Multiobjective Multicast Routing Algorithm for Traffic Engineering

Jorge Crichigno
Research Department
National Computing Center
Asunción, Paraguay
jcrichigno@cnc.una.py

Benjamín Barán
Science and Technology Department
Catholic University of Asunción
Asunción, Paraguay
bbaran@cnc.una.py

***Abstract*—This paper presents a new version of a multiobjective multicast routing algorithm (MMA) for traffic engineering, based on the Strength Pareto Evolutionary Algorithm (SPEA), which simultaneously optimizes the maximum link utilization, the cost of the tree, the maximum end-to-end delay and the average delay. In this way, a set of optimal solutions, known as Pareto set, is calculated in only one run, without a priori restrictions. Simulation results show that MMA is able to find Pareto optimal solutions. They also show that for dynamic multicast routing, where the traffic requests arrive one after another, MMA outperforms other known algorithms.**

Keywords - computer networks; multicast routing; multiobjetive optimization; evolutionary algorithms

I. INTRODUCTION

Multicast consists of simultaneous data transmission from a source node to a subset of destination nodes in a computer network [1]. Multicast routing algorithms have recently received great attention due to the increased use of new point to multipoint applications, such as radio and TV transmission, on-demand video and teleconferences. Such applications generally have some quality-of-service (QoS) parameters as maximum end-to-end delay and minimum bandwidth resources.

When a dynamic multicast problem considers various traffic requests, not only QoS parameters must be considered, but also load balancing and network resources must be taken into account. In order to avoid hot spots and to balance the network load, the common approach is to minimize the utilization of the most heavily used link in the network (α), or maximum link utilization [2]. Although this approach is useful to balance the load, it may waste network resources (i.e. sum of assigned bandwidth at each link) [2]. Therefore, minimization of the cost of the tree of each multicast group, which is given by the sum of the cost of the used links, is also desired. It is known that the complexity of computing the minimum cost tree for a given multicast group is NP-hard [3]. So, this paper improves a Multiobjective Multicast Routing Algorithm (MMA) presented in [4], to find a multicast tree optimizing several objective functions. In contrast to mono-objective algorithms [2, 3, 5], MMA finds a set of optimal solutions by minimizing simultaneously the maximum link utilization, the cost of the tree, the maximum end-to-end delay and the average delay. In this way, a whole set of Pareto solutions can be obtained in only one run [6].

The remainder of this paper is organized as follows. Section II describes related works. A general definition of a multiobjective problem is presented in Section III. The problem formulation and the objective functions are given in Section IV. The proposed algorithm is explained in Section V. Experimental environments are given in Section VI, and the results are shown in Section VII. Finally, conclusions and future works are presented in Section VIII.

II. RELATED WORK

The first reference of the problem of finding the lowest cost tree subject to an end-to-end delay in a multicast context were presented by Kompella et al. [7], who proposed an algorithm based on dynamic programming (KPP). For the same problem, Ravikumar et al. [1] presented a method based on a simple genetic algorithm. This work was improved in turn by Zhengying et al. [8] and Barbosa et al. [5]. The main disadvantage of this approach is the necessity of an a priory upper bound for the end-to-end delay that may discard solutions of low cost or low α (or both) with a delay only slightly larger than an a priori predefined upper bound.

In [2], Seok et al. proposed bifurcation and non-bifurcation schemes minimizing α to transport multicast flows with hop-count constrained. Given that schemes are NP-hard, they also proposed a heuristic algorithm consisting of two parts: 1- modifying the original graph to a hop-count constrained version; 2- finding a tree to minimize α.

In [3], Donoso et al. proposed a multi-tree traffic engineering scheme using multiple trees for each multicast group. They took into account four metrics: α, hop count, bandwidth consumption and total end-to-end delay. The method minimizes a weighted sum function composed of the above four metrics. Considering the scheme is NP-hard, the authors proposed a heuristic algorithm consisting of two steps: 1- obtaining a modified graph: all possible paths between the source node and every destination node are looked for. Then, for each destination, for each path of the destination, for each node of the path, a distance value based on hop count, bandwidth consumption and delay is computed; 2- in the

This work was supported by the National Computing Center and the Catholic University of Asunción.

modified graph, finding out the trees based on the distance values and the available capacity of the paths.

III. MULTIOBJECTIVE OPTIMIZATION PROBLEMS

A general Multiobjective Optimization Problem (MOP) includes a set of *n* decision variables, *k* objective functions, and *m* restrictions. Objective functions and restrictions are functions of decision variables. This can be expressed as:

Optimize $\quad \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x}))$.
Subject to $\quad \boldsymbol{e}(\boldsymbol{x}) = (e_1(\boldsymbol{x}), e_2(\boldsymbol{x}), \ldots, e_m(\boldsymbol{x})) \geq \boldsymbol{0}$,
where $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \boldsymbol{X}$ is the decision vector, and $\boldsymbol{y}=(y_1,y_2, \ldots, y_k) \in \boldsymbol{Y}$ is the objective vector.

$\boldsymbol{X}$ denotes the decision space while the objective space is denoted by $\boldsymbol{Y}$. Depending on the problem at hand, "*optimize*" could mean minimize or maximize. The set of restrictions $\boldsymbol{e}(\boldsymbol{x})\geq\boldsymbol{0}$ determines the set of feasible solutions $\boldsymbol{X_f}$ and its corresponding set of objective vectors $\boldsymbol{Y_f}$. The problem consists of finding $\boldsymbol{x}$ that optimizes $\boldsymbol{f}(\boldsymbol{x})$. In general, there is no unique "best" solution but a set of solutions, none of which can be considered better than the others when all objectives are considered at the same time. This derives from the fact that there can be conflicting objectives. Thus, a new concept of optimality should be established for MOPs. Given two decision vectors $\boldsymbol{u}, \boldsymbol{v} \in \boldsymbol{X}$:

$\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{f}(\boldsymbol{v})$ iff $\forall i \in \{1,2,...,k\}: f_i(\boldsymbol{u}) = f_i(\boldsymbol{v})$
$\boldsymbol{f}(\boldsymbol{u}) \leq \boldsymbol{f}(\boldsymbol{v})$ iff $\forall i \in \{1,2,...,k\}: f_i(\boldsymbol{u}) \leq f_i(\boldsymbol{v})$
$\boldsymbol{f}(\boldsymbol{u}) < \boldsymbol{f}(\boldsymbol{v})$ iff $\boldsymbol{f}(\boldsymbol{u}) \leq \boldsymbol{f}(\boldsymbol{v}) \wedge \boldsymbol{f}(\boldsymbol{u}) \neq \boldsymbol{f}(\boldsymbol{v})$

Then, in a minimization context, they comply with one of three conditions:
$\boldsymbol{u} \bullet \boldsymbol{v}$ ($\boldsymbol{u}$ dominates $\boldsymbol{v}$), iff $\boldsymbol{f}(\boldsymbol{u}) < \boldsymbol{f}(\boldsymbol{v})$
$\boldsymbol{v} \bullet \boldsymbol{u}$ ($\boldsymbol{v}$ dominates $\boldsymbol{u}$), iff $\boldsymbol{f}(\boldsymbol{v}) < \boldsymbol{f}(\boldsymbol{u})$
$\boldsymbol{u} \sim \boldsymbol{v}$ ($\boldsymbol{u}$ and $\boldsymbol{v}$ are non-comparable), iff $\boldsymbol{f}(\boldsymbol{u}) \bullet \boldsymbol{f}(\boldsymbol{v}) \wedge \boldsymbol{f}(\boldsymbol{v}) \bullet \boldsymbol{f}(\boldsymbol{u})$

Alternatively, for the rest of this work, $\boldsymbol{u} \triangleright \boldsymbol{v}$ will denote that $\boldsymbol{u}$ dominates or is equal to $\boldsymbol{v}$.

A decision vector $\boldsymbol{x} \in \boldsymbol{X_f}$ is non-dominated with respect to a set $\boldsymbol{V} \subseteq \boldsymbol{X_f}$ iff: $\boldsymbol{x} \bullet \boldsymbol{v}$ or $\boldsymbol{x} \sim \boldsymbol{v}$, $\forall \boldsymbol{v} \in \boldsymbol{V}$. When $\boldsymbol{x}$ is non-dominated with respect to the whole set $\boldsymbol{X_f}$, it is called an optimal Pareto solution; therefore, the *Pareto optimal set* $\boldsymbol{X_{true}}$ may be formally defined:
$\boldsymbol{X_{true}} = \{\boldsymbol{x} \in \boldsymbol{X_f} \mid \boldsymbol{x}$ is non-dominated with respect to $\boldsymbol{X_f}\}$. The corresponding set of objective vectors $\boldsymbol{Y_{true}} = \boldsymbol{f}(\boldsymbol{X_{true}})$ constitutes the *Optimal Pareto Front.*

IV. PROBLEM FORMULATION

For this work, a network is modeled as a direct graph $G=(V, E)$, where V is the set of nodes and E is the set of links. Let $(i,j) \in E$ be the link from node i to node j. For each link (i,j), let z_{ij}, c_{ij}, d_{ij} and t_{ij} be its capacity, cost per bps, delay and current traffic, respectively. Let $s \in V$ denote a source, $N \subseteq V - \{s\}$ denote the set of destinations, and $\phi \in \bullet^+$ the traffic demand (in bps) of a current multicast request. Let $T(s,N)$ represent a multicast tree with s as source node and N as destination set. At the same time, let $p_T(s,n)$ denote a path that connects the source node s with a destination node $n \in N$. Clearly, $p_T(s,n)$ is a subset of $T(s,N)$. Finally, let $d(p_T(s,n))$ represent the delay of the path $p_T(s,n)$, given by the sum of the link delays that conform the path, i.e.:

$$d\left(p_T(s,n)\right) = \sum_{(i,j) \in p_T(s,n)} d_{ij} \quad . \tag{1}$$

Using the above definitions, a multicast routing problem for traffic engineering may be stated as a MOP that tries to find the multicast tree $T(s,N)$ minimizing the following objectives simultaneously:

1- Maximum link utilization of the tree:

$$\alpha_T = \underset{(i,j) \in T}{Max} \left\{ (\phi + t_{ij}) / z_{ij} \right\}. \tag{2}$$

2- Cost of the tree:

$$C = \phi \sum_{(i,j) \in T} c_{ij} \quad . \tag{3}$$

3- Maximum end-to-end delay:

$$D_M = \underset{n \in N}{Max} \left\{ d\left(p_T(s,n)\right) \right\}. \tag{4}$$

4- Average delay:

$$D_A = \frac{1}{|N|} \sum_{n \in N} d\left(p_T(s,n)\right), \tag{5}$$

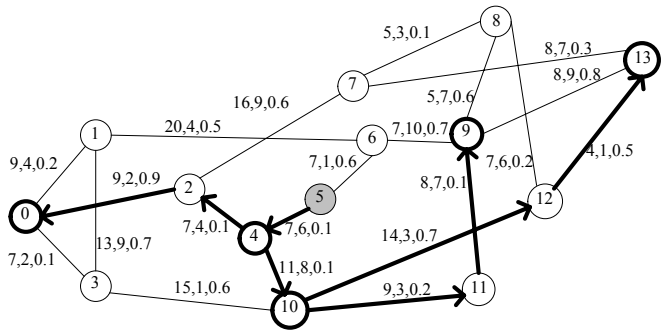
where $|N|$ denotes N's cardinality. The problem is subject to the link capacity constraint:

$$\phi + t_{ij} \leq z_{ij}, \quad \forall (i,j) \in T \quad . \tag{6}$$

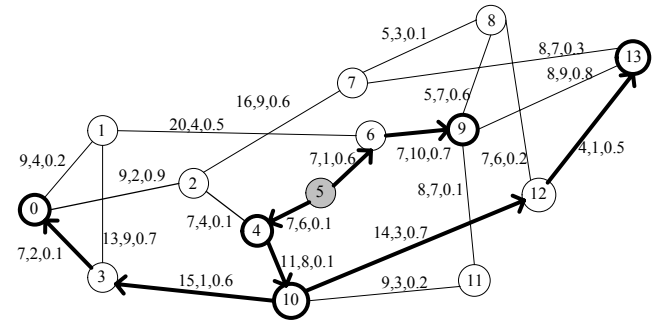
A simple example follows to clarify the notations.

Example 1. Given the network topology in Fig. 1[4], the numbers over each link (i,j) denote d_{ij} in ms, c_{ij}, and t_{ij} (in Mbps) at the current time. For each link, z_{ij}=1.5 Mbps. Suppose a traffic request arriving with ϕ=0.2 Mbps, s=5, and N={0, 4, 8, 9, 13}. Fig. 1(a) shows the tree constructed with KPP [7], subject to a maximum delay of 40 ms. KPP minimizes C subject to a bound delay. Fig. 5(b) and (c) show two alternatives that would not be found by KPP nor by other algorithms based on restrictions if a bound delay lower than 40 ms were a priori established. These alternatives may be good options since they have a bound delay only slightly larger than the predefined bound. Selecting solution (c), C would be minimized, while selecting (b), C would be only a little larger than (c) but the average delay would be lower. If α_T is the most important metric, solution (d) would be the best alternative.

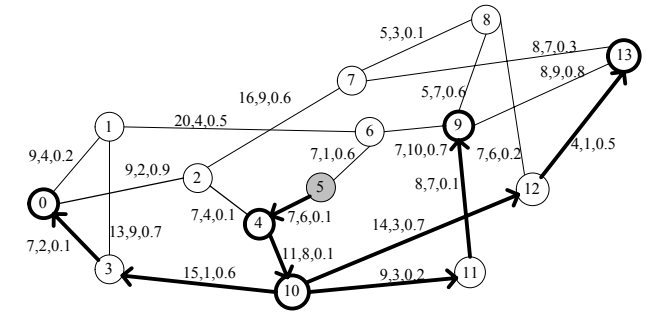
It is important to note from the mathematical formulation that the four objective functions are treated independently and should be minimized simultaneously, i.e. they are not combined to form a scalar single-objective through a linear combination (as weighted sum), nor any of them is treated as a restriction. This way, using the concept of Pareto dominance, a whole set of optimal solution is provided in one run.



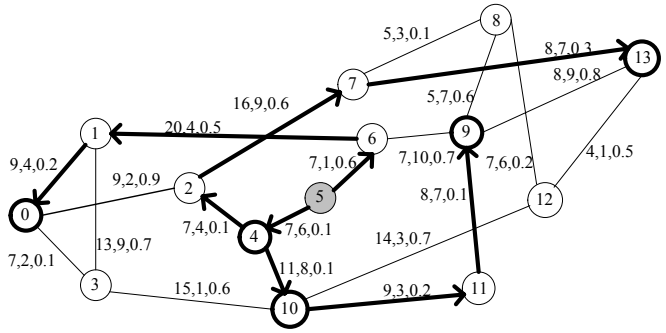
(a) *KPP*. α_T = 0.73, C = 7.2, D_M = 36, D_A = 23.8



(b) α_T = 0.6, C = 6.4, D_M = 40, D_A = 23



(c) α_T = 0.6, C = 6.2, D_M = 40, D_A = 27.2



(d) α_T = 0.53, C = 10.6, D_M = 38, D_A = 26.8

Figure 1. The NSF Net. d_{ij}, c_{ij} and t_{ij} are shown over each link. (a) to (d) show different alternative trees for the multicast request with s = 5, N = {0, 4, 9, 10, 13} and ϕ =0.2 Mbps.

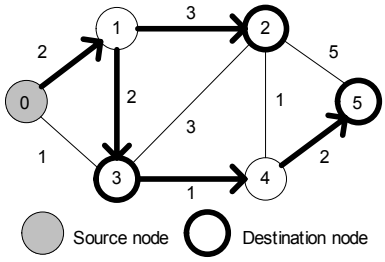
V. PROPOSED ALGORITHM

Following the SPEA scheme [6] and using the framework provided by it, the proposed algorithm holds an evolutionary population P and an external Pareto solution set P_{nd}. Starting with a random population, the individuals evolve to optimal solutions, and these solutions are included in an external optimal Pareto set. Fig. 2 shows an outline of the proposed algorithm. Each procedure is briefly explained.

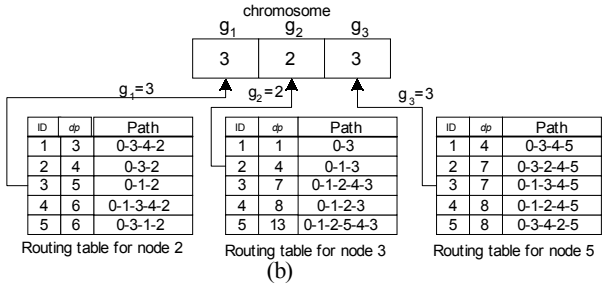
```
-Read multicast group and traffic demand
-Build routing tables
-Initialize P and P_nd
do{
   -Discard identical individuals
   -Evaluate individuals of P
   -Update non-dominated set P_nd
   -Compute fitness
   -Selection
   -Apply crossover and mutation
}while stop criterion is not verified
```

Figure 2. Proposed algorithm.

*Build routing tables***.** Let N=$\{n_1, n_2, \ldots, n_{|N|}\}$. For each $n_i \in N$, a routing table is built. It consists of the R shortest, R cheapest and R least used paths, where the use of a path is defined as the maximum link utilization along the path. R is a parameter of the algorithm. Yen's algorithm [9] was used for this task. A chromosome is represented by a string of length $|N|$ in which the element (gene) $g_i \in Z^+$ in (1,3R), represents a path between s and n_i. The relation between a chromosome, genes and routing tables is shown in Fig. 3 (b). Chromosome in (b) represents the tree in (a) [4]. In this network, d_{ij} is shown for each (i,j), and the routing tables only lists the five shortest delay path with their delay along the path (dp).



(a)



(b)

Figure 3. (a) A network with d_{ij} over the links. (b) Relation between the chromosome, genes and routing tables.

Initialize P and P_{nd}. This procedure generates $|P|$ chromosomes, where P is the evolutionary population. An external global non-dominated Pareto set will be denoted as P_{nd}. At beginning, it includes the non-dominated individuals of the initial P.

Discard identical individuals of P. Applying operations like crossover on two identical chromosomes will yield the same chromosome and the searching ability may be reduced. To

avoid this, a duplicated chromosome is replaced by a new randomly generated [4].

Evaluate individuals of P. For each $i \in P$, the objective vector composed of the four objective functions defined is calculated.

Update non-dominated set P_{nd}. Each non-dominated individuals of P is compared with the individuals in P_{nd}. If that in P is not dominated by anyone of P_{nd}, then it is copied to P_{nd}. Besides, if an individual in P_{nd} is dominated by someone in P, it is removed from the external set.

Compute fitness. The procedure is a two-step process: 1- for each $i \in P_{nd}$ a strength $q_i \in [0, 1)$ is computed, which is proportional to the number of population members $j \in P$ for which $i \triangleright j$:

$$q_i = \left| j \mid j \in P \ \wedge i \triangleright j \right| \cdot |P|^{-1}; \tag{7}$$

2- for each $j \in P$ the strength $q_j \in [1, |P|]$ is computed by summing the strength of all individuals $i \in P_{nd}$ for which $i \bullet j$ plus one:

$$q_j = 1 + \sum_{i \in P_{nd}, i \triangleright j} q_i \ . \tag{8}$$

Finally, the fitness may be calculated as the inverse of the strength; however, it is not really needed with the binary tournament selection chosen for this new approach.

Selection. The selection operator is applied on each generation over the set $P \cup P_{nd}$, to generate a new population P for the next generation. In this improved MMA version, binary tournament [10] has been implemented as selection operator. Two individuals from P are randomly selected (with replacement) and the one with the lower strength wins the tournament and is selected for the next generation.

Apply crossover and mutation operators. The two-point crossover operator is applied over each selected pair of individuals. Then, some genes in each chromosome of the new population are changed (mutated) with probability P_{mut} [10].

VI. Experimental Environment

MMA has been implemented on a 1.25 GHz AMD Athlon computer with a 256 MB of RAM. The compiler used was Borland C++ V 5.02. In order to evaluate MMA, two test problems were used.

A. Test Problem 1

The first test problem consisted of the Example 1. One hundred runs of MMA were done. The parameters were $|P|$=40, R=25 and P_{mut}=0.3. The runs stopped after 500 generations. To validate the results, an exhaustive search method which finds all optimal Pareto solutions was used. The optimal Pareto set was composed of 16 solutions presented in Table I. Table II shows the corresponding objective vectors.

B. Test Problem 2

Fig. 4 represents the NTT network [11] with d_{ij} in ms over each link (i,j). It consists of 55 nodes and 144 links. It is assumed that z_{ij}=6 Mbps and c_{ij}=1, $\forall (i,j) \in E$. From (3), it can be noted that C gives the total bandwidth consumption when c_{ij}=1. Under these conditions, 400 random traffic requests were generated, simulating a dynamic situation in which traffic requests arrive one after another. Each group was randomly selected with a size between 3 and 20. The duration of each request was exponentially distributed (with an average of sixty seconds) and the inter-arrival time randomly distributed between zero and thirty minutes. All traffic demands were set to ϕ=600 Kbps.

TABLE I. Optimal Pareto Set of Example 1

	Tree
S_1	(5,4),(4,2),(2,0),(4,10),(5,6),(6,9),(9,13)
S_2	(5,4),(4,10),(10,12),(12,13),(4,2),(2,0),(5,6),(6,9)
S_3	(5,4),(4,2),(2,0),(4,10),(10,11),(11,9),(10,12),(12,13)
S_4	(5,4),(4,10),(10,12),(12,13),(10,3),(3,0),(6,9)
S_5	(5,4),(4,10),(5,6),(6,1),(1,0),(6,9),(9,13)
S_6	(5,4),(4,10),(5,6),(6,1),(1,0),(6,9),(9,8),(8,12),(12,13)
S_7	(5,4),(4,10),(10,12),(12,13),(5,6),(6,9),(6,1),(1,0)
S_8	(5,4),(4,10),(10,11),(11,9),(4,2),(2,7),(7,13),(5,6),(6,1),(1,0)
S_9	(5,4),(4,10),(10,3),(3,0),(10,11),(11,9),(4,2),(2,7),(7,13)
S_{10}	(5,4),(4,10),(10,3),(3,0),(5,6),(6,9),(9,8),(8,12),(12,13)
S_{11}	(5,4),(4,10),(10,3),(3,0),(5,6),(6,9),(9,13)
S_{12}	(5,4),(4,10),(10,3),(3,0),(10,11),(11,9),(10,12),(12,13)
S_{13}	(5,4),(4,10),(10,3),(3,0),(10,11),(11,9),(9,8),(8,12),(12,13)
S_{14}	(5,4),(4,10),(10,3),(3,0),(10,12),(12,13),(13,9)
S_{15}	(5,4),(4,2),(2,0),(0,3),(3,10),(10,11),(11,9),(10,12),(12,13)
S_{16}	(5,4),(4,2),(2,0),(0,3),(3,10),(10,12),(12,13),(13,9)

TABLE II. Objective Vectors (α_T,C,D_M,D_A)

S_1	0.73,8,23,16.8	S_2	0.73,7,36,19.6
S_3	0.73,6.8,36,23.8	S_4	0.6,6.4,40,23
S_5	0.66,8.4,36,19.4	S_6	0.6,9.4,36,21
S_7	0.6,7.4,36,22.2	S_8	0.53,10.6,38,26.8
S_9	0.53,9.4,40,27.6	S_{10}	0.6,8.4,40,21.8
S_{11}	0.66,7.4,40,20.2	S_{12}	0.6,6.2,40,27.2
S_{13}	0.53,8.2,51,30.2	S_{14}	0.66,6,44,29
S_{15}	0.73,5.8,63,40	S_{16}	0.73,5.6,71,41.8

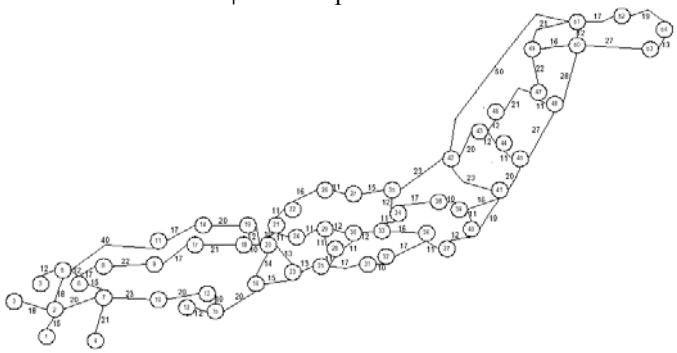


Figure 4. NTT Network.

The parameters of MMA were set to $|P|$=40, R=30 and P_{mut}=0.3. Given that MMA provides several solutions, the one with minimum α_T was finally selected. When there were two or more trees with the same best value for α_T, the one with minimum cost was chosen. For this problem, MMA was compared against the algorithm proposed by Seok et al. [2] (SLCK algorithm), a novel heuristic that solves the dynamic multicast routing problem minimizing α_T, and against the simple shortest path delay tree algorithm (SPT). The following performance figures were used:

1- Normalized maximum link utilization:

$$\alpha_N = (\alpha_W - \alpha_{MMA})/\alpha_{MMA}. \quad (9)$$

2- Normalized total bandwidth consumption:

$$B_N = (B_W - B_{MMA})/B_{MMA}. \quad (10)$$

3- Normalized total delay:

$$D_N = (D_W - D_{MMA})/ D_{MMA}, \quad (11)$$

where

W: SPT or SLCK, depending on which is compared.
α_W: maximum link utilization using W.
α_{MMA}: maximum link utilization using MMA.
B_W: total bandwidth consumption using W.
B_{MMA}: total bandwidth consumption using MMA.
D_W: total delay using W, given by the sum of the total delay of the multicast groups already in the net. The total delay of a group is the sum of the end-to-end delays to the destination nodes.
D_{MMA}: total delay using MMA.

If the average link utilization is defined as:

$$\overline{\alpha} = \frac{1}{|E|} \sum_{(i,j)\in E} \frac{t_{ij}}{z_{ij}} \quad . \quad (12)$$

Then, for the particular case in which the capacity of the links are identical, (10) also gives the normalized value of $\overline{\alpha}$. Besides the above performance figures, the following dominance metrics were taken into account:

U_{MMA}: Number of trees constructed with MMA that dominates the corresponding SLCK*'* trees.
U_{SLCK}: Number of trees constructed with SLCK that dominates the corresponding MMA*'* trees.
I: Number of indifference relationships. This occurs when the trees constructed by MMA and SLCK are non-comparables.

SLCK was set up with three values of H, an a priori parameter of this algorithm [2]. The parameters of MMA given for the two problems were chosen for experience.

VII. Results

A. Test Problem 1

The minimum, maximum and average theoretical optimal solutions found by the runs using MMA were 12, 16 and 13.54 respectively. The mean time consumed to build a tree was 80 ms. Clearly MMA has a good performance finding the Pareto Set, given that it finds at least 75% of the Pareto Front. In this way, MMA not only finds an optimal solution, but also a set of Pareto solutions. This feature is a very special characteristic, since the most adequate solution can be chosen for each particular case without a priori restrictions.

B. Test Problem 2

All the requests were accepted using MMA and SLCK, while seven were rejected for lack of link capacity when SPT was used. The mean time consumed to construct a multicast tree by MMA was 120 ms. Table III shows the values of U_{MMA}, U_{SLCK} and I. Note that, even in the best case of SLCK (H=3), 276 trees constructed by it were beaten by those of MMA. This means that MMA´s trees were clearly better than SLCK´s trees in almost 70% of the total requests (276 of 400), since they won in at least one objective without being beaten in anyone. Table III also shows that MMA and SLCK constructed non-comparables trees in some cases.

TABLE III. Comparison between MMA and SLCK

	U_{MMA}	U_{SLCK}	I
H=3	276	19	105
H=6	338	9	53
H=9	364	3	33

From Table III, the fact that MMA constructed better trees than SLCK should lead to a better load balancing and bandwidth consumption. Figs. 5(a) and (b) confirm this. Fig. 5(a) shows that maximum link utilization of the network when SLCK was used reaches values of almost 60% greater than those of MMA in some times, while in others oscillates between values of -20 and 40% greater. Fig. 5(b) shows that SLCK approximately uses between 20 and 60 % more bandwidth resources than MMA. This also means that load over the network is better balanced when MMA is used, since B_N gives $\overline{\alpha}$ so. Similarly, (c) shows that total delay is also lower when MMA is used. These mean that MMA not only balances the load better than SLCK, but also in addition is able to consume less bandwidth resources and to produces lower delay trees at the same time.

Using SPT, seven traffic requests were turned down. Figures 5 (d) to (f) are the normalized values for SPT. Fig 5(d) shows that SPT has a poor performance respect to MMA when α is considered. SPT was also clearly beaten in bandwidth consumption. As expected, Fig. 5(f) shows that using SPT the total delay was minimized.

VIII. Conclusions

This paper presents an improved approach of MMA to solve the multicast routing problem. MMA is able to optimize four objective functions simultaneously: 1- maximum link utilization, 2- cost of a tree, 3- maximum end-to-end delay and 4- average delay. MMA has a purely multiobjective approach, based on SPEA. This approach calculates not only one solution, but also an optimal Pareto set of solutions in only one run. This last feature is especially important, since the most adequate solution can be chosen for each particular case without a priori restrictions. Experimental results showed that MMA was able to find Pareto optimal solutions. They also showed that for the dynamic multicast routing problem, MMA produced better solutions than SLCK in almost 70% of the traffic request (with H=3), leading to a better load balancing over the network and consuming less bandwidth resources. At the same time, the trees of MMA had lower average delay than those of SLCK. The simulations also show that, when compared against SPT, MMA was able to balance the load far better and to consume less bandwidth resource.

In the future, the authors will consider a traffic engineering scheme using different distribution trees. At the

same time, more tests over other network topologies will be performed.

REFERENCES

[1] C. Ravikumar, and R. Bajpai, "Source-based delay bounded multicasting in multimedia networks," Computer Communications, vol. 21, 1998, pp. 126-132.

[2] Y. Seok, Y. Lee, Y. Choi, and C. Kim, "Explicit multicast routing algorithm for constrained traffic engineering," IEEE 7th International Symposium on Computer and Communications (ISCC´02), Italy, 2002.

[3] Y. Donoso, R. Fabregat, and J. Marzo, "Multi-objective optimization algorithm for multicast routing with traffic engineering," IEEE 3rd International Conference on Networking (ICN´2004), Guadeloupe, French Caribbean, March - 2004.

[4] J. Crichigno, and B. Baran, "Multiobjective multicast routing algorithm," IEEE 11th International Conference on Telecommunications (ICT'2004), Brazil, August - 2004.

[5] P. de Araujo, and G. Barbosa, "Multicast routing with quality of service and traffic engineering requirements in the Internet, based on genetic algorithm," Proceedings of the VII Brazilian Symposium on Neural Networks (SBRN´02), 2002.

[6] E. Zitzler, and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," IEEE Trans. Evolutionary Computation, vol. 3, nº 4, pp. 257-271, 1999.

[7] V. Kompella, J. Pasquale y G. Polyzos, "Multicast routing in multimedia communication," IEEE/ACM Transactions on Networking, vol. 1 nº 3, 1993, pp. 286-291.

[8] W. Zhengying, S. Bingxin, and Z. Erdun, "Bandwidth-delay-constraint least-cost multicast routing based on heuristic genetic algorithm," Computer Communications, Vol. 24, 2001, pp. 685-692.

[9] J. Yen, "Finding the k shortest loopless path in a network," *Management Science*, 17:712-716, 1971.

[10] D. Goldberg, Genetic Algorithm in Search, Optimization & Machine Learning, Addison Wesley, 1989.

[11] B. Barán, and R. Sosa, "A new approach for Antnet routing," IEEE International Conference on Computer Communication and Networks (ICCCN'2000), USA, 2000.

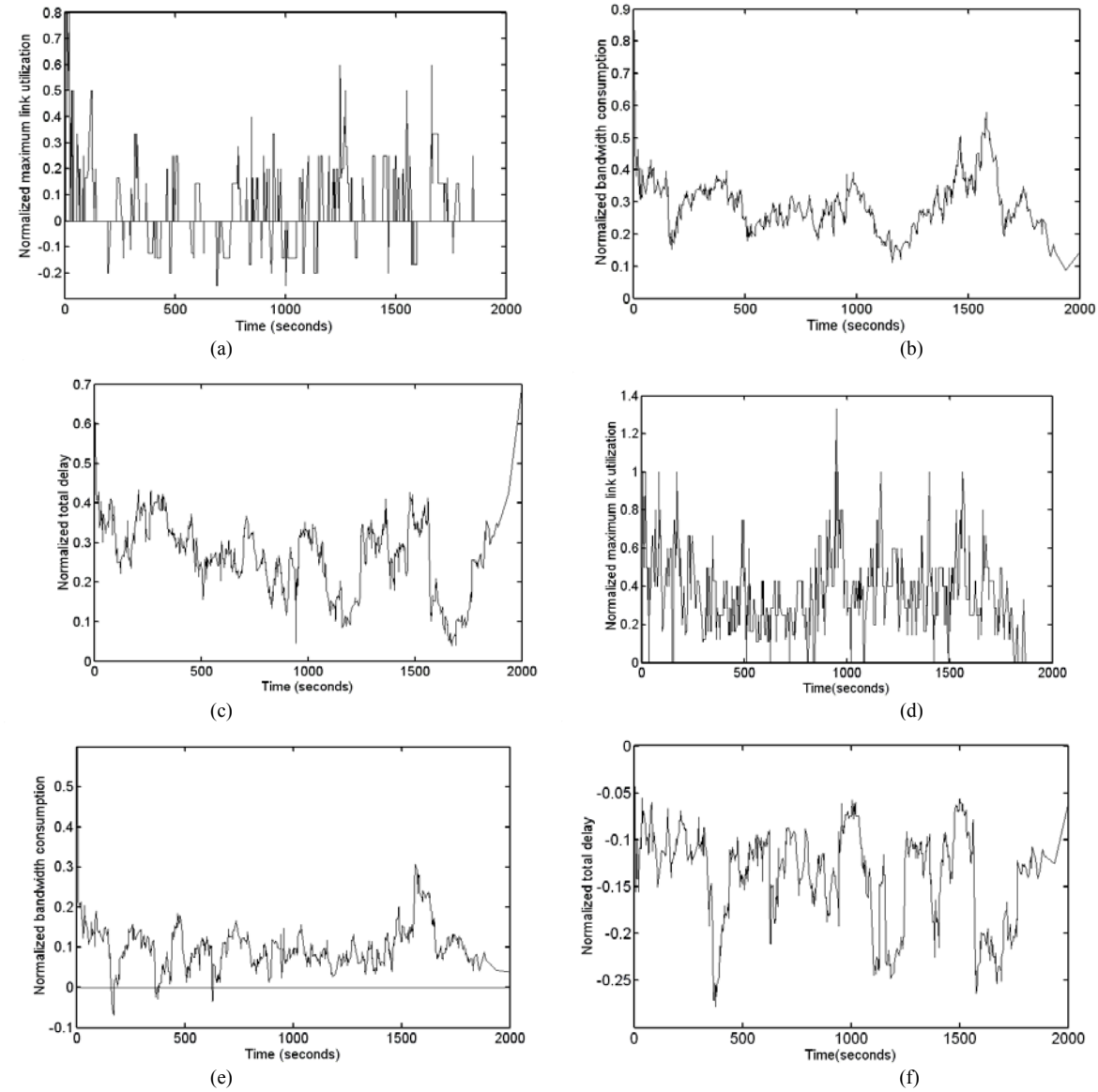


Figure 5. (a), (b) y (c): Normalized values of SLCK with H = 3. (d), (e) y (f): Normalized values of SPT.