Multi-constraint Macro-routing by using the extended full-mesh aggregation technique

Sanda Dragos
Faculty of Mathematics and Computer Science
”Babes-Bolyai” University
400084, Cluj-Napoca, Romania
sanda@cs.ubbcluj.ro

Martin Collier
School of Electronic Engineering
Dublin City University
Glasnevin, Dublin 9, Ireland
collierm@eeng.dcu.ie

Abstract

QoS routing is used for finding feasible paths that satisfy simultaneously multiple constraints and it is a NP-complete problem. The difficulty of solving such problems increases in a hierarchical context, where aggregation techniques influence the path computation process. We propose a new aggregation technique which allows the selection of multiple paths that satisfy multiple QoS constraints. Thus, the probability of finding multi-constraint paths if such paths exist increases considerable. This aggregation technique is called extended full-mesh *(EFM) and is intended for use with the* Macro-routing *protocol.*

1 Introduction

Multi-constraint routing occurs when we wish to route data based on multiple criteria (e.g., delay, bandwidth, economic cost) and is an NP-complete problem [3]. To solve such routing problems researchers resort to heuristical solutions or approximation algorithms.

Within a hierarchical routing context, the multi-constraint problem becomes even more complex [6, 5, 7], as it is very difficult to designate *optimal* multi-constraint paths for building the aggregate nodal representations. That is because one path may be the best for one constraint but may not satisfy other constraints. Thus in modelling the available paths between two nodes at one level of the hierarchy by a single logical link at a higher level, it is unclear how to map the path metrics into the logical link. The conventional methods for solving the *“best” path* problem in a hierarchical context are described in [6, 5].

1. The single path metric, where the logical link is assigned the metrics of the ”best” path according to only one QoS metric. One metric must thus be selected as this ”most important” metric.

2. The best case approach, which, for each metric, finds the best path and assigns the corresponding metric value to the logical link. No path with these collective attributes may exist at the lower level of the hierarchy.

3. The worst case approach, similar to the above, but where the *worst* path is chosen.

To avoid the optimistic representation of the best-case and the conservative representation of the worst-case approaches, an alternative is:

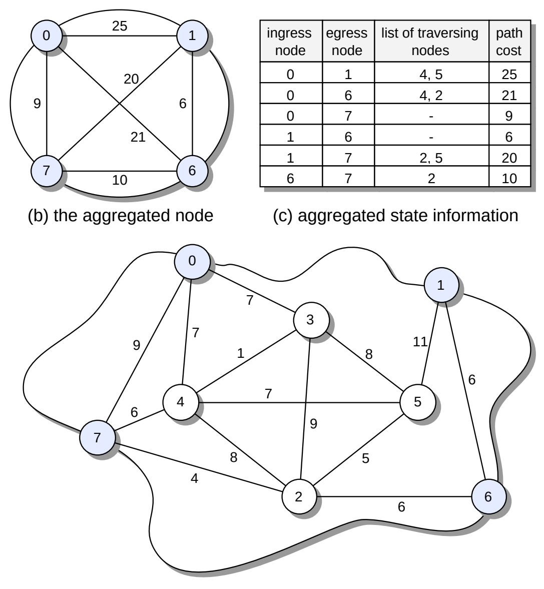
4. The averaging approach, where the metrics for *all* possible paths are averaged to obtain the metrics for the logical link.

Such approximations may result in poor path selection, or in failure to identify valid paths. This paper describes an improved aggregation mechanism and an associated routing technique which uses mobile agents.

Our hierarchical routing protocol, called *Macro-routing* [2], is able to find any source-destination path based on one QoS metric. Here, we want to extend *Macro-routing*’s capabilities to compute multi-constraints paths by introducing a new aggregation technique: the *extended full-mesh*. This new representation allows the selection of multiple paths that satisfy the multi-constraints. This postpones the radical elimination of possible good sub-paths while building the nodal aggregate representation, thus increasing the chances of finding the final multi-constrained path.

2 Macro-routing

The routing process within *Macro-routing* is performed by using mobile agents which flood the network to find best paths. Each mobile agent computes the paths and their QoS metrics in a distributed and parallel manner, while spanning the network. Within each node, a mobile agent processes the information gathered while traversing the path up to the current node with the information residing within the current node. Then, the mobile agent replicates and dispatches


(b) the aggregated node

ingress node	egress node	list of traversing nodes	path cost
0	1	4, 5	25
0	6	4, 2	21
0	7	-	9
1	6	-	6
1	7	2, 5	20
6	7	2	10

(c) aggregated state information

(a) the physical domain

Figure 1. The *full-mesh* nodal aggregation

itself to explore all other possible directions.

The hierarchical path computation starts from the domains within the first hierarchical level. Here, within each "participant" domain, mobile agents search for paths that traverse the domain satisfying the QoS constraints. One *best* path is selected for each border-to-border pair in order to build the full-mesh nodal aggregation for the next level in the hierarchy. An example of full-mesh aggregation within *Macro-routing* is depicted in Fig. 1.

Mobile agents released from each border node traverse the domain replicating themselves to search for all possible paths that satisfy the QoS constraints. Upon reaching a border node different from the starting one, each mobile agent sends the gathered information to the managing node of the current domain (see Fig. 1(c)).

Sending the information to the managing node can be done either by piggybacking on legacy messages or using mobile agents. In our implementation we used the WAVE system which has incorporated routines that allow a "virtual" direct access between any two network nodes.

The information sent by a successful[1] *wave* to the managing node comprises: the starting border node (*ingress node*), the list of traversed nodes, the path cost, and the final border node reached (*egress node*).

The path cost is partially computed within every traversed node. A mobile agent carries a variable, i.e., C, containing the intermediate path cost. Its starting value it will be $C = 0$ for additive metrics, $C = 1$ for multiplicative metrics and $C = \infty$ for concave metrics and the path cost computation performed within node i is:

for additive metrics $$C = C + L_i + N_i$$

for multiplicative metrics $$C = C \cdot L_i \cdot N_i$$

for concave metrics $$C = min\{C, L_i, N_i\}$$

where L_i represents the QoS metric for the link traversed by the mobile agent to reach the current node and N_i represents the nodal QoS metric for traversing the current node. N_i is used only for nodes within the hierarchical levels above the physical one, where nodes are aggregate representation of domains within lower levels of the hierarchy.

Amongst the paths found, one *"best"* path will be selected (as in Fig. 1(b) and 1(c)) for each ingress-egress pair to create the full-mesh aggregated representation for the next hierarchical level.

In the subsequent section we propose a new aggregation technique, the *extended full-mesh*, in order to be able to find optimal multi-constraint paths within an hierarchical routing context by using *Macro-routing*.

3 The *extended full-mesh* (EFM) aggregate representation

The *extended full-mesh* (EFM) aggregate representation is an extension of the full-mesh, done by allowing not only one "best" path between any two border nodes but multiple satisfactory paths. An example for such a representation for the domain depicted in Fig. 1(a) is presented in Fig. 2(b).

To explain the EFM aggregate representation, we first build an example for a single metric for the network topology presented in Fig. 1(a). We start by searching all the possible paths between any two border nodes and calculate their costs. Then, for each pair of border nodes we order the path costs from *best* to *worst* (in our case, in increasing order as we have an additive metric). The results are presented in Fig. 2(a). These path costs will be used in building the *extended full-mesh* aggregate representation by using intervals of metrics instead of single metric values (see Fig. 2(b)).

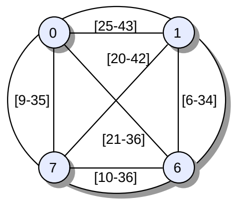
We call the range of path costs selected for one port-to-port link connection in an EFM representation the ***EFM interval***. Although we have used a single metric to describe the EFM concept, this new aggregate representation is intended for use in finding multi-constraints paths. Thus, the *EFM interval* is n-dimensional when we consider n metrics (see Fig. 3). The elements within an *EFM interval* are the costs of alternative paths with multiple metrics, and thus can be represented as *points* in the n-dimensional space - we will call them ***EFM paths***.

The *EFM interval* representation will be stored in n

[1]a *wave* that traverses the entire domain from one border node to another by following a path that satisfies the constraints. Other *waves* terminate either because the path they follow does not satisfy the constraints or because they end up in a cycle.

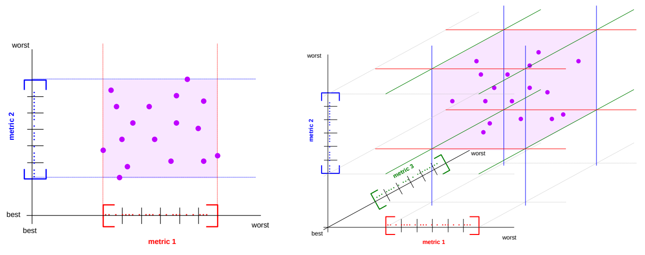
from **node** to **node**	ordered path costs
from **0** to **1**	25, 26, 27, 31, 32, 33, 42, 43
from **1** to **6**	6, 22, 32, 34
from **6** to **7**	10, 20, 22, 24, 26, 36
from **7** to **0**	9, 13, 14, 19, 20, 21, 23, 24, 25, 28, 30, 34, 35
from **0** to **6**	21, 22, 23, 26, 27, 36
from **1** to **7**	20, 24, 26, 30, 32, 42

(a) ordered path costs for every path found between any pair of border nodes



(b) the *extended full-mesh*

Figure 2. Building the *extended full-mesh* **aggregate representation for the network in Fig. 1(a) in two steps**



(a) for two metrics (b) for three metrics

Figure 3. *EFM intervals* **and** *EFM paths*

vectors, where n represents the number of metrics considered and consequently the number of dimensions of the interval space. Thus, if we consider two metrics (i.e., $n = 2$) we have two vectors: $[m_1^1, m_2^1, ..., m_k^1]$ and $\{m_1^2, m_2^2, ..., m_k^2\}$, containing k path values for the two metrics m^1 and m^2. The number of elements in all metric vectors is the same, i.e., k, as they all represent the same k *EFM paths*. Another representation of the k *EFM paths* is: $(m_1^1, m_1^2), (m_2^1, m_2^2), ...(m_k^1, m_k^2)$, where each EFM path i can be represented in the bi-dimensional space as a point by using two coordinates (m_i^1, m_i^2).

4 Macro-routing with the *extended full-mesh* aggregation

We use our hierarchical routing protocol, *Macro-routing*, to find multi-constraint paths by using the EFM aggregate representation. To explain how EFM works in a hierarchical network, we have chosen a very simple hierarchical example of three domains and two hierarchical levels. This example is depicted on the left hand side of Fig. 4. The path search is performed using two metrics: one additive and one concave. The same algorithm can be applied should we have only additive or multiplicative metrics, without modification.

Our algorithm starts from the *source* node a by initiating, as in *Macro-routing*, the search for all the participant domains. Then, within each participant domain at *Level 0* (all three domains in our example), *Macro-routing* initiates a *wave*-based search for all possible paths between all border nodes. They will be used to build the aggregate representations used by *Level 1* as depicted in Fig. 4.

We define $\&$ to be the operator for the concave metric and $+$ to be the operator for the additive metric so that:

$$x\&y = min(x, y),$$
$$\{x, y\}\&z = \{min(x, z), min(y, z)\} \text{ and}$$
$$\{x, y\}\&\{z, t\} = \{min(x, z), min(y, z), min(x, t), min(y, t)\}$$
$$\{x, y\} + z = \{(x + z), (y + z)\} \text{ and}$$
$$\{x, y\} + \{z, t\} = \{(x + z), (y + z), (x + t), (y + t)\}$$

We have within *Level 0* three participant domains: the source domain, one transit domain and the destination domain. We will present the EFM aggregation process for the first two domains.

In the source domain we have two border nodes: a and c. Thus, the EFM aggregate representation for the source domain will have in our case only a single link. Computations for determining its EFM metric starts by searching all possible paths between the two border nodes a and c.

$$a \to c = 5, 2$$
$$a \to b \to c = (1 + 2), (9\&4) = 3, 4$$
$$a \to b \to d \to c = (1 + 2 + 3), (9\&3\&5) = 6, 3$$

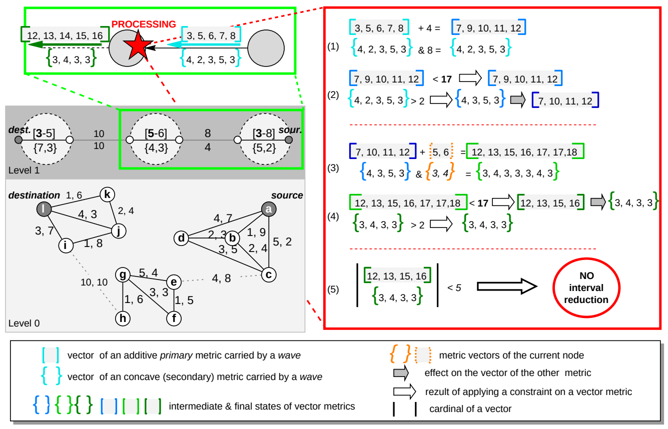

PROCESSING
12, 13, 14, 15, 16
3, 4, 3, 3
3, 5, 6, 7, 8
4, 2, 3, 5, 3
dest.
[3-5]
{7,3}
10
10
[5-6]
{4,3}
8
4
[3-8]
{5,2}
sour.
Level 1
destination
source
Level 0
NO interval reduction
vector of an additive primary metric carried by a wave
vector of an concave (secondary) metric carried by a wave
intermediate & final states of vector metrics
metric vectors of the current node
effect on the vector of the other metric
rezult of applying a constraint on a vector metric
cardinal of a vector

Figure 4. Example for two metrics (one *primary* **additive metric and one** *secondary* **concave metric)**

$a \rightarrow d \rightarrow c = (4+3), (7\&5) = 7, 5$

$a \rightarrow d \rightarrow b \rightarrow c = (4+2+2), (7\&3\&4) = 8, 3$

Therefore, the EFM nodal traversing metric (i.e., the EFM interval) for the source domain is $[3, 5, 6, 7, 8], \{4, 2, 3, 5, 3\}$.

In the transit domain we also have two border nodes: e and h. The possible paths between them are:

$e \rightarrow g \rightarrow h = (5+1), (4\&6) = 6, 4$

$e \rightarrow f \rightarrow g \rightarrow h = (1+3+1), (5\&3\&6) = 5, 3$

Thus, the EFM nodal traversing metric (i.e., the EFM interval) for the transit domain is $[5, 6], \{3, 4\}$.

At the next level of the hierarchy, i.e., *Level 1,* there is only one domain containing three nodes, all representing the aggregate format of the domains from *Level 0.* After all these aggregated representations have been composed (i.e., once *Level 1* is complete), the managing node of the *Level 1* domain initiates the search for all paths traversing this domain. Thus, a *wave* is released from the source node to find all possible paths to the destination node. When leaving the source node, the *wave* will carry with it the EFM metric for traversing the source node, i.e., $[3, 5, 6, 7, 8], \{4, 2, 3, 5, 3\}$. At the next node, this information will be used along with the information residing in the current node for the "partial" path processing phase. This process is presented in the right hand side of Fig. 4 and progresses in 5 steps:

1. compute the current metric values (for both metrics) considering the metrics of the link just passed.

2. apply the constraint tests for each newly computed metric value. In the right hand side of Fig. 4, the second element from the *secondary* vector was eliminated as it didn't pass the constraint test. Consequently, the element with the same index had to be eliminated in the *primary* vector as well.

3. compute the metric costs considering the costs for traversing the current node (i.e., the traversing node).

4. apply the constraint tests once more. Here, we eliminate the last two options within the *primary* vector. This will have the same effect on the *secondary* vector as well.

5. apply the *threshold* test. If the number of EFM paths within the EFM interval exceeds the threshold, some EFM paths have to be eliminated using one of the techniques presented in Section 5. In our case the threshold test is passed successfully (i.e., there were no more than 5 EFM paths), with no EFM path elimination.

The extension of *Macro-routing* with the EFM aggregation is a simple, distributed process which considers multiple possible paths during the computational process. However, there may be situations where the number of paths satisfying all constraints might be too large for an efficient path computation. Methods for reducing the *EFM interval* when the number of *EFM paths* exceeds a threshold T are presented in the next section.

5 Decreasing the number of EFM paths

The number of *EFM paths* may increase to a value which can negatively influence the path computation process. That is because with more candidate paths we need to perform more computations, and the size of a *wave*[2] may increase so that the algorithm generates too much control traffic.

To control the *wave* size we can keep the number of *EFM paths* limited by a threshold T. The main factors that govern the number of EFM paths are:

a) The topology of the network. The number of EFM paths may increase with the network connectivity. This, is true especially when there are enough resources to satisfy the constraints.

b) The number of border nodes per domain. This does not directly influence the number of EFM paths when building the EFM aggregate representation at a single level. However, at the next level of hierarchy the number of border nodes becomes the node degree of the aggregated domain. The arguments of condition **a)** then apply.

c) The number and severity of the constraints. We expect, in general, to find few EFM paths where the set of constraints is large and/or demanding.

d) The availability of resources.

We conclude that sparse networks, a large or demanding set of constraints, and a poverty of resources are all contexts in which a threshold may be unnecessary since the number of EFM paths expected to be found is low. In other contexts, for our algorithm to be efficient a threshold value is required to limit the number of EFM paths evaluated by *waves*.

We present below two classes of techniques for reducing the number of considered EFM paths. One (*truncation*) is a greedy method through which the best resources are occupied first; the other method (*dispersal*) tries to keep a larger spectrum of alternative EFM paths.

In describing these techniques below, we make the following assumptions:

- The methods are applied to a two-dimensional EFM interval. (although they are applicable on any n-dimensional EFM interval with $n \geq 2$).

- All metrics are normalised to lie in the range from 0 (*best*) to 1 (*worst*).

[2]The information a *wave* has to carry, in addition to that of a standard *Macro-routing wave,* is $n \cdot k \cdot c$ bytes where n is the number of metrics, k is the number of *EFM paths,* and c is the number of bytes occupied by a metric value.

5.1 Truncation of the EFM interval

Here we eliminate paths starting from the *"worst"* down until only T EFM paths remain.

The problem is that we cannot clearly distinguish the *worst* multi-constraint paths. Three possible methods to select these paths are considered below. They are schematically represented in Fig. 5.

The first method, depicted in Fig. 5(a) progressively eliminates the EFM paths that are traversed by a line $f(x,t) = 2t - x$ which lies perpendicular to the diagonal from the *worst-worst* (1,1) corner down to the *best-best* (0,0), as the parameter t decreases from 1 towards 0, until (at some value $t = t_0$) (at most) T EFM paths remain. The EFM paths that satisfy the inequality $f(x,t_0) > y$ are the selected ones.

Another method, presented in Fig. 5(b), is similar but the delimitation between the selected and the unselected paths is not a line but an arc. This arc is part of the circle centred at $x = 0, y = 0$. The circle's radius, r, varies from $\sqrt{2}$ down to 0 until the number of remaining EFM paths is no more than T. When we find the appropriate radius $r = r_0$ all EFM paths inside the arc are selected. Thus an EFM path is retained if it satisfies the inequality $\sqrt{x^2 + y^2} < r_0$.

Both these solution are relatively complicated, even for this simple two-dimensional example. In a real-time process like routing simple solutions are preferred such as that depicted in Fig. 5(c). Here, we eliminate the *worst* path with respect to a single metric. The designation of the considered metric is done either starting with a randomly chosen metric and then choosing other metrics for the next path eliminations in a round-robin fashion or based on a *priority* metric attribute. The chosen attribute may change with every path elimination allowing another metric with a higher priority to be the next decision factor.

5.2 Dispersal within the EFM interval

Another method for reducing the number of EFM paths is to reduce their density within the EFM interval. This can be done also by various methods. Fig. 6 depicts three of them.

The first method, presented in Fig. 6(a), is to divide the EFM bi-dimensional interval into T parts with $T-1$ perpendicular lines on the first diagonal. The EFM paths contained by the k^{th} partition, with $0 \leq k < T$, satisfy equation (1). Here, x and y are the coordinates used to represent the EFM paths in the bi-dimensional space.

$$\frac{k}{T} - x \leq y < \frac{k+1}{T} - x \Longrightarrow \frac{k}{T} \leq x + y < \frac{k+1}{T} \quad (1)$$

One EFM path is selected from each partition. The selection algorithms could be numerous: random selection, the closest to the origin ($x = 0$, $y = 0$), the closest to the first diagonal, etc. The drawback of this approach is that if EFM paths are not heterogeneously distributed within the interval, there may be intervals containing many EFM paths, and others containing none.

A similar method is depicted in Fig. 6(b) and divides the EFM bi-dimensional interval into T parts using arcs instead of lines. Equation 2 determines the EFM paths, contained by the k^{th} partition, with $0 \leq k < T$. We are representing EFM paths through their distance from the origin like $\sqrt{x^2 + y^2}$, where x and y are the coordinates in the bi-dimensional space.

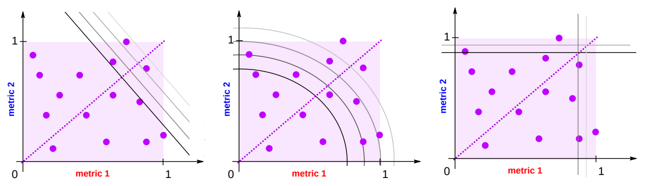
$$\frac{k\sqrt{2}}{T} \leq \sqrt{x^2 + y^2} < \frac{(k+1)\sqrt{2}}{T} \quad (2)$$

Both of these methods may be too complicated to be implemented in a real-time process like routing. Simpler algorithms which are as effective in ensuring that the set of EFM paths is widely dispersed must be found.

A third method, although relatively complex, "spaces out" the EFM interval, regardless of grouping of path costs, thereby avoiding the drawback of "partitioning" solutions. The distances between each pair of EFM paths, $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$, are measured as in equation 3.

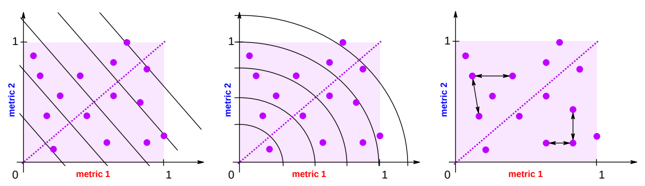
$$D_{1,2} = |P_1 - P_2| = |x_1 - x_2| + |y_1 - y_2| \quad (3)$$

Then, one of the two *closest* EFM paths will be eliminated. The process repeats until we have only T EFM paths.



(a) *normal* method (b) *radius* method (c) *simple* method

Figure 5. Methods for truncating a two metric EFM interval



(a) *normal* method (b) *radius* method (c) *distance* method

Figure 6. Methods for dispersing a two metric EFM interval

The first two models proposed for both truncation and dispersal techniques resemble solutions proposed for solving the multi-constraint path problems, with the difference that such solutions are used to select single "best" path and we want to select T paths. Thus, our *normal* method (Figs 5(a) and 6(a)) uses a similar linear function to the Lagrangian relaxation techniques (e.g. Jaffe's approximation [4]), while our *radius* method (Figs 5(b) and 6(b)) uses a similar nonlinear function to that used for instance in the TAMCRA algorithm [1].

5.3 Practical algorithms

The above algorithms require a lot of computation and are thus impractical in most applications. Instead we must find simpler algorithms. Our simulation results show that truncation algorithms result, in general, in lower-cost routes being found, but, because dispersal algorithms find a wider spread of routes, they provide better load balancing. Depending on the network topology, connectivity and load, and the nature of the traffic (e.g., the generating application) one or other of these effects may be preferable.

In adopting a realisable algorithm, we try to approximate the above two classes of algorithm.

These alternatives include:

- **First-Come-First-Stored.** The first T paths reported back to the managing node are the ones considered for the final EFM representation.
  This mechanism belongs to the truncation class, and will tend to favour paths with low delay and few hops. However, by deliberately delaying waves as they return to the managing node, in accordance with an appropriate metric, we can get the FCFS approach to roughly order paths in accordance with that metric.

- **Random selection.** Pick T paths at random from the available paths. This is a reasonable approximation to the dispersal approach, and is computationally less intensive. However, true randomness being difficult to implement in lightweight real-time processes, it will in practice be approximated using some cyclic selection process or selection based on some quasi-random attribute such as a wave checksum.

We are currently evaluating some alternative and easily implemented criteria for choosing which paths to discard.

6 Conclusions

The *extended full-mesh* aggregate representation is a compromise between the full-mesh aggregation and no aggregation. With traditional routing strategies (i.e., collecting all the routing information centrally in order to be processed) even the full-mesh representation raises scalability issues. However, by using mobile agents which are able to process the information *in situ*, the volume of such information can be larger.

The full-mesh representation works well for finding hierarchical based on a single metric. However, it does not guarantee to find a hierarchical path based on multiple metrics, even if such a path exists. That is because it is very difficult to designate a "best" path for the full-mesh aggregate representation, and so the routing algorithm may ignore a viable path. EFM allows more paths to be considered, increasing the chances of finding a viable path.

A disadvantage of the EFM aggregation technique when used with the Macro-routing protocol is that it can potentially generate too much routing traffic. Thus, the size of an EFM representation has to be limited. A number of techniques for limiting the EFM representation were presented and discussed within the paper. Further research is required to determine the best such technique, and how to choose the limit T on the number of paths stored in the EFM representation.

A final observation is that the amount of overhead generated by the Macro-routing protocol with EFM is, in principle, selflimiting. A large number of *wave* packets will be generated only when a range of routes meet the multiple constraints. However, in most cases, these constraints will include bandwidth, so that a large population of *waves* will be generated only in network conditions where capacity is available to transport them. Our simulation results to date are consistent with this thesis, but a considerable range of topologies and traffic loads must be simulated to verify that this holds in general.

References

[1] H. De Nerve and P. Van Mieghem. TAMCRA: A Tunable Accuracy Multiple Constraints Routing Algorithm. *Computer Communications*, 23:667–679, 2000.

[2] S. Dragoş and M. Collier. Macro-routing: a new hierarchical routing protocol. In *GLOBECOM*, volume 3, pages 1510–1514, 2004.

[3] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman & Co, 1979.

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

[5] T. Korkmaz and M. Krunz. Source-oriented topology aggregation with multiple QoS parameters in hierarchical ATM networks. In *IWQoS'99*, pages 137–146, 1999.

[6] W. C. Lee. Spanning tree method for link state aggregation in large communication networks. In *INFOCOM'95*, pages 297–302, 1995.

[7] K.-S. Lui, K. Nahrstedt, and S. Chen. Hierarchical QoS routing in delay-bandwidth sensitive networks. In *LCN*, pages 579–588, 2000.