Embedding Branch and Bound within Evolutionary Algorithms

Carlos Cotta and José M. Troya

Dept. Lenguajes y Ciencias de la Computación, University of Málaga
ETSI Informática (3.2.49), Campus de Teatinos, 29071 - Málaga, SPAIN
ccottap@lcc.uma.es

Abstract

A framework for hybridizing evolutionary algorithms with the branch-and-bound algorithm (B&B) is presented in this paper. This framework is based on using B&B as an operator embedded in the evolutionary algorithm. The resulting hybrid operator will intelligently explore the dynastic potential (possible children) of the solutions being recombined, providing the best combination of formae (generalized schemata) that can be constructed without introducing implicit mutation. As a basis for studying this operator, the general functioning of transmitting recombination is considered. Two important concepts are introduced, *compatibility sets*, and *granularity* of the representation. These concepts are studied in the context of different kinds of representation: orthogonal, non-orthogonal separable, and non-separable.

The results of an extensive experimental evaluation are reported. It is shown that this model can be useful when problem knowledge is available in the form of an optimistic evaluation function. Scalability issues are also considered. A control mechanism is proposed to alleviate the increasing computational cost of the algorithm for highly multidimensional problems.

1 Introduction

Evolutionary Algorithms [1] are powerful heuristics for optimization based on the principles of natural evolution, namely adaptation and survival of the fittest. These techniques are based on the iterative generation of tentative solutions for a target problem: starting from a *population* (pool) of randomly created *individuals* (solutions), a basic cycle comprising *selection* (promising solutions are picked from the pool), *reproduction* (new solutions are created by modifying selected solutions), and *replacement* (the pool is updated by replacing some existing solutions by the newly created ones) is performed. A *fitness* function measuring the goodness of solutions is used to drive the whole process, especially during the selection stage. Evolutionary computation constitutes nowadays a state-of-the-art approach to tackle hard optimization problems for which classical techniques are inadequate.

In spite of their boundaries becoming blurred nowadays, three main streams can be identified within the field of evolutionary computation: Evolutionary Programming [2], Genetic Algorithms [3], and Evolution Strategies [4]. Each of these families approaches evolutionary optimization by putting

emphasis on different aspects of the common underlying model. A disparity in both methodological and conceptual aspects of the optimization process arises from these different views of the field.

Among these methodological differences, the utilization of recombination operators (i.e., operators that create new solutions by combining information pieces taken from selected solutions) within the reproductive stage has always been a controversial issue. On one hand, many evolutionary-programming practitioners consider that recombination reduces in most cases to macromutation. On the other hand, recombination is assigned a paramount rôle by genetic-algorithm researchers. In fact, extended recombination mechanisms have been defined in which more than two individuals contribute to create a new solution [5].

These opposed arguments have motivated a plethora of theoretical studies to determine when and how to recombine. As to the first question, the most classical answer is Goldberg's building block hypothesis [6]. This hypothesis has been notably reformulated by Radcliffe [7], generalizing the concept of schema to abstract entities called *formae*, and defining representation-independent recombination operators with specific properties with respect to these formae [8]. The resulting framework (Forma Analysis) has provided very important insights on the functioning of genetic algorithms.

It is both the strength and the weakness of these representation-independent operators that their application is blind, i.e., randomly guided. The underlying idea is not to introduce any bias in the evolution of the algorithm, thus preventing premature convergence to suboptimal solutions. This intuitive idea is questionable though. First, notice that the evolution of the algorithm is in fact biased by the choice of representation and the mechanics of the particular operators. Second, there exist widely known mechanisms (e.g., spatial isolation [9, 10]) to promote diversity in the population, thus precluding (or at least hindering) extremely fast convergence to suboptimal solutions. Finally, it can be better to quickly obtain a suboptimal solution and restart the algorithm than using blind operators for a long time in pursuit of an asymptotically optimal behaviour.

This paper discusses the use of recombination operators that use problem knowledge to bias the generation of new solutions. To be precise, the problem knowledge is used to determine the best possible combination of the ancestors' features, thus removing the blindness of the recombination process. The utilization of these knowledge-augmented operators (also known as *hybrid* operators) has an additional motivation. As initially stated in [11] and later popularized in the so-called *No Free Lunch Theorem* [12] (see also [13]), using problem knowledge is not an optional mechanism for improving the performance of the algorithm, but it is a strong requirement for ensuring a minimal quality of the results. In this sense, the framework proposed and described in this paper constitutes another tool that evolutionary-algorithm designers can put into their toolbox, to be considered when trying to adapt their algorithm for a specific problem [14, 15, 16].

The remainder of the paper is organized as follows: first, and in order to make this work self-contained, the necessary concepts on forma analysis and notational details are given in Section 2. Then, the properties of different kinds of representation are studied in Section 3, introducing key concepts for the subsequent development. Next, the hybrid framework is presented in Section 4, describing its internal functioning, and analyzing factors with impact in the computational complexity

of the resulting algorithm. Subsequently, experimental results are reported in Section 5. Finally, conclusions are presented in Section 6, outlining future work as well. An appendix describing the test suite used in the experiments is also included.

2 Background

First of all, let $\Xi = \{\psi_1, \cdots, \psi_n\}$ be a set of n independent equivalence relations defined over the search space $\mathcal{S}$. Let it hold for Ξ that for all $x, y \in \mathcal{S}$, there exists $\psi_i \in \Xi$ such that $\psi_i(x, y) =$ `FALSE`, i.e., no two solutions share membership to the same equivalence classes for all equivalence relations in Ξ . In this case, Ξ is said to *cover* the search space, and each solution $x \in \mathcal{S}$ can be represented as $x = \{\eta_i, \cdots, \eta_n\}$, where η_i is the equivalence class[1] to which x belongs under ψ_i. Thus, $x = \langle \eta_i, \cdots, \eta_n \rangle \Leftrightarrow \{x\} = \cap_{i=1}^{n} \eta_i$. Each of these equivalence classes is termed a *basic forma* [7].

Formae allow encapsulating together arbitrary sets of solutions. From a practical point of view, the evolutionary-algorithm designer will identify relevant features of solutions (this can be done in a variety of ways, e.g., measuring fitness variance [17, 18]), defining equivalence relations that group together solutions sharing these features. Notice that equivalence relations are analogous to *genes*, while formae are analogous to *alleles*. The concept of *dynastic potential* can then be defined as follows:

Definition 1 (Dynastic Potential). Let $x = \{\eta_i, \cdots, \eta_n\}$ and $y = \{\zeta_i, \cdots, \zeta_n\}$ be two feasible individuals. Their dynastic potential is defined as $\Gamma(\{x, y\}) = \bigcap_{i=1}^{n} (\eta_i \cup \zeta_i)$, i.e., the set of individuals that only carry information contained in x and y. ■

Now, a recombination operator X can be defined as a probability distribution $X : \mathcal{S} \times \mathcal{S} \times \mathcal{S} \rightarrow [0, 1]$, where $X(x, y, z)$ represents the probability of generating z when recombining x and y. Subsequently, it is possible to define the concept of *Immediate Dynastic Span* [8] as follows[2]:

Definition 2 (Immediate Dynastic Span). The immediate dynastic span of two individuals x and y with respect to a recombination operator X is defined as $\Gamma_X^1(\{x, y\}) = \{z \mid X(x, y, z) > 0\}$, i.e., the set of solutions that can be obtained when X is applied on x and y. ■

Having defined these concepts, three major properties [17] can be present in a recombination operator X:

- Respect: $\{x, y\} \subseteq \eta \Rightarrow \Gamma_X^1(\{x, y\}) \subseteq \eta$.
- Proper assortment: $[(x \in \eta) \wedge (y \in \zeta) \wedge (\eta \cap \zeta \neq \emptyset)] \Rightarrow \Gamma_X^1(\{x, y\}) \cap \eta \cap \zeta \neq \emptyset$.
- Transmission: $\Gamma_X^1(\{x, y\}) \subseteq \Gamma(\{x, y\})$.

where x and y are arbitrary solutions, and η and ζ are two different formae.

The first property represents the exploitative side of recombination. A recombination operator is said to be *respectful* if, and only if, it generates descendants belonging to all basic formae common to both parents. This property becomes more important as the algorithm converges since diversity decreases and individuals are thus more likely to be similar.

On the other hand, assortment represents the exploratory side of recombination. A recombination operator is said to be *properly assorting* if, and only if, it can generate descendants carrying any

combination of compatible formae (i.e., formae η and ζ for which $\eta \cap \zeta \neq \emptyset$) taken from the parents. The assortment is said to be *weak* if it is necessary to perform several recombinations within the descendants to achieve this effect.

Finally, transmission is a very important property that captures the classical rôle of recombination. An operator is said to be transmitting if every basic forma to which the descendants belong contains at least one of the parents as well. Thus, a transmitting recombination operator combines the information present in the parents but does not introduce new information. This latter task, introducing new genetic material, is left to the mutation operator. For this reason, a non-transmitting recombination operator is said to introduce *implicit mutation*.

These three properties are not always simultaneously achievable. First, notice that gene transmission implies respect [8], but the reverse is not true in general (e.g., see [19]). Likewise, assortment and respect are not always compatible. For example, consider the edge-based representation of the TSP [20]; in the solutions $1-2-3-4-5-6$ and $1-2-4-6-3-5$, the undirected edges 23^u and 24^u are perfectly compatible but combining them excludes the common edge 12^u. In such a situation, the representation is said to be *non-separable*. Not all representations have this property. For example, consider the position-based representation of permutations [6, 18], i.e., considering the elements occurring in absolute positions as the basic features. It is easy to see that common positions can be respected and then any pair of compatible position formae (i.e., assignments of different elements to different positions) can be included. Such a representation is said to be *separable*.

Notice that an arbitrary assortment of separable formae generally results in the lack of full transmission (i.e., implicit mutation is introduced). For example, assume that $1-3-2-5-4-6$ and $1-2-3-4-6-5$ are recombined. Assorting 6 in the sixth position (first parent) and 4 in the fourth position (second parent) forces 5 to be placed in the 1st, 2nd, 3rd or 5th position, in none of which it appears in any parent.

Finally, the three properties are fully compatible in *orthogonal* representations (representations in which any tuple of formae corresponding to different equivalence relations are compatible), e.g., traditional schemata [6]. Unfortunately, no suitable orthogonal representation can be found for most problems. For that reason, usually a choice has to be made to decide which of these properties should be exhibited by the recombination operator.

In this sense, there exists empirical evidence suggesting respect as a desirable property, e.g., see [21, 18]. Furthermore, implicit mutation is usually regarded as undesirable [20]. For these reasons, transmission has been chosen as the central feature for the operators described in this work. This feature is further enhanced by the inclusion of problem-knowledge.

3 Forma Transmission and Representation Granularity

In light of the concepts presented above, a transmitting recombination can be generally considered as a process in which information is incrementally taken from the parents to construct the descendant (see Fig. 1), that is, starting from a totally unspecified solution (i.e., $\Psi_0^1 = \mathcal{S}$) properties from any of the parents are selected and assigned to the child until a fully-specified solution is obtained. As it

can be seen, each step involves determining a subset of the remaining unspecified properties of the descendant, and deciding from which parent is transmitted this information.

Fig. 1 here.

Let the notation $\xi \triangleright \Psi$ denote that, given $\Psi = \cap_{j=1}^{s} \theta_j$, $\exists j : \xi = \theta_j$, where ξ and θ_j are formae induced by the same equivalence relation $\psi_i \in \Xi$. Now, each of the pieces of information used in the recombination process sketched above will be called a *construction unit*, and can be formally defined as follows:

Definition 3 (Construction Unit). A construction unit $\Upsilon(\Psi, u, w)$ is any intersection of basic formae $\Theta \triangleq \cap_{j=1}^{g} \theta_j$, with $g \geq 1$, and $u \in \Theta$, such that $\Theta \cap \Psi \neq \emptyset$, and for any $\theta \triangleright \Theta$, it holds that $\theta \not\triangleright \Psi$, where Ψ is the partially specified solution, u and w are the parents, and g is a parameter termed *granularity of the representation*. ■

Notice that although w is not explicitly used in the above definition, it is relevant to determine what construction units are valid as it will be shown below. Construction units constitute the *information atoms* used to create the descendants, and their structure is clearly dependent of particulars of the representation. The simplest scenario is that in which the representation is orthogonal. In this case, the dynastic potential of x and y is

$$\Gamma(\{x, y\}) = \Gamma(\{\langle \eta_1, \cdots, \eta_n \rangle, \langle \zeta_1, \cdots, \zeta_n \rangle\}) \triangleq \prod_{i=1}^{n} \{\eta_i, \zeta_i\} \ , \tag{1}$$

i.e., the Cartesian product of all pairs $\{\eta_i, \zeta_i\}$, where $x \in \eta_i$, and $y \in \zeta_i$, for $1 \leq i \leq n$. Thus, it is possible to extend any partially specified solution by considering a single basic forma at a time, i.e.,

$$\Upsilon(\Psi, u, w) = \sigma(\Psi, u) = \sigma(\Psi, \langle \xi_1, \cdots, \xi_n \rangle) = \xi_i, \tag{2}$$

where $\sigma(\Psi, u) = \xi_i$ is the forma to which u belongs under the first unspecified equivalence relation (under a fixed arbitrary order) in Ψ. Let us define the concept of *dual* forma as follows:

Definition 4 (Dual Forma). The dual forma $\varpi(\eta, x, y)$ of a forma $\eta \ni x$ is $\zeta \ni y$ if, and only if, η and ζ are formae induced by the same equivalence relation $\psi_i \in \Xi$. ■

In orthogonal representations, decisions thus reduce to either considering $\Upsilon(\Psi, u, w)$ or $\Upsilon(\Psi, w, u) = \varpi(\Upsilon(\Psi, u, w), u, w)$. These representations have the finest granularity ($g = 1$). However, such a fine granularity is not always achievable. As an example, consider the position-based representation of permutations mentioned in Section 2. Clearly, using the construction units shown in Eq.(2) would be a waste of computational resources since many of the generated formae would consider repeated elements and hence would be empty (i.e., infeasible). Moreover, even when a particular forma Ψ were not infeasible per se, it might happen that $\Psi \cap \Gamma(\{x, y\}) = \emptyset$. This would be detected when in further steps all descendants Ψ' of Ψ in the decision tree shown in Fig. 1 were empty. All the computational effort needed to extend Ψ to Ψ' would have been useless. In this case, construction units must be more complex. To be precise, choosing a certain forma at a given step may force the inclusion or exclusion of other formae in further steps. This is formalized within the concept of *compatibility sets* as follows:

Definition 5 (Compatibility Set). The compatibility set of a forma η is inductively defined as:

$$\eta \triangleright K(\Psi, \eta, x, y) \tag{3}$$

$$\left[\Gamma(\{x, y\}) \cap \Psi \cap K(\Psi, \eta, x, y) \cap \varpi(\eta', x, y) = \emptyset\right] \Rightarrow \eta' \triangleright K(\Psi, \eta, x, y), \tag{4}$$

i.e., the intersection of all formae η' ($x \in \eta'$) that must be included along with η to preserve feasibility within the dynastic potential. ■

Example 6 (Compatibility Sets for Position Formae).
The position-based representation of n-element permutations is defined by a set of n equivalence relations $\Xi = \{\psi_1, \cdots, \psi_n\}$, where $\psi_i(x, y) = \texttt{TRUE}$ if, and only if, x and y have the same element at the ith position. Thus, every equivalence relation ψ_i induces n equivalence classes η_{ij}, each one comprising those solutions in which element j occurs at the ith position.

Assume now that two individuals $x = 1-2-3-4-5-6$ and $y = 6-3-2-1-5-4$ are being recombined under the assumption of transmission. If element 1 in the first position is transmitted to the descendant, so must element 6 in the sixth position (it could only appear in the 1st or 6th position, and the 1st is already occupied). This in turn implies that element 4 must be placed in the fourth position. Since element 1 appears in the fourth position of y, and this element has already been placed, no further formae need to be included in the compatibility set. These compatibility sets are termed *cycles* [18, 22] and can be inductively defined as follows [23]:

$$\eta_{ab} \triangleright K(\Psi, \eta_{ab}, x, y), \tag{5}$$

$$[\eta_{cd} \triangleright K(\Psi, \eta_{ab}, x, y) \wedge (y \in \eta_{ce}) \wedge (x \in \eta_{fe})] \Rightarrow \eta_{fe} \triangleright K(\Psi, \eta_{ab}, x, y). \tag{6}$$

Notice the cyclic nature of the compatibility set. Following the above example, the same result – i.e., $\langle 1\square\square 4\square 6\rangle$ – would have been obtained if we had started by transmitting element 4 or 6 in the 4th or 6th position respectively. On the other hand, transmitting the dual forma of η_{11}, i.e., η_{16}, would have resulted in the compatibility set $\langle 6\square\square 1\square 4\rangle$.

In the context of non-orthogonal representations, construction units are compatibility sets rather than single formae. It has been shown elsewhere that any solution in $\Gamma(\{x, y\})$ can be generated by considering equivalence relations in any arbitrary order, and taking binary decisions on the compatibility sets of the first unspecified forma and its dual one. Thus, the decision tree in Fig. 1 is in fact binary. This is a desirable property since it simplifies the decision structure.

The next step is computing the compatibility sets involved in these decisions. The following proposition shows that this computation can be very hard for non-separable representations:

Proposition 7 (Complexity of Computing Compatibility Sets). Computing compatibility sets in non-separable representations is $NP-$hard in general.

Proof of proposition 7: Consider the edge-based representation of permutations. As exemplified in Section 2, this is a non-separable representation. Now notice that the dynastic potential of two solutions x and y is the set of Hamiltonian cycles that can be found in the incomplete graph composed of those edges occurring in x or y. Deciding whether a certain forma (edge) ζ must be included

in a compatibility set $K(\Psi, \eta, x, y)$ implies determining whether a Hamiltonian cycle exists in the previously mentioned graph, forcing edges in Ψ and excluding ζ. This is known as the CONSTRAINED HAMILTONIAN CIRCUIT PROBLEM, an $NP-$hard problem [24]. □

Non-separable representations are thus difficult to tackle. Some work has been done regarding transmission in this kind of representations [25]. Nevertheless, it is clear that the study and design of efficient mechanisms for handling them is a very substantial topic. Hence, they are deferred to a further work. This does not leave a trivial problem at all, since very hard problems can still be defined in terms of separable representations. Furthermore, separable representations exhibit a nice property: compatibility sets do not depend upon the partially specified descendant (cf. [15]). This independence from decisions taken on-the-fly imply that they can be computed in advance at the beginning of the recombination process. The algorithm can subsequently handle them in the same way as single formae in orthogonal representations, i.e., as units that can be freely combined.

4 Dynastically Optimal Recombination

In light of the concepts presented above, the design of a recombination operator is addressed in this section. First, the necessity of utilizing knowledge in the recombination process is presented, introducing the dynastically optimal recombination operator. The functioning and requirements of this operator are subsequently described. Finally, the complexity of the operator is shown to be related to the granularity of the representation.

4.1 Random Transmission vs. Heuristic Recombination

A transmitting recombination operator is thus a mechanism for traversing the tree shown in Fig. 1. The easiest way for doing this is at random, i.e., starting at Ψ_0^1 and randomly selecting one of the available branches at each step. Operators exhibiting this behavior fall within the framework of Random Transmitting Recombination (RTR) [8]. According to the formulation of recombination presented in Section 2, this generic operator is defined as follows:

Definition 8 (Random Transmitting Recombination). The *Random Transmitting Recombination* operator is defined as

$$\mathrm{RTR}(x, y, z) = \begin{cases} \frac{1}{|\Gamma(\{x,y\})|} & z \in \Gamma(\{x, y\}) \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

■

Thus, RTR returns a random member of $\Gamma(\{x, y\})$, all individuals having the same probability of being selected, e.g., uniform crossover [26] in binary representations. As stated in [27], this uniform selection is inappropriate if problem-dependent knowledge is available. For example, consider the case in which the representation induced by Ξ is orthogonal. In this case, RTR traverses a complete binary tree of n levels, deciding at random in each level i whether the descendant should belong to $\eta_i \ni x$ or $\zeta_i \ni y$, where x and y are the parents. Now, if the representation is epistatic, the contribution of

each basic feature of a solution to its final quality depends upon which other features are present in the solution. In this situation, a random shuffle of this information is likely to disrupt the context in which a certain forma is immersed, thus resulting in pure macromutation. Hence, problem-dependent knowledge must be used to identify higher-order formae that can be satisfactorily linked. Moreover, even when no epistasis is involved, problem-knowledge can dramatically reduce the time required to find acceptable solutions.

The use of such heuristic knowledge is usually addressed in the literature in two ways:

1. *Memetic approach* [16]: after the child has been generated a local hill-climbing mechanism is applied[3].

2. *Patching by forma completion* [17]: a partially-specified child is generated using an operator that combines respect and assortment, e.g., the RAR_ω operator [8]. This child is then completed using either of the two following mechanisms:

   (a) *Locally optimal forma completion*: the gaps are randomly filled and then a local hill-climbing procedure is applied to them.

   (b) *Globally optimal forma completion*: the subspace of solutions matching the currently specified features is explored to determine the best completion of the child.

These two approaches are incompatible with the principle of forma transmission observed in this paper; the use of a local hill-climber will almost surely introduce a considerable amount of implicit mutation. Although this is not necessarily bad (in fact, local-search-based memetic algorithms have been quite successful in several problem domains), it complicates both a general analysis and a performance prediction. This is also true for globally optimal forma completion which, additionally, can be computationally expensive.

The approach proposed in this work tries to combine the positive features of the RTR operator and the two heuristic approaches described above. Like RTR, it is a strictly transmitting operator. Like the heuristic approaches, problem-knowledge is used to select a feasible forma combination but, unlike globally optimal forma completion, only the dynastic potential is explored and hence the computational cost is considerably reduced. This approach is termed *Dynastically Optimal Recombination*, and is defined as follows:

Definition 9 (Dynastically Optimal Recombination). Let $\phi : \mathcal{S} \to \mathbb{R}$ be the fitness function. Let the notation $\phi[\mathcal{R}]$ be used to represent the image of $\mathcal{R} \subseteq \mathcal{S}$ under ϕ, i.e., $\phi[\mathcal{R}] = \{\phi(w) \mid w \in \mathcal{R}\}$. Let $\prec$ be a partial order relation defined over $\phi[\mathcal{S}]$, such that x is a better solution than y if, and only if, $\phi(x) \prec \phi(y)$. Then, the *Dynastically Optimal Recombination* operator is defined as a transmitting recombination operator for which it holds that

$$\mathrm{DOR}(x, y, z) > 0 \Rightarrow \phi(z) \in \mathbf{sup}_{\prec} \left(\phi\left[\Gamma(\{x, y\})\right]\right). \tag{8}$$

■

Hence, DOR returns the best individual (or one of the best individuals in case there were several solutions with the same best fitness) of the dynastic potential. The way this is achieved will be shown below.

4.2 The Internal Functioning of DOR

According to the definition of the DOR operator given in Subsection 4.1, it appears that an exhaustive search mechanism must be used on the dynastic potential $\Gamma(\{x, y\})$ in order to determine the returned child. For this purpose, a very efficient option is to use branch-and-bound (B&B) for incrementally constructing solutions. This mechanism is described below in more detail.

According to the notation shown in Fig. 1, let Ψ_i^j, where $0 \leq i \leq n$, and $2^i \leq j \leq 2^{i+1} - 1$, represent a partially specified solution located at level i. Initially, $\Psi_0^1 = \mathcal{S}$; subsequently,

$$\Psi_{i+1}^{2j} \triangleq \Psi_i^j \cap \Upsilon(\Psi_i^j, x, y), \quad \text{and} \tag{9}$$

$$\Psi_{i+1}^{2j+1} \triangleq \Psi_i^j \cap \Upsilon(\Psi_i^j, y, x), \tag{10}$$

are considered, where the construction units $\Upsilon(\Psi_i^j, u, w)$ are the corresponding compatibility sets of the formae induced by the first unspecified equivalence relation in Ψ_i^j.

Now, it is necessary to use a function $\phi^* : 2^{\mathcal{S}} \rightarrow \mathbb{R}$ meeting the following requirements:

1. Monotonic growth: $\forall \mathcal{R} \subseteq \mathcal{S}\ \nexists \mathcal{T} \subset \mathcal{R} :\ \phi^*(\mathcal{T}) \prec \phi^*(\mathcal{R})$
2. Optimistic estimation: $\forall \mathcal{R} \subseteq \mathcal{S}\ \nexists r \in \phi[\mathcal{R}] :\ r \prec \phi^*(\mathcal{R})$
3. Accuracy in the limit: $\forall x \in \mathcal{S} :\ \phi^*(\{x\}) = \phi(x)$
4. Infeasibility avoidance: $\forall s \in \phi[\mathcal{S}] :\ s \prec \phi^*(\emptyset)$

Actually, ϕ^* need not be defined over all arbitrary sets of solutions, but just on sets defined by the intersection of the basic formae induced by equivalence relations in Ξ. This function comprises the available knowledge about the fitness function, providing optimistic estimations of the quality of partially-specified solutions. Following the well-known B&B algorithm [28], these estimations are used to traverse the tree shown in Fig. 1, determining the order in which formae Ψ_i^j are generated, and discarding those formae Ψ for which $\phi_{\text{best}} \prec \phi^*(\Psi)$, where ϕ_{best} is the fitness of the best-so-far generated solution. This latter value is updated whenever a forma $\Psi' = \{z\}$ is generated such that $\phi(z) \prec \phi_{\text{best}}$. Initially, $\phi_{\text{best}} = \mathbf{inf}_{\prec}\{\phi[\mathcal{S}]\}$.

Notice that, if the representation is non-epistatic, DOR admits a very simple implementation. Such implementation is based on the fact that the fitness function can be decomposed in this case as

$$\phi(\{\xi_1, \cdots, \xi_n\}) = \sum_{i=1}^{n} \phi_i(\xi_i). \tag{11}$$

For this kind of function, DOR must simply take local decisions at each level, selecting the formae that individually optimize the fitness function. Hence it has linear-time complexity. The scenario is different when epistasis is involved, as shown in the next subsection.

4.3 Representation Granularity and the Complexity of DOR

The granularity of the representation has an unquestionable impact on the time complexity of the algorithm. Consider that the size of the dynastic potential is $O(2^m)$, where m is the number of construction units that can be identified in the selected individuals. In the case of orthogonal representations, each construction unit comprises a single forma (i.e., $m = n$, where n is the dimensionality of the problem), and thus the size of the dynastic potential is $O(2^n)$. Obviously, and as mentioned above, this size is only relevant when epistasis is involved, since DOR behaves linearly otherwise.

In the case of epistatic representations, the exponential growth of the dynastic potential becomes very important. As an example, consider the curve labeled as 'g=1' in Fig. 2 (left). It shows the time[4] required for performing one hundred recombinations of randomly generated individuals as a function of the dimensionality of the problem (the design of a brachystochrone in this case). As it can be seen, the computational cost grows extremely fast for dimensionalities above 10, thus confirming the influence of the number of construction units. Of course, there exist other factors affecting the time complexity of the operator, namely the amount of knowledge used in the ϕ^* function. If this function were defined to return a constant value for underspecified solutions (i.e., if no problem-knowledge were used), DOR would reduce to an exhaustive enumeration of all members of the dynastic potential and hence $O(2^n)$ individuals would have to be evaluated. On the contrary, if ϕ^* provides good estimations of the fitness function this quantity can be dramatically reduced, so as to become affordable for a certain range of dimensionalities.

It is interesting to notice that, as the algorithm converges, individuals tend to be more similar and hence their dynastic potential is reduced. To be precise, suppose that the individuals to be recombined share membership to a set of basic formae $\Theta = \theta_{i_1} \cap \cdots \cap \theta_{i_k}$. Given that gene transmission implies respect [8], it is clear that $\Gamma(\{x, y\}) \subseteq \Theta$. Hence, the search can be started from $\Psi_0^1 = \Theta$, and $|\Gamma(\{x, y\})| \leq 2^{n-k}$.

Nevertheless, it must be taken into account that, even when the above consideration is true, it does not necessarily imply that the number of construction units in non-orthogonal representations decreases monotonically as well. This follows from the fact that having common formae allows bounding the maximum size of the dynastic potential, but its actual size may vary below that bound. This is shown in the following example.

Example 10 (Common Formae and Dynastic Potential).
Consider the permutations $1-2-3-4-5-6-7-8$ and $2-3-1-5-6-7-8-4$. From the point of view of position formae, they do not belong to any common basic forma, existing two compatibility sets in each permutation: $\langle 123\square\square\square\square\square\rangle$ and $\langle\square\square\square 45678\rangle$ from the first one and $\langle 231\square\square\square\square\square\rangle$ and $\langle\square\square\square 56784\rangle$ from the second one. Thus, there are 2^2 solutions in their dynastic potential.

On the other hand, the permutations $1-2-3-4-5-6-7-8$ and $1-3-2-4-6-5-8-7$ belong to two common position formae. However, there exist three non-common compatibility sets in each of them: $\langle\square 23\square\square\square\square\square\rangle$, $\langle\square\square\square 56\square\square\rangle$ and $\langle\square\square\square\square\square\square 78\rangle$ in the first one and $\langle\square 32\square\square\square\square\square\rangle$, $\langle\square\square\square\square 65\square\square\rangle$ and $\langle\square\square\square\square\square\square 87\rangle$ in the second one. Hence their dynastic potential has 2^3 solutions.

The above example simply illustrates that, even when diversity tends to decrease due to the

natural evolution of the algorithm, an additional control has to be put on the number of construction units should the computational cost of the algorithm become prohibitive for the problem under consideration. This is specifically true in highly multidimensional problems as shown in Fig. 2. Such a control can be achieved by tuning the granularity parameter g.

First of all, consider the case of orthogonal representations. The granularity may be increased by making construction units contain more than one basic forma. These formae can be selected at random or using a priori knowledge about the epistatic relations of these formae. To be precise, in the problem considered above there exist epistatic relations between adjacent variables. For this reason, it is convenient to choose consecutive variables, thus preserving a larger part of their context. Hence,

$$\Upsilon(\Psi_i^j, u, w) = \Upsilon(\Psi_i^j, \langle \xi_1, \cdots, \xi_n \rangle, w) = \bigcap_{k=1}^{\min(g, n-i\cdot g)} \xi_{i\cdot g+k}. \tag{12}$$

The curves labeled with 'g=2' to 'g=5' in Fig. 2 (left) show the results of the same benchmark mentioned above for higher dimensionalities. As it can be seen, when the granularity is increased, the algorithm reduces its computational cost, being capable of tackling larger problem instances. Moreover, and as it can be seen in Fig. 2 (right), there exists a very good linear relation between the granularity of the representation and the highest affordable dimensionality of the problem when a constant computational cost is kept.

Fig. 2 here.

A similar approach can be taken when the representation is non-orthogonal although, in this case, compatibility sets rather than single basic formae must be joined. For instance, this is the underlying idea of the block representation [18], defined as follows:

Definition 11 (Block Formae). A block $B(k, x, y)$ is a macro-compatibility-set inductively defined on the basis of cycles as

$$x \in \eta_{ka} \Rightarrow \forall \zeta \triangleright K(\eta_{ka}, x, y) : \ \zeta \triangleright B(k, x, y), \tag{13}$$

$$(x \in \eta_{sb}) \wedge [\eta_{rc}, \eta_{td} \triangleright B(k, x, y)] \wedge (r < s < t) \Rightarrow \forall \zeta \triangleright K(\eta_{sb}, x, y) : \ \zeta \triangleright B(k, x, y), \tag{14}$$

where k is the index of the first position of the block. In the equations above, the first parameter of compatibility sets $K(\cdot)$ – the partially specified solution – has been dropped because of the independence mentioned in Section 3. ■

Example 12 (Blocks vs. Cycles).
Consider the permutations $1-2-3-4-5-6-7-8-9$ and $4-3-2-1-6-5-9-8-7$. According to the definition of cycles presented in Example 6, the following compatibility sets can be identified in the first solution: $\langle 1\square\square 4\square\square\square\square\square\rangle$, $\langle \square 23\square\square\square\square\square\rangle$, $\langle \square\square\square\square 56\square\square\rangle$, $\langle \square\square\square\square\square\square 7\square 9\rangle$, and $\langle \square\square\square\square\square\square\square 8\square\rangle$. On the other hand, considering the previous definition of blocks, there exist just three compatibility sets: $\langle 1234\square\square\square\square\square\rangle$, $\langle \square\square\square\square 56\square\square\square\rangle$, and $\langle \square\square\square\square\square\square 789\rangle$. We obtain a lower number of construction units but each of them has a more coarse granularity.

As shown in [23], the average size of the dynastic potential of recombined solutions is more than one order of magnitude smaller using the block representation that the cycle representation. This

implies that the computational cost of DOR will be considerably reduced as well. Nevertheless, an important fact must be taken into account: reducing the dynastic potential also reduces the chances for combining valuable information taken from the parents. In an extreme situation, this could be as undesirable as the unaffordable computational cost of DOR for low (respectively high) values of g (respectively n). This is one of the issues studied in next section.

5 Experimental Results

A set of experiments has been realized in order to assess the usefulness of DOR as a tool for hybridization. The test suite used has been chosen to comprise complex problems exhibiting different properties such as orthogonality/non-orthogonality, and presence/absence of epistasis. This test suite is described in the appendix.

Except where otherwise noted, an elitist generational genetic algorithm (*popsize*=100, probability of recombination p_c=.9, *maxevals* = 10^5) using linear ranking selection (with maximum expected value $\eta^+ = 2.0$ [29]) has been utilized. In order to make a fair comparison between DOR and other classical operators, the internal partial evaluations of the former have been taken into account. More precisely, computing the optimistic evaluation $\phi^*(\Upsilon)$ of a construction unit Υ of granularity g is accounted as g/n evaluations, where n is the dimensionality of the problem. Thus, comparisons are done on the basis of an analogous utilization of resources, since the evaluation of a solution is usually considered the basic unit to measure the computational cost of an evolutionary algorithm (actually, all algorithms considered behave similarly in terms of computational time). Also, mutation is performed before recombination (hence new information is still introduced in the population but the smart recombination performed by DOR is preserved). No fine tuning of parameters has been attempted.

The first results (shown in Fig. 3) correspond to the Schwefel function (§A.1), a non-epistatic orthogonal function. In this problem, variables have been encoded using 64 bits, and the probability of mutation p_m has been set to 1/64. For comparison purposes, the experiments have been realized with some classical operators such as single-point crossover (SPX), double-point crossover (DPX), uniform crossover (UX), arithmetic crossover (AX) and random respectful recombination (R^3) [30]. It can be seen that, while most operators behave very similarly for low dimensionalities, DOR becomes nearly two orders of magnitude better than the remaining operators for 32 variables. As mentioned before, this kind of function is very easy for DOR to optimize. Thus, more difficult problems are considered henceforth.

Fig. 3 here.

The first epistatic problem considered is the design of a reactive rulebase for a mobile agent (§A.2). This rulebase must make the agent reach a destination point in minimal time starting from an arbitrary point. It is clear that this problem is strongly epistatic since learning the rulebase implies determining a set of rules whose interplay results in an appropriate behavior (the choice of the training set ensures that every rule is activated sometime). The optimistic evaluation function $\phi^*(\mathcal{B})$ –where $\mathcal{B}$ is a partially specified rulebase– is simply defined as the number of steps the agent would perform

until falling in a state for which no rule is specified, plus the Manhattan distance from the current location to the target point. A rulebase is encoded as a linear string, each position containing a symbol representing the specific action to be undertaken upon reception of the corresponding sensorial input.

The results for nine different environments are shown in Table 1. As it can be seen, DOR is much more satisfactory than SPX, DPX or UX in solving the training set: while standard operators only provide an acceptable performance on the smallest instances and with the lowest obstacle-density, DOR consistently yields satisfactory results: above 70% of the runs provide a fully successful solution for the training set (the percentage is 100% for 5 out of 9 test worlds). The exception is world W50c, a very hard instance for which none of the operators could find a full solution (it must be noted that such a solution may be non-existent). Nevertheless, DOR was capable of solving 3 out of the 5 training cases while SPX could only solve one and neither DPX nor UX could solve any of them. Furthermore, DOR also provides higher-quality results, close to the optimal/best-known solutions [31]. The results obtained on the test set are consistent with the above considerations. Notice the poor results of standard operators: the solutions provided by UX, DPX and SPX do not reach 50% success in 6 out of 9 worlds. DOR provides the overall best results, outperforming standard operators on all worlds. It must also be noted that pure B&B provides worse results than DOR in small problem instances, and runs out of memory in larger problem instances.

`Table 1 here.`

The next epistatic problem considered was the brachystochrone design problem (§A.3). In this problem, variables are encoded with 16 bits, p_m is set to 1/16 and $maxevals = 5 \cdot 10^5$. The optimistic evaluation function is $\phi^*(\langle h_0, h_1, \cdots, h_i \rangle) = t_i + d^*/v^*$, where t_i is the time needed to travel from the starting point to the ith pillar, d^* is the straight distance from the last specified location to the destination point, and v^* is the maximum of v_i (velocity at the ith pillar) and v_{n+1} (velocity at the end of the track).

The results are shown in Table 2. As it can be seen, low values of g are better when the dimensionality of the problem is small, becoming the quality of the results slightly worse when the granularity is increased. For large dimensionalities of the function (i.e., above 32), low values of g are either prohibitive or provide worse results. This is due to the fact that a higher number of construction units are manipulated and hence the algorithm consumes very quickly the allocated number of epistatic calculations.

`Table 2 here.`

To confirm these results, this same problem was tackled using a constant ratio $n/g \approx 10$, and a different base-algorithm, an evolution strategy. To be precise, a (2,20)-ES with independent stepsizes for each variable was used. These stepsizes underwent self-adaptation using a global learning rate $\tau' = (2n)^{-1/2}$ and a local learning rate $\tau = (4n)^{-1/4}$ [32]. The results are shown in Table 3. Besides obtaining globally better results than with the genetic algorithm, DOR keeps outperforming the remaining operators, thus backing up the previous results.

`Table 3 here.`

Finally, experiments were done using the $k-$Epistatic Minimal Permutation ($k-$EMP) problem

(§A.4), considering instances of different dimensionalities and degrees of epistasis. The first results correspond to 1−EMP instances. As for the previous test problems, other recombination operators for permutations such as PMX [33], three variants of OX [34, 35, 14], RCX, UCX, BX and UBX (see [18]) have been also used. The results are shown in Table 4. Notice the poor results of the different variants of order crossover. This is due to the fact that this problem is defined on the basis of position formae rather than precedence formae. For that reason, UCX (Uniform Cycle Crossover) provides the best results. It is interesting to notice that this operator is based on a blind interchange of the compatibility sets defined in Eqs.(5) and (6), i.e., UCX is $\text{RTR}^{\text{cycle}}$.

`Table 4 here.`

Table 5 shows the results of the DOR operator. In this problem, the optimistic evaluation function ϕ^* applied on a partially specified permutation simply computes the epistatic contribution of specified positions, using a lower bound for the epistatic coefficients of unspecified solutions. The granularity is tuned by defining a maximum allowed number of construction units f (recall that the size of each construction unit is not under direct control of the user). Whenever the number of construction units is greater than f, two of them are picked at random and joined. This process is repeated until at most f construction units are available. Compacting cycles into blocks as indicated in Eqs.(13) and (14) has also been tried.

`Table 5 here.`

As it can be seen, the results for very coarse granularities (i.e., low values of f) are worse than for fine granularities. $\text{DOR}^{\text{block}}$ is included in the former category since blocks are larger units than cycles. In fact, the performance of $\text{DOR}^{\text{block}}$ is intermediate between $f = 2$ and $f = 3$. Notice that the performance is stabilized around $f = 6$, with a slight tendency to decrease for higher values. In any case, the quality of the results is always much better than UCX (with the exception of the extreme situation $f = 2$).

To confirm these results, further experiments with the DOR operator have been realized with higher degrees of epistasis ($k = 2$, $k = 5$, and $k = 10$). The results are shown in Table 6. In these experiments, the behavior of the algorithm is clearer. First, notice that the quality of the results is improved when f is increased up from $f = 2$. The best results are achieved in the intermediate values $f = 5$ and $f = 6$, existing a soft degradation of performance for higher values of f. The reason for this behavior has already been mentioned: low values of f reduce the chances for transferring information from parents to descendants, while high values of f quickly consume the allocated number of epistatic calculations.

`Table 6 here.`

6 Conclusions

This work has presented a framework for hybridizing evolutionary algorithms with the branch-and-bound algorithm. Within this framework, the available knowledge about the fitness function is exploited by intelligently combining valuable parts of solutions independently discovered. As mentioned in Section 1, this particular type of hybridization is another tool available in the evolutionary-

algorithm designer's toolbox, and should not be seen as the ultimate tool for solving any problem (actually, such an assertion would be precluded by the results of [11, 12] among others). This model is useful when problem dependent knowledge can be expressed in terms of an optimistic evaluation function ϕ^*. In fact, the better these optimistic estimations are, the more effective the resulting hybrid algorithm will be. Obviously, this requires the involvement of the problem-aware user in the design process, so as to provide sensible knowledge on the innards of the target problem. The extent of this involvement clearly depends on the particular problem under consideration, and it should not be seen as a disadvantageous fact. Actually, Lawrence Davis already argued for this involvement in the "Handbook of Genetic Algorithms" [14], back in 1991. The final user is precisely the one who can provide the most valuable information in terms of wise representations of solutions, or existing *ad hoc* algorithms. This cooperation between the problem-aware user and the evolutionary-algorithm expert is certainly a recipe for success.

To study the functioning of this heuristic recombination model (DOR), we have focused on the construction units used for creating new solutions. To be precise, the size of these construction units (i.e., the so-called granularity of the representation) has been considered as a central factor for determining the computational cost of the operator and the quality of the results it provides. It has been shown that there exists a basic (ground-level) granularity for each representation. This basic granularity is minimal in the case of orthogonal representations, in which basic formae can be freely combined. However, the basic granularity is variable when dealing with non-orthogonal separable representations. In this situation, it depends upon the size of the basic transference units (i.e., the compatibility sets), and it is not under direct control of the user.

This relationship between the granularity (and hence the size of the dynastic potential of the solutions to be recombined) and the computational cost of the DOR operator has been empirically corroborated. Thus, it has been proposed to increase the granularity factor (i.e., use larger construction units) to reduce this cost. In fact, this parameter can be adjusted according to the available computational resources to allow a finer exploration.

An extensive experimental investigation has been done in order to assess the performance of DOR, and its usefulness as a hybridization mechanism. It must be stressed that the focus of this experimental investigation has been to compare DOR with other operators on a *ceteris paribus* basis. The results have been very satisfactory, since other classical operators have been outperformed on a benchmark comprising several multimodal epistatic problems. Tuning the granularity of the representation appears as an acceptable mechanism to control the computational cost of the algorithm. It has been shown that the performance is significantly degraded only for significantly larger construction units. Moreover, intermediate granularities provide better results than very fine granularities since the former consume less computational resources (and hence the algorithm can be executed for a larger number of iterations).

A very interesting extension of this work would be the use of mechanisms for adapting or self-adapting the granularity used by DOR during the run. Intuitively, it is possible to use a very coarse granularity in the early stages of evolution, using a progressively finer granularity later. Nevertheless,

a more careful study of this possibility is required.

Additionally, the use of a truncated B&B search is also a valid option to further improve the performance of the algorithm. This is based on the fact that for some problems, B&B uses more resources for establishing the optimality of the final solution than for finding it. Work is in progress in this area.

A Description of the Test Problems

A.1 Generalized Schwefel Function

The generalized Schwefel function is a non-linear minimization problem whose n-dimensional form is defined as $F(\vec{x}) = n \cdot V - \sum_{i=1}^{n} x_i \cdot \sin(\sqrt{|x_i|})$ for $x \in [-512, 511]$, where $V = \max\{x \cdot \sin(\sqrt{|x|}) : x \in [-512, 511]\}$. The value for V depends upon the floating-point accuracy of the underlying system, being about 418.98288. This function is continuous and highly multimodal for gradient-based techniques. The global minimum is 0, being its location dependent of the underlying system.

A.2 Rulebase Learning in Mobile Agents

In this problem, a mobile agent located in a two-dimensional toroidal grid-world is considered. The purpose of the agent is to reach a certain target point from its initial location within an allowed time. To do so, the agent is capable of making some elementary actions such as moving straight ahead a single grid square, turning 90º to its left, or turning 90º to its right. While navigating, the agent must avoid some obstacles distributed all over the world. For this purpose, it is equipped with proximity sensors that can inform of the presence or absence of obstacles in front of the agent, 90º to its left, or 90º to its right. In addition, these sensors can also detect whether the target point is in any of these three locations or not.

The agent is equipped with a direction sensor as well. This sensor allows determining in which of four imaginary regions of the world the target point is located. These regions are determined by computing the angle between the target point and the current agent orientation, and calculating in which quadrant it falls. The toroidal shape of the world is taken into account in these calculations.

To control the agent, a reactive rulebase must be found. This rulebase defines a stimulus-to-response mapping, i.e., the agent receives some sensorial input about its local environment and, exclusively on the basis of such information, decides which primitive action must be undertaken.

A set of nine different worlds has been considered for this problem. These worlds are named as Wxy, where $x \in \{10, 25, 50\}$ indicates the dimension of the world (each world is a $x \times x$ grid), and $y \in \{a, b, c\}$ indicates the density of obstacles (5%, 10% and 20% respectively). For each world, a training set of five cases has been selected. The fitness function is the sum of the number of steps required to reach the destination in each training case (or the number of steps taken plus the Manhattan distance to the target plus a penalty term –the maximum number of steps– if the agent could not solve the corresponding training case). Depending upon the size of the world, the agent is

allowed a maximum number of 25, 150, and 400 time steps respectively. A test set of 50, 400 and 2000 cases respectively is used to evaluate the generalizability of the agent behavior.

A.3 Design of a Brachystochrone

The design of a brachystochrone is a classical problem of the calculus of variations. This problem involves determining the shape of a frictionless track along which a cart slides down by means of its own gravity, so as to minimize the time required to reach a destination point from a motionless state at a given starting point. To approach this problem by means of evolutionary algorithms, the track is divided into a number of equally spaced pillars. Subsequently, the algorithm determines the heights of each of these pillars.

As stated above, the objective is to minimize the time required by the cart to traverse the track. This time can be calculated as the sum of the times required for moving between any two consecutive pillars, i.e., $t = \sum_{i=1}^{n+1} t_i = \sum_{i=1}^{n+1} t(h_{i-1}, h_i, \lambda, v_{i-1})$, where n is the number of pillars, h_i is the height of the ith pillar (h_0 and h_{n+1} are data of the problem), λ is the distance between consecutive pillars (a problem parameter as well), and $v_i = v(v_{i-1}, h_{i-1}, h_i)$ is the velocity at the ith pillar ($v_0 = 0$).

As it can be seen, this is also an epistatic problem: the contribution of each variable (i.e., pillar height) depends on the value of previous variables; The experiments with this function have been carried out using $h_0 = 2$, $h_{n+1} = 0$, $(n+1) \cdot \lambda = 4$, and $(2h_{n+1} - h_0) \leq h_i \leq h_0$.

A.4 $k-$Epistatic Minimal Permutation

The k-Epistatic Minimal Permutation ($k-$EMP) problem is a generalization of the Minimal Permutation (MP) problem [19]. The latter is a minimization problem defined by a $n \times n$ matrix $M = \{m_{ij} \mid 1 \leq i, j \leq n\}$ such that each row of M is a permutation of the elements $\{0, \cdots, n-1\}$ and no column has more than one zero. Subsequently, a permutation $p = p_1 p_2 \cdots p_n$ is evaluated as $\mathrm{MP}(p) = \sum_{1 \leq i \leq n} m_{i,p_i}$. The constraints posed on M ensure that there is a unique permutation (the minimal permutation) whose fitness value is 0.

The $k-$EMP problem adds epistatic relations to the expression above, being defined as

$$k - \mathrm{EMP}(p) = \sum_{1 \leq i \leq n} \left[m_{i,p_i} \cdot \prod_{j=\min(1,i-k)}^{i-1} \alpha(p_i, p_j) \right] . \tag{15}$$

In the instances considered in this work, the coefficients $\alpha(p_i, p_j)$ are drawn from a uniform distribution in $[1, 2]$.

References

[1] T. Bäck, D. Fogel, Z. Michalewicz, Handbook of Evolutionary Computation, Oxford University Press, New York NY, 1997.

[2] L. Fogel, A. Owens, M. Walsh, Artificial Intelligence Through Simulated Evolution, Wiley, New York NY, 1966.

[3] J. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Harbor, 1975.

[4] H.-P. Schwefel, Evolution strategies: A family of non-linear optimization techniques based on imitating some principles of natural evolution, Annals of Operations Research 1 (1984) 165–167.

[5] A. Eiben, P.-E. Raue, Z. Ruttkay, Genetic algorithms with multi-parent recombination, in: Y. Davidor, H.-P. Schwefel, R. Männer (Eds.), Parallel Problem Solving From Nature III, Vol. 866 of Lecture Notes in Computer Science, Springer-Verlag, Berlin Heidelberg, 1994, pp. 78–87.

[6] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.

[7] N. Radcliffe, Equivalence class analysis of genetic algorithms, Complex Systems 5 (1991) 183–205.

[8] N. Radcliffe, The algebra of genetic algorithms, Annals of Mathematics and Artificial Intelligence 10 (1994) 339–384.

[9] R. Tanese, Distributed genetic algorithms, in: J. Schaffer (Ed.), Proceedings of the 3rd International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989, pp. 434–439.

[10] P. Spiessens, B. Manderick, A massively parallel genetic algorithm, in: R. Belew, L. Booker (Eds.), Proceedings of the 4th International Conference on Genetic Algorithms, Morgan Kauffman, San Mateo CA, 1991, pp. 279–286.

[11] W. Hart, R. Belew, Optimizing an arbitrary function is hard for the genetic algorithm, in: R. Belew, L. Booker (Eds.), Proceedings of the 4th International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo CA, 1991, pp. 190–195.

[12] D. Wolpert, W. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1(1) (1997) 67–82.

[13] J. Culberson, On the futility of blind search: An algorithmic view of "no free lunch", Evolutionary Computation 6 (2) (1998) 109–128.

[14] L. Davis, Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York NY, 1991.

[15] C. Cotta, A study of hybridisation techniques and their application to the design of evolutionary algorithms, AI Communications 11 (3-4) (1998) 223–224.

[16] P. Moscato, Memetic algorithms: A short introduction, in: D. Corne, M. Dorigo, F. Glover (Eds.), New Ideas in Optimization, McGraw-Hill, 1999, pp. 219–234.

[17] N. Radcliffe, P. Surry, Fitness variance of formae and performance prediction, in: L. Whitley, M. Vose (Eds.), Foundations of Genetic Algorithms III, Morgan Kauffman, San Mateo CA, 1994, pp. 51–72.

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

[19] C. Cotta, E. Alba, J. Troya, Utilising dynastically optimal forma recombination in hybrid genetic algorithms, in: A. Eiben, T. Bäck, M. Schoenauer, H.-P. Schwefel (Eds.), Parallel Problem Solving From Nature V, Vol. 1498 of Lecture Notes in Computer Science, Springer-Verlag, Berlin, 1998, pp. 305–314.

[20] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, Berlin, 1992.

[21] T. Starkweather, S. McDaniel, K. Mathias, D. Whitley, C. Whitley, A comparison of genetic sequencing operators, in: R. Belew, L. Booker (Eds.), Proceedings of the 4$^{\text{th}}$ International Conference on Genetic Algorithms, Morgan Kauffman, San Mateo CA, 1991, pp. 69–76.

[22] I. Oliver, D. Smith, J. Holland, A study of permutation crossover operators on the traveling salesman problem, in: J. Grefenstette (Ed.), Proceedings of the 2$^{\text{nd}}$ International Conference on Genetic Algorithms, Lawrence Erlbaum Associates, Hillsdale NJ, 1987, pp. 224–230.

[23] C. Cotta, J. Troya, On the influence of the representation granularity in heuristic forma recombination, in: J. Carroll, E. Damiani, H. Haddad, D. Oppenheim (Eds.), ACM Symposium on Applied Computing 2000, ACM Press, 2000, pp. 433–439.

[24] M. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman and Co., San Francisco CA, 1979.

[25] C. Cotta, J. Aldana, A. Nebro, J. Troya, Hybridizing genetic algorithms with branch and bound techniques for the resolution of the TSP, in: D. Pearson, N. Steele, R. Albrecht (Eds.), Artificial Neural Nets and Genetic Algorithms 2, Springer-Verlag, Wien New York, 1995, pp. 277–280.

[26] G. Syswerda, Uniform crossover in genetic algorithms, in: J. Schaffer (Ed.), Proceedings of the 3$^{\text{rd}}$ International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989, pp. 2–9.

[27] C. Cotta, J. Troya, Tackling epistatic problems using dynastically optimal recombination, in: B. Reusch (Ed.), Computational Intelligence. Theory and Applications, Vol. 1625 of Lecture Notes in Computer Science, Springer-Verlag, Berlin Heidelberg, 1999, pp. 197–205.

[28] E. Lawler, D. Wood, Branch and bounds methods: A survey, Operations Research 4 (4) (1966) 669–719.

[29] T. Bäck, Selective pressure in evolutionary algorithms: A characterization of selection mechanisms, in: Proceedings of the 1st IEEE Conference on Evolutionary Computation, IEEE Press, Piscataway NJ, 1994, pp. 57–62.

[30] N. Radcliffe, Forma analysis and random respectful recombination, in: R. Belew, L. Booker (Eds.), Proceedings of the 4th International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1991, pp. 222–229.

[31] C. Cotta, J. Troya, Using a hybrid evolutionary-A* approach for learning reactive behaviors, in: S. Cagnoni, et al. (Eds.), Real-World Applications of Evolutionary Computation, Vol. 1803 of Lecture Notes in Computer Science, Springer-Verlag, Edinburgh, 2000, pp. 347–356.

[32] T. Bäck, Evolutionary Algorithms in Theory and Practice, Oxford University Press, New York NY, 1996.

[33] D. Goldberg, R. J. Lingle, Alleles, loci, and the travelling salesman problem, in: J. Grefenstette (Ed.), Proceedings of an International Conference on Genetic Algorithms, Lawrence Erlbaum Associates, Hillsdale NJ, 1985.

[34] L. Davis, Applying adaptive algorithms to epistatic domains, in: Proceedings of the 9th International Joint Conference on Artificial Intelligence, Morgan Kaufmann, Los Angeles CA, 1985, pp. 162–164.

[35] G. Syswerda, Schedule optimization using genetic algorithms, in: L. Davis (Ed.), Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York NY, 1991, pp. 335–349.

[36] P. Moscato, C. Cotta, A gentle introduction to memetic algorithms, in: F. Glover, G. Kochenberger (Eds.), Handbook of Metaheuristics, Kluwer Academic Publishers, Boston MA, 2003, pp. 105–144.

Acknowledgements

This work is partially supported by the Spanish *Comisión Interministerial de Ciencia y Tecnología* (CICYT) under grant TIC99-0754-C03-03.

Notes

[1]For simplicity, we use the same symbol to denote a equivalence class and for labeling it.

[2]This definition is formulated in a slightly different way than in [8] since we have defined recombination operators as probability distributions instead of as transformation functions.

[3]As it can be seen, the term *memetic* is used here in the classical sense of combining an evolutionary algorithm with a local-search technique; nevertheless, it must be noted that the term has a broader meaning, and can be safely used nowadays as a synonym of 'hybrid' [36]

[4]Times have been measured on a SUN Ultra1 workstation under Solaris 2.6.1.

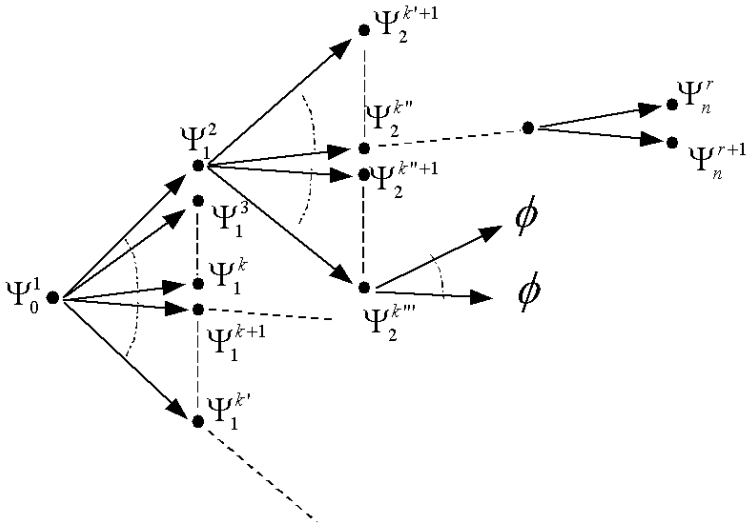

Figure 1: General view of the incremental construction of a solution. At each step, the partial solution (initially empty) is augmented with information taken from the parent. Thus, decisions are taken, regarding the subset of the properties of the resulting solution to augment, and the parent from which the information is taken. This results in an implicitly defined decision tree as shown.

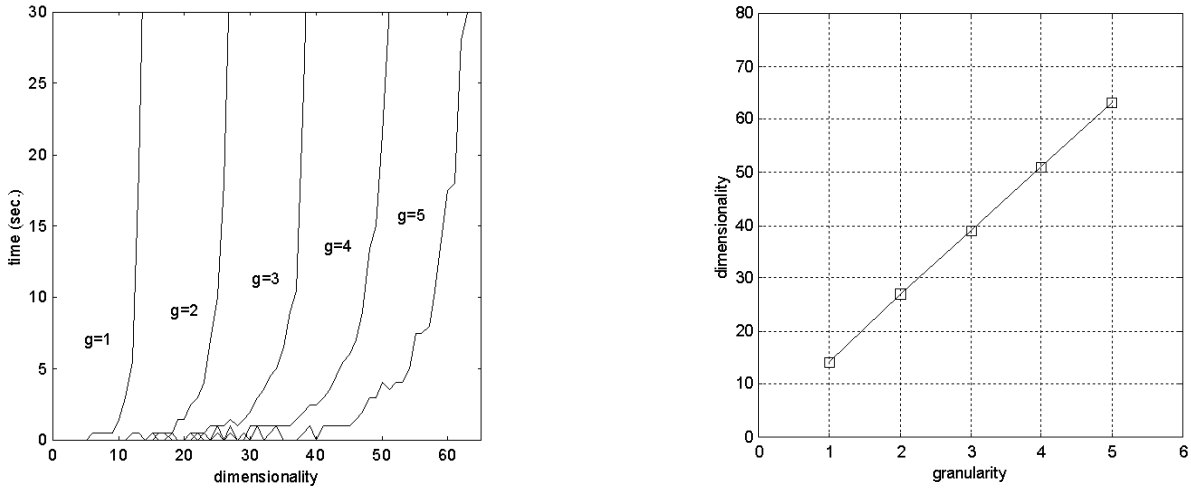

Figure 2: (Left) Times required for performing 100 recombinations of random individuals in the brachystochrone design problem. The results are shown for different granularities. (Right) Lowest dimensionalities for which the time required for performing 100 recombinations of random individuals is greater than 30 seconds.

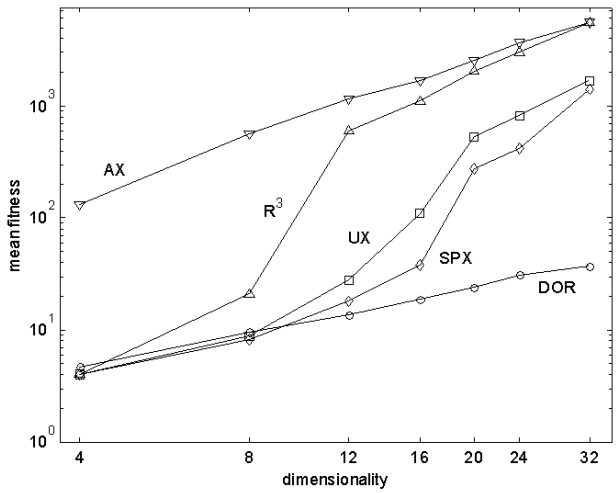

Figure 3: Scalability of different recombination operators for the Schwefel function (results averaged for 20 runs). Notice the use of logarithmic scales. The results for DPX (not shown to make the figure more readable) are virtually identical to those for SPX.

Table 1: Results of different genetic operators for the reactive-rulebase learning problem on nine different environments. The '%R' column represents the percentage of runs (out of 40) in which a fully satisfactory solution was found for the training set. The '%T' column represents the percentage of the test set solved by the best solution found.

	SPX			DPX			UX			DOR		
World	mean	%R	%T	mean	%R	%T	mean	%R	%T	mean	%R	%T
W10a	17.36	95%	74%	15.09	95%	76%	15.88	97%	65%	13.85	100%	81%
W10b	22.15	92%	78%	19.16	95%	78%	18.10	92%	80%	14.35	100%	90%
W10c	51.49	40%	46%	51.99	38%	48%	53.85	40%	42%	18.75	85%	76%
W25a	176.70	65%	58%	101.12	85%	68%	121.63	82%	66%	33.17	100%	83%
W25b	376.08	32%	31%	448.67	23%	24%	380.96	27%	27%	36.65	100%	74%
W25c	520.32	5%	13%	480.71	13%	18%	502.23	10%	16%	99.92	72%	55%
W50a	442.39	30%	34%	321.01	50%	44%	423.87	32%	35%	61.17	100%	69%
W50b	1532.52	5%	5%	1466.64	5%	10%	457.65	0%	2%	337.32	67%	49%
W50c	2021.45	0%	0%	2031.20	0%	0%	2031.20	0%	0%	1638.78	0%	3%

Table 2: Mean best fitness for the Brachystochrone design problem (averaged for 20 runs). N/A entries correspond to prohibitive granularity/dimensionality combinations.

	Number of Pillars								
Operator	8	12	16	24	32	40	48	56	64
SPX	1.1577	1.1788	1.2156	1.2713	1.3908	1.6262	1.7787	2.1049	2.5971
DPX	1.1582	1.1792	1.2100	1.3036	1.4199	1.5810	1.7871	2.1036	2.5370
UX	1.1757	1.2300	1.2862	1.3499	1.5305	1.6771	1.9379	2.2172	2.6658
AX	1.1560	1.1668	1.1828	1.2219	1.2722	1.3534	1.4841	1.6327	1.8845
R^3	1.1540	1.1627	1.1783	1.2149	1.2803	1.3683	1.5295	1.7357	1.9061
$DOR_{g=1}$	1.1530	1.1549	1.1779	N/A	N/A	N/A	N/A	N/A	N/A
$DOR_{g=2}$	1.1544	1.1628	1.1726	1.2110	N/A	N/A	N/A	N/A	N/A
$DOR_{g=3}$	1.1548	1.1798	1.1815	1.2458	1.2660	1.3827	N/A	N/A	N/A
$DOR_{g=4}$	1.1574	1.1901	1.1944	1.2485	1.2971	1.3413	1.4759	2.1635	N/A
$DOR_{g=5}$	1.1601	1.1919	1.2111	1.2494	1.2977	1.3468	1.3886	1.5386	2.1009
$DOR_{g=6}$	1.1649	1.2010	1.2053	1.2598	1.3001	1.3483	1.3695	1.4720	1.6750
$DOR_{g=7}$	1.1729	1.1998	1.2112	1.2635	1.2981	1.3646	1.4073	1.4821	1.5894

Table 3: Mean best fitness for the Brachystochrone design problem using an evolution strategy (averaged for 20 runs) .

	Number of Pillars								
Operator	8	12	16	24	32	40	48	56	64
mutation only	1.1515	1.1484	1.1471	1.1545	1.1823	1.2257	1.3064	1.4257	1.5526
AX	1.1514	1.1478	1.1463	1.1459	1.1595	1.1868	1.2018	1.2326	1.2787
R^3	1.1516	1.1481	1.1466	1.1476	1.1645	1.1907	1.2216	1.2742	1.3034
DOR	1.1511	1.1475	1.1457	1.1448	1.1454	1.1529	1.1744	1.1999	1.2282

Table 4: Results of classical operators on 1−EMP problems (averaged for 20 runs).

Number	Operator							
of elements	OX#1	OX#2	OX#3	PMX	RCX	UCX	BX	UBX
100	2534	2809	2375	795	911	647	963	945
125	4412	4667	4046	1373	1598	1066	1715	1662
150	6803	7571	6502	2114	2537	1660	2637	2665
200	13607	14809	12845	4317	5244	3468	5364	5423

Table 5: Results of the DOR operator on 1-EMP problems (averaged for 20 runs).

Number of elements	Number of construction units allowed (f)									
	blocks	2	3	4	5	6	7	8	9	10
100	572	773	346	319	316	315	302	312	297	319
125	1012	1318	533	482	478	454	481	494	466	463
150	1595	2151	773	691	668	670	657	675	648	679
200	3245	4293	1370	1193	1135	1116	1140	1142	1126	1175

Table 6: Results of the DOR operator on $k-$EMP problems (averaged for 20 runs).

	$k=2$				$k=5$				$k=10$			
f	100	125	150	200	100	125	150	200	100	125	150	200
2	1114	1959	3099	6358	3193	5693	9007	18747	14847	27396	44099	94567
3	481	750	1075	2106	1295	2081	3261	6184	6397	10114	16183	33022
4	407	634	912	1614	994	1655	2452	4706	4878	8046	12622	23733
5	410	616	939	1537	1065	1625	2278	4195	4292	7721	11771	22875
6	406	633	898	1534	1023	1591	2415	4112	4754	8140	11717	21986
7	427	633	915	1566	1239	1710	2387	4395	4887	8381	12053	22266
8	444	653	961	1577	1074	1729	2557	4736	4667	8469	12339	23024
9	432	644	930	1622	1081	1747	2483	4819	4932	8632	13565	23958
10	416	634	955	1644	1181	1866	2661	4858	5252	9073	14210	25823