A MULTIPLE QUALITY OF SERVICE ROUTING ALGORITHM FOR PNNI[1]

Hans De Neve and Piet Van Mieghem

Alcatel Corporate Research
Francis Wellesplein 1, B-2018 Antwerp, Belgium
(neveh@rc.bel.alcatel.be, mieghemp@rc.bel.alcatel.be)

Abstract

Quality of Service (QoS) provisioning generally assumes more than one QoS measure which implies that QoS routing can be categorized as an instance of routing subject to multiple constraints. The ATM-forum PNNI standard, in particular the routing part, belongs to the most promising, future-save, QoS-aware, dynamical and hierarchical routing protocols. The PNNI QoS algorithm is vendor specific and not standardized.

Here we propose an attractive multiple QoS routing algorithm, called TAMCRA, standing for Tunable Accuracy Multiple Constraints Routing Algorithm, well suited for PNNI. TAMCRA possesses tunable accuracy (coupled to the running time) via one integer parameter *k* which reflects the number of shortest paths taken into account during computation.

Introduction

In the ATM Forum, PNNI is promoted as the future-safe, powerful protocol that inherits the full QoS capabilities of ATM. Although the current Internet is a best-effort network, serious efforts are spent to offer QoS and some notion of guarantee. Thus, in short, QoS aspects in dynamic, distributed networks deserve today due attention.

A user may ask the network for a transfer of his information subject to requirements for the QoS measures (such as delay, cost, bandwidth, cell loss, etc.) along the path from source to destination. A network topology supporting QoS consists of link metrics vectors with as components the QoS measures. The QoS measure of a path can either be *additive* in which case it is the sum of the QoS measures along the path or it can be the *minimum(maximum)* of the QoS measures along the path. Min(max) QoS measures can be dealt with by omitting all links (and possibly disconnected nodes) which do not satisfy the requested min(max) QoS measure before starting to calculate a path. We call this topology filtering. Additive QoS measures prove to give more difficulties: the problem of calculating a path which is subject to more than one additive constraint is known to be NP-complete [Garey, 1979] and hence, intractable for large networks.

Although the definition of the additive, multiple constraints problem can be found in various publications [Jaffe, 1984; Henig, 1985; Iwata, 1996; Lee, 1995; Vogel, 1996; Wang, 1996] we rephrase it to avoid any misconception.

Problem

Given a graph *G* with *V* nodes and *E* links connecting the nodes, *m* additive link values $l_i(e) > 0$, $i=1,..,m$ characterizing each link and *m* positive values L_i, $i=1,..,m$ serving as constraints, does there exist a path $P(s,k,..,l,d)$ connecting source node *s* with destination node *d* such that $l_i(P)=l_i(s\_k)+...+l_i(l\_d) <= L_i$ for each $i=1,..,m$?

The NP-completeness of this problem was proved only recently by Wang (1996). A problem is NP-complete when there is at least one instance of the problem which cannot be solved in polynomial time. This implies that some multiple constraints problems *can* be solved in polynomial time. Indeed, Jaffe (1984) pointed out that, if there are only two metrics and if one of the metrics has a constant link value, then any two constraints problem can be solved with a polynomial algorithm. However, in most cases NP-complete problems cannot be solved in polynomial time. Therefore, a number of approximative solutions and heuristics have been proposed which return a path within polynomial time without guaranteeing that this path satisfies all constraints.

Existing Solutions

Jaffe (1984) worked out an algorithm to solve a multiple constraints problem with *m*=2 constraints (L_1,L_2). He proposes to replace the two link values of each link by a

[1] Proceedings 1998 IEEE ATM Workshop, May 26-29, Fairfax, VA, USA, pp.324-328

single link value which is a linear combination of the original link values,

$$l(e) = d_1.l_1(e) + d_2.l_2(e) \qquad (1)$$

Running the Dijkstra algorithm on the graph with link values condensated in this way will return a path *P(s,m,...,n,d)* between the source *s* and the destination *d* for which

$$\begin{aligned} l(P) &= \left[d_1.l_1(s\_m) + d_2.l_2(s\_m)\right] + \ldots \\ &\quad + \left[d_1.l_1(n\_d) + d_2.l_2(n\_d)\right] \\ &= d_1.\left[l_1(s\_m) + \ldots + l_1(n\_d)\right] \\ &\quad + d_2.\left[l_2(s\_m) + \ldots + l_2(n\_d)\right] \\ &= d_1.l_1(P) + d_2.l_2(P) \qquad (2) \end{aligned}$$

is minimal. Equation (2) describes a set of equilength lines in a plane, displaying the length ($l_1(P)$, $l_2(P)$) of all possible paths between the source node *s* and the destination node *d*. This is shown in Figure 1.

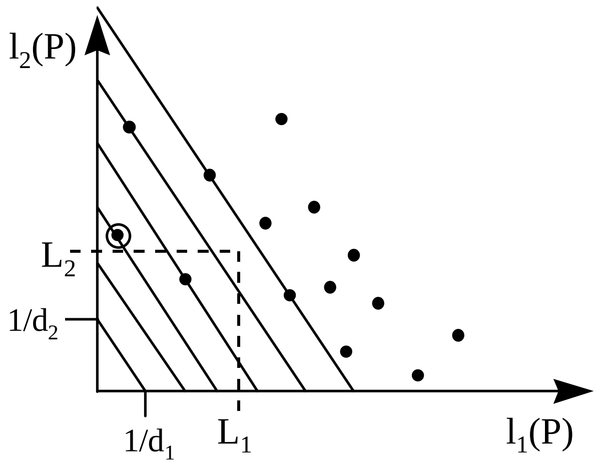

Figure 1: Distribution of paths in the ($l_1(P)$,$l_2(P)$) plane. The solution retrieved by the Dijkstra algorithm is encircled.

Each of the parallel lines $d_1.l_1(P)+d_2.l_2(P) = c$ intersects solutions with equal length *c*. All solutions lying above a certain line have length larger than the ones below or on the line. The shortest path returned by Dijkstra after condensation of the two link values (l_1,l_2) to a single value $l=d_1.l_1 +d_2.l_2$ will be the first solution intersected by a set of parallel lines with slope (d_2/d_1). This is the encircled solution shown in Figure 1 and in this example it lies outside the constraint area. Thus, this path does not satisfy all constraints although it is the shortest path according to the definition of the pathlength in (2).
Jaffe argues that the best value for the slope is given by

$$d_2/d_1 = \sqrt{L_1/L_2} \qquad (3)$$

This implies that in the worst-case scenario, one of the constraints can be violated by at most 100% [Jaffe, 1984]. The worst-case overshoot of the constraints as defined by Jaffe is not what we are most concerned about. If the requested constraints are minimum requirements then, from the point of view of a network environment offering QoS guarantees, every solution outside the constraints is equally bad. We are rather interested in the probability of not finding a path within the constraints when actually such a path does exist.

Non-linear Pathlength

When scanning the solution space with a straight equilength line, the area which is scanned outside the constraint region would be reduced if the slope of the straight equilength lines would be given by

$$d_2/d_1 = L_1/L_2 \qquad (4)$$

instead of the slope (3) proposed by Jaffe. If the slope is given by (4), then always half of the constraint area is scanned before running the risk to select a solution outside the constraint area. This is shown in Figure 2.

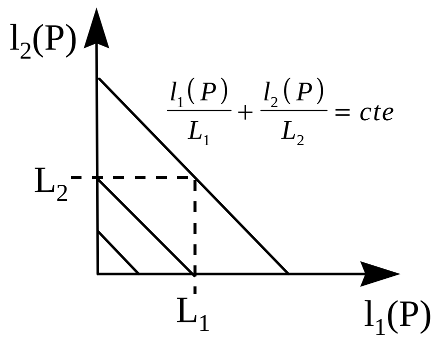

Figure 2: Straight equilength lines

The area which is scanned outside the constraint area could be even further reduced if the straight equilength lines would be replaced by curved equilength lines. Paths are sorted according to curved equilength lines which follow the boundaries of the constraint area if the length *l(P)* of a path *P* from the source node *s* to the destination node *d* is defined as a non-linear combination of the individual pathlengths as in expression (5)

$$l(P) = \sum_{i=1,\ldots,m} \left(\frac{l_i(P)}{L_i}\right)^q \qquad (5)$$

with

$$l_i(P) = \sum_{s\to d} l_i \quad (6)$$

This is shown in Figure 3.

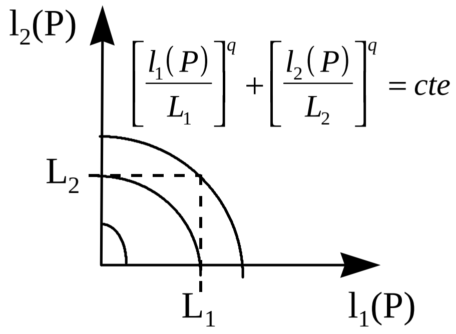
$l_2(P)$
$\left[\frac{l_1(P)}{L_1}\right]^q + \left[\frac{l_2(P)}{L_2}\right]^q = cte$
L_2
L_1
$l_1(P)$

Figure 3: Curved equilength lines

Ideally the equilenght lines should perfectly match the boundaries of the rectangular constraints, scanning the constraint area without ever selecting a solution outside the constraint area, which is achieved when $q\to\infty$. In that case the value of *l(P)* in (5) will be completely dominated by the largest term in the sum (5). This means that the path with the smallest length as defined in (5) for $q\to\infty$ is also the path with the smallest value of

$$l(P) = \max\left(\frac{l_1(P)}{L_1}, \frac{l_2(P)}{L_2}, \ldots, \frac{l_m(P)}{L_m}\right) \quad (7)$$

with

$$l_i(P) = \sum_{s\to d} l_i \quad (8)$$

Although expression (7) has not the same value as expression (5) for $q\to\infty$, the ranking of the paths according to (7) is the same as the ranking according to (5) for $q\to\infty$. Thus expression (7) is equally valid as a definition of the pathlength.

Scanning the constraint area in search for the shortest path according to (7) is shown in Figure 4. No solution outside the constraints is selected before any solution inside the constraints would be found.

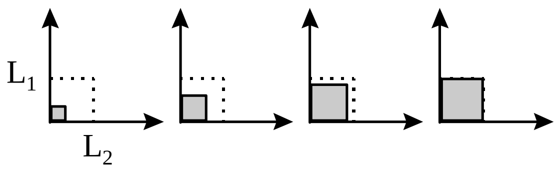

Figure 4: Scanning the constraint area when the length is defined as in (7)

If no path can be found with length

$$l(P) <= 1 \quad (9)$$

the algorithm can stop and decide that there is no path within the constraints.

The problem of finding a path which satisfies a number of constraints is thus "reduced" to the problem of finding the shortest path according to definition of the pathlength in (7).

However, finding the shortest path according to the definition in (7) is not straightforward. We propose an algorithm which is similar to the *k*-shortest path adaptation of the Dijkstra algorithm proposed by Chong et al. (1995) and use expression (7) for the pathlength.

The TAMCRA Algorithm

The principle of TAMCRA is best demonstrated with an example. Suppose we are looking for the shortest path between node 1 and node 4 in Figure 5.

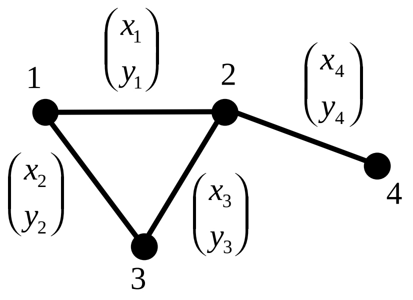
$\begin{pmatrix}x_1\\y_1\end{pmatrix}$
1
2
$\begin{pmatrix}x_4\\y_4\end{pmatrix}$
$\begin{pmatrix}x_2\\y_2\end{pmatrix}$
$\begin{pmatrix}x_3\\y_3\end{pmatrix}$
4
3

Figure 5: Example graph to demonstrate the principle of TAMCRA

If we apply the Dijkstra algorithm, in a first stage we find two possible paths from node 1 to node 2. Both are represented in Figure 6.

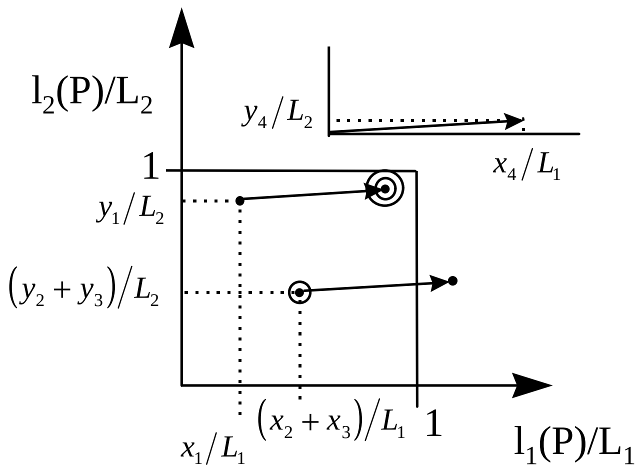

Figure 6: Representation of the two paths between node1 and node 2 in Figure 5. The length vector (x_4,y_4) represents the remaining part to node 4 . The shortest path to node 2 is encircled once, the shortest path to node 4 is encircled twice.

According to the definition of the pathlength in (7) and the positions of the paths in Figure 6, the pathlength for both paths is given by

$$l(1-2) = \max\left(\frac{x_1}{L_1}, \frac{y_1}{L_2}\right) = \frac{y_1}{L_2} \qquad (10)$$

$$l(1-3-2) = \max\left(\frac{x_2+x_3}{L_1}, \frac{y_2+y_3}{L_2}\right) = \frac{x_2+x_3}{L_1} \qquad (11)$$

In this example the values of *x* and *y* have been chosen such that *l*(1-3-2) < *l*(1-2). The Dijkstra algorithm would then conclude that the path 1-3-2-4 is the shortest path between node 1 and node 4. However, the weight vector of the link from node 2 to node 4 has been chosen such that *l*(1-3-2-4) > *l*(1-2-4). This can be seen in Figure 6 because the path 1-3-2-4 lies outside the constraint area while the other path lies within the constraint area. Thus, with the definition of the pathlength according to (7) subsections of shortest paths are not necessarily shortest paths themselves and the Dijkstra algorithm will sometimes fail to find the shortest path. Therefore, in some cases more than one path between the source node and some intermediate node needs to be stored. This is done in TAMCRA via a *k*-shortest path implementation of the Dijkstra algorithm which stores *k* paths per node instead of only one [Chong, 1995].

Non-dominated Paths

TAMCRA does not store every path. Imagine that the two paths from Figure 6 are positioned as in Figure 7.

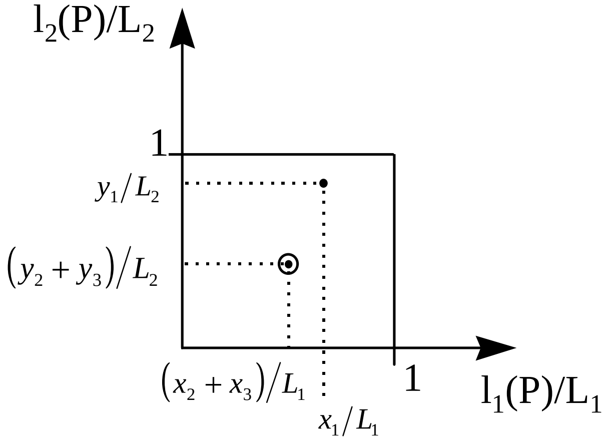

Figure 7: Two paths to the same node, the shortest path is encircled once.

If

$$\frac{x_1}{L_1} > \frac{x_2+x_3}{L_1} \quad \text{and} \quad \frac{y_1}{L_2} > \frac{y_2+y_3}{L_2} \qquad (12)$$

then also

$$\frac{x_1+x_4}{L_1} > \frac{x_2+x_3+x_4}{L_1} \quad \text{and}$$

$$\frac{y_1+y_4}{L_2} > \frac{y_2+y_3+y_4}{L_2} \qquad (13)$$

for any value of x_4, y_4. Thus, we are sure that the shortest path between source and destination will not make use of the path 1-2 and this path should not be stored. Using the terminology of Henig (1985), path 1-2 is dominated by path 1-3-2 and only non-dominated paths can lead to the shortest path.

The time-complexity of the TAMCRA algorithm is the time-complexity of a *k*-shortest path algorithm with the extra complication of verifying whether a path is not dominated by another path. This leads to the following expression for the time-complexity of the TAMCRA. algorithm:

$$O\left(kV\log(kV) + k^3 mE\right) \qquad (14)$$

In the simulations, the calculation time of TAMCRA was found to increase only linearly with *k*.

Results

We measured the probability that a non-shortest path is found as a function of the parameter *k*. This is a worst-case probability of missing a path which lies within the constraints: if there are more paths which satisfy the constraints then missing the shortest path does not necessarily mean that all paths within the constraints are missed.

The probability of missing the shortest path is estimated by calculating for each source-destination pair in a random graph, the value of *k* which is needed to find the shortest path. Figure 8 shows the result for a Waxman graph [Calvert, 1997] with 100 nodes, approximately 200 links and two link measures per link. Each link is bi-directional with the same characteristics in both directions. For each metric, the link weights are uniformly distributed random numbers between 0 and 1. This is a worst-case situation in the sense that any other scheme to attribute weights to the links has resulted in a higher probability to find the shortest path with a given value of *k*.

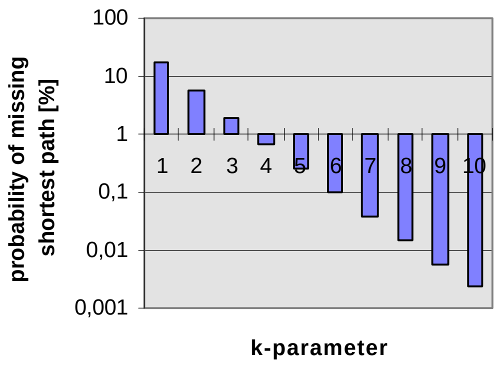

Figure 8: Probability of missing the shortest path for Waxman graphs with 100 nodes and two measures per link

If TAMCRA is applied with *k*=1, the probability of missing the shortest path in the Waxman graph with 100 nodes is about 17%. This means that for 83% of all source-destination pairs, the two-constraint problem is solved with a Dijkstra-like complexity. The probability of missing the shortest path rapidly decreases as *k* is increased: with *k*=4 the probability has dropped below 1%, which means that the TAMCRA algorithm with *k*=4 perfectly solves the two-constraints problem in more than 99% of all cases.

We have also monitored the time needed for one node to calculate the shortest path to all other nodes in the graph as a function of *k*. The calculation is done on a Sun SPARC Ultra Workstation and Figure 9 displays the result in absolute time units. As a reference we added the running time of the Dijkstra algorithm (*m*=1, *k*=1).

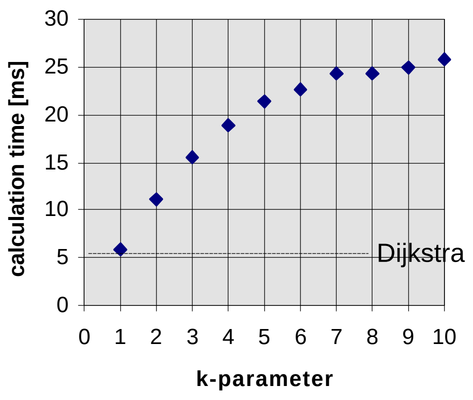

Figure 9: Calculation time of the TAMCRA algorithm for Waxman graphs with 100 nodes and two measures per link

To solve a two-constraint problem with an accuracy of 99% in a graph with 100 nodes TAMCRA needs *k*=4. The time complexity of TAMCRA with *k*=4 is about 3.7 times the time-complexity of the Dijkstra algorithm.

For the Waxman graphs with 100 nodes and two constraints, on the average only three to four paths per node are non-dominated. This means that TAMCRA does not store more than these three to four paths even if it is allowed to store more. This is why the observed calculation time saturates as the value of *k* gets larger.

Conclusions

The TAMCRA algorithm demonstrates that the routing problem in PNNI of calculating a path which has to satisfy more than one additive constraint can be solved *accurately* within polynomial time-bounds because the probability of missing the shortest path decreases very fast with increasing *k*. The NP-completeness resides in the fact that the value of *k* which is needed to solve the problem *exactly* is a (polynomial) function of the constraints. This also means that beyond a certain value of *k*, TAMCRA can solve multiple constraints problems exactly [De Neve, 1998].

References

Calvert K., M.Doar, E.Zegura, 1997, "Modelling Internet Topology", IEEE Comm. Magazine, pp.160-163

Chong E.I., S.Maddila, S.Morley, 1995, "On Finding Single-Source Single-Destination k Shortest Paths", J.Computing and Information, special issue ICCI'95, pp.40-47

De Neve H., P.Van Mieghem, 1998, "Tunable Accuracy Multiple Constraints Routing Algorithm: TAMCRA", submitted to ACM SIGCOMM '98, Vancouver Canada.

Garey M.R. and D.S.Johnson, 1979, *Computers and Intractability, A Guide to the Theory of NP-Completeness*, Freeman, San Francisco

Henig M.I, 1985, "The shortest path problem with two objective functions", European Journal of Operational Research 25, pp.281-291

Iwata A, R.Izmailov, D-S.Lee, B.Sengupta, G.Ramamurthy, and H.Suzuki, 1996, "ATM Routing Algorithms with Multiple QOS Requirements for Multimedia Internet working", IEICE Trans. Com., vol.E79-B, pp.999-1007

Jaffe J.M, 1984, "Algorithms for Finding Paths with Multiple Constraints", Networks, vol.14, pp.95-116

Lee W, M.Hluchyi, and P.Humblet, 1995, "Routing Subject to Quality of Service Constraints in Integrated Communication Networks", IEEE Network July/August 1995, pp.46-55

Vogel R., R.Herrtwich, W.Kalfa, H.Wittig, and L.Wolf, 1996, "QoS-Based Routing of Multimedia Streams in Computer Networks", IEEE J. on selected areas in communications, vol.14, no.7, pp.1235-1244

Wang Z and J.Crowcroft, 1996, "Quality-of-Service Routing for Supporting Multimedia Applications", IEEE J. on

Selected Areas in Communications, vol.14, pp.1228- 1234