Chapter 1

Multi-Objective Ant Colony Optimization: A Taxonomy and Review of Approaches

Guillermo Leguizamón* and Carlos A. Coello Coello†

Ant Colony Optimization (ACO) is one of the most representative metaheuristics derived from the broad concept known as Swarm Intelligence (SI) where the behavior of social insects is the main source of inspiration. Being a particular SI approach, the ACO metaheuristic is mainly characterized by its distributiveness, flexibility, capacity of interaction among simple agents, and its robustness. The ACO metaheuristic has been successfully applied to an important number of discrete and continuous single-objective optimization problems. However, this metaheuristic has shown a great potential to also cope with multi-objective optimization problems as evidenced by the several proposals currently available in that regard. This chapter is aimed at describing the most relevant and recent developments on the use of the ACO metaheuristic for solving multi-objective optimization problems. Additionally, we also derive a refined taxonomy of the types of ACO variants that have been used for multi-objective optimization. Such a taxonomy intends to serve as a quick guide for those interested in using an ACO variant for solving multi-objective optimization problems. In the last part of the chapter, we provide some potential paths for further research in this area.

*Guillermo Leguizamón is with LIDIC - Universidad Nacional de San Luis, San Luis, Argentina and with the UMI LAFMIA 3175 CNRS at CINVESTAV-IPN, Departamento de Computación, Av. IPN No. 2508. Col. San Pedro Zacatenco, México D.F. 07300, MÉXICO, email: legui@unsl.edu.ar

†Carlos A. Coello Coello is with CINVESTAV-IPN (Evolutionary Computation Group), Departamento de Computación. Av. IPN No. 2508. Col. San Pedro Zacatenco, México D.F. 07300, MÉXICO, email: ccoello@cs.cinvestav.mx. He is also affiliated at the UMI LAFMIA 3175 CNRS at CINVESTAV-IPN.

1.1. Introduction

The field of Computational Intelligence (CI), and particularly the algorithms based on the concept of Swarm Intelligence (SI) has been intensively studied and successfully applied to optimization problems. Among these problems are those that include multiple objectives, which usually are very common in many application areas. SI-based algorithms involve several characteristics that make them particularly suitable for solving multi-objective optimization problems (MOOPs), *e.g.*, inherently decentralized, the members of the swarm can be in charge of different objectives, different levels and types of interactions can be defined in order to share individual search experience with the rest of the swarm, etc. The most representative and developed SI algorithms include Particle Swarm Optimization (PSO) (see [1] for more details) and the Ant Colony Optimization (ACO) metaheuristic [2, 3].

In the case of ACO algorithms, an important number of proposals have shown, with different levels of success, the applicability of these algorithms to multi-objective optimization problems. As an example, two interesting reviews on this topic can be found in García-Martinez *et al.* [4], and in Angus and Woodward [5]. Similarly, a section in Coello Coello *et al.*'s book is devoted to the ACO metaheuristic [6] as an example of an alternative metaheuristic[a] for solving MOOPs.

The remainder of this chapter is organized as follows. In the next section, we present a general overview of the ACO metaheuristic for discrete and continuous problems. Section 1.3 gives a general introduction to multi-objective optimization in which the most relevant concepts are described. In Section 1.4, we present an up-to-date review of the ACO metaheuristic for multi-objective optimization problems in which the specific components of the proposals dealing with the multi-objective aspects are highlighted. A refined taxonomy of multi-objective ACO approaches is presented in Section 1.5. Some of the promising research areas within this topic are briefly described in Section 1.6. Finally, the conclusions of this chapter are provided in Section 1.7.

1.2. Ant Colony Optimization

The Ant Colony Optimization (ACO) metaheuristic [3] embodies a broad class of algorithms whose design is mainly based on the foraging behavior of

[a]By "alternative" the author means, with respect to evolutionary algorithms.

real ants. ACO algorithms were originally designed and have a long tradition in solving a specific type of combinatorial optimization problems (*i.e.*, problems for which the solution construction process can be implemented by simulating a walk through a construction graph). The seminal works of the use the ACO metaheuristic were devoted to the Traveling Salesperson Problem (TSP), a classical NP-complete problem whose main characteristics can be easily exploited to show the applicability of this metaheuristic. For example, in Dorigo [7] there is a description of the first ACO algorithm designed to solve the TSP, the so-called Ant System (AS). After that, several improvements to the AS for solving TSP were proposed: *elitist*-AS, an AS with an *elitist strategy* for updating the pheromone trail levels, AS_{rank} (a rank-based version of Ant System), $\mathcal{M}$ax-$\mathcal{M}$in Ant System ($\mathcal{MM}$AS), and the Ant Colony System (ACS) [3].

In the following, we present the main aspects to be considered when applying the ACO metaheuristic to a particular discrete problem. First, it is important to define an appropriate problem representation, *i.e.*, the construction graph and the way this represents the different problem components and connections among them as well as the definition (if any) of the problem information to be exploited. Second, the behavior of the artificial ants should be defined in order to show how each ant will walk through the construction graph to build the corresponding solutions.

Algorithm 1.1 Outline of the ACO metaheuristic

```
1: Initialize();
2: while termination-condition is NOT TRUE do
3:    BuildSolutions();
4:    PheromoneUpdate();
5:    DaemonActions(); // Optional
6: end while
```

A general outline of the ACO metaheuristic is displayed in Algorithm 1.1 in which four main activities are considered. The way in which those activities are implemented defines the possible algorithms that can be obtained, *i.e.*, AS, elitist-AS, AS-rank, $\mathcal{MM}$AS, ACS, or any other. The main activities could vary from one algorithm to another, however, they can be described in a general way as follows:

- **Initialize():** As in any typical metaheuristic algorithm, some basic tasks need to be done before starting the exploration of the

search space. In this case, the initialization of pheromone trail structure, the heuristic values (when available and used), and any other structure necessary to complete the problem representation.

- **BuildSolutions():** This activity involves the release of an independent colony of artificial ants in charge of incrementally building a solution to the problem. Each ant, at each step of the construction process, makes a local stochastic decision about the next component to be included in the solution under construction.
- **PheromoneUpdate():** The acquired experience achieved by the colony at each iteration is considered in this activity. High quality solutions will positively affect the amount of pheromone trail, *i.e.*, those edges that are part of solutions found will receive an increased amount of pheromone trail according to the goodness of these solutions. This is known as the *global pheromone update*. As in nature, a process of pheromone evaporation takes place (usual implementations of this metaheuristic decrease the amount of pheromone trail for all edges in the construction graph). Thus, the amount of pheromone corresponding to those edges that are not part of any solution at the current iteration will show a gradually diminishing pheromone intensity. It should be noticed that some ACO algorithms, such as ACS, apply a local pheromone update rule which does not depend on the solution quality. Instead, a fixed amount is deposited as soon as an edge in the construction graph is selected to make the move (the next component added to the solution under construction).
- **DaemonActions():** As single ants can not carry out some centralized actions, many ACO algorithms include some specific activities called *daemon actions*. Examples of these activities are: activation of a local search procedure or a collection of global information (*e.g.*, use of a set of the best ranked solutions) that could be used to reinforce some entrances in the pheromone trail structure.

Let us assume that the problem under consideration is the TSP. Figure 1.1 shows a possible construction graph for an instance of size $n = 5$.

Each vertex represents the problem components (*i.e.*, the cities) and each edge represents the connections (routes) between the cities where the distance ($d_{ij} = d_{ji}$) or other value (*e.g.*, cost) is associated to each edge. From the perspective of the ACO metaheuristic, two additional values are

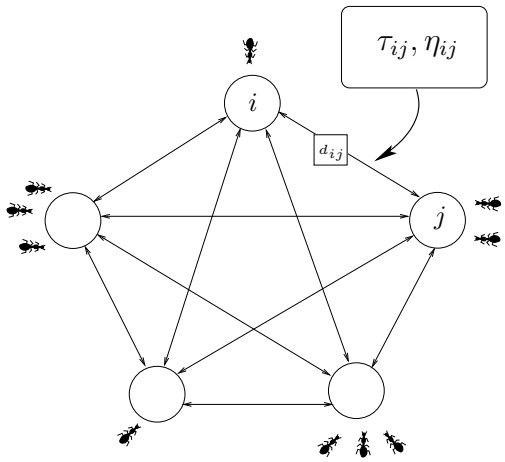

Fig. 1.1. Construction graph for an instance of the symmetric TSP of size $n = 5$

associated to each connection, the desirability of choosing edge (i, j) represented by τ_{ij}, and the problem's information, a heuristic value represented by η_{ij}. In the case of the TSP, the usual value is $\eta_{ij} = 1/d_{ij}$. These two values are then used in the construction solution process to estimate the probability p_{ij} of choosing city j from city i by ant k:

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha \eta_{ij}^\beta}{\sum_{h \in \mathcal{N}^k(i)} \tau_{ih}^\alpha \eta_{ih}^\beta} & \text{if } j \in \mathcal{N}^k(i) \\ 0 & \text{otherwise,} \end{cases} \tag{1.1}$$

where α and β are the parameters that respectively represent the importance of the pheromone trail and the heuristic information, and $\mathcal{N}^k(i)$ represents the set of cities that can be visited by ant k, *i.e.*, the feasible cities. On the side of the pheromone update process, the following equation (usually called global updating) can be applied:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \Delta\tau_{ij}, \tag{1.2}$$

where ρ is the evaporation rate and $\Delta\tau_{ij}$ represents the amount of pheromone trail deposited in edge (i, j) according to the quality of the solutions found by the whole colony that include that edge as part of the solution found. It should be noticed that equations (1.1) and (1.2) represent the basic approach followed by the early ACO algorithms, like the AS.

However, alternative ways of choosing the next problem components or for updating the pheromone trail can be used, like in the ACS and the rest of the family of the ACO algorithms.

From another point view, in the following we describe the activities previously introduced in this section for the TSP. n is the size of the instance, a is the number of ants in the colony, t the current iteration, t_{max} the maximum number of iterations (the two last ones are generally used in the "termination condition" of Algorithm 1.1). $P^{a\times n}$ is the space of all possible sets of cardinality a conformed by integer permutations of size n, and $S(t) \subset P^{a\times n}$, is the set of solutions found at time t.

- The pheromone structure is initialized with a constant value τ_0 which can be defined in different ways whereas the heuristic values are usually set as the inverse of the distance between the cities.

$$\tau(0) = \tau_0 \times \mathbf{1}^{n\times n} \text{ and } \eta_{ij} = 1/d_{ij} \text{ for } i,j \in \{1,\ldots,n\}.$$

- The solution construction process, for each ant, takes into account the current amount of the pheromone trails and the heuristic information to obtain the new sample of solutions. The whole process can be expressed as:

$$S(t) = \mathcal{BS}(\tau(t), \eta) \text{ for } t \in \{0,\ldots,t_{max}\},$$

  where $\mathcal{BS} : \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n} \rightarrow P^{a\times n}$ manages the a ants to incrementally build the solutions based on equation (1.1).
- After the whole colony has built a solutions (this applies for global updating) the new pheromone trail values are calculated as follows:

$$\tau(t+1) = \mathcal{PU}(\tau(t), S(t)), \text{ for } t \in \{0,\ldots,t_{max}-1\}$$

  where $\mathcal{PU} : \mathbb{R}^{n\times n} \times P^{a\times n} \rightarrow \mathbb{R}^{n\times n}$ performs the pheromone updating process based on equation (1.2).

For continuous problems, several proposals have been considered and studied from the perspective of the ACO metaheuristic. The first ACO extension designed to operate on continuous search spaces was introduced by Bilchev et al. [8]. Since then, several other proposals have been introduced (see [9–16]). Particularly, one relevant proposal is that introduced by Socha [17]) and further extended by Socha & Dorigo [18]. Although the different proposed versions of the ACO metaheuristic for continuous problems have been applied with different levels of success, to the best of

authors' knowledge there is no unifying approach for continuous problems as in the case of discrete problems. This has motivated more research in this area, aiming to have a more standard and widely recognized platform for the ACO metaheuristic in continuous domains. Indeed, it is expected that in the near future, more powerful and competitive versions of continuous ACO algorithms will be available as has happened in the case of other metaheuristics such as Evolutionary Algorithms and Particle Swarm Optimization.

To conclude this section, it is important to notice that ACO algorithms have been traditionally applied to single-objective problems. However, many researchers have reported encouraging results regarding the application of such algorithms to multi-objective problems. The next section is devoted to present some basic concepts about multi-objective optimization. After that, we describe the most relevant and recent advances in the design and application of MOACO algorithms.

1.3. Basic Concepts of multi-objective optimization

A multi-objective optimization problem (MOOP) can be formulated as[b]:

$$\text{minimize } \vec{f}(\vec{x}) := [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})] \tag{1.3}$$

subject to:

$$g_r(\vec{x}) \leq 0 \quad r = 1, 2, \ldots, m \tag{1.4}$$

$$h_j(\vec{x}) = 0 \quad j = 1, 2, \ldots, p \tag{1.5}$$

where $\vec{x} = [x_1, x_2, \ldots, x_n]^T$ is the vector of decision variables, $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$, $i = 1, ..., k$ are the objective functions and $g_r, h_j : \mathbb{R}^n \rightarrow \mathbb{R}$, $r = 1, ..., m$, $j = 1, ..., p$ are the constraint functions of the problem.

To describe the concept of optimality in which we are interested, we will introduce next a few definitions.

Definition 1. Given two vectors $\vec{x}, \vec{y} \in \mathbb{R}^k$, we say that $\vec{x} \leq \vec{y}$ if $x_i \leq y_i$ for $i = 1, ..., k$, and that $\vec{x}$ **dominates** $\vec{y}$ (denoted by $\vec{x} \prec \vec{y}$) if $\vec{x} \leq \vec{y}$ and $\vec{x} \neq \vec{y}$.

Definition 2. We say that a vector of decision variables $\vec{x} \in \mathcal{X} \subset \mathbb{R}^n$ is **non-dominated** with respect to $\mathcal{X}$, if there does not exist another $\vec{x}' \in \mathcal{X}$

[b]Without loss of generality, we will assume only minimization problems.

such that $\vec{f}(\vec{x}') \prec \vec{f}(\vec{x})$.

Definition 3. We say that a vector of decision variables $\vec{x}^* \in \mathcal{F} \subset \mathbb{R}^n$ ($\mathcal{F}$ is the feasible region) is **Pareto-optimal** if it is non-dominated with respect to $\mathcal{F}$.

Definition 4. The **Pareto Optimal Set** $\mathcal{P}^*$ is defined by:

$$\mathcal{P}^* = \{\vec{x} \in \mathcal{F} | \vec{x} \text{ is Pareto-optimal}\}$$

Definition 5. The **Pareto Front** $\mathcal{PF}^*$ is defined by:

$$\mathcal{PF}^* = \{\vec{f}(\vec{x}) \in \mathbb{R}^k | \vec{x} \in \mathcal{P}^*\}$$

We thus wish to determine the Pareto optimal set from the set $\mathcal{F}$ of all the decision variable vectors that satisfy (1.4) and (1.5). Note however that in practice, not all the Pareto optimal set is normally desirable (*e.g.*, it may not be desirable to have different solutions that map to the same values in objective function space) or achievable.

1.4. The ACO metaheuristic for MOOPs in the literature

An important number of proposals of ACO algorithms have shown their applicability to multi-objective optimization problems with different degrees of success. Many of those proposals were reviewed by García-Martinez *et al.* [4], by Angus and Woodward [5], and by Coello Coello *et al.* [6]. In García-Martinez *et al.* [4] a taxonomy of the ACO metaheuristic for MOOPs is proposed based on two criteria (number of structures to store the pheromone trail and number of heuristic functions) and considering a number of existing ACO algorithms for MOOPs (MOACOs). In addition, the authors include a comparative study of some of the reviewed algorithms and two multi-objective evolutionary algorithms, an improved Strength Pareto Evolutionary Algorithm (SPEA2) [19] and a Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization (NSGA-II) [20]. For the experimental study, it was considered the bi-criteria TSP. In a more recent report, Angus and Woodward [5] present an alternative and extended taxonomy of the ACO metaheuristic for MOOPs which includes five attributes to classify a particular algorithm as seen in

Table 1.1. Based on that taxonomy, the authors describe a set of relevant ACO algorithms for MOOPs as well as their main characteristics.

Table 1.1. The taxonomy proposed by Angus and Woodward [5]

Attribute	Values
Pheromone matrix	Multiple, Single
Solution construction	Targeted, Dynamic, Fixed
Evaluation	Pareto, Non-Pareto
Pheromone Update	Individual, Global
Pareto archive	Offline, Online, Elitist, None

In the remainder of this section, we extend the literature review presented in García-Martinez *et al.* [4], Angus and Woodward [5], and Coello Coello *et al.*. [6] by considering some recent and relevant proposals of the ACO metaheuristic for MOOPs. In the next section, we discuss a refined taxonomy of ACO algorithms for MOOPs.

It is worth noticing that there exist few applications of ACO algorithms for multi-objective problems in continuous domains. For example, Angus [21] proposed a Population-based ACO algorithm for Multi-Objetive Function Optimization (PACO-MOFO). PACO-MOFO is based on the Crowding Population-based ACO algorithm (CPACO) [22] (which was designed for discrete domains) and $ACO_{\mathbb{R}}$ [18]. PACO-MOFO uses an *a posteriori* preference articulation method and implements two niching approaches: crowding (for the population replacement) and fitness sharing (for the selection mechanism). The authors compared the proposed algorithm with NSGA-II [20] on a set of four widely known benchmark problems: MOP1, MOP2, MOP3, and MOP6 (see [6, 23]). Based on the summary attainment analysis performance assessment measure [24], the results show a similar performance of PACO-MOFO and NSGA-II on problems MOP2, MOP3, and MOP6, but not problem MOP1 where NSGA-II outperforms PACO-MOFO. An additional experiment was conducted to study the algorithm's scalability for which problem MOP2 was considered. In this case, PACO-MOFO showed some interesting results regarding the quality of the solutions found in the center of the Pareto front in comparison with those found by NSGA-II and in view of the capability of PACO-MOFO to find solutions when considering a higher dimensionality problem. Another proposal of MOACOs in continuous domains is the work of Garcia-Nareja and Bullinaria [25] in which they present an extended version of $ACO_{\mathbb{R}}$ [18]. To deal with several objectives they maintain an archive

of selected solutions (the approximation of the Pareto optimal set). The concept of *dominance depth* is used to compare the quality of the solutions. The criteria chosen to determine which solution should be eliminated from the archive is the *crowding distance*. The proposed algorithm is applied to a well-known set of multi-objetive problems and compared to NSGA-II [20] and the Multi-Objective Particle Swarm Optimizer (MOPSO) [26]. The solutions found by the extended $ACO_{\mathbb{R}}$ were comparable to those found by NSGA-II and MOPSO.

As one would expect, the use of ACO in discrete MOOPs is, by far, the most commonly reported in the specialized literature. Next, we present some of the most recent proposals in this regard.

Liu *et al.*, [27] presented an ant algorithm, called MO-ant, to generate Pareto fronts for multi-objective siting of Emergency Service Facilities (EFSs). They used the concept of Pareto ranking [28] to determine which were the most qualified solutions to the problem of their interest. The geographic area under study was represented as a grid map from which the ants had to find the best sites to allocate the EFSs. MO-ant does not use any heuristic information and it manages one pheromone matrix, in which each position in that matrix represents the desirability of allocating an EFS at the corresponding position in the real geographic area. In addition, this algorithm applies at each iteration a two-phase local search procedure involving a Pareto ranking of the solutions generated. During the first phase of the local search, it applies the so-called *Neighborhood Random Search* (NRS) by which the ants randomly move from one cell to another one within a certain distance. After that, all the solutions are re-evaluated and a Pareto ranking is obtained. The second phase of the local search consists in the application of the so-called *Adaptive Enumeration Neighborhood Search* (AENS) to the first solution in the set of previously ranked solutions. AENS aims at finding a better position than the current one for every ant in the colony by considering all the cells within a certain distance. If the total sum of the objective values has improved, the ant moves to the cell that produced such a global improvement. This procedure is applied until no more improvements can be observed. The new set of solutions found by AENS are then Pareto ranked. The pheromone matrix is updated by taking into account the rank of the solutions found. As can be seen, the search of the Pareto front is mainly guided by the local search procedure as well as by the pheromone values. MO-ant was applied in a real-world scenario: the multi-objective siting of fire stations in Singapore for which three objectives were considered. The results obtained were compared with

respect to a set of Pareto ranked solutions obtained out of 10,000 randomly generated solutions.

Bui *et al.* [29] investigated the effect of elitism in a multi-objective ACO algorithm under local[c], global, and mixed non-dominated solutions. The MOACO algorithm used one pheromone matrix for each of the objectives. The authors presented a systematic study on five instances of the multi-objective TSP (mTSP) considering the following elitist alternatives: a) local-best, b) global-best, and c) local-set (all solutions, dominated and non-dominated, generated in one iteration). These elitist alternatives are then used in the pheromone updating process in one of the following ways: i) local-set, ii) local-best, iii) local-best + local-set, iv) global-best, and v) global-best+local-set. In addition, the authors proposed an adaptive mechanism (aging) to control the use of an external archive for the global-best elitist alternative. The results indicate the importance of including elitism in MOACOs, for which global elitism was the mechanism that provided more effective information. Also, the adaptive strategy of aging showed an extra improvement on the quality of the solutions found.

Benlian and Zhiquan [30] proposed a MOACO-based data association method for bearings-only multi-target tracking. The algorithm uses multiples matrices, one for each of the two objectives considered: distance and slope difference. The heuristic information is shared by both pheromone trail matrices and the corresponding values are calculated as an aggregating function that combines the heuristic values of each objective. The conducted experiments showed an improved performance of the proposed MOACO with respect to the joint Maximum Likelihood method.

Mora *et al.* [31] reported a comparison of six different ACO algorithms for the Bi-criteria Military Path-Finding Problem in which they defined two objectives: minimize resources while maximizing safety on a map corresponding to a simulated battlefield. Four of the compared algorithms are variants of an enhanced version of the Compañía de Hormigas Acorazadas (CHAC) or Armoured Ant Company [32], called hexa-CHAC (hCHAC) [33]. The original CHAC is basically an ACS including one pheromone matrix for each objective as well as the corresponding heuristic information. In CHAC, two different state transition rules were tested, one based on an aggregating function that combines heuristic and pheromone information regarding the two objectives (CSTR) and the other one, based on the dominance among solutions (DSTR). An additional pa-

[c]The term *local* is used by the authors to refer to the current iteration.

rameter ($0 \leq \lambda \leq 1$) was used in CSTR and DSTR to control the relative importance of the two objectives of the problem under study. For the tested scenarios, CHAC with CSTR showed a better performance. However, such a performance was less robust than that of CHAC with DSTR. On the other hand, hCHAC represented an algorithm capable of dealing with more realistic problem conditions and constraints and the main difference with CHAC was that the scenarios were modeled as a grid of hexagons which implies a change in the construction graph, and the possibility of working with real-world images by defining an underlying information layer. The other two algorithms compared in Mora *et al.*, [31] were the so-called mono-hCHAC (a version the uses an aggregating function combining the two objectives into a scalar value) and the Multi-Objective Ant Colony System (MOACS) [34]. From this comparison, hCHAC-CSTR was considered to be the best overall performer for this particular problem, *i.e.*, the enhanced CHAC algorithm applying a combined state transition rule during the solution construction phase.

McMullen and Tarasewich [35] presented an application of the ACO metaphor to solve a multi-objective assembly line balancing problem. The approach adopted by the authors consisted of using an *ad hoc* aggregating function that combined the four objectives considered: required crew size, system utilization, probability of jobs being assembled within a certain time frame, and cost of the system design. This function (called *metric*) is involved in the computation of pheromone levels associated to a particular task i when considering its assignment to the work center j. In addition, the pheromone levels depend on a value observed regarding the historical precedence of task i with respect to the remaining task in work center j. The authors used the concept of "efficient frontier[d]" to measure the solutions quality by considering two entities: required crew size and a value obtained by a combination of system utilization, probability of jobs being assembled within a certain time frame, and cost of the system design.

Xing *et al.* [36] proposed a fuzzy multi-objective ACO algorithm with linguistically quantified decision functions for flexible job shop scheduling with an interactive decision maker (DM). The main contribution of this work is the interaction of the ACO algorithm with a DM at each iteration in order to bias the search. Particularly, the DM takes into account the distance of the best solution found at each iteration with respect to the aspiration level considering a fuzzy metric. In this case, a classical single-

[d]Efficient frontier is the term used in Operations Research to denote the Pareto front of a problem.

objective ACO algorithm is used as the search engine within a more complex process that includes a close interaction with the DM to solve a multi-objective problem.

Angus [22] extended a Population-based ACO algorithm [37] (PACO) with a Crowding population replacement scheme for the multi-objective the TSP (CPACO). The proposed CPACO algorithm uses only one pheromone matrix and individual heuristic values for each objective. The basic idea of the crowding scheme is taken from Evolutionary Computation (EC). In the CPACO algorithm, the pheromone matrix at each iteration is updated in the following way: i) the set of solutions in the population are ranked (the same approach used in NSGA-II [20]), ii) all the elements of the pheromone matrices are re-initialized to a value τ_{init}, and iii) the ranked solutions in the population produce an updating bias in the re-initialized pheromone matrices using the inverse of their ranks. Thus, the better the rank obtained for a solution, the more the amount of pheromone laid on the corresponding matrix entrances determined by that solution. The way in which CPACO combines one pheromone matrix and several heuristic values (one for each objective) follows the proposal of Barán and Schaerer [34]. CPACO was studied on a set of bi-objective test instances of TSP as well as a 4-objective version of TSP. The comparative study of CPACO with respect to the original PACO showed an improved performance of CPACO on the bi-objective instances of TSP. Although the results were not as good as for the first set of instances, CPACO showed an acceptable performance on the 4-objective TSP. The authors indicated that the pheromone update process and the use historical information are mechanisms that require further research.

In Alaya *et al.* [38], a generic ACO algorithm for multi-objective problems was presented. The proposed ant algorithm (called m-ACO) follows the design of a $\mathcal{MM}$AS and it is parameterized with the number of colonies and the number of pheromone matrices. As the values of these parameters can vary, the authors instantiated them on four different ways: m-ACO$_1(m+1, m)$, m-ACO$_2(m+1, m)$, m-ACO$_3(1, 1)$, and m-ACO$_4(1, m)$, where m represents the number of objectives. The two first variants are similar in the sense that they use $m+1$ colonies and m pheromone matrices. The difference is that the first algorithm produces solutions in colony $m+1$ by using a randomly selected pheromone matrix (*i.e.*, a random objective) to build the solutions whereas in the second variant, the pheromone values associated to colony $m+1$ correspond to the sum of the all the pheromone values associated to each objective. In the third variant, m-ACO$_3(1, 1)$,

only one colony and one pheromone matrix are involved. This is, in fact, very similar to the solution of a single-objective problem. However, the heuristic values are obtained in this case as the sum of the heuristic values associated to each objective. In addition, the pheromone update process only takes into account the non-dominated solutions. Thus, the components of each of the non-dominated solutions indicated which entrances of the pheromone matrix would be rewarded. The amount of pheromone added to each component was the same for all the non-dominated solutions. Finally, the last variant (m-ACO$_4(1, m)$) used one colony and m pheromone matrices. First, each ant of the (only) colony randomly selects one objective and then builds the solution using the corresponding pheromone matrix. The heuristic values are always calculated as in the third variant. At each iteration, *i.e.*, once the colony has built the solutions, the pheromone matrices are updated by considering in turn the best m solutions with respect to each of the m objective function values. The experimental study reported the results of the four variants described above when applied to different instances of the multi-objective multidimensional knapsack problem. The variant m-ACO$_4(1, m)$ globally achieved the best performance on the tested instances. This variant was compared with several non-elitist multi-objective evolutionary algorithms[e] (MOGA [39], NSGA [40], SPEA [41], HLGA [42], and VEGA [43]) based on the coverage of two sets measure [44] for which m-ACO$_4(1, m)$ found the best results except for some instances when compared with SPEA [41]. Although the presentation of m-ACO seems to be a generalization for multi-objective subset problems, it could also be generalized for any discrete combinatorial optimization problem.

Afshar *et al.* [45] presented a bi-colony ACO algorithm with a Non-dominated Archive (NA-ACO) which combines a novel interaction of the two involved colonies to evolve the solutions. The NA-ACO algorithm maintains one pheromone matrix and an archive of non-dominated solutions. Each colony is in charge of optimizing one of the two objectives.

[e]Elitism, in the context of multi-objective metaheuristics, normally consists of using an external archive (usually called a "secondary population") that can (or cannot) interact in different ways with the main (or "primary") population of the multi-objective metaheuristic. The main purpose of this archive is to store all the non-dominated solutions generated throughout the search process, while removing those that become dominated later in the search (called local non-dominated solutions). The approximation of the Pareto optimal set produced by an algorithm is thus the final contents of this archive. Practically all modern multi-objective evolutionary algorithms (i.e., those designed after 1999) are elitist [6].

NA-ACO works as follows: the first colony produces a set of solutions that are moved and evaluated according to the objective function assigned to the second colony. From this set of solutions, the second colony selects the best one and uses this solution to update the pheromone matrix. After that, a new set of solution is generated and moved to the first colony for evaluation according to the corresponding objective function. The new best ant is then considered for pheromone updating. The above process is repeated for N cycles after which the values of the objective functions are computed, in order to find the non-dominated solutions to be stored in the archive. The pheromone matrix is re-initialized and then updated according to the new solutions in the archive. The whole process is repeated for M iterations. NA-ACO algorithms was first studied on problems ZDT1 and ZDT2 [46], and compared with NSGA-II [20], SPEA [41], and the Pareto Archived Evolution Strategy (PAES) [47]. In addition, NA-ACO was tested on two bi-objective water-resource problems for which encouraging results were found when compared with a weighted-sum method.

In an extended abstract, Eppe [48] presented a mechanism that integrated the decision maker's preferences into multi-objective Ant Colony Optimization. The main contribution of this work was the use of a preference function (based on the PROMETHEE methodology [49]) to define a normalized and aggregated preference index. This preference index is applied to a solution at the component level for the TSP. Although no actual results were reported by the author, the proposed mechanism seems to be capable to deal with multi-objective problems.

Chica *et al.*, [50] incorporated preferences to a multi-objective ACO algorithm for a variant of the time and space assembly line balancing problem (TSALBP-1/3) which was previously studied by the authors in [51]. The preferences incorporated were represented by *a priori* information of the problem provided by the plant experts and was used to guide the search. The authors selected a Multiple Ant Colony System (MACS) as their search engine (see [34]). The main goal of the new proposed algorithm was to reduce the size of the Pareto optimal set and increase the quality of the Pareto front. For the authors a high quality Pareto front is one which presents a focused and reduced set of solutions for the decision maker. The generation of such a high quality Pareto front required the redefinition of the concept of Pareto dominance in order to include some specific preferences given by the experts and applied when there were some solutions with the same objective function values. In the case of TSALBP-1/3, the information considered was the adoption of the same value for the area and number

of stations for a fixed cycle time in the assembly line. The new dominance definition takes into account the workload of the plant and the required space for the workers' instruments. Accordingly, two measures were formulated and the corresponding preference-based dominance definitions were provided. MACS was tested in a real world problem: the assembly process of the Nissan Pathfinder engine. The results were compared to those of the Multi-Objective Randomised Greedy Algorithm (MORGA) [51] and to a modified version of the ACS reported in [51]. The results showed an improved performance of MACS with respect to the other two algorithms with respect to which it was compared, regarding each of the metrics considered in the experimental study. Following this same line of research, Chica *et al.* [52] extended this proposal by incorporating the elicitation of preferences regarding the economic factors related to the location of the plant. These preferences were included in the objective space in order to achieve a more focused Pareto front and then make easier the task of the decision maker. They used six scenarios around the world and the preferences were based on the Evolutionary Multi-Objective (EMO) preferences given in [53, 54].

Häckel *et al.* [55] developed an ACO algorithm for solving the multi-objective shortest path problem. The authors considered in their experimental study a problem with three criteria. However, their proposal can be extended to a higher number of objectives. Interestingly, they used only one pheromone matrix. However, their corresponding trail values were weighted differently according to each ant and criteria. This weighting policy was also applied to the heuristic values which were calculated by considering four alternatives: i) standard (inverted edge weights), ii) LAH/bc, iii) LAH/wc, and iv) LAH/ac; where LAH stands for Look-Ahead Heuristic considering the best (bc), worst (wc), and average (ac) cases. The corresponding LAH values were separately calculated using dynamic programming for each of the criteria involved. Two ACO algorithms were tested: the Ant System (AS) and the Ant Colony System (ACS). The ACS with LAH/wc showed the best overall performance regarding the set of *ad hoc* metrics adopted by the authors.

Chaharsooghi and Kermani [56] proposed a multi-objective ACO algorithm for solving the Multi-Objective Resource Allocation Problem (MORAP) in which the objectives considered were: maximization of workers' efficiency and minimization of the resources cost. The ACO algorithm used only one pheromone matrix and only one structure to store the heuristic values that represented a combined calculation obtained as the product of the heuristic values associated to the two separate objectives: efficiency

and cost. The pheromone matrix was updated considering a modified updating rule that took into account the concept of non-dominance between solutions, current iteration, and an *ad hoc* parameter to control the influence of the current generation on the amount of pheromone trail to be deposited. Since the approach adopted for pheromone updating can generate negative values when calculating the probability used during the construction solution phase, a mechanism was incorporated to deal with this problem. The proposal was compared and showed an improved performance with respect to a hybrid GA (hGA) using only one particular instance of MORAP. In a related work of Chaharsooghi and Kermani [57] a similar approach was adopted for solving the multi-objective Multidimensional Knapsack Problem. The main difference on the ACO algorithm proposed here is that this algorithm used multiple colonies, each one updating its own pheromone matrix associated to the two objectives considered here. The probability of selection used in the phase of solution construction was based exclusively on a combination of the pheromone values for each objective and no heuristic information was considered. This version of the ACO algorithm was applied on a set of instances of the knapsack problem and compared with NSGA-II [20]. Based on two different metrics, the authors claimed that the ACO algorithm outperformed NSGA-II when considering the solutions on the extreme portions of the Pareto front. However, by ruling out these extreme solutions, NSGA-II actually outperformed the proposed ACO algorithm.

Vieira *et al.* [58] dealt with the feature selection problem as a multicriteria problem with a single objective function. Two criteria were considered: the size of the subset of features and the features that are to be selected to build a fuzzy classifier. The ant algorithm used two colonies, two pheromone matrices and two different heuristic values, one for each criterion. The objective function to be minimized was an aggregation of the two criteria which were combined in order to measure the classification error rate and the number of features. The ant algorithm worked as follows: the first colony was in charge of selecting the number of features (one for each ant) and then, this selected number was used for the corresponding ant in the second colony in order to find the features used to build the classifier. The ant algorithm was tested on data sets taken from the UCI repository [59] and compared with some previous works. The achieved performance of the proposed ant algorithm was similar or better to that of the compared algorithms.

In Yang *et al.* [60], a multi-objective task scheduling approach for Grid

over Optical Burst Switching (GOBS) networks was proposed, considering three objectives: 1) completion time, 2) cost for using the resources, and 3) load balancing. The Multi-Objective ACO algorithm (called MOACO in this work) incorporated the idea of Pareto dominance as well as an *ad hoc* operator that combined pheromone exchanges and a sharing niching method. MOACO uses one pheromone and one heuristic information (lineal) structure associated to each of the resources in the network. The pheromone structure was initialized according to the computational capacity of each resource, *i.e.*, the higher the computational capacity, the higher the initial amount of the pheromone trail. Similarly, the heuristic values were calculated combining the corresponding initial amount of pheromone trails (*i.e.*, computational resources) and network resources. Interestingly, the probability of selection of a particular resource included a threshold value below which the corresponding resource could not choose (*i.e.*, it avoided the overuse of computational resources). With respect to the pheromone updating, the authors proposed, as in the Ant Colony System, two pheromone updating rules: local and global. The local one aims at influencing the pheromone values according to the time necessary to execute a job assigned to that particular computational resource considering a recently obtained solution, whereas the global one is similar to the local rule but the solution involved is the one obtained by searching the optimal solution found by one of the following: either pheromone exchanging or the sharing niching method. The experimental study was conducted on one particular instance of GOBS and the results were compared with respect to those of NSGA-II [20]. The results showed that MOACO outperformed NSGA-II. The authors claim that these results are due to the fact that in MOACO the pheromone values are initialized such that they incorporate information about the problem, whereas NSGA-II performs a blinder search. Additional experiments were conducted to compare separately the results of MOACO with respect to those found by a single-objective ACO algorithm considering each objective in turn.

1.5. ACO Variants for MOOP: A Refined Taxonomy

When considering the development of the Evolutionary Multiobjective Optimization (EMO) field, it is clear that ACO has a lot of room left for expansion, particularly if we consider continuous search spaces. The literature review presented by García-Martinez *et al.* [4], Angus and Woodward [5], Coello Coello *et al.*. [6], and in this chapter, show important

achievements of muti-objective ACO in several application domains within the last few years. Here, we complement this information with a refinement of the taxonomy proposed by Angus and Woodward [5]. More precisely, we reconsider the different attributes presented in [5] and some dependencies observed among them. Based on the previous discussion, we propose a taxonomy with hierarchical elements that could better help to visualize the different components of the existing MOACOs. Our taxonomy also adds an additional attribute that was discussed in [5] but was not included in their proposed taxonomy. The names and meaning of each of the attributes considered remains the same in our taxonomy as those described in [5].

Before describing our extended taxonomy, let us discuss the inclusion of the new attribute #Colonies. As indicated before, in Angus and Woodward [5] this attribute was discussed but not included. Although the design of multi-colonies ACO algorithms must consider a certain level of communication of solutions between colonies for achieving an improved performance, this alternative should also be considered when showing to the researchers a global perspective of the current and future developments of MOACOs. Particularly, those researchers involved in parallel models of the ACO metaheuristic could be very interested in the development and application of existing and new parallel models of MOACOs. On the other hand, we made a simple modification and created a hierarchical taxonomy by associating some attributes to specific places on the tree (see Figure 1.2). For example, the only way of selecting between Global and Individual pheromone update (according to the meaning given in [5]) is when the particular MOACO algorithm includes multiple pheromone matrices. Similarly, the attribute *Evaluation* has two possible values: Pareto and Non-Pareto. By choosing the option "Pareto", two alternatives (values) are possible, *i.e.*, a Pareto Archive can be used or not. Thus, the option that includes the use of a Pareto Archive allows us to select from among (at least) three options: Offline, Online, and Elitist.

Similarly to the claims of Angus and Woodward [5], the taxonomy proposed here is mainly aimed at gaining an enhanced understanding of the different up-to-date design choices that have been explored in the field the ACO variants for MOOPs.

From the above, it is clear that many possibilities can be considered when attempting to design an ACO algorithm for multi-objective problems. This makes difficult to present an unified ACO algorithm that embodies all those alternatives. In spite of that, we give in the following a possible example of a general MOACO algorithm as shown in Algorithm 1.2 for

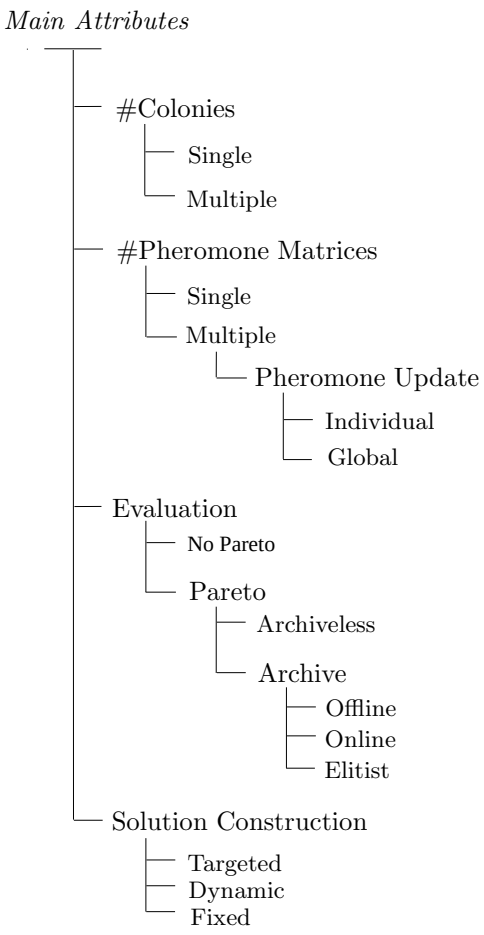

Fig. 1.2. A refined and extended taxonomy of MOACO algorithms.

solving a multi-objective TSP-like problem. The design of Algorithm 1.2 includes a single colony of ants, k pheromone matrices and k structures that maintain respectively the amount of pheromone trails and heuristic information associated to each one of the k problem objectives. This algorithm corresponds to the class of MOACO algorithms that use Pareto evaluation for the solutions and maintain an archive, in this case, a set of ranked non-dominated solutions. In the following we describe in detail each one of the main components of the MOACO presented in Algorithm 1.2. Firstly, it must be noticed that t represents the current generation, t_{max} is

the maximum number of iterations, N_a is the number of ants in the colony, and $\mathcal{A}$ is the archive that maintains the set of non-dominated solutions and it is updated at each iteration.

Algorithm 1.2 Outline of a possible MOACO for k objectives

1: Init_Pheromone_Trails(τ^i_{rj}); // for $i = 1, \ldots, k$; $r, j = 1, \ldots, n$
2: $\mathcal{A} = \emptyset$; // Archive of ranked non-dominated solutions
3: $t = 0$;
4: **while** ($t \leq T_{max}$) **do**
5: **for** $h = 1, \ldots, N_a$ **do**
6: BuildSolution$_h$(S(t), τ^1, ..., τ^k, η^1, ..., η^k);
7: **end for**
8: ArchiveUpdate($S(t)$, $\mathcal{A}$);
9: PheromoneUpdate(τ^1, ..., τ^k, $\mathcal{A}$);
10: $t = t + 1$
11: **end while**
12: return $\mathcal{A}$; // the Pareto front achieved

Algorithm 1.2 starts by initializing the k pheromone matrices (Init_Pheromone_Trails()). These initial values could depend on the particular objective. The archive of non-dominated solutions is initially empty (line 2). Function BuildSolution$_h$() is executed once for each of the N_a ants. This function is in charge of building a particular solution based on information of the k pheromone matrices and the respective heuristic information for each objective. As a general algorithm design, we do not show the way in which those values are used. Nevertheless, many possibilities can be considered, *e.g.*, building an aggregating function used to calculate the probability values involved at each step on the solution construction process. The N_a solutions obtained at iteration t are stored in $S(t)$. After that, the archive of non-dominated solutions is updated in a such way that the new created solutions in $S(t)$ compete with the solutions stored in $\mathcal{A}$ to obtain a renewed set of ranked non-dominated solutions (ArchiveUpdate()). According to the new set $\mathcal{A}$ of non-dominated solutions, all the pheromone matrices are updated taking into account the respective objective values. Finally, after T_{max} iterations, the achieved set of non-dominated solutions is returned.

1.6. Promising Research Areas

There are several topics within this area that we believe that have a high potential for future research. The main ones are the following:

- **Use in continuous optimization problems:** In contrast with the several ACO variants that have been recently proposed for continuous single-objective search spaces (see for example [15, 16, 18]), such proposals are very scarce within multi-objective optimization. Clearly, the development of multi-objective extensions of continuous variants of ACO could bring a variety of novel applications. This could potentially place ACO next to other metaheuristics which are very popular for continuous multi-objective optimization (*e.g.*, particle swarm optimization).
- **Parallelization:** ACO algorithms are, by definition, highly distributed algorithms in which a set of ants are in charge of independently building a solution. Thus, their parallelization is relatively straightforward, but such a feature has not been yet properly exploited in a multi-objective optimization context. Furthermore, ACO algorithms are flexible enough to allow the addition of many components that can be combined in many different ways in order to obtain improved ACO-based multi-objective optimizers (*e.g.*, several pheromone matrices, multiple colonies, heuristic information (when available), Pareto archives, etc). Again, the experience acquired so far regarding parallel MOEAs [6, 61, 62] could be very valuable here.
- **Hybridization:** In the last few years, the hybridization of different types of metaheuristics has become a relatively popular scheme within multi-objective optimization. For example, some MOEAs have been hybridized with powerful local search engines, giving rise to the so-called multi-objective memetic algorithms [63]. MOACOs also have a great hybridization potential, not only with local search engines, but also with other metaheuristics such as particle swarm optimization and genetic algorithms. Such hybrids are expected to become more common during the next few years.

1.7. Conclusions

In this chapter we have presented an overview of the use of the Ant Colony Optimization metaheuristic for solving multi-objective optimization problems. The chapter has provided some introductory concepts about the ACO metaheuristic and about multi-objective optimization. The most relevant features of some recent proposals within the area were briefly reviewed, too. Based on the existing MOACOs and a particular taxonomy proposed in the literature, we presented a refined taxonomy of MOACO algorithms that could help practitioners to better identify and exploit the potential of the ACO metaheuristic as a multi-objective optimizer. Nevertheless, we hope that this chapter can be found useful not only for practitioners, but also for students and researchers interested in multi-objective optimization using ACO algorithms, for such has been its main purpose.

1.8. Acknowledgments

The first author acknowledges the support from the UMI-LAFMIA 3175 CNRS at CINVESTAV-IPN and from the Universidad Nacional de San Luis, Argentina. The second author gratefully acknowledges support from CONACyT project no. 103570.

References

[1] R. C. Eberhart, Y. Shi, and J. Kennedy, *Swarm Intelligence*. The Morgan Kaufmann Series in Artificial Intelligence, (Morgan Kaufmann, April 2001), first edition.

[2] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence - From Natural to Artificial Systems*. Santa Fe Institute Studies in the Sciences of Complexity, (Oxford University Press, 1999).

[3] M. Dorigo and T. Stützle, *Ant Colony Optimization*. (Mit-Press, 2004).

[4] C. García-Martínez, O. Cordón, and F. Herrera, 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, (2007).

[5] D. Angus and C. Woodward, Multiple objective ant colony optimisation, *Swarm Intelligence*. **3**(1), 69–85, (2009).

[6] C. A. Coello Coello, D. A. Van Veldhuizen, and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*. (Springer, September 2002), second edition. ISBN 0-387-33254-3.

[7] M. Dorigo, V. Maniezzo, and A. Colorni, Ant system: Optimization

by a colony of cooperating agents, *IEEE Trans. on Systems, Man, and Cybernetics–Part B.* **26**(1), 29–41, (1996).

[8] G. Bilchev and I. Parmee. The Ant Colony Metaphor for Searching Continuous Design Spaces. In ed. T. C. Fogarty, *Evolutionary Computing. AISB Workshop*, pp. 25–39. Springer, Sheffield, UK (April, 1995).

[9] N. Monmarché, G. Venturini, and M. Slimane, On how pachycondyla apicalis ants suggest a new search algoritm, *Future Generation Computer Systems.* **16**, 937–946, (2000).

[10] J. Li and N. Satofuka, Optimization design of a compressor cascade airfoil using a Navier-stokes solver and genetic algorithms, *Proceedings of the Institution of Mechanical Engineering Part A—Journal of Power and Energy.* **216**(A2), 195–202, (2002).

[11] J. Dréo and P. Siarry. A New Ant Colony Algorithm Using the Heterarchical Concept Aimed at Optimization of Multiminima Continuous Functions. In eds. M. Dorigo, G. Di Caro, and M. Sampels, *Proceedings of the Third international Workshop on Ant Algorithms - ANTS 2002*, pp. 216–221. Springer-Verlag. Lecture Notes in Computer Science Vol. 2463, Brussels, Belgium (September, 2002).

[12] J. Dréo and P. Siarry, Continuous interacting ant colony algorithm based on dense heterarchy, *Future Generation Comp. Syst.* **20**(5), 841–856, (2004).

[13] L. Q. Ling Chen, J. Shen and H. Chen, An improved ant colony algorithm in continuous optimization, *Journal of Systems Science and Systems Engineering.* **12**(2), 224–235, (2003).

[14] S. Pourtakdoust and H. Nobahari. An Extension of Ant Colony Systems to Continuos Optimization Problems. In eds. M. Dorigo, M. Birattari, C. Blum, L. M. Gambardella, F. Mondada, and T. Stützle, *Proceedings of Ant Colony Optimization and Swarm Intelligence, 4th International Workshop, ANTS Workshop 2004*, pp. 294–301. Lecture Notes in Computer Science Vol. 3172, Brussels, Belgium, (2004). Springer-Verlag.

[15] M. Kong and P. Tian. A direct application of ant colony optimization to function optimization problem in continuous domain. In *ANTS Workshop*, pp. 324–331, (2006).

[16] X. Hu, J. Zhang, and Y. Li, Orthogonal methods based ant colony search for solving continuous optimization problems, *J. Comput. Sci. Technol.* **23** (1), 2–18, (2008).

[17] K. Socha. ACO for continuos and mixed-variable optimization. In eds. M. Dorigo, M. Birattari, C. Blum, L. M. Gambardella, F. Mondada, and T. Stützle, *Proceedings of Ant Colony Optimization and Swarm Intelligence, 4th International Workshop, ANTS Workshop 2004*, pp. 25–36, Brussels, Belgium, (2004). Springer-Verlag. Lecture Notes in Computer Science Vol. 3172.

[18] K. Socha and M. Dorigo, Ant colony optimization for continuous domains, *European Journal of Operational Research.* **185**(3), 1155–1173, (2008).

[19] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In eds. K. Giannakoglou, D. Tsahalis, J. Periaux, P. Papailou, and T. Fogarty, *EUROGEN 2001. Evolutionary Methods*

for Design, Optimization and Control with Applications to Industrial Problems, pp. 95–100, Athens, Greece, (2001).

[20] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II, *IEEE Transactions on Evolutionary Computation*. **6**(2), 182–197 (April, 2002).

[21] D. Angus. Population-based ant colony optimisation for multi-objective function optimisation. In eds. M. Randall, H. A. Abbass, and J. Wiles, *ACAL*, vol. 4828, *Lecture Notes in Computer Science*, pp. 232–244. Springer, (2007). ISBN 978-3-540-76930-9.

[22] D. Angus. Crowding Population-based Ant Colony Optimization for the Multi-objective Travelling Salesman Problem. In *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Multicriteria Decision Making (MCDM'2007)*, pp. 333–340, Honolulu, Hawaii, USA (April, 2007). IEEE Press.

[23] D. A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio (May, 1999).

[24] J. Knowles. A summary-attainment-surface plotting method for visualizing the performance of stochastic multiobjective optimizers. In *Fifth International Conference on Intelligent Systems Design and Applications (ISDA'2005)*, pp. 552–557. IEEE, (2005).

[25] A. Garcia-Najera and J. A. Bullinaria. Extending ACO_R to Solve Multi-Objective Problems. In ed. G. M. Coghill, *Proceedings of the UK Workshop on Computational Intelligence (UKCI 2007)*, London, UK, (2007). Imperial College United Kingdom.

[26] C. A. Coello Coello, G. Toscano Pulido, and M. Salazar Lechuga, Handling Multiple Objectives With Particle Swarm Optimization, *IEEE Transactions on Evolutionary Computation*. **8**(3), 256–279 (June, 2004).

[27] N. Liu, B. Huang, and X. H. Pan, Using the Ant Algorithm to Derive Pareto Fronts for Multiobjective Siting of Emergency Service Facilities, *Transportation Research Record: Journal of the Transportation Research Board*. **1935**, 120–129, (2005).

[28] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. (Addison-Wesley Publishing Company, Reading, Massachusetts, 1989).

[29] L. T. Bui, J. M. Whitacre, and H. A. Abbass. Performance Analysis of Elitism in Multi-Objective Ant Colony Optimization Algorithms. In *2008 Congress on Evolutionary Computation (CEC'2008)*, pp. 1633–1640, Hong Kong (June, 2008). IEEE Service Center.

[30] X. Benlian and W. Zhiquan, A multi-objective-ACO-based data association method for bearings-only multi-target tracking, *Communications in Nonlinear Science and Numerical Simulation*. **12**(8), 1360–1369, (2007).

[31] A. M. Mora, J. J. M. Guervós, C. Millán, J. Torrecillas, J. L. J. Laredo, and P. A. C. Valdivieso. Comparing ACO Algorithms for Solving the Bicriteria Military Path-Finding Problem. In eds. F. A. e Costa, L. M. Rocha,

E. Costa, I. Harvey, and A. Coutinho, *Advances in Artificial Life. 9th European Conference (ECAL'2007)*, pp. 665–674. Springer, Lecture Notes in Computer Science, Vol. 4648, Lisbon, Portugal (September 10-14, 2007). ISBN 978-3-540-74912-7.

[32] A. Mora, J. Merelo, C. Millan, J. Torrecillas, and J. Laredo. CHAC. A MOACO Algorithm for Computation of Bi-Criteria Military Unit Path in the Battlefield. In eds. D. Pelta and N. Krasnogor, *Proceedings of the First Workshop in Nature Inspired Cooperative Strategies for Optimization (NICSO'06)*, pp. 85–96, Granada, Spain (June, 2006).

[33] A. M. Mora, J. J. M. Guervós, C. Millán, J. Torrecillas, J. L. J. Laredo, and P. A. C. Valdivieso. Enhancing a moaco for solving the bi-criteria pathfinding problem for a military unit in a realistic battlefield. In eds. M. Giacobini, A. Brabazon, S. Cagnoni, G. D. Caro, R. Drechsler, M. Farooq, A. Fink, E. Lutton, P. Machado, S. Minner, M. O'Neill, J. Romero, F. Rothlauf, G. Squillero, H. Takagi, S. Uyar, and S. Yang, *EvoWorkshops*, vol. 4448, *Lecture Notes in Computer Science*, pp. 712–721. Springer, (2007). ISBN 978-3-540-71804-8.

[34] B. Barán and M. Schaerer. A Multiobjective Ant Colony System for Vehicle Routing Problem with Time Windows. In *Proceedings of the 21st IASTED International Conference on Applied Informatics*, pp. 97–102, Innsbruck, Austria (February, 2003). IASTED.

[35] P. R. McMullen and P. Tarasewich, Multi-objective assembly line balancing via a modified ant colony optimization technique, *International Journal of Production Research*. **44**, 27–42, (2006).

[36] L.-N. Xing, Y.-W. Chen, and K.-W. Yang. Interactive fuzzy multi-objective ant colony optimization with linguistically quantified decision functions for flexible job shop scheduling problems. In *FBIT '07: Proceedings of the 2007 Frontiers in the Convergence of Bioscience and Information Technologies*, pp. 801–806, Washington, DC, USA, (2007). IEEE Computer Society. ISBN 978-0-7695-2999-8. doi: http://dx.doi.org/10.1109/FBIT.2007.18.

[37] M. Guntsch and M. Middendorf. A Population Based Approach for ACO. In *Applications of Evolutionary Computing. EvoWorkshops 2002: EvoCOP, EvoIASP, EvoSTIM/EvoPLAN*, pp. 72–81, Kinsale, Ireland (April, 2002). Springer. Lecture Notes in Computer Science Vol. 2279.

[38] I. Alaya, C. Solnon, and K. Ghédira. Ant Colony Optimization for Multi-objective Optimization Problems. In *Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)*, vol. 1, pp. 450–457. IEEE Computer Society Press (October, 2007).

[39] C. M. Fonseca and P. J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In ed. S. Forrest, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp. 416–423, San Mateo, California, (1993). University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers.

[40] N. Srinivas and K. Deb, Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms, *Evolutionary Computation*. **2**(3), 221–248 (fall, 1994).

[41] E. Zitzler and L. Thiele, Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach, *IEEE Transactions on Evolutionary Computation*. **3**(4), 257–271 (November, 1999).
[42] P. Hajela and C. Y. Lin, Genetic search strategies in multicriterion optimal design, *Structural Optimization*. **4**, 99–107, (1992).
[43] J. D. Schaffer. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pp. 93–100, Hillsdale, New Jersey, (1985). Lawrence Erlbaum.
[44] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland (November, 1999).
[45] A. Afshar, F. Sharifi, and M. Jalali, Non-dominated archiving multi-colony ant algorithm for multi-objective optimization: Application to multi-purpose reservoir operation, *Engineering Optimization*. **41**(4), 313–325 (April, 2009).
[46] E. Zitzler, K. Deb, and L. Thiele, Comparison of Multiobjective Evolutionary Algorithms: Empirical Results, *Evolutionary Computation*. **8**(2), 173–195 (Summer, 2000).
[47] J. D. Knowles and D. W. Corne, Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy, *Evolutionary Computation*. **8**(2), 149–172, (2000).
[48] S. Eppe. Integrating the decision maker's preferences into multi objective ant colony optimization. In eds. F. Hutter and M. M. de Oca, *2nd Doctoral Symposium on Engineering Stochastic Local Search Algorithms, SLS 2009*, pp. 56–60 (August, 2009).
[49] J.-P. Brans and B. Mareschal. PROMETHEE methods. In eds. J. Figueira, S. Greco, and M. Ehrgott, *Multiple Criteria Decision Analysis. State of the Art Surveys*, pp. 163–195. Springer, New York, USA, (2005).
[50] M. Chica, Óscar Cordón, S. Damas, J. Pereira, and J. Bautista. Incorporating Preferences to a Multi-objective Ant Colony Algorithm for Time and Space Assembly Line Balancing. In eds. M. Dorigo, M. Birattari, C. Blum, M. Clerc, T. Stützle, and A. F. Winfield, *Ant Colony Optimization and Swarm Intelligence. 6th International Conference, ANTS 2008. Proceedings*, pp. 331–338. Springer, Brussels, Belgium (September, 2008).
[51] M. Chica, O. Cordón, S. Damas, J. Bautista, and J. Pereira. Multi-objective, constructive heuristics for the 1/3 variant of the time and space assembly line balancing problem: Aco and randomised greedy. Technical Report AFE-08-01, European Centre for Soft Computing, Asturias (Spain), (2008).
[52] M. Chica, Óscar Cordón, S. Damas, and J. Bautista. Integration of an EMO-based Preference Elicitation Scheme into a Multi-objective ACO Algorithm for Time and Space Assembly Line Balancing. In *2009 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making (MCDM'2009)*, pp. 157–162, Nashville, TN, USA (March 30 - April 2, 2009). IEEE Press. ISBN 978-1-4244-2764-2.
[53] K. Deb. Solving Goal Programming Problems Using Multi-Objective Genetic Algorithms. In *1999 Congress on Evolutionary Computation*, pp. 77–

84, Washington, D.C. (July, 1999). IEEE Service Center.

[54] J. Branke and K. Deb. Integrating User Preferences into Evolutionary Multi-Objective Optimization. In ed. Y. Jin, *Knowledge Incorporation in Evolutionary Computation*, pp. 461–477. Springer, Berlin Heidelberg, (2005). ISBN 3-540-22902-7.

[55] S. Häckel, M. Fischer, D. Zechel, and T. Teich. A Multi-Objective Ant Colony Approach for Pareto-Optimization Using Dynamic Programming. In *2008 Genetic and Evolutionary Computation Conference (GECCO'2008)*, pp. 33–40, Atlanta, USA (July, 2008). ACM Press. ISBN 978-1-60558-131-6.

[56] S. K. Chaharsooghi and A. H. M. Kermani, An effective ant colony optimization algorithm (ACO) for multi-objective resource allocation problem (MORAP), *Applied Mathematics and Computation*. **200**(1), 167–177 (June 15, 2008).

[57] S. K. Chaharsooghi and A. H. M. Kermani. An Intelligent Multi-Colony Multi-Objective Ant Colony Optimization (ACO) for the 0-1 Knapsack Problem. In *2008 Congress on Evolutionary Computation (CEC'2008)*, pp. 1195–1202, Hong Kong (June, 2008). IEEE Service Center.

[58] S. M. Vieira, J. ao M. C. Sousa, and T. A. Runkler. Multi-Criteria Ant Feature Selection Using Fuzzy Classifiers. In eds. C. A. Coello Coello, S. Dehuri, and S. Ghosh, *Swarm Intelligence for Multi-objective Problems in Data Mining*, chapter 2, pp. 19–36. Springer. Studies in Computational Intelligence. Vol. 242, Berlin, (2009).

[59] A. Asuncion and D. Newman. UCI machine learning repository, (2007). URL `http://www.ics.uci.edu/$\sim$mlearn/{MLR}epository.html`.

[60] Y. Yang, G. Wu, J. Chen, and W. Dai, Multi-objective optimization based on ant colony optimization in grid over optical burst switching networks, *Expert Systems with Applications*. **37**(2), 1769–1775 (March, 2010).

[61] D. A. Van Veldhuizen, J. B. Zydallis, and G. B. Lamont, Considerations in Engineering Parallel Multiobjective Evolutionary Algorithms, *IEEE Transactions on Evolutionary Computation*. **7**(2), 144–173 (April, 2003).

[62] A. Nebro, F. Luna, E.-G. Talbi, and E. Alba. Parallel Multiobjective Optimization. In ed. E. Alba, *Parallel Metaheuristics*, pp. 371–394. Wiley-Interscience, New Jersey, USA, (2005). ISBN 13-978-0-471-67806-9.

[63] C.-K. Goh, Y.-S. Ong, and K. C. Tan, Eds., *Multi-Objective Memetic Algorithms*. (Springer, Berlin, Germany, 2009). ISBN 978-3-540-88050-9.