Information Sciences 262 (2014) 15–31

Contents lists available at ScienceDirect

Information Sciences

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

Region based memetic algorithm for real-parameter optimisation

Benjamin Lacroix [a,*], Daniel Molina [b], Francisco Herrera [a]

[a] Department of Computer Science and Artificial Intelligence, Universidad de Granada, C/Periodista Daniel Saucedo Aaranda s/n, Granada, Spain
[b] Department of Computer Science and Engineering, Universidad de Cádiz, Spain

ARTICLE INFO

Article history:
Received 11 April 2012
Received in revised form 9 October 2013
Accepted 26 November 2013
Available online 4 December 2013

Keywords:
Real-parameter optimisation
Memetic algorithm
Niching strategy

ABSTRACT

Memetic algorithms with an appropriate trade-off between the exploration and exploitation can obtain very good results in continuous optimisation. That implies the evolutionary algorithm should be focused in exploring the search space while the local search method exploits the achieved solutions. To tackle this issue, we propose to maintain a higher diversity in the evolutionary algorithm's population by including a niching strategy in the memetic algorithm framework. In this work, we design a novel niching strategy where the niches divide the search space into hypercubes of equal size called regions forbidding the presence of two solutions in each region. The objective is to avoid the competition between the local search and the evolutionary algorithm. We tested this niching strategy in a memetic algorithm with local search chaining and obtained significant improvements. The resulting model also appeared to be very competitive with state-of-the-art algorithms.

1. Introduction

One of the main issues when designing an evolutionary algorithm (EA) [3] for real-coded parameter optimisation problems is to offer a good exploration of the search space and, at the same time, to exploit the most promising regions to obtain high quality solutions. Memetic algorithms (MA) were proposed [34] to manage these competing objectives. They are a hybridisation between EA and local search (LS) algorithms, mixing in one model the exploration power of EA and the exploitative power of the LS. MAs are characterised by the combination of an exploration algorithm and a local improvement algorithm.

MAs with an appropriate trade-off between the exploration and exploitation can obtain accurate solutions, improving the search [14,21]. Therefore, the key issue when designing a MA is to organise both efforts in the most cooperative way.

Niching strategies have been used in EA to either identify various optima in a fitness landscape or to maintain a strong diversity in the EA's population [13]. In our study, we consider that using niching strategy to maintain a higher diversity in the population leads to a better separation of the effort between the EA and the LS.

Thus, we design a niching strategy to limit the competition between the EA and the LS in a MA. The purpose of this method is to let the EA focus on the exploration task by limiting its exploitation power, this task being more efficiently performed by the LS method. Contrarily to most niching strategies where the niches are defined around the solutions of the population, the niches are predefined as divisions of the search space. The search space is divided into equal hypercubes each of which represent one exclusion region, not allowing more than one solution in each one. Also, the LS method is initialised to explore inside these regions. This way, there is no competition between the EA and the LS method. In order to obtain a more robust

* Corresponding author. Tel.: +34 622154665.
E-mail address: benjamin@decsai.ugr.es (B. Lacroix).

0020-0255/$ - see front matter
http://dx.doi.org/10.1016/j.ins.2013.11.032

strategy, we also propose a version with a dynamic niche size. It consists in decreasing the niche size along the search to have a great diversity in the early stage of the search and reduce it along the process.

To assess the efficiency of this strategy, we implement it in the MA-LSCh-CMA [32]. MA-LSCh-CMA is a successful MA which originality lies in its ability to apply various times the LS on the same solution. Although this algorithm obtains good results, it lacks a diversity control mechanism and does not limits the competition between the EA and the LS efforts. The association of the MA-LSCh-CMA algorithm with the niching strategy proposed gives the algorithm called Region-based MA-LSCh-CMA (RMA-LSCh-CMA).

Various studies are performed to demonstrate the influence of the niching strategy on the diversity of the EA's population and the improvements brought to the original model.

The proposed algorithm is then automatically configured using IRACE [28] for comparisons with various state-of-the-art algorithms.

This paper is structured as follows. Section 2 is dedicated to give an overview of previous work done in the field of MAs and to explain the MA-LSCh-CMA we used as case study here. In Section 3, we describe in detail the new proposal, remarking the differences with the previous model. In Section 4, the experimental framework is designed. In Section 5, we show the results and analysis of the different studies and comparisons carried out. Finally, in Section 6, we present the conclusions and future works.

2. Background

In this section we first give a small introduction on MAs. In Section 2.1, we give a broad overview of the originial concepts of memetic algorithms, their use and why they became popular. In Section 2.2, we focus in details on the MA-LSCh-CMA we used to test the efficiency of our niching strategy.

2.1. Memetic algorithms

Memetic Algorithms are a specification of Memetic Computing (MC) [41,11,23]. MC is the paradigm that uses the notion of memes. In general terms, memes are problem solvers. In MC, memes are included in a global framework allowing them to cooperate and/or compete in the problem solving.

MAs can be considered as a sub class of MC. They are the union of population-based global search and local improvements which are inspired by Darwinian principles of natural evolution and Dawkins' notion of a meme [15], defined as a unit of cultural evolution that is capable of local refinements.

In general terms, MAs are composed by a group of search algorithms cooperating and/or competing for the optimisation purpose. While this is usually accomplished by applying LS strategies to members of an EA's population, the MA paradigm also includes other kind of strategies such as the combinations of EAs with problem dependent heuristics, approximation algorithms truncated exact methods, and specialised recombination operators [26,37].

MAs have been a hot topic in the field of optimisation. Many different instantiations of MAs have been reported across a wide variety of application domains that range from scheduling [12] and floor-planning problems [48], to extending wireless sensor network lifetime [49], aerodynamic design [43], vehicle routing [20,31], engineering control problems [38] and drug design [39], to name but a few. This large body of evidence has revealed that MAs not only converge to high-quality solutions, but also search vast, and sometimes noisy, solution spaces more efficiently than their conventional counterparts. Thus, MAs are the preferred methodology for many real-world applications, and nowadays receives more attention [40,37].

MA presents many advantages which made this approach popular:

- The first main advantage of using MAs is based on the "*divide to conquer*" idea. MA separates the exploration effort from the exploitation effort in two components, the former being performed by ea EA, the latter by a LS algorithm. Not only this eases the development of each component, while classical EAs try to combine both aspects of an optimisation search in the same framework it also offers a better control on their functioning.
- By allowing an easy inclusion of problem knowledge, MAs are also considered as a guideline for addressing specific problems [37,9,35]
- MAs have arisen as a promising approach for improving the convergence speed to the Pareto front of EAs for multiobjective optimisation problems, which actually concentrate increasing research efforts [24,27].

2.2. The MA-LSCh-CMA algorithm

This section describes the general scheme of the memetic algoritm with local search chaingin and CMA-ES (MA-LSCh-CMA) and its main components. More details can be seen in [32].

2.2.1. General scheme

MA-LSCh-CMA was designed with the idea that the LS should be applied with higher intensity on the most promising regions. By promising regions, we consider the areas/regions where the solutions are maintained the most time in the population for their good fitness.

The MA-LSCh-CMA is a steady state MA which alternatively applies a Steady-State Genetic Algorithm (SSGA) as EA [45], and a CMA-ES [22] as LS method with an I_{str}. This hybridisation model allows the same solution improve several times, creating *LS chain*. Also, it uses a mechanism to store with the solution the final state of the LS parameters after each LS application. This way, the final state of a LS application over a solution will be used as the initial point of a subsequent LS application over the same solution, *continuing* the LS. The general scheme can be seen in Algorithm 1.

Algorithm 1. Pseudocode of MA-LSCh-CMA

1: Generate the initial population
2: **while** not termination-condition **do**
3: Perform the SSGA with n_{frec} evaluations
4: Build the set S_{LS} of individuals which can be refined by LS
5: Pick the best individual c_{LS} in S_{LS}
6: **if** c_{LS} belongs to an existing LS chain **then**
7: Initialise the LS operator with the LS state stored with c_{LS}
8: **else**
9: Initialise the LS operator with the default LS parameters
10: **end if**
11: Apply the LS algorithm to c_{LS} with I_{str} evaluations, giving c^r_{LS}
12: Replace c_{LS} by c^r_{LS}
13: Store the final LS state with c^r_{LS}
14: **end while**

To select the individual c_{LS} to which the LS will be applied, the following process is used (steps 4 and 5 in Algorithm 1):

1. The set S_{LS} is build with the individuals of the population that:
   (a) have never been improved by the LS.
   (b) have been improved by the LS but with an improvement (in fitness) superior to δ^{min}_{LS}.
2. If $|S_{LS}| \neq 0$, the LS is applied on the best individual in S_{LS}. If S_{LS} is empty, the whole population is reinitialised except for the best individual which is maintained in the population.

With this mechanism, if SSGA obtains a next best solution, it should be improved by the LS in the following application of the LS method.

2.2.2. The EA

The SSGA applied was specifically designed to promote high population diversity levels by means of the combination of the $BLX-\alpha$ crossover operator [19] with a high value for its associated parameter ($\alpha = 0.5$) and the *negative assortative mating* strategy (NAM) [1]. Diversity is favoured as well by means of the BGA mutation operator [36]. The replacement strategy used is *Replacement Worst, RW*. The combination *NAM-RW* produces a high selective pressure. The SSGA is described in Algorithm 2.

Algorithm 2. Pseudo-code for the SSGA

Randomly generate the population
while not termination-condition **do**
Select two parents in the population using the NAM strategy
Create an offspring c_n using $BLX-\alpha$ crossover and BGA mutation
Replace the worst individual c_{worst} in the population if $f(c_{worst}) > f(c_n)$
end while

2.2.3. The LS

The continuous LS algorithm is CMA-ES [22]. This algorithm is the *state-of-the-art* in continuous optimisation. Thanks to the adaptability of its parameters, its convergence is very fast and obtains very good results. CMA-ES is an algorithm that uses a distribution function to obtain new solutions, and adapt the distribution around the best created solutions.

The only required parameters are the initial average of the distribution $\vec{m}$ and the initial standard deviation σ. MA-LSCh-CMA sets the individual to optimise c_{LS} as $\vec{m}$, and as the initial σ value the half of the distance of c_{LS} to its nearest neighbour in the EA's population. In our proposal, this initialisation strategy is modified to focus the LS in the regions.

3. Region based memetic algorithms

This section presents the basic concepts of the novel rigid niching strategy and explains how we included it in the MA-LSCh-CMA.

Niching strategy consists in creating an area around the solutions of an EA's population where no other solution can be present. The main purpose of this tool is to maintain the diversity of the population at a higher level. Maintaining the diversity in a population prevents a fast convergence of the population and allows a better exploration of the search space. This notion is particularly interesting in MA as an EA's first task is to explore, the exploitation of the solutions being done by the LS method. In other words, throw this strategy, we offer a clearer separation between the exploration effort done by the EA and the exploitation task of the LS method.

In Section 3.1, we describe the proposed niching strategy. Including such niching strategy implied two major modifications in the MA-LSCh-CMA, the redefinition of the EA, explained in Section 3.2 and the initial parameters of the LS explained in Section 3.3. Finally, we explain the scheme of the dynamic model in Section 3.4. We named the resulting algorithm RMA-LSCh-CMA.

3.1. Basic concepts

Contrarily to most niching strategies where the niches are defined by the area surrounding solutions of the population, we propose here a strategy in which the niches are predefined as divisions of the search space, divided into hypercubes of equal size called here regions. This definition of a niche is illustrated in Fig. 1. Each dimension is divided into *ND* divisions creating a grid of equal hypercubes, that represent exclusive regions (niches) which can contain only one solution.

3.2. The SSGA in a region-based MA

One of the key issues in niching strategies is to decide what to do with a solution generated in the exclusion area of an other solution. The modifications to the SSGA are described in Algorithm 3. It consists in not allowing the generation of a solution by the SSGA in a region that is already occupied by an optimised solution of the population. By optimised, we refer to the fact that it was previously applied the LS over this solution, and the last LS applied has not brought enough improvements (upper than δ_{LS}^{min}). Then, if a solution is optimised, we consider its neighbourhood (and by consequence the region it lies in) has sufficiently been explored. On the other hand, if the solution is not optimised, the EA can replace it with a solution with a better fitness in that region. That way, we avoid unnecessary LS evaluations within the region to get a higher quality solutions. This way, we ensure that the population does not hold two solutions in the same region.

Algorithm 3. Pseudo-code for the region-based SSGA

1: Randomly generate the population
2: **while** not termination-condition **do**
3: Select two parents in the population
4: Create an offspring c_n using crossover and mutation
5: **if** c_n falls in a region containing an individual c_o **then**
6: **if** c_o is considered optimised **then**
7: Mutate c_n using the BGA mutation and go back to 5
8: **end if**
9: **end if**
10: **if** c_n falls in a region containing an individual c_o **then**
11: Replace c_o by c_n if $f(c_o) > f(c_n)$
12: **else**
13: Replace the worst individual c_{worst} in the population if $f(c_{worst}) > f(c_n)$
14: **end if**
15: **end while**

3.3. The LS in a region-based MA

In order to put the emphasis on dedicating the exploration task to the EA and the exploitation one to the LS, we have also modified the strategy for initialising the parameters of the LS. In the MA-LSCh-CMA, the initial step of the CMA-ES is set between the area limited by its neighbouring solutions. Here the CMA-ES initial step is set according to the size of the region. We want to ensure that the close surrounding of a solution are properly explored by the LS as this task will not be done by the EA. The initial standard deviation is set to half the size of the region. Apart from this modification, in order to allow a

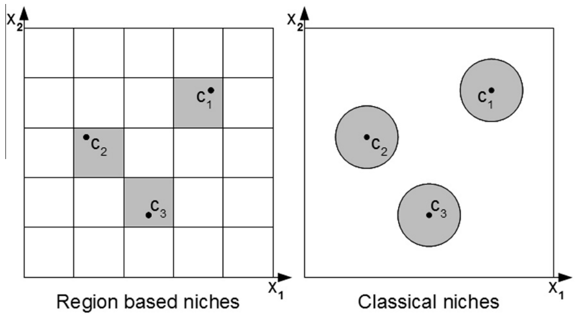

Fig. 1. Different niching strategy.

proper refinement of the solution, the LS is not influenced by the divisions of the search space. However, if at the end of the LS application, the new solution is in a region occupied, the best solution is kept and the other one is replaced by a randomly generated solution.

3.4. A dynamic number of divisions

One of the main issues when implementing a niching strategy is to define the size of the niche. It is also the case in this model and it represents the most critical parameter. Here, the size of the region is directly dependent on the number of divisions per dimensions *ND*.

A high number of divisions leads to smaller niches and thus, a poor influence on the search. On the other hand a small number of divisions creates big niches. The diversity will be high but the chances that the local search fails to reach the best solution in its surroundings are higher.

These reasons shows that the niche size can limit the effectiveness of the search. This motivates the use of a dynamic niche size in order to achieve a better robustness of the algorithm. To do so, the number of divisions is increased along the search. With bigger regions at the beginning of the search, a greater diversity is maintained to ensure a strong exploration of the search space. The number of division is then increased in order to allow a better convergence in later stages of the process.

We have decided to use a linear increase in the number of division. At each update, $ND_i = m_u \cdot ND_{i-1}$. If the total number of updates is u, an update occurs every $max\_eval/(u+1)$ where max_eval is the maximum number of fitness evaluation allowed. With this strategy, two parameters appear, ND_0, the initial number of divisions and u, the number of updates.

4. Experimental framework

We have carried out different experiments to assess the performances of the region-based niching strategy.[1] In this section, we describe the test suites used. For its low and median dimensions, the benchmark proposed in the Special Session on Real Parameter Optimisation organised in the 2005 IEEE Congress on Evolutionary Computation (CEC'2005) [47] is the basis of the study of the model and is used for all the experiments. On the other hand, another benchmark, the Soft Computing Special Issue on Large Scale Continuous Optimisation benchmark (SOCO'2011) [29] is used for offering results in a higher dimension.

Sections 4.1 and 4.2 respectively describe the CEC'2005 and SOCO'2011 benchmark. Section 4.3 lists the statistical tests used for the comparisons.

4.1. The 2005 IEEE Congress on Evolutionary Computation benchmark

For the experimental sections, we have used the benchmark proposed in the CEC'2005. The complete description of the functions can be seen in [47]. Table 1 lists those functions. The firstfive are unimodal functions (F1–F5), followed by seven basic multimodal (F6–F12), two expanded (F13–F14) and 11 hybrid composition functions (F15–F25). Those last ones are compositions of the twelve first. Note that every functions have been shifted to ensure that the global optimum is not in the center of search space. In F7 and F25, the optima are out of the ranges of initialisation. These functions have been implemented in dimension $D = 10, 30, 50$.

In order to be able to compare our results with other algorithms involved in the competition, we followed the requirements described in [47]:

- Each algorithm is run 25 times for each test function, and the average of error of the best individual of the population is computed. The *function error* value for a solution x is defined as $(f(x) - f(x^*))$, where x^* is the global optimum of the function.

[1] The source code used for these experiment can be found at http://sci2s.ugr.es/EAMHCO/srcnew/RMA-LSCh-CMA.zip

Table 1
Test functions of the CEC'2005 benchmark.

F1	Sphere function
F2	Schwefel's problem 1.2
F3	Rotated High Conditioned Elliptic Function
F4	Schwefel's Problem 1.2 with Noise in Fitness
F5	Unimodal function
F6	Rosenbrock's function
F7	Griewank's function
F8	Ackley's function
F9	Rastrigin's function
F10	Rotated Rastrigin's function
F11	Rotated Weiestrass' function
F12	Schwefels Problem 2.13
F13–F14	Expanded functions
F15–F25	Hybrid composition function

- The study has been made with dimensions $D = 10, D = 30$, and $D = 50$.
- The maximum number of fitness evaluations for each run is $10,000 \cdot D$, where D is the dimension of the problem.
- Each run stops either when the error obtained is less than 10^{-8}, or when the maximal number of evaluations is achieved.

4.2. The soft computing special issue on large scale continuous optimisation problems

To assess the scalability of our model against the state-of-the-art, we used the SOCO'2011 benchmark [29]. Table 2 lists those function. It is composed of 19 shifted functions, 11 basic functions (F1–F11) and 8 hybrid composition functions (F12–F19) which are non-separable functions built by combining two of the 11 first functions.

This benchmark has been implemented in 4 different dimensions 50, 100, 500 and 1000. In our experiment we will only work with dimension 100, as dimension 50 is already assessed in the CEC'2005 benchmark, and higher dimensions are outside the scope of this paper.

As for the CEC'2005 benchmark, each algorithm is run 25 and the mean error is kept. The maximum number of fitness evaluations for each run is $5000 \cdot D$.

4.3. Statistical tests

Non-parametric tests must be used for comparing the results of different search algorithms for this benchmark [16]. Given that the non-parametric tests do not require explicit conditions for being conducted, it is recommendable that the sample of results would be obtained following the same criterion to compute the same aggregation (average, mode, etc.) over the same number of runs for each algorithm and problem. We use the program available in http://sci2s.ugr.es/sicidm/

In particular, we have considered two alternative methods based on non-parametric tests to analyse the experimental results:

- Application of the Iman and Davenport's test and the Holm's method as post hoc procedure. The first test may be used to see whether there are significant statistical differences among the algorithms on a certain group of test algorithms. If differences are detected, then Holm's test is employed to compare the best algorithm (control algorithm) against the remaining ones.
- Application of the Wilcoxon matched-pairs signed-ranks test. With this test, the results of two algorithms may be directly compared.

Table 2
Test functions of the SOCO'2011 benchmark.

F1	Sphere function
F2	Schwefel's problem 2.21
F3	Rosenbrock's function
F4	Rastrigin's function
F5	Griewank's function
F6	Ackley's function
F7	Schwefel's problem 2.22
F8	Schwefel's problem 1.2
F9	Extended F10
F10	Bohachevsky
F11	Schaffer's function
F12–F19	Hybrid composition function

5. Experimental results

We have carried out the experiments of RMA-LSCh-CMA using the parameters' values proposed by the authors of MA-LSCh-CMA [32], except for the population size which was set to 60 in the previous model:

- the population size is 80
- the pool size for the NAM selection $N_{NAM} = 3$
- the mutation probability $P_{mutation} = 0.125$
- the number of evaluation allocated to each LS $I_{str} = 500$
- the LS/EA ratio $R_{LS} = 0.5$.

In Section 5.1, we first experiment the influence of the number of divisions. From the observation made, we then justify the use of a using a dynamic number of divisions in Section 5.2. In Section 5.3, we compare the results of our model against the original one, MA-LSCh-CMA and illustrate the influence of the niching strategy on the diversity of the population. Finally, in Section 5.4, we tune the parameters using IRACE [28] over the CEC'2005 benchmark functions to compare the performances of the RMA-LSCh-CMA against a sample of representative algorithms in Section 5.5.

5.1. Study of the number of divisions

When implementing a niching strategy, the most critical parameter is the size of the niches. In this section, we assess the influence of the number of divisions on the results and the diversity of the population. We tested three fixed values of *ND*: 10, 50 and 100.

5.1.1. Results on the CEC'2005 benchmark

We present here the results obtained by the RMA-LSCh-CMA with different values of *ND*. Detailed results can be seen in Appendix A.

Fig. 2 shows the average rankings obtained by the RMA-LSCh-CMA instances with different *ND* values on the 25 test functions with dimensions $D = 10, 30$, and 50. The mean rankings correspond to the average of the ranking of each algorithm on each function. We can note that the influence on the number of divisions depends on the dimension. Indeed, for smaller dimensions, a smaller number of divisions obtains better results while a higher number of divisions performs better on higher dimensions.

We first applied the Iman-Davenport's test to the results of the three instances of the model to assess any significant differences. Table 3 shows that there are no significant differences in dimension 10 and 50. In dimension 30, we can observe significant differences, thus we apply the Holm's test using the algorithm with best fitness, ND = 50, as the *control algorithm*. Table 4 show the results. It can be observed that ND = 50 gives significantly better results than with ND = 10 and they are fairly equivalent with ND = 100.

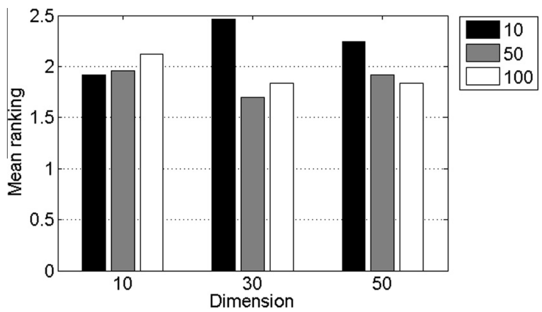

Fig. 2. Mean rankings obtained by RMA-LSCh-CMA with different number of divisions over every functions of the CEC'2005 benchmark. The lower columns corresponds to the best algorithms.

Table 3
Iman-Davenport test for significant difference between the instances of R-MA-LSCh-CMA with $ND = 10, ND = 50$ and $ND = 100$.

Dimension	*p*-Value	Significant differences?
10	0.763	No
30	0.014	Yes
50	0.333	No

Table 4
Comparison using Holm's test with $\alpha = 0.05$ of the instance where $ND = 30$ against the other instances.

i	ND	$z = (R_0 - R_i)/SE$	p	α/i	Significant difference?
2	10	2.687	0.007	0.025	Yes
1	100	0.495	0.621	0.05	No

5.1.2. Diversity study

This section aims to demonstrate the influence of the number of divisions on the diversity of the population. We analyse the evolution of the diversity along the search of RMA-LSCh-CMA with fixed values of *ND* using the *distance-to-average-point* measure as described in [51]. It consists in calculating the mean distance of each individual in the population to the average point of the population:

$$diversity(P) = \frac{1}{|L| \cdot |P|} \cdot \sum_{i=1}^{|P|} \sqrt{\sum_{j=1}^{N} (s_{ij} - \bar{s}_j)^2} \quad (1)$$

where $|L|$ is the length of the diagonal in the search space $S \subseteq \Re^N$, P the population, $|P|$ the population size, N the dimensionality of the problem, c_{ij} the jth value of the individual i and $\bar{s}_j$ the jth value of the average point of the population $\bar{s}$.

To perform this study, we ran an experiment over 25 runs in each function in dimension 10 measuring after the each pair of SSGA and LS is launched (each of which running through 500 evaluations, the diversity is calculated every approximately 1000 function evaluations).

Fig. 3 represents the evolution of the population's diversity for each instance of the RMA-LSCh-CMA.

As it was expected, the number of divisions influences the diversity. Indeed, the smaller the number of divisions, the higher is the diversity in the population remains along the search.

5.2. Dynamic vs static number of divisions

The previous section showed that the choice of the number of divisions depended on the dimension. In this section we thus assess the use of a dynamic number of divisions. In this experiment, we chose to perform 3 updates during the search,

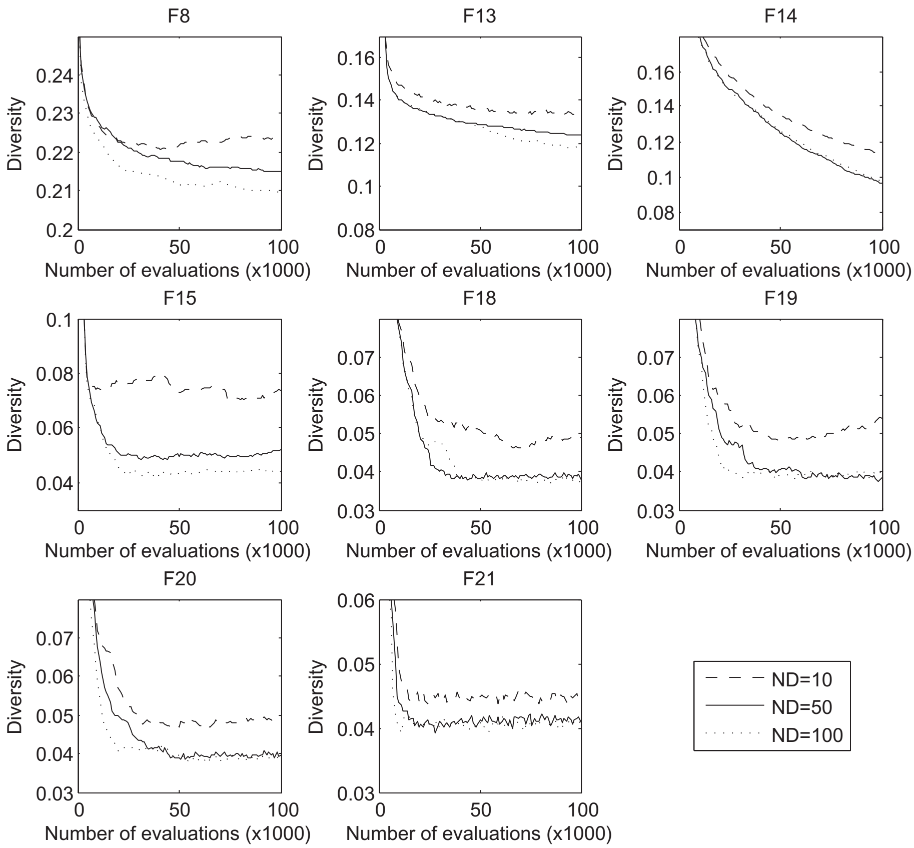

Fig. 3. Evolution of the diversity in the population for different number of divisions for different functions.

$u = 3$, with an initial number of divisions $ND_0 = 10$, giving a sequence of divisions number of 10, 20, 40 and 80. In Table 5 we compare the dynamic model with the static one with different values of *ND*.

We saw in the previous section that setting a static number of divisions influenced the results according to the dimensionality of the problem. This influence is reduced by using a dynamic number of divisions. Indeed, we can note that while, when comparing the dynamic with a static high number of divisions, both strategies are statistically equivalent in higher dimensions ($D = 30$ and $D = 50$), the dynamic model obtains better results in smaller dimensions ($D = 10$). On the other hand, when comparing it with a static small number of divisions ($ND = 10$), we obtain better performances in higher dimensions (statistically better for $D = 30$).

For the following experiments, we will use the dynamic version of the algorithm.

5.3. Comparison with the MA-LSCh-CMA

The original purpose of this work was to improve the promising results of the MA-LSCh-CMA. We analyse in this section the improvements brought by the proposed niching strategy to this algorithm and its influence on the diversity of the population using the same parameters in both algorithms.

5.3.1. Results on the CEC'2005 benchmark

The detailed results can be seen in Appendix B. Table 6 shows the Wilcoxon signed rank obtained when comparing both algorithms. We can see that the niching strategy, and the modifications it implies, improves the results in every dimensions. The results are statistically better in dimension 10, with $\alpha = 0.05$, and in dimension 50 with $\alpha = 0.1$.

5.3.2. Diversity study

We demonstrate in this section that the implementation of this niching strategy actually influences the diversity of the population. The evolution of the diversity on various functions is plotted in Fig. 4 following the conditions described Section 5.1.2.

We can see that the diversity remains higher in the population of the RMA-LSCh-CMA.

5.4. Automatic configuration

In the previous section, we demonstrated that the use of the region-based niching strategy in the LS chaining framework significantly improved the performances of this MA. However, in order to fully adapt the design of this new model to the problems at hand, we applied the automatic tuning of its parameters using the automatic configuration tool IRACE [28].

5.4.1. IRACE package

The automatic configuration tool that we use is the IRACE package. Based on previous works [4–8], it implements an automatic configuration approach based on *racing* [30]. Statistical tests are used to test for significantly inferior candidate configurations. The IRACE package, implemented as an *R* [42] package, implements a general *iterated racing* procedure. For more details on this tool, the reader can refer to [28].

Table 5
Dynamic region based MA-LSCh-CMA versus various static numbers of divisions using Wilcoxon's test.

Dimension	*ND*	*R*+ dynamic	*R*− static	*p*-Value
10	10	222.5	102.5	0.110
10	50	264.5	60.5	0.005
10	100	250	53.5	0.005
30	10	216.5	86	0.069
30	50	177	148	0.696
30	100	135.5	167	0.679
50	10	181.5	121	0.407
50	50	169.5	155.5	0.851
50	100	152.5	150	1.000

Table 6
Wilcoxon signed rank test results of RMA-LSCh-CMA vs MA-LSCh-CMA.

Dim	*R*+ RMA-LSCh-CMA	*R*− MA-LSCh-CMA	*p*-Value
10	247	78	0.022
30	202.5	99	0.152
50	211.5	90	0.089
All Dim	1927.5	854	0.004

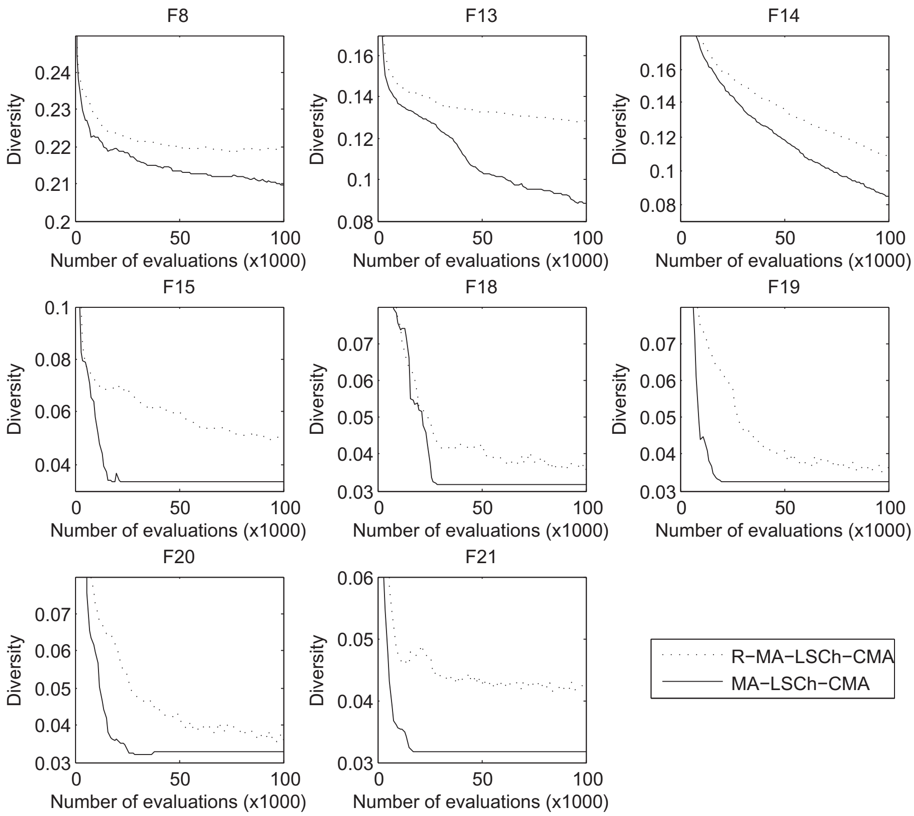

Fig. 4. Evolution of the diversity in the population of the EA of the RMA-LSCh-CMA and MA-LSCh-CMA for different functions.

The IRACE package has already been extensively tested in several research projects, leading to successful improvement over the state-of-the-art, see for instance [18,17].

The advantage of this tool is that it handles several parameter types: continuous, integer, categorical, and ordered. Continuous and integer parameters take values within a range specified by the user. Categorical parameters can take any value among a set of possible ones explicitly given by the user. An ordered parameter is a categorical parameter with a pre-defined strict order of its possible values. We also relied on its capability to parallelise the configuration phase in order to reduce considerably the amount of time required for it.

5.4.2. Application to the RMA-LSCh-CMA

We selected a set of parameters that we considered the most critical to be tuned. Those parameters are listed in Table 7 along with the ranges of search, their default values and their obtained values after tuning. The tuning budget allocated to IRACE is set to 5000. The budget corresponds to number of runs in the conditions defined by the benchmark that irace uses to perform the tuning.

In Table 8, we compare the results of the RMA-LSCh-CMA with default parameters and the ones obtained by tuning. The automatic configuration brings significant overall improvements to the model and more specifically in dimension 10 and 50.

Table 7
Parameters tuned and obtained values.

Parameters	Descriptions	Ranges	Default	Tuned
I_{str}	LS intensity, number of evaluations allocated to each LS application	$[100, 1000]$	500	950
ND_0	Initial number of divisions, defines the size of the niches/regions	$[2, 10]$	10	6
u	Number of update to be performed	$[2, 5]$	3	2
m_u	Update multiplier	$[1, 5]$	2	4
$r_{EA/LS}$	The repartition of the overall effort between the EA and the LS the higher the value the more evaluations allocated to the LS	$[0.1, 0.9]$	0.5	0.6
NP	Population size of the EA	$[40, 120]$	80	40
λ	Parameter to define the CMA-ES population size $p = 4 + \lambda ln(D)$	$[1, 10]$	3	8
μ	Defines the parent size for the CMA-ES p/μ	$[1, 5]$	2	4
α	Parameter for the $BLX - \alpha$ crossover	$[0.1, 0.9]$	0.5	0.6

Table 8
Wilcoxon signed rank test between the default version of RMA-LSCh-CMA ($R-$) and its tuned version ($R+$).

Dim	Tuned $R+$	Default $R-$	p-Value
10	241.5	62	0.011
30	160	142.5	0.830
50	234	68.5	0.019
All Dimensions	1951	832.5	0.003

For the following experiment, we use the parameters obtained by tuning and listed in Table 7.

5.5. Comparison with other algorithms

In this section, we compare the efficiency of our algorithm with IPOP-CMA-ES [2], MDE_pBX [25] and 3SOME [23]:

- IPOP-CMA-ES is the winner of the CEC2005 Real-Parameter Optimisation competition. It is a restart algorithm that uses CMA-ES and detects premature convergence and launches a restart strategy that doubles the population size on each restart. This process allows a more global approach of the search which empowers the operation of the CMA-ES on multimodal functions.
- MDE_pBX is a state-of-the-art differential evolution (DE). It uses a new mutation operator called *DE/current-to-gr_best/1*, a variant of the *DE/current-to-best/1*, which uses the best of a random group of individuals in the population instead of the global best and performs a the recombination with a random individual of the p best individuals of the population. It also adapts its parameters according to the successes and failures of each of them.
- 3SOME is an example from the MC family. It is a simple, in its concept and implementation, memetic optimiser based on the philosophical concept of Ockhams Razor. The search is divided in three stages each of which corresponds to a variations between explorations and exploitation named long middle and short distance exploration.

The experiments on these algorithms have been performed using the original source code provided by the authors or, when available, their published results.

5.5.1. Results on the CEC'2005 benchmark

Table 9 shows the results of the comparison with those three algorithms applying the Wilcoxon's test. With regards to the detailed results in Appendix Tables B.14, B.15 and B.16, our algorithm obtains signitificantly better results than 3SOME. The complexity of our algorithm that is implied by using the restricting use of region-based niching strategy in this memetic framework and the application of CMA-ES as LS method is rewarded by high performances in the functions proposed in the CEC'2005 benchmark compared to 3SOME which uses simpler and less computationally expensive components.

Concerning MDE_pBX, the overall performances of our algorithm is significantly better (with $\alpha = 0.05$). When analysing the results on each dimension individually, we can see that the superiority of our algorithm appears in higher dimensions (30 and 50 with $\alpha = 0.05$).

Finally, IPOP-CMA-ES obtains equivalent results over the whole benchmark. The differences. We note a small tendency of our algorithm to perform better on higher dimensions and worse on smaller ones although no significant difference can be detected.

Table 9
Wilcoxon signed rank test results between RMA-LSCh-CMA ($R+$) and reference algorithms ($R-$).

Dim	RMA-LSCh-CMA vs	$R+$	$R-$	p-Value
10	3SOME	291.5	10	5.13E−6
10	MDE_pBX	195.5	108	0.230
10	IPOP-CMA-ES	94.5	209	0.117
30	3SOME	299.5	25.5	5.88E−5
30	MDE_pBX	257.5	67.5	0.01
30	IPOP-CMA-ES	162.5	162.5	1
50	3SOME	282.5	42.5	6.73E−4
50	MDE_pBX	299.5	25.5	5.88E−5
50	IPOP-CMA-ES	212	113	0.191
All Dimensions	3SOME	2560.5	218	2.97E−10
All Dimensions	MDE_pBX	2239.5	541	5.11E−6
All Dimensions	IPOP-CMA-ES	1396	1387.5	1

Table 10
Wilcoxon signed ranks test results between RMA-LSCh-CMA ($R+$) reference algorithms ($R-$) on the SOCO'2011 benchmark in dimension 100.

RMA-LSCh-CMA vs	$R+$	$R-$	p-Value
MA-LSCh-CMA	119.5	70.5	0.324
3SOME	79	92	0.777
MDE_pBX	180	10	1.64E−4
IPOP-CMAES	153.5	36.5	0.017

5.5.2. Results on the SOCO'2011 benchmark

We saw in the previous section that the performance of the RMA-LSCh-CMA algorithm against the IPOP-CMA-ES was improving when increasing the dimensionality of the problems. It is thus interesting to assess the performances of this model on higher dimensions. The CEC'2005 benchmark does not propose problems in dimensions higher than 50. We thus used the Soft Computing Special Issue on Large Scale Continuous Optimisation Problems (SOCO'2011) benchmark to assess the performances of our proposal against IPOP-CMA-ES in a higher dimension. We performed the experiments in dimension 100. The detailed results can be seen in Appendix C.

We can see from Table 10 that we obtain better results with a significant level of $\alpha = 0.05$ in higher dimensions than the IPOP-CMA-ES and MDE_pBX. Against the MA-LSCh-CMA and the 3SOME, the results obtained are statistically equivalent. However, we note that 3SOME obtains slightly better results. The declining performances of our algorithm compared to this model can be explained by the decreasing quality of the search provided by CMA-ES as LS.

Not only that the CMA-ES' performance declines when increasing the dimensionality of a problem, its use becomes computationally impossible for very large scale problems. Further experiments on higher dimensions would thus require the modification of the LS method. For instance, the MA algorithm with LS chaining proposed in [33] is applied to the SOCO'2011 benchmark (in dimensions up to 1000) using a Solis Wets algorithm [46] as LS instead of CMA-ES. Alternatively, the parallel memetic structures [10] use two complementary LS methods, multiple trajectory search [50] and Rosenbrock algorithm [44]. The RMA-LSCh-CMA should thus be modified following such proposals to study its applicability on large scale optimisation problems.

6. Conclusion

The aim of this paper is to present a novel niching mechanism for MAs which can improve the solutions by maintaining the diversity of the population. It demonstrates the importance of separating the effort of the global search from the refinement of the solution. To avoid the competition between the EA and the LS, we have decided to divide the search space

Table A.11
Results in dimension 10 of the RMA-LSCh-CMA with various values of ND.

F/ND	10	50	100
F1	1.00E−008	1.00E−008	1.00E−008
F2	1.00E−008	1.00E−008	1.00E−008
F3	1.00E−008	1.00E−008	1.00E−008
F4	1.00E−008	1.00E−008	1.00E−008
F5	1.00E−008	1.00E−008	1.00E−008
F6	1.00E−008	5.68E−003	1.68E−003
F7	1.00E−008	1.00E−008	1.00E−008
F8	2.04E+001	2.04E+001	2.04E+001
F9	8.15E−001	7.96E−002	1.00E−008
F10	4.18E+000	2.35E+000	1.83E+000
F11	3.32E−001	1.29E+000	1.64E+000
F12	1.47E+002	1.22E+002	2.19E+002
F13	6.29E−001	5.69E−001	4.78E−001
F14	2.84E+000	2.52E+000	2.15E+000
F15	2.13E+002	2.67E+002	2.72E+002
F16	8.43E+001	9.09E+001	9.02E+001
F17	9.72E+001	9.34E+001	9.28E+001
F18	7.79E+002	8.47E+002	8.57E+002
F19	7.63E+002	8.03E+002	8.60E+002
F20	7.51E+002	8.21E+002	8.36E+002
F21	7.47E+002	7.70E+002	7.70E+002
F22	7.42E+002	7.35E+002	7.30E+002
F23	9.31E+002	9.47E+002	9.35E+002
F24	2.36E+002	2.12E+002	2.76E+002
F25	4.10E+002	4.06E+002	4.40E+002

into rigid niches ensuring each one only contains one solution. The search space is thus divided into equal hypercubes we called regions. In order to assess the efficiency of this strategy, we implemented it in the MA-LSCh-CMA algorithm, creating an algorithm we called RMA-LSCh-CMA. It led to two major modifications. The first one is to ensure that only one solution of the EA's population can be present in a region. This ensures that a certain diversity in the population is maintained and that the close neighbourhood of a solution will not be explored by the EA as this task is meant to be more efficiently performed by the LS method. The second modification is the initialisation of the LS. It is now initialised according to the size the regions to ensure that the region the solution to which the LS is applied is properly explored. Both modifications go in the sense of limiting the competition between the LS and EA by limiting the EA in the performance of the exploitation effort and forcing the LS to focus its search on the close surroundings of solution.

Table A.12
Results in dimension 30 of the RMA-LSCh-CMA with various values of *ND*.

F/*ND*	10	50	100
F1	1.00E−008	1.00E−008	1.00E−008
F2	1.00E−008	1.00E−008	1.00E−008
F3	1.00E−008	1.06E−008	1.00E−008
F4	2.43E+001	4.02E−001	2.98E−001
F5	9.27E+001	3.77E+001	5.70E+000
F6	2.83E+001	1.49E+001	1.57E+001
F7	6.90E−004	1.00E−008	1.00E−008
F8	2.10E+001	2.09E+001	2.09E+001
F9	6.70E+000	2.73E−002	5.69E−004
F10	2.46E+001	1.79E+001	1.71E+001
F11	4.04E+000	1.24E+001	1.60E+001
F12	1.88E+003	1.64E+003	2.26E+003
F13	3.46E+000	2.50E+000	2.17E+000
F14	1.28E+001	1.26E+001	1.27E+001
F15	3.32E+002	3.15E+002	3.14E+002
F16	9.69E+001	8.57E+001	7.55E+001
F17	9.05E+001	7.18E+001	7.36E+001
F18	9.07E+002	9.02E+002	9.02E+002
F19	9.03E+002	9.02E+002	9.02E+002
F20	9.03E+002	9.06E+002	9.06E+002
F21	5.00E+002	5.00E+002	5.00E+002
F22	8.95E+002	8.67E+002	8.76E+002
F23	5.50E+002	5.34E+002	5.57E+002
F24	2.00E+002	2.00E+002	2.00E+002
F25	2.10E+002	2.11E+002	2.13E+002

Table A.13
Results in dimension 50 of the RMA-LSCh-CMA with various values of *ND*.

F/*ND*	10	50	100
F1	1.00E−008	1.00E−008	1.00E−008
F2	1.00E−008	1.01E−008	1.03E−008
F3	1.00E−008	1.05E−008	1.00E−008
F4	3.82E+003	1.06E+003	1.83E+003
F5	2.22E+003	1.82E+003	1.70E+003
F6	1.87E+001	9.48E+000	3.79E+001
F7	1.00E−008	1.08E−003	1.00E−008
F8	2.11E+001	2.11E+001	2.11E+001
F9	5.69E−001	2.19E−002	1.00E−003
F10	6.60E+001	3.63E+001	3.81E+001
F11	1.11E+001	2.61E+001	3.27E+001
F12	1.47E+004	1.03E+004	1.18E+004
F13	6.00E+000	4.52E+000	4.06E+000
F14	2.26E+001	2.23E+001	2.21E+001
F15	3.13E+002	3.57E+002	3.01E+002
F16	5.89E+001	7.12E+001	5.33E+001
F17	9.59E+001	8.47E+001	5.92E+001
F18	8.72E+002	8.97E+002	9.21E+002
F19	8.21E+002	9.21E+002	9.20E+002
F20	8.97E+002	8.93E+002	9.21E+002
F21	5.24E+002	5.12E+002	5.00E+002
F22	9.41E+002	9.35E+002	9.12E+002
F23	5.67E+002	5.39E+002	5.53E+002
F24	2.00E+002	2.00E+002	2.00E+002
F25	2.14E+002	2.15E+002	2.19E+002

In order to limit the dependence of the model to the niche size, we also proposed a method to automatically update the number divisions per dimensions. This number is increased along the search to decrease the niche size.

In this paper, we tested our algorithm on classical benchmarks of global optimisation problems. Future works consists in two subsequent experiments.

The first is to test RMA-LSCh-CMA on large scale optimisation problems. For the scalability of the functions it is composed of, the SOCO'2011 benchmark offers such a possibility. We however saw that the limitations of using CMA-ES as LS method when increasing the dimensionality compared to other memetic models. Such experiments would imply the modification of this LS method with a more scalable one.

Table B.14
Results on the CEC'2005 benchmark in dimension 10.

F	RMA-LSCh-CMA	MA-LSCh-CMA	IPOP-CMAES	MDE_pBX	3SOME
F1	1.00E−008	1.00E−008	1.00E−008	1.00E−008	1.00E−008
F2	1.00E−008	1.00E−008	1.00E−008	1.00E−008	1.00E−008
F3	1.00E−008	1.00E−008	1.00E−008	1.00E−008	4.57E+004
F4	1.00E−008	5.54E−003	1.00E−008	1.00E−008	2.00E+002
F5	1.00E−008	6.75E−007	1.00E−008	1.00E−008	1.76E+003
F6	1.00E−008	3.19E−001	1.00E−008	1.59E−001	6.64E+001
F7	1.00E−008	1.43E−001	1.00E−008	1.27E+003	1.27E+003
F8	2.03E+001	2.00E+001	2.00E+001	2.01E+001	2.00E+001
F9	1.00E−008	1.00E−008	2.39E−001	1.00E−008	1.00E−008
F10	2.79E+000	2.67E+000	7.96E−002	4.61E+000	4.27E+001
F11	5.04E−001	2.43E+000	9.34E−001	2.20E+000	7.38E+000
F12	6.31E+001	1.14E+002	2.93E+001	9.23E+002	2.25E+002
F13	4.83E−001	5.45E−001	6.96E−001	5.08E−001	4.72E−001
F14	2.55E+000	2.25E+000	3.01E+000	2.50E+000	4.18E+000
F15	1.95E+002	2.24E+002	2.28E+002	2.67E+002	2.28E+002
F16	9.48E+001	9.18E+001	9.13E+001	9.80E+001	1.99E+002
F17	9.52E+001	1.01E+002	1.23E+002	1.08E+002	2.28E+002
F18	7.42E+002	8.84E+002	3.32E+002	6.30E+002	8.90E+002
F19	7.17E+002	8.78E+002	3.26E+002	6.24E+002	9.28E+002
F20	7.93E+002	8.63E+002	3.00E+002	6.71E+002	9.14E+002
F21	7.03E+002	7.94E+002	5.00E+002	6.54E+002	9.26E+002
F22	6.76E+002	7.53E+002	7.29E+002	7.55E+002	8.60E+002
F23	8.91E+002	8.88E+002	5.59E+002	9.03E+002	9.23E+002
F24	2.36E+002	2.28E+002	2.00E+002	2.36E+002	2.88E+002
F25	4.08E+002	4.55E+002	3.74E+002	8.60E+002	1.80E+003
F26	2.42E+000	2.94E+000	2.12E+000	3.16E+000	4.36E+000

Table B.15
Results on the CEC'2005 benchmark in dimension 30.

F	RMA-LSCh-CMA	MA-LSCh-CMA	IPOP-CMAES	MDE_pBX	3SOME
F1	1.00E−008	1.00E−008	1.00E−008	1.00E−008	1.00E−008
F2	1.00E−008	1.00E−008	1.00E−008	1.00E−008	1.00E−008
F3	1.00E−008	2.75E+004	1.00E−008	4.53E+004	1.82E+005
F4	5.94E−001	3.02E+002	1.11E+004	1.84E−007	7.88E+003
F5	1.00E−008	1.26E+003	1.00E−008	4.65E+000	1.25E+004
F6	1.00E−008	1.12E+000	1.00E−008	1.28E+000	8.40E+001
F7	4.93E−004	1.75E−002	1.00E−008	4.70E+003	4.70E+003
F8	2.10E+001	2.00E+001	2.01E+001	2.03E+001	2.00E+001
F9	1.89E−004	1.00E−008	9.38E−001	1.67E+001	1.00E−008
F10	1.95E+001	2.25E+001	1.65E+000	2.67E+001	3.39E+002
F11	5.65E+000	2.15E+001	5.48E+000	1.46E+001	3.25E+001
F12	1.59E+003	1.67E+003	4.43E+004	1.75E+005	2.84E+003
F13	2.07E+000	2.03E+000	2.49E+000	3.65E+000	1.73E+000
F14	1.26E+001	1.25E+001	1.29E+001	1.25E+001	1.37E+001
F15	3.19E+002	3.00E+002	2.08E+002	2.87E+002	2.09E+002
F16	1.41E+002	1.26E+002	3.50E+001	1.54E+002	4.20E+002
F17	2.06E+002	1.83E+002	2.91E+002	1.70E+002	4.09E+002
F18	9.05E+002	8.98E+002	9.04E+002	9.05E+002	9.68E+002
F19	9.05E+002	9.01E+002	9.04E+002	8.96E+002	9.79E+002
F20	9.01E+002	8.96E+002	9.04E+002	9.06E+002	9.56E+002
F21	5.00E+002	5.12E+002	5.00E+002	5.24E+002	1.02E+003
F22	8.41E+002	8.80E+002	8.03E+002	8.52E+002	1.13E+003
F23	5.49E+002	5.34E+002	5.34E+002	5.84E+002	8.95E+002
F24	2.00E+002	2.00E+002	9.10E+002	2.30E+002	4.13E+002
F25	2.10E+002	2.14E+002	2.11E+002	9.68E+002	1.72E+003

Then, we wish to assess the performances of RMA-LSCh-CMA on multimodal problems. We used in this study the region-based niching strategy as a diversity control mechanism for the search of a global optimum. However, niching strategies can also be applied to the identification of multiple optima in a fitness landscape.

Acknowledgement

This work is supported by the Ministerio de Ciencia e Innovacion (Spain) under Grant TIN2011-28488. Benjamin Lacroix acknowledges the support from the MIBISOC network, an Initial Training Network funded by the European Union, Grant PITN-GA-2009-238819.

Appendix A. Results of the static model on the CEC'2005 benchmark

See Tables A11–A13.

Table B.16
Results on the CEC'2005 benchmark in dimension 50.

F	RMA-LSCh-CMA	MA-LSCh-CMA	IPOP-CMAES	MDE_pBX	3SOME
F1	1.00E−008	1.00E−008	1.00E−008	1.00E−008	1.00E−008
F2	1.00E−008	3.06E−002	1.00E−008	1.00E−008	1.00E−008
F3	1.00E−008	3.21E+004	1.00E−008	8.37E+004	1.51E+005
F4	4.73E+002	3.23E+003	4.68E+005	2.97E+001	3.37E+004
F5	4.92E+002	2.69E+003	2.85E+000	2.31E+003	1.90E+004
F6	5.01E+000	4.10E+000	1.00E−008	9.55E+000	1.01E+002
F7	1.00E−008	5.40E−003	1.00E−008	6.20E+003	6.20E+003
F8	2.11E+001	2.00E+001	2.01E+001	2.02E+001	2.00E+001
F9	1.07E−003	1.00E−008	1.39E+000	5.24E+001	1.00E−008
F10	4.58E+001	5.01E+001	1.72E+000	6.07E+001	8.61E+002
F11	1.22E+001	4.13E+001	1.17E+001	3.75E+001	6.23E+001
F12	1.51E+004	1.39E+004	2.27E+005	9.47E+005	6.01E+003
F13	3.66E+000	3.15E+000	4.59E+000	9.11E+000	3.01E+000
F14	2.23E+001	2.22E+001	2.29E+001	2.24E+001	2.35E+001
F15	2.90E+002	3.72E+002	2.04E+002	3.45E+002	2.72E+002
F16	6.96E+001	6.90E+001	3.09E+001	9.78E+001	5.13E+002
F17	1.22E+002	1.47E+002	2.34E+002	1.38E+002	5.18E+002
F18	8.45E+002	9.41E+002	9.13E+002	9.33E+002	1.12E+003
F19	8.72E+002	9.38E+002	9.12E+002	9.32E+002	1.10E+003
F20	8.70E+002	9.28E+002	9.12E+002	9.35E+002	1.13E+003
F21	5.24E+002	5.00E+002	1.00E+003	5.67E+002	8.50E+002
F22	8.63E+002	9.14E+002	8.05E+002	9.00E+002	1.17E+003
F23	5.39E+002	5.39E+002	1.01E+003	5.87E+002	8.58E+002
F24	2.00E+002	2.00E+002	9.55E+002	3.61E+002	1.16E+003
F25	2.14E+002	2.21E+002	2.15E+002	1.23E+003	1.78E+003

Table C.17
Results on the SOCO'2011 benchmark in dimension 100.

F	RMA-LSCh-CMA	MA-LSCh-CMA	IPOP-CMAES	MDE_pBX	3SOME
F1	0.00E+000	0.00E+000	0.00E+000	1.41E−013	0.00E+000
F2	8.27E−011	1.26E−001	1.51E−010	6.66E+001	1.39E−008
F3	2.03E+002	1.15E+001	3.88E+000	1.70E+002	5.00E+001
F4	0.00E+000	1.47E+000	2.50E+002	2.11E+002	8.30E−001
F5	3.65E−003	0.00E+000	1.58E−003	3.65E−002	0.00E+000
F6	1.09E−012	8.07E−014	2.12E+001	3.09E+000	0.00E+000
F7	4.64E−014	0.00E+000	4.22E−004	0.00E+000	4.79E−003
F8	1.21E−004	2.26E+003	0.00E+000	2.40E−001	0.00E+000
F9	5.60E+002	5.64E+002	1.02E+002	5.57E+002	5.81E+002
F10	0.00E+000	0.00E+000	1.66E+001	3.40E+001	1.09E−002
F11	6.55E+000	6.82E−001	1.64E+002	9.92E+001	9.66E+000
F12	4.53E+000	1.25E+000	4.17E+002	1.61E+002	5.55E−002
F13	7.10E+001	1.04E+002	4.21E+002	3.47E+002	9.17E+001
F14	1.66E−001	1.00E+000	2.55E+002	1.69E+002	6.72E−001
F15	3.57E−014	4.12E−007	6.30E−001	5.11E+000	3.17E−002
F16	3.75E+000	1.29E−001	8.59E+002	2.62E+002	8.42E−002
F17	4.14E+001	2.32E+002	1.51E+003	4.65E+002	3.78E+001
F18	1.26E+000	1.68E−001	3.07E+002	1.05E+002	3.15E−002
F19	1.42E−014	0.00E+000	2.02E+001	2.22E+001	7.97E−003

Appendix B. Results of the dynamic RMA-LSCh-CMA, the MA-LSCh-CMA, IPOP-CMA-ES, MDE_pBX and 3SOME on the CEC'2005 benchmark

See Tables B14–B16.

Appendix C. Results on the SOCO'2011 benchmark in dimension 100

See Table C.17.

References

[1] C.F. Agostinho, C. Fernandes, A. Rosa, A study on non-random mating and varying population size in genetic algorithms using a royal road function, in: Proceedings of the 2001 IEEE Congress on Evolutionary Computation, IEEE Press, 2001, pp. 60–66.
[2] A. Auger, N. Hansen, A restart CMA evolution strategy with increasing population size, in: 2005 IEEE Congress on Evolutionary Computation, pp. 1769–1776.
[3] T. Back, D.B. Fogel, Z. Michalewicz (Eds.), Handbook of Evolutionary Computation, IOP Publishing Ltd., Bristol, UK, 1997.
[4] P. Balaprakash, M. Birattari, T. Stützle, Improvement strategies for the F-race algorithm: sampling design and iterative refinement, in: T. Bartz-Beielstein, M.J. Blesa, C. Blum, B. Naujoks, A. Roli, G. Rudolph, M. Sampels (Eds.), Hybrid Metaheuristics, Lecture Notes in Computer Science, vol. 4771, Springer, Heidelberg, Germany, 2007, pp. 108–122.
[5] M. Birattari, The Problem of Tuning Metaheuristics as Seen from a Machine Learning Perspective, Ph.D. thesis, Université Libre de Bruxelles, Brussels, Belgium, 2004.
[6] M. Birattari, Tuning metaheuristics: a machine learning perspective, Studies in Computational Intelligence, vol. 197, Springer, Berlin/Heidelberg, Germany, 2009.
[7] M. Birattari, T. Stützle, L. Paquete, K. Varrentrapp, A racing algorithm for configuring metaheuristics, in: W.B. Langdon et al. (Eds.), Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2002, Morgan Kaufman Publishers, San Francisco, CA, 2002, pp. 11–18.
[8] M. Birattari, Z. Yuan, P. Balaprakash, T. Stützle, F-race and iterated F-race: an overview, in: T. Bartz-Beielstein, M. Chiarandini, L. Paquete, M. Preuss (Eds.), Experimental Methods for the Analysis of Optimization Algorithms, Springer, Berlin, Germany, 2010, pp. 311–336.
[9] P. Bonissone, R. Subbu, N. Eklund, T. Kiehl, Evolutionary algorithms + domain knowledge = real-world evolutionary computation, IEEE Trans. Evol. Comput. 10 (2006) 256–280.
[10] F. Caraffini, F. Neri, G. Iacca, A. Mol, Parallel memetic structures, Inf. Sci. 227 (2013) 60–82.
[11] X. Chen, Y.S. Ong, M.H. Lim, K.C. Tan, A multi-facet survey on memetic computation, IEEE Trans. Evol. Comput. 15 (2011) 591–607.
[12] C. Cotta, A. Fernndez, Memetic algorithms in planning, scheduling, and timetabling, in: K. Dahal, K. Tan, P. Cowling (Eds.), Evolutionary Scheduling, Studies in Computational Intelligence, vol. 49, Springer, Berlin/Heidelberg, 2007, pp. 1–30.
[13] S. Das, S. Maity, B.Y. Qu, P. Suganthan, Real-parameter evolutionary multimodal optimization a survey of the state-of-the-art, Swarm Evol. Comput. 1 (2011) 71–88.
[14] L. Davis, Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, 1991.
[15] R. Dawkins, The Selfish Gene, Oxford University Press, 1990.
[16] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (2011) 3–18.
[17] J. Dubois-Lacoste, M. López-Ibáñez, T. Stützle, Automatic configuration of state-of-the-art multi-objective optimizers using the TP + PLS framework, in: N. Krasnogor, P.L. Lanzi (Eds.), Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2011, ACM Press, New York, NY, 2011, pp. 2019–2026.
[18] J. Dubois-Lacoste, M. López-Ibáñez, T. Stützle, A hybrid TP+PLS algorithm for bi-objective flow-shop scheduling problems, Comput. Oper. Res. 38 (2011) 1219–1236.
[19] L.J. Eshelman, J.D. Schaffer, Real-coded genetic algorithms in genetic algorithms by preventing incest, Found. Genet. Algorithms 2 (1993) 187–202.
[20] A.E. Fallahi, C. Prins, R.W. Calvo, A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem, Comput. Oper. Res. 35 (2008) 1725–1741 (Part Special Issue: Algorithms and Computational Methods in Feasibility and Infeasibility).
[21] D.E. Goldberg, S. Voessner, Optimizing global-local search hybrids, in: Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufman, 1999, pp. 220–228.
[22] N. Hansen, S.D. Müller, P. Koumoutsakos, Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES), Evol. Comput. 1 (2003) 1–18.
[23] G. Iacca, F. Neri, E. Mininno, Y.S. Ong, M.H. Lim, Ockhams Razor in memetic computing: three stage optimal memetic exploration, Inf. Sci. 188 (2012) 17–43.
[24] H. Ishibuchi, T. Yoshida, T. Murata, Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling, IEEE Trans. Evol. Comput. 7 (2003) 204–223.
[25] S.M. Islam, S. Das, S. Ghosh, S. Roy, P.N. Suganthan, An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization, IEEE Trans. Syst. Man Cyber. Part B Cyber.: A Publ. IEEE Syst. Man Cyber. Soc. 42 (2012) 482–500.
[26] N. Krasnogor, J. Smith, A tutorial for competent memetic algorithms: model, taxonomy, and design issues, IEEE Tran. Evol. Comput. 9 (2005) 474–488.
[27] D. Liu, K. Tan, C. Goh, W. Ho, A multiobjective memetic algorithm based on particle swarm optimization, IEEE Trans. Syst. Man Cyber. Part B: Cyber. 37 (2007) 42–50.
[28] M. López-Ibáñez, J. Dubois-Lacoste, T. Stützle, M. Birattari, The irace package, Iterated Race for Automatic Algorithm Configuration, Technical Report TR/IRIDIA/2011-004, IRIDIA, Université Libre de Bruxelles, Belgium, 2011.
[29] M. Lozano, D. Molina, F. Herrera, Editorial scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems, Soft Comput. 15 (2011) 2085–2087.
[30] O. Maron, A.W. Moore, The racing algorithm: model selection for lazy learners, Artif. Intell. Res. 11 (1997) 193–225.
[31] J.E. Mendoza, B. Castanier, C. Guret, A.L. Medaglia, N. Velasco, A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands, Comput. Oper. Res. 37 (2010) 1886–1898.
[32] D. Molina, M. Lozano, C. García-Martínez, F. Herrera, Memetic algorithms for continuous optimisation based on local search chains, Evol. Comput. 18 (2010) 27–63.
[33] D. Molina, M. Lozano, A.M. Sánchez, F. Herrera, Memetic algorithms based on local search chains for large scale continuous optimisation problems: MA-SSW-Chains, Soft Comput. 15 (2010) 2201–2220.
[34] P. Moscato, On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Toward Memetic Algorithms, Technical Report, 1989.
[35] P. Moscato, A Gentle Introduction to Memetic Algorithms, Kluwer Academic Publishers, 2003.
[36] H. Mülenbein, D. Schlierkamp-Voosen, Predictive models for the breeding genetic algorithm in continuous parameter optimization, Evol. Comput. 1 (1993) 25–49.
[37] F. Neri, C. Cotta, Memetic algorithms and memetic computing optimization: a literature review, Swarm Evol. Comput. 2 (2012) 1–14.

[38] F. Neri, E. Mininno, Memetic compact differential evolution for cartesian robot control, IEEE Comput. Intell. Mag. 5 (2010) 54–65.
[39] F. Neri, J. Toivanen, R.A. Mäkinen, An adaptive evolutionary algorithm with intelligent mutation local searchers for designing multidrug therapies for HIV, Appl. Intell. 27 (2007) 219–235.
[40] Y.S. Ong, N. Krasnogor, H. Ishibuchi, Special issue on memetic algorithms, IEEE Trans. Syst. Man Cyber. Part B: Cyber. 37 (2007) 2–5.
[41] Y.S. Ong, M. Lim, X. Chen, Research frontier: memetic computation – past, present & future, IEEE Comput. Intell. Mag. 5 (2010) 24–31.
[42] R Development Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2008.
[43] T. Rogalsky, R.W. Derksen, Hybridization of differential evolution for aerodynamic design, in: Proceedings of the 8th Annual Conference of the Computational Fluid Dynamics Society of Canada, pp. 729–736.
[44] H.H. Rosenbrock, An automatic method for finding the greatest or least value of a function, Comput. J. 3 (1960) 175–184.
[45] J.D. Schaffer (Ed.), The GENITOR Algorithm and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials is Best, Morgan Kaufman Publishers, San Mateo, 1989.
[46] F. Solis, R. Wets, Minimization by random search techniques, Math. Oper. Res. 6 (1981) 19–30.
[47] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.P. Chen, A. Auger, S. Tiwari, Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization, Technical Report, 2005.
[48] M. Tang, X. Yao, A memetic algorithm for vlsi floorplanning, IEEE Trans. Syst. Man Cyber.—Part B: Cyber. 37 (2007) 62–69.
[49] C.K. Ting, C.C. Liao, A memetic algorithm for extending wireless sensor network lifetime, Inf. Sci. 180 (2010) 4818–4833.
[50] L.Y. Tseng, C. Chen, Multiple trajectory search for large scale global optimization, in: IEEE Congress on Evolutionary Computation, 2008, CEC 2008, (IEEE World Congress on Computational Intelligence), pp. 3052–3059.
[51] R.K. Ursem, Diversity-guided evolutionary algorithms, in: Proceedings of the Congress on Evolutionary Computation, IEEE Press, 2002, pp. 1633–1640.