Evolutionary Computation: Challenges and duties

Carlos Cotta[1] and Pablo Moscato[2]

[1]Dept. Lenguajes y Ciencias de la Computación, Universidad de Málaga
ETSI Informática (3.2.49), Campus de Teatinos, 29071-Málaga, Spain
ccottap@lcc.uma.es

[2]School of Electrical Engineering and Computer Science,
University of Newcastle,
Callaghan, NSW, 2308 Australia
moscato@densis.fee.unicamp.br

14th January 2003

Abstract

Evolutionary Computation (EC) is now a few decades old. The impressive development of the field since its initial conception has made it one of the most vigorous research areas, specifically from an applied viewpoint. This should not hide the existence of some major gaps in our understanding on these techniques. In this essay we propose a number of challenging tasks that –according to our opinion– should be attacked in order to fill some of these gaps. They mainly refer to the theoretical basis of the paradigm; we believe that an effective cross-fertilization among different areas of Theoretical Computer Science and Artificial Intelligence (such as Parameterized Complexity and Modal Logic) is mandatory for developing a new corpus of knowledge about EC.

1 Introduction

On July 2, 2002, we made several queries using one of the most popular databases for retrieval of research publications, the *Web of Science*[1]. Undoubtedly, one of the best jewels of the crown of Theoretical Computer Science is the theory of NP-completeness [18], so we thought that it would be relevant to identify how many papers have been published and catalogued in this database that include either the terms *'NP-hard'* or *'NP-complete'* in either the title or the abstract or even as a keyword. It has been reported elsewhere that "thousands" of problems have been already catalogued as NP-hard, so we thought that this search would at least help to indirectly quantify the presence of NP-completeness theory in the scientific and technological literature. Surprisingly for us, only 4,111 documents contained at least one of these terms. We expected this number to be larger provided the significance and widespread usefulness of this classification.

This result is curious, we hoped to get a larger figure, since, for comparison purposes, the same database retrieved 1,361 documents containing either *'salesman problem'* or *'salesperson problem'*, in general referring to a particular problem member of the NP Optimization class (the traveling salesman problem, Min TSP) .

[1]http://www.isinet.com

Regarding the current use of "single-agent" metaheuristic optimization methods, two of them take the lead with *"Simulated Annealing"* [24] (4,676) and *"Tabu Search"* [20] (856). These metaheuristics have been introduced at least one decade after the theory of NP-completeness and they are widely used in practice. Noting that any metaheuristic method turns into a heuristic when applied to a particular problem, it is also relevant to query for *"heuristic"* or *"heuristics"* giving an impressive number of 15,933 documents in the database, most of them on algorithmic approaches to solve a problem modeled in formal mathematical terms.

There are many possible interpretations for the results of these database queries, and each of these interpretations is the amalgamation of a number of factors and conjectures. The reader may agree with us in that the great success of metaheuristics in solving in practice many hard optimization problems is certainly one of the circumstances to take into account. In our opinion, a subtle shift in research focus is also a major factor in this result. More precisely, it may be that the relative weight of *applied* research (recall that most of the works dealing with metaheuristics are of applied nature) has increased with respect to *fundamental* research. The wide availability of computing resources is crucial in this sense: testing and comparing different approaches for solving a problem can be much more amenable than complex mathematical analysis. This philosophy could be summarized in *"try to get probably good solutions to your problem, for provably good solutions are overwhelmingly hard to obtain"*. For most problem domains, we should take extreme care in order to define what can be a challenging instance, since it may be extremely easy to find optimal solutions [25], biasing the chosen scenario favoring exact methods (see the discussion in [3]).

While the lack of a proper mathematical analysis is not something to be inherently criticized from a scientific point of view, it is true that the lack of solid theoretical basis for most metaheuristics will be a remora, jeopardizing their successful utilization in the 21^{st} Century. Quoting Lewis and Papadimitriou [27]:

> *"Explaining and predicting the impressive empirical success of some of these algorithms is one of the most challenging frontiers of the theory of computation today."*

Indeed, developing formal theories for grasping the optimization dynamics of these algorithms, and to devise appropriate metaheuristics for solving specific problems appear as the major challenges researchers have to face. This is specifically true in the field of evolutionary algorithms [2] (EAs), one of the metaheuristics families with stronger impetus, yet whose foundation-knowledge corpus remains very incomplete.

In this essay, we will try to identify some of the principal challenges whose solution we believe may constitute important milestones for EA development. Each of these challenges will be described in a different section. It must be noted that their numbering is not intended to represent any relevance order. On the contrary, the reader is invited to rank them according to his/her particular vision of the field.

2 Challenge #1: Hard problems for the paradigm – Epistasis and Parameterized Complexity

It is absolutely necessary to identify and understand the relative "hardness" of finding appropriate algorithms for specific problems, in particular with respect to the computational complexity classes to which the problems belong. We believe that it would be a better attitude, particularly toward building a bridge with Theory of Computation, to try firstly to identify hard problems for EAs in relationship with known computational classes. At present, the approach of creating "toy problems" that are "hard" for the paradigm, while partly useful for identifying some particular issues that need consideration, does not lead to an articulated, systematic approach to

understand for which problems, or problem instances, the EA approach is competitive or even superior, to exact approaches or other metaheuristics. We may ask:

> *Is there any way to find efficient algorithms based on evolutionary search principles which always give good approximations to optimization problems to which it is hard to find the optimal solution ?*

From some perspective, the answer of this question is most probably "No", since under the commonly believed assumption that $P \neq NP$, we know that there are some problems that can not be approximated with efficient algorithms at all. On the other hand, this question can be relativized by answering: *"It depends on the problem"* since we know that for some problems that are equally hard to be solved to optimality, some can be very well approximated with efficient algorithms. This leaves some room for the possibility that some problems can be efficiently approximated using algorithms based on evolutionary techniques while others do not.

To study the central question presented above, we identify three complementary research directions:

- Identify NP-Optimization problems for which the evolutionary search paradigm has proved not to be competitive against the best heuristic or approximation algorithm known for those problems.
- Identify which optimization problems can be approached using an evolutionary search paradigm and identify the reasons.
- For the problems of the two groups mentioned above, it will be important to find links with the theory of computational complexity and the complexity classes (regarding approximability and, in particular, parameterized complexity [16]) that those problems belong to.

Ideally, the outcome of the above research will also provide interesting clues in terms of relating computational complexity classes with the typical measures of "EA-hardness" such as epistasis [11, 17]. This phenomenon –the non-additive fitness dependence among several *genes*– has a direct influence in the difficulty an EA faces for solving a certain problem (defined as the combination of a particular representation and fitness function). It is customary to quantify epistasis by means of a integer fixed parameter, say $p > 0$. The existence of such a parameter in the context of the discussion about complexity classes mentioned above immediately suggest a possible connection with the paradigm of *parameterized complexity* (PC) (for an interesting introduction to the general topic see [14]). This paradigm extends the classical paradigm by analyzing the complexity of problems with respect to a certain parameter (or set of parameters). Recall that classical classifications such as the conspicuous $P - NP$ dichotomy are based in a worst-case scenario. For instance, the paradigmatic SAT problem is known to be easily solvable in general (for a particular type or randomly-generated instances) except for instances located at the phase transition between satisfiability and non-satisfiability [19]. The existence of structural parameters upon which to base the complexity analysis can be very useful to isolate such scenarios.

The PC paradigm establishes a hierarchy of parameterized complexity classes $FPT \subseteq W[1] \subseteq W[2] \subseteq \cdots \subseteq W[t] \subseteq W[SAT] \subseteq W[P]$ that allows discriminating problems of different complexity according to the chosen parameter. For example, problems in the FPT (*fixed-parameter tractable*) class have algorithms whose worst-case complexity is $O(f(k)n^c)$, where k is the parameter, $f(k)$ and arbitrary function of k only, and c is a constant. In contrast, the complexity of solving problems in $W[1]$ is $O(f(k)n^{g(k)})$, substantially harder in general.

A prototypical example of an NP-complete problem whose parameterized version is fixed-parameter tractable is Vertex Cover. This problem can be defined as follows:

k-Vertex Cover

- **Instance:** An undirected graph $G(V, E)$, with $|V| = n$, an integer $k > 0$.
- **Question:** Does there exist a set $V' \subseteq V$ of k vertices, such that for every $(u, w) \in E$, it holds that $u \in V'$ or $w \in V'$?

If the size of the set V' is taken as a parameter, this problem can be shown to be in FPT [13], existing algorithms for solving it in $O(1.278^k + n)$, i.e., linear in n for fixed k, and polynomial in n for $k \in O(\log n)$. This surprising result can be achieved by combining the results of [6] and the speed-up method of [31]. Notice that while Vertex Cover would be dismissed as "probably intractable" according to its NP-hardness, it turns out to be perfectly solvable for a wide range of values for its structural parameter.

A lesson can be extracted, since we may apply these algorithms for recombination operators. They appear to be greatly advantageous when the population has begun to converge to similar individuals. We will return to this issue in the next challenge.

As mentioned above, the W-hierarchy allows encapsulating problems of increasing difficulty. The membership of a certain problem to a precise PC class is established by means of Boolean circuits. These are traditional networks of logical gates that take a potential solution as an input, and output a Boolean value indicating whether that is a solution for the problem considered. The structure of the circuit obviously depends on the problem, and its complexity determines the precise PC class to which membership is established. More precisely, an important parameter is the *weft* of the circuit. This is the maximum number of logical gates whose fan-in is unrestricted (depends on the problem data) in an input-output path. The higher the weft for a constant depth of the circuit, the higher the class the circuit belongs to.

At this point, the resemblance between the weft of a Boolean circuit and the structural interdependence of genes in epistatic representations suggests that some deep connection may exist between PC and the yet informal notion of "EA-hardness". Disentangling this connection (if it effectively exists) constitutes a very attractive challenge both for Computer Science theoreticians and EA researchers.

3 Challenge #2: Systematic design of provably good recombination operators

Recombination is undoubtedly a major component of evolutionary algorithms, at least in the case of genetic algorithms (GAs). While its intuitive rôle has been always clear (to combine the information present in a set of solutions to create new solutions), the guidelines for designing effective recombination operators have experienced a remarkable evolution. It is increasingly more accepted that instead of directly manipulating the syntactic units used to encode solutions, the operator must extract *"relevant"* information from these solutions and recombine it (with independence of whether solutions are encoded on the basis of these particular information pieces or not). Unfortunately, the concept of "relevance" is hardly defined in formal terms.

For instance, solutions of the NP optimization problem Min Traveling Salesman (Min TSP) can be encoded as permutations or even as binary strings. However, operators working directly on these encodings such as *cycle crossover* [32] or *uniform crossover* [35] will in general provide worse performance than operators extracting the relevant information. For the symmetric instances of the Min TSP it has been shown that the preservation of some features, in particular *common edges* is indeed a good strategy [28], and this lead to the proposal of several *"edge recombination"* methods [30].

We will refer to these relevant "pieces of information" as *features*. We note, however, that in most of the cases where the problem to be solved is intractable, these

features generally correspond to *predicates computable in polynomial-time*. Back to the symmetric instance of the MIN TSP example, when a recombination operator requires to *"find all* common *edges of the* $k_{par} \geq 2$ *parents"* this can be understood as checking $k_{par} \times n(n-1)/2$ predicates (where n denotes the number of cities). Each one of them corresponds to one edge between two cities, and we return the edges for which the associate predicate returns a 'Yes' for all parents.

After having identified the relevant features (let us suppose we managed to find all features of a set of parent solutions in polynomial-time), the next and obviously important step is deciding how we can use this information. While *blind* recombination operators that randomly shuffle the set of features were more typical in the past, the addition of problem-domain knowledge to guide the process is becoming increasingly popular. The terms *hybrid* GAs and *memetic algorithms* (MAs) [29] [30] have been coined to denote these methods that use *smarter* reproductive operators and periods of single-agent optimization.

There exist a plethora of mechanisms to create these *smart* recombination operators, e.g., [9, 33], but, up to the best of our knowledge, no complexity results for some of the decision problems involved have been reported. For instance, suppose we have a number of $m \geq 3$ tours from a relatively large population of size $Pop \gg m$. Let us also suppose that $m-1$ of them have lengths values which are below the current population's average length value, but one has a value well above average. To strengthen the argument we can even suppose that it is actually the longest tour in the entire population. While the preservation of edges/features present in all m parents can still make some sense, we notice that the preservation of edges/features present in the best $m-1$ parents *and not* present in the worst tour, seems also a valuable heuristic to create new solutions. Analogously, the *avoidance* of a feature present in the worst tour *and not* present in the other $m-1$ tours is certainly another appealing heuristic.

It is clear that, while there might be other heuristics of interest for special cases, the previous example clearly depicts the existence of a more general problem: *given a set of parents, find the optimal subset of features to avoid and to preserve*. This problem already appears when we have parents that can be categorized in two different classes. A natural measure of optimality is the cardinality of the set, since we expect that m is already a small number in comparison with the size of the instance, then we only expect to make a valid inference if the number of chosen features is also small.

We think that the EC community may critique itself in having not yet defined a systematic effort to understand how to extract useful features from populations of solutions. Although some *ad-hoc* approaches work for particular problems, most recombination approaches are naive attempts to solve a more fundamental issue, that of extracting particular characteristics/features that the optimal solutions might have and, possibly more important, which features *might not* be present in them.

Related with this latter point, it must be noted that we still lack a formal framework for recombination, similar for instance to that we have for Local Search [23, 38]. In this sense, an interesting new direction for theoretical research arose after the introduction of two computational complexity classes, the PMA class (for *Polynomial Merger Algorithms problems*) and its unconstrained analogue, the uPMA class. We will define the classes PMA and uPMA by referring to three analogous algorithms to the ones that define the class of *Polynomial Local Search* problems (PLS). These definitions (specially for PMA) are particularly dependent on an algorithm called *k-merger*, that will help to formalize the notion of *recombination of a set of k given solutions*, as generally used by most MAs (as well as other population approaches). The input of a k-merger algorithm is a set S_{par} of $k \geq 2$ *feasible* solutions. They can be informally called "parent" solutions and, if successful, the k-merger delivers as output *at least one* feasible solution (with some constraints). For the uPMA class the construction of the new solution is less restricted than for PMA. In general, recombination processes can be very complex with many side restrictions involving the

detection, the preservation or avoidance, and the feasible combination of *features* already present in the parent solutions.

Definition (uPMA). Let x be an instance of an optimization problem P. With $\mathcal{M}_P(S_{par}, x)$ we denote the set of all possible outputs (i.e., feasible solutions) that the *k-merger* algorithm can give if it receives as input the pair (S_{par}, x) for problem P.

A recombination problem $P/\mathcal{M}$ belongs to uPMA if there exist three polynomial-time algorithms *p-starter*, *p′-evaluator*, and *k-merger* (where p, p' and k are integer numbers such that $p' \geq p \geq k \geq 2$) that satisfy the following properties:

- Given an input x (formally a string $\in \{0,1\}^*$), the *p-starter* determines whether x is an instance of problem P and in this case produces a set of p different feasible solutions $\{y_1, y_2, \ldots, y_p\}$.
- Given an instance x of P and an input (formally a string $\in \{0,1\}^*$), the *p′-evaluator* determines whether this input represents a set of feasible solutions, i.e. $\{y_1, y_2, \ldots, y_{p'}\}$ and in that case it computes the value of the objective function associated to each one of them, i.e. $m_P(y_j, x), \forall j = 1, \ldots, p'$.
- Given an instance x of P and a set of k feasible solutions S_{par}, the *k-merger* determines whether the set S_{par} is a *k-merger optimum*, and, if it is not, it outputs at least one feasible solution $y' \in \mathcal{M}_P(S_{par}, x)$ with strictly better value of m_P (i.e. $m_P(y', x) < max\ \{m_P(y_1, x), m_P(y_2, x), \ldots, m_P(y_k, x)\ \}$ for a minimization problem, and $m_P(y', x) > min\ \{m_P(y_1, x), m_P(y_2, x), \ldots, m_P(y_k, x)\ \}$ for a maximization problem).

Analogously, the PMA class is more restricted since it embodies a particular type of recombination. For uPMA the type of recombination is implicit in the way the group neighborhood $\mathcal{M}$ is defined. However, the definition for PMA is still general enough to encompasses most of the recombination procedures used in practical population-based approaches.

Definition (PMA). A recombination problem $P/\mathcal{M}$ belongs to PMA if there exist three polynomial-time algorithms *p-starter*, *p′-evaluator*, and *k-merger* (where p, p' and k are integer numbers such that $p' \geq p \geq k \geq 2$), such that the *p-starter* and *p′-evaluator* satisfy the same properties required by the uPMA class but the *k-merger* is constrained to be of a particular type, i.e.:

- Given an instance x of P and a set of k feasible solutions S_{par}, the *k-merger* determines whether the set S_{par} is a *k-merger optimum*, and, if it is not, it does the following:
  - For each $y \in S_{par}$, it solves n_1 polynomial-time decision problems $\{\Pi_1(y), \Pi_2(y), \ldots, \Pi_{n_1}(y)\}$. Let D be a matrix of $k \times n_1$ Boolean coefficients formed by the output of all these decision problems, i.e. $D_{i,j} = \Pi_j(y_i)$.
  - It creates a set of n_2 *constraints* C, such that C can be partitioned in two subsets, i.e. $C = C_{in} \cup C_{out}$. Each constraint $c \in C$ is represented by a predicate π_c such that its associated decision problem $\Pi_c(y)$ can be solved in polynomial-time for every feasible solution y. Any predicate π_c is a polynomial-time computable function that has as input the Boolean matrix D and the instance x. It is required that *at least one* predicate π_c^* to be a non-constant function of *at least two different elements of* S_{par}.
  - It outputs at least one *offspring*, i.e. another feasible solution $y' \in \mathcal{M}_P(S_{par}, x)$ with strictly better value of m_P, (i.e. $m_P(y', x) < max\ \{m_P(y_1, x), m_P(y_2, x), \ldots, m_P(y_k, x)\ \}$ for a minimization problem, and $m_P(y', x) > min\ \{m_P(y_1, x),$

$m_P(y_2, x)$, ..., $m_P(y_k, x)$ } for a maximization problem) subject to

$$\max_{y'}[(\sum_{(c \in C_{in}) \wedge \Pi_c(y')} w_c) - (\sum_{(c \in C_{out}) \wedge \Pi_c(y')} w_c)] \quad (1)$$

where w_c is an integer weight associated to constraint c.

Conducting research to identify problems, and their associated recombination procedures, such that membership, in either PMA or uPMA, can be proved is a definitely important task. It is also hoped that after some initial attempts on challenging problems completeness and reductions for the classes can be properly defined.

We should also note that the definition are such that they would naturally give several new interesting parameterized complexity problems. So, while proving NP-hardness is a good start, we hope that the research focus should be directed towards proving many problems to be fixed-parameter tractable. In essence, that would lead toward developing "optimal" recombination operators, that while exponential on the parameters, can be polynomial on the instance size.

4 Challenge #3: Using Modal Logic and Logic Programming methods to guide the search

Looking ahead one of the possible directions that EC can take, after checking the current trends, it is then reasonable to affirm that increasingly more complex schemes evolving solutions, agents, as well as representations, will soon be implemented. The way they would handle information (actually it is a "distributed" information for it is carried by a population of solutions, which can be transmitted, recombined, and analyzed) have some points in common with *Blackboard Systems* [15]. This has been recognized in the past yet it is conspicuously hardly being mentioned in the current metaheuristics literature.

We are proposing to call these new methods as *Belief Search* and to show they can work in an EC setting, we will resort to two illustrative examples. We will assume that the formula $\mathbf{B}^i_a\phi$ has the following meaning *"agent i believes with strength (at least) a that ϕ is true"*, such that the strength values a are restricted to be rational numbers in [0,1]. Let us also suppose we accept as an axiom that from ψ being true we can deduce $\mathbf{B}^i_1\psi$ for all i. Now let us suppose that our agents are trying to solve a MIN TSP and that the particular instance being considered is Euclidean and two-dimensional. Let ϕ_k represent the proposition *"edge e_k is present in the optimum tour"* and let $\chi_{k,l}$ be true if edges e_k and e_l cross each other, and false otherwise. It can be proved (a "folk theorem") that for such particular type of TSP instances (a form of problem-domain, or better, instance-domain knowledge) *"if edges e_k and e_l cross each other, then e_k and e_l can not both be present in the optimal tour"*. Then we can assume that this is known by all agents, and by the previous axiom we can deduce that agent 2 now believes $\mathbf{B}^2_1(\chi_{k,l} \rightarrow \neg(\phi_k \wedge \phi_l))$. Now let us suppose that agent 1 believes, with strength 0.4, that *"either edge e_k or e_l, but not both, is present in the optimal tour"*. We will represent this as $\mathbf{B}^1_{0.4}\phi_{k,l}$. We will not enter into the discussion of how that agent reached that belief and we take it as a fact. Now let us suppose that another agent believes, at a level 0.7 that $\chi_{k,l} \rightarrow \phi_{k,l}$, then we write $\mathbf{B}^3_{0.7}(\chi_{k,l} \rightarrow \phi_{k,l})$. This is curious, since this kind of assumption confuses our common sense. In general we do not see any relationship between the fact that two edges cross and that we can deduce that as a consequence one of them should be present in the optimum tour. We can take this as a fact, as if a "co-evolving" algorithm, is generating these predicates to guide the search. However, note that agent 3 believes in this relationship (at a 0.7 level) for *a particular* pair of edges e_k and e_l. Now, what can we say about the *distributed belief* of this group of three agents ? How can we recombine this information ?

At this point we need to introduce a logic to recombine belief information. Discussions on which particular type of logic to guide heuristic search process is a much more elegant and useful method than keeping on discussing values of parameters based on trial-and-error experimental tests. It may also lead to improved convergence in *Estimation-of-Distributions* (EDA) metaheuristics [26].

According to one possible selection for such a logic, just picked to exemplify the discussion, we can use $\mathbf{PL}_n^{\otimes}$, a multi-agent epistemic logic recently introduced by Boldrin and Saffiotti, the opinions shared by a set of n different agents can be *recombined* in a distributed belief. Using $\mathbf{PL}_n^{\otimes}$ we can deduce $\mathbf{D}_{0.82}\phi_{k,l}$. The distributed belief about proposition $\phi_{k,l}$ is then stronger than any individual belief about it, and is even stronger than what you would get if any agent would believe the three facts. We offer now two examples on its application.

4.1 Example 1

In $\mathbf{PL}_n^{\otimes}$ we have the following axioms and inference rules, where ϕ and ψ range over formulas of $\mathcal{L}$; a, b and c over rational numbers in [0,1]; and $i = 1, \ldots, n$. The five axioms are:

(A0)	Axiom 0	All propositional tautologies	
(A1)	Axiom 1	$\mathbf{B}_0^i \perp$	
(A2)	Axiom 2	$\mathbf{B}_a^i(\phi \to \psi) \to (\mathbf{B}_b^i\phi \to \mathbf{B}_c^i\psi)$	$c \leq \min\{a, b\}$
(A3)	Axiom 3	$\mathbf{D}_a(\phi \to \psi) \to (\mathbf{D}_b\phi \to \mathbf{D}_c\psi)$	$c \leq \min\{a, b\}$
(A4)	Axiom 4	$(\wedge_{i=1}^n \mathbf{B}_{a_i}^i\phi) \to \mathbf{D}_c\phi$	$c = \oplus_{i=1}^n a_i$

The three inference rules are :

(MP)	Modus Ponens	from ϕ and $\phi \to \psi$ deduce ψ
(NEC)	Necesitation	from ϕ deduce $\mathbf{B}_1^i\phi$
(US)	Uniform substitutions	

A formula ϕ is said to be a theorem of $\mathbf{PL}_n^{\otimes}$ written $\vdash \phi$, if ϕ is obtained from **A0-A4** by a finite number of applications of **MP**, **NEC** and uniform substitutions. Then, if $\Gamma \subseteq \mathcal{L}$ we will write $\Gamma \vdash \phi$ to mean $\vdash (\wedge_{\phi \in \Gamma}) \to \phi$.

Proposition: Given $\Gamma = \{(a), (b), (c)\}$ (see below) then $\Gamma \vdash (\mathbf{D}_{0.82}\phi_{k,l} \wedge \mathbf{D}_{0.82}(\chi_{k,l} \to \phi_{k,l}))$.

(a)$\mathbf{B}_{0.4}^1\phi_{k,l}$,
(b)$\chi_{k,l}$,
(c)$\mathbf{B}_{0.7}^3(\chi_{k,l} \to \phi_{k,l})$.

Proof:

(1)$\phi_{k,l} \rightarrow (\chi_{k,l} \rightarrow \phi_{k,l})$	**A0**
(2)$\mathbf{B}^1_1(\phi_{k,l} \rightarrow (\chi_{k,l} \rightarrow \phi_{k,l}))$	**NEC**, 1
(3)$\mathbf{B}^1_1(\phi_{k,l} \rightarrow (\chi_{k,l} \rightarrow \phi_{k,l}))$	
$\rightarrow (\mathbf{B}^1_{0.4}\phi_{k,l} \rightarrow \mathbf{B}^1_{0.4}(\chi_{k,l} \rightarrow \phi_{k,l}))$	**A2**, **US**
(4)$\mathbf{B}^1_{0.4}\phi_{k,l} \rightarrow \mathbf{B}^1_{0.4}(\chi_{k,l} \rightarrow \phi_{k,l})$	**MP**, 2, 3
(5)$\mathbf{B}^1_{0.4}(\chi_{k,l} \rightarrow \phi_{k,l})$	**MP**, (a), 4
(6)$\bot\rightarrow (\chi_{k,l} \rightarrow \phi_{k,l})$	**A0**
(7)$\mathbf{B}^2_1(\bot\rightarrow (\chi_{k,l} \rightarrow \phi_{k,l}))$	**NEC**, 6
(8)$\mathbf{B}^2_0 \bot$	**A1**
(9)$\mathbf{B}^2_1(\bot\rightarrow (\chi_{k,l} \rightarrow \phi_{k,l}))$	
$\rightarrow (\mathbf{B}^2_0 \bot\rightarrow \mathbf{B}^2_0(\chi_{k,l} \rightarrow \phi_{k,l}))$	**A2**, **US**
(10)$\mathbf{B}^2_0 \bot\rightarrow \mathbf{B}^2_0(\chi_{k,l} \rightarrow \phi_{k,l})$	**MP**, 7, 9
(11)$\mathbf{B}^2_0(\chi_{k,l} \rightarrow \phi_{k,l})$	**MP**, 8, 10
(12)$\mathbf{B}^2_0(\chi_{k,l} \rightarrow \phi_{k,l}) \rightarrow (\mathbf{B}^2_1\chi_{k,l} \rightarrow \mathbf{B}^2_0\phi_{k,l})$	**A2**, **US**
(13)$\mathbf{B}^2_1\chi_{k,l} \rightarrow \mathbf{B}^2_0\phi_{k,l}$,	**MP**, 11, 12
(14)$\mathbf{B}^2_1\chi_{k,l}$	**NEC**, (b)
(15)$\mathbf{B}^2_0\phi_{k,l}$	**MP**, 14, 13
(16)$\mathbf{B}^3_1\chi_{k,l}$	**NEC**, (b)
(17)$\mathbf{B}^3_{0.7}(\chi_{k,l} \rightarrow \phi_{k,l}) \rightarrow (\mathbf{B}^3_1\chi_{k,l} \rightarrow \mathbf{B}^3_{0.7}\phi_{k,l})$	**A2**, **US**
(18)$\mathbf{B}^3_1\chi_{k,l} \rightarrow \mathbf{B}^3_{0.7}\phi_{k,l}$	**MP**, (c), 17
(19)$\mathbf{B}^3_{0.7}\phi_{k,l}$	**MP**, (c), 18
(20)$\mathbf{D}_{0.82}(\chi_{k,l} \rightarrow \phi_{k,l})$	**A4**, (5), (11), (c)
(21)$\mathbf{D}_{0.82}\phi_{k,l}$	**A4**, (a), 15, 19

We can leave to the reader the task of checking this example following section 3.3. of [4].

Another interesting exercise is the following: let ϕ_k (ϕ_l) be the predicate *"edge e_k (respectively, e_l) is present in the optimal tour"*. The task is then to deduce according to $\mathbf{PL}^{\otimes}_n$ what can be distributedly believed about individual edges e_k and e_l if in addition to the three previous agents there are also two other agents, such that $\mathbf{B}^4_{0.54}\phi_k$, $\mathbf{B}^4_{0.27}\phi_l$, $\mathbf{B}^5_{0.63}\phi_k$, $\mathbf{B}^5_{0.15}\phi_l$.

4.2 Example 2

The following are theorems of $\mathbf{PL}^{\otimes}_n$:

(TH1) $(\mathbf{B}^i_a\phi \wedge \mathbf{B}^i_b\psi) \rightarrow \mathbf{B}^i_c(\phi \vee \psi) \quad c = \max\{a, b\}$

(TH2) $(\mathbf{B}^i_a\phi \wedge \mathbf{B}^i_b\psi) \rightarrow \mathbf{B}^i_c(\phi \wedge \psi) \quad c = \min\{a, b\}$

If the agents are trying to solve an Euclidean, 2-dimensional instance of the TSP, then we also have the *instance-dependent axioms* or **IDA**s.

IDA1 $\chi_{k,l} \rightarrow \neg(\phi_k \wedge \phi_l)$

In addition, we also know that:

IDA2 $\phi_{k,l} \equiv ((\phi_k \rightarrow \neg\phi_l) \wedge (\neg\phi_k \rightarrow \phi_l))$

Proposition: *Given* $\Gamma' = \{(a), \ldots, (g)\}$ *(see below) then* $\Gamma' \vdash \boldsymbol{D}_{0.96934}\phi_{k,l}$.
(a)$\mathbf{B}^1_{0.4}\phi_{k,l}$
(b)$\chi_{k,l}$
(c)$\mathbf{B}^3_{0.7}(\chi_{k,l} \rightarrow \phi_{k,l})$
(d)$\mathbf{B}^4_{0.54}\phi_k$
(e)$\mathbf{B}^4_{0.27}\phi_l$
(f)$\mathbf{B}^5_{0.63}\phi_k$
(g)$\mathbf{B}^5_{0.15}\phi_l$

Proof:

(1)$\mathbf{B}^4_1(\chi_{k,l} \rightarrow \neg(\phi_k \wedge \phi_l))$	**IDA1**, **NEC**
(2)$\mathbf{B}^4_1(\chi_{k,l} \rightarrow (\phi_k \rightarrow \neg\phi_l))$	**A0**
(3)$\mathbf{B}^4_1\chi_{k,l}$	(b), **NEC**
(4)$\mathbf{B}^4_1(\chi_{k,l} \rightarrow (\phi_k \rightarrow \neg\phi_l)) \rightarrow (\mathbf{B}^4_1\chi_{k,l} \rightarrow \mathbf{B}^4_1(\phi_k \rightarrow \neg\phi_l))$	**A2**, **US**
(5)$\mathbf{B}^4_1\chi_{k,l} \rightarrow \mathbf{B}^4_1(\phi_k \rightarrow \neg\phi_l)$	**MP**, 2, 4
(6)$\mathbf{B}^4_1(\phi_k \rightarrow \neg\phi_l)$	**MP**, 3, 5
(7)$\mathbf{B}^4_1(\phi_k \rightarrow \neg\phi_l) \rightarrow (\mathbf{B}^4_{0.54}\phi_k \rightarrow \mathbf{B}^4_{0.54}(\neg\phi_k))$	**A2**
(8)$\mathbf{B}^4_{0.54}\phi_k \rightarrow \mathbf{B}^4_{0.54}(\neg\phi_l)$	**MP**, 6, 7
(9)$\mathbf{B}^4_{0.54}(\neg\phi_l)$	**MP**, (d), 8

analogously we can deduce:

(18)$\mathbf{B}^4_{0.27}(\neg\phi_k)$	**MP**, (e), 17
(27)$\mathbf{B}^5_{0.63}(\neg\phi_l)$	**MP**, (f), 26
(36)$\mathbf{B}^5_{0.15}(\neg\phi_k)$	**MP**, (g), 35

and now we will use one of the theorems:

(37)$(\mathbf{B}^4_{0.54}\phi_k \wedge \mathbf{B}^4_{0.27}\phi_l) \rightarrow \mathbf{B}^4_{0.54}(\phi_k \vee \phi_l)$	**TH1**, **US**
(38)$\mathbf{B}^4_{0.54}(\phi_k \vee \phi_l)$	**MP**, (d), (e), 37
(39)$(\mathbf{B}^5_{0.63}\phi_k \wedge \mathbf{B}^5_{0.15}\phi_l) \rightarrow \mathbf{B}^5_{0.63}(\phi_k \vee \phi_l)$	**TH1**, **US**
(40)$\mathbf{B}^5_{0.63}(\phi_k \vee \phi_l)$	**MP**, (f), (g), (39)
(41)$(\mathbf{B}^4_{0.54}(\neg\phi_l) \wedge \mathbf{B}^4_{0.27}(\neg\phi_k)) \rightarrow \mathbf{B}^4_{0.54}(\neg\phi_k \vee \neg\phi_l)$	**TH1**, **US**
(42)$\mathbf{B}^4_{0.54}(\neg\phi_k \vee \neg\phi_l)$	**MP**, 9, 18, 41
(43)$(\mathbf{B}^5_{0.63}(\neg\phi_l) \wedge \mathbf{B}^5_{0.15}(\neg\phi_k)) \rightarrow \mathbf{B}^5_{0.63}(\neg\phi_k \vee \neg\phi_l)$	**TH1**, **US**
(44)$\mathbf{B}^5_{0.63}(\neg\phi_k \vee \neg\phi_l)$	**MP**, 27, 36, 43

and now we combine the information using the other theorem:

(45)$(\mathbf{B}^4_{0.54}(\phi_k \vee \phi_l)) \wedge (\mathbf{B}^4_{0.54}(\neg\phi_k \vee \neg\phi_l))$ $\rightarrow \mathbf{B}^4_{0.54}((\phi_k \vee \phi_l) \wedge (\neg\phi_k \vee \neg\phi_l))$	**TH2**, **US**
(46)$\mathbf{B}^4_{0.54}((\phi_k \vee \phi_l) \wedge (\neg\phi_k \vee \neg\phi_l))$	**MP**, 38, 42, 45
(47)$\mathbf{B}^4_{0.54}\phi_{k,l}$	**A0**
(48)$(\mathbf{B}^5_{0.63}(\phi_k \vee \phi_l)) \wedge \mathbf{B}^5_{0.63}(\neg\phi_k \vee \neg\phi_l))$ $\rightarrow \mathbf{B}^5_{0.63}((\phi_k \vee \phi_l) \wedge (\neg\phi_k \vee \neg\phi_l))$	**TH2**, **US**
(49)$\mathbf{B}^5_{0.63}((\phi_k \vee \phi_l) \wedge (\neg\phi_k \vee \neg\phi_l))$	**MP**, 40, 44, 48
(50)$\mathbf{B}^5_{0.63}\phi_{k,l}$	**A0**
(51)$\mathbf{B}^3_{0.7}(\chi_{k,l} \rightarrow \phi_{k,l}) \rightarrow (\mathbf{B}^3_1\chi_{k,l} \rightarrow \mathbf{B}^3_{0.7}\phi_{k,l})$	**A2**, **US**
(52)$\mathbf{B}^3_1\chi_{k,l} \rightarrow \mathbf{B}^3_{0.7}\phi_{k,l}$	**MP**, (c), 51
(53)$\mathbf{B}^3_1\chi_{k,l}$	**NEC**
(54)$\mathbf{B}^3_{0.7}\phi_{k,l}$	**MP**, 53, 52
(55)$\mathbf{B}^2_1(\perp\!\!\rightarrow \phi_{k,l})$	**A0**, **NEC**
(56)$\mathbf{B}^2_1(\perp\!\!\rightarrow \phi_{k,l}) \rightarrow (\mathbf{B}^2_0 \perp\!\!\rightarrow \mathbf{B}^2_0\phi_{k,l})$	**A2**, **US**
(57)$\mathbf{B}^2_0 \perp\!\!\rightarrow \mathbf{B}^2_0\phi_{k,l}$	**MP**, 55, 56
(58)$\mathbf{B}^2_0 \perp$	**A1**
(59)$\mathbf{B}^2_0\phi_{k,l}$	**MP**, 58, 57
(60)$\mathbf{D}_{0.96934}\phi_{k,l}$	**A4**, (a), 47, 50, 54, 59

We note that, by computing the distributed belief of the set of solutions in an EA (or agents in an MA), *it is possible to use this information to bias constructive algorithms*. This said, a Belief-Search-based EA can also benefit from constructive heuristics already available in the literature. In addition, *exact search methods can*

prioritize some pending decisions based on the information that is distributedly believed. This may also allow parallel search by a set of agents, allowing the agents to have many alternatives instead of the *depth-first* or *best-first* guiding procedures generally used.

By no means we affirm that $\mathbf{PL}_n^{\otimes}$ *is* the definitive logic that should be used to guide EAs with Belief Search. We mention this, since $\mathbf{PL}_n^{\otimes}$ is related to multi-modal logics of partial belief and it may be the case that some other forms for connectives are more appropriate than the T-norm proposed for merging information. However, $\mathbf{PL}_n^{\otimes}$ already embodies very interesting features that we would like to highlight and we have not noticed in other logics of belief. First, it allows *nested epistemic reasoning*, i.e., an expression like $\mathbf{B}_a^i\mathbf{D}_b\phi$ can be interpreted as *agent i believes at level a that ϕ is distributedly believed at level B"*. This is very interesting since some *ad-hoc* heuristics for generalizing recombination operators, like the *rebel*, *conciliator*, and *obsequent* behaviors [3], can be interpreted in terms of an underlying nested epistemic reasoning. Second, the negation is typically modal conveying the concept of absence of information. As remarked by Boldrin and Saffiotti, this contrast with the algebraic approach of other logics in which negation represents positive information on some "orthogonal" formula. Again, this is best illustrated with the MIN TSP as our favorite example. A strong belief on a subset of $O(n)$ edges to be in the optimal solution *does not necessary mean* that the $O(n^2)$ remaining edges might not be in the optimal solution. Modal logic seems to have an interesting role in this respect. Finally, according to Boldrin and Saffiotti, $\mathbf{PL}_n^{\otimes}$ can be extended to also include a set of epistemic operators D^G, with G being a subset of the agents. The intended meaning of this is that they will combine the distributed belief of subsets of the agents.

5 Challenge #4: Learning from other metaheuristics and other open challenges

Evolutionary Computation metaheuristics are far from being the only method of choice to perform heuristic search. We have shown in the introduction how *Simulated Annealing* (SA) and *Tabu Search* (TS) are among the most popular "'single-agent" stochastic optimization methods. The key of the success is the simplicity of their implementation and the fact that for many optimization problems (and the problem instances under study) it is relatively easy to get very good solutions.

One of the authors of this chapter, back in 1989, introduced the denomination of 'memetic algorithms' (MAs) as a paradigm aimed to liberate population-search methods from the current biologically motivated metaphors at that time. Several ideas were introduced, the use of single-agent metaheuristics for individual search optimization steps, the use of different neighborhoods for the different agents, the study of correlation of local optima, etc. After more than a decade from that work, we see that several ideas have been upraised up to the point of constituting new metaheuristics, like *variable neighborhood search* (VNS) [21]. We can quote from [29]:

> *Another advantage that can be exploited is that the most powerful computers in the network can be doing the most time-consuming heuristics, while others are using a different heuristics. The program to do local search in each individual can be different. This enriches the whole, since what is a* local minima *for one of the computers is not a local minima for another in the network. Different heuristics may be working fine due to different reasons. The collective use of them would improve the final output. In a distributed implementation we can think in a division of jobs, dividing the kind of moves performed in each computing individual. It leads to an interesting concept, where instead of dividing the physical problem (assignment of cities/cells to processors) we divide the set of possible moves. This set*

is selected among the most efficient moves for the problem.

and also,

> *Is this the ultimate solution for the problems that the search involves ? Is it wise to use a set of many different moves, to continue adding different moves* ad infinitum *? Certainly not. Effective moves are those that, on the average, create a new configuration with similar values of the objective value, reflecting the efficient use of the correlation between the configurations given by the representation.*

Despite the clear coincidences present in these early discussions, and contrarily to what the reader might suspect, we are not interested in claiming that the VNS ideas were already contained in MAs. On the contrary, we view the systematic development of particular strategies as a healthy sign. If a simpler metaheuristic (SA, TS, VNS, GRASP, etc.) performs the same as a more complex method (GAs, MAs, Ant Colonies, etc.) we should either resort to the simpler method, or to the one that has less free parameters, or to the one that is easier to implement. On the other hand, such a fact challenges us to adapt the more complex methodology to beat a simpler heuristic, if that is possible at all. What we do not consider as a healthy sign, however, are the attempts to encapsulate some metaheuristics on stretched confinements. For instance, a MA is not just a "hybrid" GA, or a "parallel GA", or a GA in which all solutions are local optima. Actually this latter strategy was not part of the proposed definitions, since already the MAs in 1988 and 1989 were using SA or stochastic methods and the solutions were far from being locally optimal at the time of recombination. Not every method that uses a population and a recombination operator is a GA, not every hybrid GA is a MA. An "ant colony" metaheuristic [12] is indeed a new idea, but when the "ants" use local search, the resulting algorithm exhibits a strong resemblance to an MA.

We think that there are several "learned lessons" from work in other metaheuristics. For instance, TS decides to accept another new configuration (whether a feasible solution or not) without restriction to the relative objective function value of the two solutions. This has lead to good performance in some configuration spaces where evolutionary methods and Simulated Annealing perform poorly. A classical example of this situation is the MIN NUMBER PARTITIONING problem [3]. In addition, we have also identified some problems with evolutionary search methods in instances of the TSP in which the entries of the distance matrix have a large number of decimal digits. We believe that there is an inherent problem to be solved, for evolutionary methods to deal with fitness functions that have so many decimal digits. Traditional rank-based or fitness-based selection schemes to keep new solutions in the current population fail. It would be then reasonable to investigate whether some ideas from basic TS mechanisms could be adapted to allow less stringent selection approaches.

Problems like *STRIPS planning* [5] or the less known *Sokoban* [10] can provide good test-beds for the performance of EC methods in problems of other complexity classes. Unfortunately, although there are exceptions [36] they are seldom addressed. Other related challenges have been described in [34]. Multi-objective optimization is another interesting field full of new challenges where several metaheuristics are being evaluated [7].

In [34] we can read in their second challenge:

> *Minsky (1967) was foundational in establishing the theory of computation, but after Hartmanis (1971) there has been a fixation with asymptotic complexity. In reality lots of problems we face in building real AI systems do not get out of hand in terms of the size of problems for individual modules– in particular with behavior-based systems most of the submodules need only deal with bounded size problems.*

We have recently initiated work in an area which we have tentatively called *Evolutionary Analysis of Algorithms* [8]. This approach deals with the problem of finding, for a fixed-size, the worst-case instance for a particular algorithm; there are problems that by their intrinsic nature have been defined with a natural upper-bound on the instance size. Then the real challenging problem is to find new methods allowing "co-evolution" between the tasks of designing a better algorithm and the worst-case instance. This hopefully will lead to more robust methodology for algorithms development.

6 Conclusions

By looking back at the development of Evolutionary Computation in the previous decades, we can say that it is a healthy field. The number of researchers and published articles is steadily growing at a superlinear rate [1]. So is also the number of successful applications of these techniques. Hence, the field is now well grounded and mature enough to endeavor the challenging task of understanding how, when and why these techniques work or should be deployed on an specific problem.

We have proposed a number of challenges whose successful resolution will –in our opinion– provide major boosts for the vigorous development of the field. Obviously, these challenges are only a part of a bigger picture, as the reader will verify by reading other essays in this collection. They nevertheless reflect our view of the area, a view in which the lack of a solid theoretical corpus as well as insufficient connections with other areas of metaheuristic optimization (let alone with other areas of Theoretical Computer Science) constitute a Damocles' sword whose existence we have to face (and indeed solve).

It is up to us, EC researchers, to determine whether future EC practitioners will regard the field as a collection of elaborate recipes to be adapted to one's taste, or as a cooking book from which to learn how to cook the dish he/she likes. Admittedly, this is an ambitious objective. It is also true that some of the most optimistic perspectives about the capabilities of the paradigm a decade ago were dismissed by theoretical results such as Hart & Belew's hardness results [22] and Wolpert & Macready's No Free Lunch Theorem [37], so in principle, this could be the case for some of these challenges. However, we have to consider that these past experiences did not compromise the future of EC; on the contrary, they allowed redirecting efforts in more fruitful ways. Theoretical results cannot thus be negative, for they represent the underlying truth about the paradigm. It is to this underlying ground upon which we have to settle and adapt. Whatever the outcome of the challenges we have depicted in this essay, this should be the philosophy with which we have to react.

References

[1] J.T. Alander. Indexed bibliography of genetic algorithms and neural networks. Technical Report 94-1-NN, University of Vaasa, Department of Information Technology and Production Economics, 1994.

[2] Th. Bäck, D.B. Fogel, and Z. Michalewicz. *Handbook of Evolutionary Computation*. Oxford University Press, New York NY, 1997.

[3] R. Berretta and P. Moscato. The number partitioning problem: An open challenge for evolutionary computation ? In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 261–278. McGraw-Hill, Maidenhead, Berkshire, England, UK, 1999.

[4] L. Boldrin and A. Saffiotti. A modal logic for merging partial belief of multiple reasoners. *Journal of Logic and Computation*, 9(1):81–103, 1999. Online at `http://www.aass.oru.se/~asaffio/`.

[5] T. Bylander. The computational complexity of propositional STRIPS planning. *Artificial Intelligence*, 69(1-2):165–204, 1994.

[6] J. Chen, I.A. Kanj, and W. Jia. Vertex cover: further observations and further improvements. In *Proceedings of the 25th International Workshop on Graph-Theoretic Concepts in Computer Science*, number 1665 in Lecture Notes in Computer Science, pages 313–324. Springer-Verlag, 1999.

[7] C.A. Coello. An updated survey of evolutionary multiobjective optimization techniques: State of the art and future trends. In Peter J. Angeline, Zbyszek Michalewicz, Marc Schoenauer, Xin Yao, and Ali Zalzala, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 1, pages 3–13, Piscataway, NJ, 1999. IEEE Press.

[8] C. Cotta and P. Moscato. A mixed evolutionary-statistical analysis of an algorithm's complexity. *Applied Mathematics Letters*, 16(1):41–47, 2003.

[9] C. Cotta and J.M. Troya. Genetic forma recombination in permutation flowshop problems. *Evolutionary Computation*, 6(1):25–44, 1998.

[10] J. Culberson. Sokoban is PSPACE-complete. In E. Lodi, L. Pagli, and N. Santoro, editors, *Proceedings in Informatics 4, Fun With Algorithms*, pages 65–76, Waterloo, 1999. Carleton Scientific.

[11] Y. Davidor. Epistasis variance: A viewpoint on GA-hardness. In G.J.E. Rawlins, editor, *Foundations of Genetic Algorithms*, pages 23–35. Morgan Kaufmann, 1991.

[12] M. Dorigo, V. Maniezzo, and A. Colorni. The ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26(1):29–41, 1996.

[13] R. Downey and M. Fellows. Fixed parameter tractability and completeness I: Basic theory. *SIAM Journal of Computing*, 24:873–921, 1995.

[14] R.G. Downey, M.R. Fellows, and U. Stege. Computational tractability: The view from mars. *Bulletin of the European Association for Theoretical Computer Science*, 69:73–97, 1999.

[15] R. Englemore and T. Morgan (eds.). *Blackboard Systems*. Addison-Wesley, 1988.

[16] M. Fellows. Parameterized Complexity: The main ideas and connections to practical computing. *Electronic Notes in Theoretical Computer Science*, 61, 2002. available at `http://www.elsevier.nl/locate/entcs/volume61.html`.

[17] S. Forrest and M. Mitchell. What makes a problem hard for a genetic algorithm? some anomalous results and their explanation. In R.K Belew and L.B. Booker, editors, *Proceedings of the 4th International Conference on Genetic Algorithms*, pages 182–189, San Mateo, CA, 1991. Morgan Kaufman.

[18] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, San Francisco, 1979.

[19] I.P. Gent and T. Walsh. The SAT phase transition. In A.G. Cohn, editor, *Proceedings of 11th European Conference on Artificial Intelligence*, pages 105–109. John Wiley & Sons, 1994.

[20] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Boston, MA, 1997.

[21] P. Hansen and N. Mladenović. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130(3):449–467, 2001.

[22] W.E. Hart and R.K. Belew. Optimizing an arbitrary function is hard for the genetic algorithm. In R.K. Belew and L.B. Booker, editors, *Proceedings of the 4th International Conference on Genetic Algorithms*, pages 190–195, San Mateo CA, 1991. Morgan Kaufmann.

[23] D.S. Johnson, C.H. Papadimitriou, and M. Yannakakis. How easy is local search ? *Journal of Computers and System Sciences*, 37:79–100, 1988.

[24] S. Kirkpatrick, C.D. Gelatt Jr., and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[25] M. Krivelevich. Sparse graphs usually have exponentially many optimal colorings. *Electronic Journal of Combinatorics*, 9(1):#R27, 2002.

[26] P. Larrañaga and J.A. Lozano, editors. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation.* Kluwer Academic Publishers, Boston, 2001.

[27] H.R. Lewis and C.H. Papadimitriou. *Elements of the Theory of Computation.* Prentice-Hall, Inc., Upper Saddle River, New Jersey, 1998.

[28] K. Mathias and D. Whitley. Genetic operators, the fitness landscape and the traveling salesman problem. In R. Männer and B. Manderick, editors, *Parallel Problem Solving From Nature II*, pages 259–268, Amsterdam, 1992. Elsevier Science Publishers B.V.

[29] P. Moscato. On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. Technical Report Caltech Concurrent Computation Program, Report. 826, California Institute of Technology, Pasadena, California, USA, 1989.

[30] P. Moscato. Memetic algorithms: A short introduction. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 219–234. McGraw-Hill, Maidenhead, Berkshire, England, UK, 1999.

[31] R. Niedermeier and P. Rossmanith. A general method to speed up fixed-parameter-tractable algorithms. *Information Processing Letters*, 73:125–129, 2000.

[32] I.M. Oliver, D.J. Smith, and J.R.C. Holland. A study of permutation crossover operators on the traveling salesman problem. In J.J. Grefenstette, editor, *Proceedings of the 2^{nd} International Conference on Genetic Algorithms*, pages 224–230, Hillsdale NJ, 1987. Lawrence Erlbaum Associates.

[33] N.J. Radcliffe and P.D. Surry. Fitness Variance of Formae and Performance Prediction. In L.D. Whitley and M.D. Vose, editors, *Proceedings of the 3^{rd} Workshop on Foundations of Genetic Algorithms*, pages 51–72, San Francisco, 1994. Morgan Kaufmann.

[34] Bart Selman, Henry A. Kautz, and David A. McAllester. Ten challenges in propositional reasoning and search. In *Proceedings of the 15^{th} International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 50–54, 1997.

[35] G. Syswerda. Uniform crossover in genetic algorithms. In J.D. Schaffer, editor, *Proceedings of the 3^{rd} International Conference on Genetic Algorithms*, pages 2–9, San Mateo, CA, 1989. Morgan Kaufmann.

[36] C. H. Westerberg and J. Levine. Investigation of different seeding strategies in a genetic planner. In E.J.W. Boers, S. Cagnoni, J. Gottlieb, E. Hart, P.L. Lanzi, G. Raidl, R.E. Smith, and H. Tijink, editors, *Applications of Evolutionary Computing*, volume 2037 of *Lecture Notes in Computer Science*, pages 505–514. Springer-Verlag, Berlin Heidelberg, 2001.

[37] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

[38] M. Yannakakis. Computational complexity. In E.H.L. Aarts and J.K. Lenstra, editors, *Local Search in Combinatorial Optimization*, pages 19–55. Wiley, Chichester, 1997.