}{i-1}(\tfrac{1}{i-1}+\tfrac{1}{i-2}+...+\tfrac{1}{i-j})^{k-2}\\ &\quad+...+\tfrac{1}{i-j}(\tfrac{1}{i-j})^{k-2})\\ &\leq \tfrac{1}{i}(\tfrac{1}{i}(\tfrac{1}{i}+\tfrac{1}{i-1}+...+\tfrac{1}{i-j})^{k-2}\\ &\quad+\tfrac{1}{i-1}(\tfrac{1}{i}+\tfrac{1}{i-1}+...+\tfrac{1}{i-j})^{k-2}\\ &\quad+...+\tfrac{1}{i-j}(\tfrac{1}{i}+\tfrac{1}{i-1}+...+\tfrac{1}{i-j})^{k-2})\\ &= \tfrac{1}{i}((\tfrac{1}{i}+...+\tfrac{1}{i-j})(\tfrac{1}{i}+...+\tfrac{1}{i-j})^{k-2})\\ &= \tfrac{1}{i}(\tfrac{1}{i}+\tfrac{1}{i-1}+...+\tfrac{1}{i-j})^{k-1} \square\end{aligned}$$

The following three Lemmas (Lemmas 7, 8 and 9) are mathematic formulae to be used later.

`Lemma 7:` For a constant k, there exists a constant c such that

$$\sum_{i=1}^{\infty}\tfrac{1}{2^i}i^k \leq c.$$

`Proof:` When $k=0$, $\sum_{i=1}^{\infty}\frac{1}{2^i}i^k = \sum_{i=1}^{\infty}\frac{1}{2^i} = 1$.

Let $Y(k) = \sum_{i=1}^{\infty}\frac{1}{2^i}i^k$, $\frac{Y(k)}{2} = \sum_{i=2}^{\infty}\frac{x}{2^i}(i-1)^k$

$$\begin{aligned}\tfrac{Y(k)}{2} &= Y(k) - \tfrac{Y(k)}{2}\\ &= \tfrac{1}{2} + \sum_{i=2}^{\infty}\tfrac{1}{2^i}(i^k-(i-1)^k)\\ &\leq \tfrac{1}{2} + \sum_{i=2}^{\infty}\tfrac{1}{2^i}(k*i^{k-1})\\ &\leq k*Y(k-1)\end{aligned}$$

Thus, $Y(k) \leq 2kY(k-1) \leq 2^2k(k-1)*Y(k-2) \leq ... \leq 2^kk!Y(0) = 2^kk!$. When k is a constant, there exists a constant $c = 2^kk!$ such that $\sum_{i=1}^{\infty}\frac{1}{2^i}i^k \leq c$. $\square$

`Lemma 8:` For a constant k and $1 \leq j \leq i-1$, there exists a constant c such that

$$\sum_{n=j+1}^{i-1}(\tfrac{1}{i}+\tfrac{1}{i-1}+...\tfrac{1}{i-n})^k(\tfrac{1}{n}+...+\tfrac{1}{n-j})^k \leq c*i.$$

`Proof:` Let $W(m) = (\frac{1}{i}+\frac{1}{i-1}+...+\frac{1}{i-m})^k$. We will first derive some bounds for $W(m)$.

For $1 \leq m \leq \frac{i}{2}$, $W(m) = (\frac{1}{i}+\frac{1}{i-1}+...+\frac{1}{i-m})^k \leq (\frac{1}{i/2}+\frac{1}{i/2}+...+\frac{1}{i/2})^k \leq (m*\frac{1}{i/2})^k \leq 1^k$.

For $\frac{i}{2}+1 \leq m \leq \frac{3i}{4}$, $W(m) \leq (\frac{1}{i/2}+\frac{1}{i/2}+...+\frac{1}{i/2}+(m-\frac{i}{2})*\frac{1}{i/4})^k \leq 2^k$.

In general, for $\frac{(2^j-1)}{2^j}*i+1 \leq m \leq \frac{(2^{j+1})-1}{2^{j+1}}*i$,

$$W(m) \leq (j+1)^k.$$

For $n \leq i$, we also have $(\frac{1}{n}+...+\frac{1}{n-j})^k \leq (\frac{1}{i}+...+\frac{1}{n-j})^k = W(i-n+j)$. Thus,

$$\begin{aligned}&\sum_{n=j+1}^{i-1}(\tfrac{1}{i}+\tfrac{1}{i-1}+...\tfrac{1}{i-n})^k(\tfrac{1}{n}+...+\tfrac{1}{n-j})^k\\ &\leq \sum_{n=j+1}^{i-1} W(n)W(i-n+j)\\ &= W(i-1)W(j+1)+...+W(j+1)W(i-1)\\ &\leq W(j+1)^2+...+W(i-1)^2 \quad \text{/* } a^2+b^2 \geq 2ab \text{ */}\\ &= \sum_{n=j+1}^{i-1}(W(n))^2 \leq \sum_{n=1}^{i-1}(W(n))^2\\ &= \sum_{n=1}^{\frac{i}{2}}(W(n))^2 + \sum_{n=\frac{i}{2}+1}^{\frac{3i}{4}}(W(n))^2 + ...\\ &= \tfrac{i}{2}1^{2k} + \tfrac{i}{4}2^{2k} + \tfrac{i}{8}3^{2k} + ...\end{aligned}$$

$\leq c*i$, where c is a constant. /* Applying Lemma 7 */$\square$

`Lemma 9:` Let $0 \leq j < \frac{i}{2}$ and m be a constant, there exists a constant c such that

$$\sum_{n=j+1}^{i-1} P_k^{n,j}(\tfrac{1}{i}+...+\tfrac{1}{i-n})^m \leq c.$$

`Proof:` From Lemma 6, we have $\sum_{n=j+1}^{i-1} P_k^{n,j} < 2$. From Lemma 5, we have $P_k^{i,j} > P_k^{i+1,j}$. Hence,

$$\sum_{n=j+1}^{\frac{i}{2}} P_k^{n,j} < 2$$
$$\sum_{n=\frac{i}{2}+1}^{\frac{3i}{4}} P_k^{n,j} < 2$$
$$\sum_{n=\frac{3i}{4}+1}^{\frac{7i}{8}} P_k^{n,j} < \frac{1}{2} \sum_{n=\frac{i}{2}+1}^{\frac{3i}{4}} P_k^{n,j} < 2 * \frac{1}{2}$$
$$\sum_{n=\frac{7i}{8}+1}^{\frac{15i}{16}} P_k^{n,j} < \frac{1}{4} \sum_{n=\frac{i}{2}+1}^{\frac{3i}{4}} P_k^{n,j} < 2 * \frac{1}{4}$$
...

Thus,

$$\begin{aligned}
&\sum_{n=j+1}^{i-1} P_k^{n,j} (\tfrac{1}{i} + ... + \tfrac{1}{i-n})^m \\
&= \sum_{n=j+1}^{\frac{i}{2}} P_k^{n,j} (\tfrac{1}{i} + ... + \tfrac{1}{i-n})^m \\
&\quad + \sum_{n=\frac{i}{2}+1}^{\frac{3i}{4}} P_k^{n,j} (\tfrac{1}{i} + ... + \tfrac{1}{i-n})^m \\
&\quad + \sum_{n=\frac{3i}{4}+1}^{\frac{7i}{8}} P_k^{n,j} (\tfrac{1}{i} + ... + \tfrac{1}{i-n})^m + ... \\
&\leq 1^m \sum_{n=\frac{i}{2}+1}^{\frac{3i}{4}} P_k^{n,j} + 2^m \sum_{n=\frac{i}{2}+1}^{\frac{3i}{4}} P_k^{n,j} \\
&\quad + 3^m \sum_{n=\frac{3i}{4}+1}^{\frac{7i}{8}} P_k^{n,j} + ... \\
&\leq 1^m * 2 + 2(2^m * \tfrac{1}{2^0} + 3^m * \tfrac{1}{2^1} + 4^m * \tfrac{1}{2^2} + ...)
\end{aligned}$$
$\leq c$, where c is a constant. /* applying Lemma 7 */ □

Lemma 10 describes the relation between $A_2(i,j)$ and $A_k^2(i,j)$.

`Lemma 10:` There exists a constant c such that

$$A_k^2(i,j) \leq c * A_2(i,j).$$

`Proof:` Consider the following three cases:

- Case 1: $j \geq i-1$. In this case, $A_k^2(i,j) = 0 \leq A_2(i,j)$.
- Case 2: $\frac{i}{2} \leq j \leq i-2$. In this case,

$$\begin{aligned}
&A_k^2(i,j) \\
&= \sum_{n=j+1}^{i-1} P_k^{i,n} * P_k^{n,j} \\
&\leq \sum_{n=j+1}^{i-1} \tfrac{1}{i}(\tfrac{1}{i} + ... + \tfrac{1}{i-n})^{k-2} * \tfrac{1}{n}(\tfrac{1}{n} + ... + \tfrac{1}{n-j})^{k-2} \\
&\leq \tfrac{2}{i^2} \sum_{n=j+1}^{i-1} (\tfrac{1}{i} + ... + \tfrac{1}{i-n})^{k-2} (\tfrac{1}{n} + ... + \tfrac{1}{n-j})^{k-2} \\
&\leq \tfrac{2c_1}{i} = 2c_1 A_2(i,j)
\end{aligned}$$
/* applying Lemma 8 */

- Case 3: $0 \leq j \leq \frac{i}{2} - 1$.

$$\begin{aligned}
A_k^2(i,j) &= \sum_{n=j+1}^{i-1} P_k^{i,n} * P_k^{n,j} \\
&\leq \sum_{n=j+1}^{i-1} P_k^{n,j} \tfrac{1}{i}(\tfrac{1}{i} + \tfrac{1}{i-1} + ... + \tfrac{1}{i-n})^{k-2} \\
&\leq \tfrac{1}{i} \sum_{n=j+1}^{i-1} P_k^{n,j} (\tfrac{1}{i} + \tfrac{1}{i-1} + ... + \tfrac{1}{i-n})^{k-2} \\
&\leq \tfrac{c_2}{i} = c_2 A_2(i,j)
\end{aligned}$$
/* applying Lemma 9 */

Thus, there exists a constant $c = max(2c_1, c_2, 1)$ such that $A_k^2(i,j) \leq c * A_2(i,j)$. □

`Theorem 1:` Given an N node graph with k independent constraints, the limited path heuristic has very high probability to record all optimal QoS paths and thus, has very high probability to find a path that satisfies the QoS constraints when one exists, when each node maintains $O(|N|^2 lg(|N|))$ paths.

`Proof:` The proof of this theorem uses results in [10] that $A_2^m(|S|,0) \leq \frac{(2ln(|S|))^{m+1}}{|S|(m+1)!}$. From Lemma 10, we have $A_k^2(i,j) \leq cA_2(i,j)$, and hence $A_k^m(i,j) \leq c^{\frac{m}{2}} A_2^{\frac{m}{2}}(i,j)$. Thus, $A_k^m(|S|,0) \leq c^{\frac{m}{2}} A_2^{\frac{m}{2}}(|S|,0) \leq \frac{(2cln(|S|))^{\frac{m}{2}+1}}{|S|(\frac{m}{2}+1)!}$.

Using the formula $n! \geq \sqrt{2\pi n}(\frac{n}{e})^n$ from [3]. When $m > 4ce^2 ln(|S|)$,

$$\begin{aligned}
A_k^m(|S|,0) &\leq \frac{(2cln(|S|))^{\frac{m}{2}+1}}{|S|(\frac{m}{2}+1)!} \leq \frac{(2cln(|S|))^{\frac{m}{2}+1}}{|S|(\frac{m}{2e})^{\frac{m}{2}+1}} \\
&\leq \frac{1}{|S|}(\frac{4ce*ln(|S|)}{m})^{\frac{m}{2}+1} \leq \frac{1}{|S|}(\frac{1}{e})^{2ce^2 ln(|S|)} \\
&\leq \frac{1}{|S|^{2ce^2+1}}
\end{aligned}$$

.

The number of paths of length L between any two nodes in the graph is at most $R = |N|^L$. The probability that there exists no more than i optimal QoS paths among the $R = |N|^L$ paths is $p = 1 - \sum_{m=i+1}^{R} A_k^m(R,0)$. When $i > 4ce^2 ln(R)$, $p = 1 - \sum_{m=i+1}^{R} A_k^m(R,0) \geq 1 - \sum_{k=i+1}^{R} \frac{1}{R^{2ce^2+1}} \geq 1 - \frac{1}{R^{2ce^2}}$ Thus, when each node maintains $2ce^2 ln(|N|^L) = 2ce^2 Llg(|N|)$ paths, the probability that the node can record all optimal QoS paths of length L is very high, $1 - \frac{1}{R^{2ce^2}}$. For example, if $R = 30$, the probability is more than $1 - \frac{1}{R^{2e^2}} > 99.99999\%$. In an N node graph, the length of any QoS path is between 1 and N. Thus, maintaining $\sum_{L=1}^{|N|} 2ce^2 Lln(|N|) = O(|N|^2 lg(|N|))$ paths in each node will give very high probability to record all optimal QoS paths in the node. Thus, the limited granularity heuristic has very high probability to find a path that satisfies the QoS constraints when such a path exists, when each node maintains $O(|N|^2 lg(|N|))$ paths. □

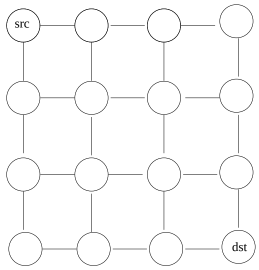
Theorem 1 establishes that the performance of the limited path heuristic is not as sensitive to the number of QoS constraints as the limited granularity heuristic. Thus, the limited path heuristic provides better performance when $k > 3$. Given that the global network state information is inherently imprecise, in practice, using an algorithm that can precisely solve the k–constrained routing problem may not have much advantage over the limited path heuristic that can solve the k–constrained routing problem with very high probability.

The proof of Theorem 1 assumes that paths of different lengths are of the same probability to be the optimal Qos paths. However, when the weights in a graph are randomly generated with uniform distribution, the paths of shorter length are more likely to be the optimal QoS paths. In addition, the probability used in the proof of Theorem 1 is extremely high. In practice, we do not need such high probability for the heuristic to be effective. A tighter upper bound for the number of optimal QoS paths to be maintained in each node for the limited path heuristic to be effective may be obtained by considering these factors. However, the formal derivation of a tighter upper bound can be complicated. In the next section, we further examine the two heuristics through simulation study.

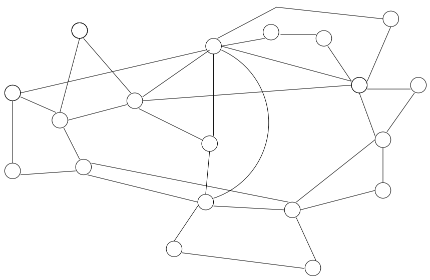
VI. Experiments

The goal of the simulation experiments is to compare the performance of the heuristics for real world network topologies and to study the impact of constants in the asymptotic bounds we derived. Two topologies, the mesh topology shown in Figure 6 (a) and the MCI backbone topology shown in Figure 6 (b), are used in the studies. In the simulation, the w_i weight of each link is randomly generated in the range of $(0.0, 10.0*i)$, for $1 \leq i \leq k$.

We compare the two heuristics with the exhaustive algorithm, $EBFA$, that guarantees finding a path that satisfies the QoS

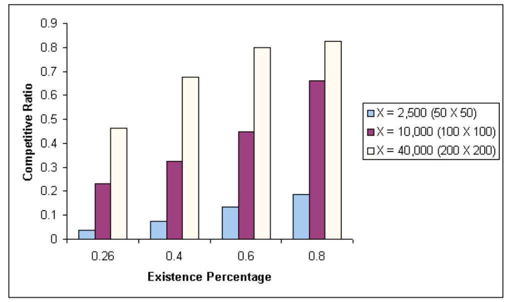


(a) A 4×4 mesh

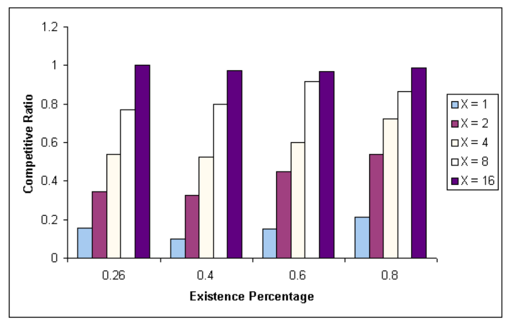


(b) MCI backbone

Fig. 6. The network topologies



(a) Limited Granularity Heuristic



(b) Limited Path Heuristic

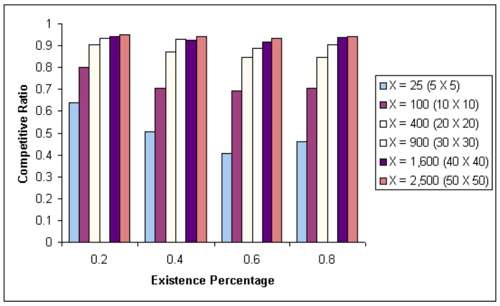
Fig. 7. Performance for 3–constrained problems in 8×8 meshes

constraints if such a path exists. Two concepts, the *existence percentage* and the *competitive ratio*, are used to describe the simulation results. The existence percentage, which indicates how difficult the paths that satisfy the QoS constraints can be found, is defined as the ratio of the total number of requests satisfied using the exhaustive algorithm and the total number of requests generated. The competitive ratio, which indicates how well a heuristic algorithm performs, is defined as the ratio of the number of requests satisfied using a heuristic algorithm and the number of requests satisfied using the exhaustive algorithm. By definition, both the existence percentage and the competitive ratio are in the range of $[0.0, 1.0]$.

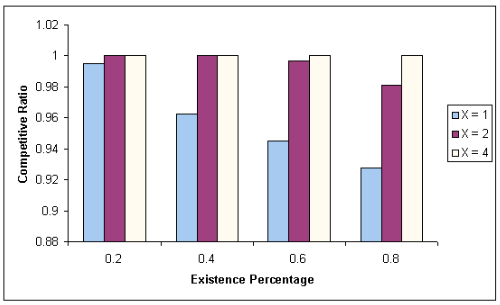
Figure 7 shows the performance of the two heuristics when they solve 3–constrained problems in 8×8 meshes. Since the performance of the two heuristics is closely related to the length of the paths, we choose the source and the destination to be the farthest apart in the mesh network as shown in Figure 6 (a). The existence percentage and the competitive ratio are obtained by solving 500 QoS routing problems with the same QoS requirement. Each routing problem uses a mesh whose weights are randomly generated. In general, when X is sufficiently large, both heuristics have high competitive ratio, which indicates that the performance is close to that of $EBFA$. However, the limited granularity heuristic must maintain a very large number of entries in each node to achieve good performance. In this experiment, the limited granularity heuristic must maintain 40000 entries (a 200×200 table) to achieve 80% competitive ratio when the existence percentage is high (0.8) and around 50% competitive ratio when the existence percentage is low (0.26). In contrast, the limited path heuristic achieves close to 100% competitive ratio for all different competitive ratios with 16 entries in each node. In addition, the existence percentage, which has a strong impact on the limited granularity heuristics, does not significantly affect the limited path heuristic. The limited path heuristic out–performs the limited granularity heuristic in terms of both efficiency and consistency.

Figure 8 shows the results when the two heuristics solve 3–constrained QoS routing problems in the MCI backbone topology. The existence percentage and the competitive ratio are obtained by solving 1000 QoS routing problems. Each of the 1000 routing problems tries to find a connection between randomly generated source and destination with the same QoS requirement. The general trend in this figure is similar to that in the previous experiment. In comparison to the limited path heuristic, the limited granularity heuristic requires significantly more resources to achieve good performance. The limited granularity heuristic must maintain 900 entries (a 30×30 table) in each node to consistently achieve 90% competitive ratio, while the limited path heuristic achieves close to 100% competitive ratio with 4 entries in each node.

Figure 9 shows the impact of the number of constraints on the performance of the heuristics using the MCI backbone topology. In this experiment, we fix the number of entries maintained at each node for both heuristics and study the performance of the two heuristics when they solve QoS routing problems with different numbers of QoS constraints. For the limited granularity heuristics, we fix the table size to be around 4,000. More specifically, we maintain in each node a linear array of 4,000

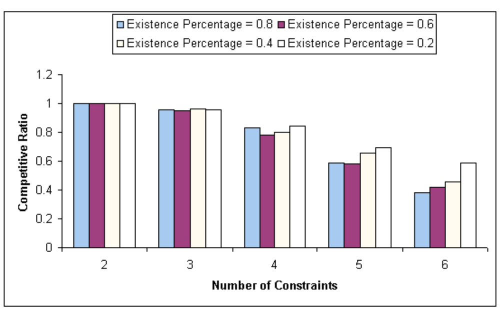


(a) Limited Granularity Heuristic

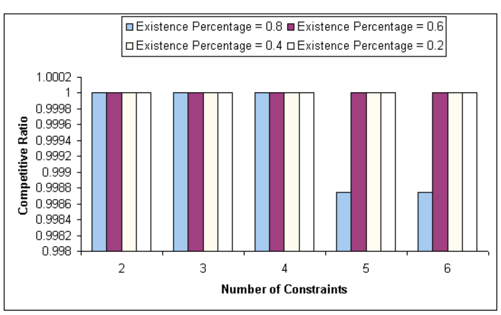


(b) Limited Path Heuristic

Fig. 8. Performance for 3–constrained problems in the MCI backbone



(a) Limited Granularity Heuristic



(b) Limited Path Heuristic

Fig. 9. Performance of the heuristics for different numbers of QoS constraints

for 2-constrained problems, a 64×64 table for 3-constrained problems, a $17 \times 17 \times 17$ table for 4-constrained problems, a $8 \times 8 \times 8 \times 8$ table for 5-constrained problems and a $6 \times 6 \times 6 \times 6 \times 6$ table for 6-constrained problems. For the limited path heuristic, we fix the table size to be 4. The results are obtained by solving 1000 QoS routing problems for each setting. As can be seen from the figure, the performance of the limited path heuristic is somewhat insensitive to the number of QoS constraints. With $X = 4$, the limited path heuristic achieves close to 100% competitive ratio for all different number of constraints. The performance of the limited granularity heuristic drastically degrades as the number of QoS constraints increases. The competitive ratio falls from nearly 100% to less than 40% when the number of constraints increases from 2 to 6. This experiment confirms that the limited path heuristic is more efficient than the limited granularity heuristic in solving general k–constrained problems when $k > 3$.

VII. Conclusion

In this paper, we study two heuristics, the limited granularity heuristic and the limited path heuristic, that can be applied to the extended Bellman–Ford algorithm to solve k–constrained QoS path routing problems. We show that although both heuristics can solve k–constrained QoS routing problems with high probability in polynomial time, to achieve high performance, the limited granularity heuristic requires much more resources than the limited path heuristic does. Specifically, the limited granularity heuristics must maintain a table of size $O(|N|^{k-1})$ in each node to achieve good performance, which results in a time complexity of $O(|N|^k|E|)$, while the limited path heuristic only needs to maintain $O(|N|^2 lg(|N|))$ entries in each node. Both our analytical and simulation results indicate that the limited path heuristic is more efficient than the limited granularity heuristic in solving general k–constrained QoS routing problems when $k > 3$, although previous research results show that both the limited granularity heuristic and the limited path heuristic can solve 2–constrained QoS routing problems efficiently. The advantage of the limited granularity heuristic is that, by maintaining a table of size $n^{k-1}N^{k-1}$, it guarantees finding $(1 - \frac{1}{n})$-approximate solutions while the limited path heuristic cannot provide such guarantee.

References

[1] S. Chen and K. Nahrstedt "On Finding Multi-Constrained Paths." *International Conference on Communications* (ICC'98), June 1998.

[2] Shigang Chen and Klara Nahrstedt " An Overview of Quality-of-Service Routing for the Next Generation High-Speed Networks: Problems and Solutions," *IEEE Network, Special Issue on Transmission and Distribution of Digital Video*, Nov./Dec. 1998.

[3] T. H. Cormen, C. E. Leiserson and R. L. Rivest, "Introduction to Algorithms.", *The MIT Press*, 1990.

[4] J.M. Jaffe "Algorithms for Finding Paths with Multiple Constraints." *Networks*, 14:95–116, 1984.

[5] Q. Ma and P. Steenkiste "Quality–of–Service Routing with Performance Guarantees", *IWQos'97*, May 1997.

[6] Ariel Orda, "Routing with End-to-End QoS Guarantees in Broadband Networks", *IEEE/ACM Transactions on Networking*, Vol. 7, No. 3, June 1999.

[7] H. F. Salama, D. S. Reeves and Y. Viniotis, "A Distributed Algorithm for Delay–Constrained Unicast Routing." *IEEE INFOCOM'97*, April 1997.

[8] Z. Wang and J. Crowcroft "Qos Routing for Supporting Resource Reservation." *IEEE JSAC*, Sept. 1996.

[9] R. Widyono, "The Design and Evaluation of Routing Algorithms for Real–time Channels." *TR–94–024*, International Computer Science Institute, UC Berkeley.

[10] X. Yuan, "On the Extended Bellman–Ford Algorithm to Solve Two–Constrained Quality of Service Routing Problems." *The Eighth International Conference on Computer Communications and Networks(IC3N'99)*, Boston, Massachusetts, October 1999.