Swarm and Evolutionary Computation 2 (2012) 1–14

Contents lists available at SciVerse ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Survey paper

Memetic algorithms and memetic computing optimization: A literature review

Ferrante Neri [a,*], Carlos Cotta [b]

[a] Department of Mathematical Information Technology, P.O. Box 35 (Agora), 40014 University of Jyväskylä, Finland

[b] Departamento de Lenguajes y Ciencias de la Computación, Escuela Técnica Superior de Ingeniería Informática, Universidad de Málaga, Campus de Teatinos, 29071 Málaga, Spain

ARTICLE INFO

Article history:
Received 10 August 2011
Received in revised form
23 October 2011
Accepted 22 November 2011
Available online 30 November 2011

Keywords:
Memetic computing
Evolutionary algorithms
Memetic algorithms
Computational intelligence optimization

ABSTRACT

Memetic computing is a subject in computer science which considers complex structures such as the combination of simple agents and memes, whose evolutionary interactions lead to intelligent complexes capable of problem-solving. The founding cornerstone of this subject has been the concept of memetic algorithms, that is a class of optimization algorithms whose structure is characterized by an evolutionary framework and a list of local search components.

This article presents a broad literature review on this subject focused on optimization problems. Several classes of optimization problems, such as discrete, continuous, constrained, multi-objective and characterized by uncertainties, are addressed by indicating the memetic "recipes" proposed in the literature. In addition, this article focuses on implementation aspects and especially the coordination of memes which is the most important and characterizing aspect of a memetic structure. Finally, some considerations about future trends in the subject are given.

1. Introduction

According to the philosophical theory of Richard Dawkins, see [1], human culture can be decomposed into simple units namely memes. Thus a meme is a "brick" of the knowledge that can be duplicated in human brains, modified, and combined with other memes in order to generate a new meme. Within a human community, some memes are simply not interesting and then will die away in a short period of time. Some other memes are somewhat strong and then, similar to an infection, will propagate within the entire community. The memes can also undergo slight modifications or combine with each other thus generating new memes which have stronger features and are more durable and prone to propagation. An example of this concept is in the gossip propagation within human communities. Some gossips are, de facto, more interesting than others and persist over time reaching all the individuals of the community. In addition, gossips can be subject to slight (or sometimes major) modifications. Sometimes these modifications make these gossips more interesting and thus more durable and capable to propagate. This example of life-time learning is also interesting in order to note a major difference between the evolution and transmission of memes and that of their biological counterpart, i.e., genes. The latter is not modified during the life-time of the individual, and is transmitted as they were inherited (of course, genetic information is mixed during sexual reproduction and can be subject to mutation as well, but this is a different process not alike to life-time learning). On the contrary, the former is much more plastic and to some extent adhere to a Lamarckian model of evolution, which also explains their comparatively faster rate of adaptation with respect to biological genes.

This charming interpretation of human culture inspired Moscato and Norman in late '80s, see [2], to define Memetic Algorithms (MAs). In their early definition, MAs were a modification of Genetic Algorithms (GAs) employing also a local search operator for addressing the Traveling Salesman Problem (TSP). While in optimization the employment of hybrid algorithms was already in use, a novel and visionary perspective to optimization algorithms in terms of memetic metaphor has been given in [3]. After their earliest definition, MAs have been looked at in a skeptical way by the computer science community. A massive diffusion, in scientific papers, of MAs occurred only ten years after their definition. One important reason is the diffusion of the No Free Lunch Theorem (NFLT); see [4]. The NFLT proves that the average performance of any pair of algorithms A and B across all possible problems is identical. Thus, if an algorithm performs well on a certain class of problems, then it necessarily pays for that with degraded performance on the set of all remaining problems, as this is the only way that all algorithms can have the same performance averaged over all functions. Strictly speaking, the proof of NFLT is made under the hypothesis that both the algorithms A and B are non-revisiting, i.e., the algorithms do not perform the fitness evaluation of the same candidate solution more often than once during the optimization run. Although this hypothesis is de facto not respected for most of the computational intelligence optimization algorithms, the concept that there is no universal optimizer had a significant impact on the scientific community.

* Corresponding author. Tel.: +358 14 260 1211; fax: +358 14 260 1021.
E-mail addresses: ferrante.neri@jyu.fi (F. Neri), ccottap@lcc.uma.es (C. Cotta).

2210-6502/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.swevo.2011.11.003

For decades, researchers in optimization attempted to design algorithms having a superior performance with respect to all the other algorithms present in literature. This approach is visible in many famous texts published in those years, e.g., [5]. After the NFLT diffusion, researchers in optimization had to dramatically change their view about the subject. More specifically, it has become important to understand the relationship between the components of the proposed algorithm A and a given optimization problem f. Thus, the problem f became the starting point for building up a suitable algorithm. The optimization algorithm needs to specifically address the features of the problem f.

Since MAs were not proposed as specific optimization algorithms, but as a broad class of algorithms inspired by the diffusion of the ideas and composed of multiple existing operators, the community started showing an increasing attention toward these algorithmic structures as a general guideline for addressing specific problems. MAs have been successfully applied, in recent years, to solve complex real-world problems and displayed a high performance in a large number of cases. For example, in [6] an ad-hoc Differential Evolution (DE) is implemented for solving the multisensor fusion problem; in [7] DE based hybrid algorithm is designed to address an aerodynamic design problem; in [8], an optimization approach is given with reference to the study of a material structure; in [9,10] a computational intelligence approach is designed for a control engineering problem while in [11,12] a medical application for Human Immunodeficiency Virus (HIV) is addressed; in [13] a DE based hybrid algorithm is implemented to design a digital filter for paper production industry; in [14] a parallel memetic approach is proposed for solving large scale problems; in [15] an aerodynamic design problem is considered for the application of the meta-Lamarckian learning; in [16] MC is applied for atomic and molecular structural problems; in [17,18] the crucial problem of balance between global and local search is analyzed in the context of multi-objective optimization; in [16] a novel class of structured population for MAs, namely Cellular MAs, is defined. Scheduling and planning problems are solved in [19–21]. In [22] a memetic approach is proposed for a neural network training in the context of a medical application. Other examples of memetic approaches are given in [23,24] for robust design and in [25,14] for a NP-hard problem.

In order to properly address the question *What is a MA?*, it is important to mention the definition of MA related to its implementation features [26]. In this case, MAs are defined in the following way.

> Memetic algorithms are population-based metaheuristics composed of an evolutionary framework and a set of local search algorithms which are activated within the generation cycle of the external framework.

The development of modern techniques which are still inspired by the cultural diffusion but do not fall within the definition of MAs suggested the concept of Memetic Computing (MC). The latter is a broad subject defined in [27], where MC is defined as "...a paradigm that uses the notion of meme(s) as units of information encoded in computational representations for the purpose of problem solving". In other words, part of the scientific community tried to extend the concept of meme for problem solving, see [10], to something broader and more innovative. The fact that ad-hoc optimization algorithms (that is, knowledge-augmented or problem-specific algorithms) can efficiently solve given problems is a well-known result from literature. On the other hand, the ultimate goal in artificial intelligence is the generation of autonomous and intelligent structures. In computational intelligence optimization, the goal is the automatic detection of the optimal optimization algorithm for each fitness landscape, or, in other terms, the on-line (i.e., during run-time) automatic design of optimization algorithms. MC can be seen then as a subject which studies complex structures composed of simple modules (memes) which interact and evolve adapting to the problem in order to solve it. This view of the subject leads to a more modern definition of MC given in [28].

> Memetic computing is a broad subject which studies complex and dynamic computing structures composed of interacting modules (memes) whose evolution dynamics is inspired by the diffusion of ideas. Memes are simple strategies whose harmonic coordination allows the solution of various problems.

In order to better highlight the difference between MAs and MC, it can be thought that MA is a class of algorithms having some specific features, i.e., population, generational structure, local search within the generation. On the other hand, MC is a subject which studies algorithmic structures composed of multiple operators. In this light, MAs should be seen as a cornerstone and founding subset of MC. The main difference between the two concepts (MC and MA) is the algorithmic philosophy behind them. While MA is an optimization algorithm, an MC approach is a linked collection of operators without any prefixed structure but with the only aim of solving the problem.

This article gathers and summarizes the main research results in the field of MAs and MC optimization. This literature review is structured in three macro-sections. Section 2 shows the structure of a classical MA. Section 3 gives a literature review of MA/MC implementations in order to address specific problem features, such as constrained problems, high computational cost and multi-objective problems. Section 4, at an abstract level, discusses the results in terms of implementation features for the coordination of multiple components. Finally, Section 5 gives the conclusion of this work.

2. General structure of memetic algorithms

In order to define the notation used in this article, let us consider a solution x, i.e., a vector of n design variables $(x_1, x_2, \ldots, x_i, \ldots, x_n)$. Each design variable x_i can take values from a domain $\mathcal{D}_i$ (e.g., an interval $[x_i^L, x_i^U]$ if variables are continuous, or a certain collection of values otherwise). The Cartesian product of these domains for each design variable is called the decision space $\mathcal{D}$. Let us consider a set of (either deterministic or stochastic) functions $f_1, f_2, \ldots, f_m$ defined in $\mathcal{D}$ and returning rational values. Under these conditions, the most general statement of an optimization problem is given by the following formulas:

$$\begin{array}{lll} \text{max / min} & f_m & m = 1, 2, \ldots, M \\ \text{subject to} & g_j(x) \leqslant 0 & j = 1, 2, \ldots, J \\ & h_k(x) = 0 & k = 1, 2, \ldots, K \\ & x_i^L \leqslant x_i \leqslant x_i^U & i = 1, 2, \ldots, n \end{array} \qquad (1)$$

where g_j and h_k are inequality and equality constraints, respectively.

If $m = 1$ the problem is single-objective, while for $m > 1$ the problem is multi-objective. The particular structure of the functions g_j and h_k in each particular problem determines its constrainedness, which is often related to the hardness of its resolution. Finally, the continuous or combinatorial nature of the problem is given by the fact that $\mathcal{D}$ is a discrete or dense set. In other words, all the problems considered in this article can be considered as specific cases of the general definition in Eqs. (1).

MAs address the problem in (1) by means of a specific algorithmic structure which can be seen as an iterated sequence of the following operations, aimed at having a population (pool) of tentative solution converge (i.e., evolve from an initial high-diversity, scattered state to a low-diversity, more homogeneous state) toward an optimal (or quasi-optimal) solution:

1. Selection of parents: Selection aims to determine the candidate solutions that will survive in the following generations and be used to create new solutions. Selection for reproduction often operates in relation with the fitness (performance) of the candidate solutions; Here, performance typically amounts to the extent to which the solution maximizes/minimizes the objective function(s) f_m (although in some cases fitness may be measured by means of a different guiding function, related to the objective function but not identical, e.g., in the SAT problem the objective function is binary – satisfied/unsatisfied – yet the most common fitness function is maximizing the number of satisfied clauses). High quality solutions have thus more chances to be chosen. For example, roulette-wheel and tournament selections can be applied. Selection can also be done according to other criteria such as diversity. In such a case, only spread out individuals are allowed to survive and reproduce. If the solutions of the population are sufficiently diversified, selection can also be carried out randomly.
2. Combination of parents for offspring generation: Combination aims to create new promising candidate solutions by blending existing solutions (parents), a solution being promising if it can potentially lead the optimization process to new search areas where better solutions may be found.
3. Local improvement of offspring: The goal of local improvement is to improve the quality of an offspring as far as possible. Candidate solutions undergo refinement which correspond the life-time learning of the individuals in the original metaphor of MAs.
4. Update of the population: This step decides whether a new solution should become a member of the population and which existing solution of the population should be replaced. Often, these decisions are made according to criteria related to both quality and diversity. Such a strategy is commonly employed in methods like scatter search and many evolutionary algorithms. For instance, a basic quality-based updating rule would replace the worst solution of the population while a diversity-based rule would substitute for a similar solution according to a distance metric. Other criteria like recency (age) can also be considered. The policies employed for managing the population are essential to maintain an appropriate diversity of the population, to prevent the search process from premature convergence (i.e., too fast convergence toward a suboptimal region of the search space), and to help the algorithm to continually discover new promising search areas.

As mentioned above, MAs blend together ideas from different search methodologies, and most prominently ideas from local search techniques and population-based search. Indeed, from a very general point of view a basic MA can be regarded as one (or several) local search procedure(s) acting on a pool *pop* of $|pop| \geqslant 2$ solutions which engage in periodical episodes of cooperation via recombination procedures. This is shown in Algorithm 1. The description of the parameters is given in Table 1.

Let us analyze this template. First of all, the Initialize procedure is responsible for producing the initial set of $|pop|$ solutions. Traditional evolutionary algorithms usually resort to simply generating $|pop|$ solutions at random (systematic procedures to ensure a good coverage of the search space are sometimes defined, although these are not often used). Opposed to this, it is typical for MAs to attempt to use high-quality solutions as starting point. This can be done either using a more sophisticated mechanism (for instance, some constructive heuristic) to inject good solutions in the initial population [29], or by using a local-search procedure to improve random solutions (see Algorithm 2).

As for the TerminationCriterion function, it typically amounts to checking a limit on the total number of iterations, reaching a maximum number of iterations without improvement, having performed a certain number of population restarts, or reaching a certain target fitness.

```
function BasicMA (in P: Problem, in par: Parameters):
Solution;
begin
    pop ← Initialize(par, P);
    repeat
        newpop_1 ← Cooperate(pop, par, P);
        newpop_2 ← Improve(newpop_1, par, P);
        pop ← Compete (pop, newpop_2);
        if Converged(pop) then
            pop ← Restart(pop, par);
        end
    until TerminationCriterion(par);
    return GetNthBest(pop, 1);
end
```

Algorithm 1: A basic memetic algorithm

```
function Initialize(in par: Parameters, in P: Problem):
Bag{Solution};
begin
    pop ← ∅;
    for j ← 1 to par.popsize do
        i ← RandomSolution(P);
        i ← LocalSearch (i, par, P);
        pop ← pop ∪ {i};
    end
    return pop;
end
```

Algorithm 2: Injecting high-quality solutions in the initial population.

```
function Cooperate (in pop: Bag{Solution}, in par:
Parameters, in P: Problem): Bag{Solution};
begin
    lastpop ← pop;
    for j ← 1 to par.numop do
        newpop ← ∅;
        for k ← 1 to par.numapps^j do
            parents ← Select (lastpop, par.arityin^j);
            newpop ← newpop ∪ ApplyOperator (par.op^j,
            parents, P);
        end
        lastpop ← newpop;
    end
    return newpop;
end
```

Algorithm 3: The pipelined Cooperate procedure.

The procedures Cooperate and Improve constitute the core of the MA. Starting with the former, its most typical realization arises from the use of two operators for selecting solutions from the population and recombining them. Of course, this procedure can be easily extended to use a larger collection of variation operators applied in a pipeline fashion [30]. As shown in Algorithm 3, this procedure comprises *numop* stages, each one corresponding to the iterated application of a particular operator op^j that takes $arityin^j$ solutions from the previous stage, generating $arityout^j$ new solutions.

As to the Improve procedure, it embodies the application of a local search procedure to solutions in the population. Notice that in an abstract sense a local search method can be modeled as a unary operator (we adhere here to a strict definition of local search as a procedure for iteratively exploring the

Table 1
Parameters used in the algorithmic description of MAs.

Parameter	Interpretation
Popsize	Size of the population (number of solutions in *pop*)
Numop	Number of operators used
Numapps	Array of size 1..numop indicating the number of times each operator is applied in the main loop
Arityin	Array of size 1..numop indicating how many input solutions are required by each operator
Arityout	Array of size 1..numop indicating how many output solutions are produced by each operator
Op	Array of size 1..numop comprising the actual operators
Preserved	Number of solutions in the current population that are preserved when a restart is made

surroundings/neighborhood of a certain solution at any given time step), and hence it could have been included within the Cooperate procedure above. However, local search plays such an important role in MAs that it deserves separate treatment. Indeed, there are several important design decisions involved in the application of local search to solutions, i.e., to which solutions should it be applied, how often, for how long, etc. See also next section.

Next, the Compete procedure is used to reconstruct the current population using the old population *pop* and the population of offspring $newpop_2$. Using the terminology commonly used by the evolution strategy [31,32] community, there exist two main possibilities for this purpose: the *plus* strategy and the *comma* strategy. The non-elitist nature of the latter makes it less prone to stagnation [33], being the ratio $|newpop|/|pop| \simeq 6$ a customary choice [34]. The generation of a large number of offspring can be somewhat computationally expensive if the fitness function is complex and time-consuming though. A suitable alternative in this context is using a plus strategy with a low value of $|newpop|$, an elitist variant which is strongly related to the so-called *steady-state* replacement strategy in GAs [35]. While this option usually provides a faster convergence to high-quality solutions, premature convergence to suboptimal regions of the search space can take place, and hence corrective measures may be required. This leads to the last component of the template shown in Algorithm 1, namely the restarting procedure.

```
function Restart (in pop: Bag{Solution}, in par: Parameters,
in P: Problem): Bag{Solution};
begin
    newpop ← ∅;
    for j ← 1 to par.preserved do
        i ← GetNthBest(pop, j);
        newpop ← {i};
    end
    for j ← par.preserved + 1 to par.popsize do
        i ← RandomSolution(P);
        i ← LocalSearch (i, par, P);
        newpop ← {i};
    end
    return newpop;
end
```

Algorithm 4: The restart procedure.

First of all, it must be decided whether the population has degraded or has not, using some measure of information diversity in the population (e.g., average Hamming distance or Shannon's entropy [36] in the discrete case, or some dispersion measure in the continuous case). Once the diversity indicator provides a value below a suitable threshold, the population can be regarded as degenerate and the restart procedure is called. Again, this can be implemented in a number of ways. A very typical strategy is to keep a fraction of the current population, generating new (random or heuristic) solutions to complete the population, as shown in Algorithm 4. The term *random-immigrant* strategy [37] has been coined to describe this procedure. Alternatively, a *strong* or *heavy* mutation operator can be activated in order to drive the population away from its current location in the search space; e.g., see [38–41].

On the basis of the definitions of MA and MC reported above, while an algorithmic characterization of MA can be given, any MC specific outline would be restrictive. In other words, while MA is a class of optimization algorithms having specific implementation features, MC is a subject and an implementation philosophy. On one hand, the concept of MC appears excessively vague as all the computer science implementations if not most of the natural sciences and engineering can be seen as a subset of MC. If we look at MC in a skeptical way, it may appear as an empty box or a label to put on every single human thought. On the other hand, the importance of MC is in the unifying role taken and the novel perspective that MC suggests to computer science community. MC considers algorithms as evolving structures composed by cooperative and competitive operators. This perspective suggests the automatic generation of algorithms by properly combining the operators (memes). We may think that a computational device stores a set of operators and combines (some of) them according to a certain criterion to efficiently address a problem. This will be a further step with respect to adaptive and self-adaptive systems in MAs, see Section 4, and compose the next level of computational intelligence.

3. Memetic computing specific implementations

This section gives a literature review about MA/MC implementations for various classes of optimization problems. More specifically the present section is divided into the following subsections:

- MAs in discrete optimization
- MAs in continuous optimization
- MAs in constrained optimization
- MAs in multi-objective optimization
- MAs in the presence of uncertainties.

3.1. MAs in discrete optimization

Discrete optimization is the search for the configuration with highest performance (optimal solution) among a set of finite candidate configurations. There are several ways to describe a discrete optimization problem. In its most general form, it can be defined as a collection of problem instances, each being specified by a pair (S, f) [42], where S is the finite set of candidate configurations, defining the decision space; f is the cost or objective function, given by a mapping $f: S \rightarrow \mathbb{Q}$.

Unlike continuous problems, discrete optimization can in principle be solved by enumeration, i.e., by exhaustively counting and evaluating all the candidate solutions. In addition, discrete problems cannot utilize the gradient for searching the directions as a minimum distance between two solutions is set.

Discrete problems and more specifically the Traveling Salesman problem (TSP) have been the earliest application domains for MAs; see [2]. Implementations of hybrid algorithms were in use even before the term MA was coined. In [43] an early attempt to hybridize an evolutionary framework with local search for solving the TSP has been presented. Subsequently, still with reference to the TSP, in [44] a visionary approach which theorizes the integration of extra components and especially crossover techniques within an evolutionary framework is presented. A similar approach is given in [45]. Another related technique, which can also be considered as an early memetic approach is the so called genetic edge recombination; see e.g., [46]. More recently, actual MAs (which fit in the definition above) have been implemented to address the TSP; in [47–49], the role and effect of local search within evolutionary algorithms is extensively studied. Large scale TSP is studied in [50]. Comparative studies about the performance of MAs on TSP are reported in [51–54].

Other combinatorial problems have also been tackled by MAs; for example in [55,56] the Quadratic Assignment Problem (QAP), in [57,58] the Graph Bi-partitioning Problem, in [59] the supply chain problem, and in [60] the communication spanning tree.

The solution of an optimization problem in a discrete space (as well as for continuous problems) must be achieved by efficiently balancing the exploitation and exploration. Exploitation is the action, performed by the algorithm, of intensively analyzing a portion of the decision space in order to quickly enhance upon the best current solution while exploration is the action which leads to the detection of a candidate solution located in an unexplored areas of the decision space. The dual concept of exploitation and exploration covers two fundamental and complementary aspects of any effective search procedure. This concept is at the basic of optimization and has been termed under the names intensification and diversification, respectively, introduced within the Tabu Search (TS) methodology [61].

MA implementations for discrete optimization problems essentially tend to combine searchers for exploring the entire decision space and searchers which focus on portions of the decision space. Local search in MAs for discrete optimization performs an intensive exploitation of the search space attempting to enhance the performance by slightly modifying some design variables. The problem of *how often* and *how* the local search is implemented is a fundamental task which has been addressed in the literature in various ways. For example, in [62] an analysis of the frequency and application point of the local search, in the context of continuous optimization, is carried out. This analysis has been extended in [63] for combinatorial optimization problems and introduced the concept of sniff (or local/global ratio) for balancing genetic and local search.

Another crucial point in combinatorial optimization is the choice of neighborhood while performing the local search. An heuristic procedure for performing the fitness landscape analysis and thus the neighborhood (and local search) selection is reported in [64]. The selection of the most convenient neighborhood structures within local search is investigated in [65].

3.2. MAs in continuous optimization

When a MA is designed two of the most relevant features to take into account are (1) the cost of local search; (2) the underlying search landscape. In order to come up with efficient memetic solvers, in continuous optimization, these features must be tackled differently with respect to the discrete case.

Regarding the cost of local search, in many combinatorial domains it is frequently possible to compute the fitness of a perturbed solution incrementally, e.g., let x be a solution and let $x' \in \mathcal{N}(x)$ be a neighboring solution; then the fitness $f(x')$ can be often computed as $f(x') = f(x) + \Delta f(x, x')$, where $\Delta f(x, x')$ is a term that depends on the particular perturbation done on x and is typically efficient to compute (much more efficiently that a full fitness computation). For example, in the context of the TSP and the 2-opt neighborhood, the fitness of a perturbed solution can be computed in constant time by calculating the difference between the weights of the two edges added and the two edges removed. This is much more difficult in the context of continuous optimization problems, which are often non-linear and hard to decompose as the sum of linearly-coupled terms. Hence local search usually has to resort to full fitness computations.

Concerning the underlying search landscape, it should be observed that the interplay among the different search operators used in memetic algorithms (or even in simple evolutionary algorithms) is a crucial issue for achieving good performance in any optimization domain. When tackling a combinatorial problem, this interplay is a complex topic since each operator may be based on a different search landscape. It is then essential to understand these different landscape structures and how they are navigated; this concept is also known ad the "one operator, one landscape" view and is expressed in depth in [66]. In the continuous domain the situation is somewhat simpler, in the sense that there exists a natural underlying landscape in D (typically $D = \mathbb{Q}^n$), namely that induced by distance measures such as Euclidean distance. In other words, in continuous optimization, the set of points which can be reached by the application of unary operators to a starting point may be represented by closed spheres of radius ϵ. On the contrary, the set of points reachable by recombination operators (recall for example the BLX-α operator) can be visualized by means of a hypercubes within the decision space. The intuitive imagery of local optima and basins of attraction naturally fits here, and allows the designer to exert some control on the search dynamics by carefully adjusting the intensification/diversification properties of the operators used.

These two issues mentioned above have been dealt in the literature on memetic algorithms for continuous optimization in different ways. Starting with the first one (the cost of local search), it emphasizes the need for carefully selecting when and how local search is applied (obviously this is a general issue, also relevant in combinatorial problems, but definitely crucial in continuous ones). This decision-making is very hard in general [67,68], but some strategies have been put forward in previous works. A rather simple one is to resort to partial Lamarckianism [69] by randomly applying local search with probability $p_{LS} < 1$. Obviously, the application frequency is not the only parameter that can be adjusted to tune the computational cost of local search: the intensity of local search (i.e., for how long is local improvement attempted on a particular solution) is another parameter to be tweaked. This adjustment can be done blindly (i.e., prefixing a constant value or a variation schedule across the run), or adaptively. For example, Molina et al. [70] define three different solution classes (on the basis of fitness) and associate a different set of local-search parameters for each of them. Related to this, Nguyen et al. [71] consider a stratified approach, in which the population is sorted and divided into n levels (n being the number of local search applications), and one individual per level is randomly selected. This is shown to provide better results than random selection. We refer to [72] for an in-depth empirical analysis of the time/quality tradeoffs when applying parameterized local search within memetic algorithms. This adaptive parameterization has been also exploited in so-called *local-search chains* [73], by saving the state of the local-search upon completion on a certain solution for later use if the same solution is selected again for local improvement. Let us finally note with respect to this parameterization issue that adaptive strategies can be taken one step further, entering into the realm of self-adaptation.

As to what the exploitation/exploration balance regards, it is typically the case that the population-based component is used to navigate through the search space, providing interesting starting points to intensify the search via the local improvement operator. The diversification aspect of the population-based search can be strengthened in several ways, such as for example using multiple subpopulations [74], or diversity-oriented replacement strategies. The latter are common in scatter search [75] (SS), an optimization paradigm closely related to memetic algorithms in which the population (or reference set in the SS jargon) is divided in tiers: entrance to them is gained by solution on the basis of fitness in one case, or diversity in the other case. Additionally, SS often incorporated restarting mechanisms to introduce fresh information in the population upon convergence of the latter. Diversification can be also introduced via selective mating, as it is done in CHC (Cross generational elitist selection, Heterogeneous recombination, and Cataclysmic mutation) [76]. A related strategy was proposed by Lozano et al. [77] via the use of negative assortative mating: after picking a solution for recombination, a collection of potential mates is selected and the most diverse one is used. Other strategies range from the use of clustering [78] (to detect solutions likely within the same basin of attraction upon which it may not be fruitful to apply local search), or the use of standard diversity preservation techniques in multimodal contexts such as sharing or crowding. It should be also mentioned that sometimes the intensification component of the memetic algorithm is strongly imbricated in the population-based engine, without resorting to a separate local search component. This is for example the case of the so-called *crossover hill climbing* [79], a procedure which essentially amount to using a hill climbing procedure on states composed of a collection of solutions, using crossover as move operator (i.e., introducing a newly generated solution in the collection – substituting the worst one – if the former is better than the latter). This strategy was used in the context of real-coded memetic algorithms in [77]. A different intensifying strategy was used by Cotta and Troya [80], by considering an exact procedure for finding the best combination of variable values from the parents (a so-called *optimal discrete recombination*; see also [81]). This obviously requires the objective function is amenable to the application of an efficient procedure for exploring the dynastic potential (set of possible children) of the solutions being recombined. We refer to [82] for a detailed analysis of diversification/intensification strategies in hybrid metaheuristics (in particular in memetic algorithms).

In some cases, it may be required to detect multiple local optima rather than only the global optimum. This problem is usually indicated as multimodal optimization problem. Obviously, this situation occurs only when there is a continuous landscape because in discrete optimization there is no absolute concept of local optimum. MC approaches have been used in various contexts to address this issue. Although this is not the focus of this survey, it is worthwhile mentioning a few memetic approaches which have been proposed in literature. For example, in [83] a memetic approach composed of sequential threshold operation, global and local search allows the detection of multiple optima under fitness constraints. In [84] a heuristic mapping is proposed in order to promote the multiple convergence within a unique evolutionary cycle. By means of a similar logic, in [85] a memetic swarm intelligence approach is used for multimodal optimization. For an extensive survey on multimodal optimization, see [86].

3.3. MAs in large scale optimization

Optimization problems, both discrete and continuous, when characterized by a high number of variables are known as large scale optimization problems, or briefly Large Scale Problems (LSPs).

The detection of an efficient solver for LSPs can be a very valuable achievement in applied science and engineering since in many applications a high number of design variables may be of interest for an accurate problem description. For example, in structural optimization an accurate description of complex spatial objects might require the formulation of a LSP; similarly such a situation also occurs in scheduling problems; see [87]. Another important example of a class of real-world LSPs is the inverse problem chemical kinetics studied in [88,89].

Several memetic approaches have been largely applied in order to solve LSPs. This fact is due to the fact that a single search logic might easily turn into stagnation or premature convergence. On the other hand, a proper coordination of multiple search operators can compensate the limits of the others and thus allow the overcome of a critical algorithmic situation characterized by no improvements. For example, in [90] a MA which integrates a simplex crossover within the DE framework has been proposed in order to solve LSPs; see also [91]. In [92], on the basis of the studies carried out in [93–95], a DE for LSPs has proposed. The algorithm proposed in [92] performs a probabilistic update of the control parameter of DE variation operators and a progressive size reduction of the population size. Although the theoretical justifications of the success of this algorithm are not fully clear, the proposed approach seems to be extremely promising for various problems. In [96], a memetic algorithm which hybridizes the self-adaptive DE described in [94] and a local search applied to the scale factor in order to generate candidate solutions with a high performance has been proposed. Since the local search on the scale factor (or scale factor local search) is independent on the dimensionality of the problem, the resulting memetic algorithm offered a good performance for relatively large scale problems; see [96]. By combining the latest two philosophies, Caponio et al. [97] propose a MA which integrates the potential of the scale factor local search within the self-adaptive DE with automatic reduction of the population size in order to guarantee a high performance, in terms of convergence speed and solution detection, for large scale problems. In a similar way, multiple strategies for DE control parameter update and population size reduction are combined in [98].

In [99], a DE framework with self-adaptively coordinated multiple mutation strategies, see [100], is hybridized in a memetic fashion with the multi-trajectory search proposed in [101]. The resulting algorithm appears very promising for handling LSPs.

Finally, another memetic approach, used for handling LSPs, is by means of structured populations. One example is given in [102] where multiple DE search strategies are reproduced within a ring topology by means of a simple and natural randomized adaptation throughout the islands of the structured populations. In this scheme, the scale factor of the most successful islands is inherited by the other islands after a perturbation which prevents from premature convergence. A more efficient scheme for handling LSPs is proposed in [103] where the premature convergence is achieved by means of the cooperative/competitive application of two simple mechanisms: the first, namely shuffling, consists of randomly rearranging the individuals over the sub-populations; the second consists of updating all the scale factors of the sub-populations.

3.4. MAs in constrained optimization

When MAs are applied to constrained optimization problems, the integration of algorithmic components in the memetic framework to handle the constraints becomes fundamental. In [104] a MA composed of a GA framework and a gradient based local search integrates the constraint violation criterion proposed in [105]: (i) the feasible individual is preferred over the infeasible one; (ii) for two feasible individuals, the individual with better fitness is preferred; and (iii) for two infeasible individuals, the individual with lower constraint violation is preferred. Their experimental results indicated that MA outperformed conventional algorithms in terms of both quality of solution and the rate of convergence. The same set of rules has been used to handle the constraints in [106], where, in the context of multi-objective optimization, a MA which makes use of a local search strategy based on the interior point method, has been proposed.

In [107] a MA composed by an evolutionary framework and Sequential Quadratic Programming (SQP) employs the constraint violation procedure described in [108]. In [109], an MA containing an adaptive penalty method and a line search technique is proposed. An agent based MA in which four local search algorithms were used for adaptive learning has been proposed in [110]. The algorithms included random perturbation, neighborhood and gradient search methods. Subsequently, another specialized local search method was designed to deal with equality constraints; see [111]. The constraints were handled again using the rules proposed in [105].

In [112] a memetic co-evolutionary differential evolution algorithm where the population was divided into two sub-populations has been proposed. The purpose of one sub-population is to minimize the fitness function, and the other is to minimize the constraint violation. The optimization was achieved through interactions between the two sub-populations. No penalty coefficient has been used in the method while a Gaussian random number was used to modify the individuals when the best solution remained unchanged over several generations.

Some domain-specific applications are solved by means of MAs for constraint optimization; see [113–116]. Boudia and Prins [114] considered the problem of cost minimization of a production–distribution system. A repair mechanism was applied for constraint satisfaction. Park et al. [116] combined a GA framework with a tunnel-based dynamic programming scheme to solve highly constrained non-linear discrete dynamic optimization problems arising from long-term planning. The infeasible solutions were repaired by randomly sampling part of the solutions and replacing some of the previous variables (regenerate partial characters). The algorithm successfully solved reasonable sized practical problems which cannot be solved by means of conventional approaches. A multistage capacitated lot-sizing problem was solved by the memetic algorithm proposed in [113] using heuristics as local search and standard recombination operators. Gallardo et al. [115] propose a multilevel MA for solving weighted constrained satisfaction problems, based on the integration of exact techniques within the MA for recombination purposes, and the use of upper coordination level involving the MA and an incomplete branch and bound derivate (beam search)—see also [117].

Some other studies, instead of dealing with conventional candidate solutions, require the encoding of mixed continuous/integer variables or the inclusion of Boolean variables; see [118]. Within this class of problems, mixed representations of the constrained Vehicle Routing Problems (VRPs) have been extensively studied in literature and several MA implementations have been proposed; see [119,120]. Multi-compartment vehicle routing problems and cumulative vehicle routing problems are studied in [121,122], respectively. Other examples of related work are given in [123–126].

3.5. MAs in multi-objective optimization

In order to tackle multi-objective optimization problems, a well designed algorithm should capable to detect a set of points representative of the Pareto front being well sparse over it. Multi-Objective MAs (MOMAs) attempt to obtain this result properly hybridizing evolutionary operators and local search. In order to pursue this aim, the selection mechanism, i.e., that mechanism that chooses which solutions should be retained and which discarded, must be well designed. A first important feature of the selection mechanism is that within a set of solutions, those that dominate the others should be chosen. However, dominance relation alone leaves many pairs of solutions incomparable. For this reason, the employment of only the dominance relation may not be able to define a single best solution in a neighborhood or in a tournament.

There are mainly two big families of multi-objective solvers (regardless of their memetic nature) and can be classified in the following way: (1) algorithms that do not combine the objective functions and perform the selection by means of a dominance based criterion; (2) algorithms that make use of combinations of objectives for selecting new individuals.

The first category is based on the dominance sorting defined in [5] and consists of a dominance-based ranking of all the solutions of a population. This mechanism has been employed by popular evolutionary algorithms for multi-objective optimization; see [127–129].

In MOMAs the selection criterion involves not only the evolutionary framework but also the local search components. In [130,131] a greedy local search method based on dominance relation is proposed. This mechanism simply allows the acceptance of a newly generated neighbor solution if it dominates the current solution. In population-based Pareto local search, see [132–134], the neighborhood of each solution of the current population is explored, and if no solution of the population weakly dominates a generated neighbor, the neighbor is added to the population. Lust and Jaszkiewicz [135] propose a method to speed-up local search algorithms based on dominance sorting. In [136] a dominance criterion is integrated into the evolutionary framework and multiple local search components such as Simulated Annealing and Rosenbrock Algorithm. In addition, Caponio and Neri [136] propose the cross dominance adaptation as a criterion to coordinate global and local search on the basis of the principles explained in [137]. These approaches have the advantage of not requiring extra parameters for performing their implementation. On the other hand, this criterion does not allow a control on the solution spread in proximity of the Pareto front. This drawback imposes the employment of extra components which guarantee the population spread (in terms of fitness values); see e.g., [138,130]. In addition, while dominance allows a good ranking when few objectives are involved, it is often unreliable when the problem handles many simultaneous objectives. In the latter case, it is likely to have sets solutions which do not dominate each other and thus the algorithm cannot perform an efficient selection.

The second category is based on the idea that if a ranking amongst the objectives can be performed then the multiple objectives can be combined to generate a single-objective optimization problem. The ranking is performed by associating to each objective a weight value. The functions combining the objectives are usually indicated as aggregation functions. When this approach is employed the algorithm obviously does not detect a Pareto front but only one solution. However, this drawback can be overcome by the use of multiple aggregation functions defined by various weight vectors. A scheduled variation of weight parameters is employed in [139,140]. A deterministic updated of the weight parameters to generate a repulsion among solution and thus dispersion in proximity of the Pareto front is proposed in [141,142]. A meta-evolution of the weights is presented in [143]. A randomized weight update, similar to a random walk local search, is proposed in [144] while a fully random update is presented in [145,146]. The employment of multiple set of weight parameters allows a natural dispersion of the solutions and thus, unlike dominance based sorting methods, no additional components are required. In addition, several speed-up techniques may easily be used in local search based on aggregation functions. On the other hand, this category of methods has the drawback that the selection of a proper set of weights must be performed. In order to overcome this problem, some research is focused on the automatic selection of the weights; see [147].

3.6. MAs in the presence of uncertainties

Uncertainties in optimization problems are very common in real-world applications due to the presence of measurement devices and approximation models. A fitness function contains uncertainties if the variable "time" takes place in the fitness evaluation of a solution. In other words, if for a given candidate solution x, the fitness calculation $f(x)$ can return different values in different moments, then the fitness function f is said to be affected by uncertainties. In the survey proposed in [148] the sources of uncertainties are categorized as (1) uncertainties due to approximation (2) uncertainties due to robustness (3) uncertainties due to noise (4) uncertainties due to time-variance. In this section the same categorization will be employed.

In some applications, the actual fitness function can be unavailable throughout the entire optimization process or, due to its excessive computational cost, can be replaced by an approximation model. When the fitness value is computed by an approximation model a slightly different value than the actual fitness is expected. In addition, an approximation procedure can be adjusted over the optimization time and alternated with the actual fitness thus resulting in multiple fitness values for a single candidate solution. In this sense, the employment of approximation models introduces an uncertainty in the landscape. In order to face this difficulty, in [149,150] the Inexact Pre-Evaluation (IPE) framework is proposed. IPE uses the expensive function in the first few generations and then uses the model almost exclusively while only a portion of the elites are evaluated with the expensive function and are used to update the model. This mechanism has been integrated into a hierarchical distributed algorithm [151]. This idea has been expanded such that each layer may use different solvers, within a memetic framework employing a gradient based search [152]. In [153] the Controlled Evaluations (CE) framework has been proposed. This framework monitors the model accuracy using cross-validation: a memory structure containing the previously evaluated vectors is split into two sets which are then used to train the approximation model. In [154], in the context of expensive multi-objective optimization, a memetic approach integrated fuzzy logic for alternating real and approximated fitness evaluation has been proposed. Another widely used option is a memetic approach employing the Trust Region (TR), i.e., a portion of the decision space where the approximation model can be reliably used; see [155–157]. In [158,159], memetic frameworks combining an EA as a global search, where at each generation every non-duplicated vector in the population is refined using a TR, has been proposed. In [160,161] the authors proposed a TR memetic framework which uses quadratic models and clustering. Zhou et al. [162] proposed a memetic framework which occasionally uses an inaccurate model capable to detect proposing solutions; see [163]. Lim at el. [164] have recently proposed a framework composed of an ensemble of approximation models as well smoothing models. Other approaches, namely model-adaptive frameworks, have been proposed [165,161,166]. Similar to the approach in [163], model-adaptive frameworks employ a set of candidate models which are automatically selected by a supervising system.

Robust parameters of a system are those parameters which lie in a region of the parameter hyperspace characterized by similar system responses. In other words, if a robust parameter is slightly perturbed, the system response only slightly varies. Robust optimization is a field of optimization theory which aims to detecting robust parameters. Reversely, if a parameter is not robust, small parameter variations can result into large variation of the system response. Very close solution, ideally identically can give very different system response and thus in robust optimization, identical solutions can be characterized by very different fitness values. In order to address these problems, in [167] an algorithm for robust optimization of digital filters where the uncertainty in performance is due to material imperfections has been proposed. In [168] the problem of optimizing a robust aircraft control system using a memetic algorithms is studied. Still in the context of aircraft design, a surrogated based approach, i.e., an approximation model, for computationally expensive optimization problems is proposed in [169]. The robust control design of a control system for an electric motor is proposed in [170] by applying a surrogate assisted model. Other examples of memetic robust design, regarding multi-objective optimization, are given in [171–173]. Lim et al. [174] addressed the problem of robust optimization when no a-priori information about the distribution of uncertainties is known. The problem of robust design in constrained multi-objective optimization is analyzed by means of a MA in [175]. In the latter work, micro-populations act as local search within the decision space. In [176] a robust airline scheduling problem where the goal was to obtain a fleet assignment which accounts for flight re-timing and aircraft rerouting has been proposed. Another robust scheduling problem, i.e., the stochastic capacitated vehicle routing problem, is addressed in [177].

The noise in optimization is a typical condition which plagues real-world applications and occurs every time measurements concur to the fitness value computation. These measurements can be physical instruments, like shown in [9], or computational devices which contain uncertainties, such as a neural network; see e.g., [178]. Some examples of memetic frameworks addressing noisy landscapes are given in the following. Kim and Abraham [179] combine a bacteria foraging algorithm with a real-coded evolutionary algorithm for addressing a control engineering design problem. The noise is handled by re-sampling and filtering. In order to tackle noisy problems by means of memetic frameworks, several algorithmic solutions have been proposed in the literature. In [180] MA based on differential evolution where the scale factor was adjusted with a line search is proposed and combined with an adaptive resampling technique. In [181] the authors considered the noisy pattern recognition problem of inexact graph matching, that is, determining whether two images match when one is corrupted by noise. Ozcan and Mohan [182] studied the problem of matching an input image to one from an available data set. The difficulty being that the input image may be partially obscured, deformed and so on which results in a noisy optimization problem. In [183], a resampling technique is integrated within a MA which uses a self-organizing map (SOM) as a local search. The algorithm was designed to solve the VRP with emphasis on noisy data. In [184,185] the authors tackled the problem of training a neural network used for controlling resource discovery in peer-to-peer (P2P) networks. In order to face this kind of problem, a diversity based adaptation is proposed. A similar approach was used in the hierarchical optimization problem proposed in [186].

Time-variance occurs when the fitness values of (at least some of) the points depend on time. This situation can be visualized as a landscape which is not stationary but moves over time, twisting and changing shape. This fact obviously implies that the position of the optima varies with time and thus, when the optima are detected, the algorithm should be able to follow the basins of attraction to find and locate them anew. It should be remarked that while the three previous categories the uncertainties are due to an erroneous estimation of the fitness value in a point, in time-variant problems the actually fitness value of a solution varies over time. In order to tackle this class of problems, in [187–189] a MA combining a binary evolutionary framework with the variable local search (VLS) operator to track optima in dynamic (time-variance) problems has been proposed. In [190] a MA based on Particle Swarm Optimization (PSO) for dynamic optimization problems has been proposed. This modified PSO employs multiple techniques for handling the time-dependence. Moser and Hendtlass [191] combined the Extremal Optimization algorithm (EO) [192] and a deterministic local search. Due to its structure, EO naturally adapts to changing environments and thus is a promising background for this class of problems. Another variant, employing the Hooke–Jeeves Algorithm, has been proposed in [193]. A comparative study on this sub-field is reported in [194]. In [195], a MC approach based on the scatter search framework for dynamic and highly constrained problems. In [196], in the context of dynamic multiobjective problems, a multistart system is achieved by accelerating the convergence of the algorithm. This aim is pursued by means of a modified gradient capable to predict the changes in the Pareto set. Wang et al. [197] proposed a MA for dynamic optimization which used a binary representation where at each generation the elite was refined by a local search algorithm and added and updated while the fitness landscape changes.

4. Coordination of the algorithmic components

When a MA or, more generally, a MC approach is designed, it is immediately clear that the final result is an algorithm composed of several parts. These parts can be called memes by following the metaphor, operators if a low level design is performed, or evolutionary framework and local search algorithms if a classical MA is considered. Regardless of the specific algorithmic implementation, a crucially important problem, if not the most important problem in MC is to determine how the memes interact during the optimization process. In order to clarify the tendencies in literature Ong et al. proposed a classification of adaptive MAs in [198]. By revisiting and updating this classification, the coordination of the memes has been performed in one of the following ways:

1. Adaptive Hyper-heuristic, see e.g., [199–201,89], where the coordination of the memes is performed by means of heuristic rules
2. Meta-Lamarckian learning, see e.g., [202,203,16,15], where the success of the memes biases their activation probability, thus performing an on-line algorithmic design which can flexibly adapt to various optimization problems
3. Self-Adaptive and Co-Evolutionary, see e.g., [204–206], where the memes, either directly encoded within the candidate solutions or evolving in parallel to them, take part in the evolution and undergo recombination and selection in order to select the most promising operators
4. Fitness Diversity-Adaptive, see e.g., [9,207,208,12,11,14,13], where a measure of the diversity is used to select and activate the most appropriate memes.

The first category includes those algorithms in which the memes are coordinated by means of a prefixed scheme or schedule. These schemes can be randomized or deterministic. In a randomized scheme the memes can be randomly activated one by one or in a sequence by applying a success rule; see [209,210]. Regarding deterministic schemes, a typical implementation is a schedule which subdivides a given budget to each meme, e.g., in [89]. A slightly more complex hyper-heuristic approach is the choice function; see [201]. This approach rewards the most promising meme(s) by reiterating its (their) application. When the memes stop being successful, the application of other memes is tried. A further example of choice function approach is given in [199] where a tabu list is employed to classify the success of the memes and their activation schedule.

Meta-Lamarckian learning is an extension and an evolution of the hyper-heuristic MAs and especially the choice functions and constitutes a fairly general and flexible framework for algorithmic design; see [15]. More specifically, a basic meta-Lamarckian learning strategy was proposed as the baseline algorithm for comparison. This basic strategy is a simple random coordination of memes without any adaptation. Then, the decision space is decomposed into sub-areas for the separate optimization of each sub-area. This approach assumes that different optimizers are suitable for different problems and thus each sub-area requires a different meme. In order to choose a suitable meme at each decision point, the strategy gathers knowledge about the ability of the memes to search on a particular region of the search space from a database of past experiences archived during the initial search. The memes identified then form the candidate memes that will compete, based on their rewards, to decide on which meme will proceed with the local improvement. In this way, memes with different specializations are coordinated and harmonically work jointly to solve the whole optimization problem. Two selection strategies, both based on the roulette wheel selection, have been tested for the meme selection; see [15,198]. It is worthwhile mentioning that besides Lamarckian and Meta-Lamarckian systems characterized by a change of the solution after the application of the non-evolutionary memes (i.e., local search components), Baldwinian systems also exist. In the latter approaches, the solutions are not modified after the employment of local search, see [211,212], and the local search biases somehow the evolutionary search. For example, while the solution is not modified the fitness can take into account (by means of a penalization factor) the potential of the genotype when the local search is applied. Although Lamarckian systems are more commonly used, in some cases the application of a Baldwinian approach appeared to be preferable. A memetic approach which is closely related to the Meta-Lamarckian learning is the so called algorithmic concept of ensemble. This algorithmic structure considers a pool of operators, search logics, constraint handling techniques etc., see e.g., [206], and selects the most suitable one on the basis of a trial and error scheme. For example, in [213], a DE framework with an ensemble of mutation strategies and parameter setting is proposed. In [214] the ensemble logic has been applied in the context of evolutionary programming.

The third category relies on the evolutionary principles for the meme development and selection. In self-adaptive MAs, each solution is composed of its genetic and memetic material. Thus, the memes are directly encoded into the solutions and their action is associated to the hosting solution. For example, the local search algorithms encoded into memetic material attempt to improve the genotype of the hosting solution or, more generally, specify the meme that will be used to perform local search in the neighborhood of the solution; see [65,215]. By mean of the application of a self-adaptive logic multiple memes evolve during the optimization process. For this reason, the term Multimeme Algorithm has been used in this context; see [216,65,215,217,218]. Co-Evolutionary MAs are conceptually similar to self-adaptive MAs but are implemented in a different way. The memetic material, composed of multiple memes, evolve in a population separated from the population of solutions. Populations of genes and memes evolve separately and simultaneously and their solutions are linked; see [219–221,205]. In another related algorithmic scheme, namely self-generating MAs, a grammar is used to specify the employment and coordination of local search [222,223].

The Fitness Diversity-Adaptive MAs automatically perform the meme coordination by analyzing the population status. In these adaptive systems, fitness diversity is used in order to estimate the population diversity; see [217,14]. This choice is done considering that for multi-variate problems the measure of genotypical distance can be excessively time- and memory-consuming and thus the adaptation might require an unacceptable computational overhead. Obviously, fitness diversity could not give an efficient estimation of population diversity, since it can happen that very different points take the same fitness values, e.g., if the points lay in a plateau. However, this fact does not affect the decision mechanism of the adaptive system for the following reasons: when the diversity is low one or more explorative local searchers, e.g., Nelder–Mead Simplex [224], are activated in order to offer an alternative search logic, and possibly to detect new promising search directions and increase the diversity. If this mechanism fails and the algorithm keeps losing diversity and converging to some areas of the decision space an exploitative local search algorithm, e.g., Rosenbrock Algorithm [225], attempts to quickly perform the exploitation of the most promising basin of attraction and thus quickly complete the search. If the fitness diversity is low, the candidate solutions in the population have a similar performance. This fact can mean either that the solutions are concentrated within a small region of the decision space, or that the solutions are distributed over one or more plateaus or over two or more basins of attraction having a similar performance. It can easily be visualized that all the listed situations are undesirable and that the activation of an alternative search move can increase the chances to detect "fresh" genotypes. In other words, although the Fitness Diversity Adaptation (FDA) does not guarantee a proper estimation of the population diversity, it is an efficient index to estimate the correct moment of the evolution which would benefit from a local search application. This adaptation logic has been proposed in [9] for the first time and subsequently used in [11]. Another diversity metric, more sensitive and capable to handle flat landscapes has been proposed in [12]. Some other metrics for DE frameworks have been proposed in [207,13]. Another diversity metric with reference to a chemical engineering problem has been proposed in [88]. A comparative analysis of the diversity metrics has been reported in [208] where the conclusion has been that a proper choice in terms of diversity metric should be carried out on the basis, not only of the problem features but also the framework features. For example, an efficient diversity metric for Evolution Strategy (ES) would likely be inadequate to measure the diversity of DE. This consideration can be seen as a consequence of the NFLT.

5. Conclusion and future trends in memetic computing

A further step of MC, or more generally of Computational Intelligence, consists of generating more "intelligent" systems capable to recognize the problem features and select/design a suitable solver. In our case this aim means that an algorithmic meta-structure performs an on-line design of the optimization algorithm. This goal is ambitious and, although a topic of discussion, has not been addressed yet. An early attempt of modeling a system capable to automatically generate MC approaches for given problems is described in [226] but no actual implementation is given.

However, the importance of MC is in leaving a general definition which allows to counterbalance a tendency reported in literature during the latest years. New optimization algorithms are often proposed in literature and these algorithms are usually inspired by the most diverse physical and biological phenomena and are presented as new paradigms and/or as modification of other computational paradigms. This approach led to a focus loss about a fundamental fact: all the optimization algorithms are the alternating combination of operators belonging to two groups, i.e. generation mechanisms of trial solution(s) and selection criteria for choosing which solutions should be retained for the following step. In this sense, the division of the field into disjoint sub-fields, such as Genetic Algorithms, Evolution Strategies, Differential Evolution, Particle Swarm Optimization, etc., should be overcome in favor of a more flexible view which sees all the optimization algorithms simply as a combination of operators concurring to the detection of the optimum.

In other words, the adoption of a MC view of optimization is the basics for the automatic generation of algorithms and for moving a further step toward the intelligence of machines. A remaining problem that will need to be addressed is how the candidate algorithms can be quickly and efficiently tested and subsequently how the algorithmic design should be performed. This will likely be one of the main technological questions for the next decade.

Acknowledgments

F. Neri is supported by the Academy of Finland, Akatemiatutkija 130600, Algorithmic Design Issues in Memetic Computing. C. Cotta is supported by Spanish MICINN under projects NEMESIS (TIN2008-05941) and ANYSELF (TIN2011-28627-C04-01), and by Junta de Andalucía under project TIC-6083.

References

[1] R. Dawkins, The Selfish Gene, Clarendon Press, Oxford, 1976.
[2] P. Moscato, M. Norman, A competitive and cooperative approach to complex combinatorial search, Technical Reports 790 (1989), Caltech Concurrent Computation Program.
[3] P. Moscato, On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms, Technical Reports 826 (1989), Caltech Concurrent Computation Program.
[4] D. Wolpert, W. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 67–82.
[5] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Co., Reading, MA, USA, 1989.
[6] R. Joshi, A.C. Sanderson, Minimal representation multisensor fusion using differential evolution, IEEE Transactions on Systems, Man and Cybernetics, Part A 29 (1) (1999) 63–76.
[7] T. Rogalsky, R.W. Derksen, Hybridization of differential evolution for aerodynamic design, in: Proceedings of the 8th Annual Conference of the Computational Fluid Dynamics Society of Canada, 2000, pp. 729–736.
[8] X.F. Fan, Z. Zhu, Y.S. Ong, Y.M. Lu, Z.X. Shen, J.-L. Kuo, A direct first principle study on the structure and electronic properties of bexzn1-xo, Applied Physics Letter 91 (121121) (2007).
[9] A. Caponio, G.L. Cascella, F. Neri, N. Salvatore, M. Sumner, A fast adaptive memetic algorithm for on-line and off-line control design of pmsm drives, IEEE Transactions on System Man and Cybernetics-Part B, Special Issue on Memetic Algorithms 37 (1) (2007) 28–41.
[10] F. Neri, E. Mininno, Memetic compact differential evolution for cartesian robot control, IEEE Computational Intelligence Magazine 5 (2) (2010) 54–65.
[11] F. Neri, J. Toivanen, R.A.E. Mäkinen, An adaptive evolutionary algorithm with intelligent mutation local searchers for designing multidrug therapies for HIV, Applied Intelligence 27 (3) (2007) 219–235.
[12] F. Neri, J. Toivanen, G.L. Cascella, Y.S. Ong, An adaptive multimeme algorithm for designing HIV multidrug therapies, IEEE/ACM Transactions on Computational Biology and Bioinformatics 4 (2) (2007) 264–278.
[13] V. Tirronen, F. Neri, T. Kärkkäinen, K. Majava, T. Rossi, An enhanced memetic differential evolution in filter design for defect detection in paper production, Evolutionary Computation 16 (2008) 529–555.
[14] J. Tang, M.H. Lim, Y.S. Ong, Diversity-adaptive parallel memetic algorithm for solving large scale combinatorial optimization problems, Soft Computing—A Fusion of Foundations, Methodologies and Applications 11 (9) (2007) 873–888.
[15] Y.S. Ong, A.J. Keane, Meta-lamarkian learning in memetic algorithms, IEEE Transactions on Evolutionary Computation 8 (2) (2004) 99–110.
[16] Q.C. Nguyen, Y.S. Ong, M.H. Lim, A probabilistic memetic framework, IEEE Transactions on Evolutionary Computation 13 (3) (2009) 604–623.
[17] H. Ishibuchi, T. Yoshida, T. Murata, Balance between genetic search and local search in memetic algorithms for multiobjective permutation flow shop scheduling, IEEE Transactions on Evolutionary Computation 7 (2003) 204–223.
[18] K. Tan, S. Chiama, A. Mamuna, C. Goha, Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization, European Journal of Operational Research 197 (2009) 701–713.
[19] S.M.K. Hasan, R. Sarker, D. Essam, D. Cornforth, Memetic algorithms for solving job-shop scheduling problems, Memetic Computing Journal 1 (1) (2009) 69–83.
[20] K.K. Lim, Y.-S. Ong, M.H. Lim, X. Chen, A. Agarwal, Hybrid ant colony algorithms for path planning in sparse graphs, Soft Computing—A Fusion of Foundations, Methodologies and Applications 12 (10) (2008) 981–994.
[21] K.C. Tan, C.Y. Cheong, C.K. Goh, Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation, European Journal of Operational Research 177 (2) (2007) 813–839.
[22] H.A. Abbass, An evolutionary artificial neural networks approach for breast cancer diagnosis, Artificial Intelligence in Medicine 25 (3) (2002) 265–281.

[23] Y.-S. Ong, P.B. Nair, K.Y. Lum, Max–min surrogate-assisted evolutionary algorithm for robust design, IEEE Transactions on Evolutionary Computation 10 (4) (2006) 392–404.
[24] D. Lim, Y.-S. Ong, Y. Jin, B. Sendhoff, B.S. Lee, Inverse multi-objective robust evolutionary design, Genetic Programming and Evolvable Machines 7 (4) (2006) 383–404.
[25] J. Tang, M.H. Lim, Y.S. Ong, Parallel memetic algorithm with selective local search for large scale quadratic assignment problems, International Journal of Innovative Computing, Information and Control 2 (6) (2006) 1399–1416.
[26] W.E. Hart, N. Krasnogor, J.E. Smith, Memetic evolutionary algorithms, in: W.E. Hart, N. Krasnogor, J.E. Smith (Eds.), Recent Advances in Memetic Algorithms, Springer, Berlin, Germany, 2004, pp. 3–27.
[27] Y.-S. Ong, M.-H. Lim, X. Chen, Memetic computation-past, present and future, IEEE Computational Intelligence Magazine 5 (2) (2010) 24–31.
[28] F. Neri, C. Cotta, P. Moscato, Handbook of Memetic Algorithms, in: Studies in Computational Intelligence, vol. 379, Springer-Verlag, 2012.
[29] P. Surry, N. Radcliffe, Inoculation to initialise evolutionary search, in: T. Fogarty (Ed.), Evolutionary Computing: AISB Workshop, in: Lecture Notes in Computer Science, No. 1143, Springer-Verlag, 1996, pp. 269–285.
[30] P. Moscato, C. Cotta, A gentle introduction to memetic algorithms, in: F. Glover, A. Kochenberger (Eds.), Handbook of Metaheuristics, Kluwer Academic Publishers, 2003, pp. 105–144.
[31] I. Rechenberg, Evolutionsstrategie: Optimierung Technischer Systeme Nach Prinzipien Der Biologischen Evolution, Frommann-Holzboog Verlag, Stuttgart, 1973.
[32] H.-P. Schwefel, Evolution strategies: A family of non-linear optimization techniques based on imitating some principles of natural evolution, Annals of Operations Research 1 (1984) 165–167.
[33] T. Bäck, Evolutionary Algorithms in Theory and Practice, Oxford University Press, New York NY, 1996.
[34] T. Bäck, F. Hoffmeister, Adaptive search by evolutionary algorithms, in: W. Ebeling, M. Peschel, W. Weidlich (Eds.), Models of Self-Organization in Complex Systems, in: Mathematical Research, No. 64, Akademie-Verlag, 1991, pp. 17–21.
[35] D. Whitley, Using reproductive evaluation to improve genetic search and heuristic discovery, in: J. Grefenstette (Ed.), 2nd International Conference on Genetic Algorithms and their Applications, Lawrence Erlbaum Associates, Cambridge, MA, 1987, pp. 108–115.
[36] Y. Davidor, O. Ben-Kiki, The interplay among the genetic algorithm operators: Information theory tools used in a holistic way, in: [231], 1992, pp. 75–84.
[37] H. Cobb, J. Grefenstette, Genetic algorithms for tracking changing environments, in: S. Forrest (Ed.), ICGA 1993, Morgan Kaufmann, San Mateo CA, 1993, pp. 529–530.
[38] E.K. Burke, J. Newall, R. Weare, A memetic algorithm for university exam timetabling, in: E.K. Burke, P. Ross (Eds.), Practice and Theory of Automated Timetabling, in: Lecture Notes in Computer Science, vol. 1153, Springer-Verlag, Berlin Heidelberg, 1996, pp. 241–250.
[39] E.K. Burke, A.J. Smith, A memetic algorithm to schedule planned grid maintenance, in: M. Mohammadian (Ed.), Computational Intelligence for Modelling, Control and Automation, IOS Press, 1999, pp. 12–127.
[40] P.M. França, J.N.D. Gupta, A.S. Mendes, P. Moscato, K.J. Veltnik, Evolutionary algorithms for scheduling a flowshop manufacturing cell with sequence dependent family setups, Computers and Industrial Engineering 48 (2005) 491–506.
[41] P.M. França, A.S. Mendes, P. Moscato, A memetic algorithm for the total tardiness single machine scheduling problem, European Journal of Operational Research 132 (2001) 224–242.
[42] C.H. Papadimitriou, K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, 1982.
[43] R.M. Brady, Optimization strategies gleaned from biological evolution, Nature 317 (1985) 804–806.
[44] J.J. Grefenstette, Incorporating problem specific knowledge into genetic algorithms, in: L. Davis (Ed.), Genetic Algorithms and Simulated Annealing, Research Notes in Artificial Intelligence, Morgan Kaufmann Publishers, 1987, pp. 42–60.
[45] P. Jog, J. Suh, D.V. Gucht, The effects of population size, heuristic crossover and local improvement on a genetic algorithm for the travelling salesman problem, in: 3rd International Conference on Genetic Algorithms, Morgan Kaufman, 1989, pp. 110–115.
[46] K. Mathias, D. Whitley, Genetic operators, the fitness landscape and the traveling salesman problem, in: [231], 1992, pp. 219–228.
[47] B. Freisleben, P. Merz, A genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems, in: 1996 IEEE International Conference on Evolutionary Computation, IEEE Press, Nagoya, Japan, 1996, pp. 616–621.
[48] B. Freisleben, P. Merz, New genetic local search operators for the traveling salesman problem, in: [234], 1996, pp. 890–900.
[49] P. Merz, B. Freisleben, Genetic local search for the TSP: New results, in: 1997 IEEE International Conference on Evolutionary Computation, IEEE Press, 1997, pp. 159–164.
[50] P. Merz, T. Fischer, A memetic algorithm for large traveling salesman problem instances, in: 7th Metaheuristics International Conference, MIC'2007, 2007.
[51] P. Merz, On the performance of memetic algorithms in combinatorial optimization, in: Second Workshop on Memetic Algorithms (WOMA II), Genetic and Evolutionary Computation Conference, GECCO 2001, Morgan Kaufmann, 2001, pp. 297–345.
[52] P. Merz, A comparison of memetic recombination operators for the traveling salesman problem, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2002, Morgan Kaufmann, 2002, pp. 472–479.
[53] P. Merz, Advanced fitness landscape analysis and the performance of memetic algorithms, Evolutionary Computation 12 (3) (2004) 303–326.
[54] P. Merz, B. Freisleben, Memetic algorithms for the traveling salesman problem, Complex Systems 13 (4) (2001) 297–345.
[55] P. Merz, B. Freisleben, A genetic local search approach to the quadratic assignment problem, in: T. Bäck (Ed.), Seventh International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1997, pp. 465–472.
[56] P. Merz, B. Freisleben, Fitness landscape analysis and memetic algorithms for the quadratic assignment problem, IEEE Transactions on Evolutionary Computation 4 (4) (2000) 337–352.
[57] P. Merz, B. Freisleben, Memetic algorithms and the fitness landscape of the graph Bi-partitioning Problem, in: A.E. Eiben, T. Bäck, M. Schoenauer, H.-P. Schwefel (Eds.), PPSN 1998, in: Lecture Notes in Computer Science, vol. 1498, Springer-Verlag, Amsterdam, The Netherlands, 1998, pp. 765–774.
[58] P. Merz, B. Freisleben, Fitness landscapes, memetic algorithms and greedy operators for graph bi-partitioning, Evolutionary Computation 8 (8) (2000) 61–91.
[59] S. Wolf, P. Merz, A hybrid method for solving large-scale supply chain problems, in: Seventh European Conference on Evolutionary Computation in Combinatorial Optimization, EvoCOP 2007, in: Lecture Notes in Computer Science, vol. 4446, Springer, 2007, pp. 219–228.
[60] T. Fischer, P. Merz, A memetic algorithm for the optimal communication spanning tree problem, in: Hybrid Metaheuristics 4th International Workshop, HM 2007, in: Lecture Notes in Computer Science, vol. 4771, Springer, 2007, pp. 170–184.
[61] F. Glover, M. Laguna, Tabu Search, Kluwer Academic Publishers, 1997.
[62] W.E. Hart, Adaptive global optimization with local search, Ph.D. Thesis, University of California, San Diego, 1994.
[63] M.W.S. Land, Evolutionary algorithms with local search for combinatorial optimization, Ph.D. Thesis, University of California, San Diego, 1998.
[64] P. Merz, B. Freisleben, Fitness landscapes and memetic algorithm design, in: D. Corne, M. Dorigo, F. Glover (Eds.), New Ideas in Optimization, McGraw-Hill, 1999, pp. 245–260.
[65] N. Krasnogor, Studies in the theory and design space of memetic algorithms, Ph.D. Thesis, University of West England, 2002.
[66] T. Jones, One operator, one landscape, Technical Reports #95-02-025 (1996), Santa Fe Institute.
[67] N. Krasnogor, J. Smith, Memetic algorithms: the polynomial local search complexity theory perspective, Journal of Mathematical Modelling and Algorithms 7 (1) (2008) 3–24.
[68] D. Sudholt, Memetic algorithms with variable-depth search to overcome local optima, in: C. Ryan, M. Keijzer (Eds.), GECCO 2008, ACM Press, Atlanta GA, 2008, pp. 787–794.
[69] C. Houck, J. Joines, M. Kay, J. Wilson, Empirical investigation of the benefits of partial lamarckianism, Evolutionary Computation 5 (1) (1997) 31–60.
[70] D. Molina, F. Herrera, M. Lozano, Adaptive local search parameters for real-coded memetic algorithms, in: [227], 2005, pp. 888–895.
[71] Q.H. Nguyen, Y.-S. Ong, N. Krasnogor, A study on the design issues of memetic algorithm, in: [228], 2007, pp. 2390–2397.
[72] N.K. Bambha, S.S. Bhattacharyya, J. Teich, E. Zitzler, Systematic integration of parameterized local search into evolutionary algorithms, IEEE Transactions on Evolutionary Computation 8 (2) (2004) 137–155.
[73] D. Molina, M. Lozano, C. García-Martínez, F. Herrera, Memetic algorithms for continuous optimization based on local search chains, Evolutionary Computation 18 (1) (2010) 1–37.
[74] H. Mühlenbein, M. Schomisch, J. Born, The parallel genetic algorithm as function optimizer, in: R.K. Belew, L.B. Booker (Eds.), ICGA 1991, Morgan Kaufmann, San Diego CA, 1991, pp. 271–278.
[75] F. Glover, M. Laguna, R. Martí, Fundamentals of scatter search and path relinking, Control and Cybernetics 39 (3) (2000) 653–684.
[76] L. Eshelman, The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination, in: G.J.E. Rawlings (Ed.), Foundations of Genetic Algorithms, Morgan Kaufmann, 1991, pp. 265–283.
[77] M. Lozano, F. Herrera, N. Krasnogor, D. Molina, Real-coded memetic algorithms with crossover hill-climbing, Evolutionary Computation 12 (3) (2004) 273–302.
[78] G. Seront, H. Bersini, A new GA-local search hybrid for continuous optimization based on multi-level single linkage clustering, in: [235], 2000, pp. 90–95.
[79] T. Jones, Crossover, macromutation, and population-based search, in: L. Eshelman (Ed.), Sixth International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo CA, 1995, pp. 73–80.
[80] C. Cotta, J. Troya, Optimal discrete recombination: hybridising evolution strategies with the A^* algorithm, in: J. Mira, J. Sánchez-Andrés (Eds.), Engineering Applications of Bio-Inspired Artificial Neural Networks, in: Lecture Notes in Computer Science, vol. 1607, Springer-Verlag, Berlin Heidelberg, 1999, pp. 58–67.
[81] C. Cotta, J. Troya, Embedding branch and bound within evolutionary algorithms, Applied Intelligence 18 (2) (2003) 137–153.
[82] M. Lozano, C. García-Martínez, Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: overview and progress report, Computers & Operations Research 37 (3) (2010) 481–497.

[83] G. Delvecchio, C. Lofrumento, F. Neri, M. Sylos Labini, A fast evolutionary-deterministic algorithm to study multimodal current fields under safety level constraints, COMPEL: International Journal for Computation and Mathematics in Electrical and Electronic Engineering 25 (3) (2006) 599–608.
[84] B.Y. Qu, P.N. Suganthan, J.J. Liang, Differential evolution with neighborhood mutation for multimodal optimization, IEEE Transactions on Evolutionary Computation (2012) (in press).
[85] Y. Wang, B. Li, T. Weise, J. Wang, B. Yuan, Q. Tian, Self-adaptive learning based particle swarm optimization, Information Sciences 181 (20) (2011) 4515–4538.
[86] S. Das, S. Maity, B.-Y. Qu, P. Suganthan, Real-parameter evolutionary multimodal optimization: A survey of the state-of-the-art, Swarm and Evolutionary Computation 1 (2) (2011) 71–88.
[87] E. Marchiori, A. Steenbeek, An evolutionary algorithm for large scale set covering problems with application to airline crew scheduling, in: Scheduling, in Real World Applications of Evolutionary Computing, in: Lecture Notes in Computer Science, Springer-Verlag, 2000, pp. 367–381.
[88] A.V. Kononova, K.J. Hughes, M. Pourkashanian, D.B. Ingham, Fitness diversity based adaptive memetic algorithm for solving inverse problems of chemical kinetics, in: [228], 2007, pp. 2366–2373.
[89] A.V. Kononova, D.B. Ingham, M. Pourkashanian, 2008, Simple scheduled memetic algorithm for inverse problems in higher dimensions: application to chemical kinetics, in: [229], pp. 3906–3913.
[90] N. Noman, H. Iba, Enhancing differential evolution performance with local search for high dimensional function optimization, in: H.-G. Beyer, U.-M. O'Reilly (Eds.), GECCO 2005, ACM Press, Washington DC, 2005, pp. 967–974.
[91] N. Noman, H. Iba, Accelerating differential evolution using an adaptive local search, IEEE Transactions on Evolutionary Computation 12 (1) (2008) 107–125.
[92] A. Zamuda, J. Brest, B. Bošković, V. Žumer, High-dimensional real-parameter optimization using self-adaptive differential evolution algorithm with population size reduction, in: [229], 2008, pp. 2032–2039.
[93] J. Brest, B. Bošković, S. Greiner, V. Žumer, M.S. Maučec, Performance comparison of self-adaptive and adaptive differential evolution algorithms, Soft Computing 11 (7) (2007) 617–629.
[94] J. Brest, S. Greiner, B. Bošković, M. Mernik, V. Žumer, Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, IEEE Transactions on Evolutionary Computation 10 (6) (2006) 646–657.
[95] J. Brest, M.S. Maučec, Population size reduction for the differential evolution algorithm, Applied Intelligence 29 (3) (2008) 228–247.
[96] F. Neri, V. Tirronen, Scale factor local search in differential evolution, Memetic Computing 1 (2) (2009) 153–171.
[97] A. Caponio, A. Kononova, F. Neri, Differential evolution with scale factor local search for large scale problems, in: Y. Tenne, C.-K. Goh (Eds.), Computational Intelligence in Expensive Optimization Problems, in: Studies in Evolutionary Learning and Optimization, vol. 2, Springer-Verlag, 2010, pp. 297–323 (Chapter 12).
[98] J. Brest, M.S. Maucec, Self-adaptive differential evolution algorithm using population size reduction and three strategies, Soft Computing—A Fusion of Foundations, Methodologies and Applications 15 (11) (2011) 2157–2174.
[99] S.-Z. Zhao, P.N. Suganthan, S. Das, Self-adaptive differential evolution with multi-trajectory search for large-scale optimization, Soft Computing—A Fusion of Foundations, Methodologies and Applications 15 (11) (2011) 2175–2185.
[100] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Transactions on Evolutionary Computation 13 (2009) 398–417.
[101] L.-Y. Tseng, C. Chen, Multiple trajectory search for large scale global optimization, Proceedings of the IEEE Congress on Evolutionary Computation. (2008) 3052–3059.
[102] M. Weber, V. Tirronen, F. Neri, Scale factor inheritance mechanism in distributed differential evolution, Soft Computing—A Fusion of Foundations, Methodologies and Applications 14 (2010) 1187–1207.
[103] M. Weber, F. Neri, V. Tirronen, Shuffle or update parallel differential evolution for large-scale optimization, Soft Computing—A Fusion of Foundations, Methodologies and Applications 15 (11) (2011) 2089–2107.
[104] S. Handoko, C. Kwoh, Y. Ong, M. Lim, A study on constrained ma using ga and sqp: analytical vs. finite-difference gradients, in: [229], 2008, pp. 4031–4038.
[105] K. Deb, An efficient constraint handling method for genetic algorithms, Computer Methods in Applied Mechanics and Engineering 186 (2000) 311–338.
[106] V. Kelner, F. Capitanescu, O. Léonard, L. Wehenkel, A hybrid optimization technique coupling an evolutionary and a local search algorithm, Journal of Computational and Applied Mathematics 215 (2) (2008) 448–456.
[107] H. Singh, T. Ray, W. Smith, Performance of infeasibility empowered memetic algorithm for CEC 2010 constrained optimization problems, in: [230], 2010, pp. 1–8.
[108] T. Ray, H.K. Singh, A. Isaacs, W. Smith, Infeasibility driven evolutionary algorithm for constrained optimization, in: E. Mezura-Montes (Ed.), Constraint Handling in Evolutionary Optimization, Studies in Computational Intelligence, Springer, 2009, pp. 145–165.
[109] X. Li, X.-M. Liang, A hybrid adaptive evolutionary algorithm for constrained optimization, in: Third International Conference on Intelligent Information Hiding and Multimedia Signal Processing, vol. 2, 26–28 2007, pp. 338–341.
[110] A.S.S.M. Barkat Ullah, R. Sarker, D. Cornforth, C. Lokan, AMA: A new approach for solving constrained real-valued optimization problems, Soft Computing 13 (8–9) (2009) 741–762.
[111] A.S.S.M. Barkat Ullah, R. Sarker, C. Lokan, An agent-based memetic algorithm (AMA) for nonlinear optimization with equality constraints, in: CEC 2009, IEEE Press, Trondheim, Norway, 2009, pp. 70–77.
[112] B. Liu, H. Ma, X. Zhang, Y. Zhou, 2007, A memetic co-evolutionary differential evolution algorithm for constrained optimization, in: [228], pp. 2996–3002.
[113] R. Berretta, L.F. Rodrigues, A memetic algorithm for a multistage capacitated lot-sizing problem, International Journal of Production Economics 87 (1) (2004) 67–81.
[114] M. Boudia, C. Prins, A memetic algorithm with dynamic population management for an integrated production–distribution problem, European Journal of Operational Research 195 (3) (2009) 703–715.
[115] J.E. Gallardo, C. Cotta, A.J. Fernández, Solving weighted constraint satisfaction problems with memetic/exact hybrid algorithms, Journal of Artificial Intelligence Research 35 (2009) 533–555.
[116] Y.M. Park, J.B. Park, J.R. Won, A hybrid genetic algorithm/dynamic programming approach to optimal long-term generation expansion planning, International Journal of Electrical Power & Energy Systems 20 (4) (1998) 295–303.
[117] J. Gallardo, C. Cotta, A. Fernández, On the hybridization of memetic algorithms with branch-and-bound techniques, IEEE Transactions on Systems, Man, and Cybernetics, Part B 37 (1) (2007) 77–83.
[118] T. Ray, R. Sarker, Genetic algorithm for solving a gas lift optimization problem, Journal of Petroleum Science and Engineering 59 (1–2) (2007) 84–96.
[119] C. Prins, A simple and effective evolutionary algorithm for the vehicle routing problem, Computers & Operations Research 31 (12) (2004) 1985–2002.
[120] C. Prins, Two memetic algorithms for heterogeneous fleet vehicle routing problems, Engineering Applications of Artificial Intelligence 22 (6) (2009) 916–928 Artificial Intelligence Techniques for Supply Chain Management.
[121] A.E. Fallahi, C. Prins, R.W. Calvo, A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem, Computers & Operations Research 35 (5) (2008) 1725–1741.
[122] S.U. Ngueveu, C. Prins, R.W. Calvo, An effective memetic algorithm for the cumulative capacitated vehicle routing problem, Computers & Operations Research 37 (11) (2010) 1877–1885.
[123] K. Hasan, R. Sarker, D. Essam, Evolutionary scheduling with rescheduling option for sudden machine breakdowns, in: [230], 2010, pp. 1913–1920.
[124] S. Hasan, R. Sarker, D. Essam, D. Cornforth, Memetic algorithms for solving job-shop scheduling problems, Memetic Computing 1 (1) (2009) 69–83.
[125] Y. Marinakis, M. Marinaki, A hybrid genetic—particle swarm optimization algorithm for the vehicle routing problem, Expert Systems with Applications 37 (2) (2010) 1446–1455.
[126] J.E. Mendoza, B. Castanier, C. Guéret, A.L. Medaglia, N. Velasco, A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands, Computers & Operations Research 37 (11) (2010) 1886–1898, metaheuristics for Logistics and Vehicle Routing..
[127] C. Coello Coello, D.A. Van Veldhuizen, G. Lamont, Evolutionary Algorithms for Solving Multi-Objective Problems, Kluwer Academic Publishers, 2002.
[128] K. Deb, Multi-objective Optimization using Evolutionary Algorithms, John Wiley and Sons LTD, Chichester, UK, 2001, pp. 147–149.
[129] K. Deb, S. Agrawal, A. Pratab, T. Meyarivan, A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II, in: [233], 2000, pp. 849–858.
[130] J. Knowles, D. Corne, M-PAES: a memetic algorithm for multiobjective optimization, in: CEC 2000, IEEE Press, San Diego CA, 2000, pp. 325–332.
[131] J. Knowles, D.W. Corne, Approximating the nondominated front using the pareto archived evolution strategy, Evolutionary Computation 8 (2) (2000) 149–172.
[132] E. Angel, E. Bampis, L. Gourves, et al., A dynasearch neighborhood for the bicriteria traveling salesman problem, in: X. Gandibleux (Ed.), Metaheuristics for Multiobjective Optimisation, in: Lecture Notes in Economics and Mathematical Systems, vol. 535, Springer-Verlag, 2004, pp. 153–176.
[133] M. Basseur, Design of cooperative algorithms for multi-objective optimization: application to the flow-shop scheduling problem, 4OR: A Quarterly Journal of Operations Research 4 (3) (2006) 255–258.
[134] L. Paquete, M. Chiarandini, T. Stützle, Pareto local optimum sets in the biobjective traveling salesman problem: An experimental study, in: X. Gandibleux, M. Sevaux, K. Sörensen, V. Tkindt (Eds.), Meta-Heuristics for Multiobjective Optimisation, in: Lecture Notes in Economics and Mathematical Systems, vol. 535, Springer-Verlag, 2004, pp. 177–199.
[135] T. Lust, A. Jaszkiewicz, Speed-up techniques for solving large-scale biobjective TSP, Computers and Operations Research 37 (2010) 521–533.
[136] A. Caponio, F. Neri, Integrating cross-dominance adaptation in multi-objective memetic algorithms, in: C.-K. Goh, Y.-S. Ong, K.C. Tan (Eds.), Multi-Objective Memetic Algorithms, in: Studies in Computational Intelligence, vol. 171, Springer, 2009, pp. 325–351.
[137] H. Ishibuchi, T. Yoshida, T. Murata, Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 204–223.
[138] C. Fonseca, P. Fleming, An overview of evolutionary algorithms in multiobjective optimisation, Evolutionary Computation 3 (1) (1995) 1–16.
[139] E.L. Ulungu, J. Teghem, P. Fortemps, D. Tuyttens, MOSA method: a tool for solving multiobjective combinatorial optimization problems, Journal of Multi-Criteria Decision Analysis 8 (1999) 221–236.

[140] Q. Zhang, A. Zhou, Y. Jin, RM-MEDA: a regularity model based multiobjective estimation of distribution algorithm, IEEE Transactions on Evolutionary Computation 12 (1) (2008) 41–63.
[141] P. Czyżżak, A. Jaszkiewicz, Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimisation, Journal of Multi-Criteria Decision Analysis 7 (1998) 34–47.
[142] M. Hansen, Tabu search for multiobjective combinatorial optimization: TAMOCO, Control and Cybernetics 29 (3) (2000) 799–818.
[143] P. Hajela, C.Y. Lin, Genetic search strategies in multicriterion optimal design, Structural Optimization 4 (1992) 99–107.
[144] P. Serafini, Simulated annealing for multiple objective optimization problems, Tenth International Conference on Multiple Criteria Decision Making 1 (1992) 87–96.
[145] H. Ishibuchi, T. Murata, Multi-objective genetic local search algorithm and its application to flowshop scheduling, IEEE Transactions on Systems, Man and Cybernetics—Part C: Applications and Reviews 28 (3) (1998) 392–403.
[146] A. Jaszkiewicz, Genetic local search for multi-objective combinatorial optimization, European Journal of Operational Research 137 (2002) 50–71.
[147] A. Jaszkiewicz, 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) (2002) 402–412.
[148] Y. Jin, J. Branke, Evolutionary optimization in uncertain environments-a survey, IEEE Transactions on Evolutionary Computation 9 (3) (2005) 303–317.
[149] K.C. Giannakoglou, Design of optimal aerodynamic shapes using stochastic optimization methods and computational intelligence, International Review Journal Progress in Aerospace Sciences 38 (1) (2002) 43–76.
[150] M.K. Karakasis, K.C. Giannakoglou, On the use of surrogate evaluation models in multi-objective evolutionary algorithms, in: Proceedings of the European Conference on Computational Methods in Applied Sciences and Engineering, ECCOMAS 2004, 2004.
[151] M. Sefrioui, J. Périaux, A hierarchical genetic algorithm using multiple models for optimization, in: [233], 2000, pp. 879–888.
[152] M.K. Karakasis, D. Koubogiannis, K.C. Giannakoglou, Hierarchical distributed evolutionary algorithms in shape optimization, International Journal of Numerical Methods in Fluids 53 (3) (2007) 455–469.
[153] Y. Jin, M. Olhofer, B. Sendhoff, A framework for evolutionary optimization with approximate fitness functions, IEEE Transactions on Evolutionary Computation 6 (5) (2002) 481–494.
[154] A. Gaspar-Cunha, A. Vieira, A multi-objective evolutionary algorithm using neural networks to approximate fitness evaluations, International Journal of Computers, Systems and Signals 6 (1) (2005) 18–36.
[155] A.J. Booker, J.E. Dennis, P.D. Frank, D.B. Serafini, V. Torczon, M.W. Trosset, A rigorous framework for optimization of expensive functions by surrogates, Structural Optimization 17 (1) (1999) 1–13.
[156] A.R. Conn, K. Scheinberg, P.L. Toint, Recent progress in unconstrained nonlinear optimization without derivatives, Mathematical Programming 79 (1997) 397–414.
[157] J.F. Rodríguez, J.E. Renaud, L.T. Watson, Trust region augmented Lagrangian methods for sequential response surface approximation and optimization, ASME Journal of Mechanical Design 120 (1) (1998) 58–66.
[158] Y.-S. Ong, P.B. Nair, A.J. Keane, Evolutionary optimization of computationally expensive problems via surrogate modeling, AIAA Journal 41 (4) (2003) 687–696.
[159] Y.S. Ong, P.B. Nair, A.J. Keane, K.W. Wong, Surrogate-assisted evolutionary optimization frameworks for high-fidelity engineering design problems, in: Y. Jin (Ed.), Knowledge Incorporation in Evolutionary Computation, Springer, Berlin, Germany, 2004, pp. 307–331.
[160] Y. Tenne, S.W. Armfield, A memetic algorithm using a trust-region derivative-free optimization with quadratic modelling for optimization of expensive and noisy black-box functions, in: S. Yang, Y.-S. Ong, Y. Jin (Eds.), Evolutionary Computation in Dynamic and Uncertain Environments, in: Studies in Computational Intelligence, vol. 51, Springer-Verlag, 2007, pp. 389–415.
[161] Y. Tenne, S.W. Armfield, A versatile surrogate-assisted memetic algorithm for optimization of computationally expensive functions and its engineering applications, in: A. Yang, Y. Shan, L. Thu Bui (Eds.), Success in Evolutionary Computation, in: Studies in Computational Intelligence, vol. 92, Springer-Verlag, 2008, pp. 43–72.
[162] Z. Zhou, Y.-S. Ong, M.-H. Lim, B. Lee, Memetic algorithms using multi-surrogates for computationally expensive optimization problems, Journal of Soft Computing 11 (10) (2007) 957–971.
[163] Y.-S. Ong, Z. Zhou, D. Lim, Curse and blessing of uncertainty in evolutionary algorithm using approximation, in: CEC 2006, IEEE Press, 2006, pp. 2928–2935.
[164] D. Lim, Y. Jin, Y.-S. Ong, B. Sendhoff, Generalizing surrogate-assisted evolutionary computation, IEEE Transactions on Evolutionary Computation 14 (3) (2010) 329–355.
[165] Y. Tenne, A model-assisted memetic algorithm for expensive optimization problems, in: R. Chiong (Ed.), Nature-Inspired Algorithms for Optimisation, in: Studies in Computational Intelligence, No. 193, Springer-Verlag, 2009, pp. 133–169.
[166] Y. Tenne, S.W. Armfield, A framework for memetic optimization using variable global and local surrogate models, Journal of Soft Computing 13 (8) (2009).
[167] K. Tagawa, M. Masuoka, M. Tsukamoto, Robust optimum design of saw filters with the taguchi method and a memetic algorithm, in: [227], 2005, pp. 2146–2153.
[168] W.-J. Shyr, Robust control design for aircraft controllers via memetic algorithms, International Journal of Innovative Computing, Information and Control 5 (10A) (2009) 3133–3140.
[169] Y.-S. Ong, P.B. Nair, K.Y. Lum, Max–min surrogate-assisted evolutionary algorithm for robust aerodynamic design, IEEE Transactions on Evolutionary Computation 10 (4) (2006) 392–404.
[170] F. Neri, X. del Toro Garcia, G.L. Cascella, N. Salvatore, Surrogate assisted local search on PMSM drive design, COMPEL: International Journal for Computation and Mathematics in Electrical and Electronic Engineering 27 (3) (2008) 573–592.
[171] S. Ono, Y. Hirotani, S. Nakayama, A memetic algorithm for robust optimal solution search-hybridization of multi-objective genetic algorithm and quasi–newton method, International Journal of Innovative Computing, Information and Control 5 (12B) (2009) 5011–5019.
[172] S. Ono, Y. Yoshitake, S. Nakayama, Robust optimization using multi-objective particle swarm optimization, Artificial Life and Robotics 14 (2) (2009).
[173] W. Song, Multiobjective memetic algorithm and its application in robust airfoil shape optimization, in: C.-K. Goh, Y.-S. Ong, K.C. Tan (Eds.), Multi-Objective Memetic Algorithms, in: Studies in Computational Intelligence, vol. 171, Springer, 2009, pp. 389–402.
[174] D. Lim, Y.-S. Ong, M.-H. Lim, Y. Jin, Single/Multi-objective inverse robust evolutionary design methodology in the presence of uncertainty, in: S. Yang, Y.-S. Ong, Y. Jin (Eds.), Evolutionary Computation in Dynamic and Uncertain Environments, in: Studies in Computational Intelligence, vol. 51, Springer-Verlag, 2007, pp. 437–456.
[175] C.-K. Goh, K.C. Tan, Evolving the tradeoffs between pareto-optimality and robustness in multi-objective evolutionary algorithms, in: S. Yang, Y.-S. Ong, Y. Jin (Eds.), Evolutionary Computation in Dynamic and Uncertain Environments, in: Studies in Computational Intelligence, vol. 51, Springer-Verlag, 2007, pp. 457–478.
[176] E.K. Burke, P. De Causmaecker, G. De Maere, J. Mulder, M. Paelinck, G.V. Berghe, A multi-objective approach for robust airline scheduling, Computers and Operations Research 37 (2010) 822–832.
[177] K. Sörensen, M. Sevaux, A practical approach for robust and flexible vehicle routing using metaheuristics and Monte Carlo sampling, Journal of Mathematical Modelling and Algorithm 8 (4) (2009) 387–407.
[178] X. Yao, Y. Liu, A new evolutionary systems for evolving artificial neural networks, IEEE Transactions on Neural Networks 8 (3) (1997) 694–713.
[179] D.H. Kim, A. Abraham, A hybrid genetic algorithm and bacterial foraging approach for global optimization and robust tuning of PID controller with disturbance rejection, in: C. Grosan, A. Abraham, H. Ishibuchi (Eds.), Hybrid Evolutionary Algorithms, Springer, 2007, pp. 171–199.
[180] E. Mininno, F. Neri, A memetic differential evolution approach in noisy optimization, Journal of Memetic Computing 2 (2010) 111–135.
[181] T. Bärecke, M. Detyniecki, Memetic algorithms for inexact graph matching, in: [228], 2007, pp. 4238–4245.
[182] E. Ozcan, C.K. Mohan, Steady state memetic algorithm for partial shape matching, in: W.V. Porto (Ed.), 7th International Conference on Evolutionary Programming, in: Lecture Notes In Computer Science, vol. 1447, Springer-Verlag, 1998, pp. 527–536.
[183] J.-C. Créput, A. Koukam, The memetic self-organizing map approach to the vehicle routing problem, Journal of Soft Computing 12 (2008) 1125–1141.
[184] F. Neri, N. Kotilainen, M. Vapa, An adaptive global-local memetic algorithm to discover resources in p2p networks, in: EvoWorkshops. 2007, pp. 61–70.
[185] F. Neri, N. Kotilainen, M. Vapa, A memetic-neural approach to discover resources in P2P networks, in: C. Cotta, J. van Hemert (Eds.), Recent Advances in Evolutionary Computation for Combinatorial Optimization, in: Studies in Computational Intelligence, vol. 153/2008, Springer, 2008, pp. 113–129.
[186] F. Neri, R.A.E. Mäkinen, Hierarchical evolutionary algorithms and noise compensation via adaptation, in: S. Yang, Y.S. Ong, Y. Jin (Eds.), Evolutionary Computation in Dynamic and Uncertain Environments. Studies in Computational Intelligence, Springer-Verlag, 2007, pp. 345–369 (Chapter 15).
[187] F. Vavak, K.A. Jukes, T.C. Fogarty, A genetic algorithm with variable range of local search for tracking changing environments, in: [234], 196, pp. 376–385.
[188] F. Vavak, K.A. Jukes, T.C. Fogarty, Adaptive combustion balancing in multiple burner boiler using a genetic algorithm with variable range of local search, in: T. Bäck (Ed.), ICGA 1997, Morgan Kaufmann, 1997, pp. 719–726.
[189] F. Vavak, K.A. Jukes, T.C. Fogarty, Performance of a genetic algorithm with variable local search range relative to frequency of the environmental changes, in: J.R. Koza (Ed.), Third Annual Conference on Genetic Programming, Morgan Kaufmann, 1998, pp. 602–608.
[190] H. Wang, S. Yang, W.H. Ip, D. Wang, A particle swarm optimization based memetic algorithm for dynamic optimization problems, Natural Computing 3 (9) (2010) 703–725.
[191] I. Moser, T. Hendtlass, A simple and efficient multi-component algorithm for solving dynamic function optimisation problems, in: [228], 2007, pp. 252–259.
[192] S. Boettcher, A.G. Percus, Extremal optimization: methods derived from co-evolution, in: W. Banzhaf, et al. (Eds.), GECCO 1999, Morgan Kaufmann, 1999, pp. 825–832.
[193] I. Moser, R. Chiong, A Hooke–Jeeves based memetic algorithm for solving dynamic optimisation problems, in: E. Corchado (Ed.), 4th International Conference on Hybrid Artificial Intelligence Systems, HAIS 2009, in: Lecture Notes in Computer Science, vol. 5572, Springer, 2009, pp. 301–309.
[194] I. Moser, R. Chiong, Dynamic function optimisation with hybridised extremal dynamics, Journal of Memetic Computing 2 (2010) 137–148.

[195] J.A. Egea, E. Balsa-Canto, M.-S.G. Garćia, J.R. Ranga, Dynamic optimization of nonlinear processes with an enhanced scatter search method, Journal of Industrial Chemical Engineering Research 48 (2009) 4388–4401.
[196] W.T. Koo, C.K. Goh, K.C. Tan, A predictive gradient strategy for multiobjective evolutionary algorithms in a fast changing environment, Journal of Soft Computing 2 (2010) 87–110.
[197] H. Wang, D. Wang, S. Yang, A memetic algorithm with adaptive hill climbing strategy for dynamic optimization problems, Journal of Soft Computing 13 (2009) 763–780.
[198] Y.S. Ong, M.H. Lim, N. Zhu, K.W. Wong, Classification of adaptive memetic algorithms: a comparative study, IEEE Transactions On Systems, Man and Cybernetics—Part B 36 (1) (2006) 141–152.
[199] E.K. Burke, G. Kendall, E. Soubeiga, A tabu search hyperheuristic for timetabling and rostering, Journal of Heuristics 9 (6) (2003) 451–470.
[200] P. Cowling, G. Kendall, E. Soubeiga, A hyperheuristic approach to scheduling a sales summit, in: Proceedings of the Third International Conference on Practice and Theory of Automated Timetabling, in: Lecture Notes in Computer Science, vol. 2079, Springer, 2000, pp. 176–190.
[201] G. Kendall, P. Cowling, E. Soubeiga, Choice function and random hyperheuristics, in: Proceedings of the Fourth Asia-Pacific Conference on Simulated Evolution and Learning, 2002, pp. 667–671.
[202] P. Korošec, J. Šilc, B. Filipič, The differential ant-stigmergy algorithm, Information Sciences (2011) (in press).
[203] M.N. Le, Y.S. Ong, Y. Jin, B. Sendhoff, Lamarckian memetic algorithms: local optimum and connectivity structure analysis, Memetic Computing Journal 1 (3) (2009) 175–190.
[204] N. Krasnogor, J. Smith, A tutorial for competent memetic algorithms: model, taxonomy, and design issues, IEEE Transactions on Evolutionary Computation 9 (2005) 474–488.
[205] J.E. Smith, Coevolving memetic algorithms: a review and progress report, IEEE Transactions on Systems, Man, and Cybernetics, Part B 37 (1) (2007) 6–17.
[206] E.L. Yu, P.N. Suganthan, Ensemble of niching algorithms, Information Sciences 180 (15) (2010) 2815–2833.
[207] A. Caponio, F. Neri, V. Tirronen, Super-fit control adaptation in memetic differential evolution frameworks, Soft Computing-A Fusion of Foundations, Methodologies and Applications 13 (8) (2009) 811–831.
[208] F. Neri, V. Tirronen, T. Kärkkäinen, T. Rossi, Fitness diversity based adaptation in multimeme algorithms: A comparative study, in: [228], 2007, pp. 2374–2381.
[209] K. Chakhlevitch, P. Cowling, Hyperheuristics: recent developments, in: C. Cotta, M. Sevaux, K. Sörensen (Eds.), Adaptive and Multilevel Metaheuristics, in: Studies in Computational Intelligence, vol. 136, Springer-Verlag, Berlin Heidelberg, 2008, pp. 3–29.
[210] P. Cowling, G. Kendall, E. Soubeiga, A hyperheuristic approach to schedule a sales submit, in: E. Burke, W. Erben (Eds.), Selected Papers From the Third International Conference on Practice and Theory of Automated Timetabling III, in: Lecture Notes in Computer Science, vol. 2079, Springer-Verlag, Berlin Heidelberg, 2001, pp. 176–190.
[211] M. Gong, L. Jiao, L. Zhang, Baldwinian learning in clonal selection algorithm for optimization, Information Sciences 180 (8) (2010) 1218–1236.
[212] Q. Yuan, F. Qian, W. Du, A hybrid genetic algorithm with the baldwin effect, Information Sciences 180 (5) (2010) 640–652.
[213] R. Mallipeddi, P.N. Suganthan, Q.K. Pan, M.F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, Applied Soft Computing 11 (2) (2011) 1679–1696.
[214] R. Mallipeddi, S. Mallipeddi, P.N. Suganthan, Ensemble strategies with adaptive evolutionary programming, Information Sciences 180 (9) (2010) 1571–1581.
[215] N. Krasnogor, B. Blackburne, E. Burke, J. Hirst, Multimeme algorithms for proteine structure prediction, in: [232], 2002, pp. 769–778.
[216] N. Krasnogor, Coevolution of genes and memes in memetic algorithms, in: Wu, A. (Ed.), Proceedings of the 1999 Genetic and Evolutionary Computation Conference Workshop Program, 1999.
[217] N. Krasnogor, J. Smith, A memetic algorithm with self-adaptive local search: TSP as a case study, in: [235], 2000, pp. 987–994.
[218] N. Krasnogor, J. Smith, Emergence of profitable search strategies based on a simple inheritance mechanism, in: L. Spector, et al. (Eds.), GECCO 2001, Morgan Kaufmann, San Francisco, California, USA, 2001, pp. 432–439.
[219] J. Smith, Co-evolution of memetic algorithms: Initial investigations, in: [232], 2002, pp. 537–548.
[220] J. Smith, Protein structure prediction with co-evolving memetic algorithms, in: CEC 2003, IEEE Press, Canberra, Australia, 2003, pp. 2346–2353.
[221] J. Smith, The co-evolution of memetic algorithms for protein structure prediction, in: Recent Advances in Memetic Algorithms, in: Studies in Fuzziness and Soft Computing, vol. 166, Springer-Verlag, 2005, pp. 105–128.
[222] N. Krasnogor, Self-generating metaheuristics in bioinformatics: the protein structure comparison case, Genetic Programming and Evolvable Machines 5 (2) (2004) 181–201.
[223] N. Krasnogor, S. Gustafson, A study on the use of self-generation in memetic algorithms, Natural Computing 3 (1) (2004) 53–76.
[224] A. Nelder, R. Mead, A simplex method for function optimization, Computation Journal 7 (1965) 308–313.
[225] H.H. Rosenbrock, An automatic method for findong the greatest or least value of a function, The Computer Journal 3 (3) (1960) 175–184.
[226] R. Meuth, M.H. Lim, Y.S. Ong, D.C.W. II, A proposition on memes and meta-memes in computing for higher-order learning, Memetic Computing Journal 1 (2) (2009) 85–100.
[227] CEC, IEEE Congress on Evolutionary Computation 2005, IEEE Press, Edinburgh, UK, 2005.
[228] CEC, IEEE Congress on Evolutionary Computation 2007, IEEE Press, Singapore, 2007.
[229] CEC, IEEE Congress on Evolutionary Computation 2008, IEEE Press, Hong Kong, 2008.
[230] CEC, IEEE Congress on Evolutionary Computation 2010, IEEE Press, Barcelona, Spain, 2010.
[231] R. Männer, B. Manderick (Eds.), Parallel Problem Solving from Nature II, Elsevier, Brussels, Belgium, 1992.
[232] J.J. Merelo Guervós, Parallel Problem Solving from Nature VII, in: Lecture Notes in Computer Science, vol. 2439, Springer-Verlag, Granada, Spain, 2002.
[233] M. Schoenauer, et al. (Eds.), Parallel Problem Solving from Nature VI, in: Lecture Notes in Computer Science, vol. 1917, Springer-Verlag, Paris, France, 2000.
[234] H.-M. Voigt, W. Ebeling, I. Rechenberger, H.-P. Schwefel (Eds.), Parallel Problem Solving from Nature IV, in: Lecture Notes in Computer Science, vol. 1141, Springer-Verlag, Berlin, Germany, 1996.
[235] L.D. Whitley, et al. (Eds.), Genetic and Evolutionary Computation Conference—GECCO 2000, Morgan Kaufmann, Las Vegas NV, 2000.