# Global Convexity in the Bi-Criteria Traveling Salesman Problem

Marcos Villagra[1,2], Benjamín Barán[1,2], and Osvaldo Gómez[1,2]

[1] National University of Asuncion
[2] Catholic University of Asuncion
{mvillagra, bbaran}@cnc.una.py, ogomez@illigal.ge.uiuc.edu

**Abstract.** This work studies the solution space topology of the Traveling Salesman Problem or TSP, as a bi-objective optimization problem. The concepts of category and range of a solution are introduced for the first time in this analysis. These concepts relate each solution of a population to a Pareto set, presenting a more rigorous theoretical framework than previous works studying global convexity for the multi-objective TSP. The conjecture of a globally convex structure for the solution space of the bi-criteria TSP is confirmed with the results presented in this work. This may support successful applications using state of the art metaheuristics based on Ant Colony or Evolutionary Computation.

**Key words:** Traveling Salesman Problem, Multi-Objective Optimization, Global Convexity.

## 1 Introduction

Metaheuristics are a class of optimization algorithms that today constitute one of the best options to solve very complex problems. These algorithms try to combine basic heuristic methods in higher level frameworks aimed at efficiently and effectively exploring a search space [1].

The research in the field of metaheuristics has evolved on the basis of trial and error [2], often motivated by the competition for improving the best known solutions for given problems, and not by identifying the reasons for the success and failure of these algorithms.

The Traveling Salesman Problem or TSP has been used as a benchmark problem for the study of many metaheuristics. The topology of the single-objective TSP has been study in [3–6], and the three-objective TSP in [2], for specific instances. In general, all these results suggest that the solution space has a globally convex structure.

Global Convexity is not convexity in the strict sense [2], but may be used to denote the empirical observation that the best local optima are gathered in a small part of the solution space, which hopefully includes the global optimum. Metaheuristics exploit this by concentrating their search in that part of the solution space [2].

This work studies the solution space topology of the bi-objective TSP in a more practical way than the studies carried out in [2], by means of two new metrics, *category* and *range* of a solution. The former, relates a solution with the number of solutions that dominate it; the latter, establishes a hierarchy in the solution space. The whole solution space was studied for random instances with 7, 8, 9 and 10 cities. Then, subsets of the solution space ware analyzed for larger problems with 100 and 150 cities. It is interesting to mention that when global convexity exists, it may be exploited in metaheuristics for multi-objective combinatorial optimization [2]. Global convexity can be used to design good algorithms or to explain the reason of success of well known metaheuristics that make good use of this property, like Ant Colony Optimization (ACO) and Evolutionary Algorithms (EA) [7].

The remainder of this work is organized as follows. Section 2 presents a general definition of a multiple objective problem. The multi-objective TSP is presented in section 3. Global convexity is described in section 4. The theoretical framework and experimental results are explained in section 5. Finally, conclusions and future work are left for section 6.

## 2 Multi-Objective Optimization Problems

A general Multi-Objective 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:

$$\begin{array}{ll} \text{Optimize} & \mathbf{y} = f(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})) \\ \text{Subject to} & \gamma(\mathbf{x}) = (\gamma_1(\mathbf{x}), \ldots, \gamma_m(\mathbf{x})) \geq 0 \\ \text{where} & \mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathbf{X} \quad \text{is the decision vector, and} \\ & \mathbf{y} = (y_1, y_2, \ldots, y_k) \in \mathbf{Y} \quad \text{is the objective vector} \end{array}$$

$\mathbf{X}$ denotes the decision space while $\mathbf{Y}$ is the objective space. Depending on the problem, "optimize" could mean minimize or maximize. The set of restrictions $\gamma(\mathbf{x}) \geq 0$ determines the set of feasible solutions $\mathbf{X}_f \subseteq \mathbf{X}$ and its corresponding set of objective vectors $\mathbf{Y}_f \subseteq \mathbf{Y}$. A multi-objective problem consists in finding $\mathbf{x}$ that optimizes $f(\mathbf{x})$. 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 comes from the fact that there can be conflicting objectives. Thus, a new concept of optimality should be established for MOPs. Given two decision vectors $\mathbf{u}, \mathbf{v} \in X$:

$$\begin{array}{lll} f(\mathbf{u}) = f(\mathbf{v}) & \text{iff} & \forall i \in 1, 2, ..., k: \quad f_i(\mathbf{u}) = f_i(\mathbf{v}) \\ f(\mathbf{u}) \leq f(\mathbf{v}) & \text{iff} & \forall i \in 1, 2, ..., k: \quad f_i(\mathbf{u}) \leq f_i(\mathbf{v}) \\ f(\mathbf{u}) < f(\mathbf{v}) & \text{iff} & f(\mathbf{u}) \leq f(\mathbf{v}) \wedge f(\mathbf{u}) \neq f(\mathbf{v}) \end{array}$$

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

$$\begin{array}{lll} \mathbf{u} \succ \mathbf{v} & (\mathbf{u} \text{ dominates } \mathbf{v}), & \text{iff } f(\mathbf{u}) < f(\mathbf{v}) \\ \mathbf{v} \succ \mathbf{u} & (\mathbf{v} \text{ dominates } \mathbf{u}), & \text{iff } f(\mathbf{v}) < f(\mathbf{u}) \\ \mathbf{u} \sim \mathbf{v} & (\mathbf{u} \text{ and } \mathbf{v} \text{ are non-comparable}), & \text{iff } \mathbf{u} \not\succ \mathbf{v} \wedge \mathbf{v} \not\succ \mathbf{u} \end{array}$$

Alternatively, $\mathbf{u} \rhd \mathbf{v}$ will denote that $\mathbf{u} \succ \mathbf{v}$ or $\mathbf{u} \sim \mathbf{v}$. A decision vector $\mathbf{x} \in \mathbf{X}_f$ is non-dominated with respect to a set $V \subseteq \mathbf{X}_f$ iff: $\mathbf{x} \rhd \mathbf{v}, \quad \forall \mathbf{v} \in V$. When $\mathbf{x}$ is non-dominated with respect to the whole set $\mathbf{X}_f$, it is called an optimal Pareto solution; therefore, the Pareto optimal set $\mathbf{X}_{true}$ may be formally defined as: $\mathbf{X}_{true} = \{\mathbf{x} \in \mathbf{X}_f : \mathbf{x} \text{ is non-dominated with respect to } \mathbf{X}_f\}$. The corresponding set of objective vectors $\mathbf{Y}_{true} = f(\mathbf{X}_{true})$ constitutes the Optimal Pareto Front.

A solution $\mathbf{z}$ is attainable if there exists a solution $\mathbf{x} \in \mathbf{X}_f$ such that $\mathbf{z} = f(\mathbf{x})$. The set of all attainable solutions is denoted as $\mathbf{Z}$. The ideal solution $\mathbf{z}^*$, is defined as $\mathbf{z}^* = (\min f_1(\mathbf{x}), \ldots, \min f_k(\mathbf{x}))$.

## 3 The Multi-Objective TSP

Given a complete, weighted graph $G = (N, E, d)$ with $N$ being the set of nodes, $E$ being the set of edges fully connecting the nodes, and $d$ being a function that assigns to each edge $\langle i, j \rangle \in E$ a vector $d_{ij}$, where each element corresponds to a certain measure (e.g. distance, cost) between $i$ and $j$, then the multi-objective TSP (MOTSP) [8] is the problem of finding a "minimal" Hamiltonian circuit of the graph, i.e., a closed tour visiting each of the $n = |N|$ nodes of $G$ exactly once, where "minimal" refers to the notion of Pareto optimality [8]. In this study, we consider symmetric problems, i.e. $d_{ij} = d_{ji}$ for all pairs of nodes $i, j$.

We will consider the bi-objective TSP:

$$\begin{array}{ll} \text{Minimize} & \mathbf{y} = f(\mathbf{x}) = (y_1 = f_1(\mathbf{x}), y_2 = f_2(\mathbf{x})) \\ \text{subject to} & f(\mathbf{x}) > 0 \\ \text{where} & \mathbf{x} = (\langle 1, 2 \rangle, \langle 2, 3 \rangle, \ldots, \langle n-1, n \rangle, \langle n, 1 \rangle) \in \mathbf{X} \\ \text{and} & \mathbf{y} = (y_1, y_2) = (f_1(\mathbf{x}), f_2(\mathbf{x})) \in \mathbf{Y} \end{array}$$

where $f_1$ and $f_2$ could be considered as the length of the tour, and the time required to traverse it respectively.

We will measure similarity of two solutions $\mathbf{x}$, $\mathbf{x}'$ by the number of common edges $\langle i, j \rangle \in \mathbf{x}, \mathbf{x}'$. On the contrary, the distance $\delta(\mathbf{x}, \mathbf{x}')$ is defined as the number of non-common edges, i.e. $n$ minus the similarity.

## 4 Global Convexity

The structure of the single-objective TSP has been studied by Boese et al. [3,4]. Their results indicate that the cost surface exhibits a globally convex structure, where good solutions are together in a small region of the search space, and the best solutions are located centrally with respect to the others.

In a minimization context, Boese suggested an analogy with a big valley structure, in which the set of local minima appears convex with one central global minimum [4]. Even though there is no standard definition of global convexity, figure 1 gives an intuitive picture of a globally convex structure.

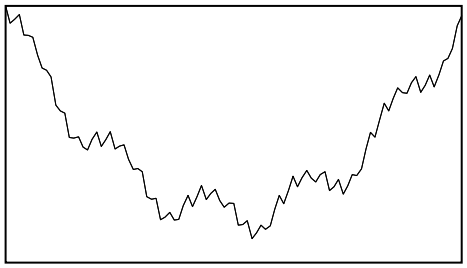


**Fig. 1.** Intuitive picture of the big valley or globally convex solution space structure

The global convexity idea is based on two assumptions [2]:

- *Convexity*: Local optima are gathered in a relatively small region of the solution space.
- *Centrality*: The best local optima are located centrally with respect to the population of local optima.

If both assumptions are valid, we should also expect that local optima are gathered in a small region close to the best local optimum [2]. Besides, any assessment of global convexity only makes sense once a topology has been established in the solution space [2].

Global Convexity has also been studied by Borges and Hansen in [2] for the three-objective TSP, by means of scalarization functions. These results were based on observed behavior rather than on theoretical analysis, and they are not very practical. In fact, Borges and Hansen reduced the multi-objective problem to a single-objective one [2], loosing several characteristics of a truly multi-objective problem, whose theoretical solution is a whole Pareto set and not an ideal solution which is not attainable in practice. Therefore, this work introduces truly multi-objective concepts as *category* and *range*, trying to achieve a more general multi-objective framework. This generalization allows a more rigorous analysis of a MOP for any number of objective functions or measurement units.

## 5 Topological Analysis of the Solution Space

Boese used the length of a tour to study the quality of a solution [4], what is completely valid in a single-objective context. For a MOP, Borges and Hansen proposed the use of scalarization functions that reduce the multi-objective problem to a single-objective one [2].

In what follows, the concepts of *category* and *range* of a solution are presented for the first time as quality metrics, to allow a further topological analysis of the bi-objective TSP.

A population $P = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{|P|}\}$ is defined as a set of valid solutions $\mathbf{x}_i \in \mathbf{X}_f$ of the bi-objective TSP, with cardinality $|P|$.

**Definition 1.** *Let $P \subset \mathbf{X}_f$ be a population, and $\mathbf{x} \in P$ a solution. The category of a solution $\mathbf{x}$ in a population $P$ is defined as:*

$$cat(\mathbf{x}, P) = |\{\mathbf{u} \in P : \mathbf{u} \succ \mathbf{x}\}|$$

Then, the *category* of the solution $\mathbf{x}$ is the number of solutions in $P$ that dominates $\mathbf{x}$. Therefore, a solution of the Pareto front will always have a 0 *category*, i.e. if $\mathbf{u} \in \mathbf{X}_{true}$ then $cat(\mathbf{u}, \mathbf{X}_f) = 0$.

**Definition 2.** *Let $P \subset \mathbf{X}_f$ be a population. The non-dominated frontier of $P$ is defined as:*

$$\mathbf{NF}(P) = \{\mathbf{u} \in P : cat(\mathbf{u}, P) = 0\}$$

*If $P = \mathbf{X}_f$ then $\mathbf{NF}(P) = \mathbf{X}_{true}$.*

**Definition 3.** *Let $P \subset \mathbf{X}_f$ be a population, and $\mathbf{x} \in P$ a solution. The range of a solution $\mathbf{x}$ in a population $P$, denoted as $rng(\mathbf{x}, P)$, is defined according to the following algorithm:*

**if** $\mathbf{x} \in \mathbf{NF}(P)$ **then** $rng(\mathbf{x}, P) = 0$
**else** $rng(\mathbf{x}, P) = 1 + rng(\mathbf{x}, P')$ *where* $P' = P - \mathbf{NF}(P)$

From now on, the use of the parameter $P$ will be omitted from the *range* and *category* notation. Therefore, they will be denoted as $rng(\mathbf{x})$ and $cat(\mathbf{x})$ respectively. The parameter $P$ is left only for ambiguous cases.

A definition of distance is now presented for the study of global convexity in the bi-objective TSP.

**Definition 4.** *Let $P \subset \mathbf{X}_f$ be a population, and $\mathbf{x} \in P$ a solution. The mean distance of a solution $\mathbf{x}$ to a population $P$ is defined as:*

$$\delta(\mathbf{x}, P) = \frac{1}{|P| - 1} \sum_{i=1}^{|P|} \delta(\mathbf{u}, \mathbf{x}) \quad \forall \mathbf{u} \in P.$$

This paper is inspired in Boese's approach [4], where different solutions of an $n$ city problem are saved in a set $P$; consequently, each solution $\mathbf{x}$ has:

- A *category* $cat(\mathbf{x})$.
- A *range* $rng(\mathbf{x})$.
- A mean distance to the other solutions of $P$ denoted as $\delta(\mathbf{x}, P)$.
- A distance to the non-dominated frontier denoted as $\delta(\mathbf{x}, \mathbf{NF}(P))$ (defined in the next section).

This work is divided in two parts. For the first part, small random instances were thoroughly analyzed, and for the second part, analyses based on larger instances from TSPLIB[1] were made.

### 5.1 Exhaustive Study of the Solution Space

The study is based on random generated instances with 7, 8, 9 and 10 cities, named litAB7, omiAB8, encAB9 and asuAB10. These problems are described in [9].

Due to the presence of multiple optimal solutions, a definition of distance to the non-dominated frontier is needed.

**Definition 5.** *Let $P \subset \mathbf{X}_f$ be a population, and $\mathbf{x} \in P$ a solution. The distance of a solution $\mathbf{x}$ to the non-dominated frontier of $P$ is defined as:*

$$\delta(\mathbf{x}, \mathbf{NF}(P)) = min\{\delta(\mathbf{x}, \mathbf{x}_i^*) : \mathbf{x}_i^* \in \mathbf{NF}(P)\} \tag{1}$$

The $e$ best solutions of $P$ will be denoted as $P_{(e)}$; e.g. $P_{(100)}$ denotes the set of the best 100 solutions of $P$, i.e., the 100 solutions with the smallest *category*.

For the calculations, an exhaustive search was made. The obtained population is the whole solution space for an $n$ city problem, i.e. $P = \mathbf{X}_f$, therefore, $|\mathbf{X}_f| = |P| = \frac{(n-1)!}{2}$, and, $\mathbf{NF}(P) = \mathbf{X}_{true}$.

For each solution $\mathbf{x} \in P$, correlations between the following variables were calculated:

- the distance to the non-dominated frontier $\delta(\mathbf{x}, \mathbf{NF}(P))$ and the *category* of a solution $cat(\mathbf{x})$, denoted as $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$;
- the mean distance to the population $\delta(\mathbf{x}, P)$ and the *category* of a solution $cat(\mathbf{x})$ denoted as $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, P))$;
- the distance to the non-dominated frontier $\delta(\mathbf{x}, \mathbf{NF}(P))$ and the mean distance to the population $\delta(\mathbf{x}, P)$ denoted as $\rho(\delta(\mathbf{x}, P), \delta(\mathbf{x}, \mathbf{NF}(P)))$;
- the mean distance to the population $\delta(\mathbf{x}, P)$ and the *range* of a solution $rng(\mathbf{x})$ denoted as $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, P))$;
- the distance to the non-dominated frontier $\delta(\mathbf{x}, \mathbf{NF}(P))$ and the *range* of a solution $rng(\mathbf{x})$ denoted as $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$;
- the *range* and *category* of a solution denoted as $\rho(rng(\mathbf{x}), cat(\mathbf{x}))$.

A summary for these values is shown in tables 1 to 4, and the figures for these correlations can be found in [9].

These results suggest that *range* and *category* are very similar quality metrics, with correlations between them larger than 0.9.

High values can be observed for the correlations $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, P))$ and $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, P))$, which suggests a concentration of very good solutions in the center of the solution space, satisfying the centrality assumption of a globally convex structure. Also, the correlations $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$ and

[1] http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/

**Table 1.** Correlations for the problem litAB7

| | $P$ | $P_{(\lvert P\rvert-1)}$ | $P_{(\frac{\lvert P\rvert}{2})}$ | $P_{(\frac{\lvert P\rvert}{4})}$ |
|---|---|---|---|---|
| $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, P))$ | 0 | 0.602793 | 0.761135 | 0.659006 |
| $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0.767441 | 0.763554 | 0.548112 | 0.585267 |
| $\rho(\delta(\mathbf{x}, P), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0 | 0.582333 | 0.633253 | 0.706702 |
| $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, P))$ | 0 | 0.628991 | 0.752562 | 0.661246 |
| $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0.786593 | 0.783295 | 0.567543 | 0.633579 |
| $\rho(rng(\mathbf{x}), cat(\mathbf{x}))$ | 0.960048 | 0.960550 | 0.937163 | 0.933828 |

**Table 2.** Correlations for the problem omiAB8

| | $P$ | $P_{(\lvert P\rvert-1)}$ | $P_{(\frac{\lvert P\rvert}{2})}$ | $P_{(\frac{\lvert P\rvert}{4})}$ |
|---|---|---|---|---|
| $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, P))$ | 0 | 0.645277 | 0.868166 | 0.782142 |
| $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0.774954 | 0.774225 | 0.627201 | 0.524996 |
| $\rho(\delta(\mathbf{x}, P), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0 | 0.549804 | 0.686872 | 0.555586 |
| $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, P))$ | 0 | 0.644026 | 0.929499 | 0.853578 |
| $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0.810660 | 0.810222 | 0.678969 | 0.594514 |
| $\rho(rng(\mathbf{x}), cat(\mathbf{x}))$ | 0.976814 | 0.976843 | 0.973864 | 0.959215 |

**Table 3.** Correlations for the problem encAB9

| | $P$ | $P_{(\lvert P\rvert-1)}$ | $P_{(\frac{\lvert P\rvert}{2})}$ | $P_{(\frac{\lvert P\rvert}{4})}$ |
|---|---|---|---|---|
| $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, P))$ | 0 | 0.572255 | 0.825523 | 0.697679 |
| $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0.641862 | 0.641802 | 0.435708 | 0.409532 |
| $\rho(\delta(\mathbf{x}, P), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0 | 0.329447 | 0.478431 | 0.371982 |
| $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, P))$ | 0 | 0.564819 | 0.894474 | 0.759983 |
| $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0.672059 | 0.671998 | 0.483163 | 0.492397 |
| $\rho(rng(\mathbf{x}), cat(\mathbf{x}))$ | 0.971607 | 0.971621 | 0.974710 | 0.971266 |

**Table 4.** Correlations for the problem asuAB10

| | $P$ | $P_{(\lvert P\rvert-1)}$ | $P_{(\frac{\lvert P\rvert}{2})}$ | $P_{(\frac{\lvert P\rvert}{4})}$ |
|---|---|---|---|---|
| $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, P))$ | 0 | 0.508309 | 0.830280 | 0.734061 |
| $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0.712957 | 0.712944 | 0.536178 | 0.488927 |
| $\rho(\delta(\mathbf{x}, P), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0 | 0.392366 | 0.577878 | 0.442818 |
| $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, P))$ | 0 | 0.493534 | 0.899061 | 0.803484 |
| $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0.750817 | 0.750809 | 0.583105 | 0.533937 |
| $\rho(rng(\mathbf{x}), cat(\mathbf{x}))$ | 0.970935 | 0.970937 | 0.976167 | 0.975501 |

$\rho(rng(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$, indicate that these solutions are gathered in a relatively small region of the solution space, satisfying the convexity assumption.

As both assumptions are fulfilled, it is expected that these solutions are close to the Pareto front, which is consistent with a globally convex structure conjecture.

The correlations for the bi-objective TSP do not present the high values obtained by Boese for the single-objective case [4]. The reason for this fact is due to the existence of a whole set of Pareto solutions. As a consequence, non-

Pareto solutions could be more central in the solution space than other Pareto solutions.

Larger problems are analyzed in the next section using the same metrics with subsets of the solution space.

### 5.2 Study of a Subset of Solutions

This analysis was based on the TSPLIB instances kroAB100, kroCD100, kroAD100, kroBC100 and kroAB150. From each of these bi-objective TSPs, $n^2$ random samples were taken. The local search algorithm 2-Opt was used for the optimization of each sample set, achieving populations containing local optima solutions. This search strategy was chosen because it presents a simple neighborhood structure, and allows the study of local optima distribution in the solution space.

The methodology of the analysis remains the same, using subsets of the solution space instead of considering the whole solution space. The correlations obtained are shown in table 5.

**Table 5.** Correlations for the instances kroAB100, kroCD100, kroAD100, kroBC100, kroAB150

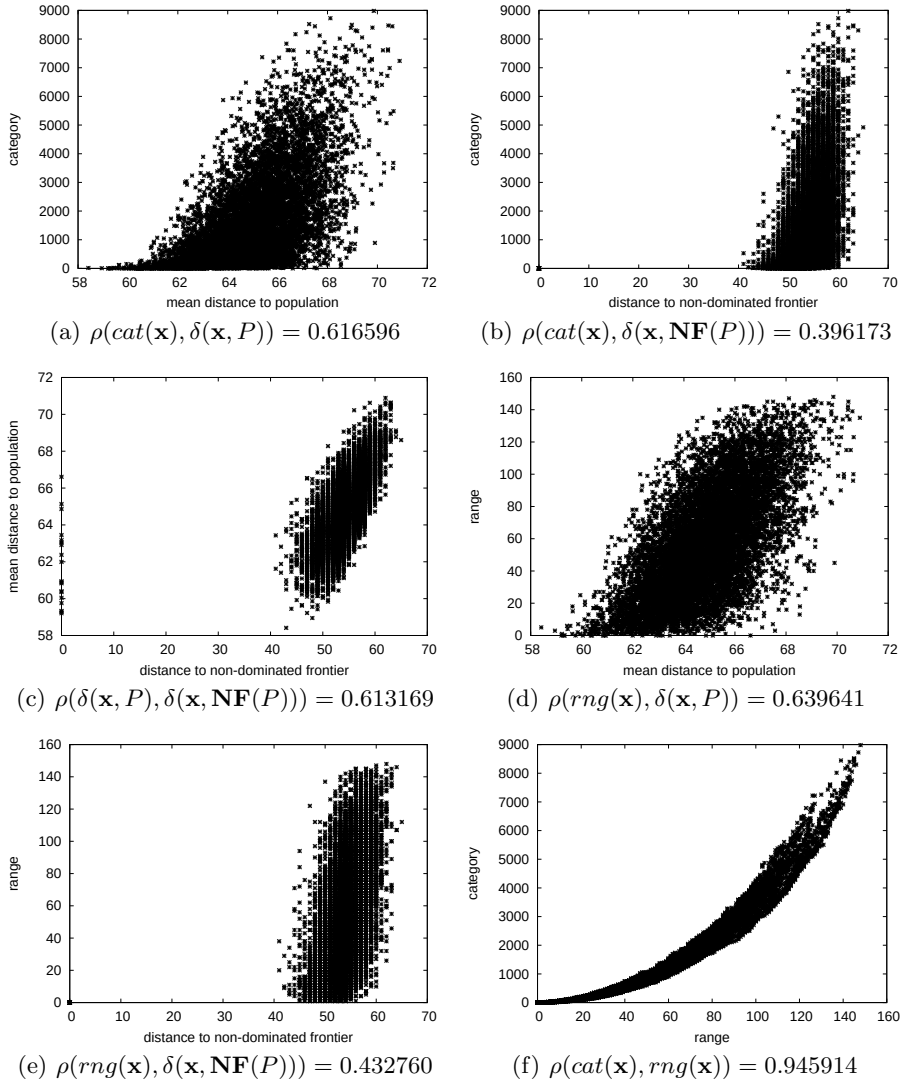
| | kroAB100 | kroCD100 | kroAD100 | kroBC100 | kroAB150 |
|---|---|---|---|---|---|
| $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, P))$ | 0.616596 | 0.596558 | 0.603098 | 0.589122 | 0.571318 |
| $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0.396173 | 0.401700 | 0.391263 | 0.396632 | 0.384648 |
| $\rho(\delta(\mathbf{x}, P), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0.613169 | 0.626019 | 0.611728 | 0.653360 | 0.633580 |
| $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, P))$ | 0.639641 | 0.628951 | 0.645090 | 0.617764 | 0.606404 |
| $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P)))$ | 0.432760 | 0.441982 | 0.449073 | 0.437721 | 0.423266 |
| $\rho(rng(\mathbf{x}), cat(\mathbf{x}))$ | 0.945914 | 0.941865 | 0.937078 | 0.941890 | 0.938438 |

The results for the problem kroAB100 are shown in figure 2. The figures for the other problems can be found in [9]. Correlations between *range* and *category*, still maintain a value larger than 0.9, which confirms their similarity, although *range* presents better results in the whole study.

A concentration of the best solutions centrally with respect to the population is observed in figures 2.a and 2.d. Despite low correlations in the previous figures (around 0.6), figure 2.c shows that solutions located centrally are closer to the non-dominated frontier, and suggests the existence of a globally convex structure. The same results were obtained for the rest of the studied instances.

## 6 Conclusions and Future Work

The concepts of *category* and *range* proved to be very effective quality metrics for the bi-objective TSP, and the generalization of these concepts can be easily made for any instance of the MOTSP. Besides, they can be used in any MOP




(a) $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, P)) = 0.616596$

(b) $\rho(cat(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P))) = 0.396173$

(c) $\rho(\delta(\mathbf{x}, P), \delta(\mathbf{x}, \mathbf{NF}(P))) = 0.613169$

(d) $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, P)) = 0.639641$

(e) $\rho(rng(\mathbf{x}), \delta(\mathbf{x}, \mathbf{NF}(P))) = 0.432760$

(f) $\rho(cat(\mathbf{x}), rng(\mathbf{x})) = 0.945914$

**Fig. 2.** Population of 10,000 optimized solutions for the instance kroAB100

without definition changes. Although these concepts seem to be very similar, *range* showed better results.

*Category*, *range*, mean distance to the population and distance to the non-dominated frontier experimentally demonstrated to be correlated, showing the topological characteristic of global convexity. These metrics could be used for the study of global convexity in other MOPs, where metaheuristics, as ACO or EA, have shown to be very efficient, and for the creation of new metaheuris-

tics that could exploit this type of structure. However, since the results were obtained experimentally, it is not certain that this structure holds for every instance of the bi-objective TSP. Nevertheless, for the single-objective case, no instance was found without a globally convex structure [6].

A problem with a known global convexity structure will allow us to limit the search to a smaller solution area, and from there, it will be possible to use another appropriate algorithms to achieve better approximations to the Pareto set.

There is a lot to do in the study of global convexity, like the creation of metaheuristics based on the exploitation of this structure, the study of instances with correlated objectives, and the use of another kind of neighborhood structure (different than 2-Opt). Also, it can be considered the development of a formal theory for global convexity, and the identification of globally convex problems. Just [3, 4, 6] refers to the subject of global convexity in the TSP, and [2] for the MOTSP.

## References

1. C. Blum, and A. Roli, Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison, ACM Computing Surveys (CSUR), 35(3), 268-308 (2003).
2. P.C. Borges, and P.H. Hansen, A Study of Global Convexity for a Multiple Objective Traveling Salesman Problem, edited by C.C. Ribeiro and P. Hansen, (Essays and Surveys in Metaheuristics, Kluwer, 2000), pp. 129-150.
3. K. Boese, A. Kahng, and S. Muddu, A new Adaptive Multi-Start Technique for Combinatorial Global Optimization, Operations Research Letters, 16, 101-113 (1994).
4. K. Boese, Cost versus Distance in the Traveling Salesman Problem, Computer Science Department, University of California, Technical Report No. 950018, 1995.
5. T. Stützle, and H.H. Hoos, Max-Min Ant System, Future Generation Computer Systems, 16(8), 889-914 (2000).
6. B. Barán, P. Gardel, and O. Gómez, Estudio del Espacio de Soluciones del Problema del Cajero Viajante, edited by M. Solar, D. Fernández and E. Cuadros-Vargas, (CLEI 2004, Arequipa, Perú, 2004), pp. 745-756.
7. O. Gómez, and B. Barán, Reasons of ACO's Success in TSP, edited by M. Dorigo, M. Birattari, C. Blum, L. Gambardella, F. Mondada and T. Stüzle, (Ant Colony Optimization and Swarm Intelligence, 4th International Workshop, ANTS 2004), pp. 226-237.
8. L. Paquete, M. Chiarandini, and T. Stützle, A Study of Local Optima in the Biobjective Travelling Salesman Problem, Computer Science Department, Darmstadt University of Technology, Technical Report No. AIDA-02-07, 2002 (Presented at the Multi-Objective Metaheuristics Workshop, MOMH 2002).
9. M. Villagra, B. Barán, and O. Gómez, Convexidad Global en el Problema del Cajero Viajante Bi-Objetivo, Centro Nacional de Computación, Universidad Nacional de Asunción, Technical Report No. 001/05, 2005 (unpublished); http://www.cnc.una.py.

# Evolutionary Algorithm for State Encoding

Valery Sklyarov, Iouliia Skliarova
University of Aveiro, Department of Electronics and
Telecommunications/IEETA, 3810-193 Aveiro, Portugal
skl@det.ua.pt, iouliia@det.ua.pt
WWW home page: http://www.ieeta.pt/~skl/
http://www.ieeta.pt/~iouliia/

**Abstract**. This paper presents an encoding technique that is common for many different logic synthesis problems. It enables us to construct a system of Boolean functions, and then to decompose this system into sub-systems in such a way that a dependency of functions, included into each sub-system, on the respective arguments is reduced. For complex applications such type of encoding has a high computational complexity and the paper proposes a novel evolutionary algorithm for the solution of this problem.

## 1 Introduction

There are many combinatorial tasks that involve encoding algorithms. These tasks appear in particular at various steps in the logic synthesis of digital circuits. One of these tasks is based on such encoding technique that enables us to construct a system of Boolean functions, and then to decompose this system into sub-systems for which we are able to reduce dependency of the functions, included into each sub-system, on the respective arguments [1,2]. Commonly the logic scheme of a finite state machine (FSM) is composed of a combinational circuit and a memory (a set of flip-flops). The combinational circuit implements a system of Boolean functions $D_1,...,D_R$ that depend on variables $x_1,...,x_L,\tau_1,...,\tau_R$. The $x_1,...,x_L$ are external input variables and $\tau_1,...,\tau_R$ bring the code of the state $a_{from}$ from which we have to carry out transition(s). The functions $D_1,...,D_R$ enable the FSM to calculate the code of the next state $a_{to}$. The lines $\tau_1,...,\tau_R$ are the outputs from the FSM memory and the lines $D_1,...,D_R$ are the inputs to the FSM memory. For example, the FSM can be described as shown in Table 1 (at the beginning let us ignore all symbols enclosed in parenthesis). Here, $a_{from}$ - is an initial state, $K(a_{from})$ and $K(a_{to})$ - are the codes of the states $a_{from}$ and $a_{to}$, respectively, $a_{to}$ - is the next state, $X(a_{from},a_{to})$ - is a product of inputs that forces a corresponding transition. We assume that FSM memory is built from D flip-flops. Let us consider various transitions from the same state. We can see that for all

conditional transitions (i.e. for all transitions except from the state $a_6$), all the sub-functions of $D_1,...,D_3$ that must be activated on transitions from a state, depend on both the state and inputs. We will say that a sub-function is active if it has to be assigned to 1. Since all the sub-functions depend on states and inputs, the relevant Boolean expressions, that are used to calculate the values $D_1,...,D_3$, contain variables from the full set $\{x_1,...,x_L,\tau_1,...,\tau_R\}$, where L - is the number of external inputs (in our example L=5) and R - is the size of the FSM memory (in our example R=3).

**Table 1.** An example of FSM

| $a_{from}$ | $K(a_{from})$ | $X(a_{from},a_{to})$ | $a_{to}$ | $K(a_{to})$ | $D(a_{from},a_{to})$ |
|---|---|---|---|---|---|
| $a_1$ | 000 (000) | $x_1x_2$ | $a_1$ | 000 (**0**00) | - (-) |
| | | not_$x_1$ not_$x_2$ | $a_2$ | 001 (**0**11) | $D_3$ ($D_2$,$D_3$) |
| | | not_$x_1x_2$ | $a_3$ | 010 (**0**01) | $D_2$ ($D_3$) |
| | | $x_1$ not_$x_2$ | $a_6$ | 101 (**0**10) | $D_1$,$D_3$ ($D_2$) |
| $a_2$ | 001 (011) | $x_3$ | $a_3$ | 010 (0**0*1***) | $D_2$ ($D_3$) |
| | | not_$x_3$ | $a_5$ | 100 (1**0*1***) | $D_1$ ($D_1$, $D_3$) |
| $a_3$ | 010 (001) | not_$x_1$ | $a_4$ | 011 (***1***00) | $D_2$,$D_3$ ($D_1$) |
| | | $x_1x_2$ | $a_5$ | 100 (***1***01) | $D_1$ ($D_1$, $D_3$) |
| | | $x_1$ not_$x_2$ | $a_7$ | 110 (***1***11) | $D_1$,$D_2$ ($D_1$, $D_2$, $D_3$) |
| $a_4$ | 011 (100) | not_$x_4$ | $a_1$ | 000 (0**00**) | - (-) |
| | | $x_4$ | $a_4$ | 011 (1**00**) | $D_2$,$D_3$ ($D_1$) |
| $a_5$ | 100 (101) | $x_1$ | $a_1$ | 000 (**000**) | - (-) |
| | | not_$x_1$ | $a_6$ | 101 (**010**) | $D_1$,$D_3$ ($D_2$) |
| $a_6$ | 101 (010) | 1 | $a_7$ | ***111*** | $D_1$,$D_2$,$D_3$ |
| $a_7$ | 110 (111) | $x_5$ | $a_2$ | 001 (0***11***) | $D_3$ ($D_2$, $D_3$) |
| | | not_$x_5$ | $a_7$ | 110 (1***11***) | $D_1$,$D_2$ ($D_1$, $D_2$, $D_3$) |

Consider all sub-functions of $D_1,...,D_3$ that are generated for proper transitions from a state. For example, sub-functions $D_1^3,...,D_3^3$, that have to be activated in transitions from the state $a_3$ (later we will also mark such sub-functions with a corresponding superscript) are the following: $D_1^3 = a_3x_1$; $D_2^3 = a_3$ (not_$x_1 \vee$ not_$x_2$); $D_3^3 = a_3$not_$x_1$ (these expressions can easily be obtained from Table 1). Note that since there exist 3 transitions from $a_3$, they can be distinguished with the aid of just two Boolean variables, such as $D_1^3,...,D_3^3$. As a result, inputs such as $x_1$ and $x_2$ can affect (and change) just two variables from the set $\{D_1^3,...,D_3^3\}$ and the remaining variables (in our example one variable from the set $\{D_1^3,...,D_3^3\}$) can be independent of external inputs from the set $X = \{x_1,...,x_L\}$, i.e. they will only depend on the current state (in our example on the state $a_3$). If the number of different (non coinciding) next states in state transitions from $a_m$ is equal to $q_m$, then (R-int$\log_2 q_m$) variables from the subset $D_1^m,...,D_R^m$ can be independent of the input variables from the set X.

Let us suppose now that the states for our example have been coded as shown in parenthesis in Table 1. The values of the sub-functions $D_1^r,...,D_3^r$ (r=1,2,...,M, M - is

the number of FSM states and for our example M=7) that do not depend on input variables, are marked with bold and italic bold fonts. Note that the bold font has been used for passive values, and italic bold font for active values of the sub-functions.

Now the combinational circuit S of the FSM can be decomposed into two sub-circuits in such a way that the first sub-circuit $S_{ax}$ implements all active values of $D_1^1,...,D_3^M$ that are not bold. The functions of $S_{ax}$ depend on both FSM inputs and states. The second sub-circuit $S_a$ implements all active values of $D_1^1,...,D_3^M$ that are bold. The functions of $S_a$ depend only on the states and for our example they are:

$D_1 = a_3 \vee a_6$; $D_2 = a_6 \vee a_7$; $D_3 = a_2 \vee a_6 \vee a_7$;

Such functions are well suited for minimization and are usually very simple. For instance, $S_a$ can be constructed from just four 2-input logic elements of types XOR, AND and OR, which convert values of $\tau_1,\tau_2,\tau_3$ to values of $D_1,D_2,D_3$. On the other hand, such kind of decomposition enables us to essentially simplify the sub-functions of $S_{ax}$, i.e. the active sub-functions that are not bold in Table 1. Finally, even for our very simple example, the proposed state encoding permits the number of logic elements to be reduced by approximately 20%.

Similar problems appear in a large number of practical applications and we will point out just some of them:

- One-level control circuits based on blocks, such as programmable logic arrays (see [3, p. 182]. It allows reducing essentially the total number of interconnections by eliminating the repeated outputs for different blocks;
- RAM-based implementation of FSMs [2], etc.

The paper presents an evolutionary algorithm that allows the encoding considered above to be achieved. It should be noted that the technique of artificial evolution has been widely used for hardware design [4]. Evolutionary algorithms (EA) are based on a process of "generate-and-test" [5] and this strategy can be applied at different levels. For example, in [6] a genetic algorithm is employed to search for circuits that represent the desired state transition function. Many examples demonstrating EAs that have been successfully employed for hardware design are presented in [7-9]. For some circuits they produced unforeseen results of very high quality (for example, [10]), which have never been obtained by human designers.

The remainder of this paper is organized in four sections. Section 2 presents the detailed description of the proposed evolutionary algorithm. Section 3 discusses feasible variations of the algorithm. Section 4 presents the results of experiments, which clearly demonstrate the advantages of the proposed encoding technique. The conclusion is in section 5.

## 2 Evolutionary Algorithm

The basic idea of EA was used for the considered problem in the traditional simple way [11]. The algorithm includes the following steps:

1. Production of an initial population composed of individuals that represent a set of randomly generated codes for a given number of variables (FSM states).
2. Evaluation of the population and measuring its fitness.

3. Variation of the population by applying such operations as reproduction, mutation and crossover. A reproduction operator makes a copy of the individual (with a probability based on its fitness) for inclusion in the next generation of the population. A mutation operator creates new individuals by performing some changes in a single individual, while the crossover operator creates new individuals (offspring) by combining parts of two or more other individuals (parents) [9].

4. Performing a selection process, where the most fit individuals survive and form the next generation.

Points 2-4 are repeated until a predefined termination condition is satisfied.

Let us assume that a population $\pi$ including $\nu$ individuals $\pi_1,...,\pi_\nu$ has been randomly generated. In order to evaluate each individual $\pi_\iota$, $\iota=1,\ldots,\nu$, it is necessary to specify a fitness function. For our problem it is very easy. Let $A=\{a_1,...,a_M\}$ be a set of variables that have to be encoded, M is the number of variables in the set A. The variables in each individual subset $A(a_{from})$ (where $A(a_{from})$ is a set of states to which there exist direct conditional transitions from the state $a_{from}$, $k_m=|A(a_{from})|>1$, $m=1,...,M$) have to be encoded in such a way that the number $w_m$ ($w_m=R\text{-}int\log_2 k_m$) of their bits with the same indices have equal values. Here $R=int\log_2 M$ is the minimum number of bits in the codes of states assuming binary encoding. Thus, any solution for which the fitness function W is equal to $\Sigma w_m$ ($k_m>1$, $m=1,...,M$) gives an optimal result (we assume that such result exists, which, in fact, is not true for a general case). Actually we can discover several optimal results and for each of them the function W has the same maximum possible value. Any of these results provides the best solution to the problem so we just have to find out the first of them.

Now the fitness can be estimated very easily. For randomly generated codes $\pi_\iota$ we have to calculate the function $W_i$ and compare the result with the value W. The less the difference $W\text{-}W_i$ the better the fitness for the individual $\pi_\iota$.

The next step produces a variation of the population and can be carried out by applying such operations as reproduction, mutation and crossover. Two kinds of reproduction have been examined and compared. The first one is based on elitist rule [11] where the best solutions in the population are certain to survive to the next generation. This rule has been implemented as follows. For reproduction purposes 10% of individuals with the best fitness have been copied to the next generation of the population. The second kind of reproduction uses the same percentage of individuals, but it is based on proportional selection [11].

The mutation operation runs on one parental individual selected with a probability based on fitness and creates one new offspring individual to be inserted into the new population at the next generation. In order to choose which parents will produce offspring, a fitness proportional selection is employed. Each parent $\pi_i$ is assigned a weight $W_i$, calculated at the previous step. The probability of selection for each parent is proportional to its weight. The main idea of the mutation operation will be illustrated by an example of state encoding for FSM with specification presented in [12]. The FSM has 10 states and the following transitions $a_{from} \Rightarrow A(a_{from})$: $a_1 \Rightarrow \{a_2,a_3,a_4\}$, $a_2 \Rightarrow \{a_2,a_4,a_5\}$, $a_3 \Rightarrow \{a_6,a_7,a_8,a_9\}$, $a_4 \Rightarrow \{a_5\}$, $a_5 \Rightarrow \{a_3\}$, $a_6 \Rightarrow \{a_5,a_7\}$, $a_7 \Rightarrow \{a_3,a_9\}$, $a_8 \Rightarrow \{a_2,a_{10}\}$, $a_9 \Rightarrow \{a_{10}\}$, $a_{10} \Rightarrow \{a_1\}$. Since M=10 and R=4 for each individual we can chose any 10 from $2^4$=16 possible codes. Suppose that at some step of EA we found the codes for an individual I shown in Table 2 and this individual has to be mutated.

**Table 2.** An individual I that has been selected for mutation operation

| Codes | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 |
|---|---|---|---|---|---|---|---|---|
| I | 0 | 0 | **4** | 8 | 0 | 0 | 0 | 3 |
| Codes | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| I | **5** | 6 | **2** | 9 | **7** | 0 | **10** | 1 |

For the individual I all the state codes have to be examined and all the weights $w^I_1,...,w^I_M$ that exceed the value 1 have to be calculated. Some (or all for the best result) of these weights correspond to an optimal result. For all weights $w^I_m$ that have an optimal value the respective states $a_m$ have to be selected (see in Table 2 **bold** underlined numbers m of states $a_m$). For example, we have the following state transitions $a_1 \rightarrow$ {$K(a_2)$=1010, $K(a_3)$=0111, $K(a_4)$=0010}, $a_2 \rightarrow$ {$K(a_2)$=1**0**1**0**, $K(a_4)$=0**0**1**0**, $K(a_5)$=1**000**}$_{opt}$, $a_3 \rightarrow$ {$K(a_6)$=1001, $K(a_7)$=1100, $K(a_8)$=0011, $K(a_9)$=1011}, $a_6 \rightarrow$ {$K(a_5)$=**1**0**00**, $K(a_7)$=**1**1**00**}$_{opt}$, etc. Optimal solutions are indicated by subscript "opt" and the respective bits (i.e. bits with coincident indices that have equal values) of the codes are marked with bold font. The mutation operation permits a new child individual to be created and includes the following steps.

*Step 1*. All the elements that correspond to an optimal solution (see **bold** underlined numbers in Table 2) are included in the new individual (offspring).

*Step 2*. The codes for the remaining elements will be randomly regenerated in such a way that just free codes (i.e. such codes that have not been already chosen at step 1) can be selected.

Crossover is the most complicated operation of the considered EA. The main idea of this operation will also be illustrated by the same example of FSM. Suppose that at some step of EA we have found the codes for two individuals I1 and I2 shown in Table 3 and these individuals were chosen to be parents for creating a new individual that is a child. For all weights $w^{I1}_m$($w^{I2}_s$) that have an optimal value the respective states $a_m$ ($a_s$) have to be selected (see **bold** underlined numbers m of states $a_m$ for the first individual I1 and *italic* underlined numbers s of states $a_s$ for the second individual I2). The parents are chosen on the base of proportional selection [11].

**Table 3.** The results of encoding for two individuals

| Codes | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 |
|---|---|---|---|---|---|---|---|---|
| I1 | 0 | 0 | **4** | 8 | 0 | 0 | 0 | 3 |
| I2 | 8 | 0 | *5* | *4* | 10 | 1 | 0 | *2* |
| Codes | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| I1 | **5** | 6 | **2** | 9 | **7** | 0 | **10** | 1 |
| I2 | 7 | 0 | *9* | *3* | 0 | 0 | 0 | 6 |

The crossover operation permits a new child individual to be created and includes the following steps.

*Step 1*. The first solution (see Table 4) is formed from the selected elements of the first individual I1 (see **bold** underlined numbers in Table 3).

*Step 2*. Permitted selected elements from the second individual I2 (see *italic* underlined numbers in Table 3) are added to the first solution (i.e. to the child). An element is allowed for step 2 if:

a) It was not included into the child during the first step; and

b) It does not have a code that has already been used during the first step.

Table 5 shows the result of step 2 for our example.

*Step 3*. All the remaining permitted elements from the first and the second individuals are added to the child. An element is allowed for step 3 if it has not yet been included in the child and:

a) It has the same code for both individuals I1 and I2; or

b) It is included in the second individual I2 and the respective code of the first individual I1 was not used for the states; or

c) It is included in the first individual I1 and the respective code of the second individual I2 was not used for the states;

Table 6 shows the result of step 3 for our example.

**Table 4.** The result of step 1

| Codes | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 |
|---|---|---|---|---|---|---|---|---|
| Child | | | 4 | | | | | |
| Codes | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| Child | 5 | | 2 | | 7 | | 10 | |

**Table 5.** The result of step 2

| Codes | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 |
|---|---|---|---|---|---|---|---|---|
| Child | | | 4 | | | | | |
| Codes | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| Child | 5 | | 2 | 3 | 7 | | 10 | |

**Table 6.** The result of step 3

| Codes | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 |
|---|---|---|---|---|---|---|---|---|
| Child | 8 | | 4 | | | 1 | | |
| Codes | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| Child | 5 | 6 | 2 | 3 | 7 | | 10 | |

*Step 4*. All the remaining states that have not been assigned yet are recorded in free boxes for codes from left to right.

Table 7 presents the final result of the crossover operation.

Individuals I1, I2 and the child can be evaluated as follows: $W_{I1}=W_{I2}=11$, $W_{child}=13$ (i.e. the child is better than any of the parents I1 and I2) and the optimal weight W=15.

There are two termination conditions for the considered EA: obtaining an optimal solution or exceeding a specified time limit.

**Table 7.** The result of step 4 that gives the final result of the crossover operation

| Codes | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 |
|---|---|---|---|---|---|---|---|---|
| Child | 8 | 9 | 4 | | | 1 | | |
| Codes | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| Child | 5 | 6 | 2 | 3 | 7 | | 10 | |

## 3 Variations of the Evolutionary Algorithm

Note that for many practical applications it is allowed that a state has more than one code. If the FSM circuit is constructed from RAM blocks then using multiple codes does not make the circuit more complicated [2]. Moreover applying this technique enables us to improve the results of encoding. It should be noted that for some practical problems an optimal solution, that only permits each state to be assigned a single unique code, cannot be obtained. For example, such solution cannot be found for the following set of state transitions: $a_1 \Rightarrow \{a_1,a_2\}$, $a_2 \Rightarrow \{a_2,a_3\}$, $a_3 \Rightarrow \{a_4,a_5\}$, $a_4 \Rightarrow \{a_1,a_3\}$, $a_5 \Rightarrow \{a_1\}$. However, if more than one code is permitted for the states we can find an optimal solution, which is: $K(a_1)=000$, $K(a_2)=001$, $K(a_3)=100$ and 101, $K(a_4)=011$, $K(a_5)=111$. The EA can be modified slightly in order to produce the proper solution. Indeed if an optimal result cannot be found within a predefined time interval we can allow using more than one code for states. Thus, the algorithm is relatively flexible when it comes to future improvements and modifications.

## 4 Experimental Results

The results of the proposed EA were estimated for more than 100 digital circuits that required the considered above encoding technique within the respective process of synthesis. Fig. 1 shows these results for 25 FSMs. We considered block-based decomposition of FSMs [3], where $R=\text{int}\log_2 M$ and $R_{av}$ is an average number of outputs for the blocks. So the considered technique makes possible the number of outputs required for each block to be decreased on average by 1.8.

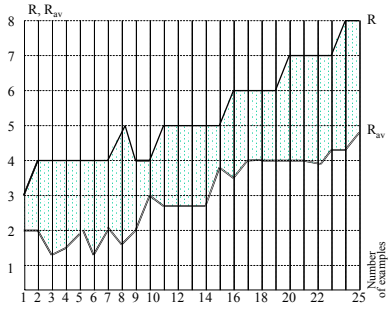



**Fig. 1.** The results of experiments.

The EA has been analyzed in several contexts. Firstly, we evaluated primary genetic operations that are reproduction (based on elitist rule and proportional selection criteria), mutation and crossover. The considered options A, B, C and D are listed below:

A: the crossover operation was carried out in order to form 90% of population for the next generation and 10% of population for the next generation was chosen with a probability based on fitness (i.e. based on proportional selection);

B: firstly the crossover operation was carried out in order to form 100% of population, secondly the mutation operation based on proportional selection was performed for 10% of individuals of the new generation and finally 10% of individuals in the next generation were replaced with 10% of randomly generated individuals;

C: the mutation operation based on proportional selection was carried out in order to form 100% of population for the next generation and 10% of individuals in the next generation were replaced with 10% of randomly generated individuals;

D: the crossover operation was carried out in order to form 90% of population for the next generation and 10% of population for the next generation was chosen based on elitist rule.

Fig. 2 shows how the execution time for all four options depends on the number of individuals in population. This dependency was considered for an FSM with 15 states (M=15) and with at maximum 4 transitions from each state. The experiments were performed on PentiumIII/800MHz/256MB. Fig. 3 shows how the number of required generations for all four options depends on the number of individuals in population.

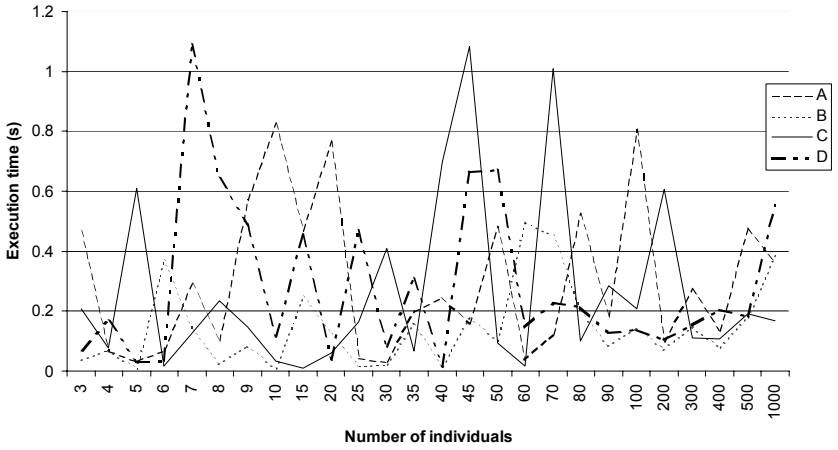



**Fig. 2**. Dependency of execution time on the number of individuals in population

Secondly, we examined practical applications that could benefit from the considered encoding technique. This enabled us to estimate some parameters, such as the expectable size of codes. Table 8 presents examples of control circuits used in assembly lines for manufacturing purposes. The number of individuals in population

was chosen to be 15. Here NG is the number of generations, W is the optimal weight, $W_e$ is the obtained weight, ET is the execution time, $G_{min}$ = max (int$\log_2|A(a_m)|$), m=1,...,M.

Thirdly, we performed a set of experiments for randomly generated examples with different initial data (such as the number of individuals in populations) and variable requirements (such as using one code for each state of FSM or employing more than one code for some states). Table 9 shows the best results for options A, B, C, D obtained for arbitrary selected examples (the option that gave the best result is indicated in the first column in parentheses). For the examples aex7 and aex8 just the option D was used and we received $W_e$<W because the value W cannot be obtained when using just one code for each state. If we allow to employ more than one code for some states then the result with $W_e$=W can be easily found (ET = 41.12 s for aex7 and ET = 9.38 s for aex8).

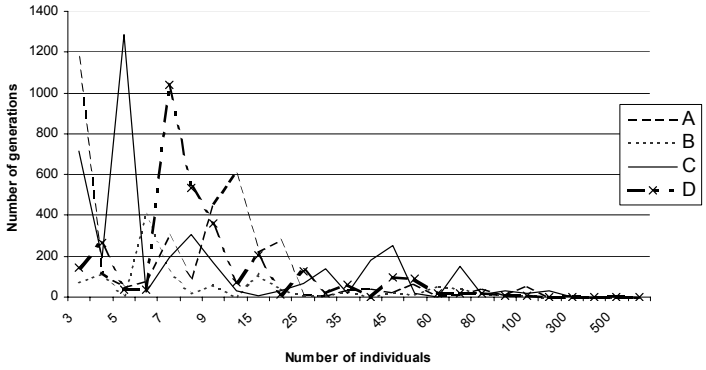



**Fig. 3.** Dependency of the number of generations on the number of individuals in population

**Table 8.** The results of experiments for practical examples

| Example | NG | W | $W_e$ | ET (s) | M | $G_{min}$ |
|---|---|---|---|---|---|---|
| Ex1 | 134 | 32 | 32 | 0.424 | 20 | 2 |
| Ex2 | 63 | 36 | 36 | 0.261 | 29 | 2 |
| Ex3 | 99 | 7 | 7 | 0.202 | 12 | 4 |
| Ex4 | 111 | 30 | 30 | 0.255 | 15 | 2 |

**Table 9.** Experiments with arbitrary selected FSMs

| Example | NG | W | $W_e$ | ET (s) | M | $G_{min}$ |
|---|---|---|---|---|---|---|
| aex1 (B) | 15 | 21 | 21 | 0.052 | 21 | 2 |
| aex2 (B,D) | 4 | 25 | 25 | 0.017 | 28 | 2 |
| aex3 (D) | 146 | 15 | 15 | 0.216 | 10 | 2 |
| aex4 (B) | 23682 | 34 | 34 | 61.767 | 15 | 3 |
| aex5 (B) | 490 | 31 | 31 | 3.897 | 47 | 2 |
| aex6 (B) | 12854 | 42 | 42 | 49.201 | 16 | 3 |
| aex7 | 5534 | 88 | 87 | 1000 | 52 | 2 |
| aex8 | 1024 | 38 | 37 | 1000 | 49 | 3 |

## 5 Conclusion

In the previous discussion we have presented the evolutionary algorithm for state encoding that allows Boolean functions to be decomposed in such a way that the dependency of sub-functions obtained as a result of the decomposition on the arguments can be reduced. The algorithm has been analyzed in several contexts. Firstly, we evaluated the primary genetic operations that are reproduction, mutation and crossover. Secondly, we examined practical applications that require the considered encoding technique. This enabled us to estimate some parameters, such as the expectable size of codes. Thirdly, we performed a set of experiments with different initial data (such as the number of individuals in populations) and variable requirements (such as using one code for each state of finite state machine or employing more than one code for some states). The examples in the paper and the results of experiments with a C++ program that implements the proposed evolutionary algorithm have shown that the considered approach is very effective.

## References

1. S. Baranov, *Logic Synthesis for Control Automata* (Kluwer Academic Publishers, 1994).

2. V. Sklyarov, Reconfigurable models of finite state machines and their implementation in FPGAs, Journal of Systems Architecture, 47, 2002, pp. 1043-1064.

3. V. Sklyarov, *Synthesis of Finite State Machines Based on Matrix LSI* (Minsk, Science and Technique, 1984).

4. J. Torresen, Possibilities and Limitations of Applying Evolvable Hardware to Real-World Applications, Proceedings of FPL, Villach, Austria, August, 2000, pp. 230-239.

5. A. Thompson, P. Layzell, and R.S. Zebulum, Exploration in Design Space: Unconventional Electronics Design Through Artificial Evolution, *IEEE Transactions on Evolutionary Computations*, vol. 3, No. 3, September, 1999, pp. 167-176.

6. C. Manovit, C. Aporntewan, and P. Chongstitvatana, Synthesis of Synchronous Sequential Logic Circuits from Partial Input/Output Sequences, Proceedings of ICES'98, Evolvable Systems: From Biology to Hardware, Springer, N 1478, 1998, pp. 98-105.

7. H. Hemmi, J. Mizoguchi, and K. Shimohara, Development and Evolution of Hardware Behaviors, Toward Evolvable Hardware, Springer, N 1062, 1996, pp. 250-265.

8. J. Mizoguchi, H. Hemmi, and K. Shimohara, Production genetic algorithms for automated hardware design through an evolutionary process, IEEE Conference on Evolutionary Computations, 1994, pp. 250-265.

9. J.R. Koza, F.H. Bennet III, D. Andre, and M.A. Keane, *Genetic Programming III* (Morgan Kaufmann Publishers, 1999).

10. J.F. Miller, P. Thomson, and T. Fogarty, Designing Electronic Circuits Using Evolutionary Algorithms. Arithmetic Circuits: A Case Study, Genetic Algorithms and Evolution Strategies in Engineering and Computer Science (John Wiley&Sons, 1998), pp. 105-131.

11. Z. Michalewicz and D.B. Fogel, *How to Solve It: Modern Heuristics* (Springer, 2000).

12. V. Sklyarov, Synthesis of Control Circuits with Dynamically Modifiable Behavior on the Basis of Statically Reconfigurable FPGAs, Proceeding of 13th Symposium on Integrated Circuits and Systems Design: SBCCI, Manaus, Brazil, 18-24 September 2000, pp. 353-358.

# Hypercube FrameWork for ACO applied to timetabling

Franklin Johnson[1], Broderick Crawford[1], and Wenceslao Palma[1]
1 Pontificia Universidad Católica de Valparaíso, Escuela de Ingenieria Informática, Valparaiso, Chile
franklin.johnson.p@mail.ucv.cl
{broderick.crawford,wenceslao.palma}@ucv.cl

**Abstract**. We present a resolution technique of the University course Timetabling problem (UCTP), this technique is based in the implementation of Hypercube framework using the Max-Min Ant System. We presented the structure of the problem and the design of resolution using this framework.

A simplification of the UCTP problem is used, involving three types of hard restrictions and three types of soft restrictions. We solve experimental instances and competition instances the results are presented of comparative form to other techniques. We presented an appropriate construction graph and pheromone matrix representation. A representative instance is solved in addition to the schedules of the school of Computer science engineering of the Catholic University of Valparaiso. The results obtained for this instance appear. Finally the conclusions are given.

## 1 Introduction

The Timetabling problems are faced periodically by each school, college and university in the world. In a basic problem, a set of events (particular classes, conferences, classes, etc) must be assigned to a set of hours of a way that all the students can attend all of their respective events. With the reservation of which restrictions of hard type which necessarily they must be satisfied and soft restrictions exist that deteriorate the quality of the generated schedule. Of course, the difficulty of any particular case of the UCTP [1] [2] depends on many factors and in addition the assignment of rooms perceivably makes the problem more difficult in general.

Many techniques have been used in the resolution of this problematic one, between these we can find evolutionary algorithms, simulated annealing, and tabu-search. Other technique has presented good results is the genetic algorithms [3]. But we looked for here specifically to represent the resolution through the ant colony optimization (ACO) and through the implementation of Hypercube framework for Max-Min Ant System (abbreviation in Spanish MTH-SHMM). We give a representation for the problem, generating an appropriate construction graph and the respective pheromone matrix associated.

In the following sections we present the UCTP problem, the problem design for Hypercube framework. The instances of the problem used and the results of the experimentation. Finally the conclusions of the work appear.

## 2 University Course Timetabling Problem (UCTP)

### 2.1 Problem description

The problem timetabling considered to make this study similar to one is presented initially by Paechter in [4]. Timetabling of university courses is a simplification of a typical problem [5]. It consists of a set of events $E$ and must to be scheduled in a set of timeslots $T = \{t_1, ..., t_k\}$ ($k = 45$, they correspond to 5 days of 9 hours each), a set of rooms $R$ in which the events will have effect, a set of students $S$ who attend the events, and a set of features $F$ required by the events and satisfied by the rooms. Each student attends a number of events and each room has a maximum capacity. A feasible timetable is one in which all the events have bee assigned a timeslot and a room so that the following hard constraints are satisfied:

- No student attends more than one event at the same time;
- The rooms must be sufficiently great for all students who attend a class and to satisfy all the features required by the event;
- Only one event is in each room at any timeslot.

In addition, All possible timetable generated is penalized for each occurrence according to the number of violations that exists of the soft constraint of problem. Some of these restrictions appear next:

- A student has a class in the last slot of the day;
- A student has more than two classes in a row;
- A student has exactly one class on a day.

Feasible solutions are always considered to be superior to infeasible solutions, independently of the numbers of soft constraint violations. In fact, in any comparison, all infeasible solutions are to be considered equally worthless. The objective is to minimize the number of soft constraint violations in a feasible solution.

## 3 Design of Hypercube Framework SHMM for Timetabling (MTH-SHMM)

### 3.1 Resolution Structure

Given restrictions presented in the previous section and the characteristics of problem, we can now consider the option to design an effective MTH-SHMM for the UCTP. We have to decide how to transform the assignment problem (to assign events to *timeslots*) into an optimal path problem which the ants can solve [12]. To

do this we must create an appropriate construction graph for the ants to follow. We must then decide on an appropriate pheromone matrix and heuristic information to influence the paths the ants will take through the graph.

We present the principal elements used to generate the UCTP solutions, presenting in a figure 1 these three elements.

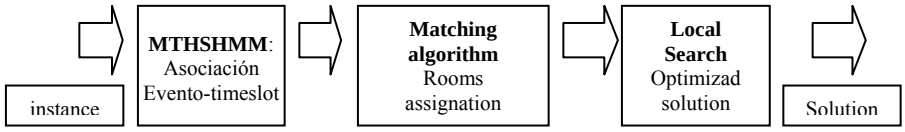



**Figure 1.** An instance of the problem is received like input, this it happens through an association process *event-timeslot*, assigns events to a *timeslot*, later a matching algorithm [8] is used for makes the assignation from rooms to each one of events associated to *timeslot*. In this point a solution is complete, but is low quality. Then a local search algorithm [16] is applied that improves the quality of the solution and gives like final result one optimal solution to the UCTP.

### 3.2 Construction graph

One of the main elements of the ACO metaheurístic is the power to model to the problem on construction graph [6] [7], that way a trajectory through the graph represents a problem solution. In this formulation of the UCTP it is required to assign each one of $|E|$ events to $|T|$ *timeslots*. Where direct representation of the construction graph this dice by $E \times T$; east dice graph we can then establish that the ants walk throughout a list of events, choosing *timeslot* for each event. The ants follow one list of events, and for each event *and,* the ants decide *timeslot t*. each event a this single time in *timeslot*, thus in each step an ant chooses any possible transition as it is in the figure 2.

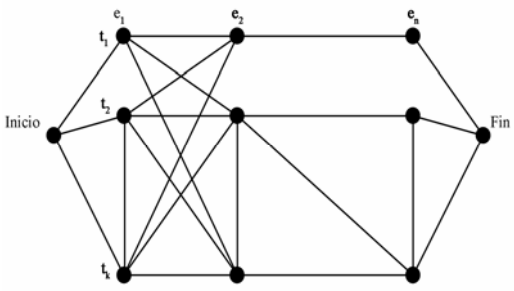


**Figure 2.** Each ant follows a list of events, and for each event $e \in E$, an ant chooses a *timeslot* $t \in T$.

The ants travel through the construction graph selecting ways of probabilistically way. Using the following function:

$$p_{(e,t_i)} = \frac{(\tau_{(e,t_i)})^{\alpha}}{\sum_{\theta \in T} (\tau_{(e,t_\theta)})^{\alpha}} \quad (1)$$

This probability function is come off the used in [6]. This function of probability directly depends on the pheromone information $\tau$, they have the possible ways to follow. The parameter α is like appendix of the original function in [6], for this case its value is 1.

### 3.3 The Pheromone matrix

In search of a pheromone matrix we represented that pheromones indicates the absolute position where the events must be placed. With this representation the pheromone matrix is given by $\tau(A_i) = \tau$, $i$=1,…,|$E$|, the pheromone does not depend on the partial assignments $A_i$. It can to observe that in this case the pheromone will be associated with nodes in the construction graph rather than edges between the nodes.

A disadvantage of this directs pheromone representation is that the absolute position of events in the *timeslots* it does not matter very much in producing a good timetable. The relative placement of events is more important. For example, given a perfect timetable, it is usually possible to permute many groups of *timeslots* without affecting the quality of the timetable.

By another side we defined that for the use of the heuristic information $\eta$ it must use a function that calculates a weighted sum of several or all of the soft and hard constrains in each assignation, which is to incur very high a computational cost stops this class of problem [8]. For this we will not use east type of information to orient the route of the ants.

### 3.4 Algorithm Description

We show the general structure of the algorithm, in which some modifications are made of presented in [9] [11]. A new assignation values to $\tau_{max}$, $\tau_{min}$ ,a new pheromone update rules. We define the assignment $A_i$ like the *timeslot* selected for the event *i.* The algorithm is the following:

```
1 Input: Problem instance I
2 τ_max ← 1
3 τ_(e, t) ← τ_max ∀ (e, t) Є E×T
4 calculate c(e, e') ∀ (e, e') Є E^2
5 calculate d(e)
6 sort E according « , resulting in e_1 « e_2 « …« e_n
7 while time limit not reached do
8   for a = 1 to m do
9       {construction process of ant a}
10      A_0 ← ∅
11      for i = 1 to |E| do
12      chooser timeslots t according to probabilities p_(e_i,t) for event e_i
13          A_i ← A_i-1 ∪ {(e_i, t)}
14      end for
15      C←matching_algorithm (A_n)
16      C_best_iteration ← best of C and C_best_iteration
17  end for
18  C_best_iteration ← applying local search to C_best_iteration
19  C_global_best ← best of C_best_iteration y C_global_besl
20 global pheromone update for τ using C_global_best , implicated to MTH
21 end while
22 Output: An optimized candidate solution C_global_best for I
```

Only the solution that causes the fewest number of hard constraint violations is selected for improvement by the Local Search. The pheromone matrix is updated only once by each iteration, and the global best solution is used for the update. Then $A_{global\_best}$ be the assignment of the best candidate solution $C_{global\_best}$ found since the beginning. The following update rule is used:

$$\tau(e,t) = \begin{cases} \rho \cdot \tau_{(e,t)} + (1-\rho) \cdot \Delta\tau^{upd}_{(e,t)} & \text{if } A_{major\_global}(e) = t, \\ \rho \cdot \tau_{(e,t)} & \text{otherwise} \end{cases} \quad (2)$$

In the right part of the equation is reduced to $\Delta\tau_{(e,t)}^{upd} = 1$if $A_{global\_best}(e) = t$ and 0 in otherwise. Where $A_{global\_best}$ is the solution used for pheromone update. With this update rule to makes sure that the pheromone values of the graph, are going to be always between values [0,1]. The rate of evaporation $\rho$ Є [0.1].

# 4 Experimentation

## 4.1 Tests

The algorithm was implemented in C++ programming language, under Linux system using GNU G++ compiler GCC 2.96. The behavior of Hypercube framework Max-Min Ant System (MTH-MMAS) was observed in the resolution of the UCTP. The used instances appear to continuation.

**Instances 1:** Instances of the UCTP are structured using a generator described in [10]. This generator allows generating classes of instances *small, medium*, which reflect varied problems of timetabling of several sizes.

**Instances 2:** In addition it was used a series of 20 instances created for International Timetabling Competition, these instances is made with the same generator used in instances 1.

The parameters study is made initially, to evaluate the best values than they must to assume these parameters. The small (small1) instances was used for using the MTH-MMAS without local search making evaluations with different ants numbers $m$ and with different evaporations factors $\rho$, the parameters of $\alpha = 1$, number on attempts = 10 and a maximum time by attempt = 90 seconds for all the tests. The results are in the following table.

**Table 1.** It presents the best results obtained when proving the instance small1.tim varying ants number $\boldsymbol{m}$ and evaporation factor $\boldsymbol{\rho}$.

| **Best solutions MTH-SHMM** | | | | | |
|---|---|---|---|---|---|
| $\boldsymbol{m}$ | **Evaluation** | **Tº seg.** | $\boldsymbol{\rho}$ | **Evaluation** | **Tº seg.** |
| 5 | 17 | 6,79 | 0,2 | 15 | 7,11 |
| 10 | 16 | 7,46 | 0,5 | 13 | 8,1 |
| 20 | 16 | 6,06 | 0,8 | 17 | 6,79 |

In the table to be observed the best results are obtained using the parameter $\boldsymbol{m}$=20 obtaining a evaluation of 16 in 6.06 seconds. And for the case of evaporation factor the best value is =0,5 in 8.1 seconds.

The values shown in the tables previously presented they belong to a series of executions that allow of experimental form to determine as are more advisable parameters to use in the execution of the algorithm of MTH-MMAS. This way we compared the algorithm of the Max-Min Ant System with and without Hypercube framework, in addition the local search is included to increment the quality of the solutions in different instances.

## 4.2 Distribution results

We show a graphic in which they are a series of boxplot which they represent relative distribution of the number of constrained violations for hypercube

framework Max-Min Ant System (MTH-SHMM) and the Max-Min Ant System (MMAS) pure for all the instances of type small and medium with which they were proven.

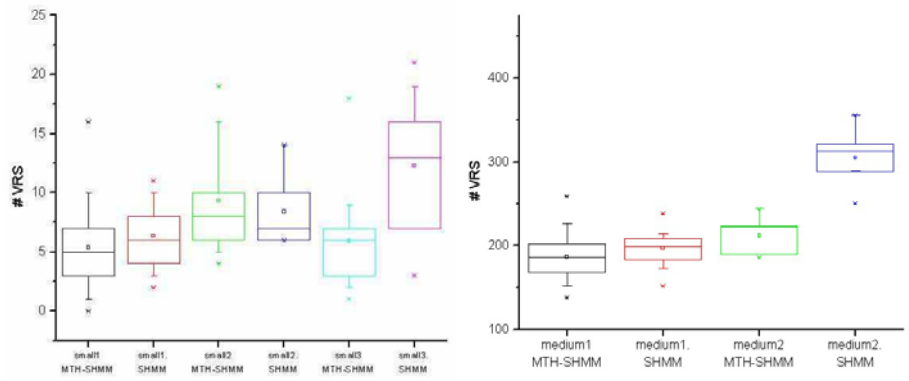



**Figure 3.** Like it is possible to be observed in most of the results for the different instance types, the obtained results using hypercube framework they are of better quality (smaller violation of soft constrain VRS) since 50% of the data represented by a horizontal line always are under the level of the same instance for the Max-Min Ant System (SHMM) pure.

### 4.3 Comparison with other techniques

Here it present a comparative picture between the solutions obtained for different instances for the UCTP doing use of different techniques like Simulated annealing, advanced search and simulated annealing with local search [13][14][16]. The results obtained for the competition instances appear.

**Table 2.** It present the best results obtained when proving the instances of the International Timetabling Competition compared with other techniques.

| Technique | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| **SA** | 45 | 25 | 65 | 115 | 102 | 13 | 44 | 29 | 17 | 61 |
| **AS** | 257 | 112 | 266 | 441 | 299 | 209 | 99 | 194 | 175 | 308 |
| **SA-LS** | 211 | 128 | 213 | 408 | 312 | 169 | 281 | 214 | 164 | 222 |
| **MTH-MMAS** | 270 | 193 | 294 | 586 | 406 | 221 | 305 | 244 | 201 | 358 |

| Technique | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|
| **SA** | 44 | 107 | 78 | 52 | 24 | 22 | 86 | 31 | 44 | 7 |
| **AS** | 273 | 242 | 364 | 156 | 95 | 171 | 148 | 117 | 414 | 113 |
| **SA-LS** | 196 | 282 | 315 | 345 | 185 | 185 | 409 | 153 | 281 | 106 |
| **MTH-MMAS** | 268 | 312 | 341 | 403 | 222 | 234 | 371 | 184 | 345 | 201 |

For these instances and compared with the other solutions the MTH-MMAS it present two characteristics a to evaluate; first it has the capacity to generate feasible solutions for these instances. These instances are of great you make difficult since they are for Timetabling competitions. Second the quality of the generated solutions is of very low category compared with the technique based on Simulated Annealing,

which has the best found historical results for these instances, but in comparison with the other instances do not present great difference. These evaluations are not feasible in order to decide if a technique is better than other, since the differences in variable results can be for different external variables.

To continuation it presents the comparison for the small and medium instances. We will compare the algorithm of MTH-MMAS and the MMAS pure with respect to Ant Colony System algorithm of Krzysztof Socha (ACS) and to algorithm based on random restart local search (RRLS).

**Tabla 3.** It present best results obtained when proving the test instances small and medium.

| Technique | Small1 | small2 | small3 | medium1 | medium2 |
|---|---|---|---|---|---|
| **RRLS** | 11 | 8 | 11 | 199 | 202 |
| **ACS** | 1 | 3 | 1 | 195 | 184 |
| **MMAS** | 3 | 6 | 3 | 152 | 250 |
| **MTH-MMAS** | 0 | 4 | 1 | 138 | 186 |

As it is possible to be observed for these instances in the MTH-MMAS present a superiority in the quality of the generated solutions (smaller VRS). Always by on the quality the solutions generated with the MMAS. We can to say that the hypercube framework it improves the quality of the ant algorithm applied.

## 5 In the Practices

### 5.1 UCV Instance

As a form to approach investigation of the project to a practice plane we implemented a resolution for the UCTP using the Hypercube framework Max-Min Ant System. This problematic is common and it is present in all type of institution of study. It is by that one has been implemented resolution to this problematic creating an instance of the problem for the Catholic University of Valparaiso and specifically for the school of Informatics Engineering.

A tool in C language was implemented, to which him the courses enter indicating the semester, assistants, if it has assistantship, times to the week that are dictated and his characteristics. In addition the rooms are entered to him, certain their capacity and characteristics. The system generates an instance introducing a factor of correlation between the events, generating therefore an instance with the same format that those of competition, small, medium. Stored this information in file ucv.tim. Ready pair to use by the MTH-MMAS algorithm

**Instance characteristic:** Total number of rooms and laboratories: 9. total number of event: 194, total Attending: 600. Number of characteristics: 5, maximum of events by student: 8, maximum of students by event: variable according to the event.

Before using the instance it was necessary to correct some parameters of MTH-MMAS algorithm implemented, since for the instance of UCV the number of

*timeslot* that they are used are 40 and not 45 like for other problems of the UCTP. in addition to an adaptation for the evaluation of soft constraint.

The instance was executed using a number of ants = 20, evaporation factor = 0.5. Time local search 100 seconds, total time by reboots = 900 seconds, number of reboots = 10. The best solution was obtained approximately to the 600 seconds with an evaluation of (# VRS) = 0. Which implies that the algorithm generated a complete timetable feasible and with the best possible quality.

Had to the quality of the solution it can be inferred that the generated instance previously that simulated the hour load of a semester of the school of computer science engineering had a low degree of correlation between courses of different semesters, thus a high performance in the resolution was obtained of the problem.

**Table 4.** Show the first 20 assigned events a its respective timeslot and rooms as a form to represent the solutions.

| **Event** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Timeslot** | 2 | 36 | 36 | 43 | 41 | 13 | 31 | 32 | 18 | 16 | 19 | 30 | 27 | 20 | 0 | 19 | 19 | 18 | 7 | 29 |
| **Room** | 0 | 0 | 1 | 3 | 0 | 4 | 3 | 4 | 2 | 1 | 3 | 3 | 2 | 3 | 4 | 0 | 4 | 4 | 4 | 3 |

In addition a file is had which has associate the classes with its respective ones events, since a class can have to correspond to several events in one week. This is a form to make a more visible and usable timetable.

## 6 Conclusion

A formal model was given to apply Hypercube framework to solve the University course timetabling problem (UCTP) making use of Max-Min Ant System, was generated an efficient model that solves instances of this problem creating good construction graph of and expressing a good pheromone matrix.

We presented the test result made for the Max-Min Ant System doing use of Hypercube framework. We was observed traverse of the given results that this propose framework is good means of resolution of combinatorial problems and for the case of the UCTP it presented good results for instances of small and medium type. Although the results were of low quality for the instances of the Competition. it emphasizes the fact that always it generates solutions feasible and for instances of normal difficulty of good evaluation. not obtain the best results for this problem, but if it improves in contrast with the Max-Min Ant System without work frame. It was managed to present a applied instance to the school of Computer science of the UCV, for which created a solution feasible thus it clarifies the fact to a technique useful in real applications.

## References


1. T. B. Cooper and J. H. Kingston. The complexity of timetable construction problems. In Proceedings of the 1st International Conference on Practice and Theory of Automated Timetabling (PATAT 1995). 1996.
2. H. M. M. ten Eikelder and R. J. Willemen. Some complexity aspects of secondary school timetabling problems. In Proceedings of the 3rd International Conference on Practice and Theory of Automated Timetabling (PATAT 2000), 2001.
3. E. K. Burke, J. P. Newall, and R. F. Weare. A memetic algorithm for university exam timetabling. In Proceedings of the 1st International Conference on Practice and Theory of Automated Timetabling (PATAT 1995), 1996.
4. B. Paechter. Course timetabling. Evonet Summer School, 2001.
5. B. Paechter, R. C. Rankin, A. Cumming, and T. C. Fogarty. Timetabling the classes of an entire university with an evolutionary algorithm. (1998).
6. M. Dorigo and G. Di Caro. The Ant Colony Optimization meta-heuristic. New Ideas in Optimization, 1999.
7. Christian Blum, Marco Dorigo, Andrea Roli, HC-ACO: The Hyper-Cube Framework for Ant Colony Optimization IEEE Transactions on Systems, Man and Cybernetics B, 34(2), 1161 - 1172, 2001.
8. Krzysztof Socha, Max-Min Ant System for International Timetabling Competition. (2003)
9. Krzysztof Socha, J. Knowles, y M. Sampels. Max-Min Ant System for the University Course Timetabling Problem.(2003)
10. http://www.dcs.napier.ac.uk/~benp
11. Thomas Stützle y H.H. Hoos, MAX-MIN Ant System. Future Genetic Computing System. 2000.
12. Christian Blum, Marco Dorigo, The Hyper-Cube Framework for Ant Colony Optimization IEEE Transactions on Systems, Man and Cybernetics B, 2004. HCF 2004.
13. http://www.or.ms.unimelb.edu.au/timetabling/ttframe.html?ttucp1.html
14. http://www.idsia.ch/Files/ttcomp2002/results.htm
15. http://www.idsia.ch/Files/ttcomp2002/IC_Problem/node3.html
16. Rossi-Doria, Blue, Knowles, Sampels, A local search for timetabling problem.(2002)

# Multitree-Multiobjective Multicast Routing for Traffic Engineering

Joel Prieto[1], Benjamín Barán[1,2], Jorge Crichigno[1]

[1] Catholic University of Asunción. Tte. Cantaluppi y Villalón. PO Box 1638, Asunción, Paraguay
jprieto@telesurf.com.py, jcrichigno@ece.unm.edu
[2] National Computer Centre, National University of Asunción. PO Box 1439. San Lorenzo, Paraguay
bbaran@cnc.una.py

**Abstract.** This paper presents a new traffic engineering multitree-multiobjective multicast routing algorithm (M-MMA) that solves for the first time the GMM model for Dynamic Multicast Groups. Multitree traffic engineering uses several trees to transmit a multicast demand from a source to a set of destinations in order to balance traffic load, improving network resource utilization. Experimental results obtained by simulations using eight real network topologies show that this new approach gets trade off solutions while simultaneously considering five objective functions. As expected, when M-MMA is compared to an equivalent singletree alternative, it accommodates more traffic demand in a high traffic saturated network.

## 1 Introduction

Multicast consists of concurrently data transmission from a source to a subset of all possible destinations in a computer network [1]. In recent years, multicast routing algorithms have become more important due to the increased use of new point to multipoint applications, like radio and TV, on-demand video and e-learning. Such applications generally have some quality-of-service (QoS) requirements 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 resource utilization [2]. These objectives cannot be met by traditional *Best Effort* Internet routing approaches.

In order to solve this problem, Traffic Engineering proposes the optimization of network resources using load-balancing techniques. The main idea behind a load balancing technique for multicast transmission is to partition a data flow into several

sub flows –or trees– between a source and all destination nodes. This objective is usually accomplished by minimizing the utilization ($\alpha$) of the most heavily used network resource, as a link (what is known as *maximum link utilization*). Load balancing technique not only reduces hot spots over the network, but also provides the possibility of supporting connections of high bandwidth requirements through several links of low capacity.

Multicast Traffic Engineering problems (MTE) simultaneously consider several objectives to be optimized; therefore, it has been recognized as a Multiobjective Optimization Problem (MOP) [3]. A lot of multiobjective algorithms for multicast routing were proposed in the literature [3-6, 8-13, 15-18]. They are generalized in the *GMM model for Dynamic Multicast Groups* [11, 18]. GMM model considers a multitree multicast load-balancing problem with splitting in a multiobjective context.

This work presents a multitree routing algorithm that solves for the first time the dynamic problem of multicast routing considering not only static routing, but also dynamic routing, where multicast groups arrive one after another into a network.

The remainder of the document is organized as follows: Section 2 presents the mathematical formulation of the problem. A brief introduction to multiobjective optimization problems appears in Section 3. A complete explanation of the proposed algorithm is presented in Section 4. Testing scenarios are shown in Section 5. The experimental results are discussed in Section 6, while the final conclusions and future works are left for Section 7.

## 2 Problem Formulation

A network is modelled 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 $z_{ij}$, $d_{ij}$ and $t_{ij} \in \Re^+$ be its capacity, delay and current traffic respectively. Let $s \in V$ denotes the source node, $N \subseteq V - \{s\}$ denote the set of destination nodes, and $\phi \in \Re^+$ the traffic demand (in kbps) of a multicast request, which is treated as a flow $f$. Let consider that $f$ can be split into a number of sub flows $f_k$ ($k$=1,2,..,$|K|$), where $|K|$ denotes the cardinality of set $K$. For each $f_k$, a multicast tree $T_k(s,N)$ must be constructed to transport a traffic $\phi_k$ , which is part of the total flow demand $\phi$, as shown in (9).

Let $p_{Tk}(s, n) \subseteq T_k(s, N)$ denote the path that connects the source node $s$ with a destination node $n \in N$ using tree $T_k$. Finally, let $d(p_{Tk}(s, n))$ and $h(p_{Tk}(s, n))$ represent the delay and the hop count of $p_{Tk}(s, n)$, i.e.,

$$d\left(p_{Tk}(s,n)\right) = \sum_{(i,j)\in p_{Tk}(s,n)} d_{ij} \qquad (1) \qquad h\left(p_{Tk}(s,n)\right) = \sum_{(i,j)\in p_{Tk}(s,n)} \mathbf{1} \qquad (2)$$

Using the above definitions, the multicast routing problem for traffic engineering treated in this paper is formulated as a MOP that tries to find a set of $|K|$ multicast trees $T_k(s,N)$ that minimizes the following five objective functions:

a- Maximal link utilization:

$$\alpha = \underset{\substack{(i,j)\in T_k \\ k\in K}}{Max} \left\{ (t_{ij} + \sum_{k=1}^{|K|} \phi_k)/z_{ij} \right\} \qquad (3)$$

b- Average delay:

$$D_A = \frac{1}{|N||K|} \sum_{k\in K} \sum_{n\in N} d\left(p_{Tk}(s,n)\right) \qquad (4)$$

c- Maximal delay:

$$D_M = \underset{\substack{n\in N \\ k\in K}}{Max} \{d(p_{Tk}(s,n))\} \quad (5)$$

d- Hop count average:

$$H_A = \frac{1}{|N||K|}\sum_{k\in K}\sum_{n\in N} h(p_{Tk}(s,n)) \quad (6)$$

e- Total bandwidth consumption:

$$BW = \sum_{k\in K} \phi_k \cdot |T_k| \quad (7)$$

subject to:

f- Link capacity constraint:

$$t_{ij} + \sum_{k\in K}\sum_{(i,j)\in Tk} \phi_k \le z_{ij} \quad (8)$$

g- Total information constraint:

$$\sum_{k=1}^{|K|} \phi_k = \phi \quad (9)$$

It should be mentioned that not all $|K|$ sub flows are necessary used. Therefore, if any $\phi_k = 0$ ($k = 1,2,..,|K|$), Eq. (4), (5) and (6) do not consider the corresponding $p_{Tk}(s, n)$ for calculation given that the tree is not used to transmit any information. Of course, the value of $|K|$ should be properly adjusted.

## 3 Multiobjective Optimization Problems

A general Multiobjective Optimization Problem (MOP) includes a set of $l$ decision variables, $r$ objective functions, and $c$ restrictions. Objective functions and restrictions are functions of decision variables. This can be expressed as:

*Optimize* $\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x}) = (g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \ldots, g_r(\boldsymbol{x}))$.
*Subject to* $\boldsymbol{e}(\boldsymbol{x}) = (e_1(\boldsymbol{x}), e_2(\boldsymbol{x}), \ldots, e_c(\boldsymbol{x})) \ge \boldsymbol{0}$,

Where $\boldsymbol{x} = (x_1, x_2, \ldots, x_l) \in \boldsymbol{X}$ is the decision vector, and
$\boldsymbol{y} = (y_1, y_2, \ldots, y_r) \in \boldsymbol{Y}$ 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 $\boldsymbol{e}(\boldsymbol{x}) \ge \boldsymbol{0}$ determines the set of feasible solutions $\boldsymbol{X_f}$ and its corresponding set of objective vectors $\boldsymbol{Y_f}$. A multiobjective problem consists in finding $\boldsymbol{x}$ that optimizes $\boldsymbol{g}(\boldsymbol{x})$. 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 $\boldsymbol{p}, \boldsymbol{q} \in \boldsymbol{X_f}$:

$\boldsymbol{g}(\boldsymbol{p}) = \boldsymbol{g}(\boldsymbol{q})$ iff $\forall i \in \{1, 2, \ldots, r\}: g_i(\boldsymbol{p}) = g_i(\boldsymbol{q})$
$\boldsymbol{g}(\boldsymbol{p}) \le \boldsymbol{g}(\boldsymbol{q})$ iff $\forall i \in \{1,2,\ldots, r\}: g_i(\boldsymbol{p}) \le g_i(\boldsymbol{q})$
$\boldsymbol{g}(\boldsymbol{p}) < \boldsymbol{g}(\boldsymbol{q})$ iff $\boldsymbol{g}(\boldsymbol{p}) \le \boldsymbol{g}(\boldsymbol{q})$ and $\boldsymbol{g}(\boldsymbol{p}) \ne \boldsymbol{g}(\boldsymbol{q})$

Then, in a minimization context, two solutions $\boldsymbol{p}, \boldsymbol{q} \in \boldsymbol{X_f}$ satisfy one and only one of the following three conditions:

$\boldsymbol{p} \succ \boldsymbol{q}$ ($\boldsymbol{p}$ dominates $\boldsymbol{q}$), iff $\boldsymbol{g}(\boldsymbol{p}) < \boldsymbol{g}(\boldsymbol{q})$
$\boldsymbol{q} \succ \boldsymbol{p}$ ($\boldsymbol{q}$ dominates $\boldsymbol{p}$), iff $\boldsymbol{g}(\boldsymbol{q}) < \boldsymbol{g}(\boldsymbol{p})$
$\boldsymbol{p} \sim \boldsymbol{q}$ ($\boldsymbol{p}$ and $\boldsymbol{q}$ are non-comparable), iff $\boldsymbol{p} \nsucc \boldsymbol{q}$ and $\boldsymbol{q} \nsucc \boldsymbol{p}$.

A decision vector $\boldsymbol{x} \in \boldsymbol{X}_f$ is non-dominated with respect to a set $\boldsymbol{Q} \subseteq \boldsymbol{X}_f$ iff: $\boldsymbol{x} \succ \boldsymbol{q}$ or $\boldsymbol{x} \sim \boldsymbol{q}$, $\forall \boldsymbol{q} \in \boldsymbol{Q}$. 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* $\boldsymbol{X}_{\boldsymbol{true}}$ may be formally defined as: $\boldsymbol{X}_{\boldsymbol{true}} = \{\boldsymbol{x} \in \boldsymbol{X}_f \mid \boldsymbol{x}$ 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.*

## 4 Proposed Algorithm

Inspired in the SPEA scheme [14] the proposed M-MMA algorithm holds an evolutionary population $P$ and an external Pareto solution set $P_{nd}$. The algorithm begins with a set of random configurations called initial population. Each individual in the population represents a potential solution to the problem.

At each generation, the individuals are evaluated using an adaptability function, also known as *fitness*, proposed by SPEA, which is based on the dominance criterion presented in section 3. Based on this value, some individuals called parents are selected. The probability of selection of an individual is related to its *fitness*. Then, genetic probabilistic operators are applied to the parent to construct new individuals that will be part of a new population. The process continues until a stop criterion (as a maximum number of generations) is satisfied. M-MMA is summarized in Fig. 1.

- Read multicast group and traffic demand
- Initialize sets $P$ and $P_{nd}$

Do {

- Discard identical individuals of $P$
- Calculate objective functions of each individual in $P$
- Apply local optimization algorithm
- Update non-dominated set $P_{nd}$
- Compute fitness
- Selection
- Construct new solutions using *crossover*

} while stop criterion is not satisfied

**Fig. 1.** M-MMA algorithm

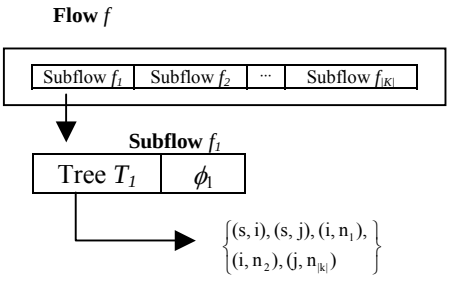



**Fig. 2.** Chromosome representation

### 4.1. Encoding

Each chromosome or individual is a candidate solution for the problem. Inspired in the GMM-model [11], an individual is represented by a set of trees transporting a flow $f$ (Fig. 2). Each flow is split in $|K|$ sub flows, as shown in (9), with a tree $T_k$ transmitting sub flow $f_k$. A tree is represented by the set of links belonging to it [6]. The field $\phi_k$ associated to each sub flow is the total information transmitted through $T_k$. This encoding scheme was selected motivated by the promising results obtained by Crichigno *et al.* [6], who conclude that better solutions are found when the trees are represented as a set of links instead of different paths.

### 4.2. Initial population

The procedure proposed in M-MMA to generate each initial solution of $P$ is shown in Fig. 3. The initialization procedure, called PrimRST (Prim Random Steiner

Tree), was proposed in [6]. Starting with a source node *s*, at each iteration, the algorithm expands the tree $T_k$ by choosing a new link from a set *A*, which contains all possible new links for the tree. A set $V_c$ contains the nodes already in the tree. The procedure continues until all destination nodes *N* are included in $V_c$. The value of $\phi_k$ is initialized as $\phi / |K|$. The value of $|K|$ should be previously decided by the traffic engineer. For the experimental results that follows, $|K| = 2$ was chosen. We have considered this small value because the problem is very complex. Moreover, in GMM model [11] the quantity of sub flows is considered as an objective function, because this algorithm is thought for MPLS networks [2], where the quantity of labels is limited. The PrimRST algorithm is iteratively used to construct each tree $T_k$ of the $|K|$ trees that constitute a chromosome, as shown in Fig. 2.

```
PrimRST(G(V,E), s, N)
- T_k = {};
- V_c = {s};
- A = {(s, j) | (s, j) ∈ E, j ∈ V};
do{
    - Choose a link (i, j) ∈ A at random.
    - A = A – {(i, j)}.
    If j ∉ V_c Then
        - T_k = T_k ∪ {(i, j)}.
        - V_c = V_c ∪ {j}.
        - A = A ∪ {(j, w) | (j, w) ∈ E , w ∉ V_c};
    End if
} while (N ∪ {s} ⊄ V_c)
- Prune useless links of T_k
- Return T_k
```

**Fig. 3.** Procedure PrimRST used to build random multicast Trees

```
Local Optimization (P, Δ_0, ε )
For i=1 until |P|
  Δ=Δ_0
  While Δ > ε
      If φ_1^i + Δ·φ^i ≤ φ^i then
          φ_1^* = φ_1^i + Δ·φ^i
          φ_2^* = φ^i - φ_1^i
          Evaluate individual f*
          If f* ≻ f^i then
              φ_1^i = φ_1^*
              φ_2^i = φ_2^*
          else
              Δ = Δ / 2
          End if
      else
          Δ = Δ / 2
      End if
  End while
End for
```

**Fig. 4.** Local Optimization Procedure

### 4.3. Local optimization

This procedure tries to optimize the amount of information $\phi_k$ to be transmitted through each sub flow, satisfying (8) and (9). In order to differentiate between two individuals of *P*, let $f^i$ be the *i*-th flow or individual of *P* (*i*=0,1,…,|*P*|), $\phi^i$ the total flow demand for that individual, and $\phi_k^i$ the *k*-information amount transmitted through sub flow $f_k^i$. Local Optimization procedure is presented in Figure 4. The process modifies the values of $\phi_k^i$ in the following way:

a) $\phi_1^i$ is increased and $\phi_2^i$ is decreased in a percentage Δ of $\phi^i$. In fact, $\phi_2^i$ is calculated as ($\phi^i$ - $\phi_1^i$). Initial value for Δ (known as $\Delta_0$) and its minimum value ε are parameters of the procedure.
b) If total information constraint (9) is fulfilled, new temporal values $\phi_1^*$ and $\phi_2^*$ are calculated and objective vector $f^*$ is evaluated; otherwise, Δ is reduced to Δ/2 and the process goes back to step a).
c) If the new solution $f^*$ dominates $f^i$, new values $\phi_1^*$ and $\phi_2^*$ are accepted as current best value and the process continues; otherwise, Δ is reduced to Δ/2 and

the procedure goes back to step a).

d) Iteration continues while $\Delta > \varepsilon$.

Once the iteration is completed, a new iteration begins, but instead of incrementing $\phi_1^i$, it is decreased.

### 4.4. Crossover

The crossover algorithm is based on the one originally presented by Zhengying *et al.* [16]. It was also used in several other publications [6, 7, 15]. The algorithm has four stages:

1. choose one tree from each parent;
2. identify common links of the selected pair. These links will be part of the child tree that will be in the next generation of *P*. Given that common links of the parents could lead to a child composed of disjoined sub-trees, new links may be added [16];
3. connect the disjoined sub-trees until a multicast tree is constructed. At this step, the sub-trees are connected at random. Each sub-tree has a root node. At each iteration, an interconnection algorithm adds a new link, which has a source-node already in a sub-tree. Two sub-trees are connected when the root of one sub-tree ($T_1$) is the destination node of the selected link, and the source node of the link belongs to the other sub-tree ($T_2$); the root of the new sub-tree is the root of $T_2$;
4. calculate $\phi_j = \left(\phi_j^p + \phi_j^q\right)/2$, for both sub flows *j*=1, 2, where $\phi_j^p$ and $\phi_j^q$ are the *j*-information amount ($\phi_j$) from the two parent trees *p* and *q*.

In order to fulfil the flow constraint given by (9), a normalized process computing $\phi_k$ is used. For a new individual, the new $\phi_k$ is given by the following equation:

$$\phi_k^{new} = \phi\left(\phi_k \Big/ \sum_{k=1}^{|K|} \phi_k\right) \quad (\mathbf{10})$$

## 5 Testing Scenario

Eight network topologies were used for testing purpose. They were: NTT (Nippon Telephone and Telegraph Co., Japan) [5], NSF (National Science Foundation, United States-US) [5], Telstra (Australia) [19], Sprintlink (US) [19], Ebone (Europe) [19], Tiscali (Europe) [19], Exodus (US) [19] and Abovenet (US) [19].

In order to compare M-MMA behaviour under several traffic loads over the network, three scenarios were defined for every topology: (a) low load, (b) high load and (c) saturation. For every scenario, Ψ traffic requests were generated, simulating a dynamic situation in which they arrive one after another. Each traffic request was created using a *groupGenerator* algorithm [7], summarized in Fig. 5.

The *groupGenerator* algorithm generates a multicast group with a destination size between $|N|_{min}$ and $|N|_{max}$; then, *random*(unif, 0, 2000) gives the arrival time of the group, with a uniform distribution between 0 and 2000 seconds. The duration of each group was exponentially distributed, with an average of 60 seconds. Finally,

```
groupGenerator
group(i)  =  groupGenerator(|N|min, |N|max);
Tbeg(i)   =  random(unif, 0,2000);
Tend(i)   =  Tbeg(i) + random(exp, 0,2000);
φ (i)     =  random(unif, φ min, φ max);
End groupGenerator
```

**Fig. 5.** GroupGenerator algorithm

the traffic demand is set to a value between $\phi_{min}$ and $\phi_{max}$. The parameters used to generate each scenario are given in Table 1.

Talavera *et al.* [7] showed that most MOEAs may suit for the task of routing multicast demand, but the main factor to define performance in a dynamical environment is the policy used to choose a specific solution from a Pareto front. They proposed different policies to perform this task, proving that the policy of choosing the closest solution to the origin provides excellent trade-off values, outperforming the traditional policy of choosing the solution with better $\alpha$. Consequently, we use that approach to select a solution from a Pareto front in our experiments. It is useful to mention that [7] concluded that average number of rejected groups might be considered an important metric to compare different algorithms and policies.

**Table 1.** Parameters used to generate testing scenarios

| Network topology | | | Scenarios load | Parameters | | | | |
|---|---|---|---|---|---|---|---|---|
| Name (Location) | Nodes | Links | | $\Psi$ | $\lvert N\rvert_{\min}$ | $\lvert N\rvert_{\max}$ | $\phi_{\min}$ | $\phi_{\max}$ |
| Telstra (Australia) | 57 | 118 | Low | 200 | 4 | 10 | 25 | 50 |
| | | | High | 300 | 10 | 25 | 50 | 200 |
| | | | Saturation | 400 | 10 | 35 | 75 | 300 |
| Sprintlink (US) | 44 | 166 | Low | 200 | 3 | 6 | 25 | 50 |
| | | | High | 300 | 9 | 12 | 50 | 200 |
| | | | Saturation | 400 | 9 | 20 | 75 | 300 |
| Ebone (Europe) | 23 | 76 | Low | 200 | 3 | 6 | 25 | 50 |
| | | | High | 300 | 5 | 10 | 50 | 200 |
| | | | Saturation | 400 | 8 | 15 | 75 | 300 |
| Tiscali (Europe) | 49 | 172 | Low | 200 | 4 | 6 | 25 | 50 |
| | | | High | 300 | 9 | 12 | 50 | 200 |
| | | | Saturation | 400 | 10 | 20 | 75 | 300 |
| Exodus (US) | 22 | 74 | Low | 200 | 3 | 6 | 25 | 50 |
| | | | High | 300 | 5 | 10 | 50 | 200 |
| | | | Saturation | 400 | 8 | 15 | 75 | 300 |
| Abovenet (US) | 33 | 84 | Low | 200 | 3 | 6 | 25 | 50 |
| | | | High | 300 | 5 | 10 | 50 | 200 |
| | | | Saturation | 400 | 8 | 15 | 75 | 300 |
| NTT (Japan) | 55 | 144 | Low | 200 | 4 | 10 | 100 | 200 |
| | | | High | 300 | 10 | 25 | 200 | 800 |
| | | | Saturation | 400 | 10 | 35 | 200 | 800 |
| NSF (US) | 14 | 42 | Low | 200 | 2 | 5 | 25 | 50 |
| | | | High | 300 | 3 | 7 | 50 | 200 |
| | | | Saturation | 400 | 6 | 9 | 75 | 300 |

For this problem, M-MMA was compared against MMA2 algorithm [6]. MMA2 is a multiobjective multicast algorithm that routes a request demand through only

one tree. We have chosen this algorithm because of its promising results when compared to other alternatives as MMA1 [4, 5] and SK [17]. The following dominance metrics were taken into account:

$D_{MMA2}$: Percentage of solutions selected using MMA2 that dominates the corresponding M-MMA solutions.

$D_{M\text{-}MMA}$: Percentage of solutions selected using M-MMA that dominates the corresponding MMA2 solutions.

$I$: Percentage of indifference relationships. This occurs when solutions found by MMA2 and M-MMA are non-comparables.

$Eq$: Percentage of solutions found by both algorithms that have equal values for objective functions.

We also have compared the amount of solutions selected by M-MMA that uses only one tree to transmit the traffic demand. Finally, percentages of rejected groups for lack of link capacity are given for each scenario.

## 6 Experimental results

Results for the simulations performed on eight network topologies are shown in tables 2, 3 and 4.

Table 2 summarizes the amount of solutions for each scenario according to the dominance metrics defined in section 5. There is not a clear dominant algorithm, given that many solutions are indifferent (in a multiobjective context) or they have identical values for the objective vectors. Shadowed cells in table 2 highlight this fact. This result is not a surprise, given that we are considering several conflicting objective functions.

**Table 2.** Classification of solutions according to dominance metrics

| Network | Scenario | $D_{MMA2}$ | $D_{M\text{-}MMA}$ | $I$ | $Eq$ | Network | Scenario | $D_{MMA2}$ | $D_{M\text{-}MMA}$ | $I$ | $Eq$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Telstra | Low | 32.50 | 5.50 | 50.50 | 11.50 | Exodus | Low | 8.50 | 2.50 | 4.00 | 85.00 |
| | High | 11.33 | 17.67 | 57.33 | 13.67 | | High | 9.33 | 7.67 | 1.00 | 82.00 |
| | Saturation | 26.00 | 8.75 | 48.00 | 17.25 | | Saturation | 6.50 | 12.00 | 10.25 | 71.25 |
| Sprintlink | Low | 3.00 | 11.50 | 2.50 | 83.00 | Abovenet | Low | 11.50 | 4.50 | 5.50 | 78.50 |
| | High | 3.67 | 16.33 | 0.67 | 79.33 | | High | 2.67 | 10.33 | 1.00 | 86.00 |
| | Saturation | 21.50 | 11.75 | 14.25 | 52.50 | | Saturation | 11.75 | 12.50 | 22.50 | 53.25 |
| Ebone | Low | 7.50 | 13.00 | 3.00 | 76.50 | NTT | Low | 0.50 | 34.50 | 0.50 | 64.50 |
| | High | 21.00 | 12.67 | 3.33 | 63.00 | | High | 2.33 | 27.67 | 0.33 | 69.67 |
| | Saturation | 7.50 | 10.75 | 29.25 | 52.50 | | Saturation | 5.50 | 22.50 | 2.75 | 69.25 |
| Tiscali | Low | 7.00 | 13.50 | 3.00 | 76.50 | NSF | Low | 4.50 | 9.50 | 6.50 | 79.50 |
| | High | 2.33 | 0.00 | 3.67 | 94.00 | | High | 0.67 | 10.33 | 0.00 | 89.00 |
| | Saturation | 9.25 | 0.75 | 28.50 | 61.50 | | Saturation | 8.00 | 15.50 | 2.75 | 73.75 |

The percentage of multicast groups routed by a single tree is given in Table 3. We should clarify that M-MMA solutions not always use multitree, given that one tree may transport the whole information $\phi$. In many cases, both algorithms found the same unitree solution. Multitree solution is used only when it is clearly better than unitree. This is the main reason why M-MMA could find better global solutions.

Actually, a mean of 63.2% of the best solutions had only one tree, and M-MMA is able to find those solutions, just as MMA2. However, in several opportunities the best solution for a given situation is multitree and therefore, only M-MMA is able to find it, making clear why M-MMA outperforms MMA2.

Finally, table 4 gives an idea about multitree performance considering the percentage of rejected groups for lack of link capacity. This result illustrates that M-MMA solutions fulfil the Traffic Engineering purpose, using load-balancing techniques in order to optimize network resources, and therefore, accommodating more traffic than a purely unitree approach like MMA2.

**Table 3.** Percentage of multicast groups routed by a single tree

| Network | Scenario | % |
|---|---|---|
| Telstra | Low | 93.50 |
| | High | 91.67 |
| | Saturation | 58.25 |
| Sprintlink | Low | 53.00 |
| | High | 85.00 |
| | Saturation | 83.75 |
| Ebone | Low | 46.00 |
| | High | 68.33 |
| | Saturation | 57.25 |
| Tiscali | Low | 82.50 |
| | High | 82.00 |
| | Saturation | 83.00 |
| Exodus | Low | 41.50 |
| | High | 53.00 |
| | Saturation | 60.25 |
| Abovenet | Low | 50.50 |
| | High | 77.33 |
| | Saturation | 74.50 |
| NTT | Low | 25.00 |
| | High | 28.33 |
| | Saturation | 46.50 |
| NSF | Low | 62.00 |
| | High | 50.00 |
| | Saturation | 42.00 |

**Table 4.** Percentage of groups rejected for lack of link capacity for both algorithms

| Network | Scenario | % Rejected by | |
|---|---|---|---|
| | | MMA2 | M-MMA |
| Telstra | Low | 0.00 | 0.00 |
| | High | 5.67 | 5.67 |
| | Saturation | 37.75 | 35.75 |
| Sprintlink | Low | 0.00 | 0.00 |
| | High | 0.33 | 0.00 |
| | Saturation | 14.00 | 9.00 |
| Ebone | Low | 0.00 | 0.00 |
| | High | 2.00 | 2.00 |
| | Saturation | 27.00 | 27.00 |
| Tiscali | Low | 0.00 | 0.00 |
| | High | 3.00 | 0.00 |
| | Saturation | 28.50 | 7.75 |
| Exodus | Low | 0.00 | 0.00 |
| | High | 0.00 | 0.00 |
| | Saturation | 10.25 | 10.00 |
| Abovenet | Low | 0.00 | 0.00 |
| | High | 0.00 | 0.00 |
| | Saturation | 22.00 | 19.50 |
| NTT | Low | 0.00 | 0.00 |
| | High | 0.33 | 0.00 |
| | Saturation | 2.50 | 1.50 |
| NSF | Low | 0.00 | 0.00 |
| | High | 0.00 | 0.00 |
| | Saturation | 1.75 | 1.25 |

## 7 Conclusion and future work

This paper presents the M-MMA algorithm, which is able to solve for the first time the GMM-model in a dynamical environment, considering multitree. The proposed algorithm treats the multiobjective problem of multicast routing in a network, splitting traffic demand into several trees (multitree context) to optimize network resource utilization. To better accomplish the optimization goal, M-MMA proposes a local optimization procedure that finds better solutions improving the relative amount of information to be transmitted through each tree.

Results obtained by simulations on dynamical environments where traffic demands come one after another show that no studied algorithm is clearly dominant. In fact, many times the best solution under the given policy had only one tree; however, the best solution for a given situation is sometimes a multitree and therefore, only

M-MMA is able to find it. As a consequence, M-MMA is able to accommodate more traffic demand under a saturated scenario. For further study, we plan to consider simultaneous routing of several multicast requests in optical networks.

## References

1. A. Tanenbaum: *Computer Networks*, Prentice Hall, 2003.
2. D. Awdeuche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus: *Requirements For Traffic Engineering Over MPLS*. RFC 2702. 1999.
3. J. Crichigno, and B. Barán: *Multiobjective Multicast Routing Algorithm*. IEEE ICT'2004, Ceará, Brazil, 2004.
4. B. Barán, and J. Crichigno: *A Multicast Routing Algorithm Using Multiobjective Optimization*. IEEE ICT'2004, Ceará, Brazil, 2004.
5. J. Crichigno, and B Barán: *Multiobjective multicast routing algorithm for traffic engineering*. IEEE ICCCN 2004, Chicago USA.
6. J. Crichigno, F. Talavera, J. Prieto, and B. Barán: *Enrutamiento Multicast utilizando Optimización Multiobjetivo*. CACIC'2004, Buenos Aires, Argentina, 2004. pp. 147-158.
7. F. Talavera, J. Crichigno, and B. Barán: *Policies for Dynamical MultiObjective Environment of Multicast Traffic Engineering*. IEEE ICT 2005, South Africa.
8. Y. Donoso, R. Fabregat, and J. Marzo: *Multi-Objective Optimization Algorithm for Multicast Routing with Traffic Engineering*. IEEE ICN 2004.
9. R. Fabregat, Y. Donoso, J.L. Marzo, and A. Ariza: *A Multi-Objective Multipath Routing Algorithm for Multicast Flows*. SPECTS 2004.
10. R. Fabregat, Y. Donoso, F. Solano, and J.L. Marzo: *Multitree Routing for Multicast Flows: A Genetic Algorithm Approach*. CCIA 2004.
11. Y. Donoso, R. Fabregat, F. Solano, J. L. Marzo, and B. Barán: *Generalized Multiobjective Multitree model for Dynamic Multicast Groups*. IEEE ICC 2005, Seul Corea.
12. A. Roy, N. Banerjee, and S. Das: *An efficient Multi-Objective QoS-Routing Algorithm for Wireless Multicasting*. INFOCOM 2002.
13. X. Cui, C. Lin, and Y. Wei: *A Multiobjective Model for QoS Multicast Routing Based on Genetic Algorithm*. ICCNMC 2003.
14. 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, pp. 257-271.
15. P. Texeira de Araújo, and G. Barbosa Oliveira: *Algoritmos Genéticos Aplicados al Ruteamiento Multicast en Internet, Contemplando Requisitos de Calidad de Servicio e Ingeniería de Tráfico*. VII Brazilian Symposium on Neural Networks (SBRN´02), 2002.
16. W. Zhengying, S. Bingxin, and Z. Erdun: *Bandwidth-delay-constraint least cost multicast routing based on heuristic genetic algorithm*. Computer Communications. 2001. Vol. 24. pp. 685-692.
17. Y. Seok, Y. Lee, Y Choi, and C. Kim: *Explicit Multicast Routing Algorithm for Constrained Traffic Engineering*. IEEE ISCC´02. Italia, 2002.
18. R. Fabregat, Y. Donoso, B. Barán, F. Solano, and J.L. Marzo: *Multi-objective Optimization Scheme for Multicast Flows: a Survey, a Model and a MOEA Solution*. IFIP/ LANC 2005.
19. Spring, R Mahajan, and D. Wetheral: *Measuring ISP topologies with Rocketfuel*. Proceedings of the ACM SIGCOMM´02 Conference, 2002.