Memetic Algorithms

20

Carlos Cotta, Luke Mathieson, and Pablo Moscato

Contents

Abstract

Memetic algorithms provide one of the most effective and flexible metaheuristic approaches for tackling hard optimization problems. Memetic algorithms address

C. Cotta (✉)
Departamento Lenguajes y Ciencias de la Computación, Universidad de Málaga, Málaga, Spain
e-mail: ccottap@lcc.uma.es

L. Mathieson
Centre for Bioinformatics, Biomarker Discovery and Information-Based Medicine, University of Newcastle, Callaghan, NSW, Australia
e-mail: luke.mathieson@newcastle.edu.au

P. Moscato
School of Electrical Engineering and Computing, Faculty of Engineering and Built Environment, The University of Newcastle, Callaghan, NSW, Australia
e-mail: pablo.moscato@newcastle.edu.au

R. Martí et al. (eds.), *Handbook of Heuristics*,
https://doi.org/10.1007/978-3-319-07124-4_29

the difficulty of developing high-performance universal heuristics by encouraging the exploitation of multiple heuristics acting in concert, making use of all available sources of information for a problem. This approach has resulted in a rich arsenal of heuristic algorithms and metaheuristic frameworks for many problems. This chapter discusses the philosophy of the memetic paradigm, lays out the structure of a memetic algorithm, develops several example algorithms, surveys recent work in the field, and discusses the possible future directions of memetic algorithms.

Keywords
Evolutionary algorithms · Hybridization · Local search · Memetic computing · Metaheuristics

Introduction

The effectiveness and efficiency of (meta)heuristics – and memetic algorithms which may be viewed as particularly good heuristics in this sense – rests upon their ability to explore the solution space thoroughly while avoiding exhaustive or near-exhaustive searching. If polynomial-time computability is taken as an approximation of tractability, then a polynomial-time algorithm can be viewed as a very clever search procedure; in these cases there is a small search space, or the search space can be reduced drastically. In dealing with intractable problems however, reducing the search space to a reasonable size is a much more difficult task. Most central is the problem of local versus global improvement; an improvement to a solution does not give any guarantee of movement toward the optimum. Actually, depending upon the problem under consideration, a local improvement may be a move *away* from the global optimum (hence the notion of deception in search algorithms [265]), or at the very least the solution may be getting closer to being trapped in a nonoptimal configuration for which no simple modification can lead to an improvement (i.e., a local optimum).

Thus, many (meta)heuristic methods include techniques for allowing non-improving alterations to a solution or for nonlocal moves across the search space in order to be able to escape from local optima [28]. Perhaps the most archetypical example of such a metaheuristic is the genetic algorithm (GA) [99, 110]: inspired by the principles of natural evolution, GAs maintain a population (i.e., a multiset) of solutions that are subject to successive phases of selection, reproduction (via recombination and mutation), and replacement. The use of a population of solutions provides a better chance of avoiding local optima than maintaining a single solution: on one hand, the search is driven by operators that (1) allow the search to take non-improving steps, most notably in the case of mutations, and (2) allow the search to move to significantly different portions of the search space, particularly by virtue of recombination. On the other hand, selection and replacement typically work on a global (population-wise) scale, meaning that non-improving solutions have a chance of persisting for a nontrivial amount of time, hence allowing escape from local optima. However, although this heuristic structure has proven quite effective, it relies almost entirely upon recombination mechanisms to improve solution quality, and evolutionary processes are slow. In particular, they are also less capable of fine-tuning solutions, that is, the progress toward a fully optimized solution once the algorithm has located its basin of attraction (i.e., the region of the search space from which a series of small – local – improvements can lead to a certain local optimum; see [131, 224]) is often sluggish. This is precisely in contrast to local search (single-solution or trajectory-based) techniques which can readily locate local optima (and hence are more sensitive to them).

To address this weakness, researchers began developing *hybridized* metaheuristics [29, 31], that is, metaheuristics which combine ideas from different search paradigms and/or different algorithms altogether. The underlying idea in such approaches is obviously trying to achieve some synergetic behavior whereby the deficiencies of a certain search technique are compensated by the combination with other techniques and their advantages are boosted due to this very same combination. This strict interpretation of the term *hybrid* has been broadened with time to encompass all forms of non-blind (i.e., not domain-independent) metaheuristics. Under this broad interpretation, hybridization is the process of augmenting a generic (problem-independent) metaheuristic with problem (or problem class, i.e., domain) knowledge. Since this augmentation is often achieved via the blend of different metaheuristic components, both interpretations are equivalent in most situations. The broad interpretation has, in any case, the advantage of fitting better into theoretical results such as those of Hart and Belew [107] and – most conspicuously – those of Wolpert and Macready [267] in the so-called no-free-lunch theorem, which states that search algorithms perform strictly in accordance with the amount and quality of the problem knowledge they incorporate. While these results spurred controversy in their time and have been refined [69, 70], the bottom line still holds.

Memetic algorithms (MAs) championed this philosophy. The denomination *memetic algorithm* was coined in [175] to characterize and codify these hybrid approaches. The term "memetic" was developed from Dawkin's [62] notion of a "meme" (from the Ancient Greek μίμημα, meaning "imitated thing") as a unit of cultural inheritance (and hence cultural evolution) – the cultural analogue of a gene. The use of the term meme was intended to capture the fact that information traits in human culture are subject to periods of lifetime learning and therefore they are different when transmitted to what they were when first acquired. This bears a strong resemblance with the Lamarckian model of evolution, whereby traits acquired during the lifetime of an individual are transmitted to its offspring. It is therefore not surprising that MAs are sometimes disguised under other denominations featuring the terms "Lamarckian" or "hybrid."

While the initial conception of memetic search did not include the idea of GAs or evolutionary algorithms (EAs) whatsoever [178], it turned out that these techniques were ideal recipients for exploiting the metaphor of MAs, namely, having a collection of "agents" alternating periods of self-improvement with phases of cooperation and competition, cf. [177]. Indeed, early MAs mixed GAs and EAs with simulated annealing and tabu search [180, 198], eventually developing the idea that MAs are EAs endowed with some kind of local search (LS) technique, leading to the restrictive definition MA = EA + LS [218]. Note however that the

Algorithm 1: A generic memetic algorithm

```
1: parfor $j$ := 1 to $\mu$ do                ▷ Initialise population $Pop$ of search agents
2:     $pop_i$ := new SearchAgent
3: end parfor
4: repeat
5:     parfor $j$ := 1 to $\mu$ do                          ▷ individual learning phase
6:         $pop_i$.learn()
7:     end parfor
8:     $Pop$.cooperate()                                      ▷ cooperation phase
9:     $Pop$.compete()                                        ▷ competition phase
10: until Termination condition is true.
```

central concept of MAs is not to tie ourselves to a particular heuristic approach or metaphor, but to provide a coherent structure for employing *several* heuristics (including exact methods [30]) that deploy *complementary* heuristics exploiting *all available knowledge*. Thus EA + LS $\subset$ MA is a consequence of this broader definition of MAs, cf. [50]. The next section will explore in more detail the structure of an MA with particular emphasis on the classical characterization of the paradigm.

Structure of a Memetic Algorithm

As mentioned above, early definitions of MAs envisioned the paradigm as a pragmatic integration of ideas from different metaheuristics. These were orchestrated in terms of a collection of search agents carrying out individual explorations (i.e., lifetime learning) of the space of solutions and engaging in periodic phases of cooperation and competition [198]. An abstract formulation of such an approach is provided in Algorithm 1. This pseudocode matches the initial conception of MAs as an inherently parallel approach whereby a collection of local searchers (simulated annealing in early developments [178]) run either concurrently or physically in parallel and establish synchronization points in which information was exchanged among them. This said, this depiction of MAs is still generic enough to encompass most actual incarnations of the paradigm as shown later, as it captures the essential feature of MAs, namely, the carefully crafted interplay between global (population-based) and local (individual-based) search. It must be noted that the terms *global* and *local* are used in connection to the mechanics of the search rather than to the ability of eventually (or asymptotically) finding the global optimum. It is certainly the case that many local search approaches (simulated annealing, tabu search, etc.) are capable of escaping from local optima and navigate the search space in order to find the global optimum. The distinctive feature of these techniques (as opposed to, e.g., genetic algorithms) is that they do this following a trajectory-based approach.

Skeleton of a Classical Memetic Algorithm

Following early works in which the population-based aspects of MAs, namely, the collection of agents and the synchronized stages of cooperation and competition, were captured by a genetic algorithm [180], the classical memetic model coalesced. The basic skeleton of such an MA is relatively straightforward, adding little additional complexity beyond that of a GA. Algorithm 2 gives a pseudocode sketch of the salient structure, using *local search* as a placeholder for any particular individual improvement heuristic including, for instance, a complete exact algorithm like branch and bound, and others that guarantee optimality of the final solution obtained when they stop. Although a small structural change to a typical GA, the inclusion of the individual improvement phase can dramatically alter its performance. This mix allows the metaheuristic to benefit from the solution diversity engendered by the evolutionary approach, but to avoid the lethargic pace of improvement via more directed optimization: instead of relying random processes subjected to fitness-based selection alone, each individual solution is optimized before the evolutionary mechanism is applied, significantly increasing the rate at which individual solutions converge to an optima.

Algorithm 2: A local-search based memetic algorithm

```
1:  $Pop$ := new Solution [$popsize$]                ▷ Create population $Pop$
2:  for $i \in Pop$ do
3:      $i$.initialise()                             ▷ Generate initial solution
4:      $i$.local-search()   ▷ Individual improvement. Comprises solution evaluation
5:  end for
6:  repeat
7:      for $j$ := 1 to #$recombinations$ do         ▷ Recombination phase
8:          $Parents$ := $Pop$.select($numparents$)   ▷ Select parent set $Parents \subseteq Pop$
9:          $c$ := $Parents$.recombine()   ▷ Recombine parents to create child $c$
10:         $c$.local-search()
11:         $Pop$.update(c)                ▷ Inserts new solution in the population
12:     end for
13:     for $j$ := 1 to #$mutations$ do              ▷ Mutation phase
14:         $i$ := $Pop$.select(1);
15:         $i_m$ := $i$.mutate()                    ▷ Mutate solution $i$
16:         $i_m$.local-search()
17:         $Pop$.update($i_m$)
18:     end for
19:     if $Pop$.converged() then          ▷ Refresh population upon convergence
20:         $Pop$.restart()
21:     end if
22: until Termination condition is true.
```

The structure of an MA is quite flexible, and the performance of the implementation, both in terms of solution quality and speed, can be affected by a number of factors. As an evolutionary, population-based metaheuristic, the typical issues regarding choice and implementation of mutation and recombination operators are inherited from the GA paradigm. For those familiar with GAs however, it should be readily apparent that the individual improvement phase is most likely to be the computational bottleneck – the improvement of every individual and the subsequence evaluation of every individual are inherently expensive simply because they are done for every individual (as shown in [167], it can easily take up to 95% of the computational cost of the algorithm). The tradeoff is that with a good choice of individual improvement heuristic, far fewer generations of mutation and recombination are required. The careful reader will also notice that the local search procedure (and, typically, any individual improvement heuristic) is highly amenable to parallelization. This helps to ameliorate the cost of the individual improvement, but more importantly lends the MA approach a high degree of scalability.

From the point of view of the different components into which a classical MA can be dissected, all of which encapsulate some portion of problem knowledge. Consider, for instance, recombination. This is the component that captures most appropriately the idea of agent cooperation. Such a cooperation is typically established between a pair of agents but can in general involve an arbitrary number of *parents* [76] (notice nevertheless that in this case some forms of heuristic recombination can be very complex [53]). The generic idea of a knowledge-augmented recombination operator is to combine, in an intelligent way, pieces of information contained in the parents. How these pieces are defined is a problem-dependent issue that arises from the issue of representation in EAs. The underlying objective of an appropriate representation would be to have solutions described by some structured collection of objects whose values truly capture solution features of relevance (i.e., ultimately responsible for determining whether a solution is good or not). Even from the beginnings of MAs, the importance of developing a suitable representation – in which evolution of the representation reflects the correlation of elements in the fitness landscape – was identified [175]. This is a substantial topic for which the interested reader is referred to, e.g., [226]. Focusing on the *smart* manipulation of these information units (however they are defined), that is, processing them in a problem-specific way instead of using domain-independent templates (such as those in [217]), the goal is picking the right combination of such units from either parent. Of course, this is easier said than done (and in fact, doing it is in general *provably* hard for arbitrary problems and/or definitions of *right* combination – see the discussion on the polynomial merger complexity class in [177, 179]), but there are numerous heuristic ideas in the literature that can be used to this end. In many cases – and following design advice already present in classical texts of hybridization pioneers, e.g., [61] – these ideas are based on the use of problem-specific heuristics such as greedy algorithms [132,185], backtracking [48], dynamic programming [123], or branch and bound [55], just to mention a few.

Mutation is another classical operator, well known for its role of maintaining a continuous supply of genetic diversity that can be subsequently exploited by the remaining operators. In certain EA models, such as evolutionary programming [86], it actually bears sole responsibility for driving the search. Note that while this latter philosophy can be also used in an MA context – see section "An Example Memetic Algorithm for WEIGHTED CONSTRAINT SATISFACTION PROBLEMS" – it is typically the case that MAs use sophisticated recombination operators such as those described before. Thus, the criticism of recombination being just a disguised form of macromutation would not apply to them. Moreover, due to the presence of a local search stage in the main evolutionary cycle, one has to be careful to pick a mutation operator whose effects on solutions cannot be trivially undone by the local search, since that would defeat the very purpose of mutation. Following this line, in some cases there are MAs that even refrain from using mutation, e.g., [163, 262]. While such a decision could be further vindicated by the fact that MAs usually feature a population-restart procedure (see line 20 in Algorithm 2) and hence premature convergence is not so troublesome, this is not the most common course of action. An appropriate mutation operator (i.e., one using a sufficiently different neighborhood to that used by the local searcher) is often utilized. In fact, it is not unusual to have more than one such mutation operator, e.g., [152, 231], much like in metaheuristic approaches such as variable neighborhood search [105] (VNS). In some cases, these multiple mutation operators are used with the purpose of exerting different degrees of perturbation (i.e., *light* and *heavy* mutations) depending on the convergence of the population [87].

As to the local search component, it can take the form of any stand-alone method such as hill climbing, simulated annealing, tabu search, variable neighborhood search, etc. [199]. The choice of a particular technique must take into account two major issues, namely, its parameterization and its interaction with the remaining components of the algorithm. Regarding the latter, and in addition to the issues discussed above in connection to the mutation operator, one has to consider the interplay between the local searcher and the recombination operator. For example, a highly intensive local search procedure may be better suited to interact with a more diversification-oriented recombination operator – see, e.g., [88]. This heuristic recipe does not necessarily conflict with the use of a powerful recombination operator (see, e.g., [91]) but underlines that the knowledge embedded in either component, recombination operator and local search heuristic, must be complementary in terms of the effect they produce in the search dynamics (much as was discussed for the mutation operator). An interesting analysis of these issues from the point of view of fitness landscapes is provided in [170]. Whatever the definition of the neighborhood is (and notice that it can be complex, even combining several simpler neighborhood schemes), it is often crucial to be able to evaluate solutions incrementally for performance reasons [106]. It is desirable to avoid having to resort to a full evaluation and only recompute the fitness contribution of the solution components that were modified. This may require the use of appropriate data structures and is normally associated to discrete optimization (the high nonlinearity – and sometimes even the lack of a closed fitness function – often makes this complicated in continuous optimization).

The parameterization of the local search heuristic is another complex issue. This includes both high-level algorithmic aspects as well as low-level parameters. The high-level aspects include factors such as when to apply local search, to which solutions it should be applied and which local search operator to apply. The low-level aspects include parameters such as the breadth (number of neighbors explored in each iteration of the local search heuristic) and depth (how many iterations of local search will be performed). Further discussion is given in [245]. Determining an adequate setting for these parameters is crucial for the performance of the algorithm since it has been shown theoretically that small parameter changes can turn a problem from being polynomial-time solvable with high probability to requiring super-polynomial (even exponential) time [144, 244]. Unfortunately, a priori design guidelines to provably avoid this kind of behavior are ruled out by intractability results [245]. Thus, design by analogy and empirical testing seem to be the handiest tools to approach this endeavor (although self-parameterization is an appealing alternative that is increasingly gaining relevance – see section "Future-Generation Memetic Algorithms"). In this regard, it has been, for example, shown in several contexts that partial Lamarckism [112], that is, not applying local search to every individual but just applying it some probability p_{LS}, can produce notably better results than a fully Lamarckian approach [49, 126] although the best value of this parameter is problem dependent. On a related note with regard to the depth of the local search, it has been also proposed in the literature to save the store of the local search together with the solution it was applied to, so as to resume the process from that point if required [173, 174].

The restarting procedure is another important element in an MA. The goal of this procedure is to perform a warm reinitialization of the population when the search is deemed stagnated (i.e., the population has converged to a suboptimal state). Of course, that stagnation can be hindered by taking preventive measures such as the light/heavy mutation scheme mentioned before, the use of spatial structure in populations [250] (see also next subsection), or some other diversity-preservation policy [236] – see also [188]. A more drastic measure may be eventually required though. For that purpose, a common approach is to keep a certain percentage of the current population and use the solution creation mechanism (the one used to create the initial population – line 3 in Algorithm 2) to complete the new population. Regarding the former, they constitute a seed that allows keeping a part of the search momentum without having to start from scratch. As to the latter, notice that they need not be purely random solutions but any available constructive heuristic can be used for this purpose.

A Note on More Complicated Memetic Algorithms

Although Algorithms 1 and 2 lay out a basic MA framework, the structure can be made significantly more complex. As the central motivation of MAs is to exploit all available information, the restriction of any particular component would be antithetical. Apart from employing different heuristics, *multiple* heuristics can be employed in concert. It is easy to combine different individual improvement heuristics, applying them to different individuals and different populations, in parallel, in sequence, or even in competition. Similarly the population-based heuristic can employ multiple improvement techniques – such approaches are well known in the GA community.

Moscato and Tinetti [181] demonstrate a more complicated MA that uses a number of heuristics in concert and to achieve different goals within the algorithm. The algorithm employs a *tree-structured population* where the population is divided into subpopulations of size 4, composed in a ternary tree structure:

1. Each subpopulation is divided into a *leader* node and three *supporters*. The supporters are stored one level below their leader.
2. The intermediate nodes in the tree hold an individual that is part of two populations; it is the leader of the three supporters lower in the tree and a support of its leader higher in the tree.
3. The number of subpopulations can be manipulated by adding levels to the tree.

Each individual can be optimized using a local search procedure that selects from a variety of local optimization moves: approximate 2-OPT, One-City Insertion, and Two-City Insertion [134,156]. Genetic recombination occurs "normally" within each subpopulation. The leader individual represents the best tour in the subpopulation. Note that the overlap of subpopulations ensures improvement propagates up the tree. The small subpopulation size, however, can quickly lead to a lack of diversity, in which case the recombination mechanism switches to an external recombination procedure for the lowest level of the tree.

In this example a number of variations on the basic structure of an MA are evident: multiple local search variants, a multipopulation variation of a GA which itself employs multiple recombination procedures. Another example of a more complex improvement strategy is given by Moscato [176], where a small population of individuals is maintained (only 16 individuals, each a binary vector), with a tabu search procedure for individual improvement. Again, in a small population, a loss of diversity is a potential drawback. To combat this, each individual notes the 16 best single-bit-flip moves available. When diversity falls below a given threshold, instead of following a simple tabu search approach, individual i makes move i from its list of best moves. This deliberate (potentially) nonoptimal has the effect of spreading the individuals further across the configuration space, increasing diversity. Once diversity is restored, the normal tabu search optimization is restored.

The continuation of these ideas has led to the development of what are now called *self-adaptive* memetic algorithms, which allow the context-specific, dynamic application of different heuristics or tuning of search parameters by the algorithm itself. See section "Future-Generation Memetic Algorithms".

Memetic Algorithms in Practice

This section presents two extended examples of memetic algorithms for specific problems – NETWORK ALIGNMENT and WEIGHTED CONSTRAINT SATISFACTION PROBLEMS – and surveys recent interesting applications of memetic algorithms in different fields.

An Example Memetic Algorithm for NETWORK ALIGNMENT

The optimization version of the basic NETWORK ALIGNMENT problem takes as input two networks G_1 and G_2 and asks for an injective partial mapping $f : V(G_1) \rightarrow V(G_2)$ between the vertices of the two networks that maximize $\sum_{u,v \in V(G_1)} \tau_f(u, v)$ where

$$\tau_f = \begin{cases} 1 \text{ if } uv \in E(G_1) \text{ and } f(u)f(v) \in E(G_2) \\ 0 \text{ otherwise} \end{cases}$$

It may be assumed that the mapping is total and bijective by adding "dummy" vertices to the smaller network. Of course τ_f is open to variation as are the precise details of f, leading to many variants of NETWORK ALIGNMENT. The decision variant of NETWORK ALIGNMENT is NP-complete [138] and W[1]-complete (Mathieson et al., 2015, Using network alignment to uncover topological structure and identify consumer behaviour modelling constructs, unpublished Manuscript), suggesting, subject to standard complexity assumptions, that no suitably efficient exact algorithm for NETWORK ALIGNMENT exists, making it a prime candidate for heuristic methods.

In developing a memetic algorithm for this (and any) problem, it is necessary (at a minimum) to select an individual solution representation, mutation, and recombination operators and an individual improvement heuristic and its attending concerns.

For NETWORK ALIGNMENT, the most direct individual representation is the mapping itself. Assuming any necessary dummy vertices have already been added, the mapping can be represented by, for example, an array of size $|V(G_1)|$ storing a permutation of $V(G_2)$. For ease of representation, it is sufficient to assign each vertex a unique integer in the range $[0, |V(G_1)|]$. An alternative representation suitable for NETWORK ALIGNMENT would be to store the alignment of the edges, making computing the basic fitness function simple, but the individual could be polynomially larger, impacting the efficiency of mutation, recombination, individual improvement, and even evaluation. Moreover care would need to be taken as to how to determine which vertices were aligned.

With this individual representation, a simple, reasonable mutation operator is that of a random shuffle, where each element of an individual is randomly swapped with another, randomly chosen element, with a given probability. A naïve recombination operator is, given two parent individuals, to select a linear segment of the individual (i.e., a set of contiguous indices) and swap the mappings for those indices between the parents (with adjustment to take care of duplication of elements), producing two children. This recombination is commonly known as a partially matched crossover [100]. However, considering the problem at hand, it is easy to see that this choice may be somewhat inefficient. The NETWORK ALIGNMENT problem, in essence, seeks to preserve as much topological structure (i.e., edge matchings) as possible – in this sense it is a relaxed GRAPH ISOMORPHISM problem. Swapping a set of arbitrarily chosen indices is unlikely to preserve interesting structure, contrary to the goal of a recombination operator, which is to produce children of higher quality than their parents by mixing the better parts of the parents, aiming to place the child solution closer to the global optima. For NETWORK ALIGNMENT, it is much more interesting to preserve neighborhoods of vertices in this regard. So a better choice of recombination operator is to select a vertex and its 2-neighborhood (all vertices at distance at most 2) as the set of indices which will be swapped.

To complete the GA component, a tournament selection process is employed to choose the individuals included in the new generation and a restart mechanism whereby the best solution is recorded and the population is restarted if no improvement has been observed after a given number of generations.

For individual improvement, a local search heuristic is used, where the neighborhood of each individual is the 2-swap neighborhood – the set of individuals obtained by swapping any two elements. The search is implemented by selecting an element in the individual and taking the optimal swap in the local neighborhood. If this is not the identity mapping, the neighbors of the preimage of the swapped vertex are placed into a list of vertices to swap. If no initial swap is found, the process is repeated with a new starting point until a swap is found or all vertices have been tested.

In combination with the skeletons given by Algorithms 1 and 2, these components constitute an MA for NETWORK ALIGNMENT. The reader will notice that, even without considering more complicated approaches, there are a number of tunable parameters present. These include the probabilities and frequencies which control mutation and recombination (as in GAs) and, more specifically for MAs, the frequency and application régime of the individual improvement step. The individual improvement may be applied, at essentially one extreme, regularly, to all individuals, or at the other extreme, only when the evolutionary progress slows and to a select few individuals, or of course in some intermediate régime. As discussed in section "A Note on More Complicated Memetic Algorithms", an adaptive approach could also be taken, allowing the algorithm to adjust these parameters itself.

An Example Memetic Algorithm for WEIGHTED CONSTRAINT SATISFACTION PROBLEMS

WEIGHTED CONSTRAINT SATISFACTION PROBLEMS (WCSPs) are a general class of combinatorial problems in which (i) solutions are assignment of values to a collection of variables, each of them taken from a possibly different domain, (ii) there are hard constraints making some particular combinations of variable values infeasible, and (iii) there are some soft constraints establishing preferences among solutions. For example, consider a school timetabling problem in which courses have to be fit into different time slots: no two courses can use the same time slot if they are taught by the same lecturer (a hard constraint), and lecturer preferences (e.g., teaching in the morning or in the afternoon) have to be respected if possible. In essence, both types of constraints can be represented by defining a collection of integer functions f_i, one for each constraint; these functions are used to weight the fulfillment/violation of the corresponding constraint, and therefore an objective function F (to be minimized, without loss of generality) can be built by summing them. Thus, it will be typically the case that hard constraints have a much larger weight (even infinite if violated) than soft constraints.

Formally, a WCSP can be characterized as a triplet $\langle \mathscr{X}, \mathscr{D}, \mathscr{F} \rangle$, where each $x_i \in \mathscr{X}$, $1 \leqslant i \leqslant n$ is a problem variable whose domain is $D_i \in \mathscr{D}$. Each function $f_j \in \mathscr{F}$, $1 \leqslant j \leqslant m$ has signature $f_j : V_j \to \mathbb{N}$, where $V_j \in 2^{\mathscr{X}}$ is the subset of variables involved in the j-th constraint. With this formulation, a naïve evolutionary approach can be defined by using the Cartesian product $\mathscr{S} = D_1 \times \cdots \times D_n$ as search space, taking the fitness function to be $F(x) = \sum_j \hat{f}_j(x)$ (where $\hat{f}_i$ is a function that picks from its argument the variables in V_j and feeds them to f_j), and utilizing standard operations for recombination and mutation. Such an approach is however going to perform poorly in general due to the lack of problem-specific knowledge. A much more sensible approach can be built on the basis of (i) a smart recombination operator and (ii) a powerful local search technique.

Regarding recombination, it is very easy to define a greedy recombination mechanism for WCSPs: (1) start from a solution s with all variables unassigned, (2) sort constraints in some particular order (arbitrary or heuristically selected) $j_1, \cdots, j_m$, and (3) traverse this ordered list of constraints, checking for each j_k the variables in V_{j_k} that are still unassigned in s, constructing two (or as many as parents) candidate sets using the assigned values in s plus the values that the remaining variables in V_{j_k} take in either parent, and keeping the candidate set v minimizing $f_{j_k}(v)$ (which is subsequently used to expand the solution s). This procedure has been used, for example, in [57] for the construction of Golomb rulers and in [225] for the construction of balanced incomplete blocks, to cite just two examples.

It is possible to define a more intensive recombination approach by taking ideas from complete techniques [59]. More precisely, a complete technique can be used to explore the set of potential solutions that can be created using a given collection of parents, returning the best solution attainable. Different possibilities can be used for this purpose such as branch and bound [55] or integer linear programming techniques [164]. A more WCSP-specific approach can be found in the use of bucket elimination (BE) [63]. BE can be regarded as a dynamic programming approach based on the successive elimination of variables and the use of an auxiliary table to store the best value of the fitness function for specific variable combinations. More precisely, BE considers some ordering of the variables (again, arbitrarily or heuristically selected – it must be noted that while the particular choice ordering is irrelevant for correction purposes, it can have a huge impact in the computational complexity of the algorithm though) $i_1, \cdots, i_n$. Then, it traverses this sequence and for each variable x_{i_k} (1) determines the constraints $\mathscr{C} \subseteq \mathscr{F}$ in which x_{i_k} is involved; (2) computes the bucket

$$B_{i_k} = \left(\cup_{f_j \in \mathscr{C}} V_j\right) \setminus \{x_{i_k}\},$$

namely, the collection of variables related to x_{i_k} in any constraint; (3) determines for each combination t of values for variables in B_{i_k} the value v_t^* for x_{i_k} such that $w = \sum_{f_j \in \mathscr{C}} \hat{f}_j(t \cdot (x_{i_k} = v))$ is minimal; and (4) removes $\mathscr{C}$ from $\mathscr{F}$ and adds a new constraint f' with domain $V' = B_{i_k}$ defined as $f'(t) = \sum_{f_j \in \mathscr{C}} \hat{f}_j(t \cdot (x_{i_k} = v_t^*))$. When all variables have been eliminated, the optimal cost w is found, and one only has to trace back the process (using the auxiliary table) to determine the best variable assignment [93]. This procedure has been used with great success in [91] for solving the Maximum Density Still Life Problem in conduction with a local search procedure based on tabu search.

A potential drawback of recombination schemes such as those defined above is scalability: the use of an exact technique for recombination is less costly than using it to solve the problem completely from scratch, but its cost will nevertheless grow with the problem size until becoming impractical at some point. To alleviate this problem, the granularity of the representation can be adjusted [54], that is, grouping variables in larger chunks which are subsequently used as basic units for the purposes of constructing solutions (hence reducing the number of potential solutions attainable and therefore the computational cost of the exact technique). In the context of the BE method described before, this approach is termed mini-buckets [64] and can be readily applied to the recombination mechanism described above [93]. Another source of difficulties is the existence of symmetries or partial isomorphisms between solutions. This scenario is typical in many WCSPs in which variables or groups thereof can be relabeled without altering the solution. In such a situation, recombination can reduce to macromutation unless it is effectively capable of identifying correspondences between variables in different parents. This is, for instance, done with success in [171] in the context of clustering genomic data. Of course, it may be very complex in general to find a perfect matching between variables in an arbitrary WCSP with symmetries. In problems for which this is deemed too complicated or time-consuming, it must be noted that a recombination-less MA – essentially a population of local searchers subject to interleaved phases of self-improvement and competition via selection/replacement, much in the line of go-with-the-winner approaches [8] – can also provide acceptable results. This is, for example, the case of the Social Golfer Problem, a WCSP with a large degree of symmetry that was successfully attacked using a memetic evolutionary programming approach [56] (an MA propelled by selection, mutation, and local improvement).

A Brief Survey of Recent Memetic Algorithm Applications

In recent years, MAs have become a significant part of the optimization toolkit and have become particularly well used in recent years. As a rough gauge, the number of academic papers (found via searching DBLP and ISI Web of Science for relevant papers with the word "memetic" in their title, abstract, or keywords) published has risen to over 300 per year since 2011, with thousands of academic publications in total since 1998. Possibly the most interesting aspect of this expanding interest in memetic algorithms is the diversity of techniques and application areas.

Memetic Algorithms in the Wild

While many algorithms developed in the areas of Computer Science and Optimization are demonstrated via application to practical problems drawn from a variety of areas, a more reliable indicator of the effectiveness of a technique is the adoption of the technique as a tool within the communities from which the problems are drawn. The following briefly surveys some of the areas in which memetic algorithms have been successfully applied. Table 1 gives an overview of the breadth of application areas for memetic algorithms, with recent references. Of course this table is far from exhaustive, even within the application areas mentioned. As a matter of fact, in some areas the number of memetic applications has deserved individualized treatment in specialized surveys, e.g., scheduling and timetabling [50], engineering and design [37], bioinformatics [24], etc. – see also [189] for a recent general application

Table 1 Some recent publications reporting on memetic algorithm applications by field of application

Application area	References
Biology	[89, 90, 186, 187, 197, 201, 238–241, 269, 270]
Chemistry	[77–80, 108, 194]
Chemical engineering	[47, 75, 140, 141, 150, 158, 242, 253–257]
Data compression	[146, 169, 246, 271, 275, 280]
Drug design	[104, 109, 130, 161, 191, 192, 251, 252]
Electronic engineering	[41, 46, 97, 98, 101, 113–118, 135, 200, 205–207, 210, 214, 272]
Finance	[15, 45, 71, 243]
Geoscience	[36, 258]
Image analysis	[66, 127, 162, 228]
Materials science & engineering	[14, 25, 124, 125, 222, 260]
Microarray analysis	[12, 13, 19, 73, 94, 148, 167, 182, 208, 237, 278]
Computer networking	[16, 215, 216, 248, 261, 276]
Oncology	[1, 39, 74, 133, 166, 230, 251, 252, 279, 281]
Operations research	[2, 6, 10, 22, 40, 68, 95, 111, 129, 159, 160, 165, 209, 212, 213, 219, 227, 259, 266, 268, 274, 277]
Physics	[5, 103, 139, 264, 273]
Power engineering	[17, 18, 26, 67, 120, 121, 136, 147, 149, 151, 157, 168, 183, 195, 220, 223, 232]

survey. The breadth of the application areas suggests a significant generality and flexibility in the memetic paradigm.

Memetic Speciation

Along with a wide set of application areas, memetic algorithms have also embraced many forms, employing a wide variety of combinations of population-based heuristics and individual improvement heuristics. Table 2 lists some of the more prominent combinations, with example references for each. Not only are different combinations of population-based heuristic and individual improvement heuristic extant, more exotic memetic algorithms that use heuristics of only one type, or multiple heuristics of each type, exist. The adaptability of memetic algorithms to parallel implementation also encourages the use of multiple different types of heuristics simultaneously – the exploitation of *all* available knowledge is, after all, the central idea of the memetic paradigm.

Table 2 Some varietal combinations of heuristics forming memetic algorithms

Population-based heuristic	Individual improvement heuristic	References
Ant colony optimization	Local search	[44, 72, 84, 154]
Bee colony optimization	–	[32, 34]
	Random optimization	[21]
	Nelder-Mead simplex	[85]
–	Nelder-Mead simplex with bidirectional random optimization	[4]
Binary differential evolution	Tabu search	[102]
Continuous differential evolution	Pool of strategies	[122, 249]
	Hooke-Jeeves-like	[211]
	Stochastic local search	[190]
Cross entropy	Hill climbing, tabu search	[11]
Genetic algorithm	Local search	[15, 22]
	Tabu search	[27, 33, 92, 153, 155, 263]
	Mathematical programming	[254]
Genetic algorithm with particle swarm optimization	–	[26, 119]
Particle swarm optimization	Local search	[20, 35, 65, 121, 274]
	Variable neighborhood search	[9]
	Sequential quadratic programming	[221]
Particle swarm optimization with differential evolution	Nelder-Mead simplex with Rosenbrock algorithm	[38]

Future-Generation Memetic Algorithms

Back in the days when MAs were just a nascent approach for optimization, different visions of what MAs would be in the future were foreseen. Among these, maybe the one which has come closest to reality refers to the self-$\star$ capabilities [23] of the paradigm and more precisely to self-generation properties. Early works envisioned that the algorithm could work on two timescales, one in which solutions would be optimized and another one in which the problem-solving strategies used to optimize solutions would be themselves optimized [177]. In essence, this has been a long-standing goal in metaheuristics. It is widely acknowledged that the design of an effective problem-solving technique is in itself a hard task. Attempting to transfer a part of this design effort to the actual metaheuristic is just the logical course of action [58] – see, for example, the corpus of research in hyperheuristics [42, 60]. This latter approach is actually related to what has been termed "meta-Lamarckian" learning [202], a memetic approach in which a collection of local searchers is available and there is a decision-maker that decides which of them should be applied to specific solutions based on different criteria (e.g., the past performance of each local searcher, the adequacy of the current solution for being improved by a certain local searcher according to past experience, etc.). A much more general approach was provided by multi-memetic or multimeme algorithms [142, 143, 145]. In this approach an encoding of a local searcher (ranging from the definition of the neighborhood or pivot rule used up to a full algorithmic description of the procedure) is attached to each solution and evolves alongside it. Thus, the algorithm not only looks for improved solutions but also for algorithmic structures capable of improving the latter. The next natural step is detaching these memes from the genes and have them evolve in separate populations [233–235], paving the way for the emergence of complex structures of interacting memes [43]. An overview of adaptation in MAs is provided in [203]. This view of memes as explicit representations of problem-solving strategies that interact in a complex and dynamic way within an evolutionary context for optimization purposes leads to the notion of memetic computing [193, 204] – see [189] for a literature review on memetic computing. A further iteration of this concept is to apply metaheuristic approaches to develop worst-case instances of a problem, which can then be fed back into the process of optimizing the algorithm. This technique has been explored in regard to sorting [51] and the TRAVELLING SALESMAN PROBLEM [3].

Another dimension along which some early ideas (farfetched at their time) about MAs may become a reality is parallel computing. The deployment of metaheuristics in parallel and/or distributed environments is by no means new [7] and has been extensively used since the late 1980s; see, for example, [184, 247]. However, the continuous evolution of computational platforms is dragging these parallel modes along, forcing them to adapt to new scenarios. Thus, whereas early works often assumed dedicated local area networks, it is nowadays more common to have emerging computational environments such as peer-to-peer networks [172] and volunteer computing networks [229], which are much more pervasive, of a larger scale and inherently dynamic. Coping with the complex, dynamic structure of the computational substrate is undoubtedly a challenge. Fortunately, population-based metaheuristics have been shown to be intrinsically robust at a fine-grain scale [128] and can be endowed with appropriate churn-aware strategies if required [196]. They are therefore ripe for being deployed on these platforms to exploit the possibilities they offer. In this line – and connected to the previous discussion on meme evolution and interaction – some initial concepts revolving around "meme pools," that is, repositories of problem-solving methods to be used synergistically, acquire a new scope more akin to service-oriented architectures [96]. Furthermore, to build on the idea of automated self-design of the MA requires the ability to keep or gather some sort of distributed knowledge about the state of the search and make design decisions on its basis. Some ideas from multi-agent systems and epistemic logic were proposed as potential tools for this purpose [52], but the concept still remains largely unexplored.

There are also opportunities for the development of MAs (and GAs) at the small scale. Any use of recombination operators is naturally limited by the expectation that the recombination step will be performed many, many times during a run of the algorithm. This leads to the requirement that a recombination operator must be able to be implemented very efficiently. Traditionally this would mean at most linear or close to linear time in the size of the individual (of course, ideally constant time). This immediately rules out the possibility of optimal recombination strategies for many problems, as typically such strategies would be NP-hard. Parameterized complexity, for example, offers some opportunity to exploit the naturally arising parameters in many recombination strategies. If such parameters are small, or can be made small, then the complexity of optimal recombination may be effectively reduced to polynomial time [53,81–83]. For further reading on the challenges raised by evolutionary approaches to optimization, many of the problems posed in [52] remain open.

Conclusion

Since their primordial conception in the late 1980s, memetic algorithms have developed to become one of the most adaptable and flexible metaheuristic approaches available. While many heuristic techniques perform well for some problems, the No-Free-Lunch theorem [267] guarantees that their performance falters on the majority of problems. Memetic algorithms, with their insistence on adaptability and utilitarianism (both on the part of the algorithm and the implementer), are free to exploit the performance of multiple approaches and choose the best suited for the problem at hand.

The adaptability, efficiency, and amenability to the current availability of large-scale parallelism, including traditional parallel architectures along with GPU computing and cloud- and peer-based approaches, along with a tendency toward modularity in implementation, have led to their adoption across a broad range of fields with excellent results. The field of memetic algorithms research has grown dramatically since 1998. With over 2000 academic papers published at a current rate of over 300 per year, the field is vibrant and dynamic. The importance and influence of memetic algorithms has grown such that Thomson Reuters selected it as one of the top ten research fronts in Mathematics, Computer Science, and Engineering in 2013 [137]. To put it simply, memetic algorithms are one of the most flexible and effective tools in the heuristic toolbox and a key technique for anyone involved in combinatorial optimization to learn.

Cross-References

▶ Adaptive and Multilevel Metaheuristics
▶ Evolution Strategies
▶ Hyper-heuristics
▶ Scatter Search
▶ Tabu Search
▶ Variable Neighborhood Search

Acknowledgments Carlos Cotta acknowledges support from the MINECO under the project EphemeCH (TIN2014-56494-C4-1-P), from the Junta de Andalucía under the project P10-TIC-6083 (DNEMESIS) and from the Universidad de Málaga, Campus de Excelencia Internacional Andalucía Tech.

Pablo Moscato acknowledges support from the Australian Research Council Future Fellowship FT120100060 and Australian Research Council Discovery Projects DP120102576 and DP140104183.

References

1. Abbass HA (2002) An evolutionary artificial neural networks approach for breast cancer diagnosis. Artif Intell Med 25(3):265–281
2. Afsar HM, Prins C, Santos AC (2014) Exact and heuristic algorithms for solving the generalized vehicle routing problem with flexible fleet size. Int Trans Oper Res 21(1): 153–175
3. Ahammed F, Moscato P (2011) Evolving L-systems as an intelligent design approach to find classes of difficult-to-solve traveling salesman problem instances. In: Di Chio C et al (eds) Applications of Evolutionary Computation. Lecture Notes in Computer Science, vol 6624. Springer, Berlin, pp 1–11
4. Ahandani MA, Vakil-Baghmisheh MT, Talebi M (2014) Hybridizing local search algorithms for global optimization. Comput Optim Appl 59(3):725–748
5. Ahn Y, Park J, Lee CG, Kim JW, Jung SY (2010) Novel memetic algorithm implemented with GA (genetic algorithm) and MADS (mesh adaptive direct search) for optimal design of electromagnetic system. IEEE Trans Magn 46(6):1982–1985
6. Al-Betar MA, Khader AT, Abu Doush I (2014) Memetic techniques for examination timetabling. Ann Oper Res 218(1):23–50
7. Alba E (2005) Parallel metaheuristics: a new class of algorithms. Wiley-Interscience, Hoboken
8. Aldous D, Vazirani U (1994) "go with the winners" algorithms. In: Proceedings of 35th IEEE Symposium on Foundations of Computer Science. IEEE Press, Los Alamitos, pp 492–501

9. Ali AF, Hassanien AE, Snasel V, Tolba MF (2014) A new hybrid particle swarm optimization with variable neighborhood search for solving unconstrained global optimization problems. In: Kromer P, Abraham A, Snasel V (eds) Fifth International Conference on Innovations in Bio-inspired Computing and Applications. Advances in Intelligent Systems and Computing, vol 303. Springer, Berlin/Ostrava, pp 151–160
10. Amaya JE, Cotta C, Fernández-Leiva AJ (2012) Solving the tool switching problem with memetic algorithms. Artif Intell Eng Des Anal Manuf 26:221–235
11. Amaya JE, Cotta C, Fernández-Leiva AJ (2013) Cross entropy-based memetic algorithms: an application study over the tool switching problem. Int J Comput Intell Syst 6(3):559–584
12. Andres Gallo C, Andrea Carballido J, Ponzoni I (2009) BiHEA: a hybrid evolutionary approach for microarray biclustering. In: Guimaraes K, Panchenko A, Przytycka T (eds) 4th Brazilian Symposium on Bioinformatics (BSB 2009). Lecture Notes in Bioinformatics, vol 5676. Springer, Berlin/Porto Alegre, pp 36–47
13. Andres Gallo C, Andrea Carballido J, Ponzoni I (2009) Microarray biclustering: a novel memetic approach based on the PISA platform. In: Pizzuti C, Ritchie M, Giacobini M (eds) 7th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics. Lecture Notes in Computer Science, vol 5483. Springer, Berlin/Tubingen, pp 44–55
14. António CC (2014) A memetic algorithm based on multiple learning procedures for global optimal design of composite structures. Memetic Comput 6(2):113–131
15. Arab A, Alfi A (2015) An adaptive gradient descent-based local search in memetic algorithm applied to optimal controller design. Inf Sci 299:117–142
16. Arivudainambi D, Balaji S, Rekha D (2014) Improved memetic algorithm for energy efficient target coverage in wireless sensor networks. In: 11th IEEE International Conference on Networking, Sensing and Control (ICNSC), Miami, pp 261–266
17. Arshi SS, Zolfaghari A, Mirvakili SM (2014) A multi-objective shuffled frog leaping algorithm for in-core fuel management optimization. Comput Phys Commun 185(10):2622–2628
18. de Assis LS, Vizcaino Gonzalez JF, Usberti FL, Lyra C, Cavellucci C, Von Zuben FJ (2015) Switch allocation problems in power distribution systems. IEEE Trans Power Syst 30(1):246–253
19. Ayadi W, Hao JK (2014) A memetic algorithm for discovering negative correlation biclusters of DNA microarray data. Neurocomputing 145:14–22
20. Aziz M, Tayarani-N MH (2014) An adaptive memetic particle swarm optimization algorithm for finding large-scale latin hypercube designs. Eng Appl Artif Intell 36:222–237
21. Baghmisheh MTV, Ahandani MA, Talebi M (2008) Frequency modulation sound parameter identification using novel hybrid evolutionary algorithms. In: International Symposium on Telecommunications. IEEE, Tehran, pp 67–72
22. Benlic U, Hao JK (2015) Memetic search for the quadratic assignment problem. Expert Syst Appl 42(1):584–595
23. Berns A, Ghosh S (2009) Dissecting self-$\star$ properties. In: Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems. IEEE Press, San Francisco, pp 10–19
24. Berretta R, Cotta C, Moscato P (2012) Memetic algorithms in bioinformatics. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin/Heidelberg, pp 261–271
25. Bertagnoli G, Giordano L, Mancini S (2014) Optimization of concrete shells using genetic algorithms. ZAMM: Zeitschrift für Angewandte Mathematik und Mechanik 94(1–2, SI): 43–54
26. Biao S, Hua HC, Hua YX, Chuan H (2014) Mutation particle swarm optimization algorithm for solving the optimal operation model of thermal power plants. J Renew Sustain Energy 6(4):043118
27. Bilal N, Galinier P, Guibault F (2014) An iterated-tabu-search heuristic for a variant of the partial set covering problem. J Heuristics 20(2):143–164

28. Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput Surv 35(3):268–308
29. Blum C, Blesa Aguilera MJ, Roli A, Sampels M (2008) Hybrid metaheuristics: an emerging approach to optimization. Studies in computational intelligence, vol 144. Springer, Berlin/Heidelberg
30. Blum C, Cotta C, Fernández AJ, Gallardo JE, Mastrolilli M (2008) Hybridizations of metaheuristics with branch & bound derivates. In: Blum C, Blesa Aguilera MJ, Roli A, Sampels M (eds) Hybrid Metaheuristics, an Emerging Approach to Optimization. Studies in Computational Intelligence, vol 144. Springer, Berlin/Heidelberg, pp 85–116
31. Blum C, Puchinger J, Raidl GR, Roli A (2011) Hybrid metaheuristics in combinatorial optimization: a survey. Appl Soft Comput 11(6):4135–4151
32. Bose D, Biswas S, Vasilakos AV, Laha S (2014) Optimal filter design using an improved artificial bee colony algorithm. Inform Sci 281:443–461
33. Boskovic B, Brglez F, Brest J (2014) Low-autocorrelation binary sequences: on the performance of memetic-tabu and self-avoiding walk solvers. CoRR abs/1406.5301
34. Bullinaria JA, AlYahya K (2014) Artificial bee colony training of neural networks: comparison with back-propagation. Memetic Comput 6(3):171–182
35. Cai K, Zhang J, Zhou C, Cao X, Tang K (2012) Using computational intelligence for large scale air route networks design. Appl Soft Comput 12(9):2790–2800
36. Caorsi S, Massa A, Pastorino M, Randazzo A (2003) Electromagnetic detection of dielectric scatterers using phaseless synthetic and real data and the memetic algorithm. IEEE Trans Geosci Remote Sens 41(12):2745–2753
37. Caponio A, Neri F (2012) Memetic algorithms in engineering and design. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin/Heidelberg, pp 241–260
38. Caponio A, Neri F, Tirronen V (2009) Super-fit control adaptation in memetic differential evolution frameworks. Soft Comput 13(8–9, SI):811–831
39. Capp A, Inostroza-Ponta M, Bill D, Moscato P, Lai C, Christie D, Lamb D, Turner S, Joseph D, Matthews J, Atkinson C, North J, Poulsen M, Spry NA, Tai KH, Wynne C, Duchesne G, Steigler A, Denham JW (2009) Is there more than one proctitis syndrome? A revisitation using data from the TROG 96.01 trial. Radiother Oncol 90(3):400–407
40. Cattaruzza D, Absi N, Feillet D, Vidal T (2014) A memetic algorithm for the multi trip vehicle routing problem. Eur J Oper Res 236(3):833–848
41. Chabuk T, Reggia J, Lohn J, Linden D (2012) Causally-guided evolutionary optimization and its application to antenna array design. Integr Comput Aided Eng 19(2):111–124
42. Chakhlevitch K, Cowling P (2008) Hyperheuristics: recent developments. In: Cotta C, Sevaux M, Sörensen K (eds) Adaptive and Multilevel Metaheuristics. Studies in Computational Intelligence, vol 136. Springer, Berlin/Heidelberg, pp 3–29
43. Chen X, Ong YS (2012) A conceptual modeling of meme complexes in stochastic search. IEEE Trans Syst Man Cybern C 42(5):612–625
44. Chen X, Ong YS, Feng L, Lim MH, Chen C, Ho CS (2013) Towards believable resource gathering behaviours in real-time strategy games with a memetic ant colony system. In: Cho S, Sato A, Kim K (eds) 17th Asia Pacific Symposium on Intelligent and Evolutionary Systems. Procedia Computer Science, vol 24. Elsevier, Seoul, pp 143–151
45. Chiam SC, Tan KC, Mamun AM (2009) A memetic model of evolutionary pso for computational finance applications. Expert Syst Appl 36(2):3695–3711
46. Chowdhury A, Giri R, Ghosh A, Das S, Abraham A, Snasel V (2010) Linear antenna array synthesis using fitness-adaptive differential evolution algorithm. In: IEEE Congress on Evolutionary Computation (CEC 2010). IEEE, Barcelona
47. Conradie AVE, Aldrich C (2010) Neurocontrol of a multi-effect batch distillation pilot plant based on evolutionary reinforcement learning. Chem Eng Sci 65(5):1627–1643
48. Cotta C (2003) Protein structure prediction using evolutionary algorithms hybridized with backtracking. In: Mira J, Álvarez J (eds) Artificial Neural Nets Problem Solving Methods. Lecture Notes in Computer Science, vol 2687. Springer, Berlin/Heidelberg, pp 321–328

49. Cotta C (2005) Memetic algorithms with partial lamarckism for the shortest common supersequence problem. In: Mira J, Álvarez J (eds) Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach. Lecture Notes in Computer Science, vol 3562. Springer, Berlin/Heidelberg, pp 84–91
50. Cotta C, Fernández A (2007) Memetic algorithms in planning, scheduling, and timetabling. In: Dahal K, Tan K, Cowling P (eds) Evolutionary Scheduling. Studies in Computational Intelligence, vol 49. Springer, Berlin, pp 1–30
51. Cotta C, Moscato P (2003) A mixed evolutionary-statistical analysis of an algorithm's complexity. Appl Math Lett 16:41–47
52. Cotta C, Moscato P (2004) Evolutionary computation: challenges and duties. In: Menon A (ed) Frontiers of Evolutionary Computation. Kluwer Academic, Boston, pp 53–72
53. Cotta C, Moscato P (2005) The parameterized complexity of multiparent recombination. In: Proceedings of the 6th Metaheuristic International Conference – MIC 2005, Universität Wien, Vienna, pp 237–242
54. Cotta C, Troya JM (2000) On the influence of the representation granularity in heuristic forma recombination. In: Carroll J, Damiani E, Haddad H, Oppenheim D (eds) Applied Computing 2000. ACM, New York, pp 433–439
55. Cotta C, Troya JM (2003) Embedding branch and bound within evolutionary algorithms. Appl Intell 18(2):137–153
56. Cotta C, Dotú I, Fernández AJ, Van Hentenryck P (2006) Scheduling social golfers with memetic evolutionary programming. In: Almeida F et al (eds) Hybrid Metaheuristics – HM 2006. Lecture Notes in Computer Science, vol 4030. Springer, Berlin/Heidelberg, pp 150–161
57. Cotta C, Dotú I, Fernández AJ, Van Hentenryck P (2007) Local search-based hybrid algorithms for finding golomb rulers. Constraints 12(3):263–291
58. Cotta C, Sevaux M, Sörensen K (2008) Adaptive and multilevel metaheuristics. Studies in computational intelligence, vol 136. Springer, Berlin/Heidelberg
59. Cotta C, Fernández Leiva AJ, Gallardo JE (2012) Memetic algorithms and complete techniques. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin/Heidelberg, pp 189–200
60. Cowling P, Kendall G, Soubeiga E (2008) A hyperheuristic approach to schedule a sales submit. In: Burke E, Erben W (eds) PATAT 2000. Lecture Notes in Computer Science, vol 2079. Springer, Berlin/Heidelberg, pp 176–190
61. Davis L (1991) Handbook of genetic algorithms. Van Nostrand Reinhold Computer Library, New York
62. Dawkins R (1976) The selfish gene. Clarendon Press, Oxford
63. Dechter R (1999) Bucket elimination: a unifying framework for reasoning. Artif Intell 113(1–2):41–85
64. Detcher R, Rish I (2003) Mini-buckets: a general scheme for bounded inference. J ACM 50(2):107–153
65. Devi S, Jadhav DG, Pattnaik SS (2011) PSO based memetic algorithm for unimodal and multimodal function optimization. In: Panigrahi B, Suganthan P, Das S, Satapathy S (eds) 2nd Swarm, Evolutionary and Memetic Computing Conference. Lecture Notes in Computer Science, vol 7076. Springer, Berlin, pp 127–134
66. Di Gesù V, Lo Bosco G, Millonzi F, Valenti C (2008) A memetic algorithm for binary image reconstruction. In: Proceedings of the 2008 International Workshop on Combinatorial Image Analysis, Buffalo. Lecture Notes in Computer Science, vol 4958, pp 384–395
67. Dimitroulas DK, Georgilakis PS (2011) A new memetic algorithm approach for the price based unit commitment problem. Appl Energy 88(12):4687–4699
68. Divsalar A, Vansteenwegen P, Sorensen K, Cattrysse D (2014) A memetic algorithm for the orienteering problem with hotel selection. Eur J Oper Res 237(1):29–49
69. Droste S, Jansen T, Wegener I (1999) Perhaps not a free lunch but at least a free appetizer. In: Banzhaf W, Daida JM, Eiben AE, Garzon MH, Honavar V, Jakiela MJ, Smith RE (eds) Proceedings of the First Genetic and Evolutionary Computation Conference – GECCO 1999. Morgan Kaufmann, Orlando, pp 833–839

70. Droste S, Jansen T, Wegener I (2002) Optimization with randomized search heuristics – the (A)NFL theorem, realistic scenarios, and difficult functions. Theor Comput Sci 287(1): 131–144
71. Du J, Rada R (2012) Memetic algorithms, domain knowledge, and financial investing. Memetic Comput 4(2):109–125
72. Duan H, Yu X (2007) Hybrid ant colony optimization using memetic algorithm for traveling salesman problem. In: IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning. IEEE, Honolulu, pp 92–95
73. Duval B, Hao JK (2010) Advances in metaheuristics for gene selection and classification of microarray data. Brief Bioinform 11(1):127–141
74. Duval B, Hao JK, Hernández JCH (2009) A memetic algorithm for gene selection and molecular classification of cancer. In: 11th Annual Conference on Genetic and Evolutionary Computation – GECCO '09. ACM, New York, pp 201–208
75. Egea JA, Balsa-Canto E, Garcia MSG, Banga JR (2009) Dynamic optimization of non-linear processes with an enhanced scatter search method. Ind Eng Chem Res 48(9): 4388–4401
76. Eiben AE, Raue PE, Ruttkay Z (1994) Genetic algorithms with multi-parent recombination. In: Davidor Y, Schwefel HP, Männer R (eds) Parallel Problem Solving from Nature III. Lecture Notes in Computer Science, vol 866. Springer, Berlin/Heidelberg, pp 78–87
77. Ellabaan M, Ong YS, Handoko SD, Kwoh CK, Man HY (2013) Discovering unique, low-energy transition states using evolutionary molecular memetic computing. IEEE Comput Intell Mag 8(3):54–63
78. Ellabaan MM, Handoko SD, Ong YS, Kwoh CK, Bahnassy SA, Elassawy FM, Man HY (2012) A tree-structured covalent-bond-driven molecular memetic algorithm for optimization of ring-deficient molecules. Comput Math Appl 64(12, SI):3792–3804
79. Ellabaan MMH, Chen X, Nguyen QH (2012) Multi-modal valley-adaptive memetic algorithm for efficient discovery of first-order saddle points. In: Bui L et al (eds) Simulated Evolution and Learning. Lecture Notes in Computer Science, vol 7673. Berlin/Heidelberg, pp 83–92
80. Ellabaan MMH, Ong YS, Nguyen QC, Kuo JL (2012) Evolutionary discovery of transition states in water clusters. J Theor Comput Chem 11(5):965–995
81. Eremeev AV (2008) On complexity of optimal recombination for binary representations of solutions. Evol Comput 16(1):127–147
82. Eremeev AV (2011) On complexity of the optimal recombination for the travelling salesman problem. In: Proceedings of the 11th European Conference on Evolutionary Computation in Combinatorial Optimization. Lecture Notes in Computer Science, vol 6622. Springer, Berlin, pp 215–225
83. Eremeev AV, Kovalenko JV (2014) Optimal recombination in genetic algorithms, Part II. Yugoslav J Oper Res 24(2):165–186
84. Fathi M, Rodriguez V, Jesus Alvarez M (2014) A novel memetic ant colony optimization-based heuristic algorithm for solving the assembly line part feeding problem. Inter J Adv Manuf Technol 75(1–4):629–643
85. Fister I, Fister I Jr, Brest J, Zumer V (2012) Memetic artificial bee colony algorithm for large-scale global optimization. In: IEEE Congress on Evolutionary Computation (CEC 2012). IEEE, Brisbane
86. Fogel LJ, Owens AJ, Walsh MJ (1966) Artificial intelligence through simulated evolution. Wiley, New York
87. França PM, Gupta JND, Mendes AS, Moscato P, Veltnik KJ (2005) Evolutionary algorithms for scheduling a flowshop manufacturing cell with sequence dependent family setups. Comput Ind Eng 48:491–506
88. Freisleben B, Merz P (1996) A genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems. In: 1996 IEEE International Conference on Evolutionary Computation, Nagoya. IEEE Press, pp 616–621

89. Gallardo JE (2012) A multilevel probabilistic beam search algorithm for the shortest common supersequence problem. PLoS ONE 7(12):1–14
90. Gallardo JE, Cotta C (2015) A GRASP-based memetic algorithm with path relinking for the far from most string problem. Eng Appl Artif Intell. https://doi.org/10.1016/j.engappai.2015.01.020
91. Gallardo JE, Cotta C, Fernández AJ (2006) A memetic algorithm with bucket elimination for the still life problem. In: Gottlieb J, Raidl G (eds) Evolutionary Computation in Combinatorial Optimization. Lecture Notes in Computer Science, vol 3906. Springer, Berlin/Heidelberg, pp 73–85
92. Gallardo JE, Cotta C, Fernández AJ (2009) Finding low autocorrelation binary sequences with memetic algorithms. Appl Soft Comput 9(4):1252–1262
93. Gallardo JE, Cotta C, Fernández AJ (2009) Solving weighted constraint satisfaction problems with memetic/exact hybrid algorithms. J Artif Intell Res 35:533–555
94. Gallo CA, Carballido JA, Ponzoni I (2009) Microarray biclustering: a novel memetic approach based on the PISA platform. In: Pizzuti C, Ritchie MD, Giacobini M (eds) Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics, Tübingen. Lecture Notes in Computer Science, vol 5483. Berlin/Heidelberg, pp 44–55
95. Gao L, Zhang C, Li X, Wang L (2014) Discrete electromagnetism-like mechanism algorithm for assembly sequences planning. Int J Prod Res 52(12):3485–3503
96. García-Sánchez P, González J, Castillo P, Arenas M, Merelo-Guervós J (2013) Service oriented evolutionary algorithms. Soft Comput 17(6):1059–1075
97. Garcia-Valverde T, Garcia-Sola A, Botia JA, Gomez-Skarmeta A (2012) Automatic design of an indoor user location infrastructure using a memetic multiobjective approach. IEEE Trans Syst Man Cybern C Appl Rev 42(5, SI):704–709
98. Ghosh P, Zafar H (2010) Linear array geometry synthesis with minimum side lobe level and null control using dynamic multi-swarm particle swarm optimizer with local search. In: Panigrahi B, Das S, Suganthan P, Dash S (eds) 1st International Conference on Swarm, Evolutionary, and Memetic Computing. Lecture Notes in Computer Science, vol 6466. Springer, Berlin/Chennai, pp 701–708
99. Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley Longman Publishing Co., Inc., Boston
100. Goldberg DE, Lingle R Jr (1985) Alleles, loci, and the traveling salesman problem. In: Grefenstette JJ (ed) Proceedings of the First International Conference on Genetic Algorithms and Their Applications. Lawrence Erlbaum Associates, Hillsdale
101. Goudos SK, Gotsis KA, Siakavara K, Vafiadis EE, Sahalos JN (2013) A multi-objective approach to subarrayed linear antenna arrays design based on memetic differential evolution. IEEE Trans Antennas and Propag 61(6):3042–3052
102. Gu X, Li Y, Jia J (2015) Feature selection for transient stability assessment based on kernelized fuzzy rough sets and memetic algorithm. Int J Electr Power Energy Syst 64: 664–670
103. Guimaraes FG, Lowther DA, Ramirez JA (2008) Analysis of the computational cost of approximation-based hybrid evolutionary algorithms in electromagnetic design. IEEE Trans Magn 44(6):1130–1133
104. Handoko SD, Ouyang X, Su CTT, Kwoh CK, Ong YS (2012) QuickVina: accelerating AutoDock Vina using gradient-based heuristics for global optimization. IEEE-ACM Trans Comput Biol Bioinf 9(5):1266–1272
105. Hansen P, Mladenović N (2001) Variable neighborhood search: principles and applications. Eur J Oper Res 130(3):449–467
106. Hao J (2012) Memetic algorithms in discrete optimization. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin/Heidelberg, pp 73–94

107. Hart W, Belew R (1991) Optimizing an arbitrary function is hard for the genetic algorithm. In: Belew R, Booker L (eds) Fourth International Conference on Genetic Algorithms. Morgan Kaufmann, San Mateo, pp 190–195
108. Hervas C, Silva M (2007) Memetic algorithms-based artificial multiplicative neural models selection for resolving multi-component mixtures based on dynamic responses. Chemom Intel Lab Syst 85(2):232–242
109. Hirsch R, Mullergoymann C (1995) Fitting of diffusion-coefficients in a 3-compartment sustained-release drug formulation using a genetic algorithm. Int J Pharm 120(2): 229–234
110. Holland JH (1992) Adaptation in natural and artificial systems. MIT, Cambridge
111. Hosseini S, Farahani RZ, Dullaert W, Raa B, Rajabi M, Bolhari A (2014) A robust optimization model for a supply chain under uncertainty. IMA J Manag Math 25(4):387–402
112. Houck C, Joines J, Kay M, Wilson J (1997) Empirical investigation of the benefits of partial lamarckianism. Evol Comput 5(1):31–60
113. Hsu CH (2007) Uplink MIMO-SDMA optimisation of smart antennas by phase-amplitude perturbations based on memetic algorithms for wireless and mobile communication systems. IET Commun 1(3):520–525
114. Hsu CH, Shyr WJ (2008) Adaptive pattern nulling design of linear array antenna by phase-only perturbations using memetic algorithms. Commun Numer Methods Eng 24(11):1121–1133
115. Hsu CH, Shyr WJ, Chen CH (2006) Adaptive pattern nulling design of linear array antenna by phase-only perturbations using memetic algorithms. In: Pan J, Shi P, Zhao Y (eds) First International Conference on Innovative Computing, Information and Control, vol 3 (ICICIC 2006). IEEE, Beijing, pp 308–311
116. Hsu CH, Chou PH, Shyr WJ, Chung YN (2007) Optimal radiation pattern design of adaptive linear array antenna by phase and amplitude perturbations using memetic algorithms. Int J Innov Comput Inf Control 3(5):1273–1287
117. Hsu CH, Shyr WJ, Ku KH, Chou PH (2008) Optimal radiation pattern design of adaptive linear phased array antenna using memetic algorithms. Int J Innov Comput Inf Control 4(9):2391–2403
118. Hsu CH, Shyr WJ, Kuo KH, Chou PH, Wu MJ (2010) Memetic algorithms for multiple interference cancellations of linear array based on phase-amplitude perturbations. J Optim Theory Appl 144(3):629–642
119. Hu M, Weir JD, Wu T (2014) An augmented multi-objective particle swarm optimizer for building cluster operation decisions. Appl Soft Comput 25:347–359
120. Hu Z, Bao Y, Xiong T (2013) Electricity load forecasting using support vector regression with memetic algorithms. Sci World J 2013:article ID 292575
121. Hu Z, Bao Y, Xiong T (2014) Comprehensive learning particle swarm optimization based memetic algorithm for model selection in short-term load forecasting using support vector regression. Appl Soft Comput 25:15–25
122. Iacca G, Neri F, Caraffini F, Suganthan PN (2014) A differential evolution framework with ensemble of parameters and strategies and pool of local search algorithms. In: Esparcia-Alcázar AI, Mora AM (eds) Applications of Evolutionary Computation. Lecture Notes in Computer Science, vol 8602. Springer, Berlin, pp 615–626
123. Ibaraki T (1997) Combination with dynamic programming. In: Bäck T, Fogel D, Michalewicz Z (eds) Handbook of Evolutionary Computation. Oxford University Press, New York NY, pp D3.4:1–2
124. Ihesiulor OK, Shankar K, Zhang Z, Ray T (2012) Delamination detection using methods of computational intelligence. In: Barsoum N, Faiman D, Vasant P (eds) Sixth Global Conference on Power Control and Optimization. AIP Conference Proceedings, vol 1499. American Institute of Physics, Las Vegas, pp 303–310
125. Ihesiulor OK, Shankar K, Zhang Z, Ray T (2014) Delamination detection with error and noise polluted natural frequencies using computational intelligence concepts. Compos B Eng 56:906–925

126. Ishibuchi H, Yoshida T, Murata T (2003) Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. IEEE Trans Evol Comput 7(2):204–223
127. Jiao L, Gong M, Wang S, Hou B, Zheng Z, Wu Q (2010) Natural and remote sensing image segmentation using memetic computing. IEEE Comput Intell Mag 5(2):78–91
128. Jiménez Laredo JL, Bouvry P, Lombraña González D, Fernández de Vega F, García Arenas M, Merelo Guervós JJ, Fernandes CM (2014) Designing robust volunteer-based evolutionary algorithms. Genet Program Evolvable Mach 15(3):221–244
129. Jolai F, Tavakkoli-Moghaddam R, Rabiee M, Gheisariha E (2014) An enhanced invasive weed optimization for makespan minimization in a flexible flowshop scheduling problem. Scientia Iranica 21(3):1007–1020
130. Jones G, Willett P, Glen R, Leach A, Taylor R (1997) Development and validation of a genetic algorithm for flexible docking. J Mol Biol 267(3):727–748
131. Jones T (1995) Evolutionary algorithms, fitness landscapes and search. PhD thesis, University of New Mexico
132. Julstrom BA (1995) Very greedy crossover in a genetic algorithm for the traveling salesman problem. In: Proceedings of the 1995 ACM Symposium on Applied Computing. ACM, New York, pp 324–328
133. Kalantzis G, Apte A, Radke R, Jackson A (2013) A reduced order memetic algorithm for constraint optimization in radiation therapy treatment planning. In: Takahashi S, Leo R (eds) 14th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking And Parallel/Distributed Computing (SNPD 2013). IEEE, Honolulu, pp 225–230
134. Kernighan B, Lin S (1972) An efficient heuristic procedure for partitioning graphs. Bell Syst J 49:291–307
135. Khan SU, Qureshi IM, Zaman F, Shoaib B, Naveed A, Basit A (2014) Correction of faulty sensors in phased array radars using symmetrical sensor failure technique and cultural algorithm with differential evolution. Sci World J 2014:article ID 852539
136. Kim J, Kim CS, Geem ZW (2014) A memetic approach for improving minimum cost of economic load dispatch problems. Math Probl Eng 2014:article ID 906028
137. King C, Pendlebury DA (2013) Web of knowledge research frontiers 2013: 100 top ranked specialties in the sciences and social sciences. http://sciencewatch.com/sites/sw/files/sw-article/media/research-fronts-2013.pdf
138. Klau GW (2009) A new graph-based method for pairwise global network alignment. BMC Bioinf 10(Suppl 1):S59
139. Kleeman MP, Lamont GB, Cooney A, Nelson TR (2007) A multi-tiered memetic multiobjective evolutionary algorithm for the design of quantum cascade lasers. In: Obayashi S, Deb K, Poloni C, Hiroyasu T, Murata T (eds) Proceeedings of the 4th International Conference on Evolutionary Multi-Criterion Optimization (EMO 2007). Lecture Notes in Computer Science, vol 4403. Springer, Berlin/Matsuhima, pp 186–200
140. Kononova AV, Hughes KJ, Pourkashanian M, Ingham DB (2007) Fitness diversity based adaptive memetic algorithm for solving inverse problems of chemical kinetics. In: IEEE Congress on Evolutionary Computation. IEEE, Singapore, pp 2366–2373
141. Kononova AV, Ingham DB, Pourkashanian M (2008) Simple scheduled memetic algorithm for inverse problems in higher dimensions: application to chemical kinetics. In: IEEE Congress on Evolutionary Computation. IEEE, Hong Kong, pp 3905–3912
142. Krasnogor N (2004) Self generating metaheuristics in bioinformatics: the proteins structure comparison case. Genet Program Evolvable Mach 5(2):181–201
143. Krasnogor N, Gustafson S (2004) A study on the use of "self-generation" in memetic algorithms. Nat Commun 3(1):53–76
144. Krasnogor N, Smith J (2008) Memetic algorithms: the polynomial local search complexity theory perspective. J Math Modell Algorithms 7(1):3–24
145. Krasnogor N, Blackburne B, Burke E, Hirst J (2002) Multimeme Algorithms for Protein Structure Prediction. Lecture Notes in Computer Science, vol 2439. Springer, Berlin, pp 769–778

146. Krishna K, Ramakrishnan K, Thathachar M (1997) Vector quantization using genetic k-means algorithm for image compression. In: 1997 International Conference on Information, Communications and Signal Processing, vol 3. IEEE Press, New York, pp 1585–1587
147. Kumar JV, Kumar DMV (2014) Generation bidding strategy in a pool based electricity market using shuffled frog leaping algorithm. Appl Soft Comput 21:407–414
148. Kumar PK, Sharath S, D'Souza RG, Chandra K (2007) Memetic nsga – a multi-objective genetic algorithm for classification of microarray data. In: 15th International Conference on Advanced Computing And Communications (ADCOM 2007). IEEE, Guwahati, pp 75–80
149. Kumle AN, Fathi SH, Broujeni ST (2014) Harmonic optimization in multi-level inverters by considering adjustable DC sources using memetic algorithm. In: 11th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON 2014). IEEE, Nakhon Ratchasima
150. Lam AYS, Li VOK (2012) Chemical reaction optimization: a tutorial. Memetic Comput 4(1):3–17
151. Li YF, Pedroni N, Zio E (2013) A memetic evolutionary multi-objective optimization method for environmental power unit commitment. IEEE Trans Power Syst 28(3):2660–2669
152. Liaw CF (2000) A hybrid genetic algorithm for the open shop scheduling problem. Eur J Oper Res 124:28–42
153. Liefooghe A, Verel S, Hao JK (2014) A hybrid metaheuristic for multiobjective unconstrained binary quadratic programming. Appl Soft Comput 16:10–19
154. Lim KK, Ong YS, Lim MH, Chen X, Agarwal A (2008) Hybrid ant colony algorithms for path planning in sparse graphs. Soft Comput 12(10):981–994
155. Lin G, Zhu W, Ali MM (2014) A tabu search-based memetic algorithm for hardware/software partitioning. Math Probl Eng 2014:article ID 103059
156. Lin S, Kernighan B (1973) An effective heuristic algorithm for the traveling salesman problem. Oper Res 21:498–516
157. Linda O, Wijayasekara D, Manic M, McQueen M (2014) Optimal placement of phasor measurement units in power grids using memetic algorithms. In: 23rd IEEE International Symposium on Industrial Electronics (ISIE 2014). IEEE, Istanbul, pp 2035–2041
158. Liu B, Wang L, Liu Y, Qian B, Jin YH (2010) An effective hybrid particle swarm optimization for batch scheduling of polypropylene processes. Comput Chem Eng 34(4):518–528
159. Liu S, Chen D, Wang Y (2014) Memetic algorithm for multi-mode resource-constrained project scheduling problems. J Syst Eng Electron 25(4):609–617
160. Liu T, Jiang Z, Geng N (2014) A genetic local search algorithm for the multi-depot heterogeneous fleet capacitated arc routing problem. Flex Serv Manuf J 26(4, SI):540–564
161. Lorber D, Shoichet B (1998) Flexible ligand docking using conformational ensembles. Protein Sci 7(4):938–950
162. Ma W, Huang Y, Li C, Liu J (2012) Image segmentation based on a hybrid immune memetic algorithm. In: Proceedings of the 2012 IEEE Congress on Evolutionary Computation, Brisbane, pp 1–8
163. Maheswaran R, Ponnambalam SG, Aranvidan C (2005) A meta-heuristic approach to single machine scheduling problems. Int J Adv Manuf Technol 25:772–776
164. Marino A, Prügel-Bennett A, Glass CA (1999) Improving graph colouring with linear programming and genetic algorithms. In: Proceedings of EUROGEN 99, Jyväskylä, pp 113–118
165. Matei O, Pop PC, Sas JL, Chira C (2015) An improved immigration memetic algorithm for solving the heterogeneous fixed fleet vehicle routing problem. Neurocomputing 150(A, SI):58–66
166. Mendes A (2011) Identification of breast cancer subtypes using multiple gene expression microarray datasets. In: Wang D, Reynolds M (eds) 24th Australasian Joint Conference on Artificial Intelligence (AI 2011). Lecture Notes in Artificial Intelligence, vol 7106. Springer Berlin/Perth, pp 92–101

167. Mendes A, Cotta C, Garcia V, França P, Moscato P (2005) Gene ordering in microarray data using parallel memetic algorithms. In: Skie T, Yang CS (eds) Proceedings of the 2005 International Conference on Parallel Processing Workshops. IEEE Press, Oslo, pp 604–611
168. Mendes A, Boland N, Guiney P, Riveros C (2013) Switch and tap-changer reconfiguration of distribution networks using evolutionary algorithms. IEEE Trans Power Syst 28(1):85–92
169. Mendoza M, Bonilla S, Noguera C, Cobos C, Leon E (2014) Extractive single-document summarization based on genetic operators and guided local search. Expert Syst Appl 41(9):4158–4169
170. Merz P (2012) Memetic algorithms and fitness landscapes in combinatorial optimization. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin/Heidelberg, pp 95–119
171. Merz P, Zell A (2002) Clustering Gene Expression Profiles with Memetic Algorithms. Lecture Notes in Computer Science, vol 2439. Springer, Berlin, pp 811–820
172. Milojičić DS, Kalogeraki V, Lukose R, Nagaraja K, Pruyne J, Richard B, Rollins S, Xu Z (2002) Peer-to-peer computing. Technical report HPL-2002-57, Hewlett-Packard Labs
173. Molina D, Lozano M, García-Martínez C, Herrera F (2010) Memetic algorithms for continuous optimisation based on local search chains. Evol Comput 18(1):27–63
174. Molina D, Lozano M, Sánchez AM, Herrera F (2011) Memetic algorithms based on local search chains for large scale continuous optimisation problems: MA-SSW-chains. Soft Comput 15(11):2201–2220
175. Moscato P (1989) On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms. Technical report 826, California Institute of Technology, Pasadena
176. Moscato P (1993) An introduction to population approaches for optimization and hierarchical objective functions: the role of tabu search. Ann Oper Res 41(1–4):85–121
177. Moscato P (1999) Memetic algorithms: a short introduction. In: Corne D, Dorigo M, Glover F (eds) New Ideas in Optimization. McGraw-Hill, London, pp 219–234
178. Moscato P (2012) Memetic algorithms: the untold story. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin/Heidelberg, pp 275–309
179. Moscato P, Cotta C (2003) A gentle introduction to memetic algorithms. In: Glover F, Kochenberger G (eds) Handbook of Metaheuristics. Kluwer Academic Publishers, Boston, pp 105–144
180. Moscato P, Norman MG (1992) A memetic approach for the traveling salesman problem implementation of a computational ecology for combinatorial optimization on message-passing systems. In: Valero M, Onate E, Jane M, Larriba JL, Suarez B (eds) Parallel Computing and Transputer Applications. IOS Press, Amsterdam, pp 177–186
181. Moscato P, Tinetti F (1992) Blending heuristics with a population-based approach: a memetic algorithm for the traveling salesman problem. Report 92–12, Universidad Nacional de La Plata
182. Moscato P, Mendes A, Berretta R (2007) Benchmarking a memetic algorithm for ordering microarray data. Biosystems 88(1–2):56–75
183. Mozaffari A, Chehresaz M, Azad NL (2013) Component sizing of a plug-in hybrid electric vehicle powertrain, part a: coupling bio-inspired techniques to meshless variable-fidelity surrogate models. Int J Bio-Inspired Comput 5(6):350–383
184. Mühlenbein H (1989) Parallel genetic algorithms, population genetics and combinatorial optimization. In: Schaffer (ed) 3rd International Conference on Genetic Algorithms. Morgan Kaufmann, San Mateo, pp 416–421
185. Nagata Y, Kobayashi S (1997) Edge assembly crossover: a high-power genetic algorithm for the traveling salesman problem. In: Bäck T (ed) Seventh International Conference on Genetic Algorithms. Morgan Kaufmann, San Mateo, pp 450–457

186. Nair SSK, Reddy NVS, Hareesha KS (2011) Exploiting heterogeneous features to improve in silico prediction of peptide status – amyloidogenic or non-amyloidogenic. BMC Bioinf 12(13):S21
187. Nair SSK, Reddy NVS, Hareesha KS (2012) Machine learning study of classifiers trained with biophysiochemical properties of amino acids to predict fibril forming peptide motifs. Protein Pept Lett 19(9):917–923
188. Neri F (2012) Diversity management in memetic algorithms. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin/Heidelberg, pp 153–165
189. Neri F, Cotta C (2012) Memetic algorithms and memetic computing optimization: a literature review. Swarm Evol Comput 2:1–14
190. Neri F, Mininno E (2010) Memetic compact differential evolution for Cartesian robot control. IEEE Comput Intell Mag 5(2):54–65
191. Neri F, Toivanen J, Cascella GL, Ong YS (2007) An adaptive multimeme algorithm for designing HIV multidrug therapies. IEEE-ACM Trans Comput Biol Bioinform 4(2):264–278
192. Neri F, Toivanen J, Makinen RAE (2007) An adaptive evolutionary algorithm with intelligent mutation local searchers for designing multidrug therapies for HIV. Appl Intell 27(3):219–235
193. Neri F, Cotta C, Moscato P (eds) (2012) Handbook of Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin/Heidelberg
194. Nguyen QC, Ong YS, Kuo JL (2009) A hierarchical approach to study the thermal behavior of protonated water clusters H+(H2O)(n). J Chem Theory Comput 5(10):2629–2639
195. Nikzad M, Farahani SSS, Tabar MB, Tourang H, Yousefpour B (2012) A new optimization method for pss design in New-England power system. Life Sci J Acta Zhengzhou Univ Overseas Ed 9(4):5478–5483
196. Nogueras R, Cotta C (2015) Studying fault-tolerance in Island-based evolutionary and multimemetic algorithms. J Grid Comput. https://doi.org/10.1007/s10723-014-9315-6
197. Noman N, Iba H (2007) Inferring gene regulatory networks using differential evolution with local search heuristics. IEEE-ACM Trans Comput Biol Bioinform 4(4):634–647
198. Norman M, Moscato P (1989) A competitive and cooperative approach to complex combinatorial search. Technical report Caltech Concurrent Computation Program, Report. 790, California Institute of Technology, Pasadena. Expanded version published at the 20th Informatics and Operations Research Meeting, Buenos Aires (20th JAIIO), Aug 1991, pp 3.15–3.29
199. Montes de Oca MA, Cotta C, Neri F (2012) Local search. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin/Heidelberg, pp 29–41
200. Oliveri G, Lizzi L, Pastorino M, Massa A (2012) A nested multi-scaling inexact-Newton iterative approach for microwave imaging. IEEE Trans Antennas Propag 60(2, 2):971–983
201. Olson BS, Shehu A (2012) Evolutionary-inspired probabilistic search for enhancing sampling of local minima in the protein energy surface. Proteome Sci 10(1):S5
202. Ong YS, Keane A (2004) Meta-Lamarckian learning in memetic algorithms. IEEE Trans Evol Comput 8(2):99–110
203. Ong YS, Lim MH, Zhu N, Wong KW (2006) Classification of adaptive memetic algorithms: a comparative study. IEEE Trans Syst Man Cybern B: Cybern 36(1):141–152
204. Ong YS, Lim MH, Chen X (2010) Memetic computation-past, present and future. IEEE Comput Intell Mag 5(2):24–31
205. Ortega JC, Gimenez D, Alvarez-Melcon A, Quesada FD (2013) Hybrid metaheuristics for the design of coupled resonator filters. Appl Artif Intell 27(5):323–350
206. Pal S, Basak A, Das S, Abraham A (2009) Linear antenna array synthesis with invasive weed optimization algorithm. In: Abraham A, Muda A, Herman N, Shamsuddin S, Huoy C (eds) International Conference of Soft Computing and Pattern Recognition. IEEE, Malacca, pp 161–166
207. Pal S, Basak A, Das S (2011) Linear antenna array synthesis with modified invasive weed optimisation algorithm. Int J Bio-Inspired Comput 3(4):238–251

208. Palacios P, Pelta D, Blanco A (2006) Obtaining biclusters in microarrays with population-based heuristics. In: Rothlauf F (ed) Proceedings of Applications of Evolutionary Computing. Lecture Notes in Computer Science, vol 3907. Springer, Berlin/Budapest, pp 115–126
209. Pan QK, Dong Y (2014) An improved migrating birds optimisation for a hybrid flowshop scheduling with total flowtime minimisation. Inform Sci 277:643–655
210. Perales-Gravan C, Lahoz-Beltra R (2008) An AM radio receiver designed with a genetic algorithm based on a bacterial conjugation genetic operator. IEEE Trans Evol Comput 12(2):129–142
211. Poikolainen I, Neri F (2013) Differential evolution with concurrent fitness based local search. In: Proceedings of the 2013 IEEE Congress on Evolutionary Computation. IEEE Press, Cancun, pp 384–391
212. Prodhon C, Prins C (2014) A survey of recent research on location-routing problems. Eur J Oper Res 238(1):1–17
213. Qin H, Zhang Z, Qi Z, Lim A (2014) The freight consolidation and containerization problem. Eur J Oper Res 234(1):37–48
214. Quevedo-Teruel O, Rajo-Iglesias E, Oropesa-Garcia A (2007) Hybrid algorithms for electromagnetic problems and the no-free-lunch framework. IEEE Trans Antennas Propag 55(3, 1):742–749
215. Quintero A, Pierre S (2003) Evolutionary approach to optimize the assignment of cells to switches in personal communication networks. Comput Commun 26(9):927–938
216. Quintero A, Pierre S (2003) Sequential and multi-population memetic algorithms for assigning cells to switches in mobile networks. Comput Netw 43(3):247–261
217. Radcliffe N (1994) The algebra of genetic algorithms. Ann Math Artif Intell 10:339–384
218. Radcliffe NJ, Surry PD (1994) Formal memetic algorithms. In: Fogarty TC (ed) AISB Workshop on Evolutionary Computing. Lecture Notes in Computer Science, vol 865. Springer, Berlin/Heidelberg, pp 1–16
219. Rager M, Gahm C, Denz F (2015) Energy-oriented scheduling based on evolutionary algorithms. Comput Oper Res 54:218–231
220. Rahiminejad A, Alimardani A, Vahidi B, Hosseinian SH (2014) Shuffled frog leaping algorithm optimization for AC-DC optimal power flow dispatch. Turk J Electr Eng Comput Sci 22(4):874–892
221. Raja MAZ, Ahmad SuI, Samar R (2014) Solution of the 2-dimensional bratu problem using neural network, swarm intelligence and sequential quadratic programming. Neural Comput Appl 25(7–8):1723–1739
222. Rao ARM, Lakshmi K (2012) Optimal design of stiffened laminate composite cylinder using a hybrid sfl algorithm. J Compos Mater 46(24):3031–3055
223. Rao BS, Vaisakh K (2013) New variants/hybrid methods of memetic algorithm for solving optimal power flow problem with load uncertainty. Int J Hybrid Intell Syst (IJHIS) 10(3):117–128
224. Richter H, Engelbrecht A (2014) Recent Advances in the Theory and Application of Fitness Landscapes, Emergence, Complexity and Computation, vol 6. Springer, Berlin/Heidelberg
225. Rodríguez Rueda D, Cotta C, Fernández Leiva AJ (2011) A memetic algorithm for designing balanced incomplete blocks. Int J Comb Optim Probl Inform 2(1):14–22
226. Rothlauf F, Goldberg DE (2002) Representations for Genetic and Evolutionary Algorithms. Physica-Verlag, Heidelberg
227. Salhi A, Rodriguez JAV (2014) Tailoring hyper-heuristics to specific instances of a scheduling problem using affinity and competence functions. Memetic Comput 6(2):77–84
228. Santamaría J, Cordón O, Damas S, García-Torres JM, Quirin A (2009) Performance evaluation of memetic approaches in 3D reconstruction of forensic object. Soft Comput 13(8–9):883–904
229. Sarmenta LF (1998) Bayanihan: web-based volunteer computing using java. In: Masunaga Y, Katayama T, Tsukamoto M (eds) Worldwide Computing and Its Applications – WWCA'98. Lecture Notes in Computer Science, vol 1368. Springer, Berlin/Heidelberg, pp 444–461

230. Schaefer G (2014) Aco classification of thermogram symmetry features for breast cancer diagnosis. Memetic Comput (3):207–212
231. Sevaux M, Dauzère-Pérès S (2003) Genetic algorithms to minimize the weighted number of late jobs on a single machine. Eur J Oper Res 151:296–306
232. Silva R, Berenguel M, Perez M, Fernandez-Garcia A (2014) Thermo-economic design optimization of parabolic trough solar plants for industrial process heat applications with memetic algorithms. Appl Energy 113(SI):603–614
233. Smith JE (2007) Coevolving memetic algorithms: a review and progress report. IEEE Trans Syst Man Cybern B Cybern 37(1):6–17
234. Smith JE (2010) Meme fitness and memepool sizes in coevolutionary memetic algorithms. In: 2010 IEEE Congress on Evolutionary Computation. IEEE Press, Barcelona, pp 1–8
235. Smith JE (2012) Self-Adaptative and Coevolving Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin/Heidelberg, pp 167–188
236. Sörensen K, Sevaux M (2006) MA | PM: memetic algorithms with population management. Comput OR 33:1214–1225
237. Speer N, Merz P, Spieth C, Zell A (2003) Clustering gene expression data with memetic algorithms based on minimum spanning trees. In: IEEE Congress on Evolutionary Computation (CEC 2003), Canberra, pp 1848–1855
238. Speer N, Spieth C, Zell A (2004) A memetic clustering algorithm for the functional partition of genes based on the gene ontology. In: IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, La Jolla, pp 252–259
239. Speer N, Spieth C, Zell A (2004) A memetic co-clustering algorithm for gene expression profiles and biological annotation. In: Congress on Evolutionary Computation (CEC 2004). IEEE, Portland, pp 1631–1638
240. Spieth C, Streichert F, Speer N, Zell A (2004) A memetic inference method for gene regulatory networks based on s-systems. In: 2004 IEEE Congress on Evolutionary Computation (CEC 2004), Portland, pp 152–157
241. Spieth C, Streichert F, Supper J, Speer N, Zell A (2005) Feedback memetic algorithms for modeling gene regulatory networks. In: IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, La Jolla, pp 61–67
242. Steimel J, Engell S (2014) Conceptual design and optimisation of chemical processes under uncertainty by two-stage programming. In: Eden M, Siirola J, Towler G (eds) 8th International Conference on Foundations of Computer-Aided Process Design, vol 34. Elsevier Science BV, Cle Elum, pp 435–440
243. Streichert F, Tanaka-Yamawaki M (2006) The effect of local search on the constrained portfolio selection problem. In: IEEE Congress on Evolutionary Computation, Vancouver, pp 2353–2359
244. Sudholt D (2009) The impact of parametrization in memetic evolutionary algorithms. Theor Comput Sci 410(26):2511–2528
245. Sudholt D (2012) Parametrization and balancing local and global search. In: Neri F, Cotta C, Moscato P (eds) Handbook of Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin/Heidelberg, pp 55–72
246. Sun X, Wang Z, Zhang D (2008) A watermarking algorithm based on MA and DWT. In: IEEE International Symposium on Electronic Commerce and Security, Guangzhou, pp 916–919
247. Tanese R (1989) Distributed genetic algorithms. In: 3rd International Conference on Genetic Algorithms. Morgan Kaufmann, San Francisco, pp 434–439
248. Ting CK, Liao CC (2010) A memetic algorithm for extending wireless sensor network lifetime. Inform Sci 180(24):4818–4833
249. Tirronen V, Neri F, Kärkkäinen T, Majava K, Rossi T (2008) An enhanced memetic differential evolution in filter design for defect detection in paper production. Evol Comput 16(4):529–555
250. Tomassini M (2005) Spatially Structured Evolutionary Algorithms. Natural Computing Series. Springer, Berlin/New York

251. Tse S, Liang Y, Leung K, Lee K, Mok S (2005) Multi-drug cancer chemotherapy scheduling by a new memetic optimization algorithm. In: IEEE Congress on Evolutionary Computation (CEC 2005), Edinburgh, pp 699–706
252. Tse SM, Liang Y, Leung KS, Lee KH, Mok TSK (2007) A memetic algorithm for multiple-drug cancer chemotherapy schedule optimization. IEEE Trans Syst Man Cybern B Cybern 37(1):84–91
253. Urselmann M, Engell S (2010) Optimization-based design of reactive distillation columns using a memetic algorithm. In: Pierucci S, Ferraris B (eds) 20th European Symposium on Computer Aided Process Engineering. Elsevier Science BV, Ischia, vol 28, pp 1243–1248
254. Urselmann M, Engell S (2015) Design of memetic algorithms for the efficient optimization of chemical process synthesis problems with structural restrictions. Comput Chem Eng 72:87–108
255. Urselmann M, Sand G, Engell S (2009) A memetic algorithm for global optimization in chemical process synthesis. In: IEEE Congress on Evolutionary Computation (CEC 2009), Trondheim, pp 1721–1728
256. Urselmann M, Barkmann S, Sand G, Engell S (2011) A memetic algorithm for global optimization in chemical process synthesis problems. IEEE Trans Evol Comput 15(5, SI):659–683
257. Urselmann M, Barkmann S, Sand G, Engell S (2011) Optimization-based design of reactive distillation columns using a memetic algorithm. Comput Chem Eng 35(5):787–805
258. Vesselinov VV, Harp DR (2012) Adaptive hybrid optimization strategy for calibration and parameter estimation of physical process models. Comput Geosci 49:10–20
259. Vidal T, Crainic TG, Gendreau M, Prins C (2014) A unified solution framework for multi-attribute vehicle routing problems. Eur J Oper Res 234(3):658–673
260. Wang G, Wang J, Chen H (2014) Application of operator libraries on laminate strength optimization of composites. In: Yang G (ed) International Conference on Materials Science, Machinery and Energy Engineering (MSMEE 2013). Advanced Materials Research, vol 853. Trans Tech Publications Ltd, Hong Kong, pp 686–692
261. Wang L, Liu J (2013) A scale-free based memetic algorithm for resource-constrained project scheduling problems. In: Yin H, Tang K, Gao Y, Klawonn F, Lee M, Li B, Weise T, Yao X (eds) 14th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2013). Lecture Notes in Computer Science, vol 8206. Springer, Hefei, pp 202–209
262. Wang L, Zheng DZ (2002) A modified genetic algorithm for job-shop scheduling. Int J Adv Manuf Technol 20:72–76
263. Wang Y, Hao JK, Glover F, Lu Z (2014) A tabu search based memetic algorithm for the maximum diversity problem. Eng Appl Artif Intell 27:103–114
264. Wanner EF, Guimaraes FG, Takahashi RHC, Lowther DA, Ramirez JA (2008) Multiobjective memetic algorithms with quadratic approximation-based local search for expensive optimization in electromagnetics. IEEE Trans Mag 44(6):1126–1129
265. Whitley LD (1991) Fundamental principles of deception in genetic search. In: Rawlins G (ed) Foundations of Genetic Algorithms I. Morgan Kaufmann, San Mateo, pp 221–241
266. Widl M, Musliu N (2014) The break scheduling problem: complexity results and practical algorithms. Memetic Comput 6(2):97–112
267. Wolpert D, Macready W (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
268. Xu J, Yin Y, Cheng TCE, Wu CC, Gu S (2014) An improved memetic algorithm based on a dynamic neighbourhood for the permutation flowshop scheduling problem. Int J Prod Res 52(4):1188–1199
269. Yang CH, Cheng YH, Chang HW, Chuang LY (2009) Primer design with specific PCr product size using memetic algorithm. In: IEEE Conference on Soft Computing in Industrial Applications (SMCIA 2008), Muroran, pp 332–337
270. Yang CH, Cheng YH, Chuang LY, Chang HW (2009) Specific PCr product primer design using memetic algorithm. Biotechnol Prog 25(3):745–753

271. Yang S, Wang S, Liu Z, Wang M, Jiao L (2014) Improved bandelet with heuristic evolutionary optimization for image compression. Eng Appl Artif Intell 31(SI):27–34
272. Yang SH, Kiang JF (2014) Optimization of asymmetrical difference pattern with memetic algorithm. IEEE Trans Antennas Propag 62(4, 2):2297–2302
273. Zaman F, Qureshi IM, Munir F, Khan ZU (2014) Four-dimensional parameter estimation of plane waves using swarming intelligence. Chin Phys B 23(7):078402
274. Zhao F, Tang J, Wang J, Jonrinaldi J (2014) An improved particle swarm optimization with decline disturbance index (DDPSO) for multi-objective job-shop scheduling problem. Comput Oper Res 45:38–50
275. Zhou J, Ji Z, Zhu Z, He S (2014) Compression of next-generation sequencing quality scores using memetic algorithm. BMC Bioinform 15(15):S10
276. Zhou M, Liu J (2014) A memetic algorithm for enhancing the robustness of scale-free networks against malicious attacks. Physica A-Stat Mech Appl 410:131–143
277. Zhu GY, Zhang WB (2014) An improved shuffled frog-leaping algorithm to optimize component pick-and-place sequencing optimization problem. Expert Syst Appl 41(15):6818–6829
278. Zhu Z, Ong YS (2007) Memetic algorithms for feature selection on microarray data. In: Liu D, Fei S, Hou Z, Zhang H, Sun C (eds) 4th International Conference on Advances in Neural Networks (ISNN 2007). Lecture Notes in Computer Science, vol 4491. Springer, Berlin/Nanjing, pp 1327–1335
279. Zhu Z, Ong YS, Zurada JM (2010) Identification of full and partial class relevant genes. IEEE-ACM Trans Comput Biol Bioinform 7(2):263–277
280. Zhu Z, Zhou J, Ji Z, Shi YH (2011) DNA sequence compression using adaptive particle swarm optimization-based memetic algorithm. IEEE Trans Evol Comput 15(5, SI):643–658
281. Zibakhsh A, Abadeh MS (2013) Gene selection for cancer tumor detection using a novel memetic algorithm with a multi-view fitness function. Eng Appl Artif Intell 26(4):1274–1281