Multiple objective ant colony optimisation

Daniel Angus · Clinton Woodward

Abstract Multiple Objective Optimisation is a fast growing area of research, and consequently several Ant Colony Optimisation approaches have been proposed for a variety of these problems. In this paper, a taxonomy for Multiple Objective Ant Colony Optimisation algorithms is proposed and many existing approaches are reviewed and described using the taxonomy. The taxonomy offers guidelines for the development and use of Multiple Objective Ant Colony Optimisation algorithms.

Keywords Multiple Objective Optimisation · Pareto Optimisation · Ant Colony Optimisation · Taxonomy

1 Introduction

Ant Colony Optimisation (ACO) (Dorigo and Stützle 2004) algorithms have been shown to be effective problem solving strategies for a wide range of problem domains, including Multiple Objective Optimisation (MOO). Given the suitability of stochastic, population-based algorithms to problem domains such as MOO (Deb 2002), it is not surprising that there has been a sustained interest from the ACO community.

While a diversity of ACO approaches is arguably good for the development of the field, it can make it difficult to objectively compare existing Multiple Objective ACO (MOACO) approaches or propose new ones since often it is not clear what element of an algorithm's design contributes to overall success or failure. A possible reason for this situation may be that, with no obvious or standard design guidelines in place, and given the number of possible ways one can adapt ACO to MOO, algorithm designers may be overwhelmed by

D. Angus (✉)
The University of Queensland, Brisbane, Australia
e-mail: d.angus@uq.edu.au

C. Woodward
Swinburne University of Technology, Melbourne, Australia
e-mail: cwoodward@swin.edu.au

an abundance of choice. Indeed this diversity is evident in the MOACO algorithms proposed in the last 10 years which are diverse in both their optimisation goals and how they are achieved.

The focus of the MOACO field to date has been mostly on solving multiple objective *combinatorial* optimisation problems; understandable given the good performance of ACO on many single objective combinatorial optimisation problems (Dorigo and Stützle 2004). In contrast to this, the Multiple Objective Evolutionary Algorithm (MOEA) community focuses in the majority of contributions on multiple objective *function* optimisation problems.[1] Most MOACO proposals tend to be extensions of well-known single objective ACO algorithms, such as Ant Colony System (Dorigo and Gambardella 1997) and $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System ($\mathcal{MMAS}$) (Stützle and Hoos 2000).

A review and analysis of many MOACO approaches was offered in (Garcìa-Martínez et al. 2007). In that paper, all known MOACO algorithms were summarised and a simple taxonomy was proposed that categorised the MOACO algorithms by the number of pheromone matrices and heuristic matrices they use. The paper specifically focused on an empirical analysis of these MOACO algorithms using the Multiple Objective Travelling Salesman Problem (MOTSP). While this analysis is useful for MOACO algorithms that are designed for similar problem domains, it takes many of the MOACO out of their original context and focuses on the modification of MOACO for the MOTSP. Our contribution is different as it aims to provide a broader review and taxonomy, while discussing MOACO in their original context.

In this paper, a new taxonomy is proposed that not only identifies similar MOACO algorithms, but also provides some simple insights into the overall expected algorithmic behaviour for intended problem domains. The taxonomy is based on features common to the ACO algorithms reviewed. These are:

– the choice of pheromone model
– the solution construction process
– whether solutions are evaluated in terms of individual objectives or all objectives
– how solutions are used to update the multiple or individual pheromone matrices
– how Pareto optimal solutions are treated.

Our taxonomy expands on concepts from the existing ACO Metaheuristic Framework (Dorigo and Di Caro 1999; Dorigo et al. 1999), maintaining the core ACO principles of probabilistic stepwise solution construction using artificial pheromone. The use of the existing ACO Metaheuristic Framework is also desirable so as to minimise the introduction of superfluous terminology where possible.

This paper is organised as follows. Several MOO concepts, including *problem solving strategies*, *objective and solution space mappings*, *Pareto optimality*, *dominance*, and *performance metrics* are introduced in Sect. 2. The new MOACO taxonomy and terminology is described in Sect. 3. In Sect. 4, existing MOACO algorithms are listed and discussed, including design guidelines for MOACO algorithms and expected algorithm performance. Concluding remarks are then offered in Sect. 5. This paper does not offer an empirical analysis of all MOACO algorithms reviewed since such a large undertaking and analysis has limited utility for discussion of such a broad range of MOACO algorithms represented by the new taxonomy.

---

[1]This does not suggest that MOEA have not been applied with success to other problem domains; in fact, many interesting applications of MOEA to the combinatorial optimisation problem domain exist.

2 Multiple Objective Optimisation

MOO has been defined as the "simultaneous optimisation of multiple objective functions" (Deb 2002). To offer a practical example, let us consider the simple task of commuting between one's home and workplace. It is often desirable to minimise monetary cost (fuel use, bus fare, train fare, etc.) while also minimising travel time. Increasing comfort and other factors such as safety and environmental impact may also be of concern. As with this simple example, it would be reasonable to surmise that for many situations there is no simple feasible solution which simultaneously optimises all objectives; often a trade-off or compromise solution must be selected. In fact, many candidate solutions may not even be feasible due to imposed constraints. For this simple example a finite set of feasible solutions will most likely exist; however, in many real-world and artificial multiple objective problem instances the number of possible solutions can be infinite or, if finite, generally intractable.

MOO is, in general, approached in three ways in the literature with respect to the *decision maker* (Hwang and Masud 1979). These different approaches may be relevant to researchers when developing MOACO methods that consider the use of information to guide search.

– *A Priori Preference Articulation*: Combines all objectives into a single objective through the use of a weight vector and aggregation function, turning the multi-objective problem into a single-objective problem to search.
– *Progressive Preference Articulation*: May have partial preference information, and this preference information is adjusted as the search continues by interpreting the results of the search.
– *A Posteriori Preference Articulation*: No preference information, instead the decision maker is presented with a set of candidate solutions (generated by some search process) to choose from.

2.1 Objective and solution space

Solution information is defined as information which refers to a solution encoding such as a set permutation in the case of the Travelling Salesman Problem (TSP) or an input vector as in continuous function optimisation (CFO). *Objective information* (or fitness) is problem specific, functional information such as the tour length in a TSP or the objective function value in CFO.

For single objective optimisation problems, the *solution information* is usually mapped onto an n-dimensional solution space (or decision variable space), with the *objective information* mapped directly from this, creating what some call a *fitness landscape*. For such domains there is an $n \rightarrow 1$ mapping between the n decision variables and the objective space. For MOO the mapping tends to be $n \rightarrow m$, where m is the number of objectives. Generally, solutions that are close in the solution space are not necessarily close in the objective space. Due to this lack of continuity, it must always be clear in which space we are applying operators such as diversity maintenance techniques.

In MOO problems, the solution space is in most cases identical to a similar single objective optimisation problem. As an example of this, the Multiple Objective TSP (MOTSP) is comprised of multiple distance matrices of the same dimensions and a single solution (a set permutation) is evaluated multiple times against each individual distance matrix. To interpret this in a real-world context, consider that each distance matrix corresponds to different types of cost between cities, e.g. time, distance, monetary value.

2.2 Dominance and Pareto optimality

An MOO problem, by definition, contains two or more dependent or independent objective functions. For the purposes of discussion here, dependent objective are those that share solution variables (e.g. $f_1(x)$, $f_2(x)$), whereas independent objectives do not share solution variables (e.g. $f_1(x)$, $f_2(y)$). In the case of dependent objectives, Purshouse and Fleming (2003) define objectives as being in *harmony* or *conflict*. Conflicting objectives will be the case where increasing the quality of one objective tends to simultaneously decrease the quality of another objective. In such instances, rather than a single optimal solution existing for all objectives, multiple trade-off solutions may exist.

In MOO, solutions are related according to the concept of *dominance* which provides MOO practitioners a means by which multiple solutions can be compared and subsequently ranked or sorted. A standard dominance measure compares two solutions (s_1, s_2) and determines which of the following conditions and definitions apply:

- If s_1 is better in all objectives than s_2, s_1 is said to *strongly dominate* s_2, denoted as $s_1 \succ\succ s_2$.
- If s_1 is not worse than s_2 in all objectives and better in at least one objective, s_1 is said to *dominate* s_2, denoted as $s_1 \succ s_2$.
- If s_1 is not worse than s_2 in all objectives, s_1 is said to *weakly dominate* s_2, denoted as $s_1 \succeq s_2$.
- If s_1 does not weakly dominate s_2, nor s_1 weakly dominate s_2, s_1 and s_2 are incomparable, denoted as $s_1||s_2$.

A *non-dominated set* is a set of solutions that are not weakly dominated by any other solution in the set. The *Pareto optimal* set, named after the economist Vilfredo Pareto, is a set of optimal non-dominated solutions. For a given problem no feasible solution exists that dominates a Pareto optimal solution. All solutions that are Pareto optimal belong to the Pareto set, while the points that these solutions map to in the objective space is the Pareto front. The number of Pareto optimal solutions making up the Pareto set, and the shape of the Pareto front is dependent on the specific problem instance. The Pareto set may be infinite, as it is the case with many Multiple Objective Function Optimisation problems.

Several standard shape descriptors are defined in the literature for specific Pareto fronts (Deb 2002) and three of these are illustrated in Fig. 1.

The principles of dominance provide a useful method to compare solutions. The simple dominance relationship can be extended for search algorithms that rely on directional
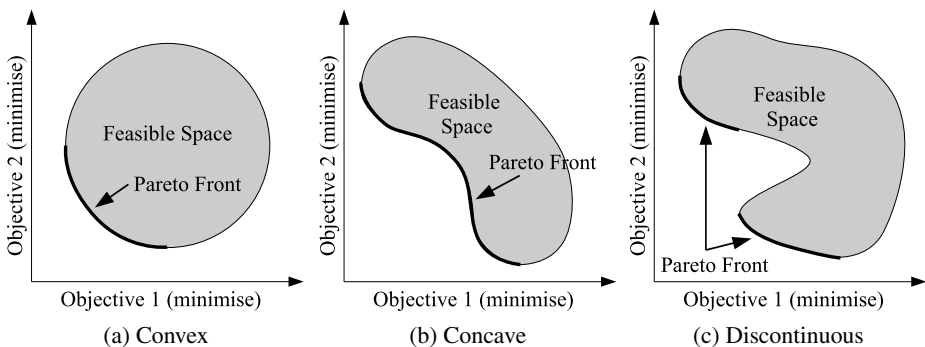
Fig. 1 Representations of different Pareto fronts plotted in the objective space

(fitness) information to guide their search process such as in the Non-dominated Sorting Genetic Algorithm (see Sect. 4.4).

2.3 Solving Multiple Objective Optimisation problems

It was mentioned earlier that in the literature MOO problems are, in general, approached in three ways: a priori, progressive and a posteriori. Knowledge of the Pareto front, or lack thereof, may aid in the choice of the technique to use as in some instances it has been demonstrated that an a priori weighting can preclude a solver from returning a solution which is Pareto optimal (Deb 2002). Furthermore, an even distribution of solutions across a Pareto front is not guaranteed with evenly spaced weight vectors, such as in the case of a convex Pareto front (Das and Dennis 1996).

In the absence of sufficient knowledge of the Pareto front, a progressive or a posteriori approach can be useful since it may provide insight into the shape and nature of the objective space. These approaches often strive to meet two goals:

1. Find solutions that are close to the Pareto front.
2. Maintain a coverage of solutions along the entire Pareto front.

These goals do not necessarily conflict, in the sense that a complete solution may exist that can satisfy both goals maximally, that is, the Pareto set may contain many diverse solutions. From an optimisation perspective though, these goals are often treated as conflicting since most search algorithms balance *exploratory* (diversity preserving) behaviour with *exploitative* (quality enhancing) behaviour.

2.4 Performance metrics

Even though performance metrics are not used in this paper, it is useful to talk briefly about them to give a complete picture of MOO and its difficulties. It should be clear that the use of performance metrics is important in stochastic algorithm design and validation, given that they reflect different facets of algorithm performance. For single objective problems, performance metrics are usually straightforward since algorithms applied to these problems may return a single best solution, and this leads to the development of a univariate distribution of results when run multiple times using different random conditions. This then means that a wealth of parametric and non-parametric statistical techniques are available to compare different experimental trials.

MOO can present a further challenge to any performance analysis process since MOO algorithms may return multiple Pareto and sub-Pareto optimal solutions rather than a single best solution. A non-trivial exercise is the development of *unary* quality metrics that, when compared, can allow the algorithm designer to make a reliable comparison between different algorithms. In Knowles et al. (2006) and Zitzler et al. (2003), authors offer a detailed analysis of many existing unary metrics that measure the distribution of solutions and their closeness to the actual Pareto front. The metrics include *Generational Distance* (Van Veldhuizen 1999), *Spacing Metric* (Deb et al. 2000), *Maximum Pareto Front Error* (Van Veldhuizen 1999) and *Extent of Approximation Set* (Deb 2002). In their analysis, the authors of Knowles et al. (2006), Zitzler et al. (2003) challenge many assumptions about the reliability of such metrics by providing case studies where these unary indicators give false or misleading assessments as to the quality of different distributions. One important conclusion is that the combination of these existing metrics is often thought to provide more reliable information, in a 'strength in numbers' sense. However, this is typically not the

case; the combination of multiple inaccurate metrics does not increase their accuracy. In their conclusions, it is argued that just one good quality unary metric would be better than multiple bad metrics.

Other, more widely accepted metrics include the *Hypervolume* (Zitzler et al. 2007) and *Epsilon* (Zitzler et al. 2003) metrics. The Hypervolume metric provides a measurement of the volume encapsulated by a reference point z and the non-dominated front. When the same z is used on multiple non-dominated sets, the set with the largest Hypervolume is the best, thus the hypervolume favours non-dominated sets that maximise the enclosed volume. The Epsilon metric provides a measure of how much modification is required to manipulate a non-dominated set to be not weakly dominated by another reference set such as the Pareto set. Consequently, the set that requires the least modification is deemed better.

Another alternative comparison method is the *Summary Attainment Surface* (Fonseca and Fleming 1996) which is in some ways inspired by a traditional univariate distribution. In traditional univariate distributions, single values represent the mean, minimum, maximum and other similar important features. With summary attainment surfaces the same features are represented. However, instead of a single value being used, a multi-dimensional surface (with the same number of dimensions as the number of objectives) is used. Once obtained, summary attainment surfaces that describe different algorithms' mean performance on a single problem can be quantitatively compared using non-parametric statistical methods like Mann-Whitney and Kruskal-Wallis rank sum tests. The summary attainment surface plotting method (Knowles 2005) can also be used to visualise and thus qualitatively compare multiple summary attainment surfaces.[2] An advantage of the approach is that it eliminates the requirement for specialised knowledge of the problem domain (such as a priori knowledge of the location of Pareto optimal solutions).

Other performance metrics tend to require knowledge of the Pareto front to obtain difference measurements. A concern is that this limits the kind of problems that can be measured since an exact solver must be used to ensure that a set of Pareto optimal solutions from across the entire Pareto front are obtained. If an approximate set is used instead, it could mean that different baselines are obtained by different researchers, thus skewing results across multiple studies.

## 3 Multiple Objective ACO taxonomy

The MOACO taxonomy presented here loosely follows the ACO Metaheuristic Framework (Dorigo and Di Caro 1999; Dorigo et al. 1999; Maniezzo and Carbonaro 1999) and as such the basic elements of the ACO Metaheuristic Framework are briefly described in Sect. 3.1. Following this, each component of the proposed MOACO taxonomy is described in terms of its possible values, and where relevant, its relationship to the ACO Metaheuristic Framework. As a reference guide, each component of the MOACO taxonomy is listed in Table 1 alongside possible values for these components.

3.1 The ACO Metaheuristic Framework

The ACO Metaheuristic Framework describes the scheduling of several processes and is represented in Fig. 2. These processes are:

---

[2]A useful *summary attainment surface* calculation utility is available from http://dbkgroup.org/knowles/plot_attainments/.
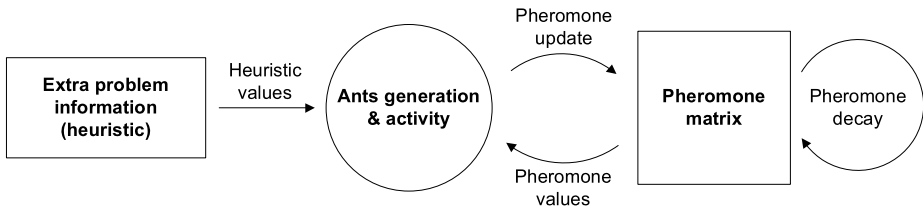
Fig. 2 Process organisation of the Ant Colony Optimisation Metaheuristic Framework. *Daemon actions* are not listed as they are generally external to the general cyclic process and if used can affect one or many of the processes listed

Table 1 Proposed MOACO taxonomy components and values

Component	Values
Pheromone matrix	Multiple, Single
Solution construction	Targeted, Dynamic, Fixed
Evaluation	Pareto, Non-Pareto
Update and decay	Individual, Global
Pareto archival	Offline, Online, Elite, None

– *Ants generation & activity*: This process is responsible for the construction of new solutions. This is achieved using probabilistic stepwise solution construction. The probability of a particular solution component being added to a growing solution is based on a combination of problem specific (heuristic) information and learned (pheromone) information of how well this component is used in past solutions. The exact combination of this information and the greediness of the selection mechanism are important implementation specific details.
– *Pheromone trail update & decay*: Once solutions have been evaluated, they can influence the pheromone matrix through a pheromone update process. To allow the replacement of old information with new information, a pheromone decay process is also employed that removes the influence of past solutions over multiple successive algorithm cycles.
– *Daemon actions*: Any actions which do not fit into the regular cyclic processes of solution generation, evaluation, update and decay are called Daemon actions. An example of such an action is the storage of an elite solution.

3.2 Pheromone matrix

In MOACO, the pheromone matrix[3] is chosen between two standard models:

– *Multiple*: Many pheromone matrices, usually one per objective. Each pheromone matrix may contain different pheromone values depending on the specific implementation. If a one-to-one pheromone to objective mapping is used then ideally the pheromone information contained in a single matrix will reflect which solution components advantage a particular objective.
– *Single*: A single pheromone matrix, much like that used in traditional single objective ACO algorithms, where each decision variable corresponds to one pheromone value.

---

[3]Note that while we use the term matrix to discuss the pheromone model, pheromone models are not limited to use a matrix representation.

3.3 Solution construction

Solution construction (*ants generation & activity*) can be biased (in the usual ACO manner) by heuristic and pheromone values which are combined using a specific weighting rule. However, what distinguishes the solution construction component type in the taxonomy presented is the specific combination of heuristic and pheromone information. These combinations are:

– *Targeted*: Information (pheromone or heuristic) pertaining to only a specific objective is used in the solution construction process.
– *Dynamic*: Information that spans more than one objective, such as multiple different heuristics or pheromone matrices, is used in the solution construction process. The combination of objective specific information is dynamic so that different objectives can be emphasised at different times during the solution construction or specific ants can combine objective information using specific weightings.
– *Fixed*: The same as dynamic; however, the combination of objective specific information is fixed a priori, usually based on domain specific knowledge. This means that pheromone and heuristic information are combined in the same proportions for the entire algorithm execution.

3.4 Evaluation

Solution evaluation and ranking in MOO has been addressed by the Evolutionary Computation community using many different methods including non-dominated sorting and Pareto ranking (Corne et al. 2001; Deb et al. 2000; Horn et al. 1994; Knowles and Corne 2000; Srinivas and Deb 1994; Zitzler et al. 2002). This guides the distinction in the proposed taxonomy of solution evaluation methods:

– *Pareto*: Solutions are evaluated for all objectives and are assigned a score that reflects how well the solution satisfies all of these objectives in a Pareto sense, such as a dominance ranking.
– *Non-Pareto*: Solutions are evaluated for one or many objectives but are assigned a score that reflects how well the solution satisfies one of these objectives, or a score based on a weighted sum of all of these objective values.

3.5 Update and decay

The *pheromone trail update & decay* method determines what information is used in the update process and what the update affects. If a single pheromone matrix is used then the algorithm designer is left only one choice, to use new solutions to update the single pheromone matrix. If multiple pheromone matrices are used:

– *Individual update*: New solutions target a specific pheromone matrix for update.
– *Global update*: New solutions can update many or all pheromone matrices.

3.6 Pareto archival

If a progressive or a posteriori approach is taken then most likely multiple Pareto optimal solutions are sought. This then raises questions about how these Pareto (or near Pareto) solutions are to be stored as the search process progresses. In the MOACO literature, this *daemon action* is addressed in four ways:

– *Offline storage*: After new solutions have been evaluated, they are used to update pheromone information in one or more pheromone matrices, depending on the specific implementation. Solutions are then added to an external, off-line storage repository (often called archive) and this population is sorted to remove any non-Pareto optimal solutions. This external population is not used for future solution construction, just as a historic record of good solutions found to date. At the end of the algorithm execution, the external population is returned as the final solutions. This is also referred to as a *hall of fame* in Evolutionary Computation literature.

– *Online storage*: After new solutions have been evaluated, they are added to a population of solutions. The population replacement policy is implementation specific and pheromone information is linked to the population so that the pheromone matrix or matrices will reflect the state of the population at any time. It is similar to the off-line storage strategy; however, there is a tighter coupling between the population and the pheromone information. At the end of the algorithm execution the current population is returned as the final solutions.

– *Elite*: Only a single elite solution is maintained. This solution may be used for pheromone update at any time or simply as a best-so-far solution to be returned at the completion of the algorithm execution.

– *No storage*: After updating the pheromone information, new solutions are discarded except when algorithm execution is finished, in which case the most recent solutions are returned as the final solutions.

4 Multiple Objective ACO discussion and analysis

A large selection of MOACO algorithms are listed in Table 2.[4] The algorithms are discussed in the forthcoming subsections according to their relationship to specific taxonomy features. Several groupings are evident in Table 2 and as such the discussion focuses on these groupings rather than restating basic algorithm details. The discussion also focuses on how these design choices relate to algorithm performance. MOACO specific design issues, such as the choice of single or multiple pheromone matrices and pheromone update and decay, are discussed with a focus on how these choices affect algorithm performance.

4.1 A priori preference articulation algorithms

A priori preference articulation is a MOO problem solving strategy. This particular strategy is useful when a problem solver is aware of the exact weighted combination of objectives that is required to satisfy a decision maker. This information can be in the form of a lexicographic ordering of the objectives or a percentage contribution per objective to an aggregate score. In these cases, it may be a waste of resources to expend effort in locating multiple Pareto optimal solutions from the entire Pareto front since many will be simply discarded by the decision maker.

---

[4]Shelokar et al. (2002) proposed an ant-inspired approach for multiple objective function optimisation problems; however, since this approach was based on the Ant Colony Metaphor for Continuous Design Spaces (Bilchev and Parmee 1995), it is strictly not an ACO algorithm per se (Socha and Dorigo 2008). While the algorithm was demonstrated as a good approach for the problems tested, it resembles rather closely to a Genetic Algorithm since it uses crossover and mutation to generate new solutions rather than the traditional ACO probabilistic stepwise construction. For these reasons it has not been included in Table 2.

Table 2 Multiple objective ACO algorithms listed according to the proposed taxonomy

Algorithm	Pheromone matrix	Update & decay[a]	Solution construction	Evaluation	Pareto archival
CPACO (Angus 2007)	Single	–	Dynamic	Pareto	Online
MACS (Barán and Schaerer 2003)	Single	–	Dynamic	Non-Pareto	Online
MOAQ (Romero and Manzanares 1999)	Single	–	Fixed	Pareto	Offline
MOACOM (Gravel et al. 2002)	Single	–	Fixed	Non-Pareto	Elite
ACOAMO (McMullen 2001)	Single	–	Fixed	Non-Pareto	Elite
SACO (T'kindt et al. 2002)	Single	–	Fixed	Non-Pareto	Elite
MACS-VRP (Gambardella et al. 1999)	Multiple	Global	Targeted	Non-Pareto	Elite
COMPETants (Doerner et al. 2003)	Multiple	Individual	Targeted	Non-Pareto	None
PACO-MO (Guntsch and Middendorf 2003)	Multiple	Global	Dynamic	Pareto	Online
BicriterionAnt (Iredi et al. 2001)[b]	Multiple	Global	Dynamic	Pareto	Offline
ParetoACO (Doerner et al. 2004)	Multiple	Global	Dynamic	Non-Pareto	Offline
MONACO (Cardoso et al. 2003)	Multiple	Global	Dynamic	Non-Pareto	None
ACO-bQAP (López-Ibáñez et al. 2004)[c]	Single, Multiple	Individual	Dynamic	Pareto, Non-Pareto	Offline, None

[a]Those algorithms listed with a dash do not require a specific update & decay classification as they only use a single pheromone matrix

[b]Although several different algorithms are discussed in this reference, only the most thoroughly analysed algorithm is included here

[c]Several combinations of algorithm components are discussed in this reference

Four algorithms identified in the literature (MOACOM, ACOAMO, SACO, and MACS-VRP) use a priori preference articulation in solving their particular problems. These approaches differ by how they combine the particular objective information, yet, whether implicitly or explicitly, they all weight their chosen problems' multiple objectives in some kind of preferential order. All algorithms indicated above use elite solution storage, and hence do not maintain populations of non-dominated solutions. As such, all of these algorithms save on computation costs[5] associated with maintaining a non-dominated solution set. The algorithms use non-Pareto solution evaluation methods, which are appropriate given that the multiple objectives are already reduced to a single score for solution comparison purposes.

An observation of these algorithms is that the particular problem domains they address tend to have a single dominant objective. For example, the Vehicle Routing Problem (VRP) by MACS-VRP aims to reduce the total number of tours and the total travel time but in all test cases one of these objectives tends to be given a preference over the other. Another observation is, that in most cases, the objectives of the problems addressed are not strictly conflicting. This seems to suggest that if a MOO problem has a single dominant objective with secondary (less critical) objectives then a priori preference articulation methods may be best suited. As each of these algorithms is reported to perform well on its particular problem, this suggests that non-Pareto solution evaluation, fixed solution construction and

---

[5]This computation cost being incurred through the checking of new solutions against the existing non-dominated set. A worst case complexity of such an operation is $O(n \cdot m \cdot d)$, where n is the size of the non-dominated set, m the number of objectives, and d the number of new solutions.

elite solution storage work well for these kinds of problem. By extension this suggests that alternative Pareto-based MOACO may be a poor fit for MOO problems where we may wish to target a specific area of the Pareto front, or for MOO problems with a single dominant objective.

4.2 Multiple versus single pheromone matrices

A pheromone matrix is the memory structure that allows historic (acquired) information to be stored so that it can be used to bias future solution construction operations. In general, MOACO algorithms use one pheromone matrix or a number of pheromone matrices that is proportional to the number of objectives, usually one per objective. The essential differences are that those algorithms that use *multiple* pheromone matrices are able to keep objective specific history information completely partitioned, whereas those that use a *single* pheromone matrix must combine this information.

The choice of the pheromone model can depend on other design decisions such as how the solution construction process uses pheromone information and how pheromone matrices are updated and decayed. As an example, the a priori preference articulation MOACO algorithms (discussed in Sect. 4.1) are designed to converge to one specific area of the Pareto front. Because of this convergence characteristic these algorithms do not require diverse, and possibly objective specific, information to be captured, except in the case of MACS-VRP. These algorithms use *non-Pareto* based solution evaluation methods and update and decay pheromone much like classical single objective ACO algorithms. For these reasons the use of a *single* pheromone matrix seems to be well justified.

Other MOACO that use a *single* pheromone matrix include CPACO and MACS. These algorithms, unlike the a priori algorithms, aim to find complete coverage of the Pareto front. Instead of relying on multiple pheromone matrices to guide objective specific solution construction, both algorithms use multiple heuristic matrices (one per objective) which are weighted in a specific way for each ant to bias solution construction toward different objective trade-offs. In other words, these algorithms achieve diversity across the Pareto front through the use of heuristics rather than pheromone. The reduction in memory required associated with using a *single* pheromone matrix versus *multiple* pheromone matrices is also cited as a motivating factor in the design of CPACO (Angus 2007). However, the actual problem size would have to be sufficiently large for this to be of concern.

To make positive use of the extra memory required by *multiple* pheromone matrices, they must contain different information by being updated or decayed non-uniformly or by being mapped differently during the pheromone update and/or solution construction process. As an example, BiCriterionAnt uses *multiple* pheromone matrices that are updated and decayed using the same objective quality values (see Sect. 4.4), but are mapped differently by the pheromone update and solution construction processes. MONACO makes the objective specific pheromone information different by applying different evaporation rates to its *multiple* pheromone matrices.

A number of MOACO algorithms are best described as multi-colony algorithms and they use *multiple* pheromone matrices. For example, COMPETants uses a separate colony for each objective, with each colony maintaining a separate pheromone matrix. A drawback of this approach is that by allowing multiple colonies to operate independently (one per objective) the algorithm would most likely focus the search on the extremes of the Pareto front, neglecting the 50/50 trade-off point. To counter this issue, the COMPETants algorithm adds agents called 'spies' on occasion to combine information from the colonies in an attempt to find trade-off points around the midpoint of the Pareto front. Another example of this is

with the MACS-VRP algorithm, which is a multi-colony ACS variant that was applied to a multiple objective vehicle routing problem. Like COMPETants, the MACS-VRP algorithm optimises each objective independently in separate colonies (one per objective) and shares solutions between these colonies for pheromone update purposes. Another aspect that is worth considering is the effect of the number of colonies used in a multi-colony MOACO. In López-Ibáñez et al. (2004), the authors, through experimental analysis, discuss how more colonies correspond to a higher exploitation level.

We already mentioned the associated reduction in memory required when using a *single* pheromone matrix instead of *multiple* pheromone matrices (Angus 2007). The advantage of *multiple* pheromone matrices is that they allow information about specific objectives to be kept separate and only recombined at solution construction, allowing *dynamic* or *targeted* solution construction with *multiple* pheromone information. In contrast, it would be of limited utility to use *multiple* pheromone matrices and use a *fixed* solution construction strategy as it would be better to simply combine this pheromone information prior to solution construction, thus removing the requirement for *multiple* pheromone matrices.

Consider that the pheromone matrix is to ACO what the population is to an Evolutionary Algorithm and that many state-of-the-art MOEA maintain a single population of solutions rather than multiple populations. Given this, it is worth considering why there are many examples of *multiple* pheromone matrices used by MOACO algorithms. This may be due to the way by which solution information is stored in the pheromone matrix. Usually, a pheromone matrix is comprised of a two dimensional matrix of floating point numbers, whereas a population contains many independent solutions and their associated objective values. If a MOACO uses a *single* pheromone matrix, it must combine all objective function values from a single solution into some aggregate score, insert this information into a pheromone matrix via pheromone update, and then discard the actual solution. Once solution information is added to a pheromone matrix, it cannot be reliably extracted again, which means that if the aggregation function was changed the pheromone matrix could not be reconstructed to reflect this change without the original solutions (which in most cases are not retained). If however *multiple* pheromone matrices are used, no aggregation is performed before pheromone update, thus solution construction can alter the aggregation of historic pheromone information in a dynamic way. An alternative to this is to use population-based ACO approaches, which allow for the recreation of pheromone, or *temporary pheromone matrices*.

As far as applications are concerned, all of the MOACO algorithms (except for MACS-VRP) that use multiple pheromone matrices tend to be designed for problems where little is known about the Pareto front, or about what type of solutions are required by a decision maker. In other words, these algorithms are more exploratory than the a priori preference articulation algorithms of Sect. 4.1 which would tend to target their search towards specific areas of the Pareto front.

4.3 Solution construction

Solution construction in ACO is characterised by the use of probabilistic stepwise solution construction. In most single objective cases, these probabilities are calculated using a weighted combination of pheromone and heuristic information, each derived from single matrices. In the MOACO case, there may be multiple pheromone and heuristic matrices, thus there are many possible ways to combine pheromone and heuristic information for the purpose of biasing solution construction. By altering the weightings of objective specific pheromone and heuristic information, an individual ant can be directed to solve specific areas of the Pareto front. Such weightings can be easily made ant specific so that multiple ants

each combine this information in a unique way, maybe to try to spread a 'colony' across the entire Pareto front; this is denoted as *dynamic* solution construction.

The CPACO, MACS, PACO-MO, BiCriterionAnt, ParetoACO, and MONACO algorithms all use *dynamic* solution construction. This solution construction technique seems to be of benefit to problems where not much is known of the specific shape of the Pareto front, or where there is no a priori weighting to guide the search process. In all of the aforementioned algorithms, the solution construction process uses an expanded form of the ACO *random proportional rule*. The specific transition rule used depends on the implementation, as each algorithm uses a specific combination of single and multiple heuristics and pheromone matrices, although most tend to give each ant its own unique weighting to combine the pheromone and heuristic information. A generalised version of the solution component probability determination with multiple pheromone and heuristic matrices is outlined in (1). In this equation, i is the current solution component (e.g. city in a TSP), j is all solution components (e.g. all cities in a TSP), k is the ant index, L is the number of pheromone matrices, τ_{ij}^l is the pheromone trail of the l-th pheromone matrix associated to the choice of component j after component i, M is the number of heuristics, η_{ij}^m is the heuristic information, derived from the m-th heuristic, on the choice of component j after component i, and N_i^k is the feasible component space from i. These algorithms all tend to achieve an even distribution of solutions across the Pareto front (Garcìa-Martínez et al. 2007), which is a desirable feature for an a posteriori preference articulated MOO algorithm.

$$
p_{ij}^k = \begin{cases} \dfrac{\prod_{l=1}^L (\tau_{ij}^l)^{\alpha_l} \cdot \prod_{m=1}^M (\eta_{ij}^m)^{\beta_m}}{\sum_{q \in N_i^k} \prod_{l=1}^L (\tau_{iq}^l)^{\alpha_l} \cdot \prod_{m=1}^M (\eta_{iq}^m)^{\beta_m}}, & \text{if } j \in N_i^k, \\ 0, & \text{otherwise.} \end{cases} \tag{1}
$$

It is possible to completely segregate objective information for the purpose of solution construction in the so-called *targeted* solution construction. In such cases, solution construction operates using only pheromone and heuristic information pertaining to a single objective. However, once these solutions are evaluated, they can influence multiple objectives' pheromone information if desired, as pheromone update is separate from solution construction. COMPETants and MACS-VRP both use *targeted* solution construction and are also examples of multi-colony ACO algorithms. These algorithms treat objectives as separable entities that are optimised by a localised colony. All solutions generated are based solely on pheromone and heuristic information of individual colonies. Even though these algorithms allow occasional sharing of solutions between colonies to encourage solutions that trade-off between these objectives, they still focus a majority of the search effort toward the extremes of the Pareto front.

Section 4.1 commented on a subset of MOACO algorithms, a priori algorithms, that use a *fixed* solution construction strategy. In such cases, some domain knowledge dictates, ahead of algorithm execution, the exact combination of heuristic and pheromone information to use. For these cases, this is a reasonable approach; if a specific objective trade-off is required, it makes sense to combine objective information in fixed proportions to concentrate search effort on a specific section of the Pareto front.

4.4 Pareto ranking

Used extensively in the MOEA literature, Pareto evaluation and ranking techniques deal with the assignment of quality to solutions in multiple objective domains. Pareto ranking is popular due to many problems associated with objective value aggregation. Objective value aggregation may introduce objective bias; can be difficult if objectives are incommensurable or if the range of objective values is unknown; and cannot be performed if preference

information is unavailable. If desired, Pareto ranking techniques can also allow multiple non-dominated solutions to be distinguished based on different factors such as how much diversity they add to a population of non-dominated solutions.

Of the MOACO algorithms listed in Table 2, MOAQ, CPACO, PACO-MO, and BiCriterionAnt use forms of Pareto ranking for solution evaluation purposes. These algorithms refrain from combining objective values together to form an aggregate score as is the case with some other non-Pareto MOACO. Of these algorithms all but CPACO use and/or keep only non-dominated solutions. This is much like many single objective ACO algorithms which tend to strongly bias elite solutions. A possible concern is that this may lead to a loss of diversity within the algorithms pheromone matrix(ces) and thus prevent an algorithm from generating an even coverage of the Pareto front. Such diversity is important given that these algorithms tend to be applied to MOO problems where the multiple objectives are equally important and as such multiple diverse Pareto optimal solutions are sought.

To abate loss of diversity, these MOACO algorithms could be modified to allow all created solutions to update pheromone, but still encourage exploitation by reducing the quality of dominated solutions using Pareto ranking techniques from existing MOEA. As an example, MOEA algorithms such as the Indicator Based Evolutionary Algorithm (IBEA) (Zitzler and Künzli 2004) measure the contribution of solutions not just to the non-dominated front but to the overall diversity they add to the population. Another alternative is to use dominance ranking according to a non-dominated sorting technique such as that of the NSGA-II algorithm (Deb et al. 2000). This ranking scheme is used by CPACO which has been shown to produce an even coverage of the Pareto front (Angus 2007).

4.5 Pareto archival

Pareto archival is the method by which multiple Pareto optimal solutions are stored for post algorithm run-time analysis or use. Apart from being useful as the actual solutions to a problem under analysis, these solutions may also be used by an algorithm during its execution. If this is the case, under the MOACO taxonomy such a strategy is considered to be an *online* Pareto archival strategy. A further implication of the use of a Pareto archival strategy is that these algorithms can be considered to be Population-based ACO (PACO) algorithms (Guntsch 2004), which are a specific subset of ACO algorithms. Examples of MOACO algorithms that use an *online* Pareto archival strategy include PACO-MO, CPACO and MACS.

Even though the MACS algorithm is not purported to be a PACO algorithm, it still contains much of the functionality of a PACO algorithm. MACS repeats the following steps:

1. Create new solutions using a single temporary pheromone matrix.
2. Evaluate new solutions and add all new solutions to a population of non-dominated solutions.
3. Re-check the population for dominated solutions and if any are found then discard them.
4. Create a new temporary pheromone matrix using the solutions in the non-dominated population.

This makes MACS a close approximation to the PACO-MO and CPACO algorithms, except that the PACO-MO algorithm uses a subset of the non-dominated population to create its temporary pheromone matrix and CPACO allows storage of dominated solutions. Perhaps the most important observation to be made of these Multiple Objective PACO algorithms is about their similarity to existing MOEA algorithms given that they all use *online* storage which is very similar to MOEA solution population structures. The CPACO algorithm is

perhaps the most similar to MOEA given that it uses the same solution evaluation strategy as the NSGA-II algorithm.

As a separate issue, perhaps overlooked by most of the algorithms that use *online* or *offline* Pareto archival, is the cost of maintaining a non-dominated solution set. The computational complexity of maintaining these sets is an issue that has been discussed at length in the MOEA community (Deb 2002, p. 33, Zitzler et al. 2002, 2007). However, such issues are, in general, not treated in the MOACO literature. Such an issue is important since simply continuously adding solutions to *online* or *offline* Pareto archives can result in a considerable computational overhead as the archive becomes large. In one sense, most MOACO algorithms may assume that the number of non-dominated solutions in the population will remain small; if so, then this issue is of little relevance. However, the use of archival methods must be considered if these algorithms are applied to other problem domains where there is scope for the archive to become quite large.

5 Conclusion

This paper discussed a number of existing MOO concepts such as *Pareto optimality*, *dominance* and *performance metrics*. The paper has also discussed existing ideas pertaining to the methods by which MOO can be solved. An original MOACO taxonomy was introduced that allows for easy identification of similar MOACO algorithms. Some of these classifications have been identified and commented on, while others have been left for identification and discussion in future work. It is hoped that the classifications identified may allow for fairer comparisons of MOACO algorithms, since algorithms can now be identified based on their optimisation goals, and similar taxonomic features.

There is a diverse mixture of existing MOACO algorithms available and it is hoped that the taxonomy presented will assist researchers new to MOACO to gain a good understanding of the different design choices that have been explored. While encouraging the use of this taxonomy in the development of new MOACO, we are not suggesting that this taxonomy be thought of as the only way to design a MOACO. While considerable effort has been made to ensure that the taxonomy encompasses existing MOACO, it is expected that future developments may fall outside the defined boundaries. The authors encourage both future refinements and additions to the taxonomy.

References

Angus, D. (2007). Crowding population-based ant colony optimisation for the multi-objective travelling salesman problem. In *2007 IEEE symposium on computational intelligence in multi-criteria decision-making (MCDM 2007)* (pp. 333–340). New York: IEEE Press.

Barán, B., & Schaerer, M. (2003). 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). Calgary: ACTA Press.

Bilchev, G., & Parmee, I. C. (1995). The ant colony metaphor for searching continuous design spaces. In T. C. Fogarty (Ed.), *LNCS: Vol. 993. Proceedings of the AISB workshop on evolutionary computation* (pp. 25–39). Berlin: Springer.

Cardoso, P., Jesus, M., & Márquez, A. (2003). MONACO—multi-objective network optimisation based on ACO. In F. S. Leal & D. Orden (Eds.), *Encuentros de geometría computacional*. Santander: Universidad de Cantabria.

Corne, D. W., Jerram, N. R., Knowles, J. D., & Oates, M. J. (2001). PESA-II: Region-based selection in evolutionary multiobjective optimization. In L. Spector, E. D. Goodman, A. Wu, W. Langdon, H. M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, & E. Burke (Eds.), *Proceedings of the genetic and evolutionary computation conference (GECCO'2001)* (pp. 283–290). San Mateo: Morgan Kaufmann.

Das, I., & Dennis, J. (1996). *A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems* (Technical Report 96–36). Houston: Rice University, Dept. Of Computational and Applied Mathematics.

Deb, K. (2002). *Wiley-Interscience series in systems and optimization. Multi-objective optimization using evolutionary algorithms* (2nd ed.). New York: Wiley.

Deb, K., Agrawal, S., Pratap, A., & Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, & H. P. Schwefel (Eds.), *LNCS: Vol. 1917. Parallel problem solving from nature (PPSN VI)* (pp. 849–858). Berlin: Springer.

Doerner, K., Hartl, R., & Teimann, M. (2003). Are COMPETants more competent for problem solving? The case of full truckload transportation. *Central European Journal of Operations Research (CEJOR)*, *11*(2), 115–141.

Doerner, K., Gutjahr, W. J., Hartl, R. F., Strauss, C., & Stummer, C. (2004). Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection. *Annals of Operations Research*, *131*(14), 79–99.

Dorigo, M., & Di Caro, G. (1999). The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New ideas in optimisation* (pp. 11–32). London: McGraw-Hill.

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

Dorigo, M., & Stützle, T. (2004). *Ant colony optimization*. Cambridge: MIT Press.

Dorigo, M., Di Caro, G., & Gambardella, L. M. (1999). Ant algorithms for discrete optimization. *Artificial Life*, *5*, 137–172.

Fonseca, C. M., & Fleming, P. J. (1996). On the performance assessment and comparison of stochastic multiobjective optimizers. In H. M. Voigt, W. Ebeling, I. Rechenberg, & H. P. Schwefel (Eds.), *LNCS: Vol. 1141. Proceedings of the 4th international conference on parallel problem solving from nature (PPSN IV)* (pp. 584–593). Berlin: Springer.

Gambardella, L. M., Taillard, E., & Agazzi, G. (1999). MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. In D. Corne, M. Dorigo, & F. Glover (Eds.), *New ideas in optimisation* (pp. 63–76). London: McGraw-Hill.

García-Martínez, C., Cordón, O., & Herrera, F. (2007). A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for bi-criteria TSP. *European Journal of Operational Research*, *180*(1), 116–148.

Gravel, M., Price, W. L., & Gagné, C. (2002). Scheduling continuous casting of aluminum using a multiple-objective ant colony optimization metaheuristic. *European Journal of Operations Research*, *143*(1), 218–229.

Guntsch, M. (2004). *Ant algorithms in stochastic and multi-criteria environments*. Ph.D. thesis, Universität Fridericiana zu Karlsruhe, Germany.

Guntsch, M., & Middendorf, M. (2003). Solving multi-criteria optimization problems with population-based ACO. In G. Goos, J. Hartmanis, & J. van Leeuwen (Eds.), *LNCS: Vol. 2632. Proceedings of the second international conference on evolutionary multi-criterion optimization (EMO 2003)* (pp. 464–478). Berlin: Springer.

Horn, J., Nafpliotis, N., & Goldberg, D. E. (1994). A niched Pareto genetic algorithm for multiobjective optimization. In *IEEE world congress on computational intelligence: Vol. 1. Proceedings of the first IEEE conference on evolutionary computation* (pp. 82–87). New York: IEEE Press.

Hwang, C. L., & Masud, A. S. M. (1979). *Lecture notes in economics and mathematical systems*: *Vol. 164. Multiple objective decision making, methods and applications: a state-of-the-art survey*. Heidelberg: Springer.

Iredi, S., Merkle, D., & Middendorf, M. (2001). Bi-criterion optimization with multi colony ant algorithms. In E. Zitzler, K. Deb, L. Thiele, C. C. Coello, & D. Corne (Eds.), *LNCS: Vol. 1993. First international conference on evolutionary multi-criterion optimization* (pp. 359–372). Berlin: Springer.

Knowles, J. (2005). A summary-attainment-surface plotting method for visualizing the performance of stochastic multiobjective optimizers. In H. Kwasnicka & M. Paprzycki (Eds.), *ISDA '05: Proceedings of the 5th international conference on intelligent systems design and applications* (pp. 552–557). Los Alamitos: IEEE Computer Society.

Knowles, J. D., & Corne, D. W. (2000). Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary Computation*, *8*(2), 149–172.

Knowles, J. D., Thiele, L., & Zitzler, E. (2006) *A tutorial on the performance assessment of stochastic multiobjective optimizers* (Technical Report TIK Report No. 214). Switzerland: Computer Engineering and Networks Laboratory, ETH Zurich.

López-Ibáñez, M., Paquete, L., & Stützle, T. (2004). On the design of ACO for the biobjective quadratic assignment problem. In M. Dorigo, M. Birattari, C. Blum, L. M. Gambardella, F. Mondada, & T. Stützle (Eds.), *LNCS: Vol. 3172. ANTS'2004, Fourth international workshop on ant algorithms and swarm intelligence* (pp. 214–225). Berlin: Springer.

Maniezzo, V., & Carbonaro, A. (1999). Ant colony optimization: an overview. In P. Hansen & C. Ribeiro (Eds.), *Proceedings of the third metaheuristics international conference (MIC'99)* (pp. 21–44). Dordrecht: Kluwer Academic.

McMullen, P. R. (2001). An ant colony optimization approach to addressing a JIT sequencing problem with multiple objectives. *Artificial Intelligence in Engineering*, *15*(3), 309–317.

Purshouse, R. C., & Fleming, P. J. (2003). Conflict, harmony, and independence: Relationships in evolutionary multi-criterion optimisation. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, & L. Thiele (Eds.), *LNCS: Vol. 2632. Proceedings of the second international evolutionary multi-criterion optimization conference (EMO 2003)* (pp. 16–30). Berlin: Springer.

Romero, C. E. M., & Manzanares, E. M. (1999). MOAQ an Ant-Q algorithm for multiple objective optimization problems. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, & R. E. Smith (Eds.), *Genetic and evolutionary computing conference (GECCO 99)* (Vol. 1, pp. 894–901). San Mateo: Morgan Kaufmann.

Shelokar, P. S., Jayaraman, V. K., & Kulkarni, B. D. (2002). Ant algorithm for single and multiobjective reliability optimization problems. *Quality and Reliability Engineering International*, *18*(6), 497–514.

Socha, K., & Dorigo, M. (2008). Ant colony optimization for continuous domains. *European Journal of Operations Research*, *185*(3), 1155–1173.

Srinivas, N., & Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, *2*(3), 221–248.

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

T'kindt, V., Monmarché, N., Tercinet, F., & Laügt, 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.

Van Veldhuizen, D. A. (1999) *Multiobjective evolutionary algorithms: classifications, analyses, and new innovations*. Ph.D. thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH.

Zitzler, E., & Künzli, S. (2004). Indicator-based selection in multiobjective search. In X. Yao, E. Burke, J. A. Lozano, J. Smith, & J. J. Merelo-Guervos (Eds.), *LNCS: Vol. 3242. Proceedings of the 8th international conference on parallel problem solving from nature (PPSN VIII)* (pp. 832–842). Berlin: Springer.

Zitzler, E., Laumanns, M., & Thiele, L. (2002) SPEA2: Improving the strength Pareto evolutionary algorithm. In: K. Giannakoglou, D. Tsahalis, J. Periaux, P. Papailou, T. Fogarty (Eds.), *EUROGEN 2001, evolutionary methods for design, optimization and control with applications to industrial problems, international center for numerical methods in engineering (CIMNE)*, Barcelona, Spain (pp. 95–100).

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., & Fonseca, V. (2003). Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, *7*(2), 117–132.

Zitzler, E., Brockhoff, D., & Thiele, L. (2007). The hypervolume indicator revisited: On the design of Pareto-compliant indicators via weighted integration. In S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, & T. Murata (Eds.), *LNCS: Vol. 4403. Proceedings of the forth international evolutionary multi-criterion optimization conference (EMO 2007)* (pp. 862–876). Berlin: Springer.