5

On Ant Colony Optimization Algorithms for Multiobjective Problems

Jaqueline S. Angelo and Helio J.C. Barbosa
Laboratório Nacional de Computação Científica LNCC/MCT
Brasil

1. Introduction

Many practically relevant problems have several objectives to be maximized or minimized, taking us into the area of Multiple Objective Optimization (MOO).
In multi-objective optimization several conflicting objectives have to be simultaneously optimized. Therefore, there is usually no single solution which would give the best values for all the objective functions considered by the decision maker.
Instead, in a typical MOO problem, there is a set of alternatives that are superior to the remainder when all the objectives are considered. This set of so-called non-dominated solutions is known as the Pareto optimum set, and provides many options for the decision-maker. Usually only one of these solutions is to be chosen.
Due to the availability of and familiarity with single-objective optimizers, it is still common to combine all objectives into a single quantity to be optimized. Classical methods are usually based on reducing the multi-objective problem to a single objective one by combining (usually linearly) the objectives into one. It must be noted here that the (possibly conflicting) objectives are also non commensurable (cost, weight, speed, etc.), which makes it difficult to combine them into a single measure.
As a result, those classical techniques have serious drawbacks (Coello et al., 2002); as they require *a priori* information about the problem (weights or thresholds), which are usually not available, and many runs are needed in order to obtain different solutions, since only one solution is obtained in each run.
However, nowadays it is becoming clear that multi-objective problems can be successfully dealt with by employing a population based stochastic technique able to produce an approximation of the true set of non-dominated solutions in a single run.
Due to the increasing complexity of the problems being tackled today, those methods are often based on nature-inspired metaheuristics such as evolutionary algorithms, particle swarm optimization, artificial immune systems, and ant colony optimization (ACO) (Gandibleux et al., 2004; Silberholz & Golden, 2009).
Due to the good results obtained by ACO algorithms in a wide range of single-objective problems (Dorigo & Stützle, 2004) it is not surprising that several algorithms based on ACO have been proposed to solve multiple-objective problems.
ACO is a constructive search technique to solve difficult combinatorial optimization problems, which is inspired by the behaviour of real ants. While walking, ants deposit pheromone on the ground marking a path that may be followed by other members of the colony. Shorter

paths, as they accumulate pheromone faster than the longer ones, have a higher probability of being used by other ants, which again reinforce the pheromone on that path.
Artificial ants probabilistically build solutions considering: a pheromone trail (matrix τ), that encodes a "memory" about the search process, and is updated by the ants; and a heuristic information (matrix η), that represents *a priori* information about the problem instance to be solved.
In recent reviews (García-Martínez et al., 2007; Angus & Woodward, 2009; López-Ibáñez & Stützle, 2010) several multi-objective ACO (MOACO) algorithms were described for combinatorial optimization problems. The first paper proposed a taxonomy for MOACO algorithms, focused on an empirical analysis of the algorithms when solving the Bi-objective Traveling Salesman Problem, classifying them according to the use of one or several pheromone trails, and one or several heuristic informations. They also categorised the algorithms as Pareto-based, when the algorithm returns a set of non-dominated solutions (the Pareto set) or non-Pareto based, when it returns a single solution as output. The second review proposed a new taxonomy expanding the previous classification based on different features of the MOACO algorithms while the last one developed a technique for automatically configuring MOACO algorithms, presenting other algorithmic components that were not considered in previous works.
The objective of this chapter is to provide a comprehensive view of the use of ant colony optimization techniques in the realm of multiple-objective problems. The taxonomies proposed in (García-Martínez et al., 2007; Angus & Woodward, 2009; López-Ibáñez & Stützle, 2010) are considered in order to provide a global view of the current MOACO algorithms and their algorithmic components. This chapter also extends these taxonomies by including other MOACO algorithms that were not previously discussed on those papers.
This chapter presents in more detail some representative MOACO algorithms, indicating how well-known single objective ACO algorithms can be extended in order to tackle multiple objectives. The problem of comparing the quality of the solutions generated by MOACO algorithms is also discussed by considering different performance metrics and an empirical attainment function.

2. Multiple objective optimization problem

In a single-objective optimization problem one is trying to find a single solution, called optimal solution, that minimizes or maximizes an specified objective function subject to given constraints. On the other hand, multi-objective optimization problems consider several objectives that have to be simultaneously optimized.
With no loss of generality, the minimization case will be considered and the MOO problem reads

$$\begin{array}{l} \text{minimize } \mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_k(\mathbf{x})] \\ \text{subject to } \mathbf{x} = (x_1, x_2, ..., x_n) \in \mathcal{S} \end{array} \tag{1}$$

where $\mathcal{S}$ is the *feasible region (set)*, $\mathbf{f}(\mathbf{x})$ is an objective vector with $k(\geq 2)$ objective functions to be minimized and $\mathbf{x}$ is a decision vector, which is a feasible solution if $\mathbf{x} \in \mathcal{S}$. The image of the feasible set, denoted by $\mathcal{Z}(= \mathbf{f}(\mathcal{S}))$, is known as the *feasible objective region (set)*. The elements of $\mathcal{Z}$ are the objective (function) vectors denoted by $\mathbf{f}(\mathbf{x})$ or $\mathbf{z} = (z_1, z_2, ..., z_k)^T$, where $z_i = f_i(\mathbf{x})$ for all $i = 1, ..., k$ are objective values.
The space, of which the feasible set $\mathcal{S}$ is a subset, is called the *decision space* and the space from which the objective values are taken is called the *objective space*.

If the objective functions are not conflicting, then a solution can be found where every objective function attains its optimum. However, in MOO frequently those objectives are conflicting (i.e, the improvement of one objective leads to another objective degradation) and possible non-commensurable (i.e., in different units). In such case, there is usually no single optimal solution, but a set of alternatives that outperform the remainder when all objectives are considered. Such solutions are called non-dominated solutions, the Pareto optimal set.
In MOO, usually only one of the solutions in the Pareto set is to be chosen. In this way, the decision maker (DM) plays an important role when choosing a single solution that better satisfies his or her preferences.
Different approaches are considered in the literature when the preference of the DM are used to guide the search (Miettinen, 1999).

- No Preference Articulation: the preferences of the DM are not taken into consideration. The problem can be solved by a simple method and the solution obtained is presented to the DM which will accept or reject it.
- A Priori Preference Articulation: known as preference-based, the hopes and opinions of the DM are taken into consideration before the solution process. Those methods require that the DM knows beforehand the priority of each objective.
- A Posteriori Preference Articulation: no preferences of the DM are considered. After the Pareto set has been generated, the DM chooses a solution from this set of alternatives.
- Interactive Preference Articulation: the DM preferences are continuously used during the search process and are adjusted as the search continues.

When the multiobjective optimization problem is converted into a simplistic single objective problem, the DM is invoked before the optimization step. In this case, the DM must have a thorough knowledge of the priority of each objective. Alternatively, the problem can be treated as a true multiobjective problem, invoking the DM either after the optimization process or in continuous interaction during the search process (Branke et al., 2008).

2.1 Pareto optimality

The definition of optimality for multi-objective problems is based on the Pareto optimality concept. Pareto dominance can be used to evaluate the relation between two candidate solutions in MOO (Tan et al., 2005). Without loss of generality, for a minimization problem, a decision vector $\mathbf{x} \in \mathcal{S}$ dominates another decision vector $\mathbf{x}' \in \mathcal{S}$ ($\mathbf{x} \prec \mathbf{x}'$) if and only if

$$f_i(\mathbf{x}) \leq f_i(\mathbf{x}') \quad \forall i \in \{1,...,k\} \quad \text{and} \quad \exists j \in \{1,...,k\} : f_j(\mathbf{x}) < f_j(\mathbf{x}') \tag{2}$$

A decision vector $\mathbf{x}^* \in \mathcal{S}$ is said to be Pareto-optimal when there is no other $\mathbf{x} \in \mathcal{S}$ that dominates $\mathbf{x}^*$. An objective vector $\mathbf{z}^* \in \mathcal{Z}$ is said to be Pareto-optimal when there is no other $\mathbf{z} \in \mathcal{Z}$ that dominates $\mathbf{z}^*$. The set of non-dominated solutions is called the *Pareto set* and the corresponding set of objective vectors is called the *Pareto front*. Therefore, the Pareto set is the best collection of solutions to the problem.
Figure 1 illustrates a decision space $\mathcal{S} \subset \mathbb{R}^3$ and an objective space $\mathcal{Z} \subset \mathbb{R}^2$. The bold line contains all the Pareto objective vectors, the Pareto front (Ehrgott, 2005).

3. Performance assessment

In multi-objective optimization, performance analysis is a difficult task, since one is trying to find a good approximation for a set: the Pareto front. Performance metrics are then needed

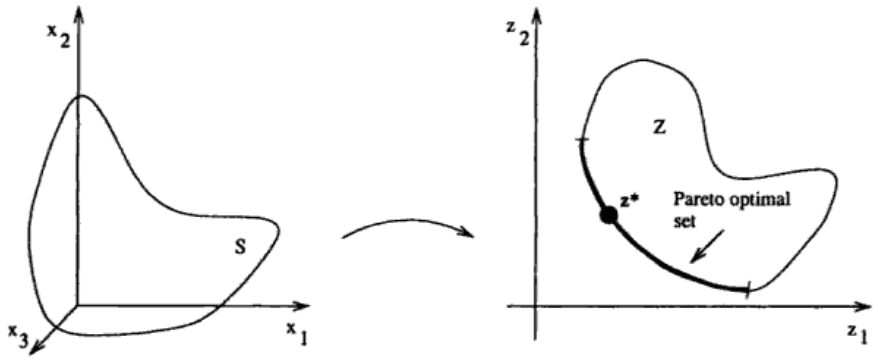


Fig. 1. Decision and objective spaces, and the Pareto front for a minimization problem with two objectives and three decision variables.

to assist the identification of the best non-dominated solution set whenever the results are difficult to interpret visually. However, the assessment problem itself is a multi-objective problem, since various aspects should be considered and no single metric encompasses them all.
Zitzler *et al* (Zitzler et al., 2000) suggest three main criteria that should be taken into account when assessing the quality of the non-dominated solutions set:

- The distance between the resulting non-dominated set and the true Pareto front should be minimized.
- A good (in most cases uniform) distribution of the obtained solutions is desirable.
- The size of the non-dominated front should be maximized, i.e., for each objective, a wide range of distinct solutions should be presented.

Many quality measures, or performance metrics, for comparing non-dominated solutions sets have been proposed. These quality measures can be classified into two categories (López-Ibáñez, 2004): unary measures, which associate a quality value to a Pareto set, using the Pareto optimal set as a reference; and binary measures, which compares two different Pareto sets.
The unary measures are defined in the case when the optimal Pareto set is known. Usually the Pareto optimal set is not known, so its necessary to calculate a pseudo-optimal Pareto set as a reference set, which is an approximation of the true Pareto optimal set.
The following topics describe some performance metrics, where ER, ONVG and $\mathcal{S}$ metrics are unary measures while $\mathcal{C}$ is a binary measure (Zitzler et al., 2000; Knowles & Corne, 2002; Li et al., 2007):

Error ratio (ER). The ER metric is defined as

$$Er(Z) = \frac{\sum_{i=1}^{n} e_i}{n}$$

where n is the number of vectors in the approximation set Z. Using Z^* as a reference set, which could be the Pareto front, $e_i = 0$ when i-th vector is in Z^*, and 1 otherwise. The ER metric measures the proportion of non-true Pareto vectors in Z. Lower values of the error ratio represent better non-dominated sets.

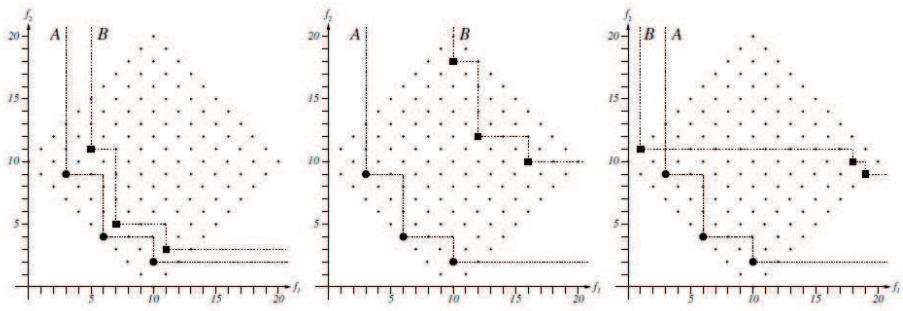


Fig. 2. The two figures on the left side indicate that the approximate set A dominates the approximate set B, but in one case the two sets are much closer than in the other case. On the right side, A and B cannot be compared, but it is clear that A gives more useful solutions than B.

Overall Non-dominated Vector Generation (ONVG). The ONVG metric is defined as $|Z|$, where Z represents a set of solutions. It measures the number of distinct non-dominated solutions obtained.

The $\mathcal{S}$ metric. The $\mathcal{S}$ metric measures the hypervolume of a multi-dimensional region covered by the set A and a reference point. It computes the size of the region dominated by A. A convenient reference point is needed so as not to induce misleading results.

The $\mathcal{C}$ metric. The $\mathcal{C}$ metric compares a pair of non-dominated sets by computing the fraction of each set that is covered by the other. $\mathcal{C}$ maps the ordered pair (A,B) into the interval $[0,1]$:

$$\mathcal{C}(A,B) = \frac{|b \in B, \exists a \in A : a \preceq b|}{|B|}$$

The value $\mathcal{C}(A,B) = 1$ means that all solutions in B are dominated by or equal to solutions in A. The opposite $\mathcal{C}(A,B) = 0$ means that none of the solutions in B are covered by the set A. It is important to note that $\mathcal{C}(A,B)$ and $\mathcal{C}(B,A)$ have to be considered, since $\mathcal{C}(A,B)$ is not necessarily equal to $1 - \mathcal{C}(B,A)$.

The performance metrics are not easy to define and it is probably not possible to establish a single metric that satisfies all users preferences in a satisfactory way, since each one of them measures different aspects of the non-dominated set found. Besides, depending on the solution's distribution over the Pareto set, some performance metrics could not adequately express the quality of the analysed set (Knowles, 2005). Figure 2 (Knowles et al., 2006) illustrates three examples of sets in the objective space and the difficulties in assessing how much better one set is over another.

Another way to measure the quality of the Pareto set is to analyse the results by means of a graphical representation (López-Ibáñez et al., 2009) based on the empirical attainment function (EAF) (Fonseca et al., 2001).

The attainment function gives the probability that an arbitrary point in the objective space is attained by (dominated by or equal to) the outcome of a single run of a particular algorithm. In practice, this function is not known, but it can be estimated collecting the outcome data from several independent runs of an algorithm.

The EAFs of two algorithms can be compared by computing the difference of the EAF values for each point in the objective space. Differences in favor of one algorithm indicate that those points are more likely to be attained by that algorithm than by its competitor, and, hence, that the performance of that algorithm is better (in that region of the objective space) than the performance of its competitor.
One can find a more detailed description of attainment functions in (Fonseca et al., 2001; Knowles et al., 2006)

4. Ant colony approach

The Ant Colony Optimization (ACO) is a metaheuristic inspired by the behaviour of real ants. Ants and other insects that live in a colony, like bees, termites and wasps, can be seen as distributed systems, that in spite of the simplicity of each individual, present a high level of social organization when observed together. Some examples of ant colony's capabilities found in (Dorigo et al., 1999) are: division of labor and task allocation, cemetery organization and brood sorting, cooperative transport and finding the shortest path between two or more locations (often between a food source and a nest).
The first ACO algorithm developed was initially applied to the Traveling Salesman Problem (Dorigo, 1992). The algorithm was based on the ant colony capability to find the shortest path between a food source and a nest. The algorithm uses artificial ants that cooperate on finding solutions to the problem through communication mediated by artificial pheromone trails.
While moving on the graph associated with the problem, artificial ants deposit pheromone on the edges traversed marking a path that may be followed by other members of the colony, which then reinforce the pheromone on that path. With this stigmergetic (Dorigo et al., 2000) communication, ants have their activities coordinated. This self-organizing behaviour results in a self-reinforcing process that leads to the formation of a path marked by high pheromone concentration, while paths that are less used tend to have a diminishing pheromone level due to evaporation.
This concept can be applied to any combinatorial optimization problem for which a constructive heuristic can be defined. The process of constructing solutions can be regarded as a walk on a construction graph where each edge of the graph represent a possible step the ant can take. ACO algorithms are essentially constructive, as ants generate solutions by adding solution components, corresponding to the edges chosen, to an initially empty solution until the solution is complete.
The ants movement is guided by (i) a *heuristic information* (η) that represents *a priori* information about the problem instance to be solved and by (ii) a *pheromone trail* (τ) that encodes a memory about the ant colony search process which is continuously updated by the ants. In many cases η is the cost of adding the component (associated with the given graph edge) to the solution under construction. These values are used by the ant's heuristic rule to make probabilistic decisions on the next node to be visited (or next edge to be used). When all ants have generated their solutions, the pheromone trail is updated considering the quality of the corresponding candidate solutions: reinforcing components of good solutions (positive feedback) and applying a certain level of pheromone evaporation in all edges.
The ACO algorithms were initially used to solve combinatorial optimization problems, inspired by the path marking behaviour, and later applied to many other problems. Also, other ant colony capabilities have inspired computer scientists to use ACO on different types of applications. A survey of ACO algorithms and applications can be found in (Dorigo et al., 2006).

In the multi-objective case García-Martínez *et al* (García-Martínez et al., 2007) proposed a taxonomy of the multi-objective ACO (MOACO) algorithms according to two different criteria: (i) the use of only one or several pheromone trails; and (ii) the use of only one or several matrices of heuristic information. In this classification some of them are Pareto-based, when they return a set of non-dominated solutions (an approximation to the Pareto set), while others, that are not Pareto-based, return a single solution as output.
The pseudo-code for a standard ACO metaheuristic in multiobjective optimization is given by Algorithm 1.

Algorithm 1: MultiobjectiveACO

Set parameters;
Initialize pheromone trails τ ;
Initialize heuristic matrix η ;
Initialize Pareto set $\mathcal{P}$ as empty;
while *termination criteria not met* **do**
    ConstructAntSolution();
    ApplyLocalSearch() (optional);
    UpdateParetoSet();
    UpdateGlobalPheromone();
Return the Pareto set $\mathcal{P}$

In the routine *ConstructAntSolution()*, from a given initial point, ants start walking according to the decision policy of choosing the next node to be visited. This movement is guided by the pheromone trail and the heuristic information associated with the problem instance. After constructing a complete path (feasible solution), it is possible to apply a local search procedure in order to improve the solution obtained. When all ants have generated their solutions, the Pareto set is updated in the procedure *UpdateParetoSet()*, keeping all non-dominated solutions generated up to this point. Then the pheromone trails are updated in *UpdateGlobalPheromone()*, considering the quality of the candidate solutions generated as well as a certain level of pheromone evaporation. Each algorithm presents different ways to choose the nodes and to update the pheromone trails.

4.1 ACO for constrained problems

A multiobjective optimization problem consists in a set of solutions $\mathcal{S}$, a set of objective functions $\mathbf{f}$ which assigns for each objective a value $f_k(s)$ to each candidate solution $s \in S$, and a set of constraints Ω that must be satisfied.
Considering the Traveling Salesman Problem the only constraint is that all cities must be visited only once. In this case, the search space is restricted to permutations of the list of cities $(1,2,...,N)$.
An ACO algorithm deals with this constraint by equipping each ant with a memory that keeps track of the cities already visited during the tour construction. Thus, their choices are limited to the cities that have not been visited yet. However, for other problems such a simple procedure may not be available. When the constraints cannot be satisfied during the solution construction, other constraint handling techniques must be adopted.
Several techniques have been proposed in the literature in order to tackle constrained optimization problems which can be classified as *direct* (feasible or interior), when only feasible elements are considered, or as *indirect* (exterior), when both feasible and infeasible elements are used during the search process (Fonseca et al., 2007).

An MOACO algorithm that makes use of an indirect technique is the MOAQ algorithm (Mariano & Morales, 1999). The problem to be solved is the design of an irrigation water network. Basically, the optimization problem consists in minimizing the cost of the network and maximizing the profit, subject to ten different constrains. Constraint handling is performed by penalizing the solutions which violate such constraints.

4.2 Algorithmic components for MOACO

In the literature one can find various alternatives for the implementation of ACO algorithms for solving multiobjective combinatorial problems. They usually differ from each other with respect to the single objective ACO algorithms they were based on (such as AS, ACS and $\mathcal{MM}$AS), as well as due to variations in their algorithmic components.

In the following, we present some of the algorithmic components that were already examined, experimentally or comparatively, in the literature (García-Martínez et al., 2007; Angus & Woodward, 2009; López-Ibáñez & Stützle, 2010) concerning MOACO algorithms.

Multiple Colonies. In a multiple colony approach, a number of ants are set to constitute a colony. Each colony independently construct solutions considering its own pheromone and heuristic information, specializing its search on particular areas of the Pareto front. The colonies can cooperate with each other by: (i) exchanging solutions (information), using a shared archive of non-dominated solutions in order to identify dominated ones; or (ii) sharing solutions for updating the pheromone information, so that solutions generated by a certain colony affect the pheromone information of other colonies.

Pheromone and Heuristic Information. There are two standard models to define the pheromone/heuristic information: using one (*single*) or various (*multiple*) matrices . When multiple matrices are utilized, usually each matrix corresponds to one objective. With respect to the pheromone information, each matrix may contain different values depending on the implementation strategy applied. If a single pheromone matrix is used, the construction step is done similarly to single objective ACO algorithms. In this case, the pheromone information associated with each objective should be combined, so as to reduce the multiple objectives into a single one. The same is applied to the heuristic information.

Pheromone and Heuristic Aggregation. Whenever multiple matrices are used, they must be aggregated, using some form of aggregation procedure. In all of these cases, weights are needed to adjust the influence of each objective. Different techniques for setting the weights may lead to different search behaviour. For the MOACO algorithms, one can find in the literature three methods to aggregate the pheromone/heuristic matrices: (i) *the weighted sum*, where matrices are aggregated by a weighted sum ($\sum_{k=1}^{K} \lambda_k \tau_{ij}^k$), with K being the number of objectives; (ii) *the weighted product*, where matrices are aggregated by a weighted product ($\prod_{k=1}^{K} (\tau_{ij}^k)^{\lambda_k}$); and (iii) *random*, where at each construction step a randomly objective is selected to be optimized. Whenever weights are used for aggregating multiple matrices two strategies can be used for setting the weights, which are: (a) *dynamically*, where the objectives are combined dynamically so that different objectives can be emphasized at different times during the solution construction process, or ants can use specific weights to combine the matrices; and (b) *fixed*, where the weights are set a priori and each objective has the same importance during the entire algorithm run.

Component	Values
Colony	Single, Multiple
Pheromone matrix	Single, Multiple
Heuristic matrix	Single, Multiple
Aggregation	Weighted product, Weighted sum, Random
Weight setting	Dynamic, Fixed
Pheromone update	Non-dominated, Best-of-objective, Elite, All
Pareto archive	Offline, Online, No-archive

Table 1. MOACO algorithmic components and values

Pheromone Update. This algorithmic component can be implemented in many different ways. When only one matrix is used it is common to perform the update as in the single objective ACO algorithms, such as, selecting the iteration-best or the best-so-far (*elite solution*) solution to update the pheromone matrix. This same procedure can be applied when multiple matrices are used. In this case, one can select a set of iteration-best or best-so-far solutions to update the pheromone matrices, with respect to each objective (*best-of-objectives solutions*). Another way to update the pheromone matrices is to collect and store the non-dominated solutions in a external set. Only the solutions in the non-dominated set (*non-dominated solutions*) are allowed to update the pheromone. In this case, the individuals can update a specific pheromone matrix or many (or all) pheromone matrices. In other cases, all ants are allowed to update the pheromone matrices (*all solutions*).

Pareto Archive. For the Pareto-based MOACO algorithms, the Pareto set must be stored and updated during the algorithm execution. In many cases, the solution in the Pareto set is used to update the pheromone information. This algorithm component indicates how this set is stored and used during the algorithm run. In the MOACO literature, one can find two ways to do it. The first one is *offline storage*: the non-dominated solutions are stored in a external set and these solutions are not used for future solution construction. The Pareto set is used to update the pheromone information, and at the end of the execution, this set is returned as the final solution. The second one is *online storage*: the solutions in the Pareto set are connected to the pheromone update procedure. Each time the Pareto set is improved, the pheromone update procedure is guided by the improved set. This technique allows the pheromone matrix or matrices to reflect the state of the non-dominated set at any time. There are cases where the Pareto set is not used to update the pheromone information but it is used as a final solution.

As a reference guide to the MOACO taxonomy, Table 1 lists the algorithmic components of the MOACO algorithms that were discussed previously.

5. MOACO Algorithms

A large number of MOACO algorithms were developed during the last few years. This section presents different MOACO algorithms proposed in the literature. The main characteristics of the algorithms will be reviewed and some of them will be compared with the mono-objective ACO algorithm which they were based on.

The taxonomy proposed is based on the algorithmic components listed in the previous section, which follows the taxonomy proposed in (García-Martínez et al., 2007; Angus & Woodward, 2009; López-Ibáñez & Stützle, 2010).

Algorithm	$[\tau]$	$[\eta]$	Colony	Agg./Weight	τ update	P. archive
MOAQ (Mariano & Morales, 1999)	1	k	multiple	w.p-w.s/-	ND	off
COMPETants (Doerner et al., 2001)	k	k	multiple	w.s./fix.	BoO	none
BicriterionAnt (Iredi et al., 2001)	k	k	single	w.p./dyn.	ND	off
SACO (Vincent et al., 2002)	1	1	single	-/-	E	none
MACS (Barán & Schaerer, 2003)	1	k	single	w.p./dyn.	ND	on
MONACO (Cardoso et al., 2003)	k	1	single	w.p./dyn.	all	off
P-ACO (Doerner et al., 2004)	k	k	single	w.s./dyn.	BoO	off
M3AS (Pinto & Barán, 2005)	1	k	single	w.p./fix.	ND	off
MOA (Gardel et al., 2005)	1	1	single	w.p./dyn.	ND	off
MAS (Paciello et al., 2006)	1	k	single	w.p. /dyn.	ND	off
MOACSA (Yagmahan & Yenisey, 2010)	1	1	single	w.p./dyn.	E	none

Table 2. Taxonomy of MOACO algorithms based on the algorithmic components listed.

Table 2 lists the MOACO algorithms that will be presented in the forthcoming subsections.

5.1 Ant system (AS) vs. multiobjective ant system (MAS)

The Ant System was the first ACO algorithm developed (Dorigo, 1992; Dorigo & Caro, 1999). The AS algorithm consists in two main phases: the ant's solution construction and the pheromone update.

The pheromone values τ_{ij} associated with arcs, are initially set to a given value τ_0, and the heuristic information $\eta_{ij} = 1/d_{ij}$ is inversely proportional to the distance between city i and j, in the case of the Traveling Salesman Problem.

At each iteration, each of the m ants in the colony constructs its solution according to the following probability of moving from city i to city j:

$$p_{ij}^h = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in \mathcal{N}_i^h} [\tau_{il}]^\alpha [\eta_{il}]^\beta} \quad \text{if } j \in \mathcal{N}_i^h, \text{ and 0 otherwise} \tag{3}$$

where α and β are two parameters that weigh the relative importance of the pheromone trail and the heuristic information, and $\mathcal{N}_i^h$ is the feasible neighbourhood of ant h in city i.

When all m ants have built a solution, the pheromone trail is evaporated according to

$$\tau_{ij} \leftarrow (1 - \rho)\tau_{ij} \tag{4}$$

where $\rho \in (0,1]$ is the pheromone evaporation rate.

After the evaporation process, the ants increment the pheromone trail matrix

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{h=1}^{m} \Delta\tau_{ij}^h \tag{5}$$

with

$$\Delta\tau_{ij}^{h} = \begin{cases} 1/f(s_h), & \text{if arc } (i,j) \text{ belongs to the tour built by the } h\text{-th ant;} \\ 0, & \text{otherwise;} \end{cases} \quad (6)$$

where, $\Delta\tau_{ij}^{h}$ is the amount of pheromone deposit by ant h, on the arcs it has visited, which is proportional to the solution s_h (tour length) built by the ant.
When the termination criteria is achieved, the algorithm returns only one solution, which contains the best tour.
The AS with multiple objectives was initially proposed by Paciello and his colleges (Paciello et al., 2006). The algorithm was tested in three different combinatorial problems, the Quadratic Assignement Problem (QAP), the Traveling Salesman Problem (TSP) and the Vehicle Routing Problem with Time Windows (VRPTW).
The heuristic informations η_{ij}^{0} and η_{ij}^{1} are set for each objective in the same way as in AS, namely, inversely proportional to the cost of adding the arc (i,j) to the solution under construction. Only one pheromone matrix is used.
In the MAS algorithm, the next node j to be visited is selected according to the following probability

$$p_{ij}^{h} = \frac{\tau_{ij}[\eta_{ij}^{0}]^{\lambda\beta}[\eta_{ij}^{1}]^{(1-\lambda)\beta}}{\sum_{l\in\mathcal{N}_i^h}\tau_{il}[\eta_{il}^{0}]^{\lambda\beta}[\eta_{il}^{1}]^{(1-\lambda)\beta}} \quad \text{if } j \in \mathcal{N}_i^h, \text{ and 0 otherwise} \quad (7)$$

To force the ants to search in different regions of the search space, λ is calculated for each ant $h \in \{1,...,m\}$ as $\lambda_h = (h-1)/(m-1)$. Thus, in the extreme cases, the ant m with $\lambda = 1$ considers only the first objective whereas ant 1 with $\lambda = 0$ considers only the second objective.
After the ants construct their solutions, the Pareto set is updated. The non-dominated solutions in the current iteration are added to the Pareto set, and those that are dominated are excluded. The pheromone trail update is performed only by ants that generate non-dominated solutions, i.e, solutions that are in the Pareto set. The rule for pheromone evaporation and deposit is applied to each solution s_p of the current Pareto set, as follows

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\Delta\tau \quad (8)$$

where ρ is the evaporation rate and $\Delta\tau$ is given by

$$\Delta\tau = \frac{1}{\sum_{k=1}^{K} f^{k}(s_p)} \quad (9)$$

A new procedure, named convergence control, is included in the MAS algorithm in order to avoid premature convergence to local optimal and a stagnation behaviour. The procedure consists in reinitializing the pheromone matrix if, for a given number of iterations, no improvement is reached, i.e, when no non-dominated solutions are found.
The MAS algorithm returns a set of non-dominated solutions, which contains the best values found during the run.

5.2 Ant colony system (ACS) vs. multiobjective ant colony system (MACS)

The Ant Colony System (ACS) (Dorigo & Gambardella, 1997a;b) is an extension of the Ant System. The ACS introduces three main changes in the AS algorithm: a local pheromone update is performed each time an ant uses an arc (i,j); a more aggressive action choice rule is used, called *pseudorandom proportional* rule; and the global pheromone update is applied at the end of each iteration by only one ant, which can be either the iteration-best or the best-so-far.

The action choice rule performed by each ant to choose the next node to be visited is applied according to the pseudorandom proportional rule, given by

$$j = \begin{cases} \arg\max_{j \in \mathcal{N}_i^h} \{\tau_{ij}[\eta_{ij}]^\beta\}, & \text{if } q \leq q_0 \\ \hat{j}, & \text{otherwise;} \end{cases} \tag{10}$$

where q is a random variable in $[0,1]$, $q_0 \in [0,1]$ is a parameter chosen by the user, β defines the relative importance of the objectives, and $\hat{j}$ is a random value obtained according to equation (3) (with $\alpha = 1$).

This algorithm implements a local pheromone update, performed every time an ant moves from one node to another, as follows:

$$\tau_{ij} \leftarrow (1-\varphi)\tau_{ij} + \varphi\tau_0 \tag{11}$$

where $\varphi \in (0,1)$ is the pheromone decay coefficient, and $\tau_0 = 1/f(s_{nn})$ is the initial value of the pheromone matrix, with n being the number of cities and $f(s_{nn})$ the length of a nearest-neighbor tour.

The global pheromone update is performed at the end of each iteration only by the best ant (the best-so-far or iteration-best ant), according to the following expression

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}^{best} \tag{12}$$

where, $\Delta\tau_{ij}^{best} = 1/f(s_{best})$ is the amount of pheromone deposit by the ant that generates the best solution.

The multiobjective version of ACS algorithm was developed by Barán and Schaerer (Barán & Schaerer, 2003), to solve a vehicle routing problem with time windows. The MACS algorithm uses a single pheromone matrix τ and two heuristic matrices, η_{ij}^0 and η_{ij}^1. The next node to be visited follows the same transition rule of ACS, but adapted to the multiobjective problem as follows:

$$j = \begin{cases} \arg\max_{j \in \mathcal{N}_i^h} \{\tau_{ij}[\eta_{ij}^0]^{\lambda\beta}[\eta_{ij}^1]^{(1-\lambda)\beta}\}, & \text{if } q \leq q_0 \\ \hat{i}, & \text{otherwise;} \end{cases} \tag{13}$$

where q, q_0 and β are parameters as defined previously, λ is computed for each ant h as $\lambda = h/m$, with m being the total number of ants, and $\hat{i}$ is a city selected according to the following probability:

$$p_{ij}^h = \frac{\tau_{ij}[\eta_{ij}^0]^{\lambda\beta}[\eta_{ij}^1]^{(1-\lambda)\beta}}{\sum_{l \in \mathcal{N}_i^h} \tau_{il}[\eta_{il}^0]^{\lambda\beta}[\eta_{il}^1]^{(1-\lambda)\beta}} \quad \text{if } j \in \mathcal{N}_i^h \text{ and, 0 otherwise} \tag{14}$$

The local pheromone update is given by

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\tau_0 \tag{15}$$

Initially τ_0 is calculated by taking the average cost of solutions in each objective function $f^1(s_h)$ and $f^2(s_h)$, as:

$$\tau_0 = \frac{1}{n.\overline{f^1(s_h)}.\overline{f^2(s_h)}} \tag{16}$$

where n is calculated as an average number of nodes.

After the construction of solutions, each one is compared to the Pareto set. Each non-dominated solution is included in the Pareto set and the dominated ones are excluded.

At the end of each iteration, τ_0 is calculated according to the previous equation, but taking the average cost of the solutions in the Pareto set. If $\tau_0' > \tau_0$, then the pheromone trails are reinitialized to the new value, otherwise, the global pheromone update is performed with each solution s_p of the current Pareto set, as follows:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \frac{\rho}{f^1(s_p)f^2(s_p)} \tag{17}$$

The MACS algorithm returns the set of non-dominated solutions found.

5.3 Ant-Q vs. multiple objective Ant-Q (MOAQ)

The Ant-Q algorithm was based on a distributed reinforcement learning technique and was first applied to the design of irrigation networks. This algorithm was proposed by Gambardella and Dorigo (Gambardella & Dorigo, 1995) before ACS. They differ from each other only in the definition of the term τ_0 (Dorigo & Stützle, 2004), which in Ant-Q is set to

$$\tau_0 = \gamma \max_{j \in \mathcal{N}_i^h} \{\tau_{ij}\} \tag{18}$$

where γ is a parameter and the maximum is taken over the set of pheromone trails on the feasible neighbourhood of ant h in node i.

The multiple objective version of Ant-Q (MOAQ) proposed by Mariano and Morales (Mariano & Morales, 1999), implements a family/colony of agents to perform the optimization of each objective. Each colony is assigned to optimize one objective considering the solutions found for the other objectives. The MOAQ also implements a reward and penalty policy. A reward is given to the non-dominated solutions while the solutions which violate the constraints are penalized.

The MOAQ uses one pheromone trail and two heuristic matrices, one for each objective. One colony, say *colony-1*, optimizes the first objective, where the state transition rule, given by

$$j = \begin{cases} \arg\max_{j \in \mathcal{N}_i^h} \{\tau_{ij} + \eta_{ij}\}, & \text{if } t \geq t_s \\ p_{ij} = \frac{\tau_{ij}+\eta_{ij}}{\sum_{l \in \mathcal{N}_i^h} \tau_{il}+\eta_{il}}, & \text{otherwise;} \end{cases} \tag{19}$$

is related to the heuristic information associated to the first objective. This same process is applied to the second colony with respect to the second objective, where the state transition rule is given as follows

$$j' = \begin{cases} \arg\max_{j \in \mathcal{N}_i^h} \{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta\}, & \text{if } t \geq t_s \\ p'_{ij} = \frac{[\tau_{ij}]^\alpha [\eta_{ij}]^\beta}{\sum_{l \in \mathcal{N}_i^h} [\tau_{il}]^\alpha [\eta_{il}]^\beta}, & \text{otherwise;} \end{cases} \tag{20}$$

where t and t_s are two variables.

The learning Q-values, which can be seen as pheromone information, are calculated using the following update rule:

$$\tau_{ij} \leftarrow (1-\alpha)\tau_{ij} + \alpha[r_{ij} + \gamma \max \tau_{pz}] \tag{21}$$

where α is the learning step, γ is the discount factor, r_{ij} is the reward given to the best solution found in each iteration and $\max \tau_{pz}$ is the maximum pheromone value in the next algorithm step.

Finally, MOAQ returns a set of non-dominated solutions as a final result. When a solution found violates any constraint, the algorithm applies a penalty to its components on the Q values.

5.4 $\mathcal{MAX} - \mathcal{MIN}$ Ant systems ($\mathcal{MM}$AS) vs. multiobjective $\mathcal{MM}$AS (M3AS)

$\mathcal{MAX} - \mathcal{MIN}$ Ant System ($\mathcal{MM}$AS), developed by Stützle and Hoos (Stützle, 1997; Stützle & Hoos, 2000) is considered one of the best performing extension of Ant System. In order to achieve a strong exploitation of the search process the $\mathcal{MM}$AS allows only the best solutions to add pheromone during the pheromone trail update procedure. Also, it imposes bounds on the pheromone trail values in order to avoid premature convergence.
The selection rule is the same as that used in AS, by applying equation (3) to choose the next node to be visited. After all ants construct their solutions, the pheromone matrix is updated by applying the AS evaporation equation (4), followed by the deposit of pheromone given by

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau_{ij}^{best} \tag{22}$$

where, $\Delta\tau_{ij}^{best} = 1/f(s_{best})$ is the amount of pheromone deposit by the ant that generates the best solution (s_{best}), which may be the best-so-far (s_{bs}) or the iteration-best (s_{ib}).
In order to avoid search stagnation, the pheromone trails are limited by lower and upper values τ_{min} and τ_{max} such that $\tau_{min} \leq \tau_{ij} \leq \tau_{max}, \forall i, j$. Additionally, the initial pheromone trails are set to the upper pheromone trail limits, so that the initial search phase promotes a higher exploration of the search space.
The $\mathcal{MMAS}$ occasionally reinitializes the pheromone trail matrix, so as to increase the exploration of edges that have small probability of being chosen, and also to avoid stagnation.
The $\mathcal{MM}$AS version with multiple objectives was proposed by Pinto and Báran (Pinto & Barán, 2005), to solve a Multicast Traffic Engineering problem. The M3AS algorithm uses one pheromone matrix and as many heuristic matrices as the number of objectives $\eta_{ij}^k = 1/d_{ij}^k$, with K being the number of objectives.
The probability of assigning the arc (i, j) is given by

$$p_{ij}^h = \frac{[\tau_{ij}]^\alpha \prod_{k=1}^K [\eta_{ij}^k]^{\lambda^k}}{\sum_{l \in \mathcal{N}_i^h} [\tau_{il}]^\alpha \prod_{k=1}^K [\eta_{il}^k]^{\lambda^k}} \quad \text{if } j \in \mathcal{N}_i^h, \text{ and 0 otherwise} \tag{23}$$

where the parameters λ^k determine the relative influence of the heuristics information.
When all ants construct their solutions the pheromone evaporation is applied using equation (4) followed by the deposit of pheromone as follows

$$\tau_{ij}^k \leftarrow \tau_{ij}^k + \Delta\tau_{ij}^k \tag{24}$$

where $\Delta\tau_{ij}^k = 1/\sum_{k=1}^K f^k(s_p)$. The ants which are allowed to update the pheromone matrix are the ones that generated non-dominated solutions.
In the end of the algorithm execution, M3AS returns the set of non-dominated solutions found.

5.5 Omicron ACO (OA) vs. multiobjective omicron ACO (MOA)

The Omicron ACO (OA) algorithm proposed by Gómez and Barán (Gómez & Barán, 2005) is inspired by $\mathcal{MM}$AS. In OA, a constant pheromone matrix τ^0 is defined, with $\tau_{ij}^0 = 1$, $\forall i, j$.
OA is a population based algorithm where a population of individuals is maintained which contains the best solutions found so far.
The first population of individuals is initialized using τ^0. Each ant of the colony constructs a complete tour using (3). At the end of the iteration, every time a new individual generates a

solution better than the worst individual and different from the others in the population, this new one replaces the worst one.
After the maximum number of iterations is reached, the pheromone matrix is updated using the solution in the population set, as follows

$$\tau_{ij} \leftarrow \tau_{ij} + \frac{O}{m} \tag{25}$$

where O (from *Omicron*) and m (number of individuals) are given values. The algorithm returns the best solution found.
The multiobjective version of the Omicron ACO (MOA) algorithm was proposed by Gardel and colleagues (Gardel et al., 2005) under the name of *Eletric Omicron*. The MOA was first applied to the Reactive Power Compensation Problem in a multiobjective context. In MOA the pheromone trails are initially set as in OA (that is, $\tau_{ij}^0 = 1$) and the two heuristic matrices η_{ij}^0 and η_{ij}^1 associated with each objective are linearly combined to define the visibility

$$\eta_{ij} = \omega_1 \eta_{ij}^0 + \omega_2 \eta_{ij}^1 \tag{26}$$

where ω_1 and ω_2 are weight values (with $\omega_1 + \omega_2 = 1$) which are are dynamically changed at each iteration of the algorithm.
The artificial ants construct the solutions by using equation (3), with η_{ij} defined by equation (26). An external set is used to save the non-dominated solutions. At the end of the iteration, the pheromone trails are updated, using equation (25) only in the solutions contained in the Pareto set.

5.6 Ant algorithm for bi-criterion optimization (BicriterionAnt)

Iredi and co-workers (Iredi et al., 2001) proposed two ACO methods to solve the Single Machine Total Tardiness Problem (SMTTP) with changeover costs,considering two objectives. The difference between the algorithms is the use of one or several ant colonies. In this section we describe the so-called BicriterionAnt algorithm, which uses only one colony of ants. It uses two pheromone trail matrices τ and τ' and two heuristic information matrices η and η', one for each objective.
In every iteration, each of the m ants generates a solution to the problem using the following probability to select the next job j:

$$p_{ij}^h = \frac{\tau_{ij}^{\lambda\alpha} \tau'^{(1-\lambda)\alpha}_{ij} \eta_{ij}^{\lambda\beta} \eta'^{(1-\lambda)\beta}_{ij}}{\sum_{l \in \mathcal{N}_i^h} \tau_{il}^{\lambda\alpha} \tau'^{(1-\lambda)\alpha}_{il} \eta_{il}^{\lambda\beta} \eta'^{(1-\lambda)\beta}_{il}} \quad \text{if } j \in \mathcal{N}_i^h, \text{ and 0 otherwise} \tag{27}$$

where α and β are parameters that weight the relative importance of the pheromone trail and the heuristic information, η and η' are heuristic values associated with edge a_{ij} according to each objective, and $\mathcal{N}_i^h$ is the current feasible neighbourhood of ant h. In order to make the ants search in different regions of the Pareto front, λ is calculated for each ant $h \in \{1, ..., m\}$, as:

$$\lambda_h = \frac{(h-1)}{(m-1)}$$

Thus, in the extreme cases, ant m, with $\lambda = 1$, considers only the first objective whereas ant 1, with $\lambda = 0$, considers only the second objective.

Once all ants generate their solutions, the pheromone trails are evaporated according to

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} \ , \ \tau'_{ij} \leftarrow (1-\rho)\tau'_{ij}$$

where $\rho \in (0,1]$ is the evaporation rate.
The pheromone update process is performed only by the ants in the current non-dominated set. Each ant updates both matrices as follows:

$$\tau_{ij} \leftarrow \tau_{ij} + 1/l \ , \ \tau'_{ij} \leftarrow \tau'_{ij} + 1/l$$

where l is the number of ants that are allowed to do the updating in the current iteration.

5.7 Multiobjective network ACO (MONACO)

The MONACO algorithm was proposed in (Cardoso et al., 2003), to solve a dynamic problem, the optimization of the message traffic in a network. Hence, in the following, we present an adaptation of the original algorithm to solve static problems, where the policy of the network does not change during the algorithm's steps, as done in (García-Martínez et al., 2007).
This algorithm uses one heuristic information $\eta_{ij} = \sum_{k=1}^{K} d_{ij}^k$ and several pheromone matrices τ^k, one for each objective, where K is the number of objectives. Each ant, considered as a message, chooses the next node on the web according to the following probability:

$$p_{ij}^h = \frac{\eta_{ij}^{\beta} \prod_{k=1}^{K} [\tau_{ij}^k]^{\alpha_k}}{\sum_{l \in \mathcal{N}_i^h} \eta_{il}^{\beta} \prod_{k=1}^{K} [\tau_{il}^k]^{\alpha_k}} \tag{28}$$

if $j \in \mathcal{N}_i^h$, and 0 otherwise.
The α_k and β parameters weigh the relative importance of the pheromone matrices and the heuristic information, respectively. By the end of each iteration, the ants that built a solution update the pheromone trail matrices in the following way:

$$\tau_{ij}^k = (1-\rho_k)\tau_{ij}^k + \Delta\tau_{ij}^k \tag{29}$$

where

$$\Delta\tau_{ij}^k = \frac{Q}{f^k(s_h)} \tag{30}$$

with ρ_k being the pheromone evaporation rate for each objective k, Q is a constant related to the amount of pheromone laid by the ants, and s_h is the solution built by the ant h. In this adaptation of MONACO, the non-dominated solutions, that form the Pareto set, are stored in an external archive.

5.8 COMPETants

Doerner, Hartl and Reimann (Doerner et al., 2001) developed the COMPETants to solve a multiobjective transportation problem.
The basic idea behind COMPETants is the use of two populations with different priority rules. In COMPETants the population size is not fixed but undergoes adaptation during the algorithm execution. The ants utilize their own pheromone and heuristic information. However, some ants of each population which generate the best solutions, called *spies*, use not only their own information (pheromone and heuristic) but also the foreign information.

Decision rule for the ants is given by the following equation

$$p_{ij}^{h} = \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in \mathcal{N}_i^h}[\tau_{il}]^{\alpha}[\eta_{il}]^{\beta}}, \quad \text{if } j \in \mathcal{N}_i^h, \text{ and 0 otherwise} \tag{31}$$

noticing that each colony targets its own pheromone and heuristic information. For the spy ants, the decision rule is given by

$$p_{ij}^{h} = \frac{[0.5\tau_{ij}^{0} + 0.5\tau_{ij}^{1}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in \mathcal{N}_i^h}[0.5\tau_{il}^{0} + 0.5\tau_{il}^{1}]^{\alpha}[\eta_{il}]^{\beta}} \quad \text{if } j \in \mathcal{N}_i^h, \text{ and 0 otherwise} \tag{32}$$

where, in this new rule, the spies combine the information of both pheromone trails.
When all ants have constructed their solutions, the pheromone update rule is performed only by a number of best ants, ranked according to the solution quality. The update rule is as follows:

$$\tau_{ij} = \rho\tau_{ij} + \sum_{\lambda=1}^{\Lambda} \Delta\tau_{ij}^{\lambda} \tag{33}$$

where Λ represents the best ants of the population, and the constant $\Delta\tau_{ij}^{\lambda}$ is represented as

$$\Delta\tau_{ij}^{\lambda} = \begin{cases} 1 - \frac{\lambda-1}{\Lambda}, & \lambda \in [1, \Lambda], \quad \text{if } j \in \mathcal{N}_i^h \\ 0, & \text{otherwise;} \end{cases} \tag{34}$$

The algorithm returns the best solution found.

5.9 SACO

T'kindt *et al* (Vincent et al., 2002) proposed the SACO algorithm to solve a 2-machine bicriteria flowshop scheduling problem.
The SACO algorithm uses only one pheromone information and one heuristic information. This algorithm was developed to solve a lexicographical problem, where only one best solution is returned at the end of the algorithm execution.
Each ant constructs a feasible solution using the pheromone information, which can be done in two different modes: (i) by an intensification mode, where an ant chooses, as the most suitable job, the one with the highest value of τ_{ij}; or (ii) by a diversification mode, where an ant uses a random-proportional rule to select the most suitable job. A parameter p_0 was created for selecting the probability of being in one of these two modes, which is given by $p_0 = log(n)/log(N)$, where n is the iteration number, with $n \in [1, N]$. When an ant has built a complete solution, a local search procedure is applied.
The pheromone evaporation is performed on every edge and the pheromone update is done only by the best solution found at each iteration, as follows:

$$\tau_{ij} \leftarrow \begin{cases} \tau_{ij} + \frac{1}{f(s)}, & \text{if arc } (i,j) \in s_{best} \\ (1-\rho)\tau_{ij}, & \text{otherwise;} \end{cases} \tag{35}$$

where s_{best} is the best objective function value found and ρ is the evaporation rate.

5.10 Pareto Ant Colony Optimization (P-ACO)

Doerner *et al* in (Doerner et al., 2004) proposed a Pareto ant colony optimization algorithm to solve the multiobjective portfolio selection problem. It is based on ACS, but the global pheromone update is performed by two specific ants, the best and the second-best ant.
P-ACO uses as many pheromone matrices as the number of objectives k, and only one heuristic information. The decision transition rule is based on ACS with k pheromone matrices

$$j = \begin{cases} \arg\max_{j \in \mathcal{N}_i^h} \{\sum_{k=1}^{K} [p_k \tau_{ij}^k]^\alpha \eta_{ij}^\beta\}, & \text{if } q \leq q_0 \\ \hat{i}, & \text{otherwise;} \end{cases} \tag{36}$$

where p_k are determined randomly for each ant and $\hat{i}$ is a node selected according to:

$$p_{ij}^h = \frac{\sum_{k=1}^{K} [p_k \tau_{ij}^k]^\alpha \eta_{ij}^\beta}{\sum_{l \in \mathcal{N}_i^h} (\sum_{k=1}^{K} [p_k \tau_{il}^k]^\alpha \eta_{il}^\beta)} \quad \text{if } j \in \mathcal{N}_i^h, \text{ and 0 otherwise} \tag{37}$$

A local pheromone update is performed every time an ant traverses an edge (i, j), by applying the following equation, considering each pheromone matrix:

$$\tau_{ij}^k = (1 - \rho)\tau_{ij}^k + \rho\tau_0 \tag{38}$$

where τ_0 is the initial pheromone value and ρ is the evaporation rate.
The global pheromone update is done only by the best and the second-best ants. The update rule for each objective k is given by:

$$\tau_{ij}^k = (1 - \rho)\tau_{ij}^k + \rho\Delta\tau_{ij}^k \tag{39}$$

with $\Delta\tau_{ij}^k$ being an increasing quantity related to the best and second-best solutions according to objective k, which is represented as

$$\Delta\tau_{ij}^k = \begin{cases} 10 & \text{if arc } (i,j) \in s_{best} \\ 5 & \text{if arc } (i,j) \in s_{second-best} \\ 0 & \text{otherwise;} \end{cases} \tag{40}$$

During the algorithm run, the non-dominated solutions are stored in a external set and are returned as a final solution.

5.11 Multiobjective ant colony system algorithm (MOACSA)

Yagmahan and Yenisey proposed a multi-objective ant colony system procedure, based on ACS, for a flow shop scheduling problem (Yagmahan & Yenisey, 2010).
The MOACSA uses one pheromone matrix and one heuristic information. At the initialization step, the initial pheromone value is calculated by

$$\tau_0 = \frac{1}{n[f^0(s_h) + f^1(s_h)]} \tag{41}$$

where n is the number of jobs and s_h is the solution built by ant h, which is calculated considering each objective.
In the construction process the ant h in job i selects the job j by applying equation (10) (with $\alpha \geq 1$) as a state transition rule. Each time an ant selects a job, it applies a local pheromone update

$$\tau_{ij} = (1 - \xi)\tau_{ij} + \xi\tau_0 \tag{42}$$

where $\xi \in (0,1)$ is the local pheromone evaporation parameter.
The global update rule is performed only by ants that generates the best solutions, as follows

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \rho\Delta\tau_{ij} \tag{43}$$

where $\Delta\tau_{ij}$ is the amount of pheromone deposit by ant h which generate the best solution in the current iteration.

6. Conclusions

This chapter reviewed the application of ant colony optimization algorithms to multiple objective problems. The extension of well known single objective ACO algorithms to tackle MOO problems has been presented. In addition, the algorithmic components that play a relevant role in MOACO algorithm design and performance have been discussed. Several algorithms from the literature which are representative of the many ways such components can be implemented have been presented. It is expected that this chapter provides the readers with a comprehensive view of the use of ACO algorithms to solve MOO problems and helps them in designing new ACO techniques for their particular applications.

7. Acknowledgments

The authors thank the support from the brazilian agencies CNPq (308317/2009-2 and 141519/2010-0) and FAPERJ (E-26/ 102.825/2008).

8. References

Angus, D. & Woodward, C. (2009). Multiple objective ant colony optimisation, *Swarm Intelligence* 3(1): 69–85.

Barán, B. & Schaerer, M. (2003). A multiobjective ant colony system for vehicle routing problem with time windows, *Proceedings of the 21st IASTED International Conference*, pp. 97–102.

Branke, J., Deb, K., Miettinen, K. & Slowiński, R. (eds) (2008). *Multiobjective Optimization: Interactive and Evolutionary Approaches (Lecture Notes in Computer Science)*, Springer.

Cardoso, P., Jesus, M. & Márquez, A. (2003). Monaco - multi-objective network optimisation based on an ACO, *In Proceedings of Encuentros de Geometria Computacional*.

Coello, C. A. C., Veldhuizen, D. A. V. & Lamont, G. B. (2002). *Evolutionary algorithms for solving Multi-Objective Problems*, Kluwer.

Doerner, K. F., Hartl, R. F. & Reimann, M. (2001). Are COMPETants more competent for problem solving? - the case of a routing and scheduling problem, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, Morgan Kaufmann, San Francisco, California, USA, p. 802.

Doerner, K., Gutjahr, W. J., Hartl, R. F., Strauss, C. & Stummer, C. (2004). Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection, *Kluwer Academic Publishers, Annals of Operations Research* (131): 79–99.

Dorigo, M. (1992). *Optimization, Learning and Natural Algorithms*, PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy.

Dorigo, M., Bonabeau, E. & Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems*, Oxford University Press, Inc., New York, NY.

Dorigo, M., Bonabeau, E. & Theraulaz, G. (2000). Ant algorithms and stigmergy, *Future Gener. Comput. Syst.* 16(9): 851–871.

Dorigo, M. & Caro, G. D. (1999). Ant Colony Optimization: A new meta-heuristic, *in* P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao & A. Zalzala (eds), *IEEE Congress on Evolutionary Computation – CEC'99*, IEEE Press, Piscataway, NJ, pp. 1470–1477.

Dorigo, M. & Gambardella, L. M. (1997a). Ant colonies for the traveling salesman problem, *BioSystems* 43: 73–81.

Dorigo, M. & Gambardella, L. M. (1997b). Ant Colony System: A cooperative learning approach to the traveling salesman problem, *IEEE Transactions on Evolutionary Computation* 1(1): 53–66.

Dorigo, M. & Stützle, T. (2004). *Ant Colony Optimization*, The MIT Press, Cambridge, Massachusetts.

Dorigo, M., Stützle, T. & Birattari, M. (2006). Ant colony optimization - artificial ants as a computational intelligence technique, *IEEE Computational Intelligence Magazine* .

Ehrgott, M. (2005). *Multicriteria Optimization*, Lecture Notes in Economics and Mathematical Systems, Springer-Verlag.

Fonseca, L. G., Capriles, P. V. S. Z., Barbosa, H. J. C. & Lemonge, A. C. C. (2007). A stochastic rank-based ant system for discrete structural optimization, *Proceedings of the 2007 IEEE Swarm Intelligence Symposium (SIS 2007)*, pp. 68–75.

Fonseca, V. G., Fonseca, C. M. & Hall, A. O. (2001). Inferential performance assessment of stochastic optimisers and the attainment function, *EMO '01: Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, Springer-Verlag, London, UK, pp. 213–225.

Gambardella, L. M. & Dorigo, M. (1995). Ant-q: A reinforcement learning approach to

the traveling salesman problem, *Twelfth International Conference on Machine Learning*, Morgan Kaufmann, pp. 252–260.

Gandibleux, X., Sevaux, M., Sörensen, K. & T'Kindt, V. (eds) (2004). *Metaheuristics for Multiobjective Optimisation*, Vol. 535, springer-verlag edn, Lecture Notes in Economics and Mathematical Systems.

García-Martínez, C., Cordón, O. & Herrera, F. (2007). A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP, *European Journal of Operational Research* 180: 116–148.

Gardel, P., Estigarribia, H., Fernández, U. & Barán, B. (2005). Aplicación del Ómicron aco al problema de compensación de potencia reactiva en un contexto multiobjetivo, *Congreso Argentino de Ciencias de la Computación - CACIC2005*, Concordia Argentina.

Gómez, O. & Barán, B. (2005). Omicron aco. a new ant colony optimization algorithm, *CLEI Electronic Journal* 8(1).

Iredi, S., Merkle, D. & Middendorf, M. (2001). Bi-criterion optimization with multi colony ant algorithms, *Lecture Notes in Computer Science (LNCS)* 1993: 359–372.

Knowles, J. (2005). A summary-attainment-surface plotting method for visualizing the performance of stochastic multiobjective optimizers, *ISDA '05: Proceedings of the 5th International Conference on Intelligent Systems Design and Applications*, IEEE Computer Society, Washington, DC, USA, pp. 552–557.

Knowles, J. & Corne, D. (2002). On metrics for comparing non-dominated sets. In Congress on Evolutionary Computation (CEC 2002).

Knowles, J., Thiele, L. & Zitzler, E. (2006). A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers, *TIK Report 214*, Computer Engineering and Networks Laboratory (TIK), ETH Zurich.

Li, X., Branke, J. & Kirley, M. (2007). On performance metrics and particle swarm methods for dynamic multiobjective optimization problems, *IEEE Congress on Evolutionary Computation*, Vol. 25-28, pp. 576–583.

López-Ibáñez, M. (2004). *Multi-objective ant colony optimization*, Master's thesis, Darmstadt University of Technology.

López-Ibáñez, M., Paquete, L. & Stützle, T. (2009). Exploratory analysis of stochastic local search algorithms in biobjective optimization, pp. 209–233.

López-Ibáñez, M. & Stützle, T. (2010). Automatic configuration of multi-objective ant colony optimization algorithms, pp. 95–106.

Mariano, C. E. & Morales, E. (1999). MOAQ an ant-Q algorithm for multiple objective optimization problems, *GECCO-99*, Vol. 1, Morgan Kaufmann, Orlando, Florida, USA, pp. 894–901.

Miettinen, K. (1999). *Nonlinear Multiobjective Optimization*, Kluwer Academic, Norwell, Massachusetts.

Paciello, J., Martínez, H., Lezcano, C. & Barán, B. (2006). Algoritmos de optimizacin multi-objetivos basados en colonias de hormigas, *XXXII Latin-American Conference on Informatics 2006 - CLEI2006*.

Pinto, D. & Barán, B. (2005). Solving multiobjective multicast routing problem with a new ant colony optimization approach, *LANC '05: Proceedings of the 3rd international IFIP/ACM Latin American conference on Networking*, ACM, New York, NY, USA, pp. 11–19.

Silberholz, J. & Golden, B. (2009). *Handbook of Metaheuristics*, Springer-Verlag, chapter Comparison of Metaheuristics.

Stützle, T. (1997). $\mathcal{MAX} - \mathcal{MIN}$ ant system for quadratic assignment problems, *Technical Report AIDA-97-04*, FG Intellektik, FB Informatik, TU Darmstadt, Germany.

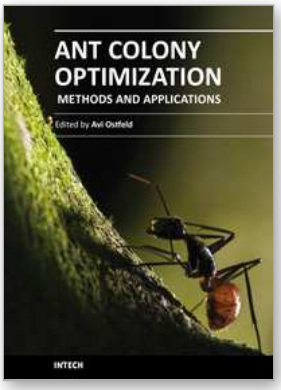
Stützle, T. & Hoos, H. H. (2000). $\mathcal{MAX} - \mathcal{MIN}$ ant system, *Future Generation Computer Systems Journal* 16(8): 889–914.

Tan, K., Khor, E. & Lee, T. (2005). *Multiobjective Evolutionary Algorithms and Applications*, Springer.

Vincent, T., Monmarche, N., Tercinet, F. & Laugt, D. (2002). An ant colony optimization algorithm to solve a 2-machine bicriteria flowshop scheduling problem, *European Journal of Operational Research* 142(2): 250–257.

Yagmahan, B. & Yenisey, M. M. (2010). A multi-objective ant colony system algorithm for flow shop scheduling problem, *Expert Syst. Appl.* 37(2): 1361–1368.

Zitzler, E., Deb, K. & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results, *Evolutionary Computation* 8(2): 173–195.



Ant Colony Optimization - Methods and Applications

Edited by Avi Ostfeld

ISBN 978-953-307-157-2
Hard cover, 342 pages
Publisher InTech
Published online 04, February, 2011
Published in print edition February, 2011

Ants communicate information by leaving pheromone tracks. A moving ant leaves, in varying quantities, some pheromone on the ground to mark its way. While an isolated ant moves essentially at random, an ant encountering a previously laid trail is able to detect it and decide with high probability to follow it, thus reinforcing the track with its own pheromone. The collective behavior that emerges is thus a positive feedback: where the more the ants following a track, the more attractive that track becomes for being followed; thus the probability with which an ant chooses a path increases with the number of ants that previously chose the same path. This elementary ant's behavior inspired the development of ant colony optimization by Marco Dorigo in 1992, constructing a meta-heuristic stochastic combinatorial computational methodology belonging to a family of related meta-heuristic methods such as simulated annealing, Tabu search and genetic algorithms. This book covers in twenty chapters state of the art methods and applications of utilizing ant colony optimization algorithms. New methods and theory such as multi colony ant algorithm based upon a new pheromone arithmetic crossover and a repulsive operator, new findings on ant colony convergence, and a diversity of engineering and science applications from transportation, water resources, electrical and computer science disciplines are presented.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Jaqueline S. Angelo and Helio J.C. Barbosa (2011). Ant Colony Algorithms for Multiobjective Optimization, Ant Colony Optimization - Methods and Applications, Avi Ostfeld (Ed.), ISBN: 978-953-307-157-2, InTech, Available from: http://www.intechopen.com/books/ant-colony-optimization-methods-and-applications/ant-colony-algorithms-for-multiobjective-optimization

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821