Multiobjective Max-Min Ant System. An application to Multicast Traffic Engineering

Diego Pinto[1,2] and Benjamín Barán[1,2]

[1] National Computing Center, National University of Asunción
P.O. Box 1439 - Paraguay
{dpinto, bbaran}@cnc.una.py
http://www.cnc.una.py
[2] School of Sciences and Technology, Catholic University "Nuestra Señora de la Asunción"
http://www.uca.edu.py

Abstract. Ant Colony Optimization (ACO) has been already established as a practical approach to solve single-objective combinatorial problems. This work proposes a new approach for the resolutions of Multi-Objective Problems (MOPs) inspired in Max-Min Ant System (MMAS). To probe our new approach, a multicast traffic-engineering problem was solved using the proposed approach as well as a Multiobjective Multicast Algorithm (MMA), a multi-objective evolutionary algorithm (MOEA) specially designed for that multicast problem. Experimental results show the advantages of the new approach over MMA considering the quantity and quality of calculated solutions.

1 Introduction

Ant Colony Optimization (ACO) is a metaheuristic proposed by Dorigo et al. that has been inspired in the behavior of natural ant colonies [1]. ACO approach has been successfully applied to the resolution of several combinatorial optimization problems [2]. In ACO, agents called ants try to find good solutions by randomly chosen edges of a constructing path with a probability proportional to a pheromone matrix that saves the knowledge of the system. When good solutions are found, the amount of pheromone in that matrix is increased, incrementing the probability of using that edge/path for the next ants. Then, the search is carried out in the proximity of good solutions, without loosing the exploratory ability, thanks to the random probability of choosing any available edge [1].

The resolutions of Multi-Objective Problems (MOPs) have been largely treated in the technical literature. Most published work treated MOPs using Multi-Objective Evolutionary Algorithms (MOEAs) [3]. However, a few publications have recently solved MOPs applying concepts based on ACO. Mariano and Morales first proposed an ant algorithm for multiobjective situations, considering a different ant colony for each objective [4]. In the same year, Gambardella et al. developed a bi-criteria ant algorithm for the vehicle routing problem (VRP) using two ant colonies - one for each objective [5]. Both previous approaches used a lexicographical order to decide the order of importance of each objective, i.e. no two objectives may have the same importance.

Iredi et al. proposed an approach for bi-criteria optimization based on multiple ant colonies without considering a lexicographical order [6]. In the same year, Schaerer and Barán presented a Multi-Objective Ant Colony System (MOACS) for the vehicle routing problem with time windows [7]. This was the first approach to use only one colony to simultaneously optimize all the objectives without *a priori* restrictions.

On the other hand, the Max-Min Ant System (MMAS) was presented by Stützle and Hoos to solve single objective problems (SOPs) [8]. MMAS is an ACO inspired algorithm, which successfully solves several combinatorial problems such as the Traveling Salesman Problem (TSP). This algorithm is considered one of the best-known ACO thanks to its ability to overcome stagnation by using an upper and a lower level of pheromone [8]. Given the recognized success of this algorithm, this work presents a new Multiobjective algorithm based of MMAS that will be called M-MMAS, or simply M3AS. This new approach simultaneous optimize all objective functions without *a priori* restrictions, finding a whole Pareto set in only one run of the algorithm.

To verify the performance of the proposed algorithm, a traditional Multi-objective Multicast Routing problem was solved [9]. M3AS was compared to a MOEA specially designed to solve that traffic engineering (TE) problem, the recently published Multiobjective Multicast Algorithm (MMA) [9] based on the Strength Pareto Evolutionary Algorithm (SPEA) [10].

The remainder of this paper is organized as follows. Section 2 gives a general definition of a Multi-Objective Problem (MOP). The Test problem is presented in Section 3. A new Multiobjective Max-Min Ant System (M3AS) is presented in Section 4. The Multiobjective Multicast Algorithm (MMA) is summarized in Section 5. The experimental environment is shown in Section 6. Finally, experimental results are discussed in Section 7 while the conclusions are left for Section 8.

2 Multi-Objective Problem Formulation

A general Multi-objective Optimization Problem [3] 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 $y = f(x) = (f_1(x), f_2(x), \ldots, f_k(x))$.
Subject to $e(x) = (e_1(x), e_2(x), \ldots, e_m(x)) \geq 0$,
where $x = (x_1, x_2, \ldots, x_n) \in X$ is the decision vector,
and $y = (y_1, y_2, \ldots, y_k) \in Y$ is the objective vector.

X denotes the decision space while the objective space is denoted by Y. Depending on the kind of the problem, "*optimize*" could mean minimize or maximize. The set of restrictions $e(x) \geq 0$ determines the set of feasible solutions $X_f \subseteq X$ and its corresponding set of objective vectors $Y_f \subseteq Y$. The problem consists in finding x that optimizes $f(x)$. 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 simultaneously taken into account. 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 $u, v \in X_f$:

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

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

$f(u) < f(v)$ iff $f(u) \leq f(v) \wedge f(u) \neq f(v)$

Then, in a minimization context, they comply with one and only one of three possible conditions:

$u \succ v$ (u dominates v), iff: $f(u)<f(v)$

$v \succ u$ (v dominates u), iff: $f(v)<f(u)$

$u \sim v$ (u and v are non-comparable), iff: $f(u) \nless f(v) \wedge f(v) \nless f(u)$

Alternatively, for the rest of this work, $u \triangleright v$ will denote that u dominates or is equal to v. A decision vector $x \in X_f$ is non-dominated with respect to a set $Q \subseteq X_f$ iff: $u \triangleright v, \forall v \in Q$. When x is non-dominated with respect to the whole set X_f, it is called an optimal Pareto solution; therefore, a *Pareto optimal set* X_{true} may be formally defined as:

$X_{true} = \{x \in X_f \mid x$ is non-dominated with respect to $X_f\}$.

The corresponding set of objective vectors $Y_{true}=f(X_{true})$ constitutes the *Optimal Pareto Front.*

3 The Test Problem

The test problem used in this paper is the Multicast Routing Problem [9] defined as the construction of a Multicast Tree in a computer network to route a given traffic demand from a source to one or more destinations. The computer network topology is modeled as a direct graph $G=(V, E)$, where V is the set of nodes and E is the set of links. The rest of the paper uses the following notation:

$(i, j) \in E$:	Link from node i to node j; $i, j \in V$.
$c_{ij} \in \Re^+$:	Cost of link (i, j).
$d_{ij} \in \Re^+$:	Delay of link (i, j).
$z_{ij} \in \Re^+$:	Capacity of link (i, j), measured in Mbps.
$t_{ij} \in \Re^+$:	Current traffic of link (i, j), measured in Mbps.
$s \in V$:	Source node of a multicast group.
$N_r \subseteq V$-$\{s\}$:	Set of destinations of a multicast group.
$n_i \in N_r$	One of $\|N_r\|$ destinations, where $\|.\|$ indicates cardinality.
$\phi \in \Re^+$:	Traffic demand in Mbps.
$T(s, N_r)$:	Multicast tree with source in s and set of destinations N_r.
$p_T(s, n_i) \subseteq T(s, N_r)$:	Path connecting source s to a destination $n_i \in N_r$. Note that $T(s, N_r)$ represent a solution x in Section 2.
$d(p_T(s, n))$	Delay of path $p_T(s, n)$ given by:

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

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

- Cost of the multicast tree:

$$C(T) = \phi \cdot \sum_{(i,j)\in T} c_{ij} \tag{2}$$

- Maximum end-to-end delay of the multicast tree:

$$DM(T) = \underset{n\in N_r}{Max} \left\{ d\left(p_T(s,n)\right) \right\} \tag{3}$$

- Average delay of the multicast tree:

$$DA(T) = \frac{1}{|N_r|} \cdot \sum_{n\in N_r} d\left(p_T(s,n)\right) \tag{4}$$

The problem is subject to the link capacity constraint $\phi + t_{ij} \leq z_{ij}\ \forall (i, j) \in T(s, N_r)$. For the rest the work $T(s, N_r) \equiv T$ for simplicity.

4 Ant Colony Optimization Approach

Ant Colony Optimization (ACO) is a metaheuristic inspired by the behavior of natural ant colonies [1]. In the last few years, ACO has received increased attention by the scientific community as can be seen by the growing number of publications and the different fields of application [8]. Even though, there are several ACO variants that can be considered, a standard approach is next presented [11].

4.1 Standard Approach

ACO uses a pheromone matrix $\tau = \{\tau_{ij}\}$ for the construction of potential good solutions. The initial values of τ are set $\tau_{ij} = \tau_0\ \forall (i, j)$, where $\tau_0 > 0$. It also takes advantage of heuristic information using $\eta_{ij} = 1/d_{ij}$. Parameters α and β define the relative influence between the heuristic information and the pheromone levels. While visiting node i, N_i represents the set of neighbor nodes that are not visited yet. The probability (p_{ij}) of choosing a node j at node i is defined in the equation (5). At every generation of the algorithm, each ant of a colony constructs a complete solution T using (5), starting at source node s. Pheromone evaporation is applied for all (i, j) according to $\tau_{ij} = (1 - \rho) .\tau_{ij}$, where parameter $\rho \in (0; 1]$ determines the evaporation rate. Considering an elitist strategy, the best solution found so far T_{best} updates τ

according to $\tau_{ij} = \tau_{ij} + \Delta\tau$, where $\Delta\tau = 1/l(T_{best})$ if $(i, j) \in T_{best}$ and $\Delta\tau = 0$ if $(i, j) \notin T_{best}$.

$$p_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{\forall g \in N_i} \tau_{ig}^{\alpha} \cdot \eta_{ig}^{\beta}} & \text{if } j \in N_i \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

4.2 Multiobjective Max-Min Ant System

The standard Max-Min Ant System (MMAS) presented by Stützle and Hoos [8], was derived from the standard ACO and it incorporated three key features to achieve a better performance:

- Only the best solution at every iteration or during the execution of the algorithm updates the pheromone trail τ.
- To avoid stagnation, the possible range of τ components is limited to an interval $[\tau_{min}, \tau_{max}]$. The upper level may be calculated as $\tau_{max} = \Delta\tau/(1\text{-}\rho)$ while the lower level as $\tau_{min} = \tau_{max}/2{\cdot}\omega$, where ω is the quantity of ants at each generation.
- Initialize the pheromone trails by setting τ_0 to some arbitrarily high value, achieving a high exploration at the start of the algorithm.

To solve multiobjective problems, the present work modifies MMAS with the following changes:

- M3AS finds a whole set of Pareto optimal solutions called Y_{know} instead of finding a single optimal solution.
- To guide the ants in the search space, two heuristics information are proposed, inspired in Schaerer and Barán [7]: $\eta^{c}_{ij} = 1/c_{ij}$ and $\eta^{d}_{ij} = 1/d_{ij}$. Parameters λ_c and λ_d define the relative influence between the two heuristics information. Let's consider the k-th ant. Then, the proposed heuristic uses $\lambda_c = k$ and $\lambda_d = (\omega - k+1)$, where $k \in \{1, 2,\ldots, \omega\}$ and ω is the total number of ants. Therefore the probability of choosing a node j at node i is defined as:

$$p_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha} \cdot (\eta_{ij}^{c})^{\lambda_c \beta} \cdot (\eta_{ij}^{d})^{\lambda_d \beta}}{\sum_{\forall g \in N_i} \tau_{ig}^{\alpha} \cdot (\eta_{ig}^{c})^{\lambda_c \beta} \cdot (\eta_{ig}^{d})^{\lambda_d \beta}} & \text{if } j \in N_i \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

- The pheromone level $\Delta\tau_t'$ is now calculated for each solution $T_t \in Y_{know}$, and it is given by:

$$\Delta\tau_t' = \frac{1}{C(T_t) + DM(T_t) + DA(T_t)} \tag{7}$$

- The pheromone matrix τ is updated according to $\tau_{ij} = \tau_{ij} + \Delta\tau_t'$, $\forall (i, j) \in T_t$ and $\forall T_t \in Y_{know}$ up to an upper level $\tau_{max} = \Delta\tau_t'/(1\text{-}\rho)$ and not below a minimum level $\tau_{min} = \Delta\tau_t'/2{\cdot}\omega(1\text{-}\rho)$.

The main algorithm of the new M3AS approach is presented in the Fig. 1 while Fig. 2 details the algorithm to build a solution tree *T*, called *Build Tree*.

Begin *M3AS*
  Read α, β, ρ, ϕ, $(s, N_r), t_{ij}$; $\tau_{ij} = \tau_0 \; \forall (i, j) \in E$; $Y_{know} = \varnothing$;
  Do {
    For k = 1 until ω repeat
      $\lambda_c = k$; $\lambda_d = \omega - k+1$;
      T_k = Build Tree (λ_c, λ_d, α, β, ϕ, τ_{ij}, (s, N_r), t_{ij})
      If ($T_k \in Y_{know}$) then
        $Y_{know} = Y_{know} \cup T_k$ - $\{T_Y | \; T_k$ ¤ $T_Y\} \; \forall T_Y \in Y_{know}$
      End If
    End For
    $\tau_{ij} = (1 - \rho) . \tau_{ij} \quad \forall (i, j) \in E$
    if $\tau_{ij} < \tau_{min}$ *then* $\tau_{ij} = \tau_{min} \quad \forall (i, j) \in E$
    Repeat $\forall T_t \in Y_{know}$
      $\tau_{ij} = \tau_{ij} + \Delta\tau_t' \quad \forall (i, j) \in T$
      if $\tau_{ij} > \tau_{max}$ *then* $\tau_{ij} = \tau_{max} \quad \forall (i, j) \in T_t$
    End Repeat
  } while stop criterion is not verified
  Return Y_{know}
End *M3AS*

Fig. 1. Main M3AS algorithm. Initially it reads the parameters and initializes the pheromone matrix τ and a set Pareto Y_{know}. At each generation, ω solutions T_k are built. The set Y_{know} is updated with non-dominated solutions T_k and dominated solutions of Y_{know} are eliminated. To update pheromone matrix τ, evaporation is first performed pheromones are latter added $\forall (i, j) \in T$ and $\forall T \in Y_{know}$.

Begin *Build Tree*
  Read λ_c, λ_d, α, β, ϕ, τ_{ij}, (s, N_r), t_{ij}; $T = \varnothing$; $N=s$;
  Repeat until found all nodes destinatios of the N_r
    Choose randomly of node i of the N
    Create list of feasible neighbors N_i *to the node i*
    Choose randomly of node j of the N_i *using* p_{ij}
    $T = T \cup (i, j)$; $N = N \cup j$;
  End Repeat
  Prune Tree T /* Not used connections are eliminated
  Return *T*
End *Build Tree*

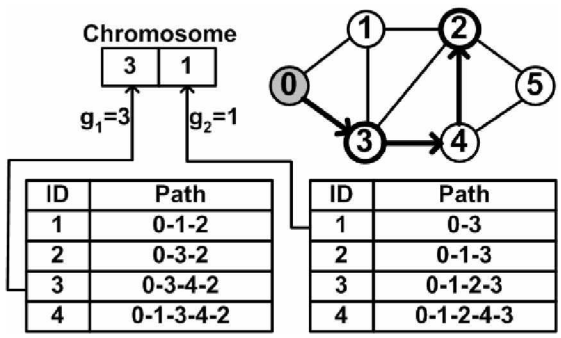
Fig. 2. This algorithm builds a solution *T*. After initialization, the main cycle randomly choose a node *i* from a set *N* of possible departure nodes. A list N_i of feasible neighbors to *i* is then created to randomly choose a node *j* using probability p_{ij} (equation 6). The link *(i, j)* is included in *T* while *j* is included to *N*. When reaching all N_r nodes, the useless links are eliminated and the algorithm is stop.

5 Multiobjective Multicast Algorithm

Multiobjective Multicast Algorithm (MMA), recently proposed in [9], is based on the *Strength Pareto Evolutionary Algorithm* (SPEA) [10]. MMA holds an evolutionary population P and an external Pareto solution set P_{nd}. Starting with a random population P of solutions, the individuals evolve to Pareto optimal solutions to be included in P_{nd}. The pseudo-code of the main MMA algorithm is shown in Fig. 3(a), while its codification is represented in Fig. 3(b).

```
Begin MMA
  Read (s, N_r) , t_ij and φ
  Build routing tables
  Initialize P
  Do {
    Discard individuals
    Evaluate individuals
    Update non-dominated set P_nd
    Compute fitness
    Selection
    Crossover and mutation
  } while stop criterion is not verified
End MMA
```

(a)



(b)

Fig. 3. (a) Pseudo-code of main MMA algorithm (b) Relationship between a chromosome, genes and routing tables for a tree with s=0 and N_r={2, 3}

The MMA algorithm begins reading the variables of the problem and basically proceeds as follows (see pseudo-code in Fig. 3(a)):

Build routing tables: For each $n_i \in N_r$, a routing table is built. It consists of the R shortest and R cheapest paths. R is a parameter of the algorithm. A chromosome is represented by a string of length $|N_r|$ in which each element (gene) g_i represents a path between s and n_i. See Fig. 3(b) to see a chromosome that represents the tree in Fig. 3(b).

Discard individuals: In P, there may be duplicated chromosomes. Thus, new randomly generated individuals replace duplicated chromosomes.

Evaluate individuals: The individuals of P are evaluated using the objective functions. Then, non-dominated individuals of P are compared with the individuals in P_{nd} to update the non-dominated set, removing from P_{nd} dominated individuals.

Compute fitness: Fitness is computed for each individual, using SPEA procedure [10].

Selection: Traditional tournament or roulette methods may be used [12]. To facilitate comparison of the proposed algorithm to MMA [9], a roulette selection operator is applied over the set $P_{nd} \cup P$ to generate the next evolutionary population P.

Crossover and Mutation: MMA uses two-point crossover operator over selected pair of individuals. Then, some genes in each chromosome of the new population are randomly changed (mutated), obtaining a new solution.

The process continues until a stop criterion, as a maximum number of generations, is satisfied.

6 Experimental Environment

Simulations were carried out using the NTT network topology illustrated in Fig. 4 [9]. Two multicast groups shown in Table 1 were used for the presented experiments. For each group, experimental results are analyzed after 160 and 320 seconds. Initially, the network was considered 50% randomly loaded on average, i.e. the initial traffic t_{ij} is around 50% of its total load capacity z_{ij}.

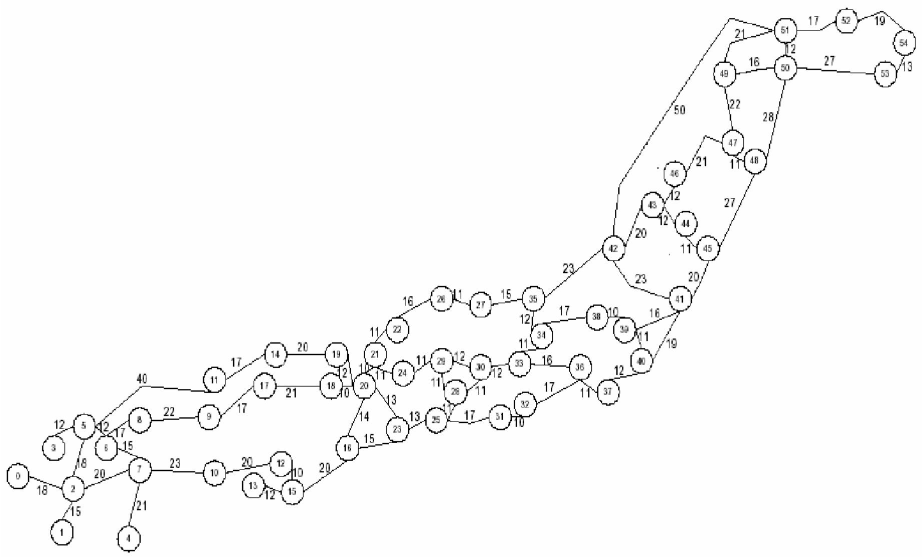


Fig. 4. “Japan NTT network” with 55 nodes and 144 links used for the simulations. Over each link *(i, j)*, a delay d_{ij} is shown.

Two algorithms (*M3AS* and *MMA)* have been implemented on a 350 MHz AMD-K6 computer with 128 MB of RAM. A Borland C++ V 5.02 compiler was used. For these experiments, the results of the proposed *M3AS* were compared to the evolutionary algorithm *MMA* [9], summarized in Section 5.

The MMA parameters were $|P| = 40$, $R = 25$ and $P_{mut} = 0.3$ and the parameters selected for the M3AS were $\alpha = 1$, $\beta = 10$, $\rho = 0.7$, $\tau_0 = 10$ and $\omega = 10$.

Table 1. Multicast Group used for the tests. Each Group has one source and $|N_r|$ destinations.

Test Group	Source {s}	Destinations $\{N_r\}$	$\lvert N_r \rvert$
Group 1 (small)	5	{0, 1, 8, 10, 22, 32, 38, 43, 53}	9
Group 2 (large)	4	{0, 1, 3, 5, 6, 9, 10, 11, 12, 17, 19, 21, 22, 23, 25, 33, 34, 37, 41, 44, 46, 47, 52, 54}	24

To calculate an approximation to the true Pareto Front, Y_{apr}, the following six-step procedure was used:

1. Each algorithm (*M3AS & MMA*) was run five times and an average was calculated for comparison to each other.
2. For each algorithm, five sets of non-dominated solutions: Y_1, $Y_2 \ldots Y_5$, were calculated, one for each run.
3. For each algorithm, an overpopulation Y_T was obtained, where $Y_T = \bigcup_{i=1}^{5} Y_i$.
4. Dominated solutions were deleted from Y_T, obtaining the Pareto Front calculated by each algorithm, as follows:
   - Y_{M3AS} (Pareto Front obtained with five runs, using *M3AS),*
   - Y_{MMA} (Pareto Front obtained with five runs, using *MMA*).
5. A set of solutions $\hat{Y}$ was obtained as: $\hat{Y} = Y_{M3AS} \cup Y_{MMA}$.
6. Dominated solutions were deleted from $\hat{Y}$, and an approximation of Y_{true}, called Y_{apr}, is finally created. Note that for practical issues $Y_{apr} \approx Y_{true}$, i.e. Y_{apr} is an excellent approximation of Y_{true}.

Table 2 presents the total number of solutions $|Y_{apr}|$ that were experimentally found for each multicast group.

Table 2. Total number of non-dominated solutions belonging to Y_{apr} for each multicast group.

	Group 1 (small)	Group 2 (large)
$\|Y_{apr}\|$	9	18

7 Experimental Results

The following tables show a comparison between the solutions found with both implemented algorithms (*M3AS & MMA*) with respect to Y_{apr}. At the same time, both algorithms are compared using the *coverage* figure of merit that counts the average number of solutions dominated by the other algorithm’s Pareto set [3, 10], as shown in tables 3 to 6. To understand those tables, the following notation is used:

$\in Y_{apr}$ average number of solutions found by each algorithm’s runs that are in Y_{apr};

Y_{apr}¤ average number of solutions found by each algorithm’s runs that are dominated by Y_{apr};

$|Y_{algorithm}|$ average number of solutions found by each algorithm;

$\%(\in Y_{apr})$ percentage of solutions found by an algorithm, i.e. $100 \cdot (\in Y_{apr}) / |Y_{apr}|$.

7.1 Experiment 1. Results obtained for a small multicast Group 1

Tables 3 and 4 present experimental results obtained for multicast group 1 after a run of 160 seconds and 320 seconds respectively. Both tables show that *M3AS* found more solutions than *MMA.* The coverage value of zero indicates that no algorithm finds dominated solutions, i.e. all found solutions are non-dominated.

Table 3. Small Multicast Group 1 - Run time = 160 seconds.

	Comparison of Solutions with Y_{apr}				Covering among algorithms	
	$\in Y_{apr}$	Y_{apr} ¤	$\lvert Y_{algorithm}\rvert$	$\%(\in Y_{apr})$	Y_{M3AS}	Y_{MMA}
Y_{M3AS}	8.4	0	8.4	93%		0
Y_{MMA}	5.2	0	5.2	57%	0	

Table 4. Small Multicast Group 1 - Run time = 320 seconds.

	Comparison of Solutions with Y_{apr}				Covering among algorithms	
	$\in Y_{apr}$	Y_{apr} ¤	$\lvert Y_{algorithm}\rvert$	$\%(\in Y_{apr})$	Y_{M3AS}	Y_{MMA}
Y_{M3AS}	9	0	9	100%		0
Y_{MMA}	5.8	0	5.8	64%	0	

7.2 Experiment 2. Results obtained for large multicast Group 2

Tables 5 and 6 present experimental results obtained for a large multicast Group 2 after a run of 160 seconds and 320 seconds respectively. Both tables show that *M3AS* found more solutions than *MMA.* However, when coverage is considered it is not clear which algorithm finds better solutions in average given that *MMA* looks better at 160 seconds, but *M3AS* finally gets better at 320 seconds. In short, it is not clear which algorithm has a better coverage, but *M3AS* always finds a larger number of solutions; therefore, it could be preferred over *MMA.*

Table 5. Large Multicast Group 2 - Run time = 160 seconds.

	Comparison of Solutions with Y_{apr}				Covering among algorithms	
	$\in Y_{apr}$	Y_{apr} ¤	$\lvert Y_{algorithm}\rvert$	$\%(\in Y_{apr})$	Y_{M3AS}	Y_{MMA}
Y_{M3AS}	4.4	4.8	9.2	24%		0.2
Y_{MMA}	4.2	0.6	4.8	23%	0.3	

Table 6. Large Multicast Group 2 - Run time = 320 seconds.

	Comparison of Solutions with Y_{apr}				Covering among algorithms	
	$\in Y_{apr}$	Y_{apr} ¤	$\lvert Y_{algorithm}\rvert$	$\%(\in Y_{apr})$	Y_{M3AS}	Y_{MMA}
Y_{M3AS}	7.2	2.8	10	40%		0.3
Y_{MMA}	4.4	1.2	5.6	24%	0.1	

7.3 General averages

Table 7 presents general averages of the comparison metrics already defined, considering all performed experiments. It can be noticed that, on average, *M3AS* is superior to *MMA*. In fact, *M3AS* found in average 64.25% of Y_{apr} solutions, while *MMA* just found 42%. Also considering Coverage, *M3AS* looks better given that it dominates more solutions calculated by *MMA*.

Table 7. General averages of comparison figures of merit.

	Comparison of Solutions with Y_{apr}				Covering among algorithms	
	$\in Y_{apr}$	Y_{apr} ¤	$\lvert Y_{algorithm} \rvert$	$\%(\in Y_{apr})$	Y_{M3AS}	Y_{MMA}
Y_{M3AS}	7.25	1.9	9.15	64.25%		0.12
Y_{MMA}	4.9	0.45	5.35	42%	0.1	

8. Conclusions

This work introduced a new approach for the resolutions of Multi-Objective Problems (MOPs) based on Ant Colony Optimization. The proposed algorithm is inspired in one of the best-known single objective *ACO*, the Max-Min Ant System (MMAS) [8]. This new approach called *Multiobjective Max-Min Ant System*, *M3AS* for short, proposed several changes to the original *MMAS* as the ability to find several Pareto solutions in only one run using heuristics information to guide the search of good solutions. Parameters λ_c and λ_d define the relative importance in the heuristic information and they may be different for each ant during the iteration. Also, the pheromone's aggregate level $\Delta\tau$ is redefined to simultaneously consider all objective functions. Finally, the update of pheromone matrix τ is carried out considering all solutions of the current Pareto set instead of using just the last best value.

To validate the new approach, the Multicast Traffic Engineering Problem was solved using the proposed *M3AS* algorithm and a recently published multiobjective evolutionary algorithm specially designed to solve that problem, known as Multiobjective Multicast Algorithm (MMA). Simulations using different networks topologies and multicast groups were performed with both algorithms and the experimental results were compared. In this paper, results were presented for an NTT network topology and two different multicast groups (one small called Group 1 and another large denoted as Group 2). Experimental results proved the proposed *M3AS* algorithm outperformed *MMA* in most simulations, finding more solutions, even though some of those solutions might be dominated by the Pareto set calculated by *MMA*, as shown in Table 5. Moreover, *M3AS* has better overall figures of merit when considering the average values given in Table 7.

In the near future, the authors plan to perform more testing, considering other objective functions and other problems, trying to test the performance of other *ACO* based algorithms in a multiobjective context.

References

1. M. Dorigo and G. Di Caro. "The Ant Colony Optimization meta-heuristic". In *New Ideas in Optimization*, pages 11-32. McGraw Hill, London, UK, 1999.
2. S. Alonso, O. Cordón, I. Fernández de Viana and F. Herrera. "*La Metaheurística de Optimización Basada en Colonias de Hormigas: Modelos y Nuevos Enfoques*". G. Joya, M.A. Atencia, A. Ochoa, S. Allende (Eds.), "*Optimización Inteligente: Técnicas de Inteligencia Computacional para Optimización*", *Servicio de Publicaciones de la Universidad de Málaga*, pages 261-313, 2004.
3. D. A. V. Veldhuizen and G. B. Lamont. "*Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art. Evolutionary Computation*", pages 125-147, 2000.
4. C. E. Mariano and E. Morales. "*MOAQ an ant-Q algorithm for multiple objective optimization problems*". In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, *Proceedings of the Genetic and Evolutionary Computation Conference*, Morgan Kaufmann, vol. 1, pages 894-901, Orlando, Florida, USA, 1999.
5. L. M. Gambardella, E. Taillard, and G. Agazzi. "*MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows*". In D. Corne, M. Dorigo, and F. Glover, *New Ideas in Optimization*, pages 63-76. McGraw-Hill, London, 1999.
6. S. Iredi, D. Merkle, and M. Middendorf. "*Bi-Criterion Optimization with Multi Colony Ant Algorithms*". In Coello C, Corne D, Deb K, Thiele L, Zitzler, *Proceedings of Evolutionary Multi-Criterion Optimization First International Conference*, EMO 2001.Lecture Notes in Computer Science vol. 1993, Springer-Verlag.
7. M. Schaerer and B. Barán. *"A Multiobjective Ant Colony System For Vehicle Routing Problem With Time Windows"*. IASTED International Conference on Applied Informatics, Innsbruck, Austria, 2003.
8. T. Stützle and H. Hoos. "*MAX-MIN Ant System*". Future Generation Computer System. 16(8): pages 889–914, June 2000.
9. J. Crichigno and B. Barán, "*Multiobjective Multicast Routing Algorithm*". IEEE ICT'2004, Ceará, Brasil, 2004.
10. 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, 1999, pages 257-271.
11. M. Guntsch and M. Middendorf. "*A Population Based Approach for ACO*". In Stefano Cagnoni, Jens Gottlieb, Emma Hart, Martin Middendorf, and G¨unther Raidl, *Applications of Evolutionary Computing, Proceedings of EvoWorkshops2002: EvoCOP, EvoIASP, EvoSTim*, Springer-Verlag ,vol. 2279, pages 71–80, Kinsale, Ireland, 2002.
12. D. Goldberg. "*Genetic Algorithm is Search, Optimization & Machine Learning*". Addison Wesley, 1989.