Computers & Operations Research 37 (2010) 481–497

Contents lists available at ScienceDirect

Computers & Operations Research

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

Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: Overview and progress report[☆]

M. Lozano[a,*], C. García-Martínez[b]

[a]Department of Computer Science and Artificial Intelligence, University of Granada, Granada 18071, Spain
[b]Department of Computing and Numerical Analysis, University of Córdoba, Córdoba 14071, Spain

ARTICLE INFO

Available online 28 February 2009

Keywords:
Hybrid metaheuristics
Evolutionary algorithms
Intensification and diversification
Iterated local search

ABSTRACT

Nowadays, a promising way to obtain hybrid metaheuristics concerns the combination of several search algorithms with strong specialization in intensification and/or diversification. The flexible architecture of evolutionary algorithms allows specialized models to be obtained with the aim of providing intensification and/or diversification. The outstanding role that is played by evolutionary algorithms at present justifies the choice of their specialist approaches as suitable ingredients to build hybrid metaheuristics.

This paper focuses on hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification. We first give an overview of the existing research on this topic, describing several instances grouped into three categories that were identified after reviewing specialized literature. Then, with the aim of complementing the overview and providing additional results and insights on this line of research, we present an instance that consists of an iterated local search algorithm with an evolutionary perturbation technique. The benefits of the proposal in comparison to other iterated local search algorithms proposed in the literature to deal with binary optimization problems are experimentally shown. The good performance of the reviewed approaches and the suitable results shown by our instance allow an important conclusion to be achieved: the use of evolutionary algorithms specializing in intensification and diversification for building hybrid metaheuristics becomes a prospective line of research for obtaining effective search algorithms.

1. Introduction

In the last few years, a new family of search and optimization algorithms have arisen based on extending basic heuristic methods by including them into an iterative framework augmenting their exploration capabilities. This group of advanced approximate algorithms has received the name *metaheuristics* (MHs) [28] and an overview of various existing methods is found in [11]. MHs have proven to be highly useful for approximately solving difficult optimization problems in practice because they may obtain good solutions in a reduced amount of time. Simulated annealing, tabu search, evolutionary algorithms (EAs), ant colony optimization, estimation of distribution algorithms, scatter search, path relinking, greedy randomized adaptive search procedure (GRASP), multi-start and iterated local search (ILS), guided local search, and variable neighborhood search (VNS) are, among others, often listed as examples of classical MHs. They have individual historical backgrounds and follow different paradigms and philosophies.

Over the last years, a large number of search algorithms were reported that do not purely follow the concepts of one single classical MH, but they attempt to obtain the best from a set of MHs (and even other kinds of optimization methods) that perform together and complement each other to produce a profitable synergy from their combination. These approaches are commonly referred to as *hybrid* MHs [83,93].

Intensification and diversification (I&D) are two major issues when designing a global search method [11]. Diversification generally refers to the ability to visit many and different regions of the search space, whereas intensification refers to the ability to obtain high quality solutions within those regions. A search algorithm should strike a tactical balance between these two sometimes-conflicting goals. Most classical MHs have several components for intensification and diversification. Blum and Roli [11] define an *I&D component* as any algorithmic or functional component that has intensification and/or diversification effect on the search process. Examples are genetic operators, perturbations of probability distributions, the

---

[☆] This work was supported by Project TIN2005-08386-C05-01.

[*] Corresponding author. Tel.: +34 958 244 258; fax: +34 958 243 317.
E-mail addresses: lozano@decsai.ugr.es (M. Lozano), cgarcia@uco.es (C. García-Martínez).

0305-0548/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2009.02.010

use of tabu lists, or changes in the objective function. Thus, I&D components are operators, actions, or strategies of MHs.

In general, providing an adequate balance between the I&D components of an MH becomes a very complicate task [94]. In fact, although most classical MHs attempt to achieve this objective in their own way, it turns out that some of them show clear trend toward intensification and others, toward diversification, i.e., they show certain specialization in intensification or diversification. An alternative approach to force MHs to have themselves responsibilities for both I&D involves the design of hybrid MHs with *search algorithms specializing in I&D*, which combine this type of algorithms with the objective of compensating each other and put together their complementary behaviors (the exploration and exploitation of the search space).

EAs [7,8,19] are stochastic search methods that mimic the metaphor of natural biological evolution. EAs rely on the concept of a *population* of individuals (representing search points in the space of potential solutions to a given problem), which undergo probabilistic operators such as mutation, selection, and (sometimes) recombination to evolve toward increasingly better fitness values of the individuals. There has been a variety of slightly different EAs that, basically, fall into four different categories, which have been developed independently from each other. These are *evolution strategies* [9], *genetic algorithms* (GAs) [29], *genetic programming* [55], and *evolutionary programming* [23]. EAs have recently received increased interest because they offer practical advantages to researchers facing difficult optimization problems (they may locate high performance regions of vast and complex search spaces). Other advantages include the simplicity of the approach, their flexibility, and their robust response to changing circumstances.

Precisely, the flexibility offered by the EA paradigm allows specialized models to be obtained with the aim of providing intensification and/or diversification, i.e., *EAs specializing in I&D* ($EA_{I\&D}$). On the one hand, beneficial diversification properties are inherent to EAs, because they manage populations of solutions, providing a natural and intrinsic way for exploring search space. Even more, many techniques were presented in the literature that favor diversity in EA population with the aim of consolidating diversification associated with these algorithms [3,13,21,29,54,62]. Then, *specialization of EAs in diversification* (EA_D) becomes really viable. On the other hand, some components of EAs may be specifically designed and their strategy parameters tuned, in order to provide an effective refinement. In fact, several *EAs specializing in intensification* (EA_I) have been presented with this aim [49,61,73].

The outstanding role played by EAs at present along with the great interest raised by their hybridizations with other algorithms [33,82] endorse the choice of their specialist approaches as suitable ingredients to build hybrid MHs with search algorithms specializing in I&D. In fact, the design of hybrid MHs with $EA_{I\&D}$ (HMH-$EA_{I\&D}$) is an innovative line of research with prospective future as way for obtaining search algorithms that may achieve accurate and reliable solutions to hard real-world problems.

The goal in this article is twofold. Firstly, we attempt to paint a more complete picture of HMH-$EA_{I\&D}$ than before. To do so, we overview existing design principles for these algorithms and align them to arrive at an insightful line of research. We cite the existing literature whenever relevant. From the literature reviewed, we have identified three lines of research in designing HMH-$EA_{I\&D}$. The first two, collaborative HMH-$EA_{I\&D}$ and integrative HMH-$EA_{I\&D}$, derive from a well-known classification for hybrid MHs and, at present, they have a consolidated background of knowledge. The third one, less explored, concerns a strategy with which $EA_{I\&D}$ may help classical MHs to improve their behavior. In particular, it involves replacing some I&D components in MHs by customized $EA_{I\&D}$ (*evolutionary I&D components*) that develop the same work more effectively. In this line, our second objective is to present an instance of this novel approach in order to complement the overview and provide additional results and insights on the study of HMH-$EA_{I\&D}$. In particular, we propose an evolutionary perturbation technique for ILS, which is a micro-EA that effectively explores in the neighborhood of particular solutions.

The remainder of this article is organized as follows. In Section 2, we give an overview of the existing research on HMH-$EA_{I\&D}$. In Section 3, we propose an ILS model with evolutionary perturbation technique that allow us to illustrate the way new HMH-$EA_{I\&D}$ instances may be built by embedding evolutionary I&D components in MHs. In addition, the benefits of the proposal in comparison to other ILS algorithms proposed in the literature to deal with binary optimization problems are experimentally shown. Finally, in Section 4, we provide the main conclusions of this work and examine future research lines. In Appendix A, we describe the features of the test suite used for experiments, in Appendix B, we explain the statistical test that was used for the experimental study, and finally, in Appendix C, we enclose a table with results of the algorithms.

2. Review of HMH-$EA_{I\&D}$

Nowadays, different authors have emphasized the need for *hybridization* of EAs with other optimization algorithms, machine learning techniques, MHs, etc. [12,33,82,90]. Some of the possible reasons for hybridization are [33,90]: (1) to improve the performance of EAs, (2) to improve the quality of the solutions obtained by EAs, and (3) to incorporate the EA as part of a larger system. This paper concerns mainly the last point and, in particular, those hybrid MHs that include a specific kind of EAs, $EA_{I\&D}$, which are characterized by their explicit trends to produce intensification and/or diversification, i.e., HMH-$EA_{I\&D}$.

In this section, we review different instances of HMH-$EA_{I\&D}$ presented in the literature. In order to do this, we have grouped them into three categories. The first two groups were derived from a well-known existing taxonomy for hybrid MHs [83], which was made on the basis of their control strategy:

- *Collaborative hybrid MHs*: They are based on the exchange of information between different MHs (and possibly other optimization techniques) running sequentially or in parallel.
- *Integrative hybrid MHs*: In this case, one algorithm is considered a subordinate, embedded component of another algorithm.

Many instances of both types of hybrid MHs were built including several $EA_{I\&D}$ as components. We denominate them, *collaborative* HMH-$EA_{I\&D}$ (Section 2.1) and *integrative* HMH-$EA_{I\&D}$ (Section 2.2), respectively.

We have identified a third category of HMH-$EA_{I\&D}$, which is related to an innovative method to improve classical MHs by means of $EA_{I\&D}$. It consists in transforming classical MHs into integrative HMH-$EA_{I\&D}$ by replacing particular I&D components by *customized* $EA_{I\&D}$, carrying out the same operation more effectively (Section 2.3).

2.1. *Collaborative* HMH-$EA_{I\&D}$

Collaborative hybrid MHs apply different self-contained MHs that exchange information about the search process (solutions, parameters, etc.) between them [20]. In this paper, we are interested on those models where some of these MHs are $EA_{I\&D}$ (Fig. 1). According to the way MHs are executed, collaborative HMH-$EA_{I\&D}$ can be subdivided into teamwork or relay ones [93], which are described in Sections 2.1.1 and 2.1.2, respectively.

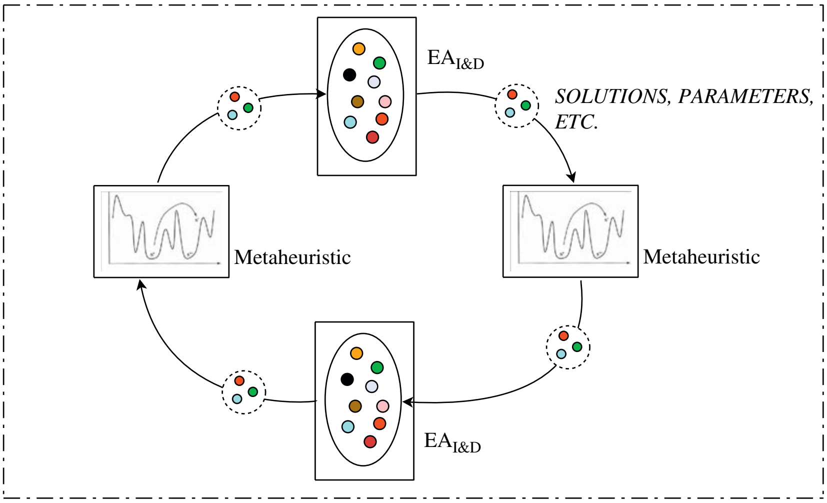

Fig. 1. Collaborative HMH-$EA_{I\&D}$.

2.1.1. *Teamwork collaborative* HMH-$EA_{I\&D}$

In *teamwork collaborative* HMH-$EA_{I\&D}$, there are several MHs, including some $EA_{I\&D}$, which work in parallel. Most of the initially proposed collaborative HMH-$EA_{I\&D}$ were based on *distributed GAs*. The basic idea of these algorithms lies in the partition of the population into several subpopulations, each one of them being processed by a GA, independently from the others [1,43]. Furthermore, a migration mechanism produces a chromosome exchange between the subpopulations. Several authors have pointed out the interest on differentiating the subpopulations by specializing the associated GAs in intensification or diversification (i.e., to use $EA_{I\&D}$). This is carried out by means of the application of different parameter values, population sizes, genetic operators, etc. This class of collaborative HMH-$EA_{I\&D}$ is also known as *heterogeneous distributed GAs*. Next, we review some of them.

Potts et al. [81] proposed a distributed GA based on binary coding, called GAMAS. GAMAS uses four subpopulations, denoted as *species* I, II, III, and IV. Initially, species II, III, and IV are created. Species II is an EA_D that uses a high mutation probability ($p_m = 0.05$). Species IV is an EA_I with low mutation probability ($p_m = 0.003$). Species III is a subpopulation for I&D, the mutation probability falls between the other two ($p_m = 0.005$). GAMAS selects the best individuals from species II, III, and IV, and introduces them into species I whenever those are better than the individuals in this subpopulation. The mission of species I is to preserve the best chromosomes appearing in the other species. At predetermined generations, its chromosomes are reintroduced into species IV, by replacing all the current elements in this species.

Tsutsui et al. [100] combine an *explorer* subpopulation with an *exploiter* one. The former is an EA_D using a coarse-grained mutation, whereas the second one is an EA_I aimed at exploiting fit local areas of the search space around the neighborhood of the best solution so far. Intensification is increased in the exploiter subpopulation by using a fine-grained mutation and a reduced population size.

Herrera et al. [43] propose *gradual distributed real-coded GAs*. They are a class of heterogeneous distributed GAs based on real coding in which subpopulations are distinguished by applying crossover operators with different degrees of exploration or exploitation. So, a parallel *multiresolution* is obtained with regard to the crossover operator, which allows a spread search along with an effective local tuning to be simultaneously achieved.

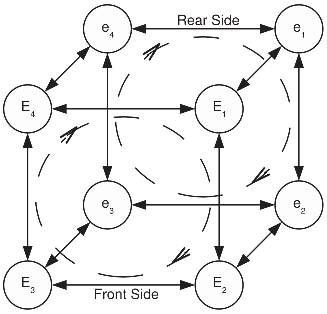

Fig. 2. Gradual distributed real-coded GAs.

Fig. 2 outlines the basic structure of the gradual distributed GAs. They are based on a hypercube topology with three dimensions with two important sides to be differentiated:

- The *front side* is devoted to exploration. It is made up of four subpopulations (EA_D), $E_1, \ldots, E_4$, to which exploratory crossover operators are applied. The exploration degree increases clockwise, starting at the lowest, E_1, and ending at the highest, E_4.
- The *rear side* is for exploitation. It is composed of four subpopulations (EA_I), $e_1, \ldots, e_4$, that undergo exploitative crossover operators. The exploitation degree increases clockwise, starting at the lowest, e_1, and finishing at the highest, e_4.

Furthermore, $EA_{I\&D}$ are adequately connected for exploiting the multiresolution in a *gradual* way, offering the refinement or the expansion of the best zones emerging. Migrations between subpopulations belonging to different categories produce these final effects:

- *Refinement*: It may be induced if migrations are produced from an exploratory subpopulation toward an exploitative one, i.e., from E_i to e_i, or between two exploratory subpopulations from a higher degree to a lower one, i.e., from E_{i+1} to E_i, or between two exploitative subpopulations from a lower degree to a higher one, i.e., from e_i to e_{i+1}.
- *Expansion*: In the case of migrations in the opposite direction, the chromosomes included may be reference points for generating diversity (with different degrees) on zones showing promising properties.

Experimental results showed that gradual distributed GAs consistently outperform sequential real-coded GAs, homogeneous

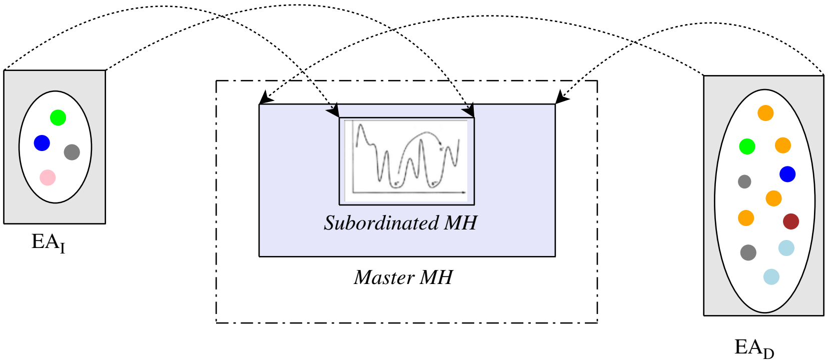

Fig. 3. Integrative HMH-$EA_{I\&D}$.

distributed real-coded GAs, and other EAs presented in the literature for dealing with continuous optimization problems.

Schlierkamp-Voosen et al. [87] present a heterogeneous distributed GA model that integrates different $EA_{I\&D}$ whose population sizes are dynamically adjusted during the course of evolving a solution. In particular, $EA_{I\&D}$ compete each other, in such a way that they gain or lose individuals depending on their *evolution quality* in relation to the others. A particular instance based on real-coding was proposed with four $EA_{I\&D}$. They were distinguished by applying a mutation operator with different step sizes (proportion or strength in which genes are mutated), which allows a search with *multiresolution* to be achieved.

So far, the collaborative HMH-$EA_{I\&D}$ reviewed combine uniquely specialized GAs. Recently, a different approach, called COSEARCH [94], has been presented, which applies, in parallel, three different kinds of MHs: an EA_D, a tabu search, and a local search procedure (LS). The three search algorithms communicate and cooperate via an *adaptive memory* that contains a history of the search already done, focusing on high quality regions of the search space. In COSEARCH, the EA_D is a GA in charge of sampling individuals in unexplored regions with regards to the content of the memory. To carry out this task, this EA_D uses a modified fitness function that favors individuals not represented in the adaptive memory. An evaluation of COSEARCH was carried out on the *quadratic assignment problem*. The comparison with a multiple tabu search showed that the cooperation increases significantly the efficiency of the method. COSEARCH yields the best known solutions for almost all tested instances. This study revealed, as well, that COSEARCH can be used as a model to design an efficient and robust method for solving combinatorial optimization problems.

Finally, we point out that teamwork HMH-$EA_{I\&D}$ are able to be executed on parallel hardware, taking advantage of this technology [4]. For example, in [2], a performance analysis of an implementation of gradual distributed GAs on parallel hardware is conducted.

2.1.2. *Relay collaborative* HMH-$EA_{I\&D}$

In *relay collaborative* HMH-$EA_{I\&D}$, an $EA_{I\&D}$ is executed, in a pipeline fashion, with another $EA_{I\&D}$ or other type of MH. The output of each algorithm is supplied as input to the next one. Most instances of relay collaborative HMH-$EA_{I\&D}$ follow the heuristic: "to protect the exploration in the initial stages and the exploitation later" (this heuristic was considered to design classical MHs, such as simulated annealing). Next, we explain two instances found in the literature.

Chelouah et al. [14] present a hybrid MH that comprises two main stages. The first stage involves the run of a specialized GA for diversification, i.e., an EA_D. The second stage involves the application of an LS process to the best individual found by the GA, with the objective of obtaining an accurate final solution. The EA_D was a GA that employs a large population, and a high mutation probability, to homogeneously cover the whole search space, and detect a promising area. It stops when one of the following conditions is reached: (1) a given number of successive generations without detection of a promising area is reached and (2) a given accuracy relating to the individuals in the population is obtained, e.g., the highest distance between the best individual found so far and the generated individuals is smaller than a given neighborhood radius. For continuous test functions having less than 10 variables, the authors obtained similar or better results than the ones supplied by other methods or other continuous GAs, but with a cheaper computational cost.

García-Martínez et al. [26] propose a procedure that determines *female* and *male* parents in the population of real-coded GAs that apply *parent-centric* real-parameter crossover operators. In general, these operators use a probability distribution to create offspring in a restricted search space around the region marked by one of the parent, the female parent. The range of this probability distribution depends on the distance among the female parent and the other parent involved in the crossover, the male parent. The female and male differentiation process proposed by these authors makes possible the design of *global real-coded* GAs (i.e., EA_D) and *local real-coded* GAs (i.e., EA_I), which are differentiated according to the considered number of female members. Furthermore, they combine these GA models in the following way: first, they run the global real-coded GA during a determinate percentage of the available evaluations, and then, they perform the local real-coded GA. The initial population for the local algorithm included the best individuals in the final population of the global one. This relay collaborative HMH-$EA_{I\&D}$ resulted very competitive with state-of-the-art on MHs for continuous optimization problems.

2.2. *Integrative* HMH-$EA_{I\&D}$

In *integrative hybrid MHs*, one MH (subordinate) becomes a component of another MH (master). *Memetic algorithms* (MAs) [57] are well-known instances of this class of algorithms. MAs combine an EA in charge of the global search with an LS procedure, which is executed within the EA run, looking for a synergy that takes benefits from both. The classic scheme of MAs applies the LS procedure on the solutions obtained by the EA with the aim of improving the accuracy of the population members. However, MAs also include combinations of EAs with problem-dependent heuristics, approximate algorithms, truncated exact methods, specialized recombination operators, etc. [68].

There are several instances of integrative HMH-$EA_{I\&D}$, where either the master MH or the subordinate one is an $EA_{I\&D}$. Most of them are MAs with $EA_{I\&D}$ (Fig. 3). In this case, the component in charge of the global search is an EA_D (Section 2.2.1) and/or the task of refining solutions is accomplished by an EA_I (Section 2.2.2).

2.2.1. *MAs with* EA_D

From initial studies on MAs, researchers noticed the convenience of strengthening diversification capacity for the master EA (in order to counteract high intensification provided by LS operator), i.e., to turn it into an EA_D. Below, we review some approaches.

Nagata et al. [72] propose to use populations for the master EA a couple of orders of magnitude bigger than those used by standard MAs. Besides, they use a specific crossover operator that generates a wide offspring variety. Both features increase diversity in the master EA. The resulting algorithm was tested on 21 symmetric and nine asymmetric instances of the traveling salesman problem, succeeding in finding optimal solution to all problems efficiently.

Distributed GAs attempt to overcome premature convergence by means of the preservation of diversity (due to the *semi-isolation* of the subpopulations). Different authors build MA models combining these EA_D with LS procedures. The model presented in [69] uses several subpopulations to locate different promising search regions. Then, hill-climbing is done. Good local minima of a subpopulation are diffused to neighboring subpopulations. Authors tested the proposal on eight continuous optimization problems and performed a speedup study obtaining promising results. In [96], a distributed MA that adapts the application of using the LS procedure is proposed. It uses the online entropy of population to obtain dynamic information about the stage of the evolutionary search process and the degree of diversity of each subpopulation. Then, it computes the number of individuals that should undergo LS according to this measure. Authors performed experiments on several large scale quadratic assignment problem benchmarks, and compared the results of the proposal with the ones of other distributed GAs and MAs. They concluded that the presented algorithm shows the ability of producing competitive solutions at significantly less computational cost.

Merz [65] shows many different combinations of LS and GAs for the traveling salesman problem while defining specific purpose crossover and mutation operators. One of them is the DPX crossover that was specifically designed to preserve diversity by means of keeping constant the appropriately defined hamming distance between the two parent tours and the generated offspring. In addition, a restart technique is employed. Clearly, the use of DPX and the restart technique make the GA to have trends for diversification. The resulting algorithm was tested on several instances contained in TSPLIB. The results showed that, with regards to other proposed approaches, MA with DPX appears to be superior in average solution quality and running times.

Lozano et al. [61] suggest employing two mechanisms to promote population diversity of EA component of real-coded MAs: *negative assortative mating* [22] and *BGA mutation* [70]. The former determines the way chromosomes are mated for applying crossover to them. A first parent is randomly selected and n_{ass} chromosomes are selected with the same method. Then, similarity between each of these chromosomes and the first parent is computed. The one with less similarity is chosen. Clearly, negative assortative mating increases genetic diversity in the population by mating dissimilar genomes with higher probability. BGA is a mutation operator specializing in continuous optimization problems. In contrast to some mutation operators that intend to make an LS at the last stages of the search process, BGA continuously provides acceptable levels of diversity to the EA. The application of these two components allows the EA to be specialized in diversification. The final proposal was tested on eight continuous optimization problems and its performance was compared with the one of other GAs and MAs obtaining better results.

Seront et al. [88] present an MA with a clustering method that reduces the total cost of LS by avoiding the multiple rediscoveries of local optima. Solutions belonging to a basin of attraction are detected by the clustering algorithm and only one LS method is started in each basin of attraction. In addition, the clustering method supplies information that can be used to maintain the diversity in the population. The proposal applies a two level selection strategy. At the local level, intra-cluster selection is applied on each cluster independently. Each cluster is treated as a separate population. Only the worst solution of each cluster is replaced by the local optima to avoid concentration of all solutions of the cluster toward a single point. At the global level, selection takes as fitness for a solution x belonging to a cluster C, the fitness of the local optima detected inside C. Global selection tries to avoid the convergence of the GA toward a single cluster. Kemenade [52] presents an MA model based on evolution strategies that capture similar ideas. This MA applies a clustering process preventing premature convergence. It uses a two-stage selection process. During the first stage, a subset of the complete population, containing the best individuals, is selected. During the second one, a clustering process is applied to the remaining individuals, and the best individual of each cluster is selected as a representative of that cluster. A new population is created by applying the evolutionary operators to these representatives only. Local optimization is applied to the representatives only. Applying local optimization to the set of representatives reduces the amount of computation required and decreases the probability of locating the same local optimum multiple times.

Parthasarathy et al. [77] and Wei et al. [101] address the issue of handling explicitly multimodal functions using MAs. In [77], the adaptive niching method via *coevolutionary sharing* of Goldberg et al. [32] is applied to stably maintain a diverse population throughout the search. This method divides the population into *businessmen* and *customers*. The former represent a list of attractors (local optima) of the problem, and customers model their basins of attraction. Each customer usually belongs to the closest businessman. Businessmen derive their fitness from their customers and customers share their fitness among other customers belonging to the same businessman. A minimum distance between businessmen is kept throughout the run. The combination of this EA_D with an LS procedure was tested on two instances of structural design problems obtaining better solutions than the ones presented in the literature. In [101], an MA is presented that applies a *clearing* procedure [80] for the same purpose. Offspring population is merged with the N best previous individuals. Then, normalized *Euclidean* distance between every pair of individuals is calculated. If this distance is inferior to a *niche radius*, the individual with lower fitness is punished reducing its fitness value to 0. This EA_D is combined with *Nelder–Mead's simplex* LS procedure to tackle continuous multimodal optimization problems. Authors claim that the proposed method alleviates premature convergence and improves weak exploitation capacities of GAs. The results indicate that, the model may locate the global optimum quickly, reliably and accurately.

Finally, we should cite the work by Krasnogor and Smith [56] where they introduce a hybridization scheme for an MA based on an adaptive helper that uses statistics from the GA population. Their MA is composed of two optimization processes, a GA and a helper that is a Monte Carlo method, which serves two purposes. First, when the population is diverse, it acts like an LS procedure and second, when the population converges, its goal is to diversify the search. Authors performed experiments on traveling salesman and protein folding problems. It was observed that the proposed approach was able to fine tuning the global search and diversifying the population, which lead to better solutions than those obtained by a standard GA and two other MAs.

2.2.2. *MAs with* EA_I

The use of EA_I models in MAs as refinement procedures is another field of increasing interest to create integrative HMH-$EA_{I\&D}$. With this idea, there have been presented several MAs that use μGAs (GA with a small population and short evolution) to refine the members of the population [49,61,73]. μGA models present some advantages over classic LS procedures. Most LS techniques lack the ability to follow the proper path to the optimum on complex search spaces. This difficulty becomes much more evident when the search space contains very narrow paths of arbitrary direction, also known as *ridges*. That is due to most LS techniques attempt successive steps along orthogonal directions that do not necessarily coincide with the direction of the ridge. However, it was observed that μGAs are capable of following ridges of arbitrary direction in the search space regardless of their direction, width, or even, discontinuities [49]. Next, we describe some recent MA models that employ μGAs as LS operators.

- The μGA presented in [49] is a GA with five individuals that encode perturbations. Aptitude values of these individuals depend on a solution given by the master EA. This feature ensures that search is focused in the neighborhood of the given solution, whereas low sized population promotes high selection pressure levels. The proposed MA based on μGA was tested against 12 different EA models, which include a simple GA and MAs with different hill-climbing operators, on five hard constrained optimization problems. Simulation results revealed that this algorithm exhibits good performance, surpassing competing algorithms in all test cases with regard to solution accuracy, feasibility rate, and robustness. We shall say that μGA has also been successfully applied in combination with multi-agent GAs [102], multi-objective EAs [64], and GAs for fuzzy modeling [76].
- The model proposed in [61] is a *crossover hill-climbing operator* (XHC) that is specifically designed to tackle continuous optimization problems. XHC is a micro-selecto-recombinative real-coded GA that maintains a pair of parents (the solution being refined and the best solution found so far) and performs repeatedly crossover on this pair until some number of offspring is reached. Then, the best offspring is selected and replaces the worst parent, only if it is better. The key idea in XHC is to take advantage of the self-adaptive ability of some crossover operators for real coding, which sample offspring according to the parent distribution without any adaptive parameter, to induce an effective local tuning. Experimental results showed that, for a wide range of problems, the real-coded MA with XHC operator consistently outperformed other real-coded MAs appeared in the literature.
- Other researchers have extended this EA_I model. In particular, in [73], an adaptive XHC, which adaptively adjusts the length of the refinement process, is proposed for MAs based on differential evolution. This EA_I model performs iterations while offspring outperforms the first parent. As soon as this condition is not fulfilled, control returns to differential evolution. Authors performed experiments on twenty continuous functions. The performance of the proposed EA_I was better than the one of other crossover-based LS strategies, and the overall performance of the hybrid model was superior to or at least competitive with some other MAs selected from literature.
- Soak et al. [91] present another MA model taking some ideas from particle swarm optimization algorithms. They apply two specific crossover operators for the degree constrained minimum spanning tree problem. One of them shows strong local capacities and was applied to a population with two individuals. Another one was employed to recombine elitist solutions only. Both operators specialize the EA for intensification. On the other hand, master EA performs mutation and updates elitist solutions according to the ideas of particle swarm optimization algorithms. The final algorithm was tested on several instances of the degree constrained minimum spanning tree problem, obtaining promising results when compared with the ones of the state of the art edge window decoder, edge set encoding, network random keys encoding and *Prüfer* number encoding.
- Mutoh et al. [71] proposes a multi-step crossover model to perform LS. It consists in repeating the crossover operation with best offspring, a number of times. First, offspring generated by the first parents are evaluated for their fitness. Then, a number of top offspring with an elite rate set beforehand are selected as the next parents. These operations are repeated for several steps. Authors also propose a set of rules adapting the number of steps this EA_I performs. Then, this model was included in a master EA performing selection on a set of solutions. Empirical studies, on four continuous optimization problems, were carried out in order to analyze the benefits of applying adaptive multi-step crossover with regards to the application of this operator with a fixed number of steps. Results showed that the proposed model could obtain an optimal solution faster than the conventional model.

Other EA_I approaches, following some other ideas different from μGAs, have been proposed as subordinate algorithms for MAs. In [25], an interesting MA model is presented for the traveling salesman problem. In this case, the subordinate EA_I is aimed at refining partial *subtours* within the solutions of the master EA. A set of continuous cities are chosen to create the subtour. The EA_I is applied to the subtour, for finding the local optimal solution to replace the original part chosen from the main tour. The subtour is an open tour. The length of the tour is calculated from the start city to the end city, not including the distance between the end city and the start city. Another important point is that all individuals in the population of the EA_I have the same start and end cities during the processing. This imperative is for avoiding the main tour becoming longer at the connection points after the recombination of the subtour. At the end of the EA_I processing, the length of the best individual is computed. If the length is shorter than the one of the original subtour, it is recombined back into the main tour to replace the original part. The proposal was tested on a simple instance of the double concentric circle. Good results were obtained when the number of cities of the subtour was set to around half the number of cities of the main tour.

Nowadays, there are promising progresses on the design of EA_I models based on *estimation distribution algorithms*. In particular, in [86], an LS procedure using a competent neighborhood structure to search in the building-block space of the tackled problem is proposed. First a linkage-group identification procedure is applied on the best members of the population. Then, a mutation operator tries every combination of values for each group of linked variables. At the end, best combinations are retained. Authors derived an analytical bound and empirically verified the scalability of the proposed model on boundedly difficult additively separable problems. The results showed that the proposal successfully solves hard problems, requiring only subquadratic number of evaluations. Then, authors suggests to integrate this EA_I-based LS procedure in a master estimation distribution algorithm in charge of global search and updating the linkage-group identification. In [60], a similar procedure for hierarchical problems, where building-blocks overlap themselves, is described. The proposal consists in the combination of *Bayesian optimization algorithm* and a hill-climbing operator using the obtained linkage model. Hill-climbing is performed for a proportion of the population to speedup convergence to good solutions. In each iteration, Bayesian optimization algorithm is applied to capture the (in)dependencies between the variables of the problem. Then, variables are considered according to the ancestral reverse ordering of variables in the obtained Bayesian network, taking the values associated with the maximal substructural fitness. Finally, the control returns to Bayesian optimization algorithm to continue the run. The results showed that incorporating the proposed hill-climbing operator in Bayesian optimization algorithm leads to a significant reduction in the number of generations necessary to solve the problem, while providing substantial speedups in terms of number of evaluations.

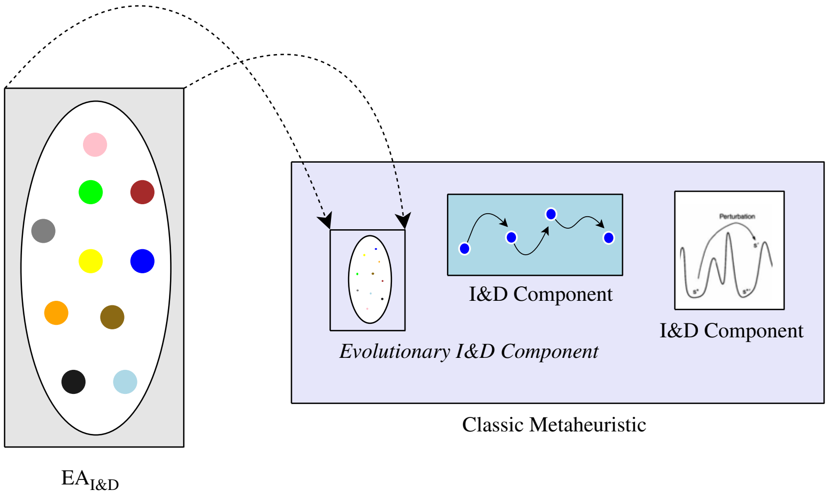

Fig. 4. MHs with evolutionary I&D components.

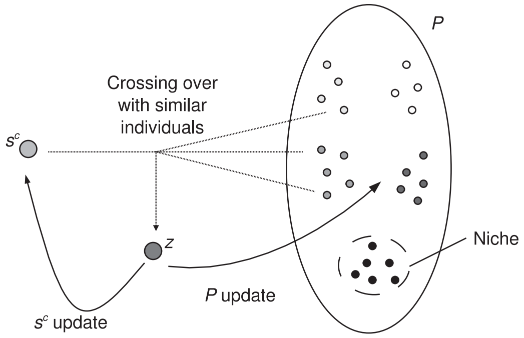

Fig. 5. General schema of BLGA.

2.3. MHs with evolutionary I&D components

We have identified a third method to build HMH-EA$_{I\&D}$, which concerns the incorporation of EA$_{I\&D}$ into classical MHs, with the aim of replacing determinate I&D components. The idea is to build *customized* EA$_{I\&D}$ playing the same role as particular I&D components, but more satisfactorily, i.e., *evolutionary I&D components* (Fig. 4).

In this way, we transform a classical MH into an integrative HMH-EA$_{I\&D}$ (because one of its components is another MH). This prospective line of research was not very explored in the past. We have only found two examples in the literature. They are two instances of multi-start LS that incorporate recent EA$_I$ as LS procedures, the *binary local GA* (BLGA) [27] and the *covariance matrix adaptation evolution strategy* (CMA-ES) [35,36].

BLGA is a steady-state GA that inserts one single new member into the population (P) in each iteration. It uses a *crowding* replacement method in order to favor the formation of *niches* in P (groups of chromosomes with high quality located in different and scattered regions of the search space). BLGA performs LS by orientating the search in the nearest niches to an external chromosome, the leader chromosome (S^c). In particular, it iteratively crosses S^c with individuals of the population belonging to the nearest niches, and then, the best solution between S^c and the offspring becomes the new leader solution, whereas the other one is inserted in the population by means of the crowding method (see Fig. 5).

An outstanding feature of BLGA is that it performs LS by describing a trajectory in the search space, as classical LS procedures do. Most LS algorithms follow a hill-climbing paradigm; they commence from a single solution and, at each step, a candidate solution is generated using a move operator of some sort. They simply move the search from the current solution to a candidate solution if the candidate has better fitness. The basic idea of BLGA is to use hill-climbing as the move accepting criterion of the search and crossover as the move operator. This scheme of LS based on crossover was first suggested by Jones [48] and O'Reilly et al. [75], and it has been followed to obtain different EA$_I$ approaches (Section 2.2.2). The main novelty of BLGA concerns the acquisition of information about the location of the best search regions (by favoring the formation of niches), which is then employed to generate individuals around S^c by means of the crossover operator. A multi-start LS was implemented using this algorithm as LS operator, which consistently outperformed other instances of this MH incorporating classic LS procedures from the literature.

CMA-ES was originally introduced to improve the LS performance of evolution strategies. Although CMA-ES even reveals competitive global search performances [38], it has exhibited effective abilities for the local tuning of solutions (it is extremely good at detecting and exploiting local structure in continuous optimization problems); in fact, it was used as continuous LS algorithm of an instance of multi-start LS, which was called L-CMA-ES [5]. At the 2005 congress of evolutionary computation, L-CMA-ES was one of the winners of the real-parameter optimization competition [37].

In CMA-ES, not only is the step size of the mutation operator adjusted at each generation, but so too is the step direction in the multidimensional problem space, i.e., not only is there a mutation strength per dimension but their combined update is controlled by a covariance matrix whose elements are updated as the search proceeds. (μ_W, λ) CMA-ES is the most commonly utilized CMA-ES model. For every generation, this algorithm generates a population of λ offspring by sampling a multivariate normal distribution:

$$x_i \sim N(m, \sigma^2 C) = m + \sigma N_i(0, C) \quad \text{for } i = 1, \ldots, \lambda,$$

where the mean vector m represents the favorite solution at present, the so-called step-size σ controls the step length, and the covariance matrix C determines the shape of the distribution ellipsoid. Then, the μ best offspring are recombined into the new mean value using a *weighted intermediate recombination*: $\sum_{i=1}^{\mu} w_i x_{i:\lambda}$, where the positive weights sum to one. The covariance matrix and the step-size are updated as well following equations that may be found in [36,38]. The default strategy parameters are given in [38]. Only

```
Iterated Local Search
    1. $S_0 \leftarrow$ Generate Initial Solution ()
    2. $S^* \leftarrow$ LocalSearch ($S_0$)
    3. while (termination conditions not met) do
        4. $S_P \leftarrow$ Perturbation ($\sigma_p$, $S^*$, history)
        5. $S_{LS} \leftarrow$ LocalSearch ($S_P$)
        6. $S^* \leftarrow$ AcceptanceCriterion ($S^*$, $S_{LS}$, history)
    7. return $S^*$
```

Fig. 6. Pseudocode algorithm for ILS.

the initial m and σ parameters have to be set depending on the problem.

Hansen et al. [36] interpret any evolution strategy that uses intermediate recombination as an LS strategy. CMA-ES employs intermediate recombination to create a single parent based on the average position of the current population. The next generation of offspring is based on a mutation distribution that surrounds this single parent. Once the initial mutation distribution has decreased, this variation of the traditional evolution strategy will behave much like an LS algorithm. Thus, since CMA-ES is extremely good at detecting and exploiting local structure, it turns out to be a particularly reliable and highly competitive EA for local optimization [5]. In fact, in order to obtain an *advanced* continuous LS algorithm, the LS characteristics of CMA-ES may be stressed by tuning some of its strategy parameters. For example, [5] recommended using a 100 times smaller initial step-size than is recommended as default and sticking to the default population size (between 10 and 15 times the search space dimensions). A different approach to enhance the LS abilities of this algorithm was introduced in [67], where it was used as LS operator of an MA.

Finally, we should point out that, in the next section, we analyze the performance of an ILS model that incorporates an evolutionary perturbation technique. The main objective is to investigate the potential of the method described in this section to obtain new HMH-EA$_{\text{I\&D}}$ and show how customized EA$_{\text{I\&D}}$ may help classical MHs to improve their behavior.

3. ILS with evolutionary perturbation technique

ILS [45,63] belongs to the group of MHs that extend classical LS methods by adding diversification capabilities. The essential idea of ILS is to perform a biased, randomized walk in the space of locally optimal solutions instead of sampling the space of all possible candidate solutions. This walk is built by iteratively applying first a *perturbation* to a locally optimal solution, then applying an LS algorithm, and finally using an acceptance criterion which determines to which locally optimal solution the next perturbation is applied. Despite its simplicity, it is at the basis of several state-of-the-art algorithms for real-world problems [16,17,45,85,95,99].

A high level description of ILS as it is described in [63] is given in Fig. 6. The algorithm starts by applying LS to an initial solution and iterates a procedure where a perturbation is applied to the current solution S^* in order to move it away from its local neighborhood; the solution so obtained is then considered as initial point for a new LS processing, resulting in another locally optimal solution S_{LS}. Then, a decision is made between S^* and S_{LS} to decide from which solution the next iteration continues.

The perturbation operator is a key aspect to consider, because it allows ILS to reach a new solution from the set of local optima by escaping from basis of attraction of the previous local optimum. The perturbation is usually non-deterministic in order to avoid cycling. For example, for the case of binary problems, the perturbation operator flips the bits with a fixed probability. Its most important characteristic is the perturbation strength (σ_p), roughly defined as the amount of changes made on the current solution. The perturbation strength should be large enough such that the LS does not return to the same local optimum in the next iteration. However, it should not be too large; otherwise the search characteristics will resemble those of a multi-start LS algorithm.

The history component in *Perturbation* and *AcceptanceCriterion* (Steps 4 and 6 in Fig. 6) indicates that also the search history may influence the decisions made in these procedures. Yet, often *Markovian* implementations of ILS are applied, i.e., the output of *Perturbation* and *AcceptanceCriterion* is independent of the search history [92].

An important aspect in the perturbation and the acceptance criterion is to introduce a bias between I&D of the search. Intensification can be reached by applying the perturbation always to the best solution found and using small perturbations. On the other hand, diversification is achieved by accepting every new solution S^* and applying large perturbations. Then, the perturbation operator arises as one of the most determinant *I&D* component of ILS. In addition, it is a key aspect to consider in the design of ILS; as claimed by [63]: "A good perturbation transforms one excellent solution into an excellent starting point for a LS".

Another determinant aspect of ILS is the mechanism to perform perturbations. This may be a random mechanism (as was aforementioned), or it may be produced by a semi-deterministic method (e.g., an LS different from the one used in the main algorithm [11]). In this respect, we should point out that different EA principles have been used to build new perturbation models. Examples are:

- *Population-based ILS* (PILS): Thierens [98] proposes an MH that combines the power of ILS with the principle of extracting useful information about the search space by keeping a population of solutions. In addition to ILS, PILS also keeps a small population of neighboring solutions and restricts the perturbation of ILS to the subspace where the current solution and a population member disagree, thus preserving their common substructure. The key assumption of the PILS algorithm is that local optimal solutions possess common substructures that can be exploited to increase the efficiency of ILS.
- *Genetic ILS* (GILS): Katayama et al. [50,51] introduced a new perturbation mechanism for an ILS instance developed for the traveling salesman problem, which was called GILS. This perturbation approach uses a crossover operator specifically designed to deal with this problem. In each iteration, GILS perturbs the best found solution, S_{best}, generating S_P. Then, it applies the crossover operator to S_{best} and S_P, producing the final solution that will be refined by the LS operator.
- *ILS with guided mutation* (ILS/GM): Zhang et al. [106] used the *guided mutation operator* [105] as perturbation operator in ILS for the quadratic assignment problem. Guided mutation uses the idea of estimation of distribution algorithms to improve conventional mutation operators. It provides a mechanism for combining global statistical information about the search space and the position information of a good solution found during the previous search for generating new trial solution.

In this section, we present an *evolutionary ILS-perturbation mechanism* that involves a micro-CHC algorithm that explores the neighborhood of the solution that should undergo the perturbation operator. First, in Section 3.1, we describe the CHC algorithm, then, in Section 3.2, we explain our proposal, and finally, in Section 3.3, we report the results from an empirical analysis carried out on this new ILS approach.

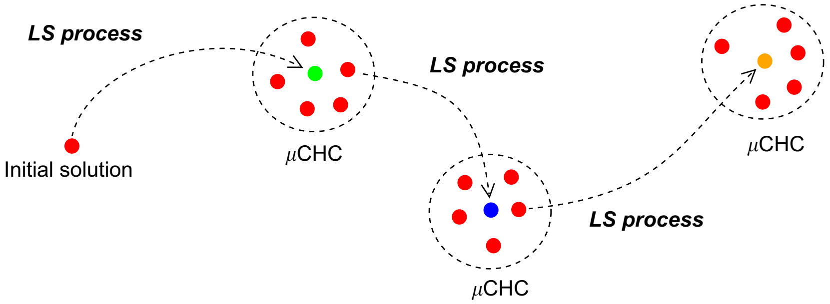

Fig. 7. ILS with evolutionary perturbation technique.

3.1. CHC algorithm

The key idea of the CHC algorithm (*Crossgenerational elitist selection, Heterogeneous recombination, and Cataclysmic mutation*) [21] concerns the combination of a selection strategy with a very high selective pressure and several components inducing a strong diversity. The four main components of the algorithm are shown as follows:

- *Elitist selection*: The N members of the current population are merged with the offspring population obtained from it and the best N individuals are selected to compose the new population. In case that a parent and an offspring have the same fitness value, the former is preferred to the latter.
- *Half uniform crossover*: It is a highly disruptive crossover that crosses over exactly half of the non-matching alleles (the bits to be exchanged are chosen at random without replacement). This way, it guarantees that the two offspring are always at the maximum Hamming distance from their two parents, thus proposing the introduction of a high diversity in the new population and lessening the risk of premature convergence.
- *Incest prevention mechanism*: During the reproduction step, each member of the parent (current) population is randomly chosen without replacement and paired for mating. However, not all these couples are allowed to crossover. Before mating, the Hamming distance between the potential parents is calculated and if half this distance does not exceed a difference threshold d, they are not mated and no offspring coming from them is included in the offspring population. The aforementioned threshold is usually initialized to $L/4$ (with L being the chromosome length). If no offspring is obtained in one generation, the difference threshold is decremented by one.
- *Cataclysmic mutation*: CHC uses no mutation in the classical sense of the concept, but instead, it goes through a process of cataclysmic mutation when the population has converged. The difference threshold is considered to measure the stagnation of the search, which happens when it has dropped to zero and several generations have been run without introducing any new individual in the population. Then, the population is reinitialized by considering the best individual as the first chromosome of the new population and generating the remaining $N-1$ ones by randomly flipping a number of its bits, determined by the cataclysmic mutation rate, p_{cm} (usually $p_{cm}=0.35$). After invoking the cataclysmic mutation, the difference threshold is reinitiated to $p_{cm}*(1-p_{cm})*L$.

The CHC algorithm was tested against different GA approaches, giving better results, especially on hard problems [103]. So, it has arisen as a reference point in the GA literature.

3.2. Evolutionary ILS-perturbation technique

As was clearly stated above, a promising research line that may be followed to improve ILS performance involves the utilization of different EA principles to design the perturbation mechanism. In this section, we present an evolutionary ILS-perturbation technique, which is based on the CHC algorithm. It will be denominated μCHC. Our main idea is to build a new ILS model, called ILS-μCHC, which follows pseudo-code in Fig. 6 replacing Step 4 by

4. $S_P \leftarrow \mu\text{CHC}(\sigma_p, S^*)$.

We have conceived μCHC to be an effective *explorer* in the neighborhood of S^*. At the beginning of this algorithm, S^* is used to create its initial population. Then, it is performed throughout a predetermined number of fitness function evaluations. The best reached individual is then considered as starting point for the next LS process (Fig. 7).

We have chosen the CHC algorithm as basis to build our evolutionary ILS-perturbation method because it suitably combines powerful diversification mechanisms with an elitist selection strategy. The filtering of high diversity by means of high selective pressure favors the creation of *useful diversity*; many dissimilar solutions are produced during the run and only the best ones are conserved in the population, allowing diverse and promising solutions to be maintained. From our point of view, this behavior is desirable for an EA assuming the work of a perturbation operator. Finally, we should point out that μCHC may be seen as an EA_D, because its main mission is to effectively explore the neighborhood of S^* by promoting useful diversity in this search region.

Next, we detail the main adaptations made on the original formulation of CHC to obtain our evolutionary ILS-perturbation technique:

1. *Population size*: μCHC manages a population with few individuals ($N=5$), and thus, it may be seen as micro-EA. In standard ILS models, the number of fitness function evaluations required by the perturbation mechanism is very low as compared with the one for the LS method. With the aim of preserving, as far as possible, the essence of ILS, we have considered an EA with a low sized population; for being able to work adequately under the requirement of spending reduced number of evaluations.
2. *Number of evaluations*: In particular, we have limited this number through the following strategy: the number of evaluations assigned to μCHC for a particular invocation will be a fixed proportion, p_{evals}, of the number of evaluations consumed by the previously performed LS method. It is worth noting that p_{evals} should be set to a low value.
3. *Initial population*: Every individual in the initial μCHC population is generated by performing standard perturbation on the current

solution, S^* (Fig. 6), using the perturbation strength σ_p (which becomes a parameter associated with μCHC).

4. *Cataclysmic mutation*: It fills the population with individuals created by the same way as initial population is built (by perturbing S^*) and preserves the best performing individual found in the previous evolution. After applying cataclysmic mutation, the difference threshold is set to $\sigma_p * (1-\sigma_p) * L$ (i.e., we have considered that $\sigma_p = p_{cm}$).
5. *Mating with S^**: Finally, we should highlight that μCHC incorporates the appealing principle in GILS [51] of recombining S^* with another solution. In addition to the typical recombination phase of CHC, our algorithm always mates S^* with an individual in the population (selected at random) and, if they are finally crossed over (attending on the incest prevention mechanism), the resulting offspring will be introduced into the offspring population of μCHC.

It is worth to mention that the first implementation of micro-EA was reported by Krishnakumar [58], who used a GA model with population size of five individuals, tournament selection, single-point crossover, elitism, and restart operator. He showed that his μGA could avoid premature convergence and performs better than a simple GA for selected multimodal problems. Specific micro-EA_l have been built to serve as LS operators for MAs (see Section 2.2.2). The novelty of our proposal involves the use of a micro-EA as diversification agent to replace the perturbation operator of ILS.

Finally, we may highlight that our proposal gathers together the idea in GILS of using a crossover operator with the one of PILS of managing a population of solutions. In this way, it attempts to combine the best of these methods.

3.3. Experiments

We have carried out experiments on a test suite composed by 19 binary optimization problems (Appendix A), in order to study the behavior of the ILS model based on the evolutionary perturbation mechanism presented in the previous section. Firstly, we detail the experimental setup and statistical method applied (Section 3.3.1), then, we analyze the results obtained from different experimental studies carried out with ILS-μCHC. In particular, our aim is: (1) to ascertain whether the innovative design of μCHC is suitable to allow this algorithm to outperform other ILS models with contemporary perturbation methods of the literature (Section 3.3.2), (2) to investigate the way the specific design of our algorithm may affect its performance (Section 3.3.3), (3) to compare its results with the ones of other ILS instances that were built with the specific objective of enhancing diversification (Section 3.3.4), and (4) to validate the innovative design for hybridizing an EA with an LS method that underlies in ILS-μCHC. In order to do this, we pit this algorithm against several MA instances, which represent the most widely accepted approach, at present, for combining these two ingredients (Section 3.3.5). The results of all executed ILS algorithms may be found in Appendix C.

3.3.1. Experimental setup and statistical method

In this section, we describe the basic scheme of the ILS algorithms (Fig. 6) compared in our experiments. They were specifically implemented to tackle optimization problems in a fixed-length binary search space:

- *LS procedure*: It is the *first-improvement hill-climbing algorithm*, which consists in having one individual and keep mutating each gene, one at a time, in a predefined random sequence, until the resulting individual is fitter than the original. In that case, the new individual replaces the original and the procedure is repeated until no improvement can be made further.
- *Initial solution*: It is a fixed-length binary string generated at random.
- *Acceptation criterion*: We have used the requirement that new solutions should have a better (or at least equal) fitness value than the current solution.

All the algorithms were executed 50 times (initial solutions were the same for the corresponding runs for all the ILS instances), each one with a maximum of 100,000 fitness function evaluations.

Non-parametric tests have been used for comparing the results of different search algorithms [46]. 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, which is, to compute the same aggregation (we have considered the average of the best fitness function found at the end of each run) over the same number of runs for each algorithm and problem. In particular, we have used the *Wilcoxon matched-pairs signed-ranks* test to compare the results of our proposal with the ones of other ILS approaches. We explain, with detail, this statistical test in Appendix B.

3.3.2. Comparison of μCHC with other perturbation methods

The main aim of this section is to compare μCHC with other contemporary perturbation techniques of the literature. In order to do this, we have implemented several ILS algorithms that follow the basic scheme described in the previous section and are distinguished uniquely by the perturbation operator:

- ILS with *standard binary-perturbation* (SILS): This perturbation method flips the bits with a fixed probability, the perturbation strength, σ_p.
- PILS [98]: The values for the parameters associated with the perturbation operator in this algorithm are $N=5$, $P_{ratio}=0.5$, and $P_{maskmut}=0.25$.
- GILS [51]: We have considered *uniform* crossover for implementing the perturbation strategy for this algorithm.
- ILS/GM [106]: We have implemented the guided mutation operator proposed in [105] to manipulate binary-coded chromosomes (β was set to 0.005). We used guidelines in [106] to adapt this operator as perturbation mechanism for ILS.

μCHC uses standard binary-perturbation for generating initial population and populations after applying cataclysmic mutation. In addition, it assumes $p_{evals}=0.25$, i.e., at each invocation, it consumes the 25% of number of evaluations utilized by the previous processing of LS procedure.

Since all the implemented ILS algorithms are distinguished uniquely by the perturbation policy, we may determine the significance of our newly proposed method. In order to make the comparison, firstly, we investigate the influence of σ_p on the performance of these ILS algorithms. In particular, we analyze the behavior of these algorithms when different values for this parameter are considered ($\sigma_p = 0.1$, 0.25, 0.5, and 0.75).

For each ILS algorithm, Fig. 8 shows the average ranking obtained by its instances with different σ_p values when compared among them. This measure is obtained by computing, for each problem, the ranking r_j of the observed results for instance j assigning to the best of them the ranking 1, and to the worst the ranking k (k is the number of instances). Then, an average measure is obtained from the rankings of this instance for all test problems. For example, if a certain instance achieves rankings 1, 3, 1, 4, and 2, on five test functions, the average ranking is $(1+3+1+4+2)/5=\frac{11}{5}$. Clearly, the lower a column is, the better its associated ILS instance is.

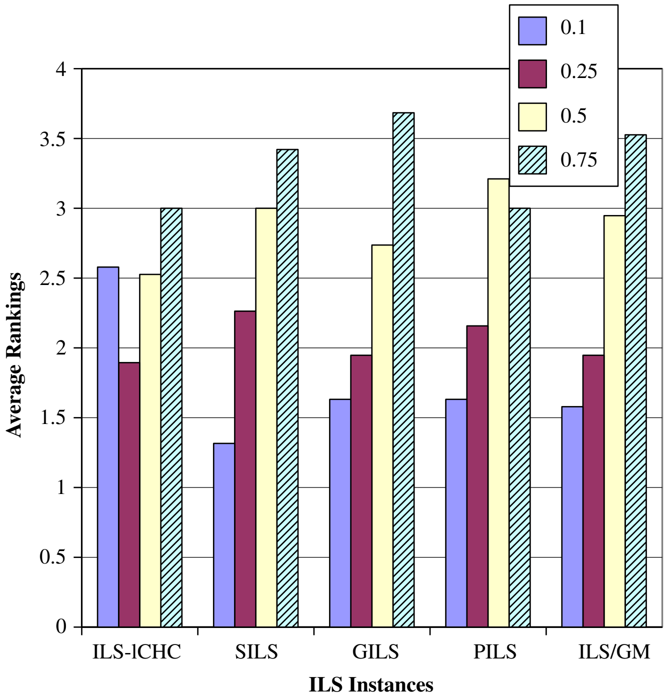

Fig. 8. Average rankings obtained by ILS instances with different σ_p values.

Table 1
ILS-μCHC ($\sigma_p = 0.25$) vs. ILS algorithms with other perturbation models (Wilcoxon's test with p-value $= 0.05$).

ILS-μCHC vs.	$R+$	$R-$	Critical value	Sig. differences?
SILS ($\sigma_p = 0.1$)	177.0	13.0	46	Yes
SILS ($\sigma_p = 0.25$)	186.0	4.0	46	Yes
SILS ($\sigma_p = 0.5$)	183.0	7.0	46	Yes
SILS ($\sigma_p = 0.75$)	171.0	19.0	46	Yes
PILS ($\sigma_p = 0.1$)	180.0	10.0	46	Yes
PILS ($\sigma_p = 0.25$)	186.0	4.0	46	Yes
PILS ($\sigma_p = 0.5$)	173.0	17.0	46	Yes
PILS ($\sigma_p = 0.75$)	164.0	26.0	46	Yes
GILS ($\sigma_p = 0.1$)	170.0	20.0	46	Yes
GILS ($\sigma_p = 0.25$)	182.0	8.0	46	Yes
GILS ($\sigma_p = 0.5$)	187.0	3.0	46	Yes
GILS ($\sigma_p = 0.75$)	186.0	4.0	46	Yes
ILS/GM ($\sigma_p = 0.1$)	167.0	23.0	46	Yes
ILS/GM ($\sigma_p = 0.25$)	185.0	5.0	46	Yes
ILS/GM ($\sigma_p = 0.5$)	187.0	3.0	46	Yes
ILS/GM ($\sigma_p = 0.75$)	185.0	5.0	46	Yes

An important remark from Fig. 8 is that the best ranked instance of ILS-μCHC uses $\sigma_p = 0.25$ while one for the other ILS algorithms employ $\sigma_p = 0.1$. This indicates that ILS-μCHC achieves its best behavior by working in more extensive neighborhoods than other ILS algorithms do, i.e., its performance becomes better by processing higher diversification levels than its competitors. Next, we investigate whether this mode of operating allows it to obtain better results. Then, we have undertaken a comparative analysis between ILS-μCHC with $\sigma_p = 0.25$ and each one of the other ILS algorithms (with all σ_p values) by means of Wilcoxon's test (Appendix B). Table 1 summarizes the results of this procedure, where the values of $R+$ (associated to ILS-μCHC) and $R-$ of the test are specified together with the critical values. Last column indicates whether our algorithm performs statistically equivalent to the other algorithm (the null hypothesis of equality of means is accepted) or there exist significant differences between them (the null hypothesis of equality of means is rejected).

Table 2
ILS-μCHC ($\sigma_p = 0.25$) vs. ILS-REx and ILS-μGA (Wilcoxon's test with p-value $= 0.05$ and 0.1).

ILS-μCHC vs.	$R+$	$R-$	Critical value ($p = 0.05/p = 0.1$)	Sig. differences? ($p = 0.05/p = 0.1$)
ILS-REx ($\sigma_p = 0.1$)	147.0	43.0	46/53	Yes/yes
ILS-REx ($\sigma_p = 0.25$)	166.0	24.0	46/53	Yes/yes
ILS-REx ($\sigma_p = 0.5$)	174.0	16.0	46/53	Yes/yes
ILS-REx ($\sigma_p = 0.75$)	163.0	27.0	46/53	Yes/yes
ILS-μGA ($\sigma_p = 0.1$)	142.0	48.0	46/53	No/yes
ILS-μGA ($\sigma_p = 0.25$)	153.0	37.0	46/53	Yes/yes
ILS-μGA ($\sigma_p = 0.5$)	170.0	20.0	46/53	Yes/yes
ILS-μGA ($\sigma_p = 0.75$)	160.0	30.0	46/53	Yes/yes

From Table 1, we clearly notice that ILS-μCHC obtained improvements with regards to the other algorithms, which are statistically significant (because all $R-$ values are lower than both $R+$ ones and critical values). These initial experiments suggest that our evolutionary ILS-perturbation technique may really enhance the operation of ILS and, thus, it becomes prospective for effectively exploring the neighborhood of S^*.

3.3.3. *Analyzing useful diversification of* μCHC

In the previous section, we have observed that the best instance of ILS-μCHC processes higher diversity levels than the ones managed by best instances of other ILS algorithms, even outperforming these ones. This fact shows that the inherent ability of μCHC to handle useful diversify (Section 3.2) becomes very profitable for the task of exploring the neighborhood of S^*. In this section, we attempt to corroborate this affirmation by comparing ILS-μCHC with alternative ILS approaches based on perturbation mechanisms that explore this region as well, but following other strategies:

- *ILS with perturbation based on random exploration* (ILS-REx): This perturbation method consists in generating n_r solutions by applying standard perturbation and selecting the best one as S_P. We should point out that this perturbation scheme was suggested in [34] for a particular type of ILS called VNS.
- *ILS with perturbation based on standard* μGA (ILS-μGA): This EA is a generational GA that uses a population size of five individuals, tournament selection, uniform crossover with probability $p_c = 1$, elitism, and no mutation [58]. The population was considered converged when less than 5% of the population bits were different from the bits of the best individual. The mechanisms to create initial population and populations after restarts are like the ones for μCHC.

The number of fitness function evaluations assigned to these two perturbation methods is computed as for μCHC (25% of number of evaluations utilized by previous invocation of LS procedure).

Now, we undertake the comparative analysis among ILS-μCHC and ILS-REx and ILS-μGA using Wilcoxon's test. Table 2 contains the results of this statistical test for p-value $= 0.05$ and 0.1.

Results outlined in Table 2 reveal that, in general, ILS-μCHC outperforms ILS-REx and ILS-μGA; being the superiority statistically significant (most $R-$ values are lower than both $R+$ ones and critical values). Only with p-value $= 0.05$, there are not significant differences between our algorithm and ILS-μGA ($\sigma_p = 0.1$), nevertheless, it shows clear advantage on this algorithm with p-value $= 0.1$.

The improvements with regards to a perturbation mechanism with random exploration and another one based on standard μGA suggest that the careful design of μCHC made with the aim of providing useful diversification really allows fruitful starting points to be found for LS processing.

3.3.4. Comparison with ILS models with strategies to enhance diversification

In this section, we compare ILS-μCHC with other ILS models built by enhancing diversification properties of standard ILS:

- *ILS with random walk acceptance* (ILS-RW): The acceptance criterion can roughly be used to control the balance between I&D for ILS search [63]. A simple way to illustrate this is to consider a *Markovian* acceptance criterion. A very strong intensification is achieved if only better solutions are accepted. At the opposite extreme is the random walk acceptance criterion (denoted by RW) which always applies the perturbation to the most recently visited local optimum, irrespective of its cost. This criterion clearly favors diversification over intensification, because it promotes a stochastic search in the space of local optima.
- *Guided restart* ILS (GRILS) [66] is based on basic ILS plus a restarting mechanism, which is used to generate the new solution that undergoes LS. This mechanism makes use of the search history of previous restarts and the best found solution. The aim is to bias the solution construction in favor of solutions which are likely not to belong to already explored regions (diversification). For each variable x_i, the frequency $freq_i$ of assignments to 1 is computed. This frequency is used as the probability to have the assignment $x_i = 0$ in the new initial solution. Therefore, the higher the frequency of 1's in the last k restart solutions, the lower the probability of having the same assignment to x_i in the new initial solution. This diversification mechanism is counterbalanced by considering also the best solution found $x^{best} = (x_1^{best}, \ldots, x_n^{best})$ to compute the probability vector. The frequency is incremented if $x_i^{best} = 0$ and decremented for the opposite assignment. Therefore, the probability vector is slightly moved toward the current most promising region. The length of the list of the last recently visited solution has been set to 10. It is worth noting that GRILS follows an idea that is opposite to the one underlying in ILS/GM [106] (see Section 3).
- *Collaborative ILS* (CILS): Another convenient way of empowering ILS exploration involves the idea of replacing a single ILS run by a population of ILS runs that interact each other in some way [92] (we call this ILS model CILS). The aim of this strategy is to avoid a stagnation behavior by, in some sense, delaying the decision on which solution one has to concentrate to find the highest solution quality; by the use of a population of ILS runs, the algorithm is not forced to concentrate the search only around the best solution found as done in single ILS runs. We have implemented a variant of CILS, called replace-worst, that starts with λ solutions each of which follows a standard ILS algorithm, except that every n_{It} iterations a copy of the current best solution replaces the worst solution in the population. We have considered $\lambda = 20$ and tried two different situations with regards to the communication strategy among ILS runs: (1) without communication (i.e., multiple independent trials of an ILS algorithm) and (2) with $n_{It} = 3$. They are denoted as CILS-wc and CILS-3, respectively.

The performance comparison among ILS-μCHC and each one of these algorithms was carried out by means of Wilcoxon's test. Table 3 has the results for p-value $= 0.05$.

Results of Wilcoxon's test in Table 3 advise us that our algorithm consistently outperforms all ILS algorithms based on strategies to enhance diversification. In particular, an interesting remark from improvement on CILS is that it becomes more fruitful executing only one ILS instance with a perturbation operator that focuses on diversification by managing a set of solutions (such as ILS-μCHC does) than favoring diversification by keeping multiple collaborative ILS runs that employ standard perturbation operator.

Table 3
ILS-μCHC ($\sigma_p = 0.25$) vs. ILS algorithms with enhanced diversification (Wilcoxon's test with p-value $= 0.05$).

ILS-μCHC vs.	$R+$	$R-$	Critical value	Sig. differences?
ILS-RW ($\sigma_p = 0.1$)	190.0	0.0	46	Yes
ILS-RW ($\sigma_p = 0.25$)	190.0	0.0	46	Yes
ILS-RW ($\sigma_p = 0.5$)	185.0	5.0	46	Yes
ILS-RW ($\sigma_p = 0.75$)	163.0	27.0	46	Yes
GRILS ($\sigma_p = 0.1$)	161.0	29.0	46	Yes
GRILS ($\sigma_p = 0.25$)	170.0	20.0	46	Yes
GRILS ($\sigma_p = 0.5$)	183.0	7.0	46	Yes
GRILS ($\sigma_p = 0.75$)	171.0	19.0	46	Yes
CILS-wc ($\sigma_p = 0.1$)	182.0	8.0	46	Yes
CILS-wc ($\sigma_p = 0.25$)	182.0	8.0	46	Yes
CILS-wc ($\sigma_p = 0.5$)	179.0	11.0	46	Yes
CILS-wc ($\sigma_p = 0.75$)	163.0	27.0	46	Yes
CILS-3 ($\sigma_p = 0.1$)	181.0	9.0	46	Yes
CILS-3 ($\sigma_p = 0.25$)	180.0	10.0	46	Yes
CILS-3 ($\sigma_p = 0.5$)	179.0	11.0	46	Yes
CILS-3 ($\sigma_p = 0.75$)	163.0	27.0	46	Yes

These results and ones obtained in previous sections allow us to conclude that our evolutionary perturbation technique may really enhance the operation of the ILS algorithm; in fact, the ILS instance performing this perturbation technique resulted very competitive with the state-of-the-art on this well-known MH for binary optimization problems. In this way, this proposal makes an important step in the progression of understanding the design of promising HMH-EA$_{I\&D}$ models by embedding evolutionary I&D components in MHs.

3.3.5. Comparison with MAs

ILS with μCHC is a new hybrid MH that combines a GA model (a customized version of CHC) with an LS method. Nowadays, there exist other kinds of hybrid MH models combining EAs and LS procedures, such as MAs (Section 2.2). One commonly used formulation of MAs applies LS to members of the EA population after recombination and mutation, with the aim of exploiting the best search regions gathered during the global sampling done by the EA. In this case, the LS procedure works within the EA. The rationale behind MAs is to provide an effective and efficient global optimization method by compensating for deficiency of EA in local exploitation and inadequacy of LS in global exploration.

Many different instantiations of MAs have been reported across a wide variety of application domains that range from scheduling and floor-planning problems, to pattern recognition, vehicle routing, control systems, aircraft, and drug design, 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 [40,41,74]. In view of that, it becomes primordial to clarify the differences between the MA approach and ILS with μCHC, in order to characterize our algorithm as a new technique for hybridizing GAs and LS methods, which becomes different from the one represented by MAs. Next, we attempt to explain them:

1. Both MAs and ILS with μCHC are *integrative* hybrid MHs (see Section 2.2). In MAs, the EA is the *master* MH, whereas in our hybrid MH, μCHC is the *subordinate* MH. μCHC has been designed as a component of ILS, and was embedded in the framework of this MH preserving its essence as much as possible.

Table 4
ILS-μCHC ($\sigma_p = 0.25$) vs. MA instances (Wilcoxon's test with p-value $= 0.05$).

ILS-μCHC vs.	$R+$	$R-$	Critical value	Sig. differences?
MA-CHC	175	15	46	Yes
MA-Hart	158	32	46	Yes

2. MAs follows the idea of *population-based* MHs, which perform search processes that describe the evolution of a set of points in the search space. On the other hand, our ILS approach belongs to the group of search algorithms working on single solutions, called *trajectory* methods, which all share the property of describing a single trajectory in the search space during the search process (see Fig. 7).
3. The main task of EAs in MAs is to act as *global diversification agents*. They attempt to induce *reliability* in the search process by ensuring that different promising search zones of the search space are the focus of the LS method throughout the run. In fact, in the MA literature, keeping population diversity while using LS together with an EA is always an issue to be addressed, either implicitly or explicitly [61,96]. On the other hand, μCHC incorporates mechanisms aimed to induce reliability in the ILS operation by exploring the neighborhood of a particular solution, S^*, i.e., it provides a kind of "*local diversification*".

Now, we are interested in determining whether our ILS model really provides an improved alternative to the scheme for combining EAs and LS procedures represented by MAs. In order to do this, we pit ILS-μCHC against two common MA implementations:

- *MA-CHC*: CHC arises as a very adequate EA for designing MAs, because it incorporates different techniques to promote high population diversity. MA-CHC consists in an original version of CHC ($N = 50$ and $p_{cm} = 0.35$; see Section 3.1) that invokes the LS algorithm (first-improvement hill-climbing algorithm) to refine every chromosome created by the crossover operator.
- *MA-Hart*: Most MA models apply the LS operator to all the individuals generated by the genetic operators [57]. However, in this case, the additional fitness function evaluations required for the LS increment considerably the computational cost of the MA, which may be prohibitive for many problems. Hart introduced a parameter, called *LS probability*, p_{LS}, that determines the probability of applying LS to every created chromosome [39]. In [39], p_{LS}=0.0625 was considered appropriate for many practical cases. We have implemented a variation of MA-CHC, called MA-Hart, which follows this MA approach.

Table 4 summarizes the existence or absence of statistical differences by using Wilcoxon's test between ILS-μCHC and the two MAs, MA-CHC and MA-Hart. The superiority of ILS-μCHC is clearly seen on the results presented in this table. Hence, it may be claimed that we might develop a hybrid MH integrating a specific version of the CHC algorithm and an LS method (by following the ideas stated in Section 2.3) that exhibits better performance compared to one of the most well-accepted models for combining GAs with LS procedures, i.e., the MA approach. We think that this becomes an interesting remark; since CHC has arisen as a reference point in the GA literature [43,103], obtaining powerful hybrid MHs incorporating this GA model as component is a topic of great importance.

4. Conclusions

In this paper, we provided an overview of different ways $\text{EA}_{\text{I\&D}}$ may be combined with other MHs, and even with other kinds of search algorithms, to obtain effective hybrid MHs. We have identified three lines of research in this topic: collaborative $\text{HMH-EA}_{\text{I\&D}}$, integrative $\text{HMH-EA}_{\text{I\&D}}$, and MHs with evolutionary I&D components. With the aim of complementing the review, we have taken an important next step along the less investigated $\text{HMH-EA}_{\text{I\&D}}$ approach by contributing with an ILS algorithm with an evolutionary perturbation method based on CHC. This algorithm has turned out to be very competitive with state-of-the-art ILS MH for binary optimization problems. Thus, we might confirm that it is possible to design evolutionary I&D components that replace, without great difficulties, determinate I&D components of classical MHs, developing their work more effectively and with a relatively low computational cost. In addition, the superior performance shown by our ILS instance in comparison with MAs indicates that its implicit innovative scheme for combining CHC with an LS method arises as a significant alternative to the one in the well-accepted memetic approach.

The good performance of the reviewed $\text{HMH-EA}_{\text{I\&D}}$ and the suitable results shown by the proposed instance allow an additional conclusion to be pointed out: the use of $\text{EA}_{\text{I\&D}}$ for building hybrid MHs becomes a prospective research area for finding more effective search algorithms. Therefore, the research line focused in this paper is indeed worth of further studies. We are currently extending our investigation to build evolutionary ILS-perturbation techniques being well-suited to deal with real-world problems (traveling salesman problem, quadratic assignment problem, etc.). Also we intend to: (1) incorporate local GAs and CMA-ES (Section 2.3) in different MHs based on LS procedures (ILS, VNS, MAs, GRASP, ant colony optimization, etc.) and (2) design versions specializing in I&D of other population-based MHs, such as ant colony optimization algorithms [18] and particle swarm optimization algorithms [53], using recent studies on these algorithms [10,15,44,84], and then, build new $\text{HMH-EA}_{\text{I\&D}}$ models with them.

Appendix A. Test suite

The test suite that we have used for different experiments consists of 19 binary-coded optimization problems. Tables 5 shows their names, reference where a detailed description may be found, the length of the binary solutions (L), whether they are formulated as maximization or minimization problems, and finally, the fitness value of the global optimum. Since problems from f_{Sch} to f_{SLE} are defined on continuous domains, their variables were encoded into bit

Table 5
Test suite.

Name	Ref.	L	Max/Min	Fit. op.
Deceptive problem (D)	[30]	120	Max	900
Massively multimodal deceptive problem (MMD)	[31]	240	Max	40
Bipolar deceptive problem (BD)	[78]	120	Max	20
Overlapping deceptive problem (OD)	[78]	150	Max	74
Trap problem (T)	[98]	180	Max	1100
Trap-5 problem (T5)	[79]	150	Max	150
Royal road problem (RR)	[24]	200	Max	200
Hierarchical if-and-only-if problem	[47]			
HIFF1		128	Max	1024
HIFF2		256	Max	2304
Zero/one multiple knapsack problem	[97]			
K1 (weing7 [42])		105	Max	1,095,445
K2 (weish26 [42])		90	Max	9584
Schwefel's function 2.21 (f_{Sch})	[26]	100	Min	0
Quartic noise function (f_{QN})	[26]	100	Min	0
Rotated generalized Rastrigin's function (f_{RRas})	[26]	100	Min	0
Generalized Griewank function (f_{Gri})	[26]	100	Min	0
Composed $f_{Gri} - f_{Ros}$ (f_C)	[26]	100	Min	0
Schaffer's function (f_{Scha})	[26]	100	Min	0
Expanded F10 (EF_{10})	[26]	100	Min	0
Systems of linear equations (f_{SLE})	[26]	200	Min	0

Table 6
Results of the algorithms.

Algorithm (σ_p)	*D*	*MMD*	*BD*	*OD*	*T*	*T* 5	*RR*	*HIFF* 1	*HIFF* 2	*K* 1	*K* 2	f_{Sch}	f_{QN}	f_{RRas}	f_{Gri}	f_C	f_{Scha}	EF_{10}	f_{SLE}
ILS-μCHC (0.1)	848.3	30.8	18.9	72.7	1057.9	121.8	126.7	568.1	1025.4	1,093,127	11,771.1	1.2e−4	3.2e−4	2.2	2.2e−2	1.4e−1	8.2e−2	2.0	389.3
ILS-μCHC (0.25)	853.6	30.8	18.6	71.5	1065.8	122.8	134.4	606.2	1097.4	1,093,321	11,757.6	1.8e−4	4.8e−4	5.2e−1	1.1e−2	1.0e−1	1.1e−1	1.8	270.9
ILS-μCHC (0.50)	853.8	29.7	18.5	70.6	1062.2	123.7	130.9	595.0	1067.2	1,093,070	11,702.0	2.1e−4	6.3e−4	5.0e−1	1.8e−2	1.5e−1	1.7e−1	1.5	231.7
ILS-μCHC (0.75)	869.6	29.4	18.5	70.0	1057.0	129.6	129.6	548.6	915.4	1,092,181	11,697.5	2.4e−4	6.9e−4	4.2e−1	2.0e−2	1.1e−1	1.5e−1	1.6	266.7
SILS (0.1)	850.6	28.4	18.8	71.0	999.3	121.9	88.6	512.2	845.4	1,084,554	11,358.3	3.3e−4	6.8e−4	6.6e−1	1.6e−2	2.3e−1	8.1e−1	6.9	339.4
SILS (0.25)	853.3	28.3	18.6	69.8	984.5	123.3	53.4	440.3	784.5	1,071,749	11,218.4	7.7e−4	2.3e−3	6.8e−1	3.2e−2	2.9e−1	2.5	7.9	475.9
SILS (0.5)	858.1	28.3	18.5	69.0	984.3	125.6	36.6	414.6	763.4	1,059,084	10,677.0	8.3e−4	7.1e−3	1.4	3.4e−2	2.6e−1	5.0	1.3e + 1	523.2
SILS (0.75)	866.3	28.3	18.6	69.2	979.9	131.1	33.4	439.0	782.2	1,053,160	10,091.3	9.7e−4	1.3e−2	2.8	4.1e−2	2.6e−1	8.4	1.5e + 1	670.3
GILS (0.1)	849.6	28.4	18.7	71.7	1005.2	122.4	77.6	553.3	913.4	1,086,851	11,442.8	1.5e−4	5.2e−4	8.4e−1	1.2e−2	2.1e−1	1.3	5.2	335.2
GILS (0.25)	851.2	28.4	18.7	70.7	995.8	122.6	78.2	481.9	833.1	1,081,913	11,339.9	3.6e−4	7.7e−4	7.0e−1	1.8e−2	2.0e−1	1.8	5.4	412.2
GILS (0.5)	853.1	28.3	18.6	69.8	986.5	123.4	53.4	436.2	782.1	1,071,898	11,242.8	6.4e−4	2.3e−3	7.6e−1	2.7e−2	1.9e−1	2.5	9.4	509.1
GILS (0.75)	855.6	28.2	18.6	69.3	982.1	124.4	40.8	415.6	769.6	1,064,176	11,054.9	6.7e−4	4.6e−3	1.1	3.2e−2	2.1e−1	3.6	1.0e + 1	519.4
PILS (0.1)	850.6	28.2	18.6	70.7	1004.9	121.9	75.7	510.9	879.9	1,085,840	11,418.8	1.4e−4	4.9e−4	6.2e−1	1.2e−2	3.0e−1	6.6e−1	7.6	354.7
PILS (0.25)	851.9	28.3	18.6	70.0	998.8	123.1	61.1	459.6	822.8	1,083,017	11,392.9	2.6e−4	7.0e−4	7.2e−1	1.6e−2	2.4e−1	7.0e−1	8.0	425.0
PILS (0.5)	857.0	28.3	18.6	69.8	992.0	125.4	57.0	449.5	811.0	1,079,100	11,355.9	2.9e−4	1.2e−3	9.2e−1	1.6e−2	2.4e−1	2.0	9.4	449.6
PILS (0.75)	866.0	28.3	18.6	70.0	995.0	130.8	57.9	459.2	829.0	1,077,558	11,352.9	3.4e−4	1.7e−3	1.1	1.6e−2	2.8e−1	2.3	1.1e + 1	459.7
ILS/GM (0.1)	848.6	28.5	18.7	71.9	1010.2	121.7	81.1	542.1	905.4	1,086,622	11,459.9	1.6e−4	4.5e−4	7.4e−1	1.1e−2	2.3e−1	1.0	6.0	336.6
ILS/GM (0.25)	851.7	28.5	18.7	70.6	997.3	122.2	81.1	494.3	829.4	1,083,295	11,368.4	3.5e−4	7.9e−4	5.4e−1	1.8e−2	2.3e−1	1.0	7.7	437.5
ILS/GM (0.5)	853.2	28.3	18.6	69.7	987.8	123.2	52.5	445.2	778.8	1,073,329	11,244.4	7.1e−4	1.9e−3	6.2e−1	2.2e−2	2.6e−1	2.4	9.2	442.7
ILS/GM (0.75)	854.8	28.4	18.6	69.4	985.3	124.4	40.0	419.1	767.3	1,066,857	11,102.6	8.9e−4	4.6e−3	9.2e−1	3.1e−2	2.3e−1	3.3	1.1e + 1	558.6
ILS-REx (0.1)	858.2	30.1	18.9	71.5	1037.6	123.3	11,2.2	573.0	960.8	1,089,066	11,618.3	2.0e−4	3.1e−4	3.4	1.8e−2	2.3e−1	1.3e−1	4.6	329.2
ILS-REx (0.25)	857.0	28.9	18.5	69.2	1005.6	126.2	68.5	471.6	830.5	1,077,038	11,258.6	7.0e−4	8.6e−4	1.6e−1	2.1e−2	2.1e−1	3.3e−1	2.7	288.9
ILS-REx (0.50)	860.3	28.7	18.5	68.5	990.1	127.3	43.5	427.4	784.5	1,064,490	10,816.5	8.1e−4	5.8e−3	1.4	4.7e−2	3.6e−1	2.9	7.4	459.7
ILS-REx (0.75)	868.7	29.0	18.6	69.1	987.0	132.3	42.4	469.2	833.4	1,052,373	10,205.9	1.0e−3	1.4e−2	2.9	5.2e−2	3.3e−1	8.2	1.3e + 1	606.2
ILS-μGA (0.1)	851.8	30.8	18.9	72.5	1049.7	122.2	124.6	604.7	1038.2	1,091,402	11,694.7	1.2e−4	3.3e−4	2.5	1.5e−2	2.0e−1	8.2e−2	2.2	304.8
ILS-μGA (0.25)	857.3	29.1	18.6	69.4	1020.2	125.7	77.6	495.0	873.9	1,081,448	11,396.3	5.5e−4	9.4e−4	2.0e−1	1.2e−2	1.4e−1	2.2e−1	1.5	173.6
ILS-μGA (0.5)	861.1	28.8	18.5	68.5	1000.0	128.0	44.0	435.4	806.0	1,067,168	10,887.6	8.6e−4	6.0e−3	1.2	4.0e−2	2.8e−1	1.2	2.8	368.5
ILS-μGA (0.75)	872.2	29.2	18.6	69.3	990.9	135.0	41.9	491.4	878.2	1,054,886	10,305.1	1.1e−3	1.5e−2	2.7	5.1e−2	2.8e−1	6.1	9.5	482.7
ILS-RW (0.1)	847.2	27.1	18.3	68.0	950.6	121.8	51.0	411.1	757.3	1,067,993	11,193.3	5.0e−4	3.8e−3	1.9	3.0e−2	2.5e−1	1.6	8.6	427.4
ILS-RW (0.25)	848.4	27.9	18.5	68.5	962.8	122.2	39.4	411.0	763.6	1,062,459	11,128.5	7.4e−4	5.6e−3	1.6	3.3e−2	3.2e−1	4.1	1.0e + 1	525.3
ILS-RW (0.5)	856.5	28.3	18.6	69.1	981.3	125.3	35.5	413.4	761.0	1,060,107	10,628.4	9.1e−4	7.4e−3	1.6	3.6e−2	3.0e−1	5.4	1.3e + 1	574.9
ILS-RW (0.75)	869.3	27.8	18.5	69.9	998.8	131.8	36.5	415.9	765.6	1,057,666	10,433.2	9.1e−4	8.3e−3	1.6	4.8e−2	3.1e−1	5.9	1.3e + 1	574.6
GRILS (0.1)	875.2	28.3	18.5	69.8	995.4	135.2	43.8	413.8	762.8	1,061,194	10,976.9	7.0e−4	5.1e−3	1.5	3.4e−2	2.3e−1	2.0	7.7	391.5
GRILS (0.25)	863.4	28.4	18.5	69.5	991.0	128.2	44.3	417.4	762.6	1,060,786	10,992.6	8.7e−4	6.0e−3	1.4	3.5e−2	2.9e−1	3.6	8.1	479.6
GRILS (0.5)	859.2	28.4	18.5	69.5	989.1	126.0	44.8	417.8	762.0	1,062,518	10,977.8	9.0e−4	6.8e−3	1.3	3.8e−2	2.6e−1	5.2	1.3e + 1	527.3
GRILS (0.75)	864.6	28.3	18.5	69.5	990.3	130.0	43.0	413.3	768.5	1,061,962	10,990.9	7.6e−4	6.3e−3	1.4	3.7e−2	2.6e−1	4.0	1.5e + 1	627.8
CILS-wc (0.1)	855.0	27.9	18.5	69.5	977.2	124.7	61.1	452.9	786.1	1,078,104	11,294.0	5.4e−4	1.7e−3	1.2	3.2e−2	8.2e−2	3.5	4.8	454.9
CILS-wc (0.25)	855.2	28.1	18.6	69.2	975.9	124.7	45.4	424.2	776.8	1,067,631	11,191.3	7.0e−4	3.9e−3	1.2	3.3e−2	8.2e−2	3.7	4.8	454.9
CILS-wc (0.5)	858.0	28.3	18.5	69.0	982.1	125.5	36.5	413.2	767.1	1,060,215	10,626.9	8.7e−4	7.4e−3	1.3	3.4e−2	8.2e−2	3.6	4.8	454.9
CILS-wc (0.75)	869.7	28.0	18.6	69.7	991.9	132.3	34.9	425.3	774.0	1,054,873	10,247.6	7.4e−4	9.0e−3	1.6	3.5e−2	8.2e−2	4.0	4.8	454.9
CILS-3 (0.1)	855.0	27.8	18.5	69.3	975.9	124.7	57.4	442.8	779.8	1,074,807	11,263.1	5.0e−4	1.9e−3	1.3	2.8e−2	8.2e−2	3.5	4.8	454.9
CILS-3 (0.25)	854.9	28.1	18.5	69.1	974.2	124.7	44.8	422.0	774.1	1,068,276	11,174.8	7.0e−4	3.9e−3	1.1	3.1e−2	8.2e−2	3.7	4.8	454.9
CILS-3 (0.5)	857.9	28.3	18.5	69.1	982.9	125.6	36.5	414.1	767.0	1,057,433	10,683.8	8.6e−4	7.2e−3	1.4	3.8e−2	8.2e−2	3.6	4.8	454.9
CILS-3 (0.75)	869.6	28.1	18.5	69.7	992.9	132.2	35.8	423.1	771.1	1,054,968	10,287.4	7.4e−4	8.8e−3	1.7	3.9e−2	8.2e−2	4.0	4.8	454.9
MA-CHC	855.8	28.1	18.5	69.1	977.5	125.2	61.3	431.5	768.7	1,084,379.9	11,409.4	7.7e−4	8.2e−4	1.2	2.5e−2	4.6e−2	3.5	2.6	373.7
MA-Hart	853.1	27.7	18.4	68.4	973.2	123.9	131.4	467.1	776.9	1,095,168.8	11,800.7	1.3e−4	7.8e−4	9.4e−1	1.7e−2	2.0e−1	2.2	11	534.5

strings using binary reflected Gray coding, with 20 binary genes assigned to each variable. The dimension of the search space is 10 for f_{SLE} and 5 for the remaining continuous test functions. These continuous functions were considered for our experimental study as benchmark functions for *comparison purposes* between search algorithms that handle binary-coded solutions. Thus, our pretension was not to improve the results obtained by other search algorithms on these problems, such as G-CMA-ES [6], which is long recognized as one of the best optimizers for continuous problems. In fact, it was the winner of the *real-parameter optimization competition organized in the* 2005 *IEEE congress on evolutionary computation* (as recognized by Langdon [59]).

Appendix B. The Wilcoxon matched-pairs signed-ranks test

Wilcoxon's test is used for answering this question: *Do two samples represent two different populations*? It is a non-parametric procedure employed in a hypothesis testing situation involving a design with two samples. It is the analogous of the paired *t*-test in non-parametrical statistical procedures; therefore, it is a pairwise test that aims to detect significant differences between the behavior of two algorithms.

The null hypothesis for Wilcoxon's test is $H_0 : \theta_D = 0$; in the underlying populations represented by the two samples of results, the average of the difference scores equals zero. The alternative hypothesis is $H_1 : \theta_D \neq 0$, but also can be used $H_1 : \theta_D > 0$ or $H_1 : \theta_D < 0$ as directional hypothesis.

In the following, we describe the test computations. Let d_i be the difference between the performance scores of the two algorithms on *i*-th out of *N* functions. The differences are ranked according to their absolute values; average ranks are assigned in case of ties. Let R^+ be the sum of ranks for the functions on which the second algorithm outperformed the first, and R^- the sum of ranks for the opposite. Ranks of $d_i = 0$ are split evenly among the sums; if there is an odd number of them, one is ignored:

$$R^+ = \sum_{d_i > 0} rank(d_i) + \frac{1}{2} \sum_{d_i = 0} rank(d_i)$$

and

$$R^- = \sum_{d_i < 0} rank(d_i) + \frac{1}{2} \sum_{d_i = 0} rank(d_i).$$

Let *T* be the smallest of the sums, $T = \min(R^+, R^-)$. If *T* is less than or equal to the value of the distribution of Wilcoxon for *N* degrees of freedom (Table B.12 in [104]), the null hypothesis of equality of means is rejected.

The obtaining of the *p*-value associated to a comparison is performed by means of the normal approximation for the Wilcoxon *T* statistic (Section VI, Test 18 in [89]). Furthermore, the computation of the *p*-value for this test is usually included in well-known statistical software packages (SPSS, SAS, R, etc.).

Appendix C. Results of the algorithms

Table 6 outlines the average of the best fitness function found at the end of each run for all algorithms executed for the experimental studies carried out in the paper.

References

[1] Alba E, Tomassini M. Parallelism and evolutionary algorithms. IEEE Trans Evol Comput 2002;6(5):443–62.

[2] Alba E, Luna F, Nebro A, Troya JM. Parallel heterogeneous genetic algorithms for continuous optimization. Parallel Comput 2004;30(5–6):699–719.

[3] Alba E, Dorronsoro B. The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. IEEE Trans Evol Comput 2005;9(2):126–42.

[4] Alba E, editor. Parallel metaheuristics: a new class of algorithms. New York: Wiley; 2005.

[5] Auger A, Hansen N. Performance evaluation of an advanced local search evolutionary algorithm. In: Proceedings of the 2005 IEEE congress on evolutionary computation. New York: IEEE Press; 2005. p. 1777–84.

[6] Auger A, Hansen N. A restart CMA evolution strategy with increasing population size. In: Proceedings of the 2005 IEEE congress on evolutionary computation. New York: IEEE Press; 2005. p. 1769–76.

[7] Bäck T. Evolutionary algorithms in theory and practice. New York: Oxford University Press; 1996.

[8] Bäck T, Fogel DB, Michalewicz Z. Handbook of evolutionary computation. Institute of Physics Publishers; 1997.

[9] Beyer H-G, Schwefel H-P. Evolution strategies: a comprehensive introduction. Nat Comput 2002;1:2–52.

[10] Blum C. ACO applied to group shop scheduling: a case study on intensification and diversification. In: Dorigo M, Di Caro G, Sampels M, editors. Proceedings of the third international workshop ANTS. Lecture notes in computer science, vol. 2463. Berlin, Heidelberg: Springer; 2002. p. 14–27.

[11] Blum C, Roli A. Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput Surv 2003;35(3):268–308.

[12] Bonissone PP, Subbu R, Eklund N, Kiehl TR. Evolutionary algorithms + domain knowledge = real-world evolutionary computation. IEEE Trans Evol Comput 2006;10(3):256–80.

[13] Chaiyaratana N, Piroonratana T, Sangkawelert N. Effects of diversity control in single-objective and multi-objective genetic algorithms. J Heuristics 2007;13(1):1–34.

[14] Chelouah R, Siarry P. Genetic and Nelder–Mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions. Eur J Oper Res 2003;148:335–48.

[15] Chen X, Li Y. A modified PSO structure resulting in high exploration ability with convergence guaranteed. IEEE Trans Syst Man Cybern B 2007;37(5):1271–89.

[16] Cordeau J-F, Laporte G, Pasin F. An iterated local search heuristic for the logistics network design problem with single assignment. Int J Prod Econ 2008;113(2):626–40.

[17] Cordón O, Damas S. Image registration with iterated local search. J Heuristics 2006;12(1–2):73–94.

[18] Dorigo M, Stützle T. Ant colony optimization. Cambridge, MA: The MIT Press; 2004.

[19] Eiben AE, Smith JE. Introduction to evolutionary computing. Berlin: Springer; 2003.

[20] El-Abd M, Kamel M. A taxonomy of cooperative search algorithms. In: Blesa MJ, Blum C, Roli A, Sampels M, editors. Hybrid metaheuristics. Lecture notes in computer science, vol. 3636. Berlin, Heidelberg: Springer; 2005. p. 32–41.

[21] Eshelman LJ. The CHC adaptive search algorithm: how to have safe search when engaging in non-traditional genetic recombination. In: Rawlins G, editor. Foundations of genetic algorithms, vol. 1. California: Morgan Kaufmann; 1991. p. 265–83.

[22] Fernandes C, Rosa A. A study on non-random mating and varying population size in genetic algorithms using a royal road function. In: Proceedings of the 2001 congress on evolutionary computation. New Jersey: IEEE Service Center; 2001. p. 60–6.

[23] Fogel DB. Evolutionary computation: toward a new philosophy of machine intelligence. New York: IEEE Press; 1995.

[24] Forrest S, Mitchell M. Relative building block fitness and the building block hypothesis. In: Whitley LD, editor. Foundations of genetic algorithms, vol. 2. California: Morgan Kaufmann; 1993. p. 109–26.

[25] Gang P, Iimura I, Nakayama S. Application of genetic recombination to genetic local search in TSP. Int J Inf Technol 2007;13(1):57–66.

[26] García-Martínez C, Lozano M, Herrera F, Molina D, Sánchez AM. Global and local real-coded genetic algorithms based on parent-centric crossover operators. Eur J Oper Res 2008;185:1088–113.

[27] García-Martínez C, Lozano M. Local search based on genetic algorithms. In: Siarry P, Michalewicz Z, editors. Advances in metaheuristics for hard optimization. Berlin: Springer; 2008. p. 199–221.

[28] Glover F, Kochenberger G, editors. Handbook of metaheuristics. Massachusetts: Kluwer Academic Publishers, 2003.

[29] Goldberg DE. Genetic algorithms in search, optimization, and machine learning. Reading, MA: Addison-Wesley; 1989.

[30] Goldberg DE, Korb B, Deb K. Messy genetic algorithms: motivation, analysis, and first results. Complex Syst 1989;3:493–530.

[31] Goldberg DE, Deb K, Horn J. Massively multimodality, deception, and genetic algorithms. In: Männer R, Manderick B, editors. Proceedings of the international conference on parallel problem solving from nature. Amsterdam: North-Holland Pub. Co; 1992. p. 37–46.

[32] Goldberg DE, Wang L. Adaptive niching via coevolutionary sharing. In: Quagliarella D, Periaux J, Poloni C, Winter G, editors. Genetic algorithms in engineering and computer science. New York: Wiley; 1997. p. 21–38.

[33] Grosan C, Abraham A. Hybrid evolutionary algorithms: methodologies, architectures, and reviews. In: Grosan C, Abraham A, Ishibuchi H, editors. Hybrid evolutionary algorithms. Berlin, Heidelberg: Springer; 2007. p. 1–17.

[34] Hansen P, Mladenović N. Variable neighborhood search. In: Glover F, Kochenberber GA, editors. Handbook of metaheuristics. Massachusetts: Kluwer Academic Publishers; 2003. p. 145–84.

[35] Hansen N, Müller SD, Koumoutsakos P. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evol Comput 2003;11(1):1–18.

[36] Hansen N, Ostermeier A. Completely derandomized self-adaptation in evolution strategies. Evol Comput 2001;9(2):159–95.

[37] Hansen N. Compilation of results on the CEC benchmark function set. Technical Report, Institute of Computational Science, ETH Zurich, Switzerland; 2005. Available at ⟨http://www.ntu.edu.sg/home/epnsugan/index_files/CEC-05/compareresults.pdf⟩.

[38] Hansen N, Kern S. Evaluating the CMA evolution strategy on multimodal test functions. In: Yao X, Burke E, Lozano JA, Smith J, Merelo-Guervós JJ, Bullinaria JA et al., editors. Proceedings of the international conference on parallel problem solving from nature. Lecture notes in computer science, vol. 3242. Berlin, Heidelberg: Springer; 2004. p. 282–91.

[39] Hart WE. Adaptive Global Optimization with Local Search. Ph.D. thesis, University of California, San Diego, California; 1994.

[40] Hart WE, Krasnogor N, Smith JE, editors. Recent advances in memetic algorithms. Studies in fuzzyness and soft computing, vol. 166. Berlin, Heidelberg, New York: Springer; 2004.

[41] Hart WE, Krasnogor N, Smith JE. Editorial introduction special issue on memetic algorithms. Evol Comput 2004;12(3):v–vi.

[42] Heitktter J. SAC-94 Suite of 0/1-multiple-knapsack problem instances, 2001. Available at ⟨http://elib.zib.de/pub/Packages/mp-testdata/ip/sac94-suite⟩.

[43] Herrera F, Lozano M. Gradual distributed real-coded genetic algorithms. IEEE Trans Evol Comput 2000;4(1):43–63.

[44] Ho SL, Yang S, Ni G, Wong HC. A particle swarm optimization method with enhanced global search ability for design optimizations of electromagnetic devices. IEEE Trans Magn 2006;42(4):1107–10.

[45] Hoos HH, Stützle T. Stochastic local search: foundations and applications. California: Morgan Kaufmann Publishers; 2004.

[46] García S, Molina D, Lozano M, Herrera F. A study on the use of nonparametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization. J Heuristics 2009; in press.

[47] Hu J, Goodman E, Seo K, Fan Z, Rosenberg R. The hierarchical fair competition (HFC) framework for sustainable evolutionary algorithms. Evol Comput 2005;13(2):241–77.

[48] Jones T. Crossover, macromutation, and population-based search. In: Eshelman L, editor. Proceedings of the sixth international conference on genetic algorithms. California: Morgan Kaufmann; 1995. p. 73–80.

[49] Kazarlis SA, Papadakis SE, Theocharis JB, Petridis V. Microgenetic algorithms as generalized hill-climbing operators for GA optimization. IEEE Trans Evol Comput 2001;5(3):204–17.

[50] Katayama K, Narihisa H. Iterated local search approach using genetic transformation to the travelling salesman problem. In: Banzhaf W, Daida J, Eiben AE, Garzon MH, Honovar V, Jakiela M et al., editors. Proceedings of the genetic and evolutionary computation conference. California: Morgan Kaufmann; 1999. p. 321–8.

[51] Katayama K, Narihisa H. A new iterated local search algorithm using genetic crossover for the travelling salesman problem. In: Proceedings of the ACM symposium on applied computing. 1999. p. 302–6.

[52] Kemenade CHM. Cluster evolution strategies, enhancing the sampling density function using representatives. In: Proceedings of the IEEE conference on evolutionary computation. New Jersey: IEEE Press; 1996. p. 637–42.

[53] Kennedy J, Eberhart RC. Swarm intelligence. Los Altos, CA: Morgan Kaufmann; 2001.

[54] Koumousis VK, Katsaras CP. A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance. IEEE Trans Evol Comput 2006;10(1):19–28.

[55] Koza JR. Genetic programing. Cambridge: The MIT press; 1992.

[56] Krasnogor N, Smith JE. A memetic algorithm with self-adapting local search: TSP as a case study. In: Proceedings of the 2000 international conference on genetic and evolutionary computation. California: Morgan Kaufmann; 2000. p. 987–94.

[57] Krasnogor N, Smith JE. A tutorial for competent memetic algorithms: model, taxonomy, and design issue. IEEE Trans Evol Comput 2005;9(5):474–88.

[58] Krishnakumar K. Micro-genetic algorithms for stationary and nonstationary function optimization. In: Proceedings of the SPIE intelligent control adaptive systems. 1989. p. 289–96.

[59] Langdon WB, Poli R. Evolving problems to learn about particle swarm optimizers and other search algorithms. IEEE Trans Evol Comput 2007;11(5): 561–78.

[60] Lima CF, Pelikan M, Sastry K, Butz M, Goldberg DE, Lobo FG. Substructural neighborhoods for local search in the Bayesian optimization algorithm. In: Runarsson TP, Beyer H-G, Burke E, Merelo-Guervós JJ, Whitley LD, Yao X, editors. Proceedings of the international conference on parallel problem solving from nature. Lecture notes in computer science, vol. 4193. Berlin, Heidelberg: Springer; 2006. p. 232–41.

[61] Lozano M, Herrera F, Krasnogor N, Molina D. Real-coded memetic algorithms with crossover hill-climbing. Evol Comput 2004;12(3):273–302.

[62] Lozano M, Herrera F, Cano JR. Replacement strategies to maintain useful diversity in steady-state genetic algorithms. Inf Sci 2008;178:4421–33.

[63] Lourenço HR, Martin OC, Stützle T. Iterated local search. In: Glover F, Kochenberger G, editors. Handbook of metaheuristics. Massachusetts: Kluwer Academic Publishers; 2003. p. 321–53.

[64] Meloni C, Naso D, Turchiano B. Multi-objective evolutionary algorithms for a class of sequencing problems in manufacturing environments. In: Proceedings of the IEEE international conference on systems, man and cybernetics, vol. 1. 2003. p. 8–13.

[65] Merz P. Memetic algorithms for combinatorial optimization problems: fitness landscapes and effective search strategies. Dissertation, University of Siegen, Germany; 2000.

[66] Milano M, Roli A. MAGMA: a multiagent architecture for metaheuristics. IEEE Trans Syst Man Cybern B 2004;33(2):925–41.

[67] Molina D, Lozano M, García-Martínez C, Herrera F. Memetic algorithms for continuous optimization based on local search chains. Evol Comput 2009; in press.

[68] Moscato P, Cotta C. A gentle introduction to memetic algorithms. In: Glover F, Kochenberger G, editors. Handbook of metaheuristics. Massachusetts: Kluwer Academic Publishers; 2003. p. 105–44.

[69] Mühlenbein H, Schomisch M, Born J. The parallel genetic algorithm as function optimizer. In: Belew R, Booker LB, editors. Proceedings of the international conference on genetic algorithms. California: Morgan Kaufmann; 1991. p. 271–8.

[70] Mühlenbein H, Schlierkamp-Voosen D. Predictive models for the breeder genetic algorithm I. Continuous parameter optimization. Evol Comput 1993;1:25–49.

[71] Mutoh A, Tanahashi F, Kato S, Itoh H. Efficient real-coded genetic algorithms with flexible-step crossover. IEEJ Electron Inf Syst 2006;126(5):654–60.

[72] Nagata Y, Kobayashi S. Edge assembly crossover: a high-power genetic algorithm for the traveling salesman problem. In: Bäck T, editor. Proceedings of the international conference on genetic algorithms. Los Altos, CA: Morgan Kaufmann; 1997. p. 450–7.

[73] Noman N, Iba H. Accelerating differential evolution using an adaptive local search. IEEE Trans Evol Comput 2008;12(1):107–25.

[74] Ong YS, Krasnogor N, Ishibuchi H. Guest editorial: special issue on memetic algorithms. IEEE Trans Systems Man Cybern Part B: Cybern 2007;37(1):2–5.

[75] O'Reilly UM, Oppacher F. Hybridized crossover-based search techniques for program discovery. In: IEEE international conference on evolutionary computation 1995. Piscataway, New Jersey: IEEE Press; 1995. p. 573–8.

[76] Papadakis SE, Theocharis JB. A GA-based fuzzy modeling approach for generating TSK models. Fuzzy Sets Syst 2002;131(2):121–52.

[77] Parthasarathy PV, Goldberg DE, Burns SA. Tackling multimodal problems in hybrid genetic algorithms. In: Whitley D, editor. Proceedings of the genetic and evolutionary computation conference. California: Morgan Kaufmann; 2001. p. 775.

[78] Pelikan M, Goldberg DE, Cantú-Paz E. Linkage problem, distribution estimation, and Bayesian networks. Evol Comput 2000;8(3):1063–6560.

[79] Pelikan M, Hartmann A, Lin TK. Parameter-less hierarchical Bayesian optimization algorithm. In: Lobo FG, Lima CF, Michalewicz Z, editors. Parameter setting in evolutionary algorithms, SCI, vol. 54. Berlin, Heidelberg: Springer; 2007. p. 225–39.

[80] Petrowski A. A clearing procedure as a niching method for genetic algorithms. In: Proceedings of the IEEE conference on evolutionary computation. New Jersey: IEEE Service Center; 1996. p. 798–803.

[81] Potts JC, Giddens TD, Yadav SB. The development and evaluation of an improved genetic algorithm based on migration and artificial selection. IEEE Trans Syst Man Cybern 1994;24:73–86.

[82] Preux Ph, Talbi E-G. Towards hybrid evolutionary algorithms. Int. Trans Oper Res 1999;6(6):557–70.

[83] Raidl GR. A unified view on hybrid metaheuristics. In: Almeida F, Blesa Aguilera MJB, Blum C, Moreno Vega JM, Pérez Pérez M, Roli A et al., editors. Hybrid metaheuristics. Lecture notes in computer science, vol. 4030. Berlin, Heidelberg: Springer; 2006. p. 1–126.

[84] Randall M. Search space reduction as a tool for achieving intensification and diversification in ant colony optimisation. In: Ali M, Dapoigny R, editors. Advances in applied artificial intelligence. Lecture notes in computer science, vol. 4031. Berlin, Heidelberg: Springer; 2006. p. 254–61.

[85] Rodriguez-Martin I, Salazar-Gonzalez JJ. An iterated local search heuristic for a capacitated hub location problem. In: Almeida F, Blesa Aguilera MJB, Blum C, Moreno Vega JM, Pérez Pérez M, Roli A et al., editors. Hybrid metaheuristics. Lecture notes in computer science, vol. 4030. Berlin, Heidelberg: Springer; 2006. p. 70–81.

[86] Sastry K, Goldberg DE. Designing competent mutation operators via probabilistic model building of neighborhoods. In: Deb K, Poli R, Banzhaf W, Beyer H-G, Burke E, Darwen P et al., editors. Proceedings of the genetic and evolutionary computation conference. Lecture notes in computer science, vol. 3103. Berlin, Heidelberg: Springer; 2004. p. 114–25.

[87] Schlierkamp-Voosen D, Mühlenbein H. Strategy adaptation by competing subpopulations. Davidor Y, Schwefel HP, Männer R, editors. Proceedings of the international conference on parallel problem solving from nature. Lecture notes in computer science, vol. 866. Berlin, Germany: Springer-Verlag; 1994. p. 199–208.

[88] Seront G, Bersini H. A new GA-local search hybrid for continuous optimization based on multi level single linkage clustering. In: Whitley D, Goldberg D, Cantú-Paz E, Spector L, Parmee I, Beyer H-G, editors. Proceedings of the genetic and evolutionary computation conference. California: Morgan Kaufmann; 2000. p. 90–5.

[89] Sheskin DJ. Handbook of parametric and nonparametric statistical procedures. Boca Raton, FL: CRC Press; 2003.

[90] Sinha A, Goldberg DE. A Survey of hybrid genetic and evolutionary algorithms. ILLIGAL Technical Report 2003004; 2004.

[91] Soak S-M, Lee S-W, Mahalik NP, Ahn B-H. A new memetic algorithm using particle swarm optimization and genetic algorithm. Knowledge-based intelligent information and engineering systems. Lecture notes in computer science, vol. 4251. Berlin: Springer; 2006. p. 122–9.

[92] Stützle T. Iterated local search for the quadratic assignment problem. Eur J Oper Res 2006;174:1519–39.

[93] Talbi E-G. A taxonomy of hybrid metaheuristics. J Heuristics 2002;8(5): 541–65.

[94] Talbi E-G, Bachelet T. COSEARCH: a parallel cooperative metaheuristic. J Math Model Algorithms 2006;5:5–22.

[95] Tang LX, Luo JX. A new ILS algorithm for parallel machine scheduling problems. J Intell Manuf 2006;17(5):609–19.

[96] Tang J, Lim MH, Ong YS. Diversity-adaptive parallel memetic algorithm for solving large scale combinatorial optimization problems. Soft Comput 2007;11(9):873–88.

[97] Thierens D. Adaptive mutation rate control schemes in genetic algorithms. In: Proceedings of the congress on evolutionary computation. New York: IEEE Press; 2002. p. 980–5.

[98] Thierens D. Population-based iterated local search: restricting neighborhood search by crossover. In: Deb K, Poli R, Banzhaf W, Beyer H-G, Burke E, Darwen P et al., editors. Proceedings of the genetic and evolutionary computation conference. Lecture notes in computer science, vol. 3103. Berlin, Heidelberg: Springer; 2004. p. 234–45.

[99] Tounsi M, Ouis S. An iterative local-search framework for solving constraint satisfaction problem. Appl Soft Comput 2008;8(4):1530–5.

[100] Tsutsui S, Ghosh A, Corne D, Fujimoto Y. A real coded genetic algorithm with an explorer and an exploiter population. In: Proceedings of the international conference on genetic algorithms. California: Morgan Kaufmann Publishers; 1997. p. 238–45.

[101] Wei L, Zhao M. A niche hybrid genetic algorithm for global optimization of continuous multimodal functions. Appl Math Comput 2005;160(3):649–61.

[102] Weicai Z, Jing L, Mingzhi X, Licheng J. A multiagent genetic algorithm for global numerical optimization. IEEE Trans Systems Man Cybern Part B 2004;34(2): 1128–41.

[103] Whitley D, Rana S, Dzubera J, Mathias E. Evaluating evolutionary algorithms. Artif Intell 1996;85:245–76.

[104] Zar JH. Biostatistical analysis. Englewood Cliffs, NJ: Prentice-Hall; 1999.

[105] Zhang Q, Sun J, Tsang EPK. Evolutionary algorithm with the guided mutation for the maximum clique problem. IEEE Trans Evol Comput 2005;9(2): 192–200.

[106] Zhang Q, Sun J. Iterated local search with guided mutation. In: IEEE congress on evolutionary computation. New York: IEEE Press; 2006. p. 924–9.