Generalized Multiobjective Multitree model solution using MOEA

BENJAMÍN BARÁN[*], RAMON FABREGAT[+], YEZID DONOSO[±],
FERNANDO SOLANO[+] and JOSE L. MARZO[+]
[*] CNC. National University of Asuncion (Paraguay)
[+] IIiA. Universitat de Girona (Spain)
[±] Universidad del Norte (Colombia)
bbaran@cnc.una.py, {ramon, fsolanod, marzo}@eia.udg.es, ydonoso@uninorte.edu.co

Abstract - In a previous paper a novel *Generalized Multiobjective Multitree model* (GMM-model) was proposed. This model considers for the first time multitree-multicast load balancing with splitting in a multiobjective context, whose mathematical solution is a whole Pareto optimal set that can include several results than it has been possible to find in the publications surveyed. To solve the GMM-model, in this paper a multi-objective evolutionary algorithm (MOEA) inspired by the Strength Pareto Evolutionary Algorithm (SPEA) is proposed. Experimental results considering up to 11 different objectives are presented for the well-known NSF network, with two simultaneous data flows.

Key-Word: - Multicast, Splitting, Traffic Engineering, Load Balancing, Multiobjective.

1 Introduction

Traffic engineering (TE) is concerned with improving the performance of operational networks, usually taking into account QoS (Quality of Service) requirements. The main objectives are to reduce congestion hot spots, improve resource utilization and provide adequate QoS for final users. These aims can be achieved by setting up explicit routes through the physical network in such a way that the traffic distribution is balanced across several traffic trunks, giving the best possible service, i.e., minimum delay, packet losses, jitter, etc.

When load balancing techniques are translated into a mathematical formulation, a heuristic or a practical implementation, different conflicting objectives are found and hence they have been considered in the literature as minimizing: maximum or average link utilization, maximum, average and / or total hop count, maximum, average and / or total delay, bandwidth consumption, flow assignation, packet loss, queue size, number of Label Switching Paths (LSPs) in a Multi-Protocol Label Switching (MPLS) implementation, jitter and different cost functions. Clearly, when all these objectives are considered, it can be seen that the problem is multiobjective, as already recognized by several authors [2, 3, 7, 8]. It should be noted that in a pure multi-objective context, no objective needs to be considered as more important than the others, no a priori weighting of the objectives is needed and no a priori constraint on any objective is necessary; therefore, the solution of a Multiple-Objective Problem (MOP) is usually a whole set of optimal compromised solutions, known as a Pareto set [10]. Approximating the set of Pareto instead of locating a single solution allows a decision marker to consider only the reduced set of alternatives, to see the trade-offs among the objectives and to determine preferences among the conflicting objectives.

One interesting solution to the balancing alternative is the multipath approach, in which data is transmitted through different paths to achieve an aggregated, end-to-end bandwidth requirement. Several advantages of using multipath routing are discussed in [6]. Links do not get overused and therefore do not get congested, and so they have the potential to aggregate bandwidth, allowing a network to support a higher data transfer than is possible with any single path. Furthermore, some authors have expanded this idea by proposing to split each flow into multiple subflows in order to achieve better load balancing [4, 11]. For a load balancing model to be general, unicast considerations are not enough and multicast should also be considered, as already proposed in [1, 2, 3, 4, 5, 7, 8, 9].

A *Generalized Multiobjective Multitree model* (GMM-model) proposed in previous work [13] is commented in section 2. A *Multi-Objective Evolutionary Algorithm* (MOEA) which is able to solve the proposed GMM-model is presented in section 3, given its recognized ability for solving MOPs [2, 3, 10, 12]. Moreover, GMM-model considers a multitree-multicast load balancing problem with splitting in a multiobjective context, for the first time, using an evolutionary approach. To illustrate the resolution of the

GMM-model using a MOEA, section 4 presents experimental results. Final conclusions and future work are left for section 5.

2 Generalized multiobjective multitree model

In previous work we have proposed a *Generalized Multiobjective Multitree model* (GMM-model) [13] that considers simultaneously for the first time, multicast flow, multitree, and splitting.

The proposed GMM-model considers a network represented as a graph $G(\mathbf{N}, \mathbf{E})$, with **N** denoting the set of nodes and **E** the set of links. The set of flows is denoted as **F.** Each flow $f \in \mathbf{F}$ can be split into $\mathbf{K_f}$ subflows that after normalization can be denoted as f_k; $k = 1, \ldots |\mathbf{K_f}|$. In this case, f_k indicates the fraction of $f \in \mathbf{F}$ it transports, and $\sum_{k=1}^{|K_f|} f_k = 1$. For each flow $f \in \mathbf{F}$ we have a source $s_f \in \mathbf{N}$ and a set of destination or egress nodes $\mathbf{T_f} \subset \mathbf{N}$. Let t be an egress node, i.e. $t \in \mathbf{T_f}$. Let $X_{ij}^{f_k t}$ denote the fraction of subflow f_k to egress node t assigned to link $(i,j) \in \mathbf{E}$, i.e. $0 \leq X_{ij}^{f_k t} \leq 1$.

GMM-model considers 11 objective functions: Maximal link utilization (ϕ_1), Total hop count (ϕ_2), Hop count average (ϕ_3), Maximal hop count(ϕ_4), Maximal hop count variation for a flow (ϕ_5), Total delay (ϕ_6), Average delay (ϕ_7), Maximal delay (ϕ_8), Maximal delay variation for a flow (ϕ_9), Total bandwidth consumption (ϕ_{10}), and number of subflows (ϕ_{11}). Moreover, considers 7 constraints: Flow conservation constraints for every source node, for every destination and for every other node; a subflow uniformity constraint, to ensure that a subflow f_k always transports the same information; link capacity constraint; constraint on the maximum number of subflows.

At this point, it is important to point out that the mathematical solution of the proposed GMM-model is a complete set **X*** of Pareto optimal solutions $x^* \in \mathbf{X^*}$, i.e. any solution x' outside the Pareto set ($x' \notin \mathbf{X^*}$) is outperformed by at least one solution x^* of the Pareto set ($\exists x^* \succ x'$); therefore, x' can not outperform x^* even if not all the objective functions are considered. Consequently, under the same set of constraints, any previous model or algorithm, that only considers a subset of the proposed objective functions, either as a SOP or MOP, can find one or more solutions calculated with the GMM-model or dominated by solutions $x^* \in \mathbf{X^*}$ of this model.

In conclusion, by using the GMM-model it is possible to calculate the whole set of optimal Pareto solutions. This includes any solution that has been previously found using most of the already published alternatives that consider any subset of the proposed objective functions. Now it may be clear why we call this model *generalized.*

3 GMM resolution using a MultiObjective Evolutionary Algorithm

To solve the GMM-model, a Multiobjective Evolutionary Algorithm (MOEA) approach has been selected because of its well-recognized advantages when solving MOPs in general and TE load balancing in particular [2, 3, 7, 8]. A MOEA, as a genetic algorithm, is inspired by the mechanics of natural evolution (based on the survival of the fittest species).

At the beginning, an initial population of P_{max} feasible solutions (known as individuals) is created as a starting point for the search. In the next stages (or generations), a performance metric, known as fitness, is calculated for each individual. In general, a modern MOEA calculates fitness considering the dominance properties of a solution with respect to a population. Based on this fitness, a selection mechanism chooses good solutions (known as parents) for generating a new population of candidate solutions, using genetic operators like crossover and mutation. The process continues iteratively, replacing old populations with new ones, typically saving the best found solutions (which is known as elitism), until a stop condition is reached.

In this paper, an algorithm based on the *Strength Pareto Evolutionary Algorithm* (SPEA) [12] is proposed. It holds an evolutionary population **P** and an external set $\mathbf{P_{nd}}$ with the best Pareto solutions found. Starting with a random population **P**, the individuals of **P** evolve to optimal solutions that are included in $\mathbf{P_{nd}}$. Old dominated solutions of $\mathbf{P_{nd}}$ are pruned each time a new solution from **P** enters $\mathbf{P_{nd}}$ and dominates old ones.

3.1 Encoding

Encoding is the process of mapping a decision variable **x** into a chromosome (the computational representation of a candidate solution). This is one of the most important steps towards solving a TE load balancing problem using evolutionary algorithms. Fortunately, it has been sufficiently studied in current literature. However, it should be mentioned that to our best knowledge, this paper is the first one that proposes an encoding process, as shown in Fig. 1, that allows the representation of several flows (unicast and / or multicast) with as many splitting subflows as needed to optimize a given set of objective functions.

In this proposal, each chromosome consists of |**F**| flows. Each flow *f*, denoted as *(Flow f)*, contains |$\mathbf{K_f}$| subflows that have resulted from splitting and which flow in several subflows (multitree, for load balancing). Inside a flow *f*, every subflow *(f,k)*, denoted as *(Subflow f,k)*, uses two fields. The first one represents a tree *(Tree f,k)* used to send information about flow *f* to the set of destinations $\mathbf{T_f}$, while the second field represents the fraction f_k of the total information of flow *f* being transmitted.

Moreover, every tree *(Tree f,k)* consists of |$\mathbf{T_f}$| different paths *(Path f,k,t)*, one for each destination $t \in \mathbf{T_f}$. Finally, each path *(Path f,k,t)* consists of a set of nodes N_l between the source node s_f and destination $t \in \mathbf{T_f}$ (including s_f and t). For optimality reasons, it is possible to define *(Path f,k,t)* as not valid if it repeats any of the nodes, because in this case it contains a loop that may be easily removed from the given path to make it feasible. Moreover, in the above representation, a node may receive the same (redundant) information by different paths of the same subflow; therefore, a correction algorithm was implemented to choose only one of these redundant path segments, making sure that any subflow satisfies the optimality criteria.

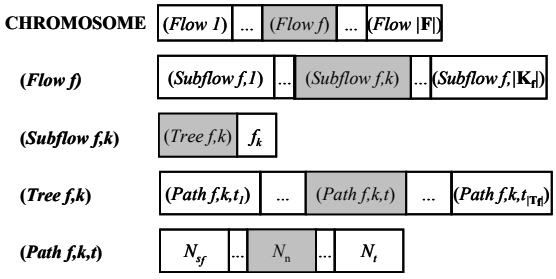

Fig 1. Chromosome representation.

3.2 Initial Population

To generate an initial population **P** of valid chromosomes we have considered each chromosome at a time, building each (*Flow f*) of that chromosome at a time. For each (*Flow f*) we first generate a large enough set of different valid paths from source s_f to each destination $t \in \mathbf{T_f}$ (see line 1 of Fig. 6). Then, an initial |$\mathbf{K_f}$| is chosen as a reasonable random number that satisfies constraints on the maximum number of subflows. To build each of the |$\mathbf{K_f}$| subflows, we randomly generate a tree with its root in s_f and leaves in $\mathbf{T_f}$ by randomly selecting a path at a time for each destination, from the previously generated set of paths.

Trees are conformed by a path-set, which can contain redundant segments; i.e. two paths belonging to a tree with different destinations can meet themselves in more than one node causing redundant subflow information transmission between those pair of nodes. To correct this anomaly, a *repair redundant segments process* is defined: the shortest path of this tree should be taken as a *pattern* and then for each of the remaining paths in the tree, find its shortest segment starting at the latest node (branching node) in the *pattern*. The resulting segment will be a *pattern* segment starting at its source to the branching node joint with the old segment starting at branching node to the destination. Later, an information fraction of f_k=1/|$\mathbf{K_f}$| is initially set.

In this initialization procedure (see lines 2-3 of Fig 6), chromosomes are randomly generated one at a time. A built chromosome is valid (and accepted as part of the initial population) if it also satisfies link capacity; otherwise, it is rejected and another chromosome is generated until the initial population **P** has the desired size P_{max}.

3.3 Selection

Good chromosomes of an evolutionary population are selected for reproduction with probabilities that are proportional to their fitness. Therefore, a *fitness function* describes the "quality" of a solution (or individual). An individual with good performance (like the ones in $\mathbf{P_{nd}}$) has a high fitness level while an individual with bad performance has a low fitness. In this proposal, fitness is computed for each individual, using the well-known SPEA procedure [12]. In this case, the fitness for every member of $\mathbf{P_{nd}}$ is a function of the number of chromosomes it dominates inside the evolutionary set **P**, while a lower fitness for every member of **P** is calculated according to the chromosomes in $\mathbf{P_{nd}}$ that dominate the individual considered. A *roulette selection operator* is applied to the union set of $\mathbf{P_{nd}}$ and **P** each time a chromosome needs to be selected.

3.4 Crossover

We propose two different crossover operators: *flow crossover* and *tree crossover*. With *flow crossover* operator (line 10.a in Fig 6), |**F**| different chromosomes are randomly selected to generate one offspring chromosome that is built using one different flow from each father chromosome, as shown in Fig 2. A father may be chosen more than once, contributing with several flows to an offspring chromosome.

The *tree crossover operator* is based on a two-point crossover operator, which is applied to each selected pair of parent chromosomes (line 10.b in Fig 6). In this case, the crossover is applied by doing tree exchanges between two equivalent flows of a pair of randomly selected parent-chromosomes, as shown in Fig. 3.

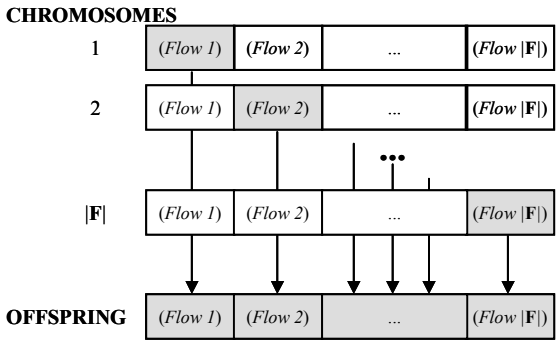

Fig 2. Flow crossover operator.

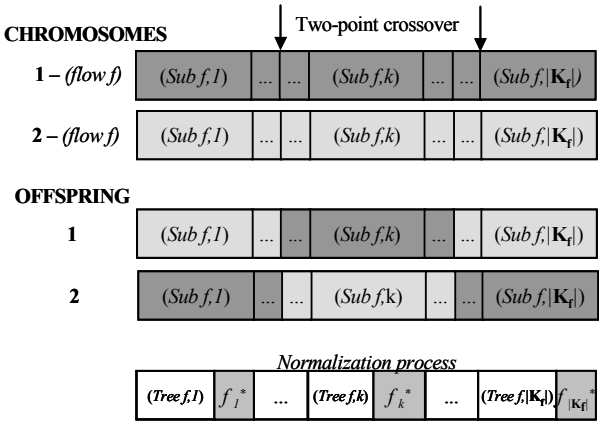

Fig 3. Tree crossover operator.

Tree crossover without normalization of the information fraction f_k usually generates infeasible chromosomes. Therefore, a normalization process $f_k^* = f_k \Big/ \sum_{k=1}^{|K_f|} f_k$,is used as a last step of a tree crossover operator.

3.5 Mutation

This operator could improve the performance of an evolutionary algorithm, given its ability to continue the search of global optimal (or near optimal) solutions even after local optimal solutions have been found, not allowing the algorithm to be easily trapped in local sub-optimal solutions. Each time that an offspring chromosome is generated, a (generally low) mutation probability p_m is used to decide if the mutation operator should be applied to this chromosome (line 11 in Fig 6).

To apply a mutation operator, we first randomly choose a *(Flow f)* of the new offspring, in order to later select (also randomly) a *(Subflow f,k)* on which the mutation will actually apply; therefore, what we implement is a subflow mutation operator. For this work, we propose a subflow mutation operator with two phases: *segment mutation* and *subflow fraction mutation*.

For *segment mutation phase*, a (*Path f,k,t*) of (*Tree f,k*) is randomly chosen. At this point, a node N_j of (*Path f,k,t*) is selected as a *Mutation Point.* The s*egment mutation phase* consists in finding a *new segment* to connect the selected *Mutation Point* to destination *t* (see Fig 4), followed by the already explained (see *B - Initial Population*) *repair redundant segment process*, to achieve better chromosome quality.

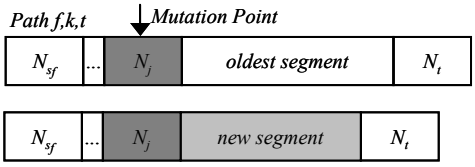

Fig 4. Segment mutation.

Finally, the *subflow fraction mutation phase* is applied to *(Subflow f,k)* by incrementing (or decrementing) flow fraction f_k in δ (see Fig 5), followed by the *normalization process* that has already been explained.

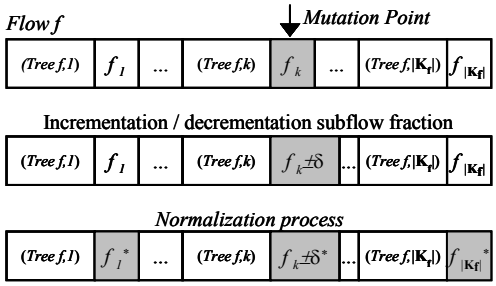

Fig 5. Subflow fraction mutation following by normalization process.

1	Obtain initial set of valid paths
2	Generate the initial population **P** of size P_{max}
3	Normalize fractions and remove redundant segments for every chromosome in **P**
4	Initialize set $\mathbf{P_{nd}}$ as empty
5	DO WHILE A FINISHING CRITERIOM IS NOT SATISFIED {
6	Add non-dominated solutions of **P** into $\mathbf{P_{nd}}$
7	Remove dominated solutions in $\mathbf{P_{nd}}$
8	Calculate fitness of individuals in **P** and $\mathbf{P_{nd}}$
9	REPEAT P_{max} times {
10	Generate new chromosomes-set **C** using a Tree crossover with Selection (in **P** ∪ $\mathbf{P_{nd}}$) and *normalization process* b Flow crossover with Selection (in **P** ∪ $\mathbf{P_{nd}}$)
11	With probability p_m mutate set **C** and *normalization process* and remove redundant segments
12	Add to **P** valid chromosomes in **C** not yet included in **P**
13	END REPEAT
14	END WHILE

Fig 6: Proposed MOEA

4 Experimental Results

Although the aim of this work is to present a general model and not to discuss the best way to solve it, this section presents a simple problem and the corresponding

experimental results using the proposed MOEA, as an illustration of what has been previously stated.

4.1 Network Topology

The chosen topology is the well-known 14-node NSF (National Science Foundation) network ($|\mathbf{N}|=14$). The costs on the links represent the delays (d_{ij}) and all links are assumed to have 1.5 Mbps of bandwidth capacity (c_{ij} = 1.5 Mbps $\forall(i,j)\in E$). Two flows with the same source, s_f=N_0, are considered. The egress subsets are $\mathbf{T_1}$={N_5, N_9} and $\mathbf{T_2}$={N_4, N_9, N_{12}}. The transmission rates are b_1=256 Kbps for the first flow and b_2=512 Kbps for the second flow.

4.2 Resolution of the Test Problem

A complete set of found non-dominated solutions (best calculated approximation to the optimal Pareto set **X*** in a run) had 748 chromosomes. This large number is due to the large number of objective functions and the fact that the same set of subtrees with different fractions f_k of the flows are considered as different solutions because each one represents another compromise between conflicting objective functions. When needed, a clustering technique, that is included in the implemented SPEA, may be used to reduce the number of calculated non-dominated solutions to a maximum desired number [12]. Since there is no space to present all 748 non-dominated solutions, Table 1 shows some of the best calculated solutions considering one objective function at a time. Each row (identified by an ID given in the first column) represents a non-dominated solution whose chromosome is omitted to save space. The following 11 columns represent the different objective functions. The cells in bold emphasize an optimal objective value. To better exemplify a solution, certain chromosomes were omitted from Table 1.

Some solutions presented in Table 1 are clearly non-dominated because they are the best ones in at least one objective, like the ones with ID=1 with the minimum value of ϕ_1 or ID=10 with the minimum value of ϕ_2 and ϕ_3 to just name a few. However, most solutions are non-dominated because they are different compromise solutions. As an example, solutions from ID=1 to ID=15 are all optimal considering ϕ_4, but each one represents a different compromise between conflicting objective functions.

In this example when a flow is not split into subflows, we potentially need the least amount of LSPs (ϕ_{11}=2) and there is no hop count or delay variations between subflows (ϕ_5=0, ϕ_9=0), as shown in solutions with ID=2, 10, 18 and 20. However, it is possible to have a delay variation (ϕ_9>0) even when there is no hop count variation (ϕ_5=0) if at least one flow is split (ϕ_{11}>2), like the non-dominated solutions with ID=16 and 17.

Table 1
Some calculated Pareto Front solutions

ID	ϕ_1	ϕ_2	ϕ_3	ϕ_4	ϕ_5	ϕ_6	ϕ_7	ϕ_8	ϕ_9	ϕ_{10}	ϕ_{11}
1	16,7%	26	3,3	4	1	288	22,5	47	17	4096	3
2	33,3%	16	3,2	4	0	180	36,0	39	0	4096	2
3	25,0%	26	3,3	4	1	275	20,9	47	17	4608	3
4	33,3%	25	3,1	4	1	277	21,8	45	17	4608	3
5	33,3%	26	3,3	4	1	282	22,1	45	17	4352	3
6	33,3%	25	3,1	4	1	277	21,5	45	17	4864	3
7	33,3%	27	3,4	4	1	305	22,8	47	11	4224	3
8	33,3%	26	3,3	4	1	282	21,9	45	17	4480	3
9	25,0%	25	3,1	4	1	279	22,2	47	17	4352	3
10	33,3%	15	3,0	4	0	163	32,6	39	0	4608	2
11	33,3%	26	3,3	4	1	282	21,8	45	17	4544	3
12	33,3%	26	3,3	4	1	282	22,0	45	17	4416	3
13	25,0%	28	3,5	4	1	315	23,3	47	11	4480	3
14	33,3%	25	3,1	4	1	271	21,4	47	11	4352	3
15	33,3%	26	3,3	4	1	282	22,2	45	17	4288	3
16	50,0%	25	3,6	5	0	245	25,2	47	2	5792	3
17	50,0%	28	4,0	7	0	254	26,5	56	2	5312	3
18	50,0%	19	3,8	5	0	154	30,8	41	0	4864	2
19	33,3%	153	4,8	10	7	1545	7,1	104	77	6176	13

4.3 Correlation Analysis

A correlation analysis between each pair of objective functions was also performed to get an idea of the real necessity of using (or not) that large a number of objective functions. A very large correlation clearly means that if one objective function is optimized another one with a high correlation is also indirectly optimized.

Table 2 presents these correlation values between the 11 objective functions, considering the whole experimental set of 748 non-dominated solutions.

Table 2
Correlations between objective functions

	ϕ_2	ϕ_3	ϕ_4	ϕ_5	ϕ_6	ϕ_7	ϕ_8	ϕ_9	ϕ_{10}	ϕ_{11}
ϕ_1	-0,22	-0,16	-0,24	-0,26	-0,25	0,19	-0,28	-0,19	0,09	-0,20
ϕ_2		0,69	0,77	0,80	1,00	-0,80	0,78	0,78	0,56	0,98
ϕ_3			0,78	0,48	0,66	-0,35	0,62	0,41	0,52	0,59
ϕ_4				0,82	0,76	-0,61	0,90	0,73	0,47	0,72
ϕ_5					0,81	-0,81	0,85	0,93	0,34	0,80
ϕ_6						-0,81	0,79	0,78	0,53	0,98
ϕ_7							-0,69	-0,78	-0,36	-0,85
ϕ_8								0,83	0,40	0,75
ϕ_9									0,33	0,77
ϕ_{10}										0,53

As shown in Table 2 there are very large correlations between some objectives, such as:

- the total hop count (ϕ_2) and total delay (ϕ_6), with a correlation of almost 1, this is easy to understand given that a longer path usually implies more delay;
- the total hop count (ϕ_2) and number of subflows (ϕ_{11}), with a value of 0.98, given that the use of splitting implies the use of multiple-routes and therefore, more links;

- the maximal hop count (ϕ_4) and maximal delay (ϕ_8), with a correlation of 0,90, because the longest path in the hop count normally has the longest delay. Since the same reasoning applies for the minimal path, it is also easy to understand the high correlation of 0,93 for
- the hop count variation (ϕ_5) and delay variation (ϕ_9). Finally,
- the total delay (ϕ_6) and the number of subflows (ϕ_{11}) with 0.98, as a logical consequence of the high correlation of both objective functions with the total hop count (ϕ_2).

More experimental results are needed to make a final conclusion, but it is clear that all objective functions are not really needed at the same time. We have considered them for sake of completeness, just to make sure that an optimal solution from a previous work that consider a given objective will also be a solution of the GMM-model.

5 Conclusions

Any optimal single objective solution, like the ones proposed in several previous papers, would be a solution of the GMM-model or dominated by a Pareto solution of it when all analysed objective functions are simultaneously considered.

To solve the presented GMM-model, a Multi-Objective Evolutionary Algorithm (MOEA) inspired by the Strength Pareto Evolutionary Algorithm (SPEA) has been implemented, proposing new encoding process to represent multitree-multicast solutions using splitting. This MOEA found a set of 748 non-dominated solutions for a very simple multicast test problem based on the well-known NSF network. A correlation analysis of this set of non-dominated solutions was also included, emphasizing that several objective functions are highly correlated and therefore, not really needed for some practical applications.

For future work, we plan to improve MOEA solving more complex problems, considering different topologies and including some objective functions that still haven't been considered, such as *Packet Loss*.

References

[1] Y. Seok, Y. Lee, Y. Choi, C. Kim. "Explicit Multicast Routing Algorithms for Constrained Traffic Engineering". *IEEE ISCC 2002.*

[2] J. Crichigno, B. Barán. "Multiobjective Multicast Routing Algorithm". *IEEE ICT 2004.*

[3] J. Crichigno B. Barán. "Multiobjective Multicast Routing Algorithm for Traffic Engineering". *IEEE ICCCN 2004.*

[4] Y. Donoso, R. Fabregat, J. Marzo. "Multi-Objective Optimization Algorithm for Multicast Routing with Traffic Engineering". *IEEE ICN 2004.*

[5] R. Fabregat, Y. Donoso, J.L. Marzo, A. Ariza. "A Multi-Objective Multipath Routing Algorithm for Multicast Flows. *SPECTS 2004.*

[6] J.C. Chen, and S.H. Chan. "Multipath Routing for Video Unicast over Bandwidth-Limited Networks". *GLOBECOM 2001.*

[7] X. Cui, C. Lin, Y. Wei. A Multiobjective Model for QoS Multicast Routing Based on Genetic Algorithm. *ICCNMC 2003.*

[8] A. Roy, S.K. Das. "QM2RP: A QoS-based Mobile Multicast Routing Protocol using Multi-Objective Genetic Algorithm". *The Journal of Mobile Communication, Computation and Information, Wireless Networks, May 2004.*

[9] Y. Cui, K. Xu, J. Wu. "Precomputation for Multi-constrained QoS Routing in High-speed Networks". *INFOCOM 2003*.

[10] D. A. van Veldhuizen. "Multiobjective Evolutionary Algorithms: Classifications, Analyses and New Innovations". *Ph.D. thesis. Air Force Institute of Technology, 1999.*

[11] C. Kim; Y. Choi; Y. Seok. Y. Lee. "A Constrained Multipath Traffic Engineering Scheme for MPLS Networks". *ICC 2002.*

[12] E. Zitzler and L. Thiele. "Multiobjective Evolutionary Algorithm: A comparative case study and the Strength Pareto Approach". *IEEE Trans. Evolutionary Computation. Vol. 3, No. 4, 1999*.

[13] B. Barán, R. Fabregat, Y. Donoso, F. Solano, JL. Marzo. "Generalized Multiobjective Multitree model". Proceedings of IIiA Research Report, ref. IIiA 04-08-RR, Institut d'Informàtica i Aplicacions, University of Girona. September 2004.