Multi-Objective Multicast Routing based on Ant Colony Optimization

Diego Pinto*  Benjamín Barán* and Ramón Fabregat+

* National Computing Center, National University of Asuncion - Paraguay
+ Science and Technology Department, Catholic University of Asunción - Paraguay
  {dpinto, bbaran}@cnc.una.py
+ Institut d’ Informàtica i Aplicacions - Universitat of Girona. Girona, Spain
  {ramon.fabregat}@udg.es

Abstract. This work presents a new multiobjective algorithm based on ant colonies, which is used in the construction of the multicast tree for data transmission in a computer network. The proposed algorithm simultaneously optimizes cost of the multicast tree, average delay and maximum end-to-end delay. In this way, a set of optimal solutions, know as Pareto set, is calculated in only one run of the algorithm, without a priori restrictions. The proposed algorithm was inspired in a Multi-objective Ant Colony System (MOACS). Experimental results prove the proposed algorithm outperforms a recently published Multiobjective Multicast Algorithm (MMA), specially designed for solving the multicast routing problem.

Keywords: Evolutionary Algorithms, Traffic Engineering, Multicast Routing, Multi-objective Optimization, Pareto Front and Ant Colony Optimization.

1. Introduction

Multicast consists of simultaneous data transmission from a source node to a subset of destination nodes in a computer network. 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. Another important consideration in Traffic Engineering is the cost of the tree, understanding cost as other parameters to be minimized, such as: hop count, bandwidth utilization, and others. In this is way; the Multicast Traffic Engineering Problem should be treated as a Multi-Objective Problem (MOP) [13].

Ant Colony Optimization (ACO) is a meta-heuristic proposed by Dorigo et al. [4] inspired by the behavior of ant colonies. In the last few years, ACO has empirically shown its effectiveness in the resolution of several different NP-hard combinatorial optimization problems. ACO uses a colony of artificial ants, i.e. a set of simple agents that work in a cooperative way and communicate by means of artificial pheromone in the search of better
solutions. Several algorithms based on the ACO approach consider the multicast routing problem as a mono-objective problem, minimizing the cost of the tree under multiple constrains. In [8] Y. Liu and J. Wu propose the construction of a multicast tree, where only the cost of a tree is minimized. On the other hand, Gu et al. consider multiple parameters of Quality of Service as constrains while minimizing the cost of the tree [7]. These algorithms treat the Traffic Engineering Multicast problem as a mono-objective problem with several constrains. The main disadvantage of this approach is the necessity of an a priori predefined upper bound that can exclude good trees from the final solution.

This work proposes for the first time to solve the Traffic Engineering Multicast problem using the Multi-Objective Ant Colony System (MOACS), introduced in [9]. This algorithm optimizes several objectives simultaneously. Experimental results have recently demonstrated that MOACS is the best multi-objective ACO algorithm for the bi-objective Traveling Salesman Problem (TSP) [6].

Besides, to verify the results obtained with the proposed algorithm, it is compared to a Multi-objective Multicast Algorithm (MMA) [3]. MMA is based on the Strength Pareto Evolutionary Algorithm (SPEA) and it simultaneously minimizes three objectives functions for the static case in [1], while in [2] optimizes four objectives for the dynamic case. In summary, this work takes one the finest ant colony multi-objective algorithms, adapting it to the Traffic Engineering Multicast problem.

2. 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:

- $(i,j) \in E$: Link from node $i$ to node $j$; where $i, j \in V$.
- $c_{ij} \in \mathbb{R}^+$: Capacity of link $(i, j)$.
- $d_{ij} \in \mathbb{R}^+$: Delay of link $(i, j)$.
- $s \in V$: Source node of a multicast group.
- $N_r \subseteq V \setminus \{s\}$: Set of destinations of a multicast group.
- $\phi \in \mathbb{R}^+$: Traffic demand, in bps.
- $T(s,N_r)$: Multicast tree with source in $s$ and a set of destinations $N_r$.
- $p_T(s,n) \subseteq T(s,N_r)$: Path connecting a source node $s$ with a destination node $n \in N_r$.
- $d(p_T(s,n))$: Delay of path $p_T(s,n)$, given by the sum of the delays of the path, i.e.:

$$d(p_T(s,n)) = \sum_{(i,j) \in p_T(s,n)} d_{ij}$$ (1)

Using the above definitions, a multicast routing problem may be stated as a MOP [13] that tries to find the multicast tree $T(s,N)$ that simultaneously minimizes the following objectives:

\footnote{For the rest of this work $T = T(s,N_r)$ for further simplicity.}
a. Cost of the tree:  \[ f_1(T) = \phi \cdot \sum_{(i,j) \in T} c_{ij} \] (2)

b. Maximum end-to-end delay:  \[ f_2(T) = \max_{n \in N_r} \{ d(p_f(s,n)) \} \] (3)

c. Average delay:  \[ f_3(T) = \frac{1}{|N_r|} \sum_{n \in N_r} d(p_f(s,n)) \] (4)

Considering two solutions \( T \) and \( T' \), for the same multicast group \((s,N_r)\):
\[
\begin{align*}
    x &= \begin{bmatrix} f_1(T) & f_2(T) & f_3(T) \end{bmatrix} \\
    z &= \begin{bmatrix} f_1(T') & f_2(T') & f_3(T') \end{bmatrix}
\end{align*}
\]
only one of the following three conditions can be given:

- \( x \) dominates \( z \): \( x_i \leq z_i \land x_i \neq z_i \forall i \in \{1,2,3\} \)
- \( z \) dominates \( x \): \( z_i \leq x_i \land z_i \neq x_i \forall i \in \{1,2,3\} \)
- \( x \sim z \): \( x \) and \( z \) are non-comparable

Alternatively, for the rest of this work, \( x \sim z \) will denote that \( x \preceq z \) or \( z \preceq x \). A decision vector \( T \) is non-dominated with respect to a set \( Q \) iff: \( T \preceq T' \), \( T \in Q \).

When \( T \) is non-dominated with respect to the whole domain of feasible solutions, it is called an optimal Pareto solution; therefore, the Pareto optimal set \( X_{true} \) may be formally defined as:
\[
X_{true} = \{ T \in X_f | T \text{ is non-dominated with respect to } X_f \} \quad (6)
\]
The corresponding set of objectives \( Y_{true} = f(X_{true}) \) constitutes the Optimal Pareto Front.

3. Multi-objective Ant Colony Optimization algorithm

The Multi-objective Ant Colony Optimization algorithm (MOACS), proposed in [9], is a generalization of the Ant Colony System (ACS) [5]. This approach uses a colony of ants for the construction of \( m \) solutions \( T \) at every generation. Then, the known Pareto Front \( Y_{know} \) [13] is updated, including all non-dominate solutions. Finally, the pheromone matrix \( \tau_{ij} \) is updated. Figure 1 presents a MOACS general procedure.

Read multicast group \((s,N_r)\) and traffic demand \( \phi \)
Initialize \( \tau_{ij} \)
while stop criterion is not verified
    repeat for \( k=1 \) to \( m \)
        \( T = \) Build Tree (Algorithm 3)
        if \( (T \not\in Y_{know}) \) then
            \( Y_{know} = Y_{know} \cup \{T\} \)
        end if
    end repeat
    Update of \( \tau_{ij} \)
end while

Figure 1. General Procedure of MOACS (Algorithm 1)
The update of pheromone matrix $\tau_{ij}$ depends on the state of $Y_{\text{know}}$. If $Y_{\text{know}}$ was modified, then $\tau_{ij}$ is re-initialized ($\tau_{ij} = \tau_0$) to improve exploration; otherwise, a global update of $\tau_{ij}$ is made using the solutions of $Y_{\text{know}}$ for a better exploitation, as shown in Figure 2.

repeat for every $T \in Y_{\text{know}}$
  repeat for every $(i, j) \in T$
    $\tau_{ij} = (1 - \rho) \tau_0 + \rho \Delta t$
  end repeat
end repeat

Figure 2. Global Update of $\tau_{ij}$ (Algorithm 2)

with:

\[
\Delta \tau = \frac{1}{\sum_{T \in Y_{\text{know}}} (f_1(T) + f_2(T) + f_3(T))}
\]  

(7)

where:

- $f_1(T)$ Normalized cost of $T$, given by equation (2).
- $f_2(T)$ Normalized average delay of $T$, given by equation (3).
- $f_3(T)$ Normalized maximum end-to-end delay of $T$, given by equation (4).
- $\rho \in (0, 1]$ Trail persistence.

An ant begins the construction of a solution in the source $s$. A non-visited node is pseudo-randomly [9] selected at each step. This process continues until all desired destinations are reached. Consider $N$ as the list of possible starting nodes, $N_i$ as the list of feasible neighboring nodes to node $i$, $D_r$ as the set of destinations already reached and $\phi$ as another trail persistence parameter. Figure 3 shows the procedure to find a solution $T$.

Initialize $T$, $N$ and $D_r$
Repeat until ($N = \emptyset \lor D_r = N$) 
  Select node $i$ of $N$ and build set $N_i$
  if ($N_i = \emptyset$) then
    $N = N - i$ /* erase node without feasible neighbor */
  else
    Select node $j$ of $N_i$ /*pseudo-random rule */
    $T = T \cup (i, j)$
    $N = N \cup j$
    if ($j \in D_r$) then
      $D_r = D_r \cup j$ /*node $j$ is node destination*/
    end if
  end if
  $\tau_{ij} = (1 - \phi) \tau_0 + \phi \tau_0$ /*update pheromone*/
end repeat
Prune Tree $T$ /* eliminate not used link*/

Figure 3. Procedure to Build Tree (Algorithm 3)
4. Multi-objective Multicast Algorithm

The Multi-objective Multicast Algorithm (MMA), proposed in [1], is based on the *Strength Pareto Evolutionary Algorithm (SPEA)* [12]. This algorithm maintains an evolutionary population $P$ and an external set of Pareto solutions $P_{nd}$. Starting with a random population, the individuals evolve to the desired solutions, as shown in Figure 4 [1].

```
Read multicast group $(s,N_r)$ and traffic demand $\phi$
Build routing tables
Initialize $P$ and $P_{nd}$
while until stop criterion is not verified
  Discard identical individuals
  Evaluate individuals of $P$
  Update non-dominated set $P_{nd}$
  Compute fitness
  Selection
  Apply crossover and mutation
end while
```

**Figure 4.** General Procedure of MMA (Algorithm 4)

*Build routing tables* is a procedure that builds possible paths from a source $s$ to each destination of a multicast group. It usually selects the $R$ shortest, and $R$ cheapest paths, where $R$ is a parameter of the algorithm. A chromosome is represented by a string of length $|N_r|$ in which an element (gene) $g_i$ represents a path [1], as shown in Figure 5.

```
ID  Path  0-3  0-1  0-1-2-4  0-1-3  0-1-2-4-3
1   2     0-3-2
2   3     0-1-2
3   4     0-1-3-2
```

**Figure 5.** Relationship among a chromosome, genes and routing tables.

*Initialize $P$ and $P_{nd}$* generates $|P|$ chromosomes, where $P$ is an evolutionary population. The best non-dominated solutions found so far is saved in an external set $P_{nd}$. Procedure *Discard identical individuals of $P$* replaces duplicated solutions with new randomly generated solutions, while procedure *Evaluate individuals of $P$* calculates the 3 objectives for each individual.

*Update non-dominated set $P_{nd}$* include in $P_{nd}$ non-dominated solutions of $P$, and it erases any dominated solution of $P_{nd}$. Then, fitness is computed as in [12]. The selection operator is later applied over the set $P \cup P_{nd}$ to generate a new population $P$. Finally, *crossover and mutation* operators are applied using 2-point crossover and changing some genes in each chromosome of the new population.
5. Experimental Results

Experimental tests were carried out using the NTT network [10] consisting of 55 nodes and 144 links. Four tests were performed for the 4 groups presented in Table 1. Each test consists of 3 runs for 40, 160 and 320 seconds. Both algorithms, MOACS and MMA, have been implemented on a 350 MHz AMD-K6 computer with 128 MB of RAM. The compiler used was Borland C++ V 5.02.

Table 1. Multicast Group used for the tests

| Group | s | N_r | |N_r| |
|-------|---|-----|---|-----|
| Group 1 | (5) | {0, 1, 8, 10, 22, 32, 38, 43, 53} | 9 |
| Group 2 | (4) | {0, 1, 3, 5, 9, 10, 12, 23, 34, 37, 41, 46, 52} | 14 |
| Group 3 | (4) | {0, 1, 3, 5, 6, 9, 10, 12, 17, 22, 23, 25, 34, 37, 41, 46, 47, 52, 54} | 19 |
| Group 4 | (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 |

5.1. Comparison Procedure

The comparison procedure used for each multicast group was the following:

a) Each algorithm was run five times to calculate an average.

b) For each algorithm, five sets of non-dominated solutions were obtained \((Y_1, Y_2..Y_5)\) and an overpopulation \(Y_T\) was calculated as the union of the five sets.

c) Dominated solutions were deleted from \(Y_T\), forming the Pareto set of each algorithm:

\[
Y_{MOACS} = \text{(Pareto Front obtained of the 5 runs using MOACS)}
\]

\[
Y_{MMA} = \text{(Pareto Front obtained of the 5 runs using MMA)}
\]

d) A set of solutions \(\hat{Y}\) was obtained as follows:

\[
\hat{Y} = Y_{MOACS} \lor Y_{MMA}
\]  \hspace{1cm} (8)

e) Dominated solutions were eliminated from \(\hat{Y}\), to obtain an approximation of \(Y_{true}\), called \(Y_{apr}\). Table 2 presents the number of solutions \(T \in Y_{apr}\) found for every multicast group.

Table 2. Amount of Optimal Solutions for each Multicast Group.

| \(|Y_{apr}|\) | Group 1 | Group 2 | Group 3 | Group 4 |
|-----------|---------|---------|---------|---------|
|           | 9       | 18      | 24      | 18      |

5.2. Results

The odd tables of each test present the average number of solutions of each algorithm that are in \(Y_{apr}\), denoted as \([\in Y_{apr}]\). The set of solutions that are dominated by \(Y_{apr}\) is denoted as \([Y_{apr}^d]\). The number of found solutions is \(|Y_{alg}|\) and the percentage of solutions present in \(Y_{apr}\) is \([\% (\in Y_{apr})]\). The following steps explain how to read Table 3 considering MMA.

a) Row \(Y_{MMA}\), column \([\in Y_{apr}]\) indicates that 5.8 solutions in average belongs to \(Y_{apr}\).

b) Row \(Y_{MMA}\), column \([Y_{apr}^d]\) indicates that 0 solutions are dominates by \(Y_{apr}\).

\footnote{Note that for practical issues \(Y_{apr} = Y_{true}\), i.e. \(Y_{apr}\) is an excellent approximation of \(Y_{true}\).}
c) Row $Y_{MMA}$, column $[|Y_{alg}|]$ indicates that in average 5.8 solutions were found by MMA.

d) Row $Y_{MMA}$, column $[\%(\in Y_{apr})]$ indicates that MMA finds 64% of $Y_{apr}$ solutions.

The even tables of each experiment present the covering figure among algorithms [11]. Only results for group 1 and group 4 are presented.

Experiment 1. Results for multicast group 1 (see Table 1)

a) In Tables 3, 5 and 7 MOACS finds almost all solutions of $Y_{apr}$, overcoming MMA.
b) All found solutions belong to $Y_{apr}$; therefore, the coverings are 0 in Tables 4, 6 and 8.

c) Row $Y_{MMA}$, column $[|Y_{alg}|]$ indicates that in average 5.8 solutions were found by MMA.
d) Row $Y_{MMA}$, column $[\%(\in Y_{apr})]$ indicates that MMA finds 64% of $Y_{apr}$ solutions.

The even tables of each experiment present the covering figure among algorithms [11]. Only results for group 1 and group 4 are presented.

Experiment 1. Results for multicast group 1 (see Table 1)

a) In Tables 3, 5 and 7 MOACS finds almost all solutions of $Y_{apr}$, overcoming MMA.
b) All found solutions belong to $Y_{apr}$; therefore, the coverings are 0 in Tables 4, 6 and 8.

c) Row $Y_{MMA}$, column $[|Y_{alg}|]$ indicates that in average 5.8 solutions were found by MMA.
d) Row $Y_{MMA}$, column $[\%(\in Y_{apr})]$ indicates that MMA finds 64% of $Y_{apr}$ solutions.

The even tables of each experiment present the covering figure among algorithms [11]. Only results for group 1 and group 4 are presented.

Experiment 2. Results for multicast group 2 (see Table 1)

a) In this last experiment characterized for a larger number of destinations multicast group, the MOACS also demonstrated to be better than the MMA. In fact, MOACS obtained a larger number of solutions belonging to $Y_{apr}$, for all run times.
b) Notice that MOACS solutions dominate more solutions than the MMA on average for 160 and 320 seconds (Tables 12 and 14); although not at 40 seconds.

Experiment 2. Results for multicast group 2 (see Table 1)

a) In this last experiment characterized for a larger number of destinations multicast group, the MOACS also demonstrated to be better than the MMA. In fact, MOACS obtained a larger number of solutions belonging to $Y_{apr}$, for all run times.
b) Notice that MOACS solutions dominate more solutions than the MMA on average for 160 and 320 seconds (Tables 12 and 14); although not at 40 seconds.

Experiment 4. Results for multicast group 4 (see Table 1)

a) In this last experiment characterized for a larger number of destinations multicast group, the MOACS also demonstrated to be better than the MMA. In fact, MOACS obtained a larger number of solutions belonging to $Y_{apr}$, for all run times.
b) Notice that MOACS solutions dominate more solutions than the MMA on average for 160 and 320 seconds (Tables 12 and 14); although not at 40 seconds.

Experiment 4. Results for multicast group 4 (see Table 1)

a) In this last experiment characterized for a larger number of destinations multicast group, the MOACS also demonstrated to be better than the MMA. In fact, MOACS obtained a larger number of solutions belonging to $Y_{apr}$, for all run times.
b) Notice that MOACS solutions dominate more solutions than the MMA on average for 160 and 320 seconds (Tables 12 and 14); although not at 40 seconds.
6. Conclusions

Ant algorithms proved to be a promising approach to solve the multicast routing problem. Considering the presented experimental results, MOACS is able to find 69.9% of the best solutions in average, while MMA could only find 42.1%. Besides, the $Y_{MOACS}$ has a better coverage then $Y_{MMA}$ proving its capacity to treat this kind of problems.

As future work, we will consider other objective functions, as maximum link uses and experiments with a dynamic environment and other ACO’s versions.

References

