Quality of Service Routing: Heuristics and Approximation Schemes with a Comparative Evaluation

Ravi Ravindran [1], K.Thulasiraman[2], Anindya Das[3], Kaiyuan Huang[1] ,Gang Luo[1] and Guoliang Xue[4]
1.Nortel Networks, Ottawa, 2. University of Oklahoma, Norman, 3.Universityof Western Ontario, London and 4.Arizona State Univeristy, Tempe

ABSTRACT

In this paper we consider a class of quality of service routing problems that can be formulated as the bounded delay minimum cost path problem. This is an NP-hard problem [2]. So heuristic approaches have been presented in the literature. In a recent work [11] we extensively evaluated the performance of a heuristic called the LHWHM algorithm originally presented in [7]. We have found the LHWHM algorithm (which is based on Dijkstra's minimum cost path algorithm) to be very effective for implementation in a real network environment. In this paper we present a new heuristic which is based on the Bellman-Ford-Moore algorithm for the min-cost path problem. Unlike the LHWHM algorithm this algorithm is also suitable for distributed implementations. We discuss several issues pertaining to an efficient implementation of this algorithm and compare its features with the LHWHM [7] algorithm. We prove that this heuristic produces a feasible path satisfying the delay constraint whenever such a path exists. We conclude with an experimental evaluation of the new heuristic in comparison with the LHWHM algorithm , a recent Lagrangian-Relaxation based heuristic [9] and the approximation algorithm due to Hassin [10] by running them on a large number of graphs. We believe that the LHWHM algorithm and our new BFM based heuristic are serious candidates for implementation in real network environments.

1. INTRODUCTION

In recent years, there has been considerable emphasis on designing communication protocols which deliver certain performance guarantees. This has been the result of an explosive growth in high bandwidith real time applications which require stringent QoS guarantees. In this paper we study routing algorithms that guarantee such QoS requirements. Specifically we will be studying the routing algorithms which aim at minimizing the cost of a path from a source node to a destination node subject to the total delay of the path being within a certain limit. This discussion is also applicable to paths involving other link parameters which are additive (along the path). Several papers discussing different aspects of this problem have appeared in the literature [1]-[7]. In designing routing protocols two min-cost path algorithms play an important role. They are: Dijkstra's algorithm and Bellman-Ford-Moore (BFM) algorithm [8]. Whereas Dijkstra's algorithm is inherently sequential in nature, the BFM algorithm is amenable for an elegant distributed implementation. The problem we are interested in is the design of routing protocols that satisfy multiple constraints: minimizing cost subject to the delay requirements. This problem, also known as the bounded delay minimum cost path problem, has been shown to be NP-complete [2]. So in the literature heuristics approaches to get good approximate solutions have been presented [1-7]. In a recent paper Luo, Huang,Wang, Hobbs and Munter [7] presented a heuristic for the bounded-delay minimum cost path problem. In [11] an extensive study of this heuristic has been made. Also, several variations of this heuristic have been discussed in [11]. These studies indicate that the LHWHM heuristic has several features which make it ideal for implementation in a real network environment. In this paper, we develop a new heuristic based on the BFM algorithm. Several issues arising in the implementation of our new heuristic are discussed. This algorithm is also amenable for distributed implementation. We present an experimental evaluation of these heuristics and compare their performance with a recent heuristic called *Lagrangian Relaxation based Aggregated Cost* (LARAC) heuristic which is based on the Lagrangian Relaxation method of operations research [9]. We also give a comparison with the performance of Hassin's approximation algorithm [10].

2. BOUNDED DELAY MIN-COST PATH PROBLEM (BDMCP)

Consider a point-to-point communication network represented as a directed graph $N = (V, E)$, where V is the set of nodes and E is the set of links in N. A link directed from node u to node v is denoted by $e = (u , v)$. Each link e is associated with two non-negative real numbers, cost $c(e)$ and delay $d(e)$. The link cost $c(e)$ may be either a monetary cost or some measure of the link's utilization. The link delay $d(e)$ is a measure of delay a packet experiences when traversing the link e. We also specify two nodes, s and t, as the source and the destination nodes, respectively. An undirected network may be viewed as a directed network with each link $e = (u, v)$ replaced by two oppositely oriented links $e_1 = (u, v)$ and $e_2 = (v, u)$. In this case $c(e_1) = c(e_2)$ and $d(e_1) = d(e_2)$. If $e = (u, v)$, then $c(e)$ and $d(e)$ are also denoted as $c_{u,v}$ and $d_{u,v}$ respectively. We define a path P from node v_o to node v_k as an alternating sequence of distinct nodes and links such that $P(v_0, v_k) = v_0 , e_1 , v_1, e_2 ..... , e_k, v_k$ where $e_i = (v_{i-1}, v_i) \in E$,
for $1 \le i \le k$. The cost $c(P)$ and delay $d(P)$ of the path P_are define as:

$$c(P) = \sum_{e \in P} c(e) \text{ and}$$

$$d(P) = \sum_{e \in P} d(e).$$

Suppose we are given a real number T which serves as a measure of the maximum allowable delay on any s-t path in N, then we

0-7803-7448-7/02/$17.00 ©2002 IEEE

call an *s-t* path *P* feasible if $d(P) \leq T$ .The **Bounded Delay Minimum Cost Path Problem (BDMCP)** is to find a feasible *s-t* path which has the smallest cost. For example, in the network *N* in Fig.1 with delay constraint of $T= 5$, the shortest *s-t* path is the path P_0 ={s, 2, 4, t}. But this path does not satisfy the delay constraint of $T= 5$. The *s-t* paths P_1={s, 1, 4, t} and P_2 = {s, 2, 3, *t*} are feasible with cost 5 and cost 8, respectively. P_1 is the feasible min-cost path.

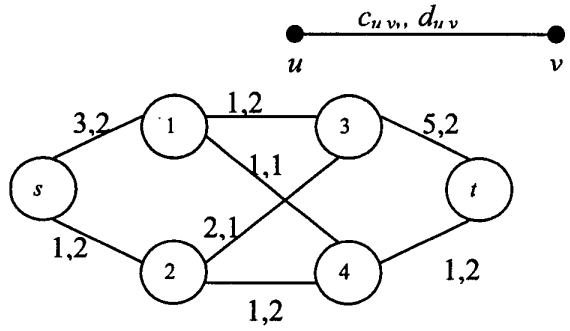

Fig. 1

3. BELLMAN-FORD-MOORE BASED HEURISTIC FOR THE BDMCP PROBLEM

We present in this section a new heuristic based on the BFM algorithm to be called the BFM- BDMCP algorithm. We first present the BFM algorithm. We then present our new heuristic for the BDMCP problem. This will be followed by a discussion of certain fundamental characteristics of this heuristic which distinguish it from the LHWHM algorithm and also certain fundamental implementation issues which arise because of these unique features.

3.1 The BFM Algorithm

As in Dijkstra's algorithm, the BFM algorithm associates with each node *u* two variables CLABEL(*u*) and PRED(*u*). Initially CLABEL(*s*) =0, CLABEL(*u*) = ∞ ,for all $u \in$ V and $u \neq s$ and PRED(*u*) =*u* for all $u \in$ V.

The *general step* in the algorithm is :
Pick any edge (*u,v*) such that CLABEL(*u*) ≠∞ and CLABEL(*v*) > CLABEL(*u*) + $c_{u,v}$.
Set CLABEL(*v*) = CLABEL(*u*) + $c_{u,v}$ and PRED(*v*) =*u*.
If no such edge is available, the BFM algorithm terminates.
At termination CLABEL(*t*) gives the cost of a minimum cost *s-t* path. This path can be traced starting at PRED(*t*) and working backwards towards the source node *s*.
We next present an efficient *O(mn)* implementation of the BFM algorithm [8]. In the following discussion scanning a node *u* means examining all the edges (*u,v*) and labeling the neighbor nodes of *u*, if possible. A sweep of the BFM algorithm refers to the process of scanning all the nodes as 1,2,3,.....*n* in that order. It can be shown that after performing *n* sweeps the BFM algorithm will terminate, resulting in the time complexity of O(*mn*) .The above implementation of the BFM algorithm permits two choices which are presented below.

Version 1 *(Asynchronous)*
While scanning a node *u*, use the current value of CLABEL(*u*) to label the neighbors of node *u*.
Version 2 ***(Synchronous)***
While scanning node *u* during a sweep, use the value of CLABEL(*u*) at the end of the previous sweep for updating the labels of the neighbors of node *u*.
Asynchronous distributed implementation of BFM algorithm uses *Version 1* and synchronous distributed implementation distributed implementation uses *Version 2* .So these two versions will be referred to asynchronous and synchronous versions, respectively. Note that there is no significant difference between these two versions.

3.2 BFM-BDMCP Heuristic

Our new heuristic for the BDMCP problem is based on the BFM algorithm. As in the case of the LHWHM algorithm, we first compute D(*v*) for each node *v*, where D(*v*) is the minimum delay of any *v-t* path. Again each node *v* is associated with the variables CLABEL(*v*), DLABEL(*v*) and PRED(*v*). Initialization of the BFM-BDMCP algorithm is as follows.
CLABEL(*s*) =0, CLABEL(*u*) = ∞ , for all $u \in$ V and $u \neq s$, and DLABEL(*u*) =0 for all $u \in V$
For reasons which will be made clear later, we are not at this point specifying the initialization of PRED(*v*) values or the mechanism for the update of these values as the algorithm proceeds. Scanning a node *u* involves the following:
For each neighbor *v* of *u* do:
If DLABEL(*u*) + $d_{u,v}$ + D(*v*) ≤ *T* and CLABEL(*v*) > CLABEL(*u*) + $c_{u,v}$ then set
CLABEL(*v*) = CLABEL(*u*) + $c_{u,v}$.

A formal presentation of the BFM-BDMCP algorithm is as follows:
BFM-BDMCP Algorithm
1. Initialize the variables CLABEL(*v*), and DLABEL(*v*) for all nodes $v \in V$.
2. Perform a Sweep.

If no node CLABEL value is updated during a sweep, STOP (the value of CLABEL(t) gives the cost of the final feasible *s-t* path and is an approximate solution to the BDMCP problem). Otherwise repeat Step 2. We can use the synchronous version or the asynchronous version while implementing the Step 2. We next proceed to highlight certain characteristics of the BFM-BDMCP algorithm.
Suppose while labeling from node *u*, the CLABEL(*v*) of neighbor node *v* is updated . We can view this as "node *u* initiating a wave to node *v*". Basically this means that node *u* has extended a current *s-u* path to an *s-v* path. In the case of the LHWHM algorithm, during an iteration only one node (the most recently permanently labeled node) is scanned. Thus at most n new waves (equivalently, paths) are initiated. And only one of these waves gets chosen for propagation of waves in the subsequent iteration. In fact at most n waves will get a chance to propagate new waves during the entire algorithm. On the other hand, in the case of the BFM-BDMCP algorithm, iteration

corresponds to a sweep, that is, the scanning of all the nodes of the network. This means that during a sweep a number of waves are initiated. Therefore a large number of paths become considered for further extensions to feasible $s-t$ paths. This is because every node whose CLABEL gets updated during a sweep gets a chance to initiate a wave (a new path) for exploration and does not have to wait until it becomes permanently labeled as in the case of the LHWHM algorithm. It is for this reason that we expect the BFM-BDMCP algorithm to outperform the LHWHM algorithm. This is illustrated further next.

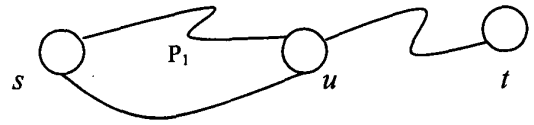

Fig. 2

Consider Fig. 2. Suppose node u is updated along path P_1 with new labels
CLABEL(u) = x_1 and DLABEL(u) =y_1 .
This causes a new wave, say WAVE1, to be initiated by u. Suppose at a subsequent sweep node u is updated along path P_2 with new CALBEL(u) = x_2 and DLABEL(u) = y_2. This causes a new wave, say, WAVE2, to be initiated by node u. Clearly $x_2 < x_1$. Suppose y_2 is much larger than y_1, then WAVE2 may detect in a subsequent sweep that it cannot extend itself to a feasible s-t path. On the other hand WAVE1, because of the lower value of y_1 may be able to reach t and label it, thereby finding a feasible s-t path. In the case of the LHWHM algorithm WAVE1 would not even have been initiated, thereby losing an opportunity to discover a feasible s-t path.
We now examine the impact of the above scenario on the tracing of the s-t path using the PRED values, starting from node t and moving backwards towards s.
Suppose we initialize PRED(i)= i for all i $\in$ V and set PRED(i) = j whenever i gets updated by while scanning node j. If we use this scheme for updating PRED values, then in the scenario (Fig 2) we have just considered PRED(u) will be first set to k (the node preceding u in the path P_1) when u is labeled by k along path P_1. In a subsequent step PRED(u) will again be reset to w, the node preceding u in P_2. WAVE1 reaches t and labels it. But WAVE2 does not reach t. So when we trace the s-t path from t we reach node u and find node w as a predecessor and not k which was the cause for initiating WAVE1 and labeling t. Thus using the scheme used in storing and updating PRED values as in LHWHM algorithm will not help in correctly identifying the final s-t path.
So we now propose an alternate scheme to remedy this problem. In our scheme PRED(v) is a list of entries. Each entry in the list has three components (x, y, z). Initially PRED(v) for every node v has only one entry, namely (v,0,0). When the labels of a node v are updated by node u then an entry (x,y,z) is added to the list PRED(v) where x ,y and z are defined as follows.
x = u, y = CLABEL(u), and
z = DLABEL(u).

With this initialization of PRED(v) lists, our scheme for tracing the find s-t path is :

1. Let the final entry in PRED(t) be (x,y,z). Note that t received its labels from node x, and y = CLABEL(x) and z = DLABEL(x) are the values of these variables, when x labeled t. Also note that x is the predecessor of t in the final s-t path.
2. Search in the PRED(x) list for an entry (x', y', z') such that y = y' + $c_{x',x}$ and z = z' + $d_{x',x}$ Then x' is the predecessor of x in the final s-t path. Next set x = x', y = y', z = z'
3. Repeat step 2 until $x = s$.

It would appear that to implement the above scheme for tracing the s-t path we need to store a large number of entries in PRED lists. This will be the case only if we keep adding an entry to PRED(v) list every time v is updated. This is not necessary because we need to keep only those entries which cause node v to propagate new waves in a sweep. Since there will be at most n sweeps, the size of the PRED array would be of $O(n)$. The proof that the algorithm guarantees to find a feasible path if one exists has been given in [11]. Also the time complexity of this algorithm is $O(mn)$.

4. SIMULATION AND PERFORMANCE EVALUATION

In this section we present an evaluation of the performance of the LHWHM and the BFM-BDMCP algorithms and compare them with the performance of the LARAC algorithm [9]. We only report results for the regular graphs $H_{k,n}$, proposed by Harary [8]. Here k refers to the vertex degree and n refers to the number of vertices.
The edge costs and delays are selected as follows:
Edge costs are randomly generated in the range 1 to 50 and delays are assigned values as follows: $d_{ij} = 50 - c_{i,j}$.
The motivation for this choice is to test the heuristics under what we believe to be a worst-case scenario.
To generate the optimum solutions we used a dynamic programming algorithm. The results of the experiments comparing the LHWHM, BFM-BDMCP and LARAC algorithms are presented in Table 1. The simulations were carried out by generating source and destination randomly for each graph. The delay T for the algorithms has been calculated as 10% more than the shortest delay path value. It is observed that for smaller networks all the three algorithms generate nearly optimal solutions, but as the graph gets bigger the number of feasible paths also increase, in this case the BFM-BDMCP algorithm outperforms the LHWHM algorithm. Since the BFM algorithm inspects many more paths than the LHWHM algorithm, the time taken for the algorithm is also seen to increase. LARAC returns paths closer to the optimal in most of the cases, but the time taken for computing the paths was generally more than that taken by the LHWHM algorithm by a factor of 10. The OPT column in the tables refers to the optimal cost.

4.1. Results comparing BFM-BDMCP Heuristic with Hassin's FPAS Algorithm

The results of our comparison with Hassin's FPAS (fully polynomial time approximation scheme) are presented in Table 2. We first ran the BFM-BDMCP algorithm on the K-connected

graphs. For each graph, using the optimal values we computed the ε-value. These ε values are given as inputs to Hassin's algorithm. Note that for a given value of ε, Hassin's algorithm guarantees a solution which is within (1+ε) of the optimum value. In Table 2, T1 refers to the time taken by Hassin's algorithm as proposed originally, and T2 refers to the time taken when the solution produced by the BFM-BDM heuristic is used as an upper bound in Hassin's algorithm. Though, as expected, Hassin's algorithm takes a considerably longer time than the BFM-BDMCP algorithm, it produces optimal solutions in most cases. This means that for small values of ε the algorithm produces optimal solutions. If we first run the BFM-BDMCP algorithm and then use the result as an upper bound in Hassin's algorithm, the time taken (T2) gets reduced to a considerable extent

Nodes	Eps(ε)	BFM-BDMCP	FPAS			OPT
		cost	cost	T1 (sec)	T2 (sec)	cost
100	0.146	401	350	31.6	20.6	350
200	0.033	1553	1503	1020	306	1503
250	0.156	737	637	461	196	637
300	0.16	1451	1250	737	442	1250
350	0.13	3056	2703	1533	471	2703
400	0.151	1256	1091	1693	298	1091
450	0.01	859	851	806	603	851
500	0.15	5157	4855	2050	1382	4475

Table 2

5. CONCLUSIONS

In this paper we discussed one of the QoS routing problem formulated as the bounded delay minimum cost path problem. (BDMCP). We presented a new heuristic called the BFM-BDMCP algorithm which is based on the Bellman-Ford-Moore algorithm for the min-cost path problem. We discussed several issues pertaining to efficient implementations of this algorithm. We concluded with an experimental evaluation of this heuristic in comparison with the the LHWHM algorithm, the LARAC heuristic and Hassin's approximation algorithm. It is observed that the average cost of the path returned by the BFM-BDMCP algorithm is better than that generated by the LHWHM algorithm, but the time taken by the BFM-BDMCP algorithm in certain cases in just too high. LARAC for most of the cases returns paths closer to the optimal. The time taken for computing the paths was noticed to be generally more than that for the LHWHM algorithm by a factor of 10. We believe that the LHWHM and the BFM-BDMCP algorithms are serious candidates for implementation in real network environments as they are extension of the standard Dijkstra's and BFM algorithms used in routing protocols.

NODES	LHWHM		BFM-BDMCP (Async)		BFM-BDMCP (Sync)		LARAC		OPT
	cost	time (sec)	cost	time	cost	time	cost	time	cost
100	168	1.54E-04	168	.0023	168	.0055	168	.00104	168
250	198	7.05E-04	198	.0268	198	0.046	198	.00514	198
400	552	2.08E-03	552	.0282	522	0.314	533	0.035	500
550	2650	7.60E-03	2650	.0986	2501	1.946	2423	0.233	2400
800	950	6.30E-03	950	0.145	807	1.83	900	0.088	751
1000	5651	3.25E-02	5550	9.75	5451	12.28	4702	0.68	4701
1400	10750	0.089	10400	41.12	10100	49.84	8765	1.91	8750
1800	16200	0.1709	14600	74.77	14600	100.8	13107	2.17	12890
2000	3801	0.028	3801	1.69	3450	13.09	3127	0.94	3100
2500	97	1.70E-02	97	23.47	97	33.15	97	0.04	97
3000	296	8.80E-02	296	41.91	296	78.15	296	.109	296

Table 1

6. REFERENCES

[1] Jaffery M.Jaffe, "Algorithms for finding Paths with Multiple Constraints", *Networks*,Vol. 14 (1984), pp.95-116.1

[2] Z.Wang and J.Crowcroft, "Quality-of-Service Routing for Supporting Multimedia Applications," *IEEE on Selected Areas in Communications*,vol.14, no.7,pp. 1228-1234, September 1996.

[3] J.Garcia-Luna-Aceves and J.Behrens, "Distributed Scalable Routing Based on Vectors of Link States," *IEEE Journal on Selected Areas in Communications*, vol.13,no.8,pp.1383-1395, October 1995.

[4] DouglasS.Reeves and Hussein F.Salama, "A Distributed algorithm for Delay-Constrained Unicast Routing". *IEEE/ACM transactions on Networking*, vol 8,no 2, pp. 239-250 April 2000.

[5] Roch A.Gue'rin, "QoS Routing in Networks with Inaccurate Information: Theory and Algorithms,"*IEEE/ACM Transactions on Networking*. Vol.7,no.3,June 1999

[6] R.WIDYONO, "The Design and Evaluation of Routing Algorithms for Real-Time Channels," *Tech.Rep.ICSI TR-94-024*, University of California at Berkely, International computer Science Institute, June 1994.

[7] Gang Luo, Kaiyuan Huang, Jianli Wang, Chris Hobbs and Ernst Munter, "Multi-Qos Constraints Based Routing for IP and ATM Networks," in *Proceedings of IEEE Workshop on QoS Support for Real-Time Internet Applications*, June 1, 1999, Vancouver Canada.

[8] K.Thulasiraman and M.N.S.Swamy, *Graphs: Theory and Algorithms*, Wiley Interscience, 1992.

[9] Alpar Juttner, Balazs Szviatovszki, Ildiko Mecs, Zsolt Rajko, " Lagragian Relaxation Based Method for the QoS Routing Problem", *IEEE INFOCOM-2001*, pp. 859-868

[10] R.Hassin , "Approximation Schemes for the Restricted Shortest Path Problem", *Mathematics of Operation Research*, 17(1), 1992, pp.36-42

[11] Ravi S. Ravindran, " Distributed and Sequential Heuristics for QoS Routing in Communication Networks", *M.S Thesis,Computer Science Dept. University of Oklahoma, Norman*, 2000