A New Modified Algorithm for Multi Constraint Routing

Yavor Tomov[1] and Georgi Iliev[2]

Abstract – **In modern telecommunications the necessity of supporting a large number of broadband services requires a guaranteed QoS routing. The basic task of routing is to find a path from source to destination. This path should satisfy these requirements. When the needed QoS depends on more than one parameter, the problem is known as Multi Constraint problem (MCP). Our goal in this article is to propose a new modified heuristic algorithm able to solve the problem.**

Keywords – **QoS, MCP, Routing.**

I. Introduction

To support broadband services such as the real time services, the QoS routing is the major problem. Routing has to deal two basic task – to support and update the information of the network and to find a path from source to destination. In cases when an application has no special requirements for the parameters (bandwidth, delay, jitter, packet loss) which define the QoS, finding the path from source to destination is known as The Shortest Path Problem. In other cases when two or more parameters have to be guaranteed from source to destination the problem is known as Multi Constraint Problem (MCP). MCP is NP hard [1] Depending on these parameters they may be classified as additive, multiplicative or concave. The paper is about the additive parameters, where the weight of this parameter (end to end) is equal to the sum of the weights of all links on the path. A path which satisfies all constraints is called a feasible path. There exist two basic routing strategies [2]: source routing and distributed routing. The source routing calculates the entire path locally in the source node. The distributed routing calculates the path in the intermediate note between the source and the destination.

The algorithms able to solve the MCP can generally be classified as heuristic and exact. The heuristic algorithms do not guarantee finding a feasible path(if it exists), while the exact algorithms do it. For this reason they are not applicable in practice. They are usually used to evaluate heuristic algorithms.

Our goal in this paper is to propose a new heuristic, modified, distributed routing algorithm, able to solve the MCP with two constraints.

Notation

Each real network can be presented as a graph G(V, E) where V is the set of nodes and E is the set of links. The nodes are routers , while the links are physical or logical connections between them. Each link $e \in E$ is associated with two – dimensional link vector $\vec{w}(w_1, w_2)$. The paths are noted as p, the source node as s, the destination node as t, any intermediate node as u. A path is feasible if (1) and (2) are correct.

$$w_1(p) = \sum_{e \in p} w_1 \le C_1 \qquad (1)$$

$$w_2(p) = \sum_{e \in p} w_1 \le C_2 \qquad (2)$$

where C_1 and C_2 are the constraints.

II. Related work

In the proposed algorithm, we present the two weights on each link as a linear combination. There exist many algorithms using this technique. They are known as algorithms with mixed metrics. Some of these algorithms use linear path length [3][4][5][6][7], others use non - linear path length[8][9]. The main advantage of linear path length algorithms is that they can implement Dijkstra's algorithm [10], while their major drawback is that they can return a path, which is not feasible (path outside the feasible region).

The major advantage of the algorithms that use the non - linear path length is that they can scan the feasible region precisely. Their drawback however is that subsections of shortest path are not always shortest paths. As a result of using Dijkstra's algorithm they are likely to fail. Fig.1 shows the way the two types of algorithms scan the feasible region. Fig 2 shows that the algorithm with linear path length works, Fig 3.shows that the algorithm fails.

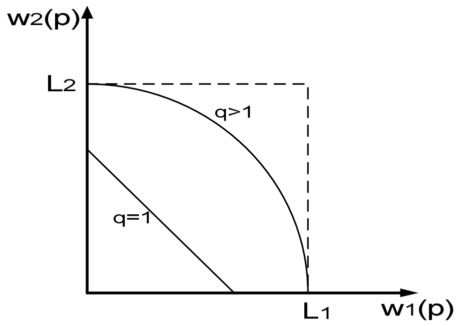

Fig. 1. Two types of algorithms scan the feasible region [3][8][9]

[1]Yavor Tomov is with the Faculty of Telecommunications at Technical University of Sofia, 8 Kl. Ohridski Blvd, Sofia 1000, Bulgaria, E-mail: qvor_tomov@abv.bg.

[2]Georgi Iliev is with the Faculty of Telecommunications at Technical University of Sofia, 8 Kl. Ohridski Blvd, Sofia 1000, Bulgaria. E-mail gli@tu-sofia.bg

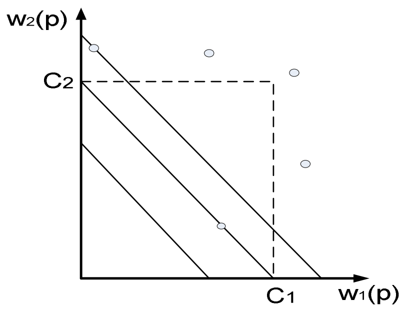

Fig. 2. The algorithm works [3]

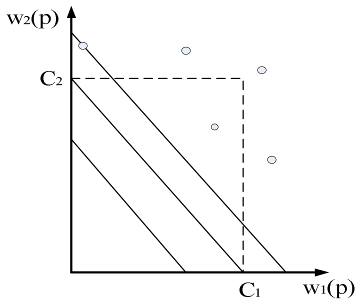

Fig. 3. The algorithm fails [3]

The first who proposed each link to be presented as linear combination was Jaffe[3].

$$w(e) = d_1 w_1 + d_2 w_2 \qquad (3)$$

Where d_1 and d_2 are Lagrange multipliers.

Iwata [4] suggests a heuristic algorithm to solve MCP This algorithm computes the shortest path, based on the first parameter and checks whether all constraints are satisfied. If they are - the algorithm returns this path. If they are not - the algorithm finds the shortest path with respect to the second parameter and this is repeated until a feasible path is found.
In [5] the author proposed to reduce the search space increasing in this way the probability to find a solution.
Feng [6] proposed an algorithm based on [5],reducing the search space and implementing the k-th shortest path, suggested by Chong[11].
Based on these algorithms we worked out a heuristic algorithm able to deal with MCP in cases with two constraints.

III. Our algorithm

In this Section, we first prove an important theorem.
Theorem: If we use Dijkstra's algorithm to minimize the linear cost function on a graph that contains at least one feasible path, the algorithm returns a path where at least one $w(p) \le C$.

We assume that there is at least one feasible path p^* Using Dijkstra's algorithm, it returns the path p that minimizes the following linear cost function

$$g(p) = w_1 + kw_2 \qquad (4)$$

where k is positive multiplier. This path however is not feasible

$$g(p) < g(p^*) \qquad (5)$$

$$w_1(p) + kw_2(p) < w_1(p^*) + kw_2(p^*) \qquad (6)$$

Since p* is feasible path it follows

$$w_1(p) \le C_1 \text{ and } w_2(p) \le C_2 \qquad (7)$$

From (6) and (7) follow that at least one

$$w(p) \le C \qquad (8)$$

With (8) we prove this basic theorem.

Our basic idea in this algorithm is to minimize two linear cost functions for each intermediate node – one from u to s and the other from u to t. First the algorithm finds the shortest paths from s to t with respect to w_1 and w_2. Then the algorithm defines k multipliers, as follows:

$$k_1 = \frac{C_1 - w_1(p_1)}{C_2 - w_2(p_2)} \qquad (9)$$

The node s shares this value with all other nodes. Each of the intermediate nodes calculates two paths-one from u to s, and another – from u to t.These paths are calculated by the nodes using Dijkstra's algorithm to minimize the cost function

$$g(p) = w_1 + k_1 w_2 \qquad (10)$$

If some of the nodes finds a feasible path, it returns it to s, if not according to our theorem each path contains not more than one $w > C$. In this case each of the nodes checks whether w_1 or w_2 does not satisfy the constraint. If it is w_1 - then the node decreases k_1 to k_2. If w_2 does not satisfy the constraint – the node increases k_1 to k_2.
In the next step each node computes the shortest path from u to s with respect to (10),and the path from u to t with respect to

$$g(p) = w_1 + k_2 w_2 \qquad (11)$$

If the path

$$p(s \to t) = p(u \to s) + p(u \to t) \; (10)$$

satisfies the constraint, the node returns this path. If not – the node changes the position of the two cost functions.

PSEUDO CODE

```
1. Dijkstra $w_1(s \to t) \to w_1(p_1)$
```

```
2.  IF $w_1(p_1) \le C_1$ AND $w_2(p_1) \le C_2$
3.  RETURN path
4.  END IF
5.   Dijkstra $w_2(s \to t) \to w_2(p_2)$
6.  IF $w_1(p_2) \le C_1$ AND $w_2(p_2) \le C_2$
7.  RETURN path
8.  END IF
9.   IF $w_1(p_1) > C_1$ OR $w_2(p_2) > C_2$
10.  No Solution
11. . END IF
12.  $k_1 = \frac{C_1 - w_1(p_1)}{C_2 - w_2(p_2)}$
13.  FOR each link
14.  $w(e) = w_1 + k_1 w_2$
15.  END FOR
16.  Dijkstra $p(s \to t)$
17.  IF $w_1(p) \le C_1$ AND $w_2(p) \le C_2$
18. RETURN path
19. END IF
20. FOR each node $u$
21. Dijkstra $p(u \to t, k_1)$ ;
22. Dijkstra $p(s \to u, k_1)$
23.  $p(s \to t) = p(s \to u) + p(u \to t)$
24.   IF $w_1(p) \le C_1$ AND $w_2(p) \le C_2$
25.    RETURN path
26.   ELSE
27.     IF $w_1(p) \le C_1$ AND $w_2(p) > C_2$
28.      $k_2 > k_1$
29.      DO
30.        Dijkstra $p(u \to s, k_2)$ ,
31. Dijkstra $p(u \to t, k_1)$
32.     $p(s \to t) = p(u \to s, k_2) + p(u \to t, k_1)$
33.       UNTIL $w_2(p) \le C_2$
34.          IF $w_1(p) \le C_1$ AND $w_2(p) \le C_2$
35.              RETURN path
36.            ELSE
37.              Dijkstra $p(u \to s, k_1)$
38. Dijkstra $p(u \to t, k2)$
39.     $p(s \to t) = p(u \to s, k_1) + p(u \to t, k_2)$
40.              IF $w_1(p) \le C_1$ AND $w_2(p) \le C_2$
41.                   RETURN path
42.                END IF
43.              END IF
44.          END IF
45.        END IF
46. IF $w_1(p) > C_1$ AND $w_2(p) \le C_2$
47. $k_2 < k_1$
48.        DO
49.          Dijkstra $p(u \to s, k_2)$ ,
50. Dijkstra $p(u \to t, k_1)$
51.     $p(s \to t) = p(u \to s, k_2) + p(u \to t, k_1)$
52.        UNTIL $w_2(p) \le C_2$
53.           IF $w_1(p) \le C_1$ AND $w_2(p) \le C_2$
54.              RETURN path
55.            ELSE
56.              Dijkstra $p(u \to s, k_1)$
57.             ,Dijkstra $p(u \to t, k2)$
58.     $p(s \to t) = p(u \to s, k_1) + p(u \to t, k_2)$
59.              IF $w_1(p) \le C_1$ AND $w_2(p) \le C_2$
60.                   RETURN path
61.                END IF
62.              END IF
63.           END IF
64.        END IF
65. END FOR
```

From line 1 to line 8 we implement Iwata's algorithm. On line 12 the algorithm calculates k_1 based on LARAC [5] algorithm. From line 20 to the last line the algorithm contains our basic idea which we exposed above.

EXAMPLE

Let we have a graph with six nodes. Each link is associated with two weights - w_1 and w_2. Our task is to find a path from source A to destination F with the given constraints $C_1 = 5$ and $C_2 = 10$ Fig. 4. There are four paths from A to F: ABDF=6, 3; ACEF=4, 11; ABCEF=5, 10; ACBDE=9, 6. If we apply only one linear cost function (like in most algorithms) the algorithm returns the path ABDF or the path ACEF(according to the choice of k)as solutions.

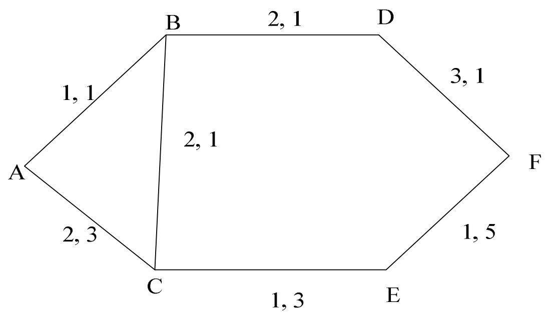

Fig. 4. Graph with two weights on each link

However these solutions are not correct.
Applying our algorithm it will first find the shortest paths from s to t with respect to w_1 and w_2 These paths are ACEF and ABDF. Both paths however are not solution. Next the algorithm will calculate $k = \frac{1}{7}$ based on (9). It will find again the shortest path based on (4). This path is ACEF, but it is not a solution either. In the next step the nodes C and B will calculate the path p based on (12), by $k = \frac{1}{7}$. For node B the path will be ABDF while for node C the path will be ACEF. The two nodes will check which one (w_1 or w_2) does not satisfy the constraint. Node B will establish that this is w_1, while for node C it is w_2. Then the node will decrease k and when $k \le \frac{1}{8}$ with respect to the path from B to F, it will find the path BCEF. The node will return the path ABCEF which is the feasible path. The procedure is the same concerning node C. When $k \ge \frac{11}{10}$ for the path from C to A the node will return the path ABCEF.

IV. Conclusion

We have created an algorithm that is based on five concepts:

1. The algorithm is distributive
2. Iwata's algorithm is implemented
3. Reduced search space[5]
4. Look – ahead approach[12][13]
5. Two linear functions for each intermediate node

In some algorithms with linear path length[6],as well as in other with non linear path length[8][9], a solution is searched applying the concept of k – th shortest path. This however leads to considerable increase in complexity of the algorithm.
We suggest another approach. Each of the intermediate nodes explores a set of paths that could be feasible. This idea is similar to the idea proposed by Korkmaz in his algorithm [14]. He uses one linear and one non linear function, while we use two linear function.
In the example looked through it becomes clear that Iwata's and LARAC algorithms are not able to solve the problem of finding the feasible path, while our algorithm finds it successfully

.Acknowledgement

For the acknowledgement use the unnumbered section layout.

References

1. M.R. Garey and D.S. Johnson. Computers and Intractability: A guide to the Theory of NP-Completeness. ISBN 0-7167-1044-7. W.H. Freeman and Company,San Francisco, 1979
2. S. Chen, K. Nahrstedt, An overview of quality of service routing for next-generation high-speed networks: problems and solutions, IEEE Networks 12 (6) (1998) 64-79
3. J.M. Jaffe. Algorithms for finding paths with multiple constraints. Networks,14:95—116, 1984
4. A. Iwata, R. Izmailov, D. Lee, B. Sengupta, G.Ramamurthy, and H. Suzuki. ATM routing algorithms with multiple QoS requirements for multimedia internetworking. IEICE Trans. Commun., E79-B(8):999—1007, Aug. 1996
5. A.Juttner, B. Szviatovszki, I. Mecs, Z. Rajko, "Lagrange relaxation based method for the QoS routing problem", Proceedings of the INFOCOM 2001 Conference, vol. 2, IEEE, 2001, pp. 859–868
6. Gang Feng, " Exact algorithms for multi-constrained QoS routing ," International Conference on Computer, Communication and Control Technologies (CCCT'03), Orlando, USA, Vol. 2, pp. 340 – 345, July 31 – Aug 2, 2003
7. P.Khadivi, S.Samavi, T.D.Todd, "Multi-constraint QoS routing using a new single mixed metrics", Journal of Network and Computer Applications, Vol. 31, Issue 4, pp 656-676, Nov. 2008.
8. H. De Neve and P. Van Mieghem. TAMCRA: a tunable accuracy multiple constraints routing algorithm. Computer Communications, 23:667—679, 2000.
9. Van Mieghem P. and F. A. Kuipers, "Concepts of Exact Quality of Service Algorithms," IEEE/ACM Trans. Net Vol. 12, pp. 851 - 862, 2004
10. E.W. Dijkstra. A note on two problems in connexion with graphs. Numerische Mathematik, (1):269—271, 1959.
11. E.I. Chong, S. Maddila, and S. Morley. On finding single-source single-destination k shortest paths. Journal of Computing and Information, special issue ICCI'95, pages 40—47, July 1995.
12. S. Lin, "Computer solutions of the traveling salesman problem," Bell Syst. Tech. J., vol. 44, no. 10, pp. 2245–2269, 1965.
13. A. Newell and G. Ernst, "The search for generality," in Proc. IFIP Congr., vol. 1, 1965, pp. 17–24.
14. T. Korkmaz, M. Krunz, "Multi-constrained optimal path selection", Proceedings of the INFOCOM 2001 Conference, vol. 2, IEEE, Anchorage, Alaska, 2001, pp. 834–843.