Article:
Giagkiozis, I., Purshouse, R.C. and Fleming, P.J. (2013) An overview of population-based algorithms for multi-objective optimisation. International Journal of Systems Science, 46 (9). 1572 - 1599. ISSN 0020-7721

https://doi.org/10.1080/00207721.2013.823526

International Journal of Systems Science
Vol. 00, No. 00, 00 Month 20xx, 1–42

RESEARCH ARTICLE

An Overview of Population-Based Algorithms for Multi-Objective Optimization

Ioannis Giagkiozis[a]* and Robin C. Purshouse[a] and Peter J. Fleming[a]

[a]*Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, UK, S1 3JD*

(*14 August 2012*)

In this work we present an overview of the most prominent population-based algorithms and the methodologies used to extend them to multiple objective problems. Although not exact in the mathematical sense, it has long been recognized that population-based multi-objective optimization techniques for real-world applications are immensely valuable and versatile. These techniques are usually employed when exact optimization methods are not easily applicable or simply when, due to sheer complexity, such techniques could potentially be very costly. Another advantage is that since a population of decision vectors is considered in each generation these algorithms are implicitly parallelizable and can generate an approximation of the entire Pareto front (PF) at each iteration. A critique of their capabilities is also provided.

Keywords: Genetic Algorithms; Ant Colony Optimization; Particle Swarm Optimization; Differential Evolution; Artificial Immune Systems; Estimation of Distribution Algorithms

1. Introduction

Population based optimization techniques (PBOTs) have received much attention in the past 30 years. This can be attributed to their inherent ability to escape local optima, their extensibility to multiple objective problems (MOPs) and their ability to incorporate the handling of linear and non-linear inequality and equality constraints in a straightforward manner. They are applicable to a wide set of problems and impose very few constraints on the problem structure. For example continuity and differentiability are not required. PBOTs also perform well on NP-hard[1] problems where exhaustive search is impractical or simply impossible. Since there is a population of decision vectors, the Pareto front (PF) can be approximated (theoretically) to arbitrary precision. Namely, assuming that the Pareto front is continuous and that the selected algorithm is able to converge to the optimal points, by increasing the number of solutions the *resolution* of the Pareto front approximation can be moderated at will.

An additional difficulty with MOPs, apart from the apparent increase in complexity, is that the notion of a solution is very different from the single objective case. What does it mean then to say an algorithm *solves* an MOP? When dealing with single objective problems comparison of different objective function values is simple. If the task is to minimize an objective function then solutions can directly be compared since scalars are totally ordered. A *solution* then is a decision vector that achieves this goal.

*Corresponding author. Email: i.giagkiozis@sheffield.ac.uk

[1]Non-deterministic Polynomial time (NP)

ISSN: 0020-7721 print/ISSN 1464-5319 online

DOI: 10.1080/00207721.20xx.CATSid
http://www.informaworld.com

Assuming that the problem under consideration is being *solved* for practical purposes, that is; it is a real-world problem and a design choice is to be made. Even if multiple competing objectives are under consideration only one solution can usually be employed. This requires that a human expert or decision maker (DM) to resolve the situation by selecting a small subset of the solutions presented by the algorithm. So in this scenario, or more formally *a posteriori* preference articulation, the algorithm is endowed with the task of finding a set of alternative solutions. Subsequently the DM will evaluate these solutions according to his/her preferences and make a selection. To facilitate this process the algorithm guides a population of decision vectors toward a tradeoff surface, be it convex, concave, or even discontinuous. This tradeoff surface should enable the DM to choose a solution that they believe would serve their purposes well. So it becomes apparent that in MOPs a DM plays an integral role in the solution procedure. There are various paradigms of interaction involving the DM and the solution process but in the present work *a posteriori* preference articulation is the main focus; for other types see (Miettinen 1999). This choice is based on the fact that this particular method of interaction with the DM allows a greater degree of separation between the algorithm and the decision making process. Also this is invariably the reason as to why this paradigm is usually employed by researchers when developing new optimization techniques: it enables the testing process to be conducted independently of the problem or application and most importantly it need not involve the DM at this stage. It should also be noted that we do not mention specific applications in this work. The only exception to this rule is when an application explicitly results in creation of an algorithm highly regarded in the community. The reason for this decision is that this approach enables us to present a more concise conceptual view of PBOTs. A view that we hope would be helpful to the practitioner in search of an approach to solve a particular problem.

Other recent works that review population-based optimization methods are due to Liu et al. (2011) and Zhou et al. (2011). Although the work of Liu et al. (2011) is not a survey, the authors present an interesting approach in unifying concepts employed in PBOTs, and as such contains useful reference material about previous attempts. Zhou et al. (2011) present a more thematic overview of MOEAs and also provide an extensive list of applications of EAs in real-world problems. Although the work of Zhou et al. (2011) can serve as an excellent reference for researchers, there is an implicit assumption that the reader is familiar with certain concepts, and, as a result, it appears to be targeted towards experts in the field. Additionally we have found no previous work that elaborates on individual features present in each algorithm family and provides a critique. We also include a section on available software that implements the algorithm families reviewed in this work.

The purpose of this work is to provide a general overview of the most prominent population-based multi-objective optimization methodologies, serving as a map or a starting point in the search for a suitable technique, thus serving as a complement to existing works (Zlochin et al. 2004, Dorigo and Blum 2005, Jin and Branke 2005, Reyes-Sierra and Coello 2006, Tapia and Coello 2007, AlRashidi and El-Hawary 2009, Das and Suganthan 2010, Liu et al. 2011, Zhou et al. 2011). As in cartography, maps are meant to contain important landmarks but in order to enhance legibility a vast amount of detail has been abstracted or omitted. In that spirit, hybrid algorithms, for example memetic algorithms (Moscato 1989), are not considered here, without any implications for the contribution they make. Additionally although Genetic Programming (GP) instigated by Koza (1996) is based on the same principles as Genetic Algorithms (Holland 1975) it is also not considered here, since in the view of the authors, the main focus of GP is not multi-objective optimization. Additionally, the internal representation of a solution in GP is usually a simple graph which is radically different from the representation of the methods we consider here, namely real, discrete or binary linear encoding.

The remainder of this paper is organized as follows. In Section 2 we portray a chronology of developments in population-based multi-objective optimization algorithms and in Section 3 a generalization of the functional principles of population-based optimization techniques is dis-

tilled. Subsequently, in Section 4 seven algorithm families are presented in their main form, namely as they apply to single-objective problems and in the last subsection we elaborate on the ways that domain knowledge can be utilized to decide which of these methods would be more appropriate. As the methods used to extend the aforementioned algorithmic families to multi-objective problems are often very similar, we discuss these in Section 5 in a more general context and discuss how they have been applied in Section 6. In Section 7 we comment on available software for implementing the methods discussed in this work and also provide a listing of the supported algorithm families. The paper concludes with Section 8 and Section 9. In Section 8, these methodologies are assessed against a set of criteria which describe a range of problem features. Section 9 promising research directions are discussed along with a summary and conclusion of this work.

2. Chronology

The origins of the inspiration to use a population of trial solutions in an optimization algorithm can be traced back to (Holland 1962) as a mixture of automata theory and theoretical genetics. Since then several algorithms have been developed that utilize a population of decision vectors and by means of applying variation and exploration operators this population evolves to *adapt* to the *environment*. The *environment*, is represented by an evaluation (or objective) function, dictates which individuals in the population would be fit to enter the next generation. Holland (1975) formalized the idea presenting a more concrete framework for optimization which he named Genetic Algorithms (GA). The scheme quickly gained acceptance and Schaffer (1985) proposed an algorithm, the vector evaluated genetic algorithm (VEGA), based on Holland's GA to tackle multiple objectives simultaneously. The following year Farmer et al. (1986) published a paper describing how knowledge about the human immune system can be used to create adaptive systems. This Artificial Immune System (AIS) shares many commonalities with the classifier system described by Holland (1975). Two years later, Goldberg (1989) introduced a conceptual algorithm of how Pareto dominance relations could be utilized to implement a multi-objective GA. This idea is heavily used to this day when extending single objective algorithms to address MOPs.

Independently and in parallel with Holland, Rechenberg (1965) introduced Evolution Strategies (ES) in the mid sixties. The early versions of ES had only two individuals in their population due to the limited computational resources available at the time. However, in subsequent years multi-membered versions of ES were introduced by Rechenberg and further developed and formalized by Schwefel (1975).

Another important approach was introduced by Dorigo et al. (1991) which they called the Ant System (AS). The AS, inspired by the ability of ants to find the shortest route from a food source to their nest, was used to solve combinatorial problems and almost a decade later Dorigo and Di Caro (1999) presented a generalization of the AS which they named Ant Colony Optimization (ACO). In the mid 90s Storn and Price (1995) brought forward a novel heuristic, Differential Evolution (DE). Its main characteristic was its conceptual simplicity and its wide applicability to a variety of problems with continuous decision variables. Later that same year, Eberhart and Kennedy (1995) introduced Particle Swarm Optimization (PSO), a new family of optimisers inspired by the flocking behaviour of pack animals. Almost a full circle is complete by the latest addition to PBOTs, namely Estimation of Distribution algorithms (EDAs) (Mühlenbein and Paass 1996, Larranaga and Lozano 2002). A full circle in the sense that concepts derived from nature were employed as inspiration for more than two decades, helping researchers solve complex problems. EDAs are one of the first metaheuristic algorithms not to rely on nature inspired mechanisms to drive their search. Instead, EDAs depend on probabilistic models of better performing individuals in the population to generate the entire or part of the next generation.

3. General Structure

A single objective optimization problem is defined as,

$$\begin{aligned} &\min_{\mathbf{x}} f(\mathbf{x}) \\ &\text{subject to } \begin{cases} g_i(\mathbf{x}) \leq 0 & i = 1, \ldots, m \\ h_i(\mathbf{x}) = 0 & i = 1, \ldots, d \\ \mathbf{x} \in \mathbf{dom}\, f & \mathbf{dom}\, f \subset \mathrm{R}^n \end{cases} \end{aligned} \tag{1}$$

where m and d are the number of inequality and equality constraints respectively, n is the size of the decision vector $\mathbf{x}$ and $\mathbf{dom}\, f$ is the domain of definition of the objective function. The type of the functions f, g and h as well as the domain of f, are the key factors that determine the type of the problem defined by (1). For example when the objective function is convex and is defined over a convex domain, and, the inequality constraints are convex functions while the equality constraints are affine, then (1) is a convex optimization problem (Boyd and Vandenberghe 2004). An optimization problem defined by (1) is considered to be solved when a decision vector $\tilde{\mathbf{x}} \in \mathbf{dom}\, f$ is found that,

$$\begin{aligned} &f(\tilde{\mathbf{x}}) \leq f(\mathbf{x}), \text{ for all } \mathbf{x} \in \mathbf{dom}\, f, \\ &\text{and } \begin{cases} g_i(\tilde{\mathbf{x}}) \leq 0 & i = 1, \ldots, m \\ h_i(\tilde{\mathbf{x}}) = 0 & i = 1, \ldots, d\,. \end{cases} \end{aligned} \tag{2}$$

Such a decision vector, $\tilde{\mathbf{x}}$, is said to be a *global minimum*. A local minimum is defined as,

$$\begin{aligned} &f(\tilde{\mathbf{x}}) \leq f(\mathbf{x}), \text{ for all } \mathbf{x} \in \{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_c\|_2 \leq r, r \in \mathrm{R}, \mathbf{x}_c \in \mathbf{dom}\, f\}, \\ &\text{and } \begin{cases} g_i(\tilde{\mathbf{x}}) \leq 0 & i = 1, \ldots, m \\ h_i(\tilde{\mathbf{x}}) = 0 & i = 1, \ldots, d\,, \end{cases} \end{aligned} \tag{3}$$

and is usually what can be found for nonconvex problems, since it is usually impracticable to search the entire feasible set so as to ensure that the global minimum is found.

A useful device is the definition of the *feasible set* in decision space, namely,

$$S = \{\mathbf{x} : \mathbf{x} \in \mathbf{dom}\, f, g_i(\tilde{\mathbf{x}}) \leq 0, i = 1, \ldots, m,\ h_i(\tilde{\mathbf{x}}) = 0,\ i = 1, \ldots, d\,\}, \tag{4}$$

which is also called, the *feasible region*. With the help of (4), (1) can be rewritten in a more compact form,

$$\begin{aligned} &\min_{\mathbf{x}} f(\mathbf{x}) \\ &\text{subject to } \mathbf{x} \in S. \end{aligned} \tag{5}$$

Population-based meta-heuristics operate on a set of decision vectors, termed as *population*, which in this work we denote as $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ where $\mathbf{x}_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,n})$, N is the number of decision vectors in the *population* (or *size* of the population) and n is the number of decision variables in each decision vector. It should also be mentioned that decision vectors are in some contexts referred to as *individuals*, this is mostly to emphasize the nature-derived principles used in various algorithms and is employed to promote conceptual coherency rather than functional discrimination.

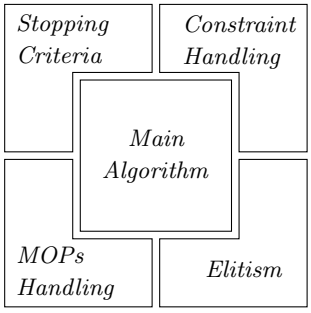


Figure 1.: PBOT components.

PBOTs, despite the diversity in their problem solving approach, share more similarities than differences. In general they are comprised of five parts, (see Fig. (1)): the main algorithm, an extension to tackle multi-objective problems (MOPs) (discussed in Section 5), an extension to deal with constrained optimization problems, a part to maintain promising solutions and a part to halt the algorithm execution based usually on some notion of convergence. The main algorithm deals primarily with single objective optimization, and is comprised of three parts. Those three parts effectively drive the search combining information within the population (recombination), randomly perturbing some individuals to enhance search space exploration (mutation), these two are lumped into a single operator termed *variation* in *Alg. 1* to maintain generality. and an operator to select promising individuals to be part of the new generation (selection). By recombination we mean the superset of operators, usually crossover, that utilize two or more decision vectors to form one or more new decision vectors. All three parts of the main algorithm have some deterministic and stochastic features. If these three operations are chosen correctly the population $\mathbf{X}^{(G+1)}$ will have a better chance being *superior* to the previous generation $\mathbf{X}^{(G)}$. Superiority in this context is determined in different ways depending on the type of objective function, for example if it is a scalar objective function then for a minimization problem a superior solution is one that evaluates to a smaller number compared to its predecessor[1] . In the case of a multi-objective problem this issue is more ambiguous, see Section 5. It should be noted however, that there is an implicit assumption here. Namely, that there is some information in the parent population which can be exploited to *construct* better solutions in the offspring population. Should this assumption fail so will the algorithm in identifying optimal, or near optimal, solutions.

Algorithm 1 PBOT Conceptual Algorithm

```
1: Initialize Population
2: Evaluate Objective Function
3: repeat
4:     Evaluate population quality
5:     Apply Variation Operator
6:     Evaluate Objective Function
7: until Termination Condition is Met
```

The general algorithm in PBOTs can be seen in *Alg. 1*, the population is usually initialized within the feasible region, if possible, and uniformly distributed in the absence of prior information. Most population-based metaheuristics are based on empirical knowledge and techniques that would steer the search toward desirable regions of S or directly replicate mechanisms found in nature.

[1]This is valid in a minimization context, while the opposite is true in a maximization context.

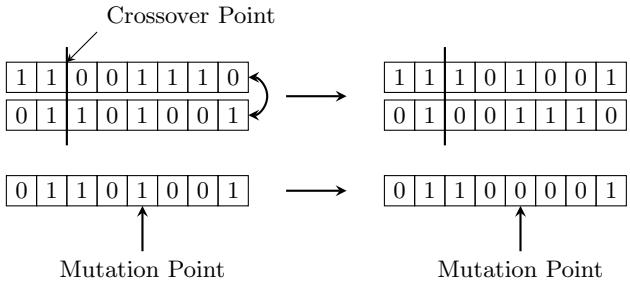


Figure 2.: Single point crossover and bit-flip mutation operator in Genetic Algorithms.

4. Algorithm Families

The main algorithm, at least as was initially introduced, of the seven algorithm families is presented in this section. The motivation for this short introduction is to provide context for our discussion in Section 4.8, Section 5 and Section 8.

4.1. *Genetic Algorithms*

GAs are a family of stochastic meta-heuristic search algorithms with links to cybernetics (Holland 1962, Goldberg and Holland 1988) and philosophically connected to Darwin's theory of evolution (Darwin 1858). Social dynamics are simulated based on the premise that the *survival of the fittest* paradigm does produce qualitatively better *individuals* with the passage of several generations. Individuals are in effect decision vectors whose fitness is evaluated using the objective function. The representation of decision vectors in the basic GA is a binary *string* termed *chromosome* comprised of *genes* (decision variables). Each chromosome is mapped to a *phenotype*. A phenotype is a representation that the objective function can directly utilize to evaluate the fitness of an individual. The main operators used in GAs are crossover and mutation. These

Algorithm 2 Simple Genetic Algorithm

1: Initialize Population
2: **repeat**
3: Evaluate Objective Function
4: Apply Crossover and Mutation
5: Select solutions for next population
6: **until** Termination Condition is Met

two operators are combined in a fashion to balance exploitation and exploration of the search space. Crossover and mutation, see Fig. (2), are applied to the chromosomes thus facilitating the creation of an abstraction layer with respect to the phenotype. This layer of abstraction insulates the internal implementation of GAs from the problem structure, specifically the representation type of the decision variables. The main operators in GAs have been extended to deal directly with continuous decision variables (Deb and Agrawal 1994, Ono and Kobayashi 1997, Voigt et al. 1995) which is useful when full machine precision is required and all decision variables are continuous.

4.2. *Evolution Strategies*

The introduction of ES is due to Rechenberg (1965). This early version of ES was a two-membered mutation based algorithm the so called (1 + 1)-ES. Although ES share similar features and

inspirational background to GAs, there are two distinct differences. First ES do not employ a crossover operator, and second, ES utilize real decision variables as opposed to GAs that originally employed binary encoded *strings*. However, these differences have blurred over time due to developments on both sides. Effectively both methodologies pursued adaptation and were inspired by the cybernetic movement of that era.

There is a general consensus that both mutation and recombination operators perform an important role and eventually both were employed to various degrees in these two approaches. The main argument concerns their effect on the evolving population. Holland, with his famous building block hypothesis (BBH) (Holland 1975), promoted the idea that the recombination operator had a positive influence on the evolving population due to short *schemas* with good *fitness* recombining to form even better individuals. Beyer argued that this was not the case; his suggestion was that the crossover operator had the effect of *genetic repair* (Beyer 1997). Beyers' hypothesis was that the features that *survive* the crossover operator were the ones common to both parents and not the small *schemas* with desirable elements.

As previously mentioned the first version of ES was $(1+1)$-ES. This notation refers to the fact that there are two members in the population, namely the *parent* and one *offspring* and the selection for the *surviving* individual is performed among the two. The $+$ sign essentially refers to the selection mechanism. Some later versions of ES use the comma notation, denoting that the *parent* takes no part in the selection process and new individuals are selected only from the offspring pool. The main *search* operator in $(1+1)$-ES is the mutation operator defined as follows,

$$\mathbf{x}^{(G+1)} = \mathbf{x}^{(G)} + \mathcal{N}(\mathbf{0}, \sigma) \tag{6}$$

were $\mathcal{N}(\mathbf{0}, \sigma)$ is a vector of random numbers of size n drawn from the normal distribution with zero mean and σ standard deviation. Here $\mathbf{x}^{(G)}$ represents the parent solution and $\mathbf{x}^{(G+1)}$ the offspring. If $f(\mathbf{x}^{(G+1)}) \leq f(\mathbf{x}^{(G)})$ the offspring is retained and the parent is discarded. Otherwise the same parent is used in the next generation.

ES were extended to a *true* population-based optimization methodology by Schwefel (1981). Schwefel introduced $(\mu + \lambda)$-ES as well as (μ, λ)-ES. Here μ is the number of parents producing λ offspring. The resulting population, after the selection process, is reduced to the number of parents μ. Schwefel combined these two variants to enable a parallel implementation of ES as well as to experiment with adaptation in the control parameters such as σ (Bäck et al. 1991). For further details on the $(\mu + \lambda)$-ES the reader is referred to (Schwefel 1981, Bäck et al. 1991, Beyer and Schwefel 2002).

4.3. *Artificial Immune Systems*

The immune system (IS) is a fine tuned concert of a vast array of cells acting in synchronism in order to protect the body from pathogens. Pathogens are substances or microorganisms that can affect the balance of the body (*homeostasis*), impairing its normal functioning and even cause death. The immune system responds, *immune response*, to such threats creating specialized cells capable of neutralizing pathogens and restoring balance. In order for this procedure to progress smoothly the immune system must attack the source of the disturbance while minimizing the effect this action has on healthy cells in the body - *specificity*. Once a particular threat has been dealt with, the second time the body is exposed to the same pathogen the immune response will be swifter and stronger to such a degree that the organism might not even realize that it had been afflicted by a pathogen, so the immune system also exhibits *memory*.

Pathogens have certain features that the immune system can identify. These features are called *antigens*, and their presence in the organism is detected by *antibodies*. Antibodies are highly specialized molecules that have the capacity to identify specific antigens. An antibody,

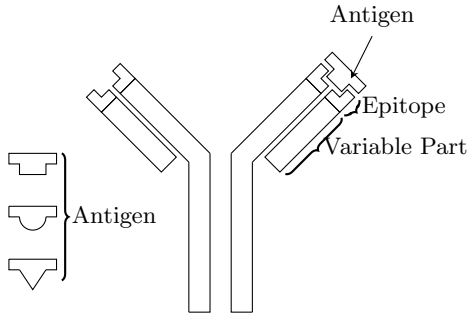
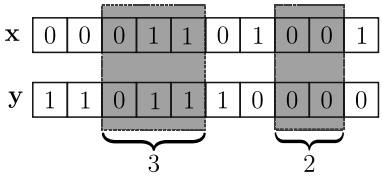


Figure 3.: Antibody schematic diagram.

see Fig. (3), identifies an antigen by binding on it, thus serving as a beacon signalling other cells in the immune system to devour the intruder. Although antibodies are highly specialized, their diversity is enormous thus enabling the immune system to attack a vast variety of antigens. When a particular antigen is identified the antibody that successfully identified it will proliferate by cloning via a process termed *the clonal selection principle* (Mohler et al. 1980, Dasgupta and Forrest 1999). Once the pathogens have been eliminated, superfluous antigens are dissolved mainly via the process of *apoptosis* (Krammer 2000). However some cells responsible for the proliferation of this particular type of antibodies survive as memory cells (Parijs and Abbas 1998), and if the host is infected again by the same pathogen these memory cells will be activated to produce antibodies much faster compared with the first encounter, *primary response*, with this particular antigen.

Although this introduction to the immune system will serve the purpose of illustrating the parallelism of the biological system with Artificial Immune Systems (AIS), it does not even scratch the surface of the delicate intricacies present in the immune system. The interested reader is referred to (Parham and Janeway 2005).

AIS have been employed in various applications, such as intrusion detections systems, virus detection, anomaly detection fault diagnosis, and pattern recognition (Dasgupta and Attoh-Okine 1997). AIS have been used in place of fitness sharing techniques in GAs to maintain diversity and distribute the population evenly along the PF (Coello and Cortés 2005). Another interesting use is in time dependent optimization problems where AIS successfully *adapt* to a changing problem (Gasper and Collard 1999). Despite these applications, direct use of AIS to optimization problems prior to 1999 is scarce. This could potentially be attributed to the complexity involved in implementing an AIS.



$$d_H(\mathbf{x}, \mathbf{y}) = 5$$

$$f(\mathbf{y}, \mathbf{x}) = d_H(\mathbf{x}, \mathbf{y}) + \underbrace{3^p + 2^p}_{m}$$

Figure 4.: Similarity measure used to simulate the clonal selection principle (Coello and Cortés 2002). Here, $\mathbf{x}, \mathbf{y}$ represent an antigen and antibody respectively. The similarity measure, $f(\cdot, \cdot)$, is used to modulate the mutation strength in an inverse proportional manner.

Although there is no single uniform framework that describes AIS for optimization, a representative algorithm based on the principles of immune systems is due to Coello and Cortés (2002). Here we describe the main features of this method, while for further details the reader is referred to Coello and Cortés (2002, 2005). Coello and Cortés (2002) draw upon the the clonal selection principle by labelling elite individuals as antigens; these are solutions that are non-dominated and feasible. Inferior individuals where labeled as *antibodies*, namely solutions that are either infeasible, dominated or both. Subsequently a modified Hamming distance metric was employed for fitness assignment,

$$f(\mathbf{x}, \mathbf{y}) = d_H(\mathbf{x}, \mathbf{y}) + m, \tag{7}$$

where m is an additional biasing factor that depends on the number of similar groups found between, $\mathbf{x}$ and $\mathbf{y}$. An illustration of the calculations involved in evaluating (7) as well as a definition of m is given in Fig. (4). Subsequently the mutation strength for the generation of new individuals is determined by a reverse proportional relationship with (7) (Coello and Cortés 2002). The aforementioned algorithm cannot be classified as a modified genetic algorithm as new individuals are formed only by means of mutation. In this respect the method presented by Coello and Cortés (2002) is similar to ES.

4.4. *Ant Colony Optimization*

Ants come from the same family as wasps and bees Choe and Perlman (1997), namely *Hymenoptera Fromicidae*. As individuals, ants' capabilities are limited. Apart from their extraordinary strength their sight is very limited and their hearing and sense of smell depend on their antennas. Despite these facts, ants as a collective exhibit very interesting behaviour patterns (Deneubourg et al. 1990). In their search for food, ants explore their nest surroundings in a random-walk pattern. However, when an ant discovers food, it returns to the nest releasing pheromones (Deneubourg et al. 1990) of varying intensity depending on the quality and quantity of the discovered food source. Other ants close to the pheromone trail are more likely to follow that path and, if there is still food to be found at the end of the trail, they release pheromones as well, thus reinforcing the pheromone scent which in turn increases the probability that more ants will follow that path. Alternatively when an ant reaches the destination indicated by the pheromone trail and does not discover anything interesting, no pheromone is released and progressively the path becomes less and less attractive as pheromones evaporate.

Dorigo et al. (1991) introduced an algorithm, *Ant System* (AS), inspired by the emergent behaviour of ants. The AS was tailored to solve combinatorial problems and the travelling salesman problem (TSP) was used as a test bed, see Fig. (5). Almost a decade later Dorigo and Di Caro (1999) formalized the AS and other similar methods in a unified framework, ant colony optimization (ACO). ACO is comprised of N individuals called *ants*. These are distributed among μ cities and each ant progressively creates a candidate solution for the problem. The ants *decide* which city they will next visit based on the pheromone trail (τ) associated with each edge leading to a particular city and the separating distance of the cities. Assuming that an ant starts off from city A in Fig. (5), it has three options, namely to go to B, C or D. Each of these edges has a pheromone trail associated with it. The probabilistic transition rules are controlled by two parameters, namely τ and η. The parameter τ can be initialized using prior information, if available, or with the same value for all edges. Initially η represented the *visibility* of the ants which depended on the distance of a city from the next and was defined as, $\eta_{AB} = \frac{1}{d_{AB}}$ for the edge connecting city A with B. The smaller η_{AB} is, thus the larger the distance d_{AB} is, the less likely is that an ant in city A to choose the city B for its next step in the tour, see (9). As ants *traverse* across the edges constructing different solutions the pheromone across the edges is

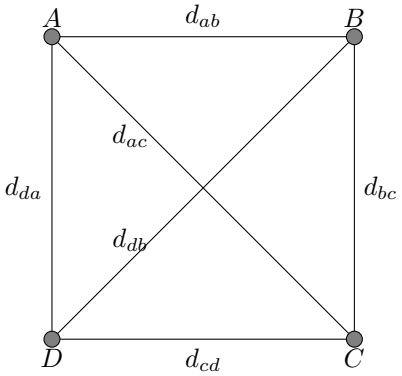


Figure 5.: Travelling salesman problem with four cities. Although the edges connecting the cities seem equidistant they need not be. The distance of two cities is represented by d.

updated as,

$$\tau_{i,j} = (1-\rho)\tau_{ij} + \sum_{s=1}^{N} \Delta\tau_{i,j}^{s} \tag{8}$$

where ρ is defined as the evaporation rate of the pheromones, and $\Delta\tau_{i,j}^{s}$ is the amount of pheromone added by ant s on the edge connecting the i^{th} and j^{th} node. Therefore, the larger the deposit of pheromones across an edge i, j the larger $\tau_{i,j}$ is, hence the probability that more ants will select this path is increased. The probabilistic transition rule, based on τ and η, that govern the *behaviour* of ants is summarised by the following relation,

$$p_{i,j} = \begin{cases} \frac{\tau_{i,j}^{\alpha}\eta_{i,j}^{\beta}}{\sum_{m=1}^{\mu}\tau_{i,m}^{\alpha}\eta_{i,m}^{\beta}} & \text{if } j \text{ is a valid destination} \\ 0 & \text{if } j \text{ is not a valid destination,} \end{cases} \tag{9}$$

where a valid destination for an ant is a node that the particular ant is allowed to visit at that particular stage. The probability that an ant on node i will visit a node j is given by $p_{i,j}$, and μ is the number of nodes in the problem. For the TSP problem a node is a *valid* destination if a particular ant has not visited that node. Also α and β are real and positive numbers used to favour either τ or η; if $\alpha = \beta$ the information gained from τ is considered equally important with η.

The general ACO algorithm has the following stages,

Step 1 Initialize pheromone values τ for all the edges. Usually all the edges receive the same pheromone value at the start of the algorithm unless prior information is available indicating that favouring certain edges would increase convergence speed.
Step 2 Construct a solution for each ant, $\mathbf{x}_s$.
Step 3 Update the pheromone values for each edge.
Step 4 Go to **Step 2** until stopping criteria are met.

The rules for constructing a solution $\mathbf{x}_s$ for each ant are problem-dependent.

Since 1991, ACO has been successfully applied to various combinatorial problems such as vehicle routing, sequential ordering, project scheduling, multiple knapsack and protein folding (Dorigo et al. 2006).

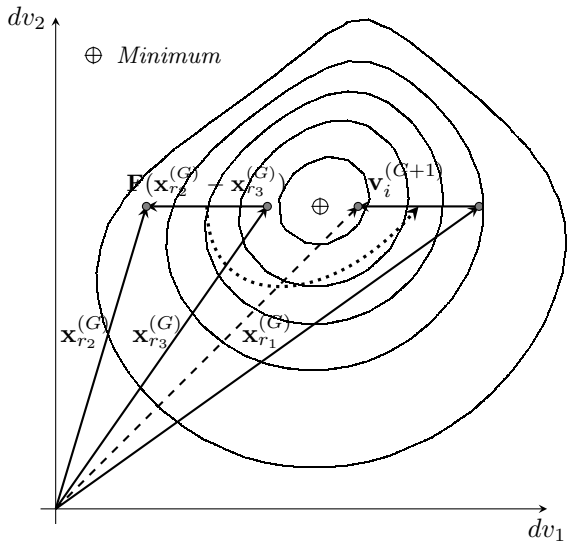


Figure 6.: Illustration of the mutation operator in DE, in this case $F = 1$ and $\mathbf{x}_i^{(G+1)} = \mathbf{v}_i^{(G+1)}$ since for this instance $f(\mathbf{x}_i^{(G+1)}) < f(\mathbf{x}_i^{(G)})$, and, dv_1 and dv_2 stand for decision variable one and two respectively.

4.5. *Differential Evolution*

DE was introduced by Storn and Price (1995) as a global search method over real decision vectors, although later extended versions for integer (Lampinen and Zelinka 1999) and binary representations (Pampara et al. 2006) were introduced. The major strengths of DE are its conceptual simplicity, ease of use and implementation; also the number of tuning parameters is very small and the same parameter values with no or very little change are found to be applicable to a wide range of problems.

For a population of size N and decision vectors of dimension n, the current generation is

$$\mathbf{x}_i^{(G)} \text{ for } i = \{1, 2, \ldots, N\}. \tag{10}$$

The basic DE algorithm has three stages in its iteration phase: mutation, crossover and selection. During the mutation stage all decision vectors in the population are perturbed resulting in a temporary new population $\mathbf{V}^{(G+1)}$ of the same size with $\mathbf{X}^{(G)}$. The mutation operator in DE is described as,

$$\begin{gathered} \forall\, \mathbf{x}_i^{(G)},\ i = \{1, 2, \ldots, N\}, \\ \mathbf{v}_i^{(G+1)} = \mathbf{x}_{r_1}^{(G)} + F\left(\mathbf{x}_{r_2}^{(G)} - \mathbf{x}_{r_3}^{(G)}\right), \end{gathered} \tag{11}$$

and if $f(\mathbf{v}_i^{(G+1)}) < f(\mathbf{x}_i^{(G)})$ the newly formed parameter vector $\mathbf{v}_i^{(G+1)}$ is assigned to $\mathbf{x}_i^{(G+1)}$, otherwise $\mathbf{x}_i^{(G)}$ is retained in the next generation. The parameters $r_1, r_2, r_3 \in \{1, 2, \ldots, N\}$ are sampled at random without replacement from the set $\{1, \ldots, N\} - \{i\}$ for each individual in the population. The parameter $F \in [0, 2]$ is a scaling factor controlling the variation $(\mathbf{x}_{r_2}^{(G)} - \mathbf{x}_{r_3}^{(G)})$.

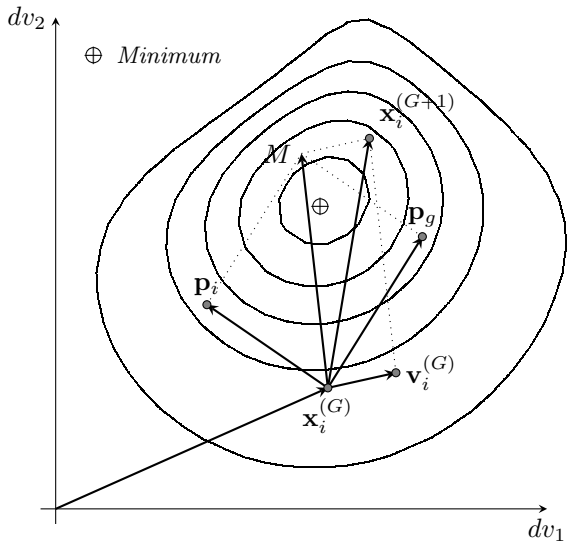


Figure 7.: Particle update illustration in particle swarm optimization. Where, $M = c_1\mathcal{U}(0,1)(\mathbf{p}_i - \mathbf{x}_i^{(G)}) + c_2\mathcal{U}(0,1)(\mathbf{p}_g - \mathbf{x}_i^{(G)})$ and $c_1\mathcal{U}(0,1), c_2\mathcal{U}(0,1) = 1$ for clarity. As before, dv_1 and dv_2 stand for decision variables 1 and 2 respectively.

The crossover operator in DE is defined as

$$\mathbf{u}_i^{(G+1)} = \left(u_{i,1}^{(G+1)}, u_{i,2}^{(G+1)}, \ldots, u_{i,n}^{(G+1)}\right),$$

where

$$u_{i,j}^{(G+1)} = \begin{cases} v_{i,j}^{(G+1)} & \mathcal{U}(0,1) \leq C_r \text{ or } j = ridx(i), \\ x_{i,j}^{(G)} & \mathcal{U}(0,1) > C_r \text{ and } j \neq ridx(i), \end{cases} \tag{12}$$

$$j = 1, 2, \ldots, n\,,$$

and C_r is the crossover rate parameter, bounded in the range $[0,1]$, $ridx(i)$ is a random index in the range $\{1,\ldots,n\}$,[1] and $\mathcal{U}(0,1)$ is a random number sampled from a uniform distribution in the domain $[0,1]$. Once the new decision vectors $\mathbf{u}_i^{(G+1)}$ have been generated, the selection is based on the *greedy* criterion, which can be stated as

$$\mathbf{x}_i^{(G+1)} = \begin{cases} \mathbf{u}_i^{(G+1)} & \text{if } f(\mathbf{u}_i^{(G+1)}) < f(\mathbf{x}_i^{(G)}) \\ \mathbf{x}_i^{(G)} & \text{otherwise.} \end{cases} \tag{13}$$

So the DE algorithm progresses exactly as *Alg. 2* but using the crossover, mutation and selection operators defined above, in place of the variation operator. Some general guidelines on tuning DE can be found in (Storn 1996).

4.6. *Particle Swarm Optimization*

Particle swarm optimization, introduced in (Eberhart and Kennedy 1995, Kennedy and Eberhart 1995) was inspired by the flocking behaviour of birds and swarm theory. In PSO, as observed in nature, each agent has a rather limited repertoire of behaviours while the collective exhibits

[1] As a reminder, n represents the number of decision variables.

complex expressions. In the initial PSO algorithm the particles (decision vectors) use a simple update rule to update the velocity and, consecutively, the position of each particle. An archive is maintained that contains the best achieved objective function values for each particle

$$\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N\}, \tag{14}$$

so $f(\mathbf{p}_i) \leq f(\mathbf{x}_i^{(G)})$ and if $f(\mathbf{p}_i) > f(\mathbf{x}_i^{(G+1)})$, then $\mathbf{p}_i = \mathbf{x}_i^{(G+1)}$ and so on. Additionally a global best position is maintained $\mathbf{p}_g$ for which the following condition must hold $f(\mathbf{p}_g) \leq f(\mathbf{x}_i^{(G+1)})$, for all $i \in \{1, 2, \ldots, N\}$. If this is not true $\mathbf{p}_g$ is updated using the following rule $\mathbf{p}_g = \{\mathbf{x}_i : \min_i f(\mathbf{x}_i)\}$. The velocity update rule is:

$$\begin{aligned} \mathbf{v}_i^{(G+1)} = \mathbf{v}_i^{(G)} &+ c_1 \mathcal{U}(0,1) \left(\mathbf{p}_i - \mathbf{x}_i^{(G)}\right) \\ &+ c_2 \mathcal{U}(0,1) \left(\mathbf{p}_g - \mathbf{x}_i^{(G)}\right), \end{aligned} \tag{15}$$

where c_1 and c_2 are positive constants and $\mathcal{U}(0,1)$ is a random number in the range $[0,1]$. A suggested value for c_1 and c_2, for problems when no prior information is available, is that both are set to 2 (Kennedy and Eberhart 1995). This would effectively result in a multiplier with a mean value of one, thus balancing in the mean the bias toward the point $\mathbf{p}_g$ and $\mathbf{p}_i$ for each particle (Shi and Eberhart 1998). A modification to (15) was presented by Shi and Eberhart (1998) introducing a multiplying factor w to $\mathbf{v}_i^{(G)}$, the *inertia weight*, resulting in the following velocity update relation,

$$\begin{aligned} \mathbf{v}_i^{(G+1)} = w \cdot \mathbf{v}_i^{(G)} &+ c_1 \mathcal{U}(0,1) \left(\mathbf{p}_i - \mathbf{x}_i^{(G)}\right) \\ &+ c_2 \mathcal{U}(0,1) \left(\mathbf{p}_g - \mathbf{x}_i^{(G)}\right) \ , \end{aligned} \tag{16}$$

where w can be constant, a function of the current generation G or even a function of a metric measuring the convergence of the algorithm. (For example the normalized hypervolume indicator could be utilized.) A value of 1 for w results in the regular velocity update rule, while a value below 1 progressively decreases the average velocity of the particles biasing the search to regions local to the particles. Alternatively, a value above 1 leads to a progressive increase in the velocity of particles resulting in a more explorative behaviour. Subsequently the position of the new particles is calculated in the following way,

$$\mathbf{x}_i^{(G+1)} = \mathbf{x}_i^{(G)} + \mathbf{v}_i^{(G+1)} \ . \tag{17}$$

PSO does not have an explicit selection operator, although the archive $\mathbf{p}$ could qualify as such.

Several variants of the described PSO algorithm have been devised, the interested reader is referred to (del Valle et al. 2008, Tripathi et al. 2007). Other interesting methods that cannot be classified as particle swarm optimizers but inherit some features are the group search algorithm (He et al. 2009) and several variants of the bee colony optimization (Akay and Karaboga 2012, Karaboga et al. 2012,?).

4.7. *Estimation of Distribution Algorithms*

In 1995, several crossover operators in GAs appeared that used probability distributions, for example, simulated binary crossover (Deb and Agrawal 1994), unimodal normal distribution

crossovers (Ono and Kobayashi 1997) and fuzzy recombination (Voigt et al. 1995), to name but a few. These crossover operators, although they did not recombine more than two or three individuals from the population, did use a probability distribution that was based on the parent solutions to generate the offspring. Estimation of distribution algorithms can, in a way, be seen as an expansion to the idea behind these crossover operators. The generalization is straightforward. Instead of using a crossover and mutation operator in the GA, a probability distribution over the most prominent of solutions can be estimated and then utilized to produce new individuals in the population (Mühlenbein and Paass 1996). Arguably, another source of inspiration in the creation of EDAs has come from the statistics community, for example, the cross entropy method (Rubinstein 1999, 2005) was initially used for rare event estimation and probability collectives (Bieniawski et al. 2004) has a game theoretic foundation (Bieniawski 2005, Wolpert et al. 2006).

A typical EDA, see *Alg. 3*, proceeds very similarly to a GA. The difference is that on every generation a subset of the population, usually the *better* half of the population, is selected and based on that subset a probabilistic model is created. Then this model is sampled and the resulting new individuals are merged with the old population while maintaining the population size constant.

Algorithm 3 Estimation of Distribution Algorithm

1: Initialize population $\mathbf{X}^{(0)}$
2: Evaluate objective function
3: **repeat**
4: Select promising solutions $\mathbf{X}_p$ from $\mathbf{X}^{(G)}$
5: Create a probabilistic model P based on $\mathbf{X}_p$
6: Generate new solutions $\mathbf{X}_n$ by sampling P
7: Evaluate the objective function for $\mathbf{X}_n$
8: Combine $\mathbf{X}_n$ and $\mathbf{X}^{(G)}$ to create $\mathbf{X}^{(G+1)}$
9: **until** Termination condition is met

While EDAs have successfully been applied to a diverse problem set, involving real and discrete decision variables, outperforming rival algorithms (Hauschild and Pelikan 2011), it is not so trivial to generate the required probabilistic model. Also, the model type strongly depends on the decision variable type, whether or not the decision variables are coupled, to what extent and in what way. Assuming that all decision variables are independent, while in fact there are multiple interdependencies would grossly mislead the algorithm (Pelikan et al. 2002). Additionally these challenges increase in difficulty when dealing with MOPs (Laumanns and Ocenasek 2002, Sastry et al. 2005). Despite these difficulties, further research in EDAs seems promising due to their inherent adaptive nature that enables them to scale well compared with other algorithms for some problems (Shah and Reed 2011). Another strong point is that prior knowledge can effectively be exploited by biasing directly the initial population (Sastry 2001) or by biasing the model creation procedure (Muhlenbein and Mahnig 2002). However as seen in Fig. (14) the research activity in EDAs is still in relatively low levels, a fact that can be ascribed to the difficulty of creating a probabilistic model capable of capturing dependencies in a way that doesn't substitute one difficult problem, namely the original optimization problem, with another, that of building a probabilistic model.

4.8. *So Why Not a Single Approach?*

As is seen in Fig. (14), the number of publications appearing per year is constantly increasing for all the above algorithm families, albeit not at the same pace. However given their similarities and according to the *survival of the fittest* principle one would expect some methods to fade

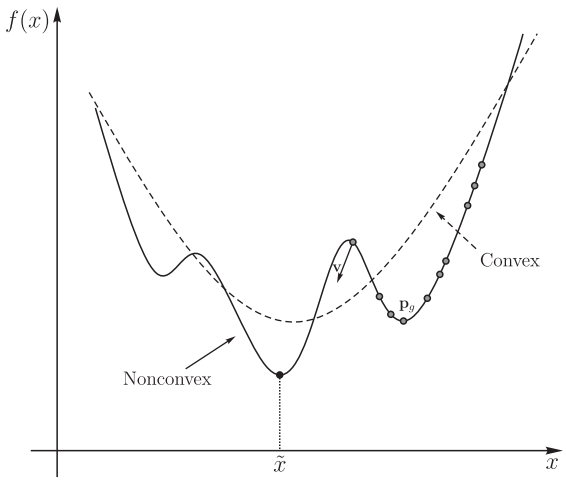


Figure 8.: Illustration of a convex (dashed line) and nonconvex (continuous line) multi-modal function. The points on the nonconvex function represent a *swarm* of solutions in a PSO algorithm.

away. This seems not to be the case for the above seven algorithm families, which is one of the reasons we elected to present them in this work.

This *survival* must mean that there is a niche that every family fulfills better than the others, while no algorithm family is *dominated*[1] by any other in terms of performance. This result is not surprising and is partially explained in (Wolpert and Macready 1997). However, that niche seems to be rarely expressed, with a few exceptions of course, for example ACO are renown for their applications on combinatorial problems, see Fig. (15), but what of the others? At this point we exploit what (Ullman 2009) considers to be a bad practice in research, namely for researchers to explore the unsolved problems reading the conclusions of published articles and focusing their work on the problems that they can solve. If this is in fact what is common practice, which (Michalewicz 2012) seems to agree with, then we will be able, to some extent, to identify the strengths and weaknesses of the aforementioned algorithm families by the number of papers published with respect to different problems, see Fig. (15).

Consider convex optimization algorithms. In this class gradient information is exploited to determine a direction of search. However real-world problems do not always conform to the structure that such a class of optimization methods can attack. One way that a problem can fail the convexity test is illustrated in Fig. (8). This type of problems are called multi-modal and have multiple local optimal. In this setting a particle swarm optimization algorithm would very well suited as it is conceivable that in the depicted scenario in Fig. (8) a PSO based algorithm has the potential to overcome the ridges and locate eventually the global optimum, $\tilde{x}$. However the fact that the swarm of solutions in Fig. (8) has reached to that location is due to their *following* the solution $\mathbf{p}_g$ which in turn follows a path that is similar to that of a gradient descent algorithm. For this problem an algorithm with more *aggressive* stochastic operators would require more function evaluations compared to PSO. Problems with this structure are prevalent in evolutionary algorithm test functions, for example see (Deb et al. 2002, Huband et al. 2006, Saxena et al. 2011). For the same reasons differential evolution would have the potential to perform well in such a class of problems.

Now consider a different type of nonconvex function, see Fig. (9). In this case the *ridge* would be more difficult to *cross* using PSO, however it would be much easier for an algorithm whose stochatic component is more dominant, for example ES or EDAs. Of course given the number

[1]See the next section for a definition of dominance.

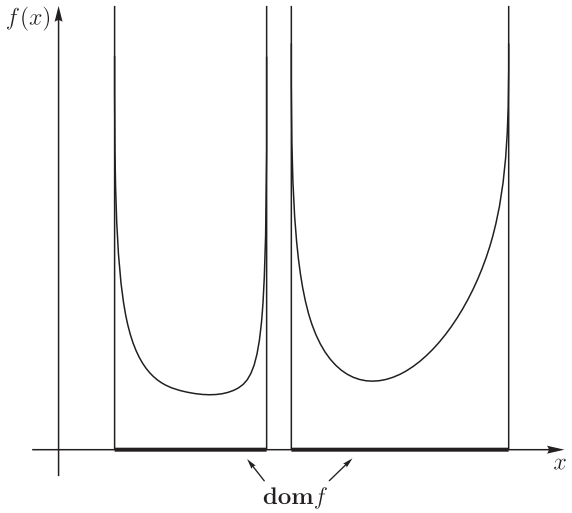


Figure 9.: A discontinuous objective function. The minimum is located in the left lobe however algorithms with *weak* stochastic components that most of their initial population is located in the right lobe will have difficulties in locating the global optimum.

of cross-over operators available for GAs it could be argued that such a behaviour could be simulated by judicious selection of such operators. Moreover we neglected the fact that all the aforementioned methods have tunable parameters, which in turn alter their *behaviour*. Nevertheless, the different conceptual paradigms may enable the practitioner select these parameters more easily for a certain type of algorithms when applied to a problem with a structure that *favours* that particular algorithm family.

5. Multi-Objective Problems

When the objective function is vector valued, that is $\mathbf{F} : \mathrm{R}^n \to \mathrm{R}^k$ then the optimization problem becomes more complex. This complexity stems from the fact that now there exists the possibility that there is no single objective function value $\mathbf{F}(\tilde{\mathbf{x}}) \leq \mathbf{F}(\mathbf{x})$ for all $\mathbf{x} \in \mathrm{R}^n$, as was the case for single objective problems. Therefore, in all but the most trivial case where all the scalar objective functions are *harmonious*, namely when all the objectives are positively correlated (Purshouse and Fleming 2003), only a partial ordering can be induced without the preference structure of the decision maker.

5.1. *Problem Setting*

A multi-objective minimization problem can be defined as follows:

$$\begin{gathered} \min_{\mathbf{x}} \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})) \\ \text{subject to } \mathbf{x} \in S, \end{gathered} \tag{18}$$

where k is the number of objective functions, S, is the feasible set in decision space, $\mathbf{x}$ is the decision vector and, $f_i(\mathbf{x})$, are scalar objective functions. Let $\mathbf{F} : S \to Z$, namely the forward image of the objective function, then the set, Z, is the feasible set in objective space. The $\min_{\mathbf{x}} \mathbf{F}(\mathbf{x})$ notation is interpreted as: minimise the vector valued function $\mathbf{F}(\mathbf{x})$ over all $\mathbf{x} \in S$ and should not be confused with the min operator which returns the minimum element of a set. It should also be noted that this definition could be used to describe a constrained or

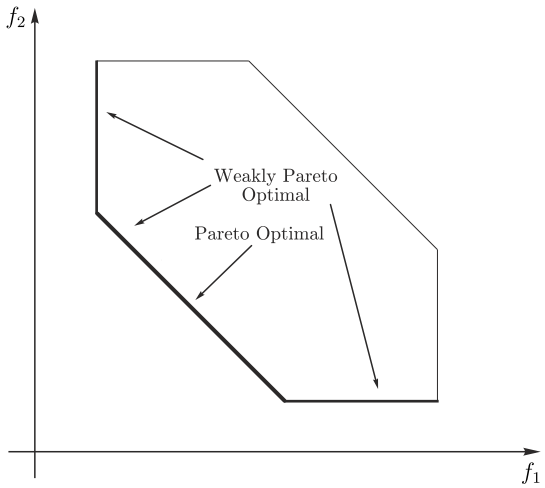


Figure 10.: Pareto optimal set and weakly Pareto optimal set. Note that the Pareto optimal set is a subset of the weakly Pareto optimal set.

unconstrained minimization problem depending on how S is defined, see (5). Further the fact that minimization is assumed is not restrictive because the problem of maximising $-f$ is equivalent to the problem of minimising f and vice versa. In the special case where $k = 1$, (18) becomes a single objective minimization problem. It is implicitly assumed that the scalar objective functions are mutually competing and perhaps are incommensurable while the goal is to minimise all of them simultaneously. If this is not the case then no special treatment is needed since minimising one of the scalar objective functions automatically results in minimization of the rest.

5.2. *Definitions*

The problem that arises in MOPs is that direct comparison of two objective vectors is not as straightforward as in single objective problems. In single objective problems when both $\mathbf{x}, \tilde{\mathbf{x}} \in S$ and $f(\tilde{\mathbf{x}}) < f(\mathbf{x})$ it is clear[1] that the decision vector $\tilde{\mathbf{x}}$ is superior to $\mathbf{x}$. This is not the case when two or more objectives are considered simultaneously and there exists no *a priori* preference toward a particular objective.

If the relative importance of the objectives is unspecified, one way to partially order the objective vectors, $\mathbf{z} \in Z$, is to use the Pareto[2] dominance relations, initially introduced by Edgeworth (1881) and further studied by Pareto (1896). Specifically, in a minimization context, a decision vector $\tilde{\mathbf{x}} \in S$ is said to be **Pareto optimal** if there is no other decision vector $\mathbf{x} \in S$ such that $f_i(\mathbf{x}) \leq f_i(\tilde{\mathbf{x}})$, for all i, and, $f_i(\mathbf{x}) < f_i(\tilde{\mathbf{x}})$ for at least one $i = 1, \ldots, k$. Namely there exists no other decision vector that maps to a clearly superior objective vector. Similarly, a decision vector $\tilde{\mathbf{x}} \in S$ is said to be **weakly Pareto optimal** if there is no other decision vector $\mathbf{x} \in S$ such that $f_i(\mathbf{x}) < f_i(\tilde{\mathbf{x}})$ for all $i = 1, \ldots, k$, see Fig. (10). Furthermore, a decision vector $\tilde{\mathbf{x}} \in S$ is said to **Pareto-dominate** a decision vector $\mathbf{x}$ iff $f_i(\tilde{\mathbf{x}}) \leq f_i(\mathbf{x})$, $\forall i \in \{1, 2, \ldots, k\}$ and $f_i(\tilde{\mathbf{x}}) < f_i(\mathbf{x})$, for at least one $i \in \{1, 2, \ldots, k\}$ then $\tilde{\mathbf{x}} \preceq \mathbf{x}$. So, in terms of generalised inequalities, if $\mathbf{F}(\tilde{\mathbf{x}}) \preceq \mathbf{F}(\mathbf{x})$ and $\mathbf{F}(\tilde{\mathbf{x}}) \neq \mathbf{F}(\mathbf{x})$, then $\tilde{\mathbf{x}} \preceq \mathbf{x}$. Also, a decision vector $\tilde{\mathbf{x}} \in S$ is said to **strictly dominate**, in the Pareto sense, a decision vector $\mathbf{x}$ iff $f_i(\tilde{\mathbf{x}}) < f_i(\mathbf{x})$, $\forall i \in \{1, 2, \ldots, k\}$ then $\tilde{\mathbf{x}} \prec \mathbf{x}$. That is, if $\mathbf{F}(\tilde{\mathbf{x}}) \prec \mathbf{F}(\mathbf{x})$, then $\tilde{\mathbf{x}} \prec \mathbf{x}$. It should be noted at this point, that when $\prec, \preceq$ are used in decision space, their meaning is mostly symbolic and is used to reflect the dominance relations in the objective space. The Pareto dominance relations with respect to a

[1]For a minimization problem.
[2]Referred to as Edgeworth-Pareto dominance relations by some authors.

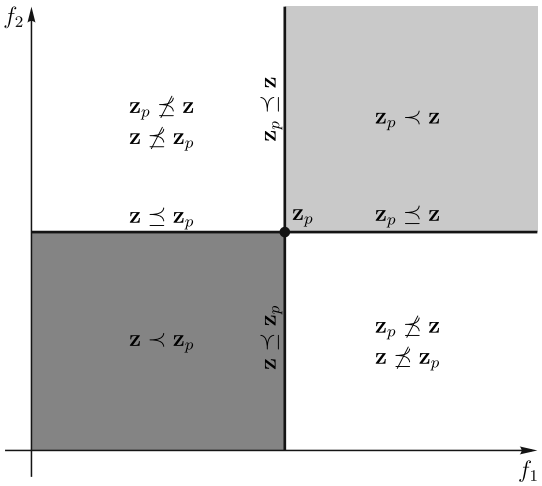


Figure 11.: Pareto dominance relations in the objective space. The dark gray region contains clearly better solutions compared with $\mathbf{z}_p$, the light gray region clearly worse solutions, while, the two other regions contain solutions incomparable to $\mathbf{z}_p$.

solution in objective space, $\mathbf{z}_p$, are depicted in Fig. (11).

Most multi-objective problem solvers attempt to identify a set of Pareto optimal solutions, this set is a subset of the **Pareto optimal set** (PS) which is also referred to as **Pareto front** (PF). The Pareto optimal set is defined as follows: $\mathcal{P} = \{\mathbf{z} : \nexists\, \tilde{\mathbf{z}} \preceq \mathbf{z}, \forall\, \tilde{\mathbf{z}} \in Z\}$. The decision vectors that correspond to the set, $\mathcal{P}$, are also called the Pareto set and are denoted as $\mathcal{D}$, namely $\mathbf{F} : \mathcal{D} \rightarrow \mathcal{P}$. The most widely used measures of the *quality* of the obtained Pareto front are the generational distance (GD), its inverted version (IGD) (Van Veldhuizen 1999) and the hypervolume indicator (Zitzler and Thiele 1999). A detailed discussion of these and other metrics is beyond the scope of this work, however there is a number of excellent works that address this issue, for example Zitzler et al. (2003), Zhou et al. (2011).

6. Methods for Extending PBOTs to MOPs

In what follows the main methodologies used to extend population-based optimization techniques to address multi-objective problems are discussed. These are separated into 3 broad categories, namely:

- Pareto-Based Methods

  Pareto-based methods employ Pareto-dominance relations to evaluate the quality of the population. These methods are still used to this day, however it would appear that their ability to handle problems with more than 3 objectives (many-objective problems) is somewhat limited (Ishibuchi et al. 2008).
- Decomposition-Based Methods

  Decomposition methods employ a scalarizing function and a set of weighting vectors to *decompose* a multi-objective problem into a set of single objective subproblems. Upon solution of this set of subproblems it is hoped that a good approximation of the Pareto front is obtained. There is evidence to suggest that this way of dealing with multi-objective problems is much more scalable for many-objective problems. However, there are still difficulties to be resolved for this type of methods. For example the distribution of solutions on the Pareto front is controlled by the selection of weighting vectors. This issue is currently being addressed (Jiang et al. 2011a,b, Gu et al. 2012, Giagkiozis et al. 2012, 2013) but there are still many unresolved questions.

- Indicator-Based Methods
  This type of methods for multi-objective problems is also promising, for example see (Wang et al. 2012), and are based on metrics developed to measure the *quality* of the solution set obtained from a PBOT. The most prevalent of these indicators has been the hypervolume indicator which was introduced in the context of multi-objective optimization by Zitzler and Thiele (1999).

Schaffer (1985) was the first to extend GAs to become multi-objective problem solvers. In retrospect, his approach may appear simple, but for its time it was quite a conceptual leap since VEGA considered all objectives simultaneously. VEGA, see *Alg. 4*, partitioned the population $\mathbf{x}$ into k equally sized randomly selected sub-populations, where k is the number of objectives. Then each partition is assigned to an objective and individuals are ranked according to their performance for the corresponding objective and a mating pool is formed using proportionate selection as described in (Goldberg 1989). Crossover and mutation operate on the entire popula-

Algorithm 4 VEGA

$\mathbf{X}^{(1)} \leftarrow Initialize$
$G \leftarrow 1$
$ps \leftarrow N/k$ ▷ Partition size
repeat
  $\mathbf{X}^{(G)} \leftarrow \text{Shuffle}_{rows}(\mathbf{X}^{(G)})$
  for $i \leftarrow 1, k$ **do** ▷ Partition the population
    $D_i \leftarrow \mathbf{X}^{(G)}_{ps\times(i-1)+1\rightarrow i\times ps}$
    $E_i \leftarrow \text{Evaluate}(D_i)$
    $S_i \leftarrow \text{Proportionate Selection}(E_i, D_i)$
  end for
  $\tilde{\mathbf{X}}^{(G)} \leftarrow Join(S_{1,\ldots,k})$
  $\tilde{\mathbf{X}}^{(G)} \leftarrow \text{Crossover}(\tilde{\mathbf{X}}^{(G)})$
  $\mathbf{X}^{(G+1)} \leftarrow \text{Mutate}(\tilde{\mathbf{X}}^{(G)})$
  $G \leftarrow G + 1$
until $G \leq MaxGenerations$

tion in the hope that linear combinations of the fitness functions would arise thus estimating the entire PF. This however was to some extent problematic since a phenomenon called *speciation* (Schaffer 1985) did on some occasions occur. For instance in objective functions with a concave PF, VEGA fails to approximate the PF as parts of the population drift toward the edges favouring one of the objective functions. Other methods employing the same or very similar approach are due to Fourman (1985) and Kursawe (1991), a variant applied to evolution strategies. A more recent work investigating VEGA is due to (Toroslu and Arslanoglu 2007).

6.1. *Pareto-Based Methods*

Fonseca and Fleming (1993) were the first to use Pareto dominance relations in a multiple objective genetic algorithm (MOGA), while incorporating progressive preference articulation enabling the decision maker to guide the search interactively. The dominance relation was used to rank the individuals in a population in a way similar to a set of selection methods proposed by Fourman (1985). Every individual in the population, after evaluation of the objective function, is ranked using the following relation,

$$r_i = 1 + p_i \tag{19}$$

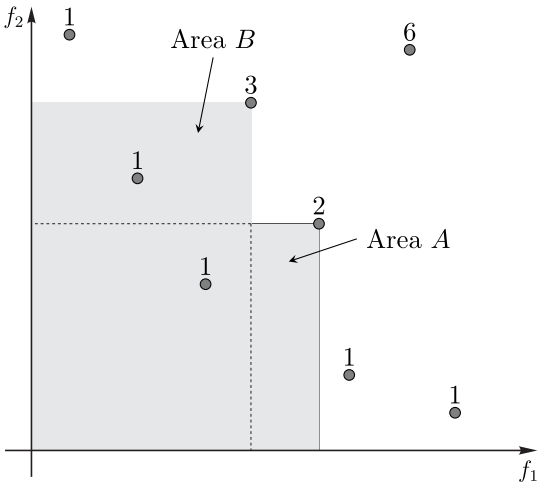


Figure 12.: Ranking method used in MOGA. The numbers above the points represent the rank of the individual that results in that objective vector. The worst rank possible is N.

where p_i is the number of individuals dominating the decision vector $\mathbf{x}_i$. The idea behind this method of ranking, see Fig. (12), is that misrepresented sections of the PF will increase the selection pressure to that direction of the front. As an example, the objective vector with rank 3 as seen in Fig. (12) is dominated by two individuals while the objective vector with rank 2 is dominated only by one. Alternatively, this means that within area A there is only one solution, while in area B there are two suggesting higher concentration of solutions. This information is used to induce better spread in the objective vectors that are not part of the current PF approximation so as to maintain a relatively even *supply* of objective vectors in all regions of the PF. For the objective vectors that are part of the PF approximation, Fonseca and Fleming introduced an adaptive factor, σ_{share} to penalize objective vectors that are less than σ_{share} apart. Lastly a facility for progressive preference articulation was embedded in MOGA providing the means for the DM to narrow down the search to *interesting* regions of the PF (Fonseca and Fleming 1993, 1995). This facility is implemented by incorporating a goal attainment method in the ranking procedure. Let $\mathbf{g} = \{g_1, \ldots, g_k\}$ be the goal vector and $\mathbf{z}_1 = \{z_1, \ldots, z_k\}$ and $\mathbf{z}_2$ be two objective vectors. A *preference* relation can be expressed such that when $k - s$ of the k objectives are met then $\mathbf{z}_1$ is *preferable* to $\mathbf{z}_2$ if and only if,

$$
\begin{aligned}
z_{(1,1\ldots k-s)} \preceq z_{(2,1\ldots k-s)} \text{ or } & \\
& \{(z_{(1,1\ldots k-s)} = z_{(2,1\ldots k-s)}) \wedge \\
& [(z_{(1,k-s+1\ldots k)} \preceq z_{(2,1\ldots k-s+1\ldots k)}) \vee \\
& \quad (z_{(2,k-s+1\ldots k)} \npreceq g_{(k-s+1\ldots k)})]\} \qquad (20)
\end{aligned}
$$

The rest of the MOGA utilized Gray encoding for the chromosomes, two point reduced surrogate crossover and the standard binary mutation.

Another prominent algorithm, the non-dominated sorting GA (NSGA), that utilized the Pareto dominance relations was proposed by Srinivas and Deb (1994). This method is almost identical to the non-dominated sorting idea proposed by Goldberg (1989). Non-dominated sorting is very similar to a well established concept in non-cooperative game theory in which candidates need to select a *winning* strategy while considering what their opponents' strategy might be. This idea is usually referred to as *eliminating dominated strategies* which is an iterative process of deleting dominated strategies that, if chosen, would lead to lower payoffs relative to the opponents and

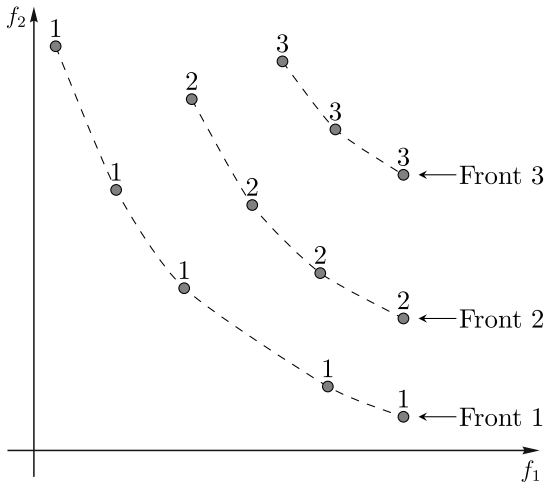


Figure 13.: Non-dominated sorting method as used in NSGA.

thus result in unfavourable outcomes. Interested readers are referred to (Bernheim 1984). Non-dominated sorting, see Fig. (13), works as follows:

Step 1 Evaluate the objective function for each individual in the population $\mathbf{X}$.

Step 2 Find the non-dominated individuals in the population, $\mathbf{X}_{nd}$, and remove them from the current population $\mathbf{X}_{new} = \{\mathbf{x} : \mathbf{x} \in (\mathbf{X} \cap \mathbf{X}_{nd})^c\}$.

Step 3 Repeat **Step 2** until no solutions remain in $\mathbf{X}_{new}$.

Each non-dominated set identified by this process is labelled as a *front* and given a rank according to the position it has in the iteration. For example, the first identified *front* would be *front 1*, the second *front 2* and so on. Subsequently the individuals of each front are assigned fitness values using the following procedure starting from the first *front*,

Step 1 Assign to all the individuals in *front 1* a fitness value N, where N is the number of individuals in the population.

Step 2 Use sharing to penalize clustered solutions in the current *front*.

Step 3 Assign the lowest fitness minus a small value ϵ to all individuals in the next *front* and go to **Step 2** until all individuals in all fronts have been assigned a fitness value.

The sharing distance σ_{share} is calculated based on distance in the decision variable space rather than the objective space and is held fixed throughout the algorithm execution. Finally NSGA employed the roulette-wheel selection operator (Goldberg 1989) and the usual bitwise mutation operator, see Fig. (2). The fact that NSGA uses sharing based on the distance of the decision vectors, seems counter intuitive since equally spaced decision vectors do not necessarily map to equally spaced objective vectors unless the mapping is affine which is usually not the case for most objective functions. This can potentially mislead the algorithm as to which solutions are densely clustered and which are not, resulting in exactly the opposite effect. Although, as the authors state, sharing can be performed based on the objective vectors (Srinivas and Deb 1994), they do not mention how σ_{share} would be selected in that case. Non-dominated sorting is the basis of an improved version of NSGA, namely NSGA-II (Deb et al. 2002), which is actively used for benchmaring and improved upon to this day, see (Zhang and Li 2007, Li and Zhang 2009, Ben Said et al. 2010, Wang et al. 2012, Bui et al. 2012).

One problem identified for Pareto-based methods for multi-objective problems, was that good solutions could be lost if they were not retained using some mechanism and the cost to rediscover them can potentially be prohibitive. This situation is exacerbated when the cost, computational or otherwise, of evaluating the objective function is high; this is often the case in most real-world applications. A solution to this predicament is the use of *elitism*, in other words the retention

of highly performing individuals in the population. Zitzler and Thiele (1999) introduced an algorithm named the strength pareto evolutionary algorithm (SPEA) that used elitism. Their approach was not the first to utilize an external archive to retain good performing individuals, however it was certainly one of the earliest and most elegant attempts.

SPEA, in addition to the current population $\mathbf{X}$, maintains an archive population $\tilde{\mathbf{X}}$ of maximum size $\tilde{N}$ which is usually smaller than the population size N. Zitzler and Thiele (1999) suggest $\tilde{N} = 0.25N$. At the start of the algorithm the external archive is empty and the population is initialized randomly within the decision variable limits or as appropriate to the problem. After the objective function has been evaluated the population is searched for non-dominated solutions which are copied to the archive $\tilde{\mathbf{X}} = \mathbf{X}_{nd}$. This direct assignment is performed only on the first iteration of the algorithm. In subsequent iterations the archive updating procedure is different if the total population in the archive exceeds $\tilde{N}$. When the archive size becomes larger than $\tilde{N}$, the superfluous solutions are removed based on a crowding algorithm that the authors of SPEA introduced. This crowding algorithm maintains the *elite* solutions that have maximal spread along the PF. After the archive size is properly reduced to its maximum size, the population and the *elite* individuals are assigned fitness values in the following way:

Step 1 Assign fitness values, called *strength* (s), to the population in the archive $\tilde{\mathbf{X}}$, using the following relation, $s_i = \frac{n_i}{N+1}$, where n_i is the number of individuals that the i^{th} solution in the archive dominates in the regular population $\mathbf{X}$.

Step 2 The regular population $\mathbf{X}$ is then assigned fitness values according to, $\mathfrak{f}_i = 1 + \sum_{\mathbf{x}_j \preceq \mathbf{x}_i} s_i$. Note that in SPEA lower fitness value is considered to be better. Therefore, if a decision vector $\mathbf{x}_i$ is not dominated by any individual in the population $\mathbf{X}$, its fitness, $\mathfrak{f}_i$, would be equal to 1.

The way the archive is maintained and used in SPEA ensures that the population $\mathbf{X}$ is rewarded when it approaches the PF. In a way the archive is used as a *moving target* to which the regular population aspires. Again the rest of the algorithm utilizes roulette wheel selection and random bit mutation.

6.2. *Decomposition-Based Methods*

As mentioned in Section 6, decomposition methods depend on scalarizing functions to break down a multi-objective problem into a set a single objective subproblems. The premise of this approach is that, the methods described in Section 4, can be applied almost unaltered. This benefit however, takes its toll as the selection of the weighting vectors controls the distribution of solutions on the Pareto front (Giagkiozis et al. 2012, 2013). This issue has been investigated in some detail in (Giagkiozis et al. 2013), where it is shown that the algorithm scalability to many-objective problems can be significantly affected if an arbitrary method is employed in the selection of the weighting vectors.

A family of scalarizing functions is the weighted metrics method (Miettinen and Mäkelä 2002):

$$\min_{\mathbf{x}} \left(\sum_{i=1}^{k} w_i |f_i(\mathbf{x}) - z_i^\star|^p \right)^{\frac{1}{p}}, \tag{21}$$

where $\mathbf{w} = (w_1, \ldots, w_k)$ is referred to as weighting vector and w_i are the weighting coefficients. The weighting coefficients can be viewed as factors of relative importance of the scalar objective functions in $\mathbf{F}(\cdot)$. The weighting coefficients must be $w_i \geq 0$ and $\sum_{i=1}^{k} w_i = 1$, also $p \in [1, \infty)$. However p is usually an integer or equal to ∞. A potential drawback of weighted metrics based scalarizing functions is that the ideal vector, $\mathbf{z}^\star$, has to be known *a priori*. However this vector

can be estimated adaptively during the process of optimization (Zhang and Li 2007). When $p = \infty$ the Chebyshev scalarizing function is obtained:

$$\min_{\mathbf{x}} \|\mathbf{w} \circ |\mathbf{F}(\mathbf{x}) - \mathbf{z}^{\star}| \, \|_{\infty}. \tag{22}$$

The $\circ$ operator denotes the Hadamard product which is element-wise multiplication of vectors or matrices of the same size. The key result that makes (22) very interesting is that for every Pareto optimal solution there exists a weighting vector with coefficients $w_i > 0$, for all $i = 1, \ldots, k$ (Miettinen 1999). Meaning that all Pareto optimal solutions can be obtained using (22). This result is quite promising, although in current practice the choice of weighting vectors is made primarily using *ad hoc* methods, see (Das and Dennis 1996, Jaszkiewicz 2002, Zhang and Li 2007), and so the points on the Pareto front cannot be controlled effectively.

The immune system based algorithm proposed by Yoo and Hajela (1999) used the weighting method (Miettinen 1999, pp. 27), which is obtained by setting $p = 1$ and $\mathbf{z}^{\star} = \mathbf{0}$ in (21). The weighting method aggregates all the objective functions into a single function using the following relation,

$$\begin{gathered} \min_{\mathbf{x}} \; g(\mathbf{x}) = \sum_{i=1}^{k} w_i f_i(\mathbf{x}) \\ \sum_{i=1}^{k} w_i = 1, \text{ and, } w_i \geq 0, \text{ for all } i \\ \text{subject to } \mathbf{x} \in S, \end{gathered} \tag{23}$$

where w_i represents weight of the i^{th} objective function. When all of the objective functions are equal in importance to the decision maker, the weights can be set to $w_i = \frac{1}{k}$, where k is the number of objective functions. Using the weighting method results in an approximate solution for one point of the PF. To explore more points, various combinations of the weights have to be used. Yoo and Hajela (1999) randomly generated a set of weight combinations in an attempt to uniformly sample the PF, and then used these different weighting vectors to guide the search.

Some difficulties encountered in the above implementation are that the weighting method is used and its weaknesses could affect the algorithm in several ways. For instance, evenly spaced weighting vectors do not necessarily produce an even distribution of points along the PF and two widely differing weight vectors need not produce significantly different points along the PF (Miettinen 1999). An additional shortcoming of the weighting method is that it cannot guarantee that all the Pareto optimal points can be obtained (Miettinen 1999, pp. 80). It is apparent, even in this early application of AIS to MOPs that there is a *jump* in algorithm complexity when compared to a GA. This complexity inevitably leads to an increased demand on computational resources. This cost will have to be justified especially since GAs, as can be seen in (Fonseca and Fleming 1993), can explicitly deal with constraints as well. However, the computational cost of most algorithms can often be ignored when compared with the cost of evaluating the objective function. Nevertheless this increase in algorithm complexity is not due to the use of the weighting method.

Parsopoulos and Vrahatis (2002) introduced a generalization of PSO to MOPs based on a methodology presented by Jin et al. (2001) using some variations of the weighting method, see (23). Jin et al. (2001) argued that weighting methods, or more specifically the conventional weighting method, could not approximate a concave PF due to the fact that the surface of a

concave PF is *unstable* and attracts solutions toward the two extremes[1] for which the weights are $(0, 1)$ and $(1, 0)$ respectively, this view is further supported in (Giagkiozis and Fleming 2012). However, in real world applications, the PF *shape* is usually not known a priori. To deal with this predicament Jin et al. (2001) suggested that if the optimiser is run with weights of one of the two extremities of the PF, that is either $(0, 1)$ or $(1, 0)$ and then gradually or abruptly the weighting vectors are exchanged the optimiser should be able to traverse the entire PF. To capture the Pareto optimal solutions Jin et al. suggested an archiving technique that would, hopefully, result in a good approximation of the entire PF. The problem with this technique is that the optimiser is effectively segmented in two phases, the first one with the fixed weighting vectors and the second one where the population is allowed to *slide* along the PF. To successfully accomplish this task, a metric is needed to measure the convergence rate of the first phase so that the second phase is initiated. This task is not trivial because it presupposes that the minimum is already known. Another potential problem is that if disjoint regions in the decision variable space map to neighbouring objective values, this approach will have difficulty approximating the PF. Lastly, it is difficult to envisage how would this method scale to problems with more than 2 objectives. Despite these difficulties, the aforementioned approach seems to perform reasonably well for the test problems used in (Parsopoulos and Vrahatis 2002, Jin et al. 2001).

As mentioned in Section 3, the main algorithm in PBOTs is usually built with single objective optimization in mind and as such the extension to multiple-objectives, as it will be apparent by now, requires a number of considerations. For instance, diversity preserving operations and *eliteness* preserving strategies inevitably result in higher computational costs. And for algorithms utilizing directly the non-dominated sorting strategy (see Section 6.1), the cost is even higher. Relatively recently a multi-objective evolutionary algorithm based on decomposition (MOEA/D) was introduced by Zhang and Li (2007) as an alternative way of extending DE and evolutionary algorithms (EAs), in general, to deal with MOPs. The approach depends on one of several available decomposition techniques, - weighted sum, Chebyshev (Miettinen 1999) and normal boundary intersection (Das and Dennis 1996) decompositions - with each having its own strengths and weaknesses. The minimization problem from Section 5.1, when using the Chebyshev decomposition, can be restated as follows,

$$\begin{aligned} \min_{\mathbf{x}} g_{\infty}(\mathbf{x}, \mathbf{w}^s, \mathbf{z}^{\star}) &= \max_{i=1\ldots k} (w_i^s | f_i(\mathbf{x}) - z_i^{\star}|) \\ &\text{for all } s = \{1, \ldots, N\}, \\ &\text{subject to } \mathbf{x} \in S, \end{aligned} \tag{24}$$

where $\mathbf{w}^s$ are N evenly distributed weighting vectors. The idea behind this is that g_{∞} is a continuous function of $\mathbf{w}$ (Zhang and Li 2007). The authors' hypothesis was that for N evenly distributed weighting vectors the same number of single objective sub-problems is generated and their simultaneous solution should result in evenly distributed Pareto optimal points, assuming the objectives are normalized (Zhang and Li 2007). MOEA/D has been very successful, and an updated version was the winner of the CEC'09 competition for unconstrained problems (Zhang et al. 2009).

6.3. *Indicator-Based Methods*

The need for comparative analyses regarding the strengths and weaknesses of different algorithms led to the introduction of several *indicators*, see (Zitzler et al. 2003), measuring various *qualities* of the resulting Pareto optimal set approximation. With the advent of such indicators, some

[1] In the case of two objectives.

researchers grasped the opportunity to utilize these within the evolution process, see (Zitzler and Künzli 2004, Wagner et al. 2007). One important indicator is a derivative of the Lebesgue measure (Halmos 1974), initially introduced in this context by (Zitzler and Thiele 1999); this indicator is now commonly referred to as the hypervolume indicator (Bader and Zitzler 2008). The most favourable quality of the hypervolume indicator is its monotonicity with regard to Pareto dominance (Bader and Zitzler 2008). This implies that the hypervolume indicator can be used in place of Pareto dominance relations for fitness assignment. However, this indicator is not without drawbacks since most known algorithms to calculate the Lebesgue measure are of exponential complexity with respect to the number of objectives (Bader and Zitzler 2008). A limitation that imposes a restriction to the number of objectives that can be considered in an algorithm based on this indicator.

Recently Bader and Zitzler (2008, 2011) introduced an algorithm based on the hypervolume indicator for many objective optimization[1]. Their idea is that precise calculation of the hypervolume is not required, per se, to enable a useful ranking of the population. Under this premise they developed a methodology which uses Monte Carlo sampling to approximately determine the value of the hypervolume and subsequently use this to assign fitness to the decision vectors. The results produced were quite favourable for HypE and the authors demonstrated that their proposed technique can effectively tackle as many as fifty objectives (Bader and Zitzler 2008).

6.4. *Commentary on the Methods*

Pareto dominance-based methods have been heavily criticized for their apparent inability to scale *gracefully* for an increasing number of objectives (Teytaud 2006, Ishibuchi et al. 2008, Hadka and Reed 2012). The main arguments are that the number of non-dominated solutions increases to a degree that any comparison becomes meaningless and there is no clear way for the search to continue (Ishibuchi et al. 2008). This view is further supported by theoretical arguments (Teytaud 2006, Giagkiozis and Fleming 2012). This, of course, is only true when all the scalar objectives are competing, otherwise increasing the number of objectives may even render the problem easier to solve (Scharnow et al. 2004, Brockhoff et al. 2007). Nevertheless, this issue is not exclusive to Pareto-based methods, rather it is a problem that all methods encounter, albeit to varying degrees (Giagkiozis and Fleming 2012). Another issue with Pareto-based methods is that of controlling the distribution of solutions on the Pareto front. This is one of the reasons for the introduction of modifications to the Pareto-dominance definition such as ε-dominance (Laumanns et al. 2002), cone ε-dominance (Batista et al. 2011) and other methods (Zitzler et al. 2001, Purshouse and Fleming 2007, Adra and Fleming 2009). All these additions to the main algorithm can become quite costly as the number of objectives increases. An alternative is the use of decomposition-based methods, which transform this into a different problem - that of selecting the weighting vectors to produce a desired distribution (Jiang et al. 2011b, Tan et al. 2012, Gu et al. 2012, Giagkiozis et al. 2012, 2013). In contrast, the benefit of indicator-based methods is that if an indicator that reflects well the preferences of the analyst and the decision maker is identified, then no additional considerations are necessary as the algorithm is using this metric directly. This, however, incurs a high computational cost (Zitzler et al. 2003, Bader and Zitzler 2008, Bader 2010) and it is unclear how the preferences of the decision maker can be incorporated.

Name	Language	License	GA	ES	AIS	ACO	DE	PSO	EDA	Reference
ECJ	Java	AFLv3	✓	✓	×	×	✓	✓	×	Panait et al. (2010)
Opt4J	Java	LGPL	✓	×	×	×	✓	✓	×	Lukasiewycz et al. (2011)
JGAP	Java	GPL	✓	×	×	×	×	×	×	Meffert and Rotstan (2010)
EvA2	Java	LGPL	✓	✓	×	×	✓	✓	×	Kronfeld et al. (2010)
jMetal	Java	LGPL	✓	✓	×	×	✓	✓	×	Durillo and Nebro (2011)
EO	C++	LGPL	✓	✓	×	×	✓	✓	✓	Keijzer et al. (2002)
MOMH Lib++	C++	LGPL	✓	×	×	×	×	×	×	Jaszkiewicz and Dabrowski (2012)
Open Beagle	C++	LGPL	✓	✓	×	×	×	×	×	Gagné and Parizeau (2006)
ParadisEO	C++	CeCILL	✓	✓	×	×	✓	✓	✓	Liefooghe et al. (2007)
Shark	C++	GPL	×	✓	×	×	×	×	×	Igel et al. (2008)
PISA	C	Mixed	✓	×	×	×	×	×	×	Bleuler et al. (2003)
Heuristic Lab	C#	GPL	✓	✓	×	×	×	✓	×	Wagner et al. (2010)

Table 1.: Software libraries for population-based optimization methods.

7. Software Libraries for Population-Based Optimization

There is a number of available software libraries for most of the algorithm families discussed in this work, however, as can be seen from Table 1, some algorithms are far better supported than others. The most striking observation in Table 1 is that it appears that no library, at least in the set of libraries we include here, has support for artificial immune systems or ant colony optimization algorithms. This does not mean that there is absolutely no open source software available for these methods, it simply means that popular libraries have not included them explicitly. In our view, the reason for this is that ACO is closely linked with the problem representation which creates difficulties when creating an abstraction in order for it to be included in a library. Nevertheless, there are several implementations available, see Dorigo et al. (2013). No freely available implementation of AIS could be located; this is consistent with the observation that there are relatively few publications on this algorithm family, see Fig. (14).

Although every library listed in Table 1 has its merits, jMetal (Durillo and Nebro 2011), PISA (Bleuler et al. 2003) and ParadisEO (Liefooghe et al. 2007) seem to be the most widely used by the community.

8. Discussion

There is little doubt that evolutionary algorithms have progressed with great pace over the last 30 years, see Fig. (14). However, there is little advice available to the practitioner faced with the challenge of choosing a suitable algorithm. In a way this choice is a multi-objective problem in itself, and quite a challenging one. Nevertheless, the fact that there is an absence of concrete guidelines is not accidental and can be attributed to several factors. For instance, not all the methodologies mentioned are easy to study analytically and, thus a rigorous and conclusive comparison is rather difficult. Also most studies are performed on test problems such as the ZDT problems (Zitzler et al. 2000), the DTLZ problems (Deb et al. 2002) or the WFG problem set (Huband et al. 2006) and others (Li and Zhang 2009). Furthermore it is rare to find a real-world application in the literature that encompasses the true complexity of problems encountered in practice (Michalewicz 2012). There is no dispute over the point that there are reported applications in the literature, for example see Zhou et al. (2011). However the scale of such problems pales in comparison to the size of problems that interior point methods can address, to the extent that Michalewicz (2012) calls them *toy problems*. To give a perspective of the size of this gap, consider that interior point methods can solve problems with several

[1]Many meaning more than three in this context.

	Algorithm Family						
Features	GA	ES	AIS	ACO	DE	PSO	EDA
Continuous	5	5	2	1	5	5	4
Discrete	5	1	5	5	2	2	5
Mixed	1	1	-	1	1	1	1
Combinatorial	2	3	3	5	2	2	4
Complexity	4	4	3	2	5	4	1
Cost	4	4	3	3	5	5	1
P.Sensitivity	2	3	2	3	4	3	4
No of Parameters	2	3	2	3	4	4	5
MOS	5	4	4	2	5	3	2
Prior Information	2	2	3	4	2	2	5

Table 2.: Relative strengths and weaknesses of algorithm families for MOPs. The values in the table are relative to other algorithm families compared with the support for the particular feature. A value of 5 translates that this particular family of algorithms is very well suited for this type of problem or in general that support for a feature is strong. A "-" signifies that there is no support for a particular feature.

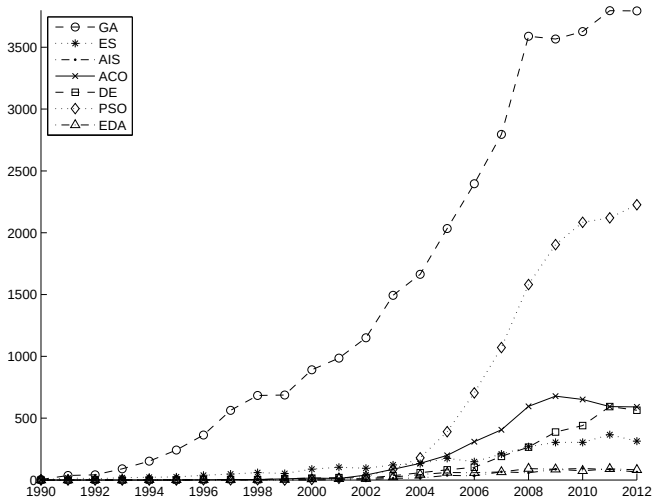


Figure 14.: Number of publications per year for each algorithm family. Results obtained from ISI Web of Knowledge. The queries used to obtain the number of papers can be seen in Appendix A.

million decision variables, constraints and objectives, while their convergence rate remains almost unaffected (Gondzio 2012), while to this day evolutionary algorithms mostly deal with problems with 2 or 3 objectives with relatively few addressing more, but no more than approximately 10, and with fewer than 100 decision variables (Kim et al. 2012, Chiong and Kirley 2012, de Lange et al. 2012, Wei et al. 2012, Tan et al. 2012). Moreover there is not a single study available[1] that establishes a strong positive correlation between superior algorithm performance on test problems and real-world (and real-scale) problems. However it would be an unfair statement to say that the community is unaware of these issues. A strong indication of this awareness, and a positive outcome in our view, is the development of evolutionary algorithms based on surrogate models (Poloni et al. 2000, Jin et al. 2002, Ong et al. 2006, Adra et al. 2009, Lim et al. 2010). The main difference of such algorithms is that they mostly operate using a model of the objective function, which is potentially very expensive to evaluate. This approach is taken a step further

[1]To the authors' best knowledge.

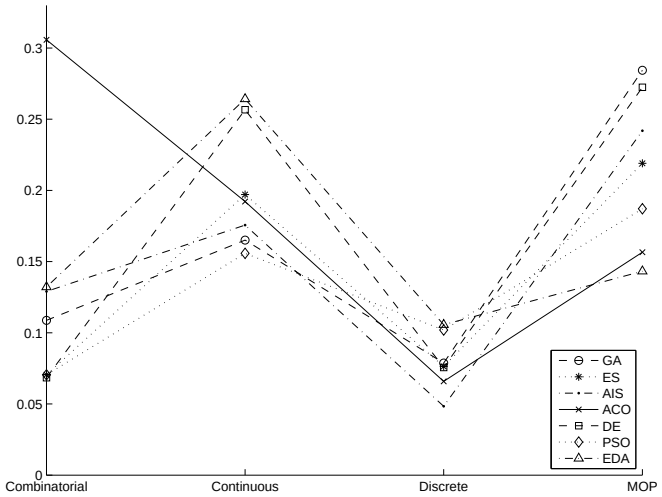


Figure 15.: Proportion of papers released with mention to a specific feature in all the algorithm families. These results where extracted from ISI Web of Knowledge and the employed queries can be found in Appendix A.

by using a convex surrogate model (Wanner et al. 2008, da Cruz et al. 2011). The benefits of such an approach should be obvious, in light of our discussion on interior point methods. Some may argue that by using such a simple objective function, a quadratic in the case of Wanner et al. (2008), much is lost in terms of fidelity of the solution. However, if such a method is combined with the scalability of interior point methods, the fact that a simple model is used is overshadowed by the potentially much larger scalability of the resulting algorithm which would allow the practitioner to consider the problem in its entirety and not just a small part of it. This in turn may help lessen the gap between theory and reality that (Michalewicz 2012) is talking about. Nevertheless, not much work has been done toward the direction proposed by Wanner et al. (2008) so its merits are yet to be evaluated in practice. The algorithms we mention in this paragraph are on some occasions referred to as memetic algorithms (MAs), and their main characteristic is that they incorporate some form of local search - often gradient based - to improve solutions. For an excellent survey of this topic the reader is referred to Chen et al. (2011).

In this section, as an initial guide to the practitioner, we summarize, and to some extent justify, the advantages and disadvantages of the algorithms: see Table 2. Although every possible effort has been made to validate the features and their corresponding ratings in Table 2, the ratings presented inevitably have some bias introduced by the authors. This fact is ascribed to the sheer volume of material available[1] relating to population-based multi-objective optimization algorithms, rendering an attempt to thoroughly compare the different algorithms extremely difficult, if not impossible.

The basis for the assigned ratings in Table 2 is now addressed. Factors that influenced these ratings are the following:

- Degree of Support for a Feature

  The degree of support for a feature is determined by how well established it is for algorithms in a particular family. By "established", we mean that the research community consistently improves upon the feature and that this process, relative to the study interval (past 30 years), is not entirely novel. For example, AIS have a score of 2 for continuous

[1]According to ISI Web of Knowledge, from 1990 to 2012 there have been published approximately 55 000 papers relating to one or more of the seven algorithm families discussed in this work.

problems since this extension is very recent (Castro and Von Zuben 2010a,b) and involves hybridization of the underlying algorithmic framework. For the same reason, all algorithmic families score poorly for mixed type decision variables (Table 2: Mixed), since systematic studies on this topic are extremely scarce.

- Reported Performance

  Another influencing factor is the reported performance, within the body of references in this work, of individual methods. For example, there is a clear way to *bias* the model building process in EDAs so as to encompass prior information when compared with any of the other algorithm families, hence they are assigned a top score in Table 2: Prior Information. To some extent, this argument is also true for ACO. Also, for combinatorial, continuous, discrete and mixed type problems we used the queries in Appendix A to find the proportion of published work that addresses these problem types for each algorithm family, Fig. (15); this was also taken into account as it is an indicator of the research focus of the community. Our assumption is that, if the proportion of the research output in a family of algorithms is higher for a particular class of problems there is a better chance for practitioners to identify a work that addresses a similar problem and that the potential for new positive developments is also increased.

Problem Types

Optimization problems can be broadly placed in four categories:

- Continuous - Problems with continuous decision variables.
- Discrete - Problems with discrete decision variables.
- Mixed - Problems with one or more discrete decision variables with the remaining variables being continuous.
- Combinatorial

The algorithm families reviewed in this work support these categories in varying degrees. In what follows, we provide a commentary on their support for these problem types.

Continuous

ES has supported this type since its inception (Rechenberg 1965), the same is true for PSO (Eberhart and Kennedy 1995) and DE (Storn and Price 1995), while GAs have been extended by means of several crossover and mutation operators for continuous problems (Deb and Agrawal 1994, Ono and Kobayashi 1997, Voigt et al. 1995). EDA is ranked at 4 since its support for real decision variables is relatively recent (Bosman and Thierens 2000, Thierens and Bosman 2001). AIS-based algorithms have adopted continuous decision variables quite recently (Castro and Von Zuben 2010a,b). However, the employed methodology resembles EDA algorithms since a Gaussian network model (Castro and Von Zuben 2010a) is created to enable the underlying AIS method deal with this type of decision variable. This means that the method in Castro and Von Zuben (2010a) is in essence a hybrid of two algorithmic families, namely AIS and EDA. Admittedly the reported performance in (Castro and Von Zuben 2010b) is comparable with, and in some instances better than, the compared algorithms. For this reason, AIS score 2 in Table 2, since their ability to handle continuous decision variables is much better than ACO, but research involving AIS with continuous decision variables is quite sparse. ACO has relatively poor support for this type (Bilchev and Parmee 1995, Socha and Dorigo 2008).

Discrete

GAs initially used discrete representation (Holland 1975) as well as AIS (Farmer et al. 1986), EDAs (Baluja and Caruana 1995) and ACO (Colorni et al. 1991). ES, DE and PSO have been developed for problems with continuous decisions variables (Storn and Price 1995, Eberhart and

Kennedy 1995), however some extensions exist that extend these classes of algorithms to tackle problems requiring discrete decision variables, see (Pan et al. 2007) for DE, (Cai and Thierauf 1996) for ES and (Kennedy and Eberhart 1997) for PSO.

Mixed

Tracking published resources in this category proved challenging. Although, in the authors' experience, mixed type decision variables are often necessary for real world applications, the literature is very scarce on this topic. This fact is reflected in the relative rankings, which, for all algorithmic families, are the same. One extension of ACO to mixed type decision variables is presented in Socha (2004), and an application using EDAs in Ocenasek and Schwarz (2002). Regarding ES there are some studies dealing with mixed-integer decision variables (Bäck et al. 1995, Emmerich et al. 2000), however the bulk of the research is directed toward real decision variable problems. Additionally some examples in DE and PSO are Pampara et al. (2006), Lampinen and Zelinka (1999) and Gaing (2005) respectively. GAs have the potential to excel here as the abstraction used to represent decision variables (chromosome) can be intuitively extended.

Combinatorial

To some extent all the studied algorithmic families support or have the ability to tackle combinatorial problems, although ACO has a strong lead in this type of problems (Bell and McMullen 2004). Also ACO papers consistently feature in the top 10 most cited papers on combinatorial optimization problems (SCOPUS 2012). EDAs are ranked 4 only because there seems to be a smaller body of research compared with ACO in this category. However EDAs are gaining ground quite rapidly (SCOPUS 2012). Some exemplars of successful applications can be found in (Larranaga and Lozano 2002, Hauschild and Pelikan 2011). Regarding the rest of the families and considering the fact that most are employed in combinatorial problems in a hybrid form with local search techniques (Puchinger and Raidl 2005, Hertz and Widmer 2003), ES and AIS are given a slight advantage due to their increased use (SCOPUS 2012). However this point is debatable since it has been shown (Zhang and Li 2007), that a framework based on decomposition, which can utilize GAs or DEs as its main search algorithm, does perform admirably on multi-objective knapsack problems.

Applicability

Complexity - In Terms of Implementation

By far the easiest techniques to implement are DE (Storn and Price 1995) and PSO (Eberhart and Kennedy 1995) followed by ES (Rechenberg 1965), GAs and AIS. ACO is given a complexity of 2 due to the fact that the practitioner usually has to adapt the algorithmic process significantly to solve a particular problem (López-Ibáñez and Stützle 2010) and EDAs are deemed to be the most complex to implement since quite elaborate techniques are required (Hauschild and Pelikan 2011). It should be noted that these ratings apply to the *main algorithm* and not to extensions to MOPs since the total complexity would be greatly affected by the selected methodology.

Cost - In Terms of Required Resources

This more or less mirrors the implementation complexity which is fairly reasonable. Although it should be stated that EDAs have much lower memory requirements (Hauschild and Pelikan 2011), their computational cost per iteration is higher in comparison with the other of the algorithm families. This is especially true when more elaborate probabilistic models are created, which is the case for hierarchical-BOA (Pelikan et al. 2005) and RM-MEDA (Zhang et al. 2008). DE and PSO have the highest score, since their updating rules do not require elaborate calculations as can be seen in Section 4.5 and Section 4.6. Following these are ESs and GAs, which are

still relatively not very demanding. For example, ES use normal distributions which are more expensive to calculate compared with the small number of addition and multiplication operations in DE and PSO, and GAs have some crossover operators that can slightly increase their cost, for example Deb and Agrawal (1994).

Number of Parameters Requiring Selection

For this feature, we stress the point that the main algorithms are assessed and not their many extensions to MOPs, with or without constraints. Here DE (Storn and Price 1995), EDAs (Hauschild and Pelikan 2011) and ES (Rechenberg 1965) score highest due to their inherent adaptive nature. EDAs are awarded the top score since most of their updating rules are calculated during the algorithm execution based on some measure of optimality, and such measures can be found in abundance in statistics. For example, optimal updating rules can be formulated using the Kullback-Leibler divergence as is the case for the cross-entropy method (De Boer et al. 2005). Therefore, since there are ways to *optimally* update the *direction* of search, the need for controlling parameters is, relative to other families, less significant. (Add a sentence about GAs ES and ACO.)

Multi-Objective Scalability (MOS)

In this category, ACO is ranked 2 since it seems that extending ACO to more than 2 or 3 objectives is extremely difficult (López-Ibáñez and Stützle 2010) or at least no attempts have been brought forward so far[1]. This is also supported by the fact that articles for multi-objective optimization algorithms based on ACO do not appear in the top 20 cited papers (SCOPUS 2012). For the rest, a rank of 4 was given if the methodologies to extend a particular family have been relatively recently addressed; such is the case for EDAs, PSO and AIS. It seems natural that GAs and ES score highly here since their techniques have the longest history. Additionally, regarding EDAs, the problem is that the method used to extend this family to MO-problems affects the complexity of the probabilistic model. For instance, if a Bayesian network is used as in Pelikan and Sastry (2006), it is not very easy to envisage how this will be directly extended to many-objectives.

Use of Prior Information

The highest scores for this feature are assigned to ACO and EDA based algorithms. For EDAs this is due to the ease with which prior information can be incorporated (Hauschild and Pelikan 2011). This type of bias can be induced by altering the initial parameters of the probabilistic model (Hauschild and Pelikan 2011). ACO also has probabilistic transitional rules, hence to bias them towards promising regions is relatively straightforward and is common practice when the problem is formulated , for example, see López-Ibáñez and Stützle (2010). In the remainder of the reviewed algorithm families, prior information about a problem has to be embedded, usually on an ad-hoc basis. For example, a GA/DE based algorithm MOEA/D (Zhang and Li 2007), required significant changes in order to be applied to the multi-objective 0-1 knapsack problem.

9. Conclusion

It is hoped that Table 1 will provide the practitioner, an overview of algorithm capabilities for MOPs and enable them to select the most appropriate approach for their application. Additionally, as we mentioned in Section 8, there is some subjectivity involved in the creation of Table 1; hence it should be taken as an invitation to our fellow researchers for a debate, which we envisage would have two main outcomes, both of which are very important:

[1]To the authors' best knowledge.

- It will potentially change the relative importance ranks that we have assigned, something that would be beneficial to practitioners.
- It may raise awareness of research areas that deserve more attention.

The idea behind this approach is based in Finance, which financial engineers call *price discovery* (Fabozzi et al. 1994). In short, price discovery describes the process by which the market (buyers and sellers) determines the price of *securities* as a result of the *supply* and *demand* mechanism. However, the outlet for such a mechanism is also important as the rankings must be updated as often as possible. For this reason we are in the process of designing a web-site to host this mechanism so that it is available to everyone (Giagkiozis 2012).

Seven algorithmic families have been considered as separate entities, although it should be noted that clear boundaries cannot be explicitly drawn. For this reason researchers start naming algorithms that fall into these categories, Evolutionary Algorithms. This choice is further justified in Section 3. Diffusion of information between the studied methodologies is quite rapid and common. For instance, ES have come to bear a great resemblance to GAs. Both techniques employ mutation, recombination and selection and the methods for extending GAs to multiple objective problems have successfully been applied in ES as well as in EDAs, AIS, DE and PSO. Some recombination operators in GAs have similar features to the recombination operator in DE. ACO uses probabilistic transition rules in a similar fashion to EDAs, especially when ACO has continuous decision variables.

It is also evident that more and more researchers lean towards the development of algorithmic processes that exhibit strong adaptive behaviour and provide more information than just the PF approximation at the end of the optimization procedure. This is partly due to the fact that computational resources have become cheaper and more accessible and partly due to the ever increasing complexity of systems requiring more elaborate, effective and informative techniques. Additionally, as MOEA research moves toward many objectives, i.e. problems with more than three objectives, research is moving away from Pareto-based algorithms. This is mostly due to the difficulty that is posed in many dimensions for methods employing this ranking scheme (Purshouse and Fleming 2003, Hughes 2005). However there are still a number of issues with decomposition-based algorithms. For instance, as the number of dimensions increases the distance of the weighting vectors effectively increases, and recombination of neighbouring solutions become problematic as a result. Due to this problem, many solutions that are potentially Pareto-optimal can be lost or remain un-utilized. A suitable solution to this problem has not yet emerged and is a very interesting direction for future research. Several other approaches, that seem promising have emerged, for example cone ε-dominance (Batista et al. 2011) and δ-ball dominance (Chen et al. 2011). However these methods are quite novel and have not yet been tested for more than three objectives; further investigations are required to reveal their potential merit. Another promising research direction is that of many-objective optimization in the presence of noise, extending to time-varying problems and real-time optimization.

Arising from this overview, it becomes apparent that the current trend in population-based multi-objective algorithms is toward estimation of distribution algorithms and indicator based algorithms. It is the authors' view that a combination of these two methodologies could produce interesting and innovative approaches for many-objective problems. Additionally, more attention from the research community should be ascribed to problems with mixed type decision variables due to their frequent presence in real world problems. Another somewhat not fully developed aspect is notation, especially in regard to Pareto dominance relations, where there is incoherence in this field. Finally a unified mathematical framework seems possible and its development would be highly beneficial; some steps toward this direction can are seen in Laumanns et al. (2000), Liu et al. (2011).

Acknowledgment

The first author would like to thank Dr S.N. Walker for his insightful comments on this work.

Appendix A. Web of Knowledge Queries

Following are the queries used to obtain Fig. (14) and Fig. (15).

Genetic Algorithms

```
("genetic" AND ("algorithm" OR "algorithms")
AND ("optimization" OR "optimisation"))
```

Evolution Strategies

```
("evolution" AND ("strategy" OR "strategies")
AND ("optimization" OR "optimisation"))
```

Artificial Immune Systems

```
("artificial immune" AND ("system" OR "systems")
AND ("optimization" OR "optimisation"))
```

Ant Colony Optimization

```
(("ant colony" OR "ant system")
AND ("optimization" OR "optimisation"))
```

Differential Evolution

```
(("differential evolution")
AND ("optimization" OR "optimisation"))
```

Particle Swarm Optimization

```
("particle swarm"
AND ("optimization" OR "optimisation"))
```

For every algorithm family we made the search more specific linking the above queries and the following queries with an `AND`.

Continuous Problems

```
("continuous" OR "real"
OR "WFG" OR "DTLZ" OR "ZDT")
```

Discrete Problems

```
("discrete" OR "binary")
```

Combinatorial Problems

```
("combinatorial" OR "travelling salesman"
OR "traveling salesman" OR "vehicle routing"
OR "knapsack problem"   OR "assignment problem"
OR "scheduling problem" OR "city courier problem"
OR "land use planning")
```

Multi-Objective Problems

```
("multi objective"   OR "many objective"
OR "multiobjective"  OR "manyobjective"
OR "bi criterion"    OR "multicriterion"
OR "multi criterion" OR "multiple criteria")
```

References

Adra, S., and Fleming, P. (2009), "A diversity management operator for evolutionary many-objective optimisation," in *Evolutionary multi-criterion optimization*, Springer, pp. 81–94.

Adra, S., Dodd, T., Griffin, I., and Fleming, P. (2009), "Convergence Acceleration Operator for Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, 13(4), 825–847.

Akay, B., and Karaboga, D. (2012), "A modified artificial bee colony algorithm for real-parameter optimization," *Information Sciences*, 192, 120–142.

AlRashidi, M., and El-Hawary, M. (2009), "A survey of particle swarm optimization applications in electric power systems," *IEEE Transactions on Evolutionary Computation*, 13(4), 913–918.

Bäck, T., Schütz, M., Ack, T., and Utz, M. (1995), "Evolution Strategies for Mixed-Integer Optimization of Optical Multilayer Systems," *Evolutionary Programming*, p. 33.

Bäck, T., Hoffmeister, F., and Schwefel, H.P. (1991), "A Survey of Evolution Strategies," in *International Conference on Genetic Algorithms*, Morgan Kaufmann, pp. 2–9.

Bader, J., and Zitzler, E. (2008), "HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization," Technical report, Computer Engineering and Networks Laboratory (TIK), ETH Zurich.

Bader, J., and Zitzler, E. (2011), "HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization," *IEEE Transactions on Evolutionary Computation*, 19(1), 45–76.

Bader, J., *Hypervolume-Based Search for Multiobjective Optimization: Theory and Methods*, Johannes Bader (2010).

Baluja, S., and Caruana, R. (1995), "Removing the Genetics from the Standard Genetic Algorithm," in *Machine Learning International Workshop then Conference*, pp. 38–46.

Batista, L., Campelo, F., Guimarães, F., and Ramírez, J. (2011), "Pareto Cone ε-Dominance: Improving Convergence and Diversity in Multiobjective Evolutionary Algorithms," in *Evolutionary Multi-Criterion Optimization*, Springer, pp. 76–90.

Bell, J., and McMullen, P. (2004), "Ant Colony Optimization Techniques for the Vehicle Routing Problem," *Advanced Engineering Informatics*, 18(1), 41–48.

Ben Said, L., Bechikh, S., and Ghédira, K. (2010), "The r-dominance: a new dominance relation for interactive evolutionary multicriteria decision making," *Evolutionary Computation, IEEE Transactions on*, 14(5), 801–818.

Bernheim, B. (1984), "Rationalizable Strategic Behavior," *Econometrica*, pp. 1007–1028.

Beyer, H. (1997), "An Alternative Explanation for the Manner in Which Genetic Algorithms Operate," *BioSystems*, 41(1), 1–15.

Beyer, H., and Schwefel, H. (2002), "Evolution Strategies - A Comprehensive Introduction," *Natural computing*, 1(1), 3–52.

Bieniawski, S., Wolpert, D., and Kroo, I. (2004), "Discrete, continuous, and constrained optimization using collectives," in *Proceedings of 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, New York*, Citeseer.

Bieniawski, S. (2005), "Distributed optimization and flight control using collectives," stanford university.

Bilchev, G., and Parmee, I. (1995), "The ant colony metaphor for searching continuous design spaces," *Evolutionary Computing*, pp. 25–39.

Bleuler, S., Laumanns, M., Thiele, L., and Zitzler, E. (2003), "PISA — A Platform and Programming Language Independent Interface for Search Algorithms," in *Evolutionary Multi-Criterion Optimization (EMO 2003)*, Lecture Notes in Computer Science, Berlin: Springer, pp. 494 – 508.

Bosman, P., and Thierens, D. (2000), "Expanding from Discrete to Continuous Estimation of Distribution Algorithms: The IDEA," in *Parallel Problem Solving from Nature*, Springer, pp. 767–776.

Boyd, S., and Vandenberghe, L., *Convex Optimization*, Cambridge University Press (2004).

Brockhoff, D., Friedrich, T., Hebbinghaus, N., Klein, C., Neumann, F., and Zitzler, E. (2007), "Do additional objectives make a problem harder?," in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, ACM, pp. 765–772.

Bui, L., Abbass, H., Barlow, M., and Bender, A. (2012), "Robustness Against the Decision-Maker's Attitude to Risk in Problems With Conflicting Objectives," *Evolutionary Computation, IEEE Transactions on*, 16(1), 1–19.

Cai, J., and Thierauf, G. (1996), "Evolution Strategies for Solving Discrete Optimization Problems," *Advances in Engineering Software*, 25(2-3), 177–183.

Castro, P., and Von Zuben, F. (2010a), "GAIS: A Gaussian Artificial Immune System for Continuous Optimization," *Artificial Immune Systems*, pp. 171–184.

Castro, P., and Von Zuben, F. (2010b), "A Gaussian Artificial Immune System for Multi-Objective optimization in continuous domains," in *Conference on Hybrid Intelligent Systems*, IEEE, pp. 159–164.

Chen, X., Ong, Y., Lim, M., and Tan, K. (2011), "A multi-facet survey on memetic computation," *Evolutionary Computation, IEEE Transactions on*, 15(5), 591–607.

Chen, Y., Zou, X., and Xie, W. (2011), "Convergence of Multi-Objective Evolutionary Algorithms to a Uniformly Distributed Representation of the Pareto Front," *Information Sciences*, 181(16), 3336–3355.

Chiong, R., and Kirley, M. (2012), "Effects of Iterated Interactions in Multiplayer Spatial Evolutionary Games," *Evolutionary Computation, IEEE Transactions on*, 16(4), 537–555.

Choe, J., and Perlman, D. (1997), "Social Conflict and Cooperation Among Founding Queens in Ants (Hymenoptera: Formicidae)," *The evolution of social behavior in insects and arachnids*, p. 392.

Coello, C., and Cortés, N. (2002), "An Approach to Solve Multiobjective Optimization Problems Based on an Artificial Immune System," in *First International Conference on Artificial Immune Systems (ICARIS2002)*, pp. 212–221.

Coello, C., and Cortés, N. (2005), "Solving Multiobjective Optimization Problems Using An Artificial Immune System," *Genetic Programming and Evolvable Machines*, 6(2), 163–190.

Colorni, A., Dorigo, M., Maniezzo, V. et al.(1991), "Distributed optimization by ant colonies," in *Proceedings of the first European conference on artificial life*, Vol. 142, pp. 134–142.

da Cruz, A., Cardoso, R., Wanner, E., and Takahashi, R. (2011), "Using convex quadratic approximation as a local search operator in evolutionary multiobjective algorithms," in *Evolutionary Computation (CEC), 2011 IEEE Congress on*, IEEE, pp. 1217–1224.

Darwin, C., *The origin of species*, 811, Hayes Barton Press (1858).

Das, I., and Dennis, J. (1996), "Normal-Boundary Intersection: An Alternate Method for Generating Pareto Optimal Points in Multicriteria Optimization Problems," Technical report, DTIC Document.

Das, S., and Suganthan, P. (2010), "Differential evolution: A survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, (99), 1–28.

Dasgupta, D., and Attoh-Okine, N. (1997), "Immunity-Based Systems: A Survey," in *IEEE International Conference on Systems, Man, and Cybernetics*, Vol. 1, IEEE, pp. 369–374.

Dasgupta, D., and Forrest, S. (1999), "Artificial Immune Systems in Industrial Applications," in *International Conference on Intelligent Processing and Manufacturing of Materials*, Vol. 1,

IEEE, pp. 257–267.

De Boer, P., Kroese, D., Mannor, S., and Rubinstein, R. (2005), "A Tutorial on the Cross-Entropy Method," *Annals of Operations Research*, 134(1), 19–67.

de Lange, R., Samoilovich, I., and van der Rhee, B. (2012), "Virtual Queuing at Airport Security Lanes," *European Journal of Operational Research*.

Deb, K., and Agrawal, R. (1994), "Simulated binary crossover for continuous search space," *Complex Systems*, 50(9), 115–148.

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

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2002), "Scalable Multi-Objective Optimization Test Problems," in *Congress on Evolutionary Computation*, may, Vol. 1, pp. 825 –830.

del Valle, Y., Venayagamoorthy, G., Mohagheghi, S., Hernandez, J., and Harley, R. (2008), "Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems," *IEEE Transactions on Evolutionary Computation*, 12(2), 171–195.

Deneubourg, J., Aron, S., Goss, S., and Pasteels, J. (1990), "The self-organizing exploratory pattern of the argentine ant," *Journal of Insect Behavior*, 3(2), 159–168.

Dorigo, M., Birattari, M., and Stutzle, T. (2006), "Ant Colony Optimization," *IEEE Computational Intelligence Magazine*, 1(4), 28–39.

Dorigo, M., and Blum, C. (2005), "Ant colony optimization theory: A survey," *Theoretical computer science*, 344(2-3), 243–278.

Dorigo, M., and Di Caro, G. (1999), "Ant Colony Optimization: A New Meta-Heuristic," in *Congress on Evolutionary Computation*, Vol. 2, IEEE.

Dorigo, M., Maniezzo, V., and Colorni, A. (1991), "The ant system: An autocatalytic optimizing process," *TR91-016, Politecnico di Milano*.

Dorigo, M., Sttzle, T., Farooq, M., Blum, C. et al., "Ant Colony Optimization," (2013).

Durillo, J.J., and Nebro, A.J. (2011), "jMetal: A Java framework for multi-objective optimization," *Advances in Engineering Software*, 42(10), 760 – 771.

Eberhart, R., and Kennedy, J. (1995), "A New Optimizer Using Particle Swarm Theory," in *International Symposium on Micro Machine and Human Science*, IEEE, pp. 39–43.

Edgeworth, F., *Mathematical Psychics: An Essay on the Application of Mathematics to the Moral Sciences*, 10, CK Paul (1881).

Emmerich, M., Grötzner, M., Groß, B., and Schütz, M. (2000), "Mixed-Integer Evolution Strategy for Chemical Plant Optimization with Simulators," *Evolutionary Design and Manufacture-Selected papers from ACDM*, pp. 55–67.

Fabozzi, F., Modigliani, F., and Ferri, M., *Foundations of financial markets and institutions*, Vol. 3, Prentice Hall (1994).

Farmer, J., Packard, N., and Perelson, A. (1986), "The immune system, adaptation, and machine learning," *Physica D: Nonlinear Phenomena*, 22(1-3), 187–204.

Fonseca, C., and Fleming, P. (1993), "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization," in *Conference on Genetic Algorithms*, Vol. 423, pp. 416–423.

Fonseca, C., and Fleming, P. (1995), "Multiobjective Genetic Algorithms Made Easy: Selection Sharing and Mating Restriction," in *International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*, IET, pp. 45–52.

Fourman, M. (1985), "Compaction of symbolic layout using genetic algorithms," in *Proceedings of the 1st International Conference on Genetic Algorithms*, L. Erlbaum Associates Inc., pp. 141–153.

Gagné, C., and Parizeau, M. (2006), "Genericity in Evolutionary Computation Software Tools: Principles and Case Study," *International Journal on Artificial Intelligence Tools*, 15(2), 173–194.

Gaing, Z. (2005), "Constrained Optimal Power Flow by Mixed-Integer Particle Swarm Optimization," in *Power Engineering Society General Meeting*, IEEE, pp. 243–250.

Gasper, A., and Collard, P. (1999), "From GAs to Artificial Immune Systems: Improving Adaptation in Time Dependent Optimization," in *IEEE Congress on Evolutionary Computation*, Vol. 3, IEEE.

Giagkiozis, I., "http://ioannis-giagkiozis.staff.shef.ac.uk/," (2012).

Giagkiozis, I., and Fleming, P. (2012), "Methods for Many-Objective Optimization: An Analysis," Research Report No. 1030.

Giagkiozis, I., Purshouse, R., and Fleming, P. (2012), "Generalized Decomposition and Cross Entropy Methods for Many-Objective Optimization," Research Report No. 1029, Department of Automatic Control and Systems Engineering, The University of Sheffield.

Giagkiozis, I., Purshouse, R., and Fleming, P. (2013), "Generalized Decomposition," in *Evolutionary Multi-Criterion Optimization* Lecture Notes in Computer Science, Springer Berlin.

Goldberg, D. (1989), "Genetic Algorithms in Search, Optimization, and Machine Learning," .

Goldberg, D., and Holland, J. (1988), "Genetic Algorithms and Machine Learning," *Machine Learning*, 3(2), 95–99.

Gondzio, J. (2012), "Interior point methods 25 years later," *European Journal of Operational Research*, 218(3), 587–601.

Gu, F., Liu, H., and Tan, K. (2012), "A Multiobjective Evolutionary Algorithm Using Dynamic Weight Method," *International Journal of innovative Computing, Information and Control*, 8(5B), 3677–3688.

Hadka, D., and Reed, P. (2012), "Diagnostic Assessment of Search Controls and Failure Modes in Many - Objective Evolutionary Optimization," *Evolutionary Computation*.

Halmos, P., *Measure Theory*, Vol. 18, Springer (1974).

Hauschild, M., and Pelikan, M. (2011), "A Survey of Estimation of Distribution Algorithms," .

He, S., Wu, Q., and Saunders, J. (2009), "Group Search Optimizer: An Optimization Algorithm Inspired by Animal Searching Behavior," *IEEE Transactions on Evolutionary Computation*.

Hertz, A., and Widmer, M. (2003), "Guidelines for the use of Meta-Heuristics in Combinatorial Optimization," *European Journal of Operational Research*, 151(2), 247–252.

Holland, J. (1962), "Outline for a logical theory of adaptive systems," *Journal of the ACM (JACM)*, 9(3), 297–314.

Holland, J. (1975), "Adaptation in natural and artificial systems," .

Huband, S., Hingston, P., Barone, L., and While, L. (2006), "A Review of Multiobjective Test Problems and A Scalable Test Problem Toolkit," *IEEE Transactions on Evolutionary Computation*, 10(5), 477–506.

Hughes, E.J. (2005), "Evolutionary Many-Objective Optimisation: Many Once or One Many?," in *IEEE Congress on Evolutionary Computation*, Vol. 1, pp. 222–227.

Igel, C., Heidrich-Meisner, V., and Glasmachers, T. (2008), "Shark," *Journal of Machine Learning Research*, 9, 993–996.

Ishibuchi, H., Tsukamoto, N., and Nojima, Y. (2008), "Evolutionary Many-Objective Optimization: A Short Review," in *IEEE Congress on Evolutionary Computation*, june, pp. 2419 –2426.

Jaszkiewicz, A. (2002), "On the Performance of Multiple-Objective Genetic Local Search on the 0/1 Knapsack Problem - A comparative Experiment," *IEEE Transactions on Evolutionary Computation*, 6(4), 402–412.

Jaszkiewicz, A., and Dabrowski, G., "MOMH Multiple-Objective MetaHeuristics," `http://home.gna.org/momh/index.html` (2012).

Jiang, S., Cai, Z., Zhang, J., and Ong, Y.S. (2011a), "Multiobjective Optimization by Decomposition with Pareto-Adaptive Weight Vectors," in *International Conference on Natural Computation*, july, Vol. 3, pp. 1260 –1264.

Jiang, S., Zhang, J., and Ong, Y. (2011b), "Asymmetric Pareto-adaptive Scheme for Multiobjective Optimization," (Vol. 7106, Springer Berlin Heidelberg, pp. 351–360.

Jin, Y., and Branke, J. (2005), "Evolutionary Optimization in Uncertain Environments - A Survey," *IEEE Transactions on Evolutionary Computation*, 9(3), 303–317.

Jin, Y., Olhofer, M., and Sendhoff, B. (2001), "Dynamic Weighted Aggregation for Evolutionary Multi-Objective Optimization: Why Does it Work and How?," .

Jin, Y., Olhofer, M., and Sendhoff, B. (2002), "A Framework for Evolutionary Optimization with Approximate Fitness Functions," *IEEE Transactions on Evolutionary Computation*, 6(5), 481 – 494.

Karaboga, D., Ozturk, C., Karaboga, N., and Gorkemli, B. (2012), "Artificial bee colony programming for symbolic regression," *Information Sciences*.

Keijzer, M., Merelo, J., Romero, G., and Schoenauer, M. (2002), "Evolving Objects: A General Purpose Evolutionary Computation Library," (Vol. 2310, Springer Berlin Heidelberg, pp. 231–242.

Kennedy, J., and Eberhart, R. (1995), "Particle Swarm Optimization," in *IEEE International Conference on Neural Networks*, Vol. 4, IEEE, pp. 1942–1948.

Kennedy, J., and Eberhart, R. (1997), "A Discrete Binary Version of the Particle Swarm Algorithm," in *IEEE International Conference on Systems, Man, and Cybernetics*, Vol. 5, IEEE, pp. 4104–4108.

Kim, J., Han, J., Kim, Y., Choi, S., and Kim, E. (2012), "Preference-Based Solution Selection Algorithm for Evolutionary Multiobjective Optimization," *Evolutionary Computation, IEEE Transactions on*, 16(1), 20–34.

Koza, J., *On the Programming of Computers by Means of Natural Selection*, Vol. 1, MIT press (1996).

Krammer, P. (2000), "CD95's deadly mission in the immune system," *NATURE-LONDON-*, pp. 789–795.

Kronfeld, M., Planatscher, H., and Zell, A. (2010), "The EvA2 Optimization Framework," in *Learning and Intelligent Optimization Conference, Special Session on Software for Optimization (LION-SWOP)*, eds. C. Blum and R. Battiti, Jan., no. 6073 in Lecture Notes in Computer Science, LNCS, Venice, Italy: Springer Verlag, pp. 247–250.

Kursawe, F. (1991), "A Variant of Evolution Strategies for Vector Optimization," *Parallel Problem Solving from Nature*, pp. 193–197.

Lampinen, J., and Zelinka, I. (1999), "Mixed integer-discrete-continuous optimization by differential evolution," in *Proceedings of the 5th International Conference on Soft Computing*, Citeseer, pp. 71–76.

Larranaga, P., and Lozano, J., *Estimation of distribution algorithms: A new tool for evolutionary computation*, Vol. 2, Springer Netherlands (2002).

Laumanns, M., and Ocenasek, J. (2002), "Bayesian optimization algorithms for multi-objective optimization," *Parallel Problem Solving from NaturePPSN VII*, pp. 298–307.

Laumanns, M., Thiele, L., Deb, K., and Zitzler, E. (2002), "Combining Convergence and Diversity in Evolutionary Multiobjective Optimization," *Evolutionary Computation*, 10(3), 263–282.

Laumanns, M., Zitzler, E., and Thiele, L. (2000), "A Unified Model for Multi-Objective Evolutionary Algorithms with Elitism," in *IEEE Congress on Evolutionary Computation*, Vol. 1, IEEE, pp. 46–53.

Li, H., and Zhang, Q. (2009), "Multiobjective Optimization Problems with Complicated Pareto Sets, MOEA/D and NSGA-II," *IEEE Transactions on Evolutionary Computation*, 13(2), 284–302.

Liefooghe, A., Basseur, M., Jourdan, L., and Talbi, E.G. (2007), "ParadisEO-MOEO: A framework for evolutionary multi-objective optimization," in *Evolutionary multi-criterion optimization*, Springer, pp. 386–400.

Lim, D., Jin, Y., Ong, Y., and Sendhoff, B. (2010), "Generalizing surrogate-assisted evolutionary computation," *Evolutionary Computation, IEEE Transactions on*, 14(3), 329–355.

Liu, B., Wang, L., Liu, Y., and Wang, S. (2011), "A unified framework for population-based

metaheuristics," *Annals of Operations Research*, pp. 1–32.

López-Ibáñez, M., and Stützle, T. (2010), "An Analysis of Algorithmic Components for Multiobjective Ant Colony Optimization: A Case Study on the Biobjective TSP," *Artifical Evolution*, pp. 134–145.

Lukasiewycz, M., Glaß, M., Reimann, F., and Teich, J. (2011), "Opt4J: a modular framework for meta-heuristic optimization," in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, ACM, pp. 1723–1730.

Meffert, K., and Rotstan, N., "JGAP: Java genetic algorithms package," (2010).

Michalewicz, Z. (2012), "Quo Vadis, Evolutionary Computation?," *Advances in Computational Intelligence*, pp. 98–121.

Miettinen, K., *Nonlinear Multiobjective Optimization*, Vol. 12, Springer (1999).

Miettinen, K., and Mäkelä, M. (2002), "On Scalarizing Functions in Multiobjective Optimization," *OR Spectrum*, 24(2), 193–213.

Mohler, R., Bruni, C., and Gandolfi, A. (1980), "A systems approach to immunology," *Proceedings of the IEEE*, 68(8), 964–990.

Moscato, P. (1989), "On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms," Technical report.

Muhlenbein, H., and Mahnig, T. (2002), "Evolutionary Optimization and the Estimation of Search Distributions with Applications to Graph Bipartitioning," *International journal of approximate reasoning*, 31(3), 157–192.

Mühlenbein, H., and Paass, G. (1996), "From Recombination of Genes to the Estimation of Distributions I. Binary Parameters," *Parallel Problem Solving from Nature*, pp. 178–187.

Ocenasek, J., and Schwarz, J. (2002), "Estimation Distribution Algorithm for Mixed Continuous-Discrete Optimization Problems," *Intelligent technologies: theory and applications: new trends in intelligent technologies*, 76, 227.

Ong, Y.S., Nair, P., and Lum, K. (2006), "Max-Min Surrogate-Assisted Evolutionary Algorithm for Robust Design," *IEEE Transactions on Evolutionary Computation*, 10(4), 392 – 404.

Ono, I., and Kobayashi, S. (1997), "A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover," *Journal of Japanese Society for Artificial Intelligence*, 14(6), 246–253.

Pampara, G., Engelbrecht, A., and Franken, N. (2006), "Binary Differential Evolution," in *IEEE Congress on Evolutionary Computation*, IEEE, pp. 1873–1879.

Pan, Q., Tasgetiren, M., and Liang, Y. (2007), "A Discrete Differential Evolution Algorithm for the Permutation Flowshop Scheduling Problem," in *Conference on Genetic and Evolutionary Computation*, ACM, pp. 126–133.

Panait, L., Balan, G., Paus, S., Skolicki, Z., Popovici, E., Sullivan, K., Harrison, J., Bassett, J., Hubley, R., Chircop, A. et al.(2010), "ECJ: A java-based evolutionary computation research system," .

Pareto, V. (1896), "Cours D'Économie Politique," .

Parham, P., and Janeway, C., *The immune system*, Garland Science New York (2005).

Parijs, L., and Abbas, A. (1998), "Homeostasis and self-tolerance in the immune system: turning lymphocytes off," *Science*, 280(5361), 243.

Parsopoulos, K., and Vrahatis, M. (2002), "Particle swarm optimization method in multiobjective problems," in *Proceedings of the 2002 ACM symposium on Applied computing*, ACM, pp. 603–607.

Pelikan, M., Goldberg, D., and Lobo, F. (2002), "A Survey of Optimization by Building and Using Probabilistic Models," *Computational Optimization and Applications*, 21(1), 5–20.

Pelikan, M., Goldberg, D., and Tsutsui, S. (2005), "Hierarchical Bayesian Optimization Algorithm: Toward A New Generation of Evolutionary Algorithms," in *SICE Annual Conference*, Vol. 3, IEEE, pp. 2738–2743.

Pelikan, M., and Sastry, K., *Scalable Optimization via Probabilistic Modeling: From Algorithms*

to Applications, Vol. 33, Springer Verlag (2006).

Poloni, C., Giurgevich, A., Onesti, L., and Pediroda, V. (2000), "Hybridization of A Multi-Objective Genetic Algorithm, A Neural Network and A Classical Optimizer for A Complex Design Problem in Fluid Dynamics," *Computer Methods in Applied Mechanics and Engineering*, 186(24), 403 – 420.

Puchinger, J., and Raidl, G.R. (2005), "Combining Metaheuristics and Exact Algorithms in Combinatorial Optimization: A Survey and Classification," in *Lecture Notes in Computer Science*, Vol. 3562, pp. 41–53.

Purshouse, R., and Fleming, P. (2003), "Evolutionary Many-Objective Optimisation: An Exploratory Analysis," in *IEEE Congress on Evolutionary Computation*, Vol. 3, IEEE, pp. 2066–2073.

Purshouse, R., and Fleming, P. (2007), "On the Evolutionary Optimization of Many Conflicting Objectives," *IEEE Transactions on Evolutionary Computation*, 11(6), 770 –784.

Purshouse, R.C., and Fleming, P.J. (2003), "Conflict, Harmony, and Independence: Relationships in Evolutionary Multi-Criterion Optimisation," in *Conference on Evolutionary Multi-Criterion Optimization*, Berlin: Springer, pp. 16–30.

Rechenberg, I. (1965), "Cybernetic Solution Path of An Experimental Problem," .

Reyes-Sierra, M., and Coello, C. (2006), "Multi-objective particle swarm optimizers: A survey of the state-of-the-art," *International Journal of Computational Intelligence Research*, 2(3), 287–308.

Rubinstein, R. (1999), "The Cross-Entropy Method for Combinatorial and Continuous Optimization," *Methodology and Computing in Applied Probability*, 1(2), 127–190.

Rubinstein, R. (2005), "A Stochastic Minimum Cross-Entropy Method for Combinatorial Optimization and Rare-event Estimation," *Methodology and Computing in Applied Probability*, 7(1), 5–50.

Sastry, K. (2001), "Efficient Cluster Optimization Using Extended Compact Genetic Algorithm with Seeded Population," in *Conference on Genetic and Evolutionary Computation*.

Sastry, K., Goldberg, D., and Pelikan, M. (2005), "Limits of Scalability of Multiobjective Estimation of Distribution Algorithms," in *IEEE Congress on Evolutionary Computation*, Vol. 3, IEEE, pp. 2217–2224.

Saxena, D., Zhang, Q., Duro, J., and Tiwari, A. (2011), "Framework for many-objective test problems with both simple and complicated pareto-set shapes," in *Evolutionary Multi-Criterion Optimization*, Springer, pp. 197–211.

Schaffer, J. (1985), "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms," in *Conference on Genetic Algorithms*, L. Erlbaum Associates Inc., pp. 93–100.

Scharnow, J., Tinnefeld, K., and Wegener, I. (2004), "The analysis of evolutionary algorithms on sorting and shortest paths problems," *Journal of Mathematical Modelling and Algorithms*, 3(4), 349–366.

Schwefel, H. (1975), "Evolutionsstrategie und Numerische Optimierung," Technische Universität Berlin.

Schwefel, H. (1981), "Numerical Optimization of Computer Models," .

SCOPUS, (2012), "http://www.scopus.com/," .

Shah, R., and Reed, P. (2011), "Comparative Analysis of Multiobjective Evolutionary Algorithms for Random and Correlated Instances of Multiobjective D-Dimensional Knapsack Problems," *European Journal of Operational Research*, 211(3), 466–479.

Shi, Y., and Eberhart, R. (1998), "A Modified Particle Swarm Optimizer," in *IEEE International Conference on Evolutionary Computation*, may, IEEE, pp. 69 –73.

Socha, K. (2004), "ACO for Continuous and Mixed-Variable Optimization," *Ant Colony Optimization and Swarm Intelligence*, pp. 53–61.

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

Srinivas, N., and Deb, K. (1994), "Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms," *Evolutionary Computation*, 2(3), 221–248.

Storn, R. (1996), "On the Usage of Differential Evolution for Function Optimization," in *Conference of the North American Fuzzy Information Processing Society*, IEEE, pp. 519–523.

Storn, R., and Price, K. (1995), "Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces," *International Computer Science Institute-Publications TR*.

Tan, Y.Y., Jiao, Y.C., Li, H., and Wang, X.K. (2012), "MOEA/D + uniform design: A new version of MOEA/D for optimization problems with many objectives," *Computers & Operations Research*, (0).

Tapia, M., and Coello, C. (2007), "Applications of Multi-Objective Evolutionary Algorithms in Economics and Finance: A Survey," in *IEEE Congress on Evolutionary Computation*, Vol. 2007, pp. 532–539.

Teytaud, O. (2006), "How entropy-theorems can show that approximating high-dim Pareto-fronts is too hard," in *Bridging the Gap between Theory and Practice-Workshop PPSN-BTP*.

Thierens, D., and Bosman, P. (2001), "Multi-Objective Mixture-Based Iterated Density Estimation Evolutionary Algorithms," pp. 663–670.

Toroslu, I.H., and Arslanoglu, Y. (2007), "Genetic algorithm for the personnel assignment problem with multiple objectives," *Information Sciences*, 177(3), 787 – 803.

Tripathi, P., Bandyopadhyay, S., and Pal, S. (2007), "Multi-Objective Particle Swarm Optimization with Time Variant Inertia and Acceleration Coefficients," *Information Sciences*, 177(22), 5033–5049.

Ullman, J. (2009), "Advising students for success," *Communications of the ACM*, 52(3), 34–37.

Van Veldhuizen, D. (1999), "Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations," in *Evolutionary Computation*.

Voigt, H., Mühlenbein, H., and Cvetkovic, D. (1995), "Fuzzy recombination for the breeder genetic algorithm," in *Proc. Sixth Int. Conf. on Genetic Algorithms*, Citeseer.

Wagner, S., Beham, A., Kronberger, G., Kommenda, M., Pitzer, E., Kofler, M., Vonolfen, S., Winkler, S., Dorfer, V., and Affenzeller, M. (2010), "Heuristiclab 3.3: A unified approach to metaheuristic optimization," in *Actas del séptimo congreso español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB'2010)*, p. 8.

Wagner, T., Beume, N., and Naujoks, B. (2007), "Pareto-, Aggregation-, and Indicator-Based Methods in Many-Objective Optimization," in *Evolutionary Multi-Criterion Optimization*, Springer, pp. 742–756.

Wang, R., Purshouse, R., and Fleming, P. (2012), "Preference-Inspired Co-Evolutionary Algorithms for Many-objective Optimisation," *IEEE Transactions on Evolutionary Computation*, PP(99), 1.

Wanner, E., Guimarães, F., Takahashi, R., and Fleming, P. (2008), "Local search with quadratic approximations into memetic algorithms for optimization with multiple criteria," *Evolutionary computation*, 16(2), 185–224.

Wei, L., Oon, W., Zhu, W., and Lim, A. (2012), "A reference length approach for the 3D strip packing problem," *European Journal of Operational Research*, 220(1), 37–47.

Wolpert, D., and Macready, W. (1997), "No Free Lunch Theorems for Optimization," *IEEE Transactions on Evolutionary Computation*, 1(1), 67–82.

Wolpert, D., Strauss, C., and Rajnarayan, D. (2006), "Advances in distributed optimization using probability collectives," *Advances in Complex Systems*, 9(4), 383–436.

Yoo, J., and Hajela, P. (1999), "Immune network simulations in multicriterion design," *Structural and Multidisciplinary Optimization*, 18(2), 85–94.

Zhang, Q., and Li, H. (2007), "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition," *IEEE Transactions on Evolutionary Computation*, 11(6), 712–731.

Zhang, Q., Liu, W., and Li, H. (2009), "The Performance of a New Version of MOEA/D on

CEC09 Unconstrained MOP Test Instances," in *IEEE Congress on Evolutionary Computation*, IEEE, pp. 203–208.

Zhang, Q., Zhou, A., and Jin, Y. (2008), "RM-MEDA: A Regularity Model-Based Multiobjective Estimation of Distribution Algorithm," *IEEE Transactions on Evolutionary Computation*, 12(1), 41–63.

Zhou, A., Qu, B., Li, H., Zhao, S., Suganthan, P., and Zhang, Q. (2011), "Multiobjective evolutionary algorithms: A survey of the state-of-the-art," *Swarm and Evolutionary Computation*.

Zitzler, E., Deb, K., and Thiele, L. (2000), "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results," *Evolutionary Computation*, 8(2), 173–195.

Zitzler, E., and Künzli, S. (2004), "Indicator-Based Selection in Multiobjective Search," in *Parallel Problem Solving from Nature*, Springer, pp. 832–842.

Zitzler, E., Laumanns, M., Thiele, L. et al.(2001), "SPEA2: Improving the Strength Pareto Evolutionary Algorithm," in *EUROGEN*, 103, pp. 1–21.

Zitzler, E., and Thiele, L. (1999), "Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach," *IEEE Transactions on Evolutionary Computation*, 3(4), 257–271.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C., and da Fonseca, V. (2003), "Performance Assessment of Multiobjective Optimizers: An Analysis and Review," *IEEE Transactions on Evolutionary Computation*, 7(2), 117–132.

Zlochin, M., Birattari, M., Meuleau, N., and Dorigo, M. (2004), "Model-based search for combinatorial optimization: A critical survey," *Annals of Operations Research*, 131(1), 373–395.