Omicron ACO

Osvaldo Gómez
Universidad Nacional de Asunción
Centro Nacional de Computación
Asunción, Paraguay
ogomez@cnc.una.py

and

Benjamín Barán
Universidad Nacional de Asunción
Centro Nacional de Computación
Asunción, Paraguay
bbaran@cnc.una.py

Abstract

Ant Colony Optimization (ACO) is a metaheuristic inspired by the foraging behavior of ant colonies that has been successful in the resolution of hard combinatorial optimization problems like the TSP. This paper proposes the Omicron ACO (OA), a novel population-based ACO alternative designed as an analytical tool. To experimentally prove OA advantages, this work compares the behavior between the OA and the $\mathcal{MMAS}$ as a function of time in two well-known TSP problems. A simple study of the behavior of the OA as a function of its parameters proves its robustness.

Keywords: Artificial Intelligence, Ant Colony Optimization, Omicron ACO, $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System.

1 Introduction

Ant Colony Optimization (ACO) is a metaheuristic proposed by Dorigo et al. that has been inspired by the foraging behavior of ant colonies [3]. In the last few years ACO has empirically shown its effectiveness in the resolution of several different NP-hard combinatorial optimization problems; however, still little theory is available to explain the reasons underlying ACO's success. Birattari et al. developed a formal framework of ant programming with the goal of gaining deeper understanding on ACO [1], while Meuleau and Dorigo studied the relationship between ACO and Stochastic Gradient Descent [10]. Gutjahr presented a convergence proof for a particular ACO algorithm called Graph-based Ant System (GBAS) that has an unknown empirical performance [7]. He proved that the GBAS converges, with a probability that could be made arbitrarily close to 1, to the optimal solution of a given problem instance. Later, Gutjahr demonstrated for a time-dependent modification of the GBAS that its current solutions converge to an optimal solution with probability exactly equal to 1 [8]. Stützle and Dorigo presented a short convergence proof for a class of ACO algorithms called $\text{ACO}_{gb,\tau_{min}}$ [11], where *gb* indicates that the global best pheromone update is used, while τ_{min} indicates that a lower limit on the range of the feasible pheromone trail is forced. They proved that the probability of finding the optimal solution could be made arbitrarily close to 1 if the algorithm is run for a sufficiently large number of iterations.

In search of new ACO analytical tools, a simple algorithm preserving certain characteristics of ACO was developed. This is how the Omicron ACO (OA) was conceived and its name comes from the main parameter used, which is *Omicron* (*O*). OA was motivated by the ideas behind ACO, i.e. the search for nearby good solutions. OA was first designed with theoretical motivations to study convergence properties [4], but proved to be very useful also in practical applications. Therefore, this paper compares OA with respect to one of the best-known ACO algorithms, the $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System ($\mathcal{MMAS}$) proposed by Stützle and Hoos [12]. OA is a more straightforward utilization of the successful reasons of ACO. As a consequence, OA outperforms $\mathcal{MMAS}$ in the preliminary experimental results shown in this work. Besides, its simplicity and decreased sensibility to input parameters make it easy to configure.

This paper is organized as follows. In Section 2 the test problem and the definitions are presented. The standard ACO approach, the ideas that motivated OA and its pseudocode are given in Section 3. In Section 4, a performance comparison between $\mathcal{MMAS}$ and OA, and a simple study of the OA as a function of its parameters are made. Experimental results are explained in Section 5. Finally, the conclusions and future work are presented in Section 6 .

2 Test Problem

In this paper the symmetric Traveling Salesman Problem (TSP) is used as a test problem for comparing the analyzed algorithms. The TSP is a hard combinatorial optimization problem, easy to understand, which has been considerably studied by the scientific community. Researchers have applied ACO successfully to this problem [3, 12]. To make performance comparisons, standard TSP instances extracted from TSPLIB[1] library have been used in this work. The TSP can be represented by a complete graph $G = (N, A)$ with N being the set of nodes, also called cities, and A being the set of arcs fully connecting the nodes. Each arc (i, j) is assigned a value $d(i, j)$ which represents the distance between cities i and j. The TSP is the problem of finding the shortest closed tour visiting each of the $n = |N|$ nodes of G exactly once. For symmetric TSPs, the distances between the cities are independent of the direction of traversing the arcs, that is $d(i, j) = d(j, i)$ for every pair of nodes. Suppose that r_x and r_y are TSP tours or solutions over the same set of n cities. For this work, $l(r_x)$ denotes the length of tour r_x. The distance between r_x and r_y is defined as n minus the number of edges contained in both r_x and r_y.

3 Ant Colony Optimization

Ant Colony Optimization (ACO) is a metaheuristic inspired by the behavior of ant colonies [3]. In the last few years, ACO has received increased attention by the scientific community as can be seen by the growing number of publications and the different fields of application [12]. Even though there exist several ACO variants, what can be considered a standard approach is next presented [5].

3.1 Standard Approach

ACO uses a pheromone matrix $\tau = \{\tau_{ij}\}$ for the construction of potential good solutions. The initial values of τ are set $\tau_{ij} = \tau_{init}\ \forall(i, j)$, where $\tau_{init} > 0$. It also takes advantage of heuristic information using

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

$\eta_{ij} = 1/d(i,j)$. Parameters α and β define the relative influence between the heuristic information and the pheromone levels. While visiting city i, $\mathcal{N}_i$ represents the set of cities not yet visited. The probability of choosing a city j at city i is defined as

$$\mathcal{P}_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{\forall g \in \mathcal{N}_i} \tau_{ig}^{\alpha} \cdot \eta_{ig}^{\beta}} & \text{if } j \in \mathcal{N}_i \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

At every generation of the algorithm, each ant of a colony constructs a complete tour using (1), starting at a randomly chosen city. Pheromone evaporation is applied for all (i,j) according to $\tau_{ij} = (1-\rho) \cdot \tau_{ij}$, where parameter $\rho \in (0,1]$ determines the evaporation rate. Considering an elitist strategy, the best solution found so far r_{best} updates τ according to $\tau_{ij} = \tau_{ij} + \Delta\tau$, where $\Delta\tau = 1/l(r_{best})$ if $(i,j) \in r_{best}$ and $\Delta\tau = 0$ if $(i,j) \notin r_{best}$. For one of the best performing ACO algorithms, the $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System ($\mathcal{MMAS}$) [12], minimum and maximum values are imposed to τ (τ_{min} and τ_{max}).

3.2 Omicron ACO

OA was initially inspired by $\mathcal{MMAS}$, an ACO currently considered among the best performing ACOs for the TSP [12]. It is based on the hypothesis that it is convenient to search for nearby good solutions [2, 12].

The main difference between $\mathcal{MMAS}$ and OA is the way the algorithms update the pheromone matrix. In OA, a constant pheromone matrix τ^0 with $\tau_{ij}^0 = 1$, $\forall i,j$ is defined. OA maintains a population $P = \{P_x\}$ of m individuals or solutions, the best unique ones found so far. The best individual of P at any moment is called P^*, while the worst individual P_{worst}.

In OA the first population is chosen using τ^0. At every iteration a new individual P_{new} is generated, replacing $P_{worst} \in P$ if P_{new} is better than P_{worst} and different from any other $P_x \in P$. After K iterations, τ is recalculated. First, $\tau = \tau^0$; then, O/m is added to each element τ_{ij} for each time an arc (i,j) appears in any of the m individuals present in P. The above process is repeated every K iterations until an end condition is reached (see pseudocode for details). Note that $1 \leq \tau_{ij} \leq (1+O)$, where $\tau_{ij} = 1$ if arc (i,j) is not present in any P_x, while $\tau_{ij} = (1+O)$ if arc (i,j) is in every $P_x \in P$.

Similar population-based ACO algorithms (P-ACO) [5, 6] were designed by Guntsch and Middendorf for dynamic combinatorial optimization problems. The main difference between the OA and the *Quality* Strategy of P-ACO is that OA does not allow identical individuals in its population. Also, OA updates τ every K iterations, while P-ACO updates τ every iteration.

Next, the pseudocode of the OA considering a TSP with n cities is presented.

Pseudocode of the main *Omicron ACO*

```
Input parameters: n, matrix D = {d_ij}, O, K, m, α, β
Output parameter: P  (m best found solutions)

τ = Initialize the pheromone matrix ()
P = Initialize the population (τ)
REPEAT UNTIL end condition
    REPEAT K TIMES
        P_new = Construct a solution (τ)
        IF (length(P_new) < length(P[0])) and (P_new ≠ all elements of P)
            P = Update population (P_new , P)
    τ = Update pheromone matrix (P)
```

Pseudocode of the function *Initialize the pheromone matrix ()*

```
REPEAT for every arc (i,j)
      τ[i,j] = 1
```

Pseudocode of the function *Initialize the population (τ)*

```
x = 0
WHILE x < m
    P_new = Construct a solution (τ)
    IF  P_new ≠ all chosen elements of P
        P[x] = P_new
        x = x + 1
P = Sort P from worst to best considering the tour length   /*  P_worst = P[0]  */
```

Pseudocode of the function *Construct a solution (τ)*

```
P_new[0] = Select a city randomly
x = 1
WHILE x < n
   P_new[x] =  Select a city randomly considering equation (1)
   x = x + 1
```

Pseudocode of the function *Update population (P_{new} , P)*

```
P[0] = P_new
P = Sort P efficiently from worst to best considering the tour length
      /*  P_worst = P[0]  */
```

Pseudocode of the function *Update pheromone matrix (P)*

```
τ = Initialize the pheromone matrix ()
x = 0
WHILE x < m
   REPEAT for every arc (i,j) of P[x]
      τ[i,j] = τ[i,j] + O/m
   x = x + 1
```

4 Experimental Results

For the following experimental results, a 2 GHz computer with 256 MB of RAM was used. The programming language chosen was C and the operating system was Linux. First, a comparative study between $\mathcal{MMAS}$ and OA is presented. Then, the behavior of OA as a function of its parameters is studied.

4.1 Comparative Study Between OA and $\mathcal{MMAS}$

As a performance reference, the parameters for the $\mathcal{MMAS}$ algorithm were extracted from [12], where $\alpha = 1$ and $\beta = 2$ were always used and the same problems were solved. To make a fair comparison, the same parameters $\alpha = 1$ and $\beta = 2$ were also used for the OA. The rest of the parameters were found empirically, searching for a balanced behavior between the speed of convergence and the quality of the final solution, choosing $O = 600$, $m = 25$ and $K = 1,000$. Because no attempt was done to optimize OA parameters or to make them time-dependent, $\mathcal{MMAS}$ with pheromone trail smooth was not chosen for comparison.

Empirical observations were done as a function of time given that the concept of iteration or generation is not the same for both algorithms. In general, OA produces better solutions than $\mathcal{MMAS}$ from the very beginning and converges to a slightly better result. As an example, for the problem eil51 with 51 cities studied in [12], the behavior of the best solution found for each algorithm was studied. The mean of the evolution of both algorithms in 25 runs can be seen in Figure 1. In Figure 2 the ranges are modified to show the clear advantage of OA at the convergence stage.

The OA results are promising considering two reasons. First, the minimum tuning work made in the algorithm parameters and second, the fact that partial results show an increased robustness, since using exactly the same parameters similar results are observed in the 100 cities problem kroA100.

In Figure 3 we observe the mean in 25 runs of the evolution of the best length of both algorithms for a larger problem, the well known kroA100 [12]. In Figure 4 the ranges are modified, as in Figure 2, to show the advantage of OA over $\mathcal{MMAS}$, considering convergence. Once more, considering mean behavior in 25 runs, OA outperforms $\mathcal{MMAS}$.

Note that best results were obtained and bigger instances were solved using $\mathcal{MMAS}$ with local search [12]. In these preliminary tests, only little instances of the TSP were solved (because no local search was implemented) to make a comparison between both algorithms without any interference.

4.2 Simple Study of the Behavior of OA as a Function of its Parameters

To verify the robustness of OA, its behavior has been observed as a function of its parameters. In Figure 5 (a) the evolution of the mean of the best individual's length is observed; 25 runs were made using $m = 25$, $K = 1,000$ and $O = 300$, $O = 600$ and $O = 1,200$. In Figure 5 (b), the 25 runs were made using $O = 600$, $K = 1,000$ and $m = 13$, $m = 25$ and $m = 50$, while in Figure 5 (c) using $O = 600$, $m = 25$ and $K = 500$, $K = 1,000$ and $K = 2,000$.

Clearly, a significant variation of the parameters used in Section 4.1 did not alter considerably the behavior of the algorithm. Using a greater value of O, the behavior is almost identical, while a smaller value shows a

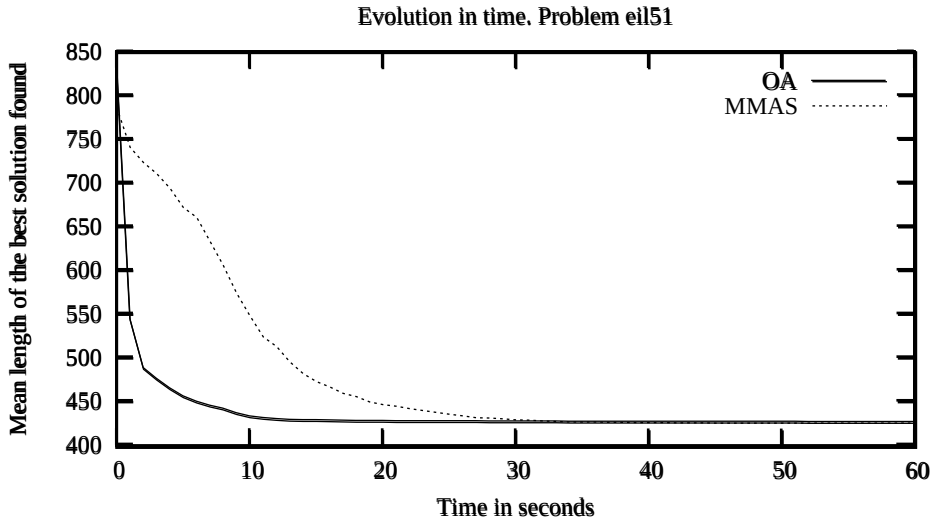

Figure 1: Comparison between the mean behavior of the proposed OA and the $\mathcal{MMAS}$

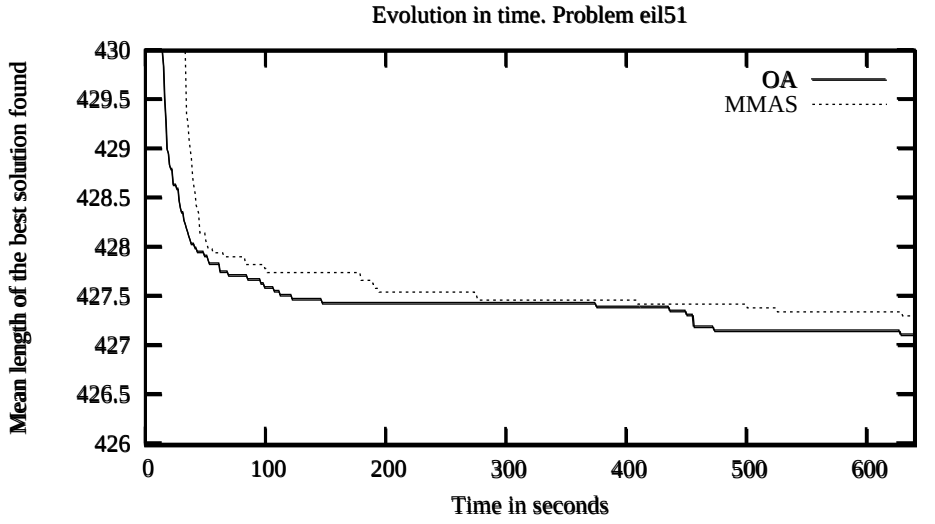

Figure 2: Comparison between the mean behavior of the proposed OA and the $\mathcal{MMAS}$ using different ranges, allowing a detailed observation of the convergence characteristics

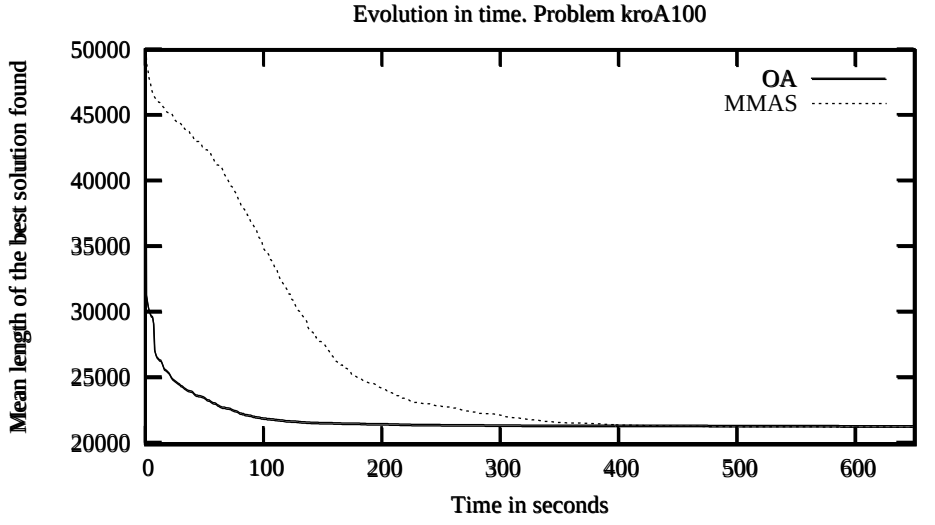

Figure 3: Comparison between the mean behavior of the proposed OA and the $\mathcal{MMAS}$

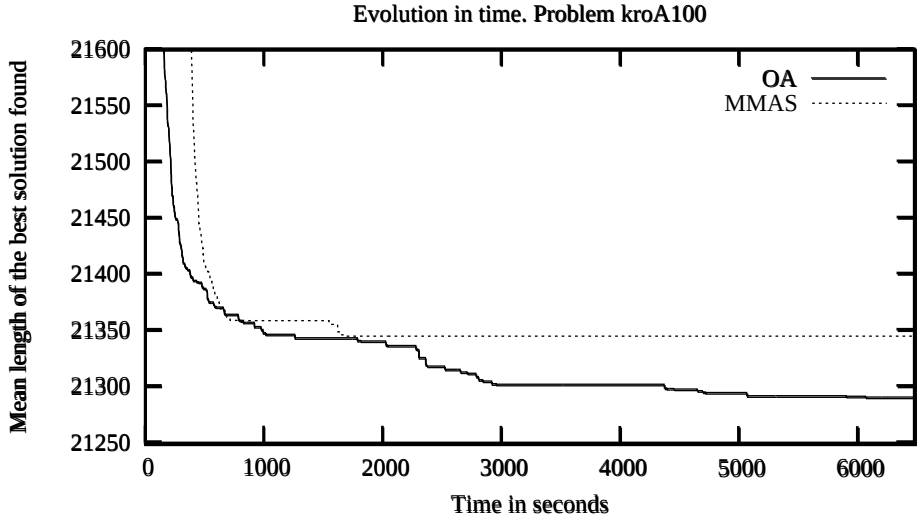

Figure 4: Comparison between the mean behavior of the proposed OA and the $\mathcal{MMAS}$ using different ranges, allowing a detailed observation of the convergence characteristics

slight fall in its performance. This is due to the diminished strength of the search for nearby good solutions. Using a larger m, the search was slower at the beginning but resulted in a slightly better solution at the end. A smaller m did the opposite. This can be seen as if the increase of the number of individuals delays the search, but ensures a good final search zone. By increasing the parameter K, a slower initial progress was observed, while decreasing it did the opposite. This can be understood because the more frequent update of the pheromone matrix allows to search for nearby better solutions in advance.

5 Explaining Experimental Results

To explain the reasons why OA outperforms $\mathcal{MMAS}$, one of the best-known ACO algorithms [12], it is useful to remember two main reasons why ACO is a good algorithm for a TSP with globally convex structure [2, 4, 9].

1. Given a population of good enough solutions, better solutions may be found with larger probability closer (considering the distance concept explained in Section 2) to good solutions than to bad ones. All ACO algorithms, including OA and $\mathcal{MMAS}$ are based on this known property.

2. Because the existence of local optimal solutions, it is better to search within a region defined by a whole population of good solutions than only close to the best-known one. Even though all ACO algorithms use this property in one way or another, OA is the one that best exploits this property in an explicit way, looking for good solutions mainly in the subspace spanned by the best m known solutions, without concentrating its search mainly around the best current solution which usually is only a local optimum.

Therefore, the main reason OA outperforms $\mathcal{MMAS}$ is its ability to search explicitly in a whole subspace Ω spanned by m good solutions, instead of mainly increasing the amount of pheromone of only one good solution. At the beginning, this property makes OA converge faster to a good region, given that it uses more information of each cycle (up to m pheromone updates, instead of just one). At the end, OA searches for the best solution close to a whole subspace Ω, without giving more importance to any of the m individuals of population P. On the contrary, other ACOs as $\mathcal{MMAS}$ look for new solutions mainly near the best-known solution, which usually is only a local optimum.

Finally, it may be noticed that OA completely forgets old solutions that are not members of the population, while $\mathcal{MMAS}$ evaporates the pheromone very slowly at the arcs of these old bad solutions, slowing down its convergence.

6 Conclusions and Future Work

This paper presents Omicron ACO (OA), a new algorithm inspired by one of the best ACO algorithms, the $\mathcal{MMAS}$ [12]. OA uses one of the principles of the $\mathcal{MMAS}$ success more directly, the search for nearby good solutions in problems of combinatorial optimization like the TSP with globally convex structure of its

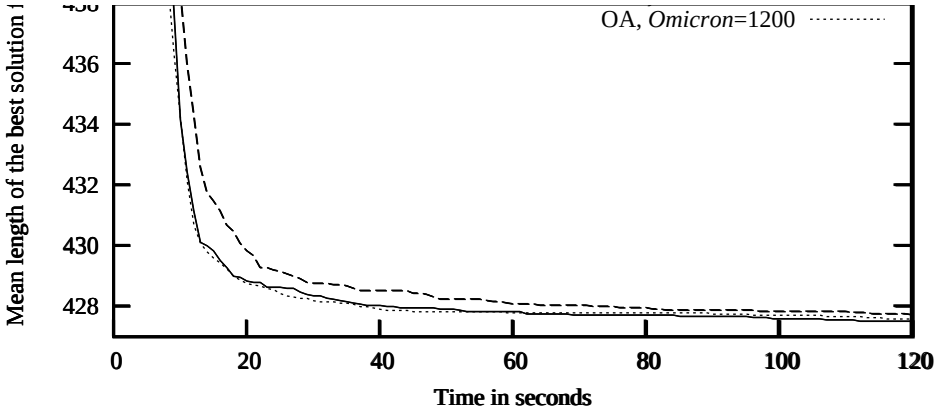

FIGURE (b). Evolution in time for different *m*

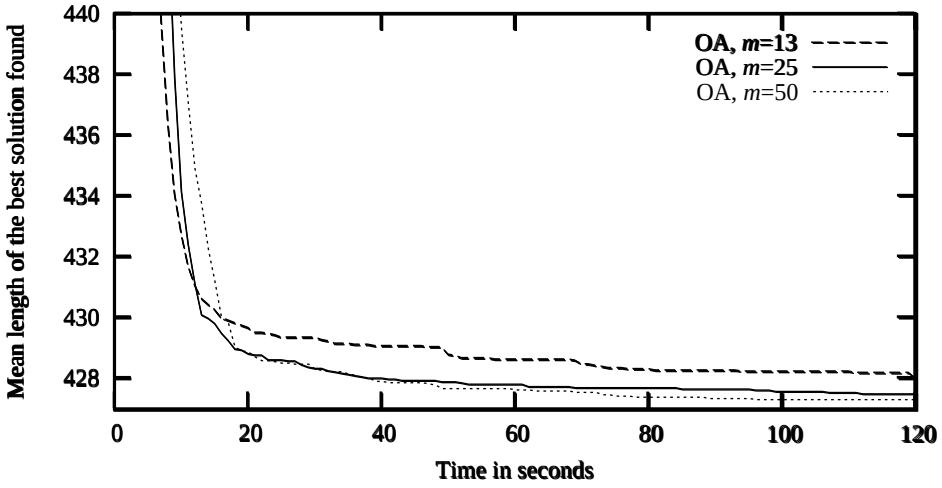

FIGURE (c). Evolution in time for different *K*

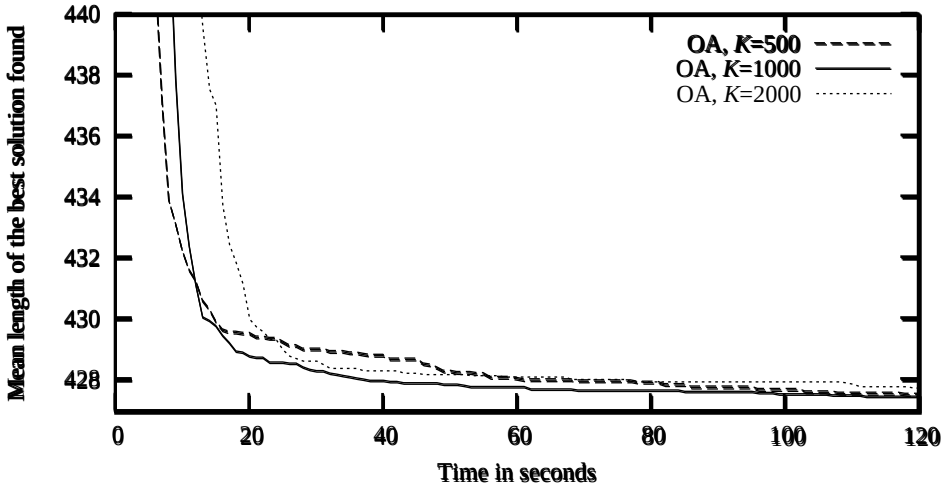

Figure 5: Comparison among the mean behavior of the proposed OA for the problem eil51, for different values of the parameters O, m and K

search space [2]. This new OA algorithm outperforms $\mathcal{MMAS}$ for these preliminary tests and it is also more robust with relation to its initial parameters; therefore, it is easier to configure.

Finally, its conceptual simplicity and the fact that it uses the same foundations as ACO allow a deeper study of the reasons of ACO's success.

After these encouraging preliminary results, the authors are working on the comparison of OA for TSPLIB problems to other ACO algorithms as P-ACO and $\mathcal{MMAS}$ with pheromone trail smooth and local search. Future work may concentrate on a deeper theoretical study of OA and its application to other combinatorial optimization problems in comparison with other world-class metaheuristics.

References

[1] M. Birattari, G. Di Caro, and M. Dorigo. For a Formal Foundation of the Ant Programming Approach to Combinatorial Optimization. Part 1: The problem, the representation, and the general solution strategy. Technical Report TR-H-301, ATR-Human Information Processing Labs, Kyoto, Japan, 2000.

[2] Kenneth D. Boese. Cost Versus Distance in the Traveling Salesman Problem. Technical Report 950018, University of California, Los Angeles, Computer Science Department, May 19, 1995.

[3] Marco Dorigo and Gianni Di Caro. The Ant Colony Optimization Meta-Heuristic. In David Corne, Marco Dorigo, and Fred Glover, editors, *New Ideas in Optimization*, pages 11–32. McGraw-Hill, London, 1999.

[4] Osvaldo Gómez and Benjamín Barán. Reasons of ACO's Success in TSP. In M. Dorigo, M. Birattari, C. Blum, L. M. Gambardella, F. Mondada, and T. Stützle, editors, *Proceedings of ANTS 2004 - Fourth International Workshop on Ant Colony Optimization and Swarm Intelligence*, volume 3172 of *LNCS*, Brussels, September 2004. Springer-Verlag.

[5] Michael Guntsch and Martin Middendorf. A Population Based Approach for ACO. In Stefano Cagnoni, Jens Gottlieb, Emma Hart, Martin Middendorf, and Günther Raidl, editors, *Applications of Evolutionary Computing, Proceedings of EvoWorkshops2002: EvoCOP, EvoIASP, EvoSTim*, volume 2279, pages 71–80, Kinsale, Ireland, 3-4 2002. Springer-Verlag.

[6] Michael Guntsch and Martin Middendorf. Applying Population Based ACO to Dynamic Optimization Problems. In *Ant Algorithms, Proceedings of Third International Workshop ANTS 2002*, volume 2463 of *LNCS*, pages 111–122, 2002.

[7] Walter J. Gutjahr. A graph-based Ant System and its convergence. *Future Generation Computer Systems*, 16(8):873–888, June 2000.

[8] Walter J. Gutjahr. ACO Algorithms with Guaranteed Convergence to the Optimal Solution. *Information Processing Letters*, 82(3):145–153, May 2002.

[9] T. C. Hu, Victor Klee, and David Larman. Optimization of globally convex functions. *SIAM Journal on Control and Optimization*, 27(5):1026–1047, September 1989.

[10] Nicolas Meuleau and Marco Dorigo. Ant colony optimization and stochastic gradient descent. *Artificial Life*, 8(2):103–121, 2002.

[11] T. Stützle and M. Dorigo. A Short Convergence Proof for a Class of Ant Colony Optimization Algorithms. *IEEE Trans. on Evol. Comput.*, 6:358–365, August 2002.

[12] Thomas Stützle and Holger H. Hoos. $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System. *Future Generation Computer Systems*, 16(8):889–914, June 2000.