Soft Comput (2012) 16:2115–2133
DOI 10.1007/s00500-012-0881-x

ORIGINAL PAPER

Arbitrary function optimisation with metaheuristics

No free lunch and real-world problems

Carlos García-Martínez · Francisco J. Rodriguez · Manuel Lozano

Published online: 3 July 2012

Abstract No free lunch theorems for optimisation suggest that empirical studies on benchmarking problems are pointless, or even cast negative doubts, when algorithms are being applied to other problems not clearly related to the previous ones. Roughly speaking, reported empirical results are not just the result of algorithms' performances, but the benchmark used therein as well; and consequently, recommending one algorithm over another for solving a new problem might be always disputable. In this work, we propose an empirical framework, *arbitrary function optimisation framework*, that allows researchers to formulate conclusions independent of the benchmark problems that were actually addressed, as long as the context of the problem class is mentioned. Experiments on sufficiently general scenarios are reported with the aim of assessing this independence. Additionally, this article presents, to the best of our knowledge, the first thorough empirical study on the no free lunch theorems, which is possible thanks to the application of the proposed methodology, and whose main result is that no free lunch theorems unlikely hold on the set of *binary real-world problems*. In particular, it is shown that exploiting reasonable heuristics becomes more beneficial than random search when dealing with binary real-world applications.

Keywords Empirical studies · No free lunch theorems · Real-world problems · General-purpose algorithms · Unbiased results

1 Introduction

Wolpert and Macready (1997) presented the No Free Lunch (NFL) theorems for optimisation, which, roughly speaking, state that every non-revisiting algorithm performs equally well on average over all functions [or closed under permutations (c.u.p.) sets (Schumacher et al. 2001)]. This result had a profound impact on researchers that were seeking a superior *general-purpose* optimiser, because none would be better than random search without replacement. In particular, the evolutionary computation community was shocked among the most in the 1980s, because evolutionary algorithms were expected to be widely applicable and to have an overall superior performance than other methods (Jiang and Chen 2010). From then on, most effort was concentrated on solving concrete problem classes with specific-purpose optimisers, where performance superiority might really be attained.

One of the most critical aspects of NFL is that it inevitably casts doubts on most empirical studies. According to the NFL theorems, empirically demonstrated performance superiority of any algorithm predicts performance inferiority on any other problem whose relationship to the first is unknown (Marshall and Hinton 2010). Thus, several researchers have warned that "*from a theoretical point of view, comparative evaluation of search algorithms is a dangerous enterprise*" (Whitley and Watson 2005). At the bottom, reported

Research Projects TIN2011-24124 and P08-TIC-4173.

C. García-Martínez (✉)
Department of Computing and Numerical Analysis,
University of Córdoba, Córdoba 14071, Spain
e-mail: cgarcia@uco.es

F. J. Rodriguez · M. Lozano
Department of Computer Sciences and Artificial Intelligence
CITIC-UGR, University of Granada, Granada 18071, Spain
e-mail: fjrodriguez@decsai.ugr.es

M. Lozano
e-mail: lozano@decsai.ugr.es

empirical results are not just the product of algorithms' performances, but the benchmark (and running conditions) used therein as well; and consequently, recommending one algorithm over another for solving a problem of practical interest might be always disputable. Whitley and Watson (2005) encourage researchers to prove that test functions applied for comparing search algorithms (i.e., benchmarks) really capture the aspects of the problems they actually want to solve (real-world problems). However, this proof is usually missed, leading to a research deterioration that has favoured the apparition of too many new algorithms, ignoring the question whether some other algorithm could have done just as well, or even better. For the sake of research, it is crucial that researchers "*consider more formally whether the methods they develop for particular classes of problems actually are better than other algorithms*" (Whitley and Watson 2005).

In this work, we propose an empirical framework, *arbitrary function optimisation framework*, that lightens the dependence between the results of experimental studies and the actual benchmarks that are used. This fact allows the formulation of conclusions more general and interesting than "Algorithm A performs the best on this particular testbed". In fact, we have applied our framework on sufficiently general scenarios, the realms of the NFL theorems in particular, by a thorough experimentation (375.000 simulations, consuming 5 months of running time on eight computing cores, on a potentially infinite set of problem instances from many and different problem classes), which allows us to formulate the following conclusions:

1. Our framework certainly allows to formulate significant conclusions regardless of the actual problem instances addressed.
2. NFL theorems do not hold on a representative set of instances of binary problems from the literature.
3. Our framework is consistent with the NFL implications, providing the corresponding empirical evidence for c.u.p. set of problems.
4. NFL theorems unlikely hold on the set of *binary real-world problems*. In fact, we approximate the probability of the opposite to the value $1.6e{-}11$. In particular, it is shown that exploiting any reasonable heuristic, the evolutionary one among them, becomes more beneficial than random search when dealing with binary real-world applications.

Thus, this work additionally presents a clear and innovative depiction of the implications and limitations of NFL theorems that is interesting for researchers that are not used to the theoretical perspective of most NFL works. In particular, this work becomes very useful, especially in the light of incorrect interpretations that appear in some recent papers that suggest that ensembles, hybrids, or hyper-heuristics might overcome NFL implications [Dembski and Marks (2010) prove that this idea is incorrect]. In addition, our empirical methodology assists researchers to make progress on the development of competitive general-purpose strategies for real-world applications, evolutionary algorithms in particular, as well as providing scientific rigour to their empirical studies on particular problem classes.

Regarding our results, it is important to notice that it is not our intention to deny or faithfully support the NFL theorems on any scenario, propose or defend the application of one algorithm as the universal general-purpose solver, nor underestimate the utility of exploiting problem knowledge. On the contrary, our results just show that the hypothesis of NFL hardly holds on the set of interesting binary problems and that knowledge exploitation should be evaluated with regard to the performance of competitive general-purpose strategies.

An interesting added feature of this work is that it is accompanied by an associated website (García-Martínez et al. 2011a) where source codes, results, and additional comments and analysis are available to the specialized research community. Proper references are included along this work. Readers can access this material by appending the given section names to the website url.

This work is structured as follows: Section 2 overviews the literature about the NFL theorems relevant for the rest of the paper. Section 3 depicts the proposed arbitrary function optimisation framework. Section 4 analyses the conclusions that can be obtained from the application of our framework when comparing several standard algorithms on a large set (potentially infinite) of representative binary problems found in the literature. Section 5 studies the scenario where standard algorithms and their non-revisiting versions are applied on c.u.p. set of functions. Section 6 analyses the probability for the NFL to hold on the set of binary real-world problems. Section 7 discusses the lessons learned, and Sect. 8, the new challenges on analysing algorithms' performances. Section 9 presents the conclusions.

2 Revision on no free lunch theorems and their implications

This section overviews the intuitive interpretations and implications of NFL. The corresponding formal notation can be consulted at the webpage (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#NFL_theorems).

2.1 No free lunch theorems

The original "No Free Lunch Theorems for Optimization" (Wolpert and Macready 1997) can be roughly summarised as follows:

> *For all possible metrics, no* (non-revisiting) *search algorithm is better than another* (random search without replacement among others) *when its performance is averaged over all possible discrete functions.*

The formal definition of this theorem is provided at the webpage (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Original_NFL_theorem).

Later, Schumacher et al. (2001) sharpened the NFL theorem by proving that it is valid even in reduced set of benchmark functions. More concretely, NFL theorems apply when averaging algorithms' performances on a set of functions F if and only if F is closed under permutations (c.u.p.) [Whitley and Rowe (2008) reduced even more the set of functions by analysing concrete sets of algorithms]. F is said to be c.u.p. if for any function $f \in F$ and any permutation π of the search space, $f \circ \pi$ is also in F. The formal definition can be consulted at (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Sharpened_NFL_theorem). Notice, as an example, that the number of permutations of the search space of a problem with N binary variables is the factorial number $2^N!$

This redefinition of the NFL theorem incorporates two direct implications:

1. The union of two c.u.p. sets of functions (or the union of two sets of functions for which NFL holds) is c.u.p. This implication is deeper analysed by Igel and Toussaint (2004) leading to a more general formulation of the NFL theorem. In particular, NFL theorems are independent of the dimensions of the used functions if NFL holds at every particular dimension. This is formally shown at (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Sharpened_NFL_theorem).
2. Given any two algorithms and any function from a set c.u.p., there exists a *counter-acting function* in the same set for which the performances of the first algorithm on the first function and the second algorithm on the second function are the same (the inverse relation with the same two functions does not need to be true).

It is interesting to remark that the application of non-revisiting search algorithms on problems with many variables usually involves impossible memory and/or computation requirements (as an example, an unknown problem with n binary variables requires 2^n evaluations to guarantee to be solved to optimality), and therefore revisiting solutions is allowed for most approaches. Recently, Marshall and Hinton (2010) have proved that allowing revisiting solutions breaks the permutation closure, and therefore, performance differences between real algorithms may really appear. Moreover, they presented an approach to quantify the extent to which revisiting algorithms differ in performance according to the amount of revisiting they allow. Roughly speaking, one algorithm is expected to be better than another on a set of arbitrary functions if its probability for revisiting solutions is lower than that of the second algorithm. Subsequently, this idea allowed them to affirm that random search without replacement is expected to outperform any non-minimally revisiting algorithm on an unknown set of functions. Even when allowing revisiting solutions is a necessity (and the available number of evaluations is considerably inferior to the search space's size), we may intuitively suppose that random search (with replacement) is still expected to outperform any non-minimally revisiting algorithm, because its probability for revisiting solutions is usually very low. Therefore, random search seems to theoretically become always a competitor that general-purpose methods cannot outperform, whether NFL do or do not apply.

2.2 The role of knowledge

As a conclusion from NFL theorems, researchers have to acknowledge that any performance superiority shown by one algorithm on a certain problem class must be due to the presence of specific knowledge, of that problem class, this algorithm disposes and manages more fruitfully than the other algorithms (Whitley and Watson 2005; Wolpert and Macready 1997). Then:

> *...the business of developing search algorithms is one of building special-purpose methods to solve application-specific problems. This point of view echoes a refrain from the Artificial Intelligence community: "Knowledge is Power"* (Whitley and Watson 2005).

Once more, we realise that the existence of a general-purpose search method (one that does not apply specific problem knowledge) is impossible (at least none better than random search without replacement) (Whitley and Watson 2005).

2.3 The set of interesting functions

Droste et al. (1999, 2002) claim that classical NFL requires un-realistic scenarios because the set containing all functions cannot be described or evaluated with available resources. Then, they analyse a more realistic scenario that contains all the functions whose complexity is restricted, and they regard time, size, and Kolmogoroff complexity measures. For the analysed situations they proved that, though NFL does not hold on these restricted scenarios, one should not expect much by well-chosen heuristics, and

formulated the *almost-NFL theorem*. In particular, they describe a simple non-artificial problem that simulated annealing or an evolution strategy would hardly solve.

Igel and Toussaint (2003) define some additional quite general constraints on functions that they claim to be important in practice, which induce problem classes that are not c.u.p. In particular, they prove that when the search space has some topological structure, based on a nontrivial neighbourhood relation on the solutions of the search space, and the set of functions fulfils some constraints based on that structure, the problem set cannot be c.u.p. The basic idea is that the permutation closure of any set of functions breaks any nontrivial neighbourhood relation on the search space, and therefore no constraint is fulfilled by every function of the set.

One of those constraints that is largely accepted in most real-world problems is that similar solutions often have similar objective values, which is related with the steepness concept used in Igel and Toussaint (2003), the strong causality condition mentioned therein, and the continuity concept from mathematics. This fact is particularly interesting because it implies that NFL does not apply on the mentioned problems, which leads to algorithms potentially having different performances, as noted by Droste et al. (1999), Igel and Toussaint (2003), Jiang and Chen (2010), and Schumacher et al. (2001). In particular, it leaves open the opportunity to conceive (almost-) general-purpose search algorithms for the set of objective functions that are supposed to be important in practice.

In fact, most general-purpose solvers usually make the previous assumption, i.e., they assume that similar solutions (similar codings in practise, at least under a direct encoding (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Some_Considerations)) are often expected to lead to similar objective values. In particular, in Dembski and Marks II (2009) it is pointed out that *"...problem-specific information is almost always embedded in search algorithms. Yet, because this information can be so familiar, we can fail to notice its presence"*. As some typical examples, we may annotate that knowledge may come from the encoding of solutions or the preference for exploring the neighbourhood of previous good solutions. Besides, we may point out that biology researchers used to admit that similar DNA sequences tend to produce similar transcriptions and finally, similar environmentally attitudinal characteristics. This has led them to suggest that NFL theorems do not apply to the evolution of species (Blancke et al. 2010).

Unfortunately, that assumption has not been proven for the whole set, or a minimally significant portion, of functions with practical interest. By now, one option is to take record of the interesting functions we know that fulfil that assumption (whose counter-acting functions are supposed to be of no practical interest), and approximate the probability of the statement: "The set of real-world problems is not c.u.p." according to the *Laplace's succession law* (Laplace 1814), originally applied to the sunrise certainty problem.

It is worth noting that the possibility of conceiving minimally interesting general-purpose search algorithms for the set of real-world problems is not against the aforementioned almost-NFL theorem (Droste et al. 2002). On the one hand, the set of real-world problems might be even smaller than complexity-restricted scenarios this theorem refers to. And on the other hand, the presence of simple but hard problems with regard to a kind of optimisers does not prevent them to be minimally interesting for the general class of real-world problems.

3 Arbitrary function optimisation framework

This section presents our arbitrary function optimisation framework. Section 3.1 provides its definition and Sect. 3.2 describes three instantiations used for the following experiments.

3.1 Definition

We define *arbitrary function optimisation framework* as an empirical methodology, where a set of algorithms J are compared according to their results on a set of functions F, with the following properties:

- F must be sufficiently large and represents all the characteristics of the problem classes we want to draw conclusions from, by means of proofs or sufficient arguments. This way, biased conclusions are avoided as long as the context of the problem class is mentioned. Ideally, F should be infinite.
- A significant number of simulations of the algorithms in J on *uniform randomly chosen functions of* F are performed (see Fig. 1). In fact, if F is considerably large, an every algorithm-instance simulation methodology is not viable for finite studies. Therefore, a random selection of the functions to be optimised may avoid possible bias on conclusions practitioners may draw. We suggest practitioners to assure that all the algorithms in J tackle exactly the same functions, due to the limitations of random number generators. In Sect. 4.4.3, we address the issue of determining the actual number of necessary simulations.
- Simulation repetitions (to run an algorithm on the same problem instance more than once, with different seeds for the inner random number generators) are not necessary as long as the usage of random number

generators can be assumed to be correct. That means that the result of just one run (with a non-faulting initialising seed for the random number generator) of the algorithm A on the randomly chosen problem instance is sufficient, when the average performance on the problem class F is the subject of study. This assumption is empirically checked below. Nevertheless, practitioners should be advised of the limitations of random number generators they use and therefore, repetitions, with new randomly chosen functions, are always recommended.

Though the basic idea of this framework is simple, arbitrary function optimisation (Fig. 1), it has not been proposed earlier for comparison studies according to the best of our knowledge. There is only one recent work (Jiang and Chen 2010) that empirically and implicitly generates functions randomly from a mathematically defined class of problems. Our innovation is the explicit proposal of a new comparison methodology for any empirical study and an extensive analysis of the framework with regard to the terms of NFL and functions with practical interest.

3.2 Instantiations

In this work, we develop several experiments where different algorithms are analysed on different situations under the proposed framework's rules. These situations are particular instantiations of our framework with concrete problem classes and additional empirical details with the aim of inspecting the possibilities that it is able to provide:

- *Experiment I: A potentially infinite set of representative binary problems from the literature* (Sect. 4). Performance differences between the tested algorithms, independent of the problem instances actually addressed, may appear here.
- *Experiment II: The permutation closure of the previous set* (Sect. 5). This is performed to validate our framework, because it should provide empirical evidence of the NFL theorems, the first one reported so far to the best of our knowledge (regarding the permutation closure of a potentially infinite set of representative problems from the literature).

```
Input: Algorithm A, Problem class F, Additional
   Empirical Conditions EC;
Output: A result of A on F;

1: f ← generate/sample random instance from F;
2: result ← apply A on f according to EC;
3: return  result;
```

Fig. 1 A simulation of an algorithm on the problem class that F represents

- *Experiment III: Real-world binary problems* (Sect. 6). The aim is to find objective evidence of the significance of the NFL theorems on real-world binary problems. Since this set is extremely large and unknown, the use of our framework becomes essential for obtaining results independent of the problem instances actually addressed.

As mentioned earlier, it is not our intention to deny or faithfully support the NFL theorems on any scenario, propose or defend the application of one algorithm as the universal general-purpose solver, nor underestimate the utility of exploiting problem knowledge. On the contrary, our results will just show that: (1) NFL hardly holds on the set of interesting binary problems, (2) performance differences between general-purpose algorithms may appear on that set, and most importantly (3) knowledge exploitation should be evaluated with regard to the performance of competitive general-purpose strategies.

4 Experiment I: standard algorithms and binary problems from the literature

In this first experiment, our aim is to evaluate the possibility of formulating conclusions from an empirical study under the arbitrary function optimisation framework, which are expected to be independent of the actual instances on which algorithms are simulated.

4.1 Benchmark problems

We have analysed a well-defined set of problem classes for this study, *static combinatorial unconstrained single-objective single-player simple-evaluation optimisation problems whose solutions can be directly encoded as arrays of binary variables*, for now on, binary problems. Descriptions for previous terms are provided at (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Benchmark_Problems).

We have taken into account all the binary problem classes we found in the literature at the beginning of our study. Notice that the whole set of binary problems is c.u.p., whereas this is not clear for those appearing in the literature. Table 1 lists their names, references for a detailed definition, problem instances' dimensions, i.e., number of binary variables, and an approximation of the number of potential different instances that could be generated. More detailed comments on the problems are provided in García-Martínez et al. (2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Benchmark_Problems). The code is as well available at http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Benchmark_Problems. Of course, it is likely that there exist

relevant problem classes that have not been included in this study (because we did not know about their existence in the literature). We are aware of that and in Sect. 6 we analyse the dependence between our results and the existence of those binary problems. Nevertheless, it is worth of mentioning that, according to the best of our knowledge, most research papers use an inferior number of problem classes and even a much smaller number of instances in their experiments.

4.2 Algorithms

We have selected five standard algorithms with clear distinctions among each other, to analyse whether our framework is able to detect performance differences among them and allows us to formulate interesting conclusions such as pointing out the reasons for these differences.

It is important to know that our intention is to assess the possibility of presenting clear results on the issue of general-purpose benchmarking for binary problems. It is not our intention to present a winning proposal, nor to identify the best presented algorithm ever for binary optimisation. Thus, neither algorithms' parameters have been tuned at all, nor every presented algorithm or the current state-of-the-art has been applied. Instead of that, standard and well-known algorithms have been applied with reasonable parameter settings. We are sure that better results can be obtained. We think that our framework may be indeed excellent for studies intending to show that concrete optimisation methods are generally better than others or that particular parameter settings outperform some others. However, this is out of the scope of this study.

The selected algorithms are:

- *Random Search* (RS) It just samples random solutions from the search space (with replacement) and returns the best sampled solution. When the search space is sufficiently large, RS rarely revisits solutions in limited simulations, and thus, its search process can be regarded similar to the one of RS without replacement. Therefore, we cannot expect significant averaged performance differences between RS and any other algorithm on set of problems c.u.p., as long as revisiting solutions is unlikely.
- *Multiple Local Search* (MLS) Local search algorithms exploit the idea of neighbourhood to iteratively improve a given solution (continuity), and they are extensively applied for combinatorial optimisation. When dealing with arrays of binary variables, the most widely applied neighbourhood structure is that produced by the one-flip operator, i.e., two binary solutions are neighbours if the second one can be obtained by flipping just one bit of the first one. We will apply a first-improvement strategy when exploring the neighbourhood of the current solution of the local search. MLS starts applying local search on a random sampled solution of the search space. Then, another local search is launched every time the current one gets stuck on a local optimum. At the end of the run, the best visited solution is returned.

Table 1 Binary problems

Problem	Name	n	Instances
1	Onemax (Schaffer and Eshelman 1991)	$\{20,\ldots,1{,}000\}$	$\sum_L 2^L$
2.a	Simple deceptive (Goldberg et al. 1989)	$\{20,\ldots,1{,}000\}$, $L \equiv 0 \pmod 3$	$\sum_L 2^L$
2.b	Trap (Thierens 2004)	$\{20,\ldots,1{,}000\}$, $L \equiv 0 \pmod{36}$	$\sum_L 2^L$
2.c	Overlap deceptive (Pelikan et al. 2000)	$\{20,\ldots,1{,}000\}$, $L \equiv 1 \pmod 2$	$\sum_L 2^L$
2.d	Bipolar deceptive (Pelikan et al. 2000)	$\{20,\ldots,1{,}000\}$, $L \equiv 0 \pmod 6$	$\sum_L 2^L$
3	Max-sat (Smith et al. 2003)	$\{20,\ldots,1{,}000\}$	$<\sum_{L,C_l,N_c}(2L)^{C_l \cdot N_c}$, $C_l \in \{3,\ldots,6\}$, $N_c \in \{50,\ldots,500\}$
4	NK-land (Kauffman 1989)	$\{20,\ldots,500\}$	$<<\sum_{L,k,r} r^{(2^k)\cdot L}$, $r \in [0,1]$, $k \in \{2,\ldots,10\}$
5	PPeaks (Kauffman 1989)	$\{50,\ldots,500\}$	$<\sum_{L,N_p} 2^{L\cdot N_p}$, $N_p \in \{10,\ldots,200\}$
6	Royal-road (Forrest and Mitchell 1993)	$\{20,\ldots,1{,}000\}$, $L \equiv 0 \pmod{L_{BB}}$, $L_{BB} \in \{4,\ldots,15\}$	$\sum_{L,L\_BB} 2^L$
7	HIFF (Watson and Pollack 1999)	$\{4\ldots,1{,}024\}$, $L = k^p$, $k \in \{2,\ldots,5\}$, $p \in \{4,5\}$	$\sum_L L! \cdot 2^L$
8	Maxcut (Karp 1972)	$\{60,\ldots,400\}$	178
9	BQP (Beasley 1998)	$\{20,\ldots,500\}$	165
10	Un-knapsack (Thierens 2002)	$\{10,\ldots,105\}$	54

- *Simulated Annealing* (SA) (Kirkpatrick et al. 1983) is commonly said to be the first algorithm extending local search methods with an explicit strategy to escape from local optima. The fundamental idea is to allow moves resulting in solutions of worse quality than the current solution to escape from local optima. The probability of doing such a move is managed by a temperature parameter, which is decreased during the search process. We have applied a standard SA method with the logistic acceptance criterion and geometric cooling scheme. The temperature is cooled every a hundred iterations by the factor 0.99. The initial temperature is set in the following manner for every simulation: first, two random solutions are generated; we set a desired probability of accepting the worst solution from the best one, in particular, 0.4; then, we compute the corresponding temperature according to the logistic acceptance criterion. We shall remark that, though this temperature initialisation mechanism consumes two fitness evaluations, they have been disregarded when analysing the performance of the algorithm. This way, the subsequent analysis becomes a bit clearer.
- *Tabu Search* (TS) (Glover and Laguna 1997) is among the most cited and used metaheuristics for combinatorial optimisation problems. TS propitiates the application of numerous strategies for performing an effective search within the candidate solution space, and the interested reader is referred to the previous reference. Our TS method implements a short-term memory that keeps trace of the last binary variables flipped. The tabu tenure is set to n/4. Its aspiration criterion accepts solutions better than the current best one. In addition, if the current solution has not been improved after a maximum number of global iterations, which is set to 100, the short-term memory is emptied and TS is initiated from another solution randomly sampled from the search space.
- *Cross-generational elitist selection, Heterogeneous recombination, and Cataclysmic mutation* (CHC) (Eshelman and Schaffer 1991) It is an evolutionary algorithm involving the combination of a selection strategy with a very high selective pressure and several components inducing diversity. CHC was tested against different Genetic Algorithms, giving better results, especially on hard problems (Whitley et al. 1996). So, it has arisen as a reference point in the literature of evolutionary algorithms for binary combinatorial optimisation. Its population consists of 50 individuals.

Source codes are provided at (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Algorithms_2). These algorithms are stochastic methods, and therefore, they apply random number generators initialised with a given seed. We shall remark that the initial seed of the algorithms is different from the seed used to randomly sample the problem instance from our testbed.

4.3 Comparison methodology

In this work, our comparison methodology will consist in comparing the best sampled solution after a given limited number of evaluations; however, other methodologies such as the best result after a maximal computation time are valid as well (out of the NFL context in this case). Every algorithm, for each simulation, will be run with the same budget of fitness evaluations (10^6). For every simulation, we will keep trace of the best visited solution and the instant when it is improved (number of consumed evaluations so far) along the whole run, which lets us to carry out performance differences analysis at different phases of the search process.

Non-parametric tests (Garcia et al. 2009a, b) have been applied for comparing the results of the different algorithms. In particular, mean ranking for each algorithm is firstly computed according to the Friedman test (Friedman 1940; Zar 1999). This measure is obtained by computing, for each problem, the ranking r_j of the observed result for algorithm j assigning to the best of them the ranking 1, and to the worst the ranking $|J|$ (J is the set of algorithms, five in our case). Then, an average measure is obtained from the rankings of this method for all the test problems. Clearly, the lower the ranking, the better the associated algorithm. Second, the Iman and Davenport test (Iman and Davenport 1980) is applied for checking the existence of performance differences between the algorithms, and finally, the Holm test (Holm 1979), for detecting performance differences between the best ranked algorithm and the remainder. These two last statistical methods take as inputs the mean rankings generated according to the Friedman test and will be applied with 5 % as the significance factor.

At this point, we might wonder if NFL theorems still hold on c.u.p. sets when this performance comparison is applied. The doubt may come from the fact that we are applying a performance measure that use *relative* performance differences of more than one algorithm, whereas the original NFL theorems were proved for absolute performance measures of the algorithms, one by one. In fact, Corne and Knowles (2003) showed that NFL does not apply when applying comparative measures for multiobjective optimisation. To solve this issue we have proved the following theorem and corollary at (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#NFL\Holds\on\Friedman\Ranking\Assignment).

Theorem 1 *Given a set of functions c.u.p., and two non-revisiting algorithms A and B, the number of functions where the best result of A is c_A and the one of B* is *c_B is equal to the number of functions where the best result of A* is *c_B and the one of B is c_A).*

Corollary 1 *The Friedman ranking value is the same for all the compared non-revisiting algorithms, and equal to half the number of algorithms plus one, if the function set is c.u.p.*

A maximum of 1,000 simulations per algorithm will be executed, each one with a random problem instance. Every i-th simulation of any algorithm tackles exactly the same problem instance, and probabilistically different from the j-th simulation with $i \neq j$. That is because the sequence of seeds for the different simulations is generated by another random generator instance whose seed is initially fixed and unique per whole experiment. Since our results might be influenced by this initial seed, we will repeat the whole experiment with different initial seeds. We shall remark that some researchers discourage the application of an elevated number of simulations when results are statistically analysed (Derrac et al. 2011). Their claim is that increasing the number of simulations (problems, in case of non-arbitrary function optimisation) usually decreases the probability of finding real performance differences. In our case, under the arbitrary function optimisation framework, we think that performing as many simulations as possible actually decreases the probability of finding performance differences if and only if performance differences do actually not exist, as well as it increases that probability if performance differences actually exist. Therefore, our framework would help researchers to observe the reality through the obtained results. In particular, these claims are tested in two of the experiments in this paper, when performance differences seem to (Sect. 4) and not to appear (Sect. 5).

We may point out that all the experimentation developed in this paper took more than 5 months of running time on a hardware that allowed parallel execution of eight sequential processes. We may even include that memory was another limitation since some processes, those corresponding to Sect. 6, needed in some cases more than three gigabyte of RAM memory. Of course, we do not intend that researchers applying our methodology spend this excessive time on experiments before submitting their papers. In fact, our opinion is that conclusions and experiments must be properly balanced. We just encourage practitioners to support their conclusions under the arbitrary function optimisation framework, i.e., with a large number of problem instances (characterising the subject of study) and a limited number of simulations, each one with a random instance. In our case, we think that our claims and conclusions demanded this thorough study and extensive empirical study.

4.4 Results

This section collects the results of the first experiment and analyses them from different points of view.

All the results are presented in the form of summarising statistical analysis and graphs. In particular, no tables are reported. The main reason is that the raw data relevant for the subsequent analysis and graphs, available at (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Results_3), extended to 3GB of plain text files.

4.4.1 First analysis

Table 2 shows the mean ranking of the algorithms when the best results at the end of the runs (after 10^6 evaluations) are averaged over the 1000 simulations. The Iman Davenport test finds significant performance differences between the algorithms because its statistical value (756.793) is greater than its critical one (2.374) with p value = 0.05. Then, we apply the Holm test to find significant performance differences between the best ranked method (TS) and the others. A plus sign (+) in the corresponding row of Table 2 means that the Holm test finds significant differences between the best ranked algorithm and the corresponding one. As it is shown, TS gets the best ranking and the Holm test finds significant performance differences with regard to all other algorithms.

4.4.2 Dependence with regard to the initial seed

Since these results may be influenced by the initial seed that generates the sequence of 1,000 test problems to be addressed, we have repeated exactly the same experiment with different initial seeds 50 times, i.e., each experiment has used a different sequence of 1,000 test problems involving the same or new problem instances. Figure 2 represents the ranking distributions of the algorithms over the 50 experiments by boxplots, which show the maximum and minimum rankings along with the first and third quartiles.

It can be seen that the rankings of the different algorithms are almost constant values and therefore, their dependence with regard to the initial seed that generates the sequence of problems is very small. This fact indicates that developing experiments on 1,000 random problem instances is more than enough to perceive the real performance differences of the algorithms. In addition, we may point out that the corresponding Iman-Davenport and Holm analysis always found significant differences between the

Table 2 Rankings and Holm test results

Algorithm	Ranking	Sig
TS	2.052	Winner
SA	2.546	+
MLS	2.758	+
CHC	2.874	+
RS	4.769	+

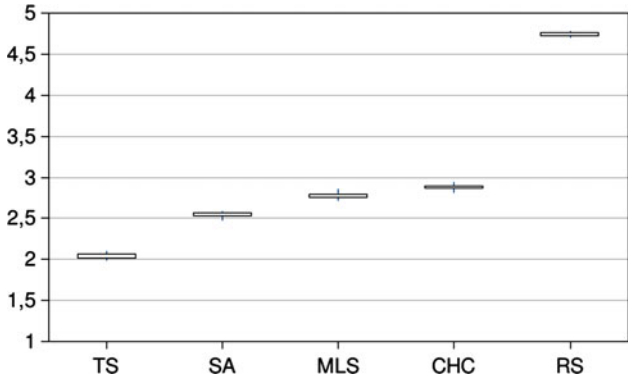

Fig. 2 Rankings distributions on 50 experiments

best ranked algorithm and every other algorithm. According to this result, we may conclude that practitioners following our proposed methodology do not need to repeat the whole experiment many times, though a minimal number of repetitions (from two to five) is convenient to check the independence with regard to the initial seed.

4.4.3 Dependence with regard to the number of simulations

We address now the number of simulations needed to obtain the results in previous experiments. Since every previous experiment executed a total of 1,000 simulations per algorithm, we wonder if similar results can be obtained with a reduced number of simulations. Figure 3a presents the ranking evolution of the algorithms as long as the number of simulations is increased. Each line corresponds to the averaged value of the ranking of the algorithm over the previous 50 experiments. The areas that go with the lines cover every ranking value of each algorithm over the 50 experiments, from the highest ranking ever obtained to the lowest on these 50 experiments.

We can see that at the left of the graph, when very few simulations have been performed (less than 5), the areas are wide and overlap one another, meaning that the rankings of the algorithms are very changeable across the different 50 experiments. That is due to the initial seed of each experiment generates different sequences of problems to be solved. In some cases, some algorithms are favoured because they deal well with the selected problems, getting good rankings, whereas the other algorithms do not; and in some other cases, the opposite occurs. As long as more simulations are carried out per experiment, areas get narrower, which means that rankings become more stable leading to the appearance of significant differences on the previous statistical analysis. In addition, we may observe that there is no necessity of many simulations to visually appreciate performance differences: TS is clearly the best ranked algorithm from 100 simulations onward, and RS is clearly the worst ranked one from ten simulations onward (just the number of problem classes). Ranking evolution inspection is suggested for practitioners applying our framework to be able to reduce the number of simulations.

4.4.4 Online analysis

Finally, we study the averaged performance of the algorithms along the runs, i.e., according to the number of consumed evaluations. We will call this measure the *online performance*. Figure 3b shows the mean rankings (over 1,000 simulations) of the online performance of the algorithms. As done previously, lines are for the averaged value of the mean rankings over 50 experiments, and areas cover all the mean rankings values, from the highest to the lowest on these 50 experiments. Notice that the online ranking value is not monotonous as convergence graphs use to be. That is because ranking is a relative performance measure, and thus, if all the algorithms improve their results except the algorithm A, then, the ranking value of A deteriorates.

The first noticeable observation is that areas that go with the averaged mean ranking values are extremely narrow from the very beginning (even the first ten evaluations) until the end of the runs. That means that averaged online performances (mean over 1,000 simulations) are almost always the same, i.e., the general online performance depends solely on the algorithms' heuristics, and not on the sequence of problems actually tackled (considering our testbed).

All the algorithms start with the same averaged ranking value, 3, because we forced every algorithm to start with the same initial solution, generated randomly according to the random number generator and the same seed. Subsequently, algorithms seem to go through the following three stages (a much deeper analysis is provided at (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Online_analysis).

– *Beginning* Algorithms perform their initial steps, which are deeply influenced by their base characteristics, and are not expected to provide high-quality solutions. At this stage, algorithms have not developed their full potential yet. We may summarise this state as follows:

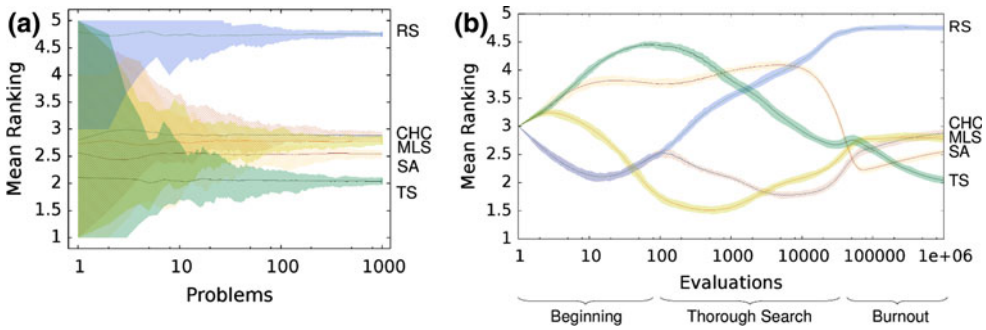

Fig. 3 Evolution of ranking values: **a** as the number of simulations increases and **b** according to the consumed evaluations

sampling random points from the search space (RS and CHC) is better than exploring the neighbourhood of one solution (MLS, SA, and TS).

- *Thorough Search* Algorithms start applying their heuristics continuously (notice the logarithmic scale) looking for better solutions. At this point, efficient heuristics attain the best rankings (MLS, CHC, *"and SA"*), and the application of no heuristic (RS) leads to worse results.
- *Burnout* Algorithms have already developed all their potentials and have reached good solutions. This fact makes the subsequent progress more difficult, and algorithms burn their last resources on the event, each time more unlikely, of finding a new best solution. Thus, their ranking values use to deteriorate or remain constant. Notice that the logarithmic scale implies that the mentioned last resources are still very large, in particular, around the 90 % of the run. The diversification biased search of TS is the only algorithm that seems to avoid a burnout state, still improving its results. On the other hand, the lack of heuristic of RS prevents it to attain the quality of the solutions of its competitors, though it unlikely revisits solutions.

4.5 Discussion

Based on the previous results, we may conclude that there exists empirical evidence that suggests that NFL theorems do not hold on our set of problems. It has been clear that performance differences between the algorithms may and do appear on the randomly selected problem instances. However, since our testbed is not c.u.p., performance differences were certainly expected.

According to our results, TS generally attains the best results at 10^6 evaluations. However, we observed that other algorithms achieved better results at an inferior number of evaluations. What should we expect for simulations longer than 10^6 evaluations? On the first hand, we shall remind that Fig. 3b used a logarithmic scale, and therefore, TS dominated the best ranking values from around the 10^5th evaluation, i.e., the 90 % of the run. So, if rankings might change, they would unlikely occur on simulations with a reasonable number of evaluations (let us say that more than 10^6 evaluations starts becoming unreasonable). On the other hand, having analysed the behaviour of the algorithms in Sect. 4.4.4, it seems difficult for algorithms that begin to revisit solutions to overtake TS. To the best of our knowledge, we might only forecast ranking changes in the rare event that the probability of randomly sampling the global optimum (by RS) was higher than the probability for TS to iteratively avoid revisiting solutions. In that case, the ranking of RS might become equal or even better than the one of TS. However, we strongly think that this event would happen on unreasonably long simulations.

Much more relevant than previous conclusions is the fact that they have been formulated without knowing the actual set of problems addressed. Therefore, we may certainly conclude that our arbitrary function optimisation framework allows researchers to formulate significant conclusions that are independent of the particular set of functions used, as long as the context of the problem class is mentioned (in our case, static combinatorial unconstrained single-objective single-player simple-evaluation binary optimisation problems).

Finally, we end this discussion providing some guidelines for practitioners dealing with problem solving by means of metaheuristics. Recently, some researchers have claimed that *"many experimental papers include no comparative evaluation; researchers may present a hard problem and then present an algorithm to solve the problem. The question as to whether some other algorithm could have done just as well (or better!) is ignored"* (Whitley and Watson 2005). It is supposed that this is not the way to go. We propose that researchers presenting a new general-purpose method develop an empirical study similar to ours and always applying the corresponding state-of-the-art approach as the baseline. However, new approaches incorporating knowledge for concrete problems should be compared with the corresponding state-of-the-art general-purpose algorithm as well. In fact, researchers must prove that the problem knowledge their proposal

exploits lets it to attain better results than the general-purpose solver, which is supposed to exploit less problem knowledge. Otherwise, the specialisation by using that problem knowledge is useless. In the case of binary optimisation, either context-independent or problems with a natural binary encoding, researchers should, from now on, compare their approaches with regard to CHC, SA, and TS at different search stages (and if it is possible, some more recent context-independent solvers claimed to be competitive (García-Martínez and Lozano 2010; García-Martínez et al. 2012; Gortázar et al. 2010; Rodriguez et al. 2012), although our empirical framework had not been followed), until a new better general-purpose method is found. In any case, we claim that our empirical framework should be followed, i.e.,

1. The number of problem instances must be as large as possible.
2. Simulations should deal with randomly selected problem instances.
3. Enough simulations and repetitions with different sequences of problems (initial seed) should be performed.
4. Performance differences analysis at different search stages are recommended.
5. Care on the independence between the seeds of algorithms, problem generators, simulations, and experiments must be taken.

5 Experiment II: NFL on a closed under permutations problem set

In this section, we validate our empirical framework with regard to the NFL implications for c.u.p. problem sets, i.e., we expect the framework to make experiments to show that the performance of any two algorithms is the same when c.u.p. sets are analysed (at least, for non-revisiting algorithms). This kind of empirical NFL evidence poses a real challenge because c.u.p. sets are usually excessively large (and so experiments would be). In fact, it has not been reported previously, to the best of our knowledge, at least for standard problem sizes [Whitley and Watson (2005) present some results for a problem with three different candidate solutions].

To carry out our goal, we will repeat our previous experiments on a *pseudo*-closed under permutations (PCUP) set of problems. In particular, we have devised a procedure to obtain a PCUP set from any set of binary problems. This procedure samples a random problem from the original set and wraps a function implementing a random permutation of the search space around it. Detailed considerations about the PCUP procedure are provided at (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Some_Considerations). The concrete procedure performs the following steps:

1. First, seeds for the PCUP procedure and the original problem are provided.
2. Second, the new PCUP problem is given to the solver.
3. Each time a solution must be evaluated, a new random solution is sampled from the search space according to the seed of the PCUP procedure and the original solution, i.e., given a particular seed, the PCUP procedure defines a deterministic function that maps original solutions to random solutions.
4. The random solution is evaluated according to the original problem.
5. The fitness obtained is provided to the algorithm as the fitness value of the original solution.

5.1 Empirical framework

We have performed experiments on a PCUP set of problems where original ones are taken from the set described in Sect. 4.1. However, the number of binary variables had to be limited to 32, because of the time and memory requirements of the PCUP procedure. In particular, each time a problem instance was selected, it was automatically reduced by optimising the first n' variables, with n' randomly selected from $\{20, \ldots, 32\}$. When needed, restrictions on the length of the binary strings were imposed. The empirical methodology is the one depicted in Sect. 4.3 (under the arbitrary function optimisation framework), but, in this case, global experiments repetitions have been limited to 10.

In this case, two set of algorithms have been used separately. On the one hand, we have applied the same algorithms presented in Sect. 4.2, referenced to as original algorithms. On the other hand, we have applied previous algorithms having incorporated a memory mechanism that lets them avoid revisiting solutions. In particular, every new candidate solution is first tested against the memory mechanism. If that solution was previously visited, a completely new solution is provided to the algorithm, until the search space has not completely been explored. Otherwise, the original candidate solution is returned to the algorithm. These algorithms will be referenced to as non-revisiting ones. The reader must realise that the applied memory mechanism alters the behaviour of the algorithms slightly. For instance, regarding the original SA, a rejection of solution s_j from solution s_i does not prevent the algorithm accepting s_j in the future. However, when SA applies our memory mechanism, the first rejection of any solution s_j prevents the algorithm for evaluating that solution again in any future event.

5.2 Results

Figure 4 presents the results on the PCUP set of problems: Fig. 4a and c shows the ranking evolution of both sets of algorithms, original and non-revisiting ones, respectively, as long as the number of simulations per experiment, i.e., problems, is increased; Fig. 4b and d depict the online performance of both sets of algorithms, i.e., the averaged value of the mean rankings along the number of consumed evaluations. Each line corresponds to the averaged value of the rankings of the algorithms and the areas that go with the lines cover every ranking value, both over ten experiments with different initial seeds.

Analysing these figures, we may remark that

- when averaging over many PCUP problem instances (Fig. 4a, c), there seems to exist performance differences between the original algorithms and there have not between the non-revisiting versions. In fact, the corresponding statistical analysis (not reported here), by means of the Iman-Davenport and Holm tests, supported these impressions.
- Online performance is almost constant and equal for every algorithm, except SA and CHC (this latter from 10^5 evaluations).

5.3 Discussion

The fact that the online performance of the algorithms are constant and equal (Fig. 4d and RS, MLS, and TS in 4b) means that there seems not to exist any moment along the run, i.e.,any search stage characterised by the operations carried out by the solvers (see Sect. 4.4.4), at which performance differences appear. In other words, no matter the heuristics that solvers implement and apply at any event, averaged superiority is impossible. In fact, NFL theorems do not just deal with the performance of the algorithms at the end of the runs, but with the complete sequence of discovered points, i.e., no matter the analysed length of the *traces* (output of visited fitness values), the averaged performance is the same for any two algorithms.

The performance differences of CHC and SA with regard to TS, MLS, and RS (Fig. 4a, b) may be explained by the hypothesis that says that the probability of revisiting solutions of these two methods is higher, and thus, their performances are expected to be inferior (Marshall and Hinton 2010). More details on this hypothesis are provided at (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Discussion).

It should be clear that these experiments give the empirical NFL evidence we expected initially. Therefore, we may conclude that our arbitrary function optimisation framework is really able to approximate the real averaged performance of different search algorithms, either when there are or there are no differences between them (Sects. 4.4 and 5.2, respectively) and even when the testbed is much larger than the allowed number of simulations, or the problem instances actually addressed are not known.

Finally, the interested reader might like an explanation for the thin line in Fig. 4d connecting evaluations 1 and 2. The reason is that, our memory mechanism was initialised with two solutions, the initial one, which is forcedly the same for every algorithm, and its bit-level complement. These two solutions are evaluated and reported. Therefore, this fact makes every implemented non-revisiting algorithm to sample the same two first solutions, and

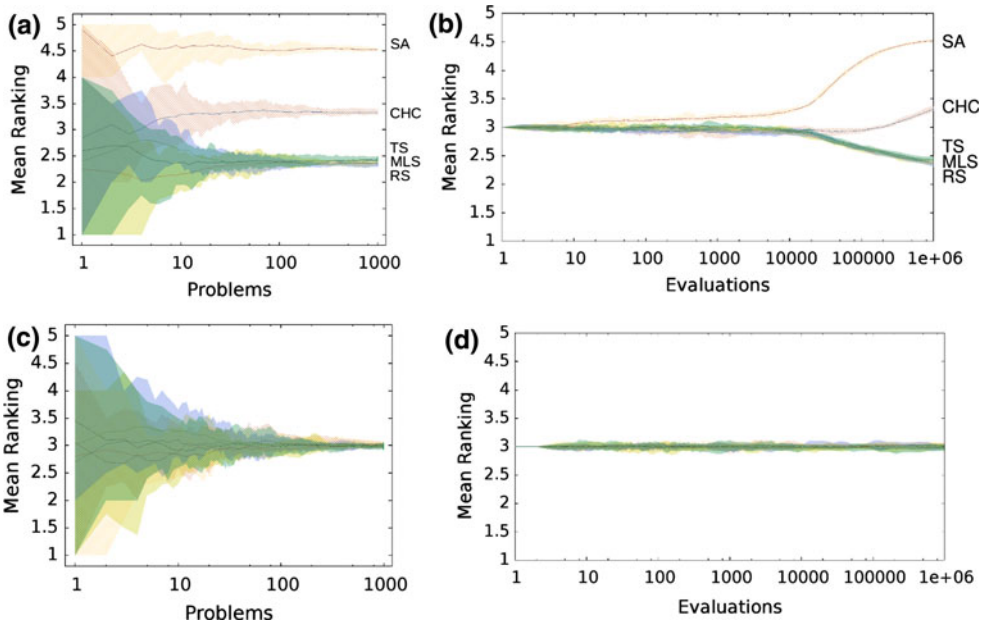

Fig. 4 Results on the PCUP set of problems: **a** and **c** show the ranking evolution of both sets of algorithms, original and non-revisiting ones, respectively, as long as the number of problems is increased; **b** and **d** depict the corresponding online performance of both sets of algorithms

consequently, to get exactly the same ranking values at these two steps.

6 Experiment III: NFL hardly holds on the set of interesting binary problems

In this section, we focus the study on the set of interesting binary problems. Our aim is to assess if our arbitrary function optimisation framework is able to shed some light on the possible validity of the NFL implications on the set of real-world binary problems. Many researchers have suggested that these theorems do not really hold on real-world problems (Igel and Toussaint 2003; Jiang and Chen 2010; Koehler 2007). However, sententious proofs have not been given so far, or they have analysed a very reduced class of problems.

Due to the limitation that we cannot perform experiments on the whole set of interesting binary problems (because none really knows it), we apply the following methodology:

1. To perform experiments, under the arbitrary function optimisation framework, on a subset of problems that is supposed to be representative of the real set that we want to analyse, i.e., interesting binary problems (Sect. 6.1).
2. Given the results on the subset of problems, to analyse the implications for the NFL theorems to hold on the real set we want to draw conclusions from (Sect. 6.2).
3. Finally, to compute the probability for the NFL to hold by means of Laplace's succession law (Laplace 1814), i.e., according to the number of elements in the subset that do or do not support the NFL implications (Sect. 6.3).

6.1 Experiments on a subset of problems

At this point, we have to assume that binary problems appearing in the literature, those with an a priory definition, may represent the real set of binary functions with practical interest, at least with a minimally sufficient degree. The reader may understand that this assumption, though unproven, is completely necessary. The contrapositive argument means that binary problems appearing in the literature do not represent binary real-world problems at any degree, and therefore, research by now would have been just a futile challenging game. It is worth remarking that there exist several problems in the literature that were artificially constructed to give support to the NFL (like those in the study of Sect. 5); however, these functions lack of a definition out of the NFL context and a clear understanding on their own.

Therefore, we perform simulations of the non-revisiting algorithms on the original set of problems described in Sect. 4.1 and under the methodology described in Sect. 4.3. Due to excessive running times, the dimension of the problems were limited to $\{20, \ldots, 32\}$ as for the experiments of Sect. 5. Besides, 500 simulations were performed per experiment, 10 experiment repetitions, and each simulation was given a maximum of 200.000 evaluations. Figure 5a depicts the online performance of the algorithms. As a minor comment, notice that, with the exception of SA, Figs. 5a and 3b are quite similar, i.e., the memory mechanism does not change the characterising conduct of the methods. A possible explanation for the slightly different performance of SA was commented in Sect. 5.1.

6.2 Analysis of the NFL implications

Assuming NFL theorems on the set of binary real-world problems implies that the real averaged online performance of the compared non-revisiting algorithms resembles the graph in Fig. 4d. It is clear that Figure 5a differs from the expected behaviour. Therefore, if NFL is assumed there must exist a set of interesting *counter-acting* binary problems that balances the behaviour shown in Fig. 5a, i.e., the online performance of the algorithms on the counter-acting problems can be deduced (no experiments were performed) as that presented in Fig. 5b. Though we do not know the set of counter-acting problems, we can analyse the performance of the algorithms' heuristics on them, and get characterising features of this set. Among others, we notice that:

- First, if you are given too few evaluations, exploring the neighbourhood of a completely random solution (MLS, SA, and TS) should be probabilistically better than sampling random solutions from the whole search space (RS and initialisation in CHC). In addition, you would do better by exploring the whole neighbourhood of that solution (which is what best-improvement strategy does in TS), than exploring the neighbourhoods of new solutions you may find during that process (which is what SA and first-improvement strategy of MLS do). Even more, exploring iteratively the neighbourhoods of better solutions you find (MLS) seems to be the worst thing you can do, and you would do better by exploring neighbourhoods of randomly picked solutions (SA).
- As soon as you dispose of enough evaluations, ignore everything and just apply RS. Any other heuristic similar to the ones in this work leads to worse results. In fact, according to the NFL, the heuristics that helped the algorithms to produce a fruitful search on the original problems are the ones responsible for the ineffective performance on the counter-acting ones. This is the main reason for RS attaining the best ranking values. Additionally, it is interesting to see that SA makes

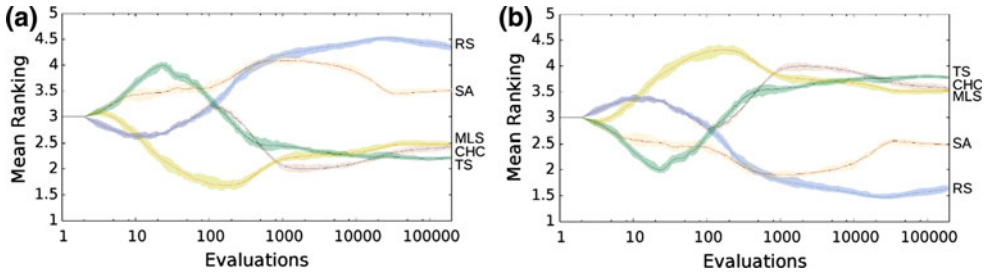

Fig. 5 Online performance (**a**) on a representative subset of problems and (**b**) on the corresponding counter-acting one

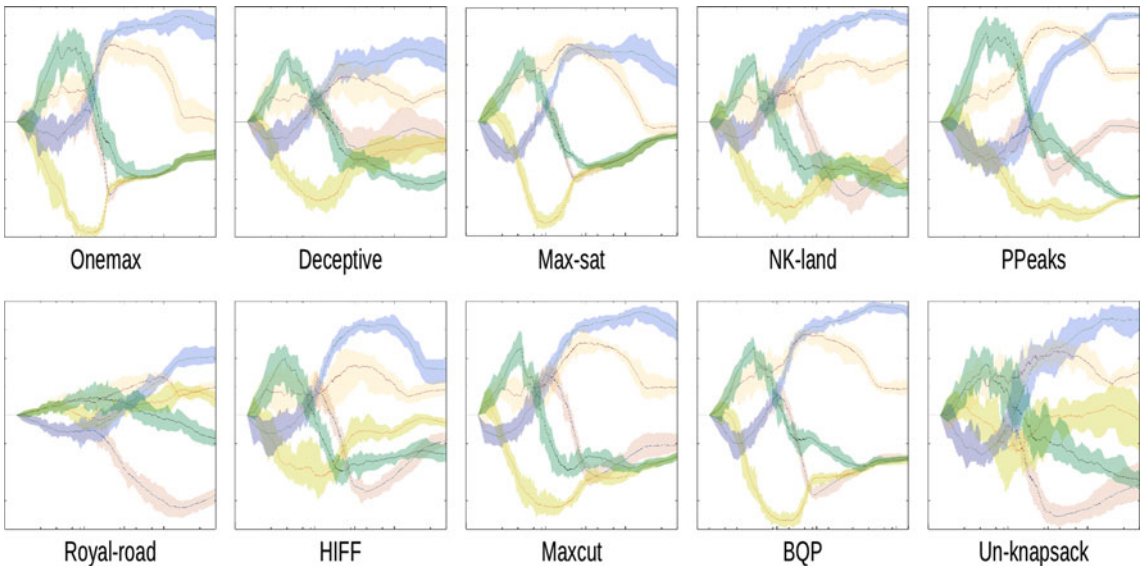

Fig. 6 Online performance per problem class

a good progress while its temperature is high, i.e., while it is randomly wandering through the search space. As soon as that temperature comes down, i.e., the wandering is biased towards better solutions (approximately after 10^5 evaluations), the ranking gets worse.

– Ignore everything about population diversity and search diversification (though this idea, which is a particular interpretation of Fig. 5b, may be certainly criticised, it is mentioned here with the aim of showing researchers that NFL theorems may profoundly shake the foundations of any heuristic). In fact, we may see that CHC makes good progress when it is supposed to be fighting the genetic drift problem (although that event might not be really occurring on that testbed). And, on the other hand, the metaheuristic aimed at carrying out a diversified intelligent search process (TS) is never making its ranking better.

6.3 Application of Laplace's succession law

Next, we compute the probability of finding one problem class belonging to such a counter-acting testbed. Since we have used a wide set of binary problem classes, we should expect our testbed to contain one or several classes belonging to such a counter-acting one. Even though all these problem classes have been regarded for computing the averaged online performance in Fig. 5a, it should not be strange to find *outlier* classes differing from the mean behaviour and resembling Fig. 5b, if the NFL really applies. In particular, we look for problem classes for which RS gets poor ranking values at the beginning and becomes one of the best at the end.

Figure 6 shows the online performance of non-revisiting algorithms on previous simulations, but separately according to the problem class tackled. We notice that, the online performance graphs of every problem class is more similar to the one in Fig. 5a than the one in Fig. 5b, i.e., none of the problem classes facilitates the emergence of algorithms' conducts similar to the ones for the counter-acting problems. Even the one corresponding to the royal-road class, which differs the most, shows that RS is good at the beginning and the worst at the end.

Therefore, given the fact that we have not found any binary problem class from the literature that shows the counter-acting behaviour, and we do not know the complete set of interesting binary problems (which may contain

counter-acting and non-counter-acting problem classes), we may approximate the probability for finding a counter-acting problem class according to the Laplace's succession law (Laplace 1814), i.e., $(N^c + 1)/(N + 2) = 1/(N + 2) = 1/12 \simeq 8\ \%$ (N^c is the number of counter-acting problems found, 0, and N is the number of problem classes used, 10). Besides, if we tried to prove the NFL theorem by adding new problems to our testbed, we should make the probability of sampling a counter-acting problem equal to the probability of sampling one of the original problems (Igel and Toussaint 2004). Since the probability of sampling any problem class is the same, we would need to include ten counter-acting problem classes. Then, we can approximate the probability of finding empirical evidence for the NFL theorems on the real-world problems, as the probability of finding ten interesting and counter-acting problem classes, i.e., $1/12^{10} \simeq 1.6e{-}11$.

Therefore, we may conclude that it is almost impossible for the NFL to hold on the set of binary problems from the literature. Subsequently, having made the assumption that binary problems from the literature are representative of the real interesting binary problems set, then, we conclude that it is almost impossible for the NFL to hold on the set of interesting binary problems. Finally, we may appoint that, according to our experience, performing more experiments on new binary problem classes that likely promote the algorithms' conducts shown in Fig. 5a (instead of those shown in Fig. 5b), either because a direct encoding is available, similar solutions are expected to have similar objective values, or any other reason, would simply make the above probability much smaller.

7 Lessons learned

The application of our arbitrary function optimisation framework in this work has allowed us to better understand the implications and limitations of the NFL theorems. This has been possible thanks to the advantage of our framework of letting researchers to formulate conclusions less dependent on the actual set of benchmark functions. In particular, we may appoint the following main results:

- *NFL theorems do not likely apply on the set of interesting binary problems* We have shown that it is necessary, for the NFL to hold, the existence of a set of counter-acting functions, whose implications on the online performance of the algorithms has been described in Sect. 6.2. However, we have not found any binary problem class from the literature, out of 10, that promoted such kind of conditions. Therefore, we have applied Laplace's methodology for computing the probability for the sun to rise once again and have approximated the probability of finding a counter-acting (non-NFL-biased) binary problem as 8 %. Finally, we have computed the probability for the NFL to hold as the probability of finding ten counter-acting problem classes, which is $1.6e{-}11$.
- *General-purpose search algorithms really apply common problem knowledge* This is a fact that other authors had pointed out previously (Dembski and Marks II 2009). However, it was still unproven if that problem knowledge was effective on large sets of common problem classes. According to our experience, there are two sources of knowledge, not to be underestimated, that general-purpose algorithms may effectively apply (at least on the set of interesting binary problems):
  - *Natural encoding* It is the assumption that the communication rules for the algorithm and the problem are the same, so that variables on which the algorithm works on are actually the relevant variables of the problem, and there is a certain independence degree between them. For further discussion on natural encoding, please refer to (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL#Some\Considerations). We can summarise this idea with the question, "do algorithm and problem speak the same language"?
  - *Continuity* It is the classic assumption that similar inputs produce similar outputs, so similar solutions are expected to have similar objective values. In fact, every metaheuristic applying neighbourhood-based operators (such as local search approaches or TS) strongly relies on this assumption. From this study, it has been clear that not every general-purpose method exploits that knowledge equally well, and therefore, performance differences may really appear on the set of real-world problems. This fact is due to each metaheuristic incorporating some other kinds of knowledge that may be relevant as well, such as population diversity is important, high-quality solutions are usually clustered in prominent regions of the search space, or to reach the global optimum it is necessary to scape from local optima, among others.
  - *The permutation closure of a problem class invalids the common sources of knowledge* Igel and Toussaint (2003) showed that the permutation closure of a problem class breaks the continuity hypothesis on the one hand. On the other hand, it involves some considerations on the encoding scheme for candidate solutions that are discussed at (García-Martínez et al. 2011a, http://www.uco.es/grupos/kdis/

kdiswiki/index.php/AFO-NFL#Some_Considerations). In fact, any permutation of the search space, except the identity, implies that natural encoding is not available because the problem speaks a different language. Knowing that language would become a source of information that allowed the algorithm to use the natural encoding, but it would not be a general-purpose method anymore.

- *To revisit previous solutions, when that implies re-evaluations, should be avoided in general* At least in static problems, our results provide empirical evidence that revisiting solutions makes the algorithm's progress slower, on either set of representative problems (Sect. 4.4.4) or PCUP sets (Sect. 5.2). This evidence is in part in agreement with the results of Marshall and Hinton (2010). The main difference is that we have covered set of problems not c.u.p. as well, where we have concluded that RS may not be the best approach.
- *Random search is not an option for most real-world problems* Several publications from the context of NFL claim that practitioners should apply RS along with any other approach for solving particular problems, at least for comparison issues. As it is shown in Sect. 4, when natural encoding availability and continuity are reasonable assumptions, the application of no heuristic (RS) is probably worse than the application of any rational heuristic, such as neighbourhood-based explorations. This fact does not mean that new proposals do not have to be evaluated with regard to RS, and even less with regard to existing competitive approaches, but that none should expect RS to provide competitive results in this case.
- *Specialised methods must be critically analysed*: Some researchers claim that "*too many experimental papers (especially conference papers) include no comparative evaluation; researchers may present a hard problem (perhaps newly minted) and then present an algorithm to solve the problem. The question as to whether some other algorithm could have done just as well (or better!) is ignored*" (Whitley and Watson 2005), and the suspicion is certainly not new (Barr et al. 1995; Hooker 1995). With the aim of promoting an interesting research progress, authors should prove somehow that the proposals they present are characterised by the effective and efficient usage of the knowledge they are supposed to be exploiting. Thus, we think that specialised approaches must precisely show advantages against the application of general-purpose (state-of-the-art) methods. In fact, when no advantages are found, we have to question if the problem knowledge the method is supposed to be exploiting is or is not more relevant than the little knowledge the general-purpose method uses.

8 Future challenges

We think that this line of research is really worth studying further. In fact, questions addressed in this work are not solved in other fields, where they pose complex challenges. We may appoint the following:

- *Real parameter optimisation* In the past years, many methods for this kind of problems have been devised, and different benchmarks have been proposed with the aim of clarifying the knowledge of the field (García-Martínez et al. 2011b, Hansen 2005; Herrera et al. 1998; Lozano et al. 2011). Though deterministic simulations on standard set of benchmark functions have promoted some consensus among this research community, there are still some others that dare question the independence between the conclusions formulated and the benchmark tackled. We think that our arbitrary function optimisation may become an excellent tool for dispelling those doubts. On the other hand, though NFL theorems are not valid in pure real-parameter optimisation (Auger and Teytaud 2007, 2008) because the search space is infinite, they may still hold for the way it is usually dealt with. As Whitley and Watson (2005) claim, "*As soon as anything is represented in a computer program it is discrete. Infinite precision is a fiction, although it is sometimes a useful fiction*".
- *Integer programming* This field is very large and covers problems of very different nature (Chen et al. 2010; Jünger et al. 2009). It has been proved that some heuristics are powerful on some kind of problems whereas they perform poorly with regard to others in other scenarios (though there may exist polynomial procedures that map one problems into others). These results support the idea that good performance unavoidably requires the exploitation of some specific problem knowledge, and thus, no sufficiently effective general-purpose approach exists in this field. On the other hand, it has been proved that NFL theorems are not valid for many particular problem classes of this kind (Koehler, 2007). Therefore, NFL theorems might not hold on more general sets of this kind of problems. This latter hypothesis does not suggest that there certainly exists an effective general-purpose solver, but that several minimally general-purpose approaches might provide reasonable results for different groups of problems of this kind. We devise that this possibility could be really interesting in at least two different situations:

  - If you are given a new integer programming problem you know little information about, you could develop a first approximation by applying this kind of general-purpose approaches. Then, results could help you to analyse the problem and locate the specific pieces of problem knowledge that lead to success.
  - If you think you already know the problem knowledge that can be exploited to effectively and efficiently solve the problem, you can prove your certainty by comparing your results with regard to those of more general effective approaches.

  In any case, we suggest our arbitrary function optimisation framework to be assumed in order to formulate conclusions as less dependent on the problem instances as possible.
- *Multiobjective optimisation* Several researchers have already pointed out that multiobjective optimisation problems are not out of the NFL concerns (Corne and Knowles 2003; Service, 2010). However, the particular case of dealing only with multiobjective problems with practical interest has not been studied yet. We may think that a study similar to the one in Sect. 6, but with multiobjective problems, may shed some light for this case.
- *Hybrid algorithms, ensembles, hyper-heuristics, and others* Many researchers have presented search models that combine several approaches in an attempt to overcome their individual limitations and benefit from their respective advantages (Blum et al. 2011; Lozano and García-Martínez 2010; Talbi 2002). Interestingly, some publications argue the combination as a medium for escaping from the NFL's claws (and in some cases, the combination is not even analysed with regard to the sole application of one of the approaches). Recently, Dembski and Marks II (2010) showed that NFL theorems apply to the concept of higher-level searchers, and thus, to combinations of algorithms as well. As for the multiobjective case, designing new algorithms as the combination of previous ones that perform more effectively and efficiently is still a possibility when regarding just the set of problems with practical interest.
- *Empirical studies in general* Finally, we may assume that our proposed framework is sufficiently general to be applicable on almost any empirical context, such as biological or industrial ones, as long as resources allow so. The general idea is, given two or more models to be compared,
  - it is desirable to dispose of an elevated number of scenarios (potentially infinite).
  - Sufficient simulations of the models are performed.
  - Each simulation applies one of the models on an uniform randomly sampled scenario.

9 Conclusion

In this paper, we have presented the arbitrary function optimisation framework as an empirical methodology for comparing algorithms for optimisation problems.

We have proved that the application of our arbitrary function optimisation framework allows researchers to formulate relevant conclusions that are independent of the problem instances actually addressed, as long as the context of the problem class is mentioned. In fact, our framework has allowed us to develop the first thorough empirical study on the NFL theorems, to the best of our knowledge, which has shown that NFL theorems hardly hold on the set of binary real-world problems. In fact, we have approximated the probability of the opposite to the value $1.6e{-}11$.

Finally, we have collected the lessons learned from our study and presented challenges for other connected research fields.

Acknowledgments Beliefs usually need to be critically analysed before becoming real knowledge. Being loyal to this idea, the authors would like to express that this study would not have been initiated without the fact that, their journal submissions proposing new approaches, and analysed on many different kinds of problems, were sometimes rejected on the claim that *''according to the NFL, if your proposal wins, then it loses on the rest of problems that have not been analysed''*. Therefore and being honest with ourselves, this study, we are really glad of having developed, is in part thanks to the corresponding reviewers and deciding editors' comments that put us on the way.

References

Auger A, Teytaud O (2007) Continuous lunches are free. In: Proceedings of the genetic and evolutionary computation conference, ACM Press, New York, pp 916–922

Auger A, Teytaud O (2008) Continuous lunches are free plus the design of optimal optimization algorithms. Algorithmica 57(1):121–146. doi:10.1007/s00453-008-9244-5

Barr RS, Golden BL, Kelly JP, Resende MG, Stewart WRJ (1995) Designing and reporting on computational experiments with heuristic methods. J Heuristics 1:9–32

Beasley J (1998) Heuristic algorithms for the unconstrained binary quadratic programming problem. Techical report, The Management School, Imperial College

Blancke S, Boudry M, Braeckman J (2010) Simulation of biological evolution under attack, but not really: a response to Meester. Biol Philos 26(1):113–118. doi:10.1007/s10539-009-9192-8

Blum C, Puchinger J, Raidl GR, Roli A (2011) Hybrid metaheuristics in combinatorial optimization: a survey. Appl Soft Comput 11(6):4135–4151

Chen DS, Batson R, Dang Y (2010) Applied integer programming: modeling and solution. Wiley, Chichester

Corne DW, Knowles JD (2003) No free lunch and free leftovers theorems for multiobjective optimisation problems. Evol Multi

Crit Optim LNCS 2632:327–341. doi:10.1007/3-540-36970-8_23
Dembski WA, Marks II RJ (2009) Conservation of information in aearch: measuring the cost of auccess. IEEE Trans Syst Man Cybernet Part A 39(5):1051–1061. doi:10.1109/TSMCA.2009.2025027
Dembski WA, Marks II RJ (2010) The Search for a aearch: measuring the information cost of higher level search. J Advan Comput Intell Intell Informatics 14(5):475–486
Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1(1):3–18
Droste S, Jansen T, Wegener I (1999) Perhaps not a free lunch but at least a free appetizer. In: Proceedings of the genetic and evolutionary computation conference (GECCO'99), Morgan Kaufmann, pp 833–839
Droste S, Jansen T, Wegener I (2002) Optimization with randomized search heuristics—the (A)NFL theorem, realistic scenarios, and difficult functions. Theor Comput Sci 287(1):131–144
Eshelman L, Schaffer J (1991) Preventing premature convergence in genetic algorithms by preventing incest. In: Belew R, Booker L (eds) International conference on genetic algorithms. Morgan Kaufmann, San Mateo, pp 115–122
Forrest S, Mitchell M (1993) Relative building block fitness and the building block hypothesis. In: Whitley L (ed) Foundations of genetic algorithms 2. Morgan Kaufmann, San Mateo, pp 109–126
Friedman M (1940) A comparison of alternative tests of significance for the problem of m rankings. Ann Math Stat 11(1):86–92
Garcia S, Fernández A, Luengo J, Herrera F (2009a) A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. Soft Comput 13(10):959–977
Garcia S, Molina D, Lozano M, Herrera F (2009b) A Study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization. J Heuristics 15(6):617–644
García-Martínez C, Lozano M (2010) Evaluating a local genetic algorithm as context-independent local search operator for metaheuristics. Soft Comput 14(10):1117–1139
García-Martínez C, Lozano M, Rodriguez FJ (2011a) Arbitrary function optimization. No free lunch and real-world problems. http://www.uco.es/grupos/kdis/kdiswiki/index.php/AFO-NFL
García-Martínez C, Rodríguez-Díaz FJ, Lozano M (2011b) Role differentiation and malleable mating for differential evolution: an analysis on large-scale optimisation. Soft Comput 15(11):2109–2126. doi:10.1007/ s00500-010-0641-8
García-Martínez C, Lozano M, Rodríguez-Dìaz FJ (2012) A simulated annealing method based on a specialised evolutionary algorithm. Appl Soft Comput 12(2):573–588
Glover F, Laguna M (1997) Tabu search. Kluwer Academic Publishers, Norwell
Goldberg D, Korb B, Deb K (1989) Messy genetic algorithms: motivation, analysis, and first results. Complex Syst 3:493–530
Gortázar F, Duarte A, Laguna M, Martí R (2010) Black box scatter search for general classes of binary optimization problems. Comput Operat Res 37(11):1977–1986. doi:10.1016/j.cor.2010.01.013
Hansen N (2005) Compilation of results on the CEC benchmark function set. Technical report, Institute of Computational Science, ETH Zurich, Switzerland
Herrera F, Lozano M, Verdegay J (1998) Tackling realcoded genetic algorithms: operators and tools for behavioral analysis. Artif Intell Rev 12(4):265–319
Holm S (1979) A simple sequentially rejective multiple test procedure. Scand J Stat 6:65–70
Hooker JN (1995) Testing heuristics: We have it all wrong. J Heuristics 1(1):33–42. doi:10.1007/ BF02430364
Igel C, Toussaint M (2003) On classes of functions for which no free lunch results hold. Inform Process Lett 86(6):317–321
Igel C, Toussaint M (2004) A no-free-lunch theorem for non-uniform distributions of target functions. J Math Modell Algorithms 3(4):313–322
Iman R, Davenport J (1980) Approximations of the critical region of the Friedman statistic. Commun Stat 9:571–595
Jiang P, Chen Y (2010) Free lunches on the discrete Lipschitz class. Theor Comput Sci 412(17):1614–1628. doi:10.1016/j.tcs.2010.12.028
Jünger M, Liebling T, Naddef D, Nemhauser G, Pulleyblank W et al (eds) (2009) 50 Years of integer programming 1958–2008: from the early years to the state-of-the-art. Springer, Berlin
Karp R (1972) Reducibility among combinatorial problems. In: Miller R, Thatcher J (eds) Complexity of computer computations. Plenum Press, New York, pp 85–103
Kauffman S (1989) Adaptation on rugged fitness landscapes. Lect Sci Complex 1:527–618
Kirkpatrick S, Gelatt Jr C, Vecchi M (1983) Optimization by simulated annealing. Science 220(4598):671–680
Koehler GJ (2007) Conditions that obviate the no-free- lunch theorems for optimization. INFORMS J Comput 19(2):273–279. doi:10.1287/ijoc.1060.0194
Laplace PS (1814) Essai philosophique sur les probabilités. Technical report, Paris, Courcier
Lozano M, García-Martínez C (2010) Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: overview and progress report. Comput Operat Res 37:481–497
Lozano M, Herrera F, Molina D (eds) (2011) Scalability of evolutionary algorithms and other metaheuristics for large scale continuous optimization problems, vol 15. Soft Computing
Marshall JAR, Hinton TG (2010) Beyond no free lunch: realistic algorithms for arbitrary problem classes. In: IEEE Congr Evol Comput 1:18–23
Pelikan M, Goldberg D, Cantú-Paz E (2000) Linkage problem, distribution estimation, and bayesian networks. Evol Comput 8(3):311–340
Rodriguez FJ, García-Martínez C, Lozano M (2012) Hybrid metaheuristics based on evolutionary algorithms and simulated annealing: taxonomy, comparison, and synergy test. IEEE Trans Evol Comput. doi:10.1109/TEVC.2012.2182773
Schaffer J, Eshelman L (1991) On crossover as an evolutionary viable strategy. In: Belew R, Booker L (eds) Proceedings of the international conference on genetic algorithms. Morgan Kaufmann, San Mateo, pp 61–68
Schumacher C, Vose MD, Whitley LD (2001) The No Free Lunch and Problem Description Length. In: Proc. of the Genetic and Evolutionary Computation Conference, pp 565–570
Service TC (2010) A no free lunch theorem for multiobjective optimization. Inform Process Lett 110(21):917–923. doi:10.1016/j.ipl.2010.07.026
Smith K, Hoos H, Stützle T (2003) Iterated robust tabu search for MAX-SAT. In: Carbonell J, Siekmann J (eds) Proceedings of the Canadian society for computational studies of intelligence Conference, vol LNCS 2671. Springer, Berlin, pp 129–144
Talbi E (2002) A taxonomy of hybrid metaheuristics. J Heuristics 8(5):541–564
Thierens D (2002) Adaptive mutation rate control schemes in genetic algorithms. In: Proceedings of the congress on evolutionary computation, pp 980–985

Thierens D (2004) Population-based iterated local search: restricting neighborhood search by crossover. In: Proceedings of the genetic and evolutionary computation conference, vol LNCS 3103. Springer, Berlin, pp 234–245

Watson R, Pollack J (1999) Hierarchically consistent test problems for genetic algorithms. Proc Congr Evol Comput 2:1406–1413

Whitley D, Rowe J (2008) Focused no free lunch theorems. In: Proceedings of the genetic and evolutionary computation conference. ACM Press, New York, pp 811–818. doi:10.1145/1389095.1389254

Whitley D, Watson JP (2005) Complexity theory and the no free lunch theorem. Search Methodologies, Springer, Berlin, pp 317–339. doi:10.1007/0-387-28356-0_11

Whitley D, Rana S, Dzubera J, Mathias E (1996) Evaluating evolutionary algorithms. Artif Intell 85:245–276

Wolpert D, Macready W (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82

Zar J (1999) Biostatistical analysis. Prentice Hall, Upper Saddle River