Structure and Metaheuristics

Yossi Borenstein
University of Essex
Colchester, U.K
yboren@essex.ac.uk

Riccardo Poli
University of Essex
Colchester, U.K
rpoli@essex.ac.uk

ABSTRACT

Metaheuristics have often been shown to be effective for difficult combinatorial optimization problems. The reason for that, however, remains unclear. A framework for a theory of metaheuristics crucially depends on a formal *representative* model of such algorithms. This paper unifies/reconciles in a single framework the model of a black box algorithm coming from the no-free-lunch research (e.g. Wolpert et al. [25], Wegener [23]) with the study of *fitness landscape*. Both are important to the understanding of meta-heuristics, but they have so far been studied separately. The new model is a natural environment to study meta-heuristics.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: ANALYSIS OF ALGORITHMS AND PROBLEM COMPLEXITY

General Terms

Algorithms,Theory

Keywords

Theory, Representation, No Free Lunch, Heuristics

1. INTRODUCTION

During the last 20 years many algorithms (metaheuristics) have been proposed to efficiently explore search spaces on which no knowledge is available [2]. Usually a search algorithm tries to infer the position of good new solutions in the search space based on previously sampled solutions.

Many metaheuristics are inspired by powerful natural or physical processes. Ant Colony Optimization (ACO), Evolutionary Algorithms (EAs) and Simulated Annealing (SA) are examples of such algorithms. ACO and EAs are inspired by nature; SA is inspired by the annealing process of metals.

These and other metaheuristics have been applied successfully to an ever increasing number of hard combinatorial optimization problems such as TSP, vehicle routing, job shop scheduling, and bin packing. However, in many cases, their remarkable empirical success is not associated to correspondingly robust theoretical foundations.

It is of a particular interest to study the different search algorithms in one, single framework. The motivations for this are:

- Even though different metaheuristics are inspired by different processes they are all designed to perform well in the *black box scenario*, and, indeed, results obtained for the black box scenario have implications for all metaheuristics at the same time.
- Despite the apparent differences, metaheuristics have a lot in common. This has led to the development of hybrid metaheuristics which are often proven to be more powerful than the traditional ones. Furthermore, there are general common properties which are expected to affect performance of all metaheuristics as identified in [2].

Wolpert and Macready were perhaps the first to consider the general properties of the black box scenario. With their No-free-lunch theorem (NFLT) [25] they put an end to the hope of developing a general-purpose robust optimization algorithm by proving that such an algorithm does not exist.

Despite the vast impact of the NFLT, its implication on real-world problems and real-world algorithms is not clear. In particular, it is argued that the set of problems that the NFLTs consider is not related to real-world problems. Culberson [5] showed that the number of problems considered by the NFLTs is much higher than the one considered by the class of NP or even PSPACE problems. English [9, 8] made similar arguments using the notion of Kolmogorov complexity [10]. Droste, Jansen and Wegener [6] described the NFLTs scenario as non-realistic and, finally, Igel and Toussaint [11] made similar arguments for the sharper version of the NFL [21].

It seems clear that, in order to obtain meaningful results, it is essential to consider more *realistic scenarios*. Wegener et al. [23, 24, 7] describe black-box algorithms as randomized decision trees and use Yao's minimax principle to derive lower bounds for the black-box complexity of particular, more realistic, classes of problems (e.g., NIAH and unimodal functions).

It is well known that search spaces which corresponds to real world problems include many symmetries (e.g., automorphisms of a TSP graph). Rowe, Vose and Wright [19, 20] investigate this scenario for GA. They formally define a notion of structure in the search space, and study the condi-

GECCO'06, July 8–12, 2006, Seattle, Washington, USA.

tions for which mutation and crossover respect the symmetries induced by the structure. In section 3 we consider the symmetries of the search space in a similar way, albeit, using different notation. In section 4 we generalize this notion (i.e., symmetries in the search space) for the case of fitness landscapes. We define the orbit of a function f w.r.t. the symmetries of the search space as a *structural class of problems*. We argue that this is a natural way to group problems in the black box scenario.

The efficiency of metaheuristics strongly depends on the choice of representation and operators. Respecting the symmetries of the search space is an important property of an efficient search-operators – in fact, designing such operators is, perhaps, the main motivation for [19, 20, 13]. In section 5 we define a generic model of random search heuristics which explicitly assumes that the search operators respect the structure of the search space. The motivations for this are twofold, firstly, this is true for many metaheuristics (in particular, whenever the search operators are defined over the structure, they respect the structure). Moreover, even if for some metaheuristics (or representations) this is not the case at the present, arguably, given the active, fruitful research on principled design, this will be the case in the near future. Our model is constructed such that it has identical performance over all functions which belong to the same structural class.

The implications of this are twofold. Firstly, thanks to our way of grouping problems for the black box scenario on the basis of structure, it is now possible to explore how classes of real-world problems connect to the black box scenario. We follow this approach in [3]. Secondly, a unified realistic model for metaheuristics can be an important starting point for the development of a meaningful theoretical framework. We discuss some of the possible implications of our model for future research in section 6. It is important to stress that by "realistic" we mean that we aim to describe in detail the common properties of existing metaheuristics. We do not consider, at this stage, other important aspects of realistic search algorithm like space restrictions.

2. MOTIVATIONS

There is a gap between the theoretical study of metaheuristics and the way that they are used in practice. On one hand, metaheuristics strongly depend on the structure of the search space, yet, formal models of black-box heuristics make not attempt to model and use this structure explicitly, hiding it, instead, within some probability distribution.

On the other hand, the extensive research done in the field of fitness landscapes (e.g., [16]) focuses on the structure alone, ignoring that it is meaningful to study the features of a landscape only as long as the search operators are "matched" to the landscape structure.

In this paper, we combine the notions of formal model of black-box heuristics and fitness landscapes into one integrated framework. This provides a natural setting where to define and study meta-heuristics. As we argue in section 5, many pre-existing metaheuristics fit our framework. Furthermore, we give some formal results that although easy and intuitive are very powerful and general and can be extended to much less obvious results. The contribution of this paper is, therefore, a *change in perspective* on how to formally model meta-heuristics.

3. STRUCTURE

It is a common conjecture (that has been elaborated in the literature on NFLTs) that real-world problems have structure. Metaheuristics exploit this structure to find solutions efficiently (that is, without using problem-specific knowledge). The notion of structural search space was formally defined in [19, 20]. Row, Vose and Wright consider the situation where there is a group of permutations which act on the search space. They define, this way, the many symmetries often arises in combinatorial search spaces and study the conditions for which mutation and crossover respect these symmetries. In this section we follow a similar approach, however, rather than considering the automorphisms of a graphs we use the notion of metric-spaces and isometry transformation.

The neighborhood structure is often defined by a metric function $d : X \times X \to \mathbb{R}$ that gives the distance between any two points in the search space X. The tuple (X, d) is also called a *metric space*. Consider all the transformations $\sigma : X \to X$. Some of them (the trivial one being $\sigma(x) = x$) preserve the distance relation between any two points in the search space. That is, the distance between the points x, y for example equals the distance between $\sigma(x), \sigma(y)$. More formally:

Definition 1. Let (X, d) be a metric space. The transformation $\sigma^d : X \to X$ is a distance preserving transformation (or an isometry) if:

$$\forall x, y \in X \;\; d(x, y) = d(\sigma^d(x), \sigma^d(y))$$

The *isometry group* is the set of all isometries under function composition.

We give two examples of distance preserving transformations. In the first, we consider the Euclidean N-space in which X is infinite. In the second, we consider the Hamming distance defined over a binary string of size n. Note that the transformation on a finite space can be seen as a permutation of X.

Consider the Euclidean N-space, which is real N-space $\mathbb{R}^N$ with the Euclidian distance metric, in which the distance is:

$$d(x, y) = ||x - y||, x, y \in \mathbb{R}^N.$$

The *translation* $\sigma_a(x) = x + a$ where $a \in \mathbb{R}^N$, is an isometry over the metric space $(d, \mathbb{R}^N)$. Naturally:

$$\begin{aligned} \forall x, y \;\; d(x, y) &= ||x - y|| = ||x + a - y - a|| \\ &= ||(x + a) - (y + a)|| = d(\sigma_a(x), \sigma_a(y)) \end{aligned}$$

Any continuous search space, when represented on digital computer, is essentially finite. Since we are interested to investigate search algorithms which run on a digital computer we can assume that the search space X, although quite large, as well as the space of all possible fitness values Y, is finite. For this reason we restrict our attention to *finite* search spaces. Since for a finite search space, translation is essentially a permutation (e.g., a rotation), we will use the term *distance preserving permutation*.

The Hamming distance is defined for the bit-string representation in the following way: $d_H(x, y) = \sum_i \delta(x_i \neq y_i)$ where $\delta(x) = 1$ if x is true, 0 otherwise. It is easy to show that the exclusive-or (XOR) operator is a distance preserving permutation over the metric space defined by (d_H, X).

LEMMA 1. *Let $x_0 \in X$. The permutation $\sigma^{x_0}(x) \equiv x \oplus x_0$ is a distance preserving permutation.*

PROOF. We need to show that:

$$\forall x, y \in X \; d_H(x, y) = d_H(\sigma^{x_0}(x), \sigma^{x_0}(y)).$$

$$\begin{aligned}
\forall x, y \; d_H(x, y) &= \sum_{i=1}^{n} \delta(x_i \neq y_i) \\
&= \sum_{i=1}^{n} (x_i \oplus y_i) \\
&= \sum_{i=1}^{n} (x_i \oplus (x_0 \oplus x_0) \oplus y_i) \\
&= \sum_{i=1}^{n} (x_i \oplus x_0) \oplus (x_0 \oplus y_i)) \\
&= \sum_{i=1}^{n} (\sigma^{x_0}(x_i)) \oplus (\sigma^{x_0}(y_i)) \\
&= d_H(\sigma^{x_0}(x), \sigma^{x_0}(y))
\end{aligned}$$

□

XOR is not the only way to define isometry in the binary search space. A second way to do so is to permute some of the bit positions in the string – since the Hamming distance is computed per bit, this transformation is isometric as well. Incidently, any isometry of a fixed length binary string, can be described as a composition of a permutation and a (XOR) mask (See [18] for more details). Naturally, structural symmetries exist in other combinatorial spaces as well. [19, 20] discuss this extensively and give several examples.

4. STRUCTURE AND FITNESS LANDSCAPE

While isometry is defined over a metric space $\{X, d\}$, a problem in the black box scenario is defined by the triple $\{X, d, f\}$, where $f : X \to Y$ is the fitness function. The triple $\{X, d, f\}$ is also known as a fitness landscape [17]. Many attempts have been made to characterize the properties of landscapes to discriminate difficult ones w.r.t. easy ones. Well-known examples are isolation, multimodality, auto-correlation and fitness distance correlation. These methods try to characterize, one way or another, the connection (correlation) between the fitness function and the neighborhood structure. Isolation is a case in which the neighborhood of the optimum is characterized by low fitness values. Multimodality measures the number of local optima. Auto-correlation and fitness distance correlation measure explicitly the correlation between the fitness and distance functions.

Naturally, the structure of a landscape does not depend on a fitness of a particular solution but rather on the relation between the different solutions. A unimodal landscape is unimodal regardless of the identity of the global optimum. Consider for example the *onemax* problem: $f(x) = \sum_{i=1}^{n} x_i$. The global optimum for this function is the string of all ones. However, naturally, there are $|X|$ different problems which have exactly the same structural characteristics, namely: $f_{x_{trgt}}(x) = \sum_{i=1}^{n} \delta(x_i = x_{trgt_i})$, where $x_{trgt} \in X$ specifies the identity of the global optimum. It is possible to generalize the *onemax* problem because it is easy to define it in terms of the relation between the fitness function and the neighborhood structure. In particular, the fitness, $\sum_{i=1}^{n} \delta(x_i = x_{trgt_i})$ of x can be written as $n - d_H(x_{trgt}, x)$, that is, in terms of the distance function. But is it possible to generalize other problems where this relation is more complicated?

Based on a distance preserving permutation of a metric space, we can now define a distance preserving permutation of a function. We argue that such a permutation preserves all the structural properties of the fitness landscape.

Definition 2. Let (X, d) be a metric space, σ^d a distance preserving permutation and Σ the isometry group. For any $f : X \to Y$:

1. The permutation $\sigma^d f$ of f is the function $\sigma^d f : X \to Y$ defined by $\sigma^d f(x) = f(\sigma^{d^{-1}}(x))$.

2. The set $F = \{g | \exists \sigma^d \in \Sigma, g = \sigma^d f\}$ is the *structural class* (or the orbit) of f.

A distance preserving permutation of a function preserves the relations between the fitness values and the neighborhood structure. It simply make explicit the many symmetries which exist in the space of all possible problems. We believe that this is a good way to group problems in the black-box scenario. While, by definition, this way preserves the structure of the landscape, in the following we show explicitly that the fitness distance correlation (FDC), for example, as defined by [12], is the same for a function and any distance preserving permutation of the function.

LEMMA 2. *Let f be a function, $\sigma^d f$ a distance preserving permutation of the function and*

$$r(f) = \frac{\frac{1}{n}\sum_i (f(x_i) - \overline{f})(d(x_i, x_{trgt}) - \overline{d})}{\sum_i (f(x_i) - \overline{f}) \sum_i (d(x_i, x_{trgt}) - \overline{d})}$$

the fitness distance correlation of f. Then:

$$r(f) = r(\sigma^d f)$$

PROOF. We consider the FDC for one global optimum measured on the entire search space. The permutation of a function does not change the values of the fitness function and the two functions are defined over the same search space. Also, $\overline{f} = \overline{\sigma^d f}$. So, the denominator in the FDC formula is unaffected by the permutation. Let x_{trgt1} be the optimum for f and $x_{trgt2} = \sigma^d(x_{trgt1})$ the optimum for $\sigma^d f$. We need to show:

$$\frac{1}{n}\sum_i (f(x_i) - \overline{f})(d(x_i, x_{trgt1}) - \overline{d}) =$$
$$\frac{1}{n}\sum_i (\sigma^d f(x_i) - \overline{f})(d(x_i, x_{trgt2}) - \overline{d})$$

$r(\sigma^d f) = \frac{1}{n}\sum_i (\sigma^d f(x_i) - \overline{\sigma^d f})(d(x_i, x_{trgt2}) - \overline{d}) =$

(reordering the summation)

$\frac{1}{n}\sum_i (\sigma^d f(\sigma^d(x_i)) - \overline{\sigma^d f})(d(\sigma^d(x_i), x_{trgt2}) - \overline{d}) =$

(by definition: $\sigma^{d^{-1}} f(x) = f(\sigma^d(x))$)

$\frac{1}{n}\sum_i (f(x_i) - \overline{\sigma^d f})(d(\sigma^d(x_i), x_{trgt2}) - \overline{d}) =$

by definition $x_{trgt2} = \sigma^d(x_{trgt1})$ and,

$d(x_i, x_{trgt1}) = d(\sigma^d(x_i), \sigma^d(x_{trgt1}))$ hence:

$\frac{1}{n}\sum_i f(x_i) - \overline{\sigma^d f})(d(\sigma^d(x_i), x_{trgt2}) - \overline{d}) =$

$\frac{1}{n}\sum_i f(x_i) - \overline{f})(d(x_i, x_{trgt1}) - \overline{d}) = r(f)$

□

To conclude, let us go back to the example of *onemax*. It is easy to map the original problem, using the *xor* permutation, to all the other functions that have similar structure. Thus, the structural class F, when f is *onemax* (i.e. the structural class of *onemax*), gives the same family of functions as we previously obtained by specifying explicitly the function $f(x) = n - d(x_{trgt}, x)$. Similarly, regardless of the complexity of the fitness function – or even without knowing the exact formulation of the fitness function – we can rigorously define a class of problems which have exactly the same structural properties. In this next section we show that this implies identical performance of any metaheuristic.

5. SEARCH ALGORITHMS AND STRUCTURE

Any function f implemented in a digital computer can be considered as a mapping between two finite sets [25]. That is $f : X \to Y$, where X and Y are finite. In the most general case, a randomized search heuristic can be represented as a mapping from a multi-set of previously visited points to a new (not necessarily unvisited) point in X. Droste, Jansen and Wegener [7], [23] [24] suggested a formal definition (table 1), similar to [25, 22], of a black-box algorithm.

This definition generalizes most, if not all, existing randomized search heuristics. The number of queries (i.e., fitness evaluations) made until a sufficiently good f_{opt} is found, is usually used in order to evaluate the performance of the algorithm (see [25], [7] for further discussion). As mentioned in the introduction, this definition let to very useful NFLTs and, thanks to Yao's minimax principle, meaningful bounds for different classes of problems.

However, the generality of this model restricts the possible contribution of any theoretical result: the NFLTs are critiqued for not applying to the real-world scenario. The bounds given by Droste et al. focus on the distribution of instances of different problems – they do not take into account the *a priori* biases of typical heuristics designed for the black box scenario.

To account for this, we will modify this definition and will argue that our modified version is still a proper generalization of many metaheuristics. Using this prototype, we then prove that the expected performance of such algorithms on any function taken from an isometry group is the same.

Table 1: Black box algorithm

1. Choose some probability distribution p on S and produce a random search point $x_1 \in S$ according to p. Compute $f(x_1)$.
2. In step t, stop if the considered stopping criteria is fulfilled. Otherwise, depending on the properties of $I(t) = (x_1, f(x_1), ..., x_{t-1}, f(x_{t-1}))$ choose some probability distribution $p_{I(t)}$ on S and produce a random search point $x_t \in S$ according to $p_{I(t)}$. Compute $f(x_t)$.

Informally, similarly to Droste et al. [6], we assume that any *reasonable* search algorithm has no a priori preference for specific search regions and it selects points according to their fitness value. However, unlike previous approaches we also explicitly assume that the algorithm decides which new points to sample on the basis of their distance from existing points.

Before introducing the formal model, we begin by presenting some terminology:

- Let X, Y be finite. A function is represented by $f : X \to Y$.
- The multi-set $I(t) = \{(x_i, f(x_i))\}$ represents the points that the algorithm has sampled (i.e., the *information* that the algorithm has) by time t.
- We define the function $S(x, I(t)) = \{(d(x, x_i), f(x_i))\}$. This function, given x and the multiset $I(t)$ returns a multiset with exactly the same number of elements as $I(t)$ (and with same cardinality), but where each point previously sampled by the algorithm, x_i, is replaced by the distance $d(x, x_i)$ between x and x_i. In other words, $S(x, I(t))$ describes the *structural relationship* between x and the sets $I(t)$.

Informally, the search algorithm has three phases: a selection phase (select "promising" points from $I(t)$), an exploration phase (based on the selected points - generate new points) and an updating phase (deciding which points to keep in $I(t)$). Table 2 gives a formal definition of a structural black-box algorithm - note that for the sake of simplicity - we denote $I(t)$ as I.

This definition differs, compared to [24, 23, 7], in one important aspect: the probability distribution used to generate new solutions p_t is a function of time and, more importantly, of $S(x, I(t))$ rather than $I(t)$. As a result it depends *solely* on the fitness values and structural/neighbourhood relationship between new potential samples and the points already sampled. Naturally, the question now is: is this definition still general enough to account for existing metaheuristics?

Blum and Roli [2] give a survey on the most important modern metaheuristics. They distinguish between population-based metaheuristics and metaheuristics that use trajectory methods. We choose a representative metaheuristic of each class (genetic algorithm and simulated annealing respectively) and show that our definition is an appropriate generalization of it. As previously mentioned, our

Table 2: Structural black box algorithm

1. Initialize $I(0)$ by choosing m random search points $x_1, \cdots, x_m \in X$ using some prior probability distribution p_0 and then computing the corresponding fitness values $f(x_i)$.
2. In step t, stop if the stopping criteria are fulfilled. Otherwise:
   (a) Choose some probability distribution $p_t = p(S(x, I(t)), t)$ on X.
   (b) Produce n random search points $x_1, \cdots, x_n \in X$ by sampling the probability distribution p_t.
   (c) Compute the corresponding fitness values $f(x_i)$.
   (d) Set $I(t+1) = \text{merge}(I(t), \{(x_i, f(x_i))\}, t)$.

Table 3: Simulated Annealing (SA)

```
s ← GenerateInitialSolution()
T ← T_0
while termination conditions not met do
    s' ← PickAtRandom(N(s))
    if (f(s') < f(s)) then
        s ← s'
    else
        Accept s' as new solution with probability p(T, s', s)
    endif
    Update(T)
endwhile
```

model is only appropriate for metaheuristics with search operators that respect the structure of the search space. As previously mentioned, given the active research done in principled design, we believe that operators which do not respect the structure (and hence, are not captured by our model) will be replaced, eventually, with operators which do [19].

Let us start from trajectory methods. These are characterized by a trajectory in the search space. The search process can be seen as the evolution in discrete time of a discrete dynamical system. The algorithm starts from an initial state and describes a trajectory in the state space. A successor solution is either among the neighborhood of the current solution or chosen randomly [2]. This general description makes it clear that at unbiased trajectory methods are specific instances of our structural black box search algorithm. As an example, we will show explicitly that Simulate annealing (SA) fits our definition.

Simulate annealing (table 3) is perhaps one of the most studied metaheuristic. It was one of the first to use an explicit strategy to escape local minima. The fundamental idea is to allow the selection solutions of worse quality than the current solution, with a probability which decreases over time.

Let us see how SA can be described by our algorithm: the size of the multi-set I is 1. The procedure GenerateInitialSolution() is modelled in step 1 of our algorithm. In order to see how the temperature parameter T is effectively modelled, we just need to think of it as a function of time, i.e., $T(t)$. So, although the probability of accepting solutions (in the merge phase of our algorithm) depends on the temperature, since the temperature is a function of time, the acceptance probability is effectively a function of time too. The sampling probability distribution $p(S(x, I(t)), t)$ is zero for all x such that the distance between x and the point currently in $I(t)$ is greater than 1 (i.e., x is not a direct neighbour of the current search point). Otherwise, $p(S(x, I(t)), t)$ is a constant.

Table 4: Evolutionary Algorithm

```
P ← GenerateInitialPopulation()
Evaluate(P)
while termination conditions not met do
    P' ← Select(P)
    P'' ← Recombine(P')
    P''' ← Mutate(P'')
    Evaluate(P''')
    P ← Select(P ∪ P''')
endwhile
```

Population based methods deal in every iteration with a set of solutions rather than a single one. It is more difficult to give a generic description (like the one given for trajectory methods) for this kind of methods. However, for our argument to hold, it is sufficient to demand that all the search operators are defined over the neighborhood structure. In particular, we focus on perhaps the most studied population based method – the evolutionary algorithm (EA). There are several variations of EAs (e.g., Genetic Algorithms, Genetic Programming, Evolutionary Strategies, etc.). The algorithm given in table 4 is a typical one.

In EAs the size of the multi-set I is defined according to the population size. Similarly to SA, GenerateInitialPopulation() is modelled in our first step. The actual implementation of Select(P) differs from one EA to another. The different selection mechanisms include: tournament selection (with various tournament sizes), fitness proportional selection, rank selection and more. Despite the differences, all of them depend solely on the fitness of the solutions f_i, not the corresponding x_i's. The probability distribution $p(S(x, I(t)), t)$ is sufficient, therefore, to describe all of them.

As opposed to SA, in EAs there are two search operators: recombination and mutation. Since the two operators are applied in sequence, it is sufficient to show that each of them can be described by a probability distribution based on $S(x, I(t))$ alone. The mutation operator is defined similarly to the search operator in SA. That is, for each $x \in X$ a mutation of a given value is the probability distribution defined over the neighborhood $B_r(x)$. As we already argued for SA – this depends only on $S(x, I(t))$.

It is more difficult to analyze the recombination operators. They are defined differently for different representations (e.g., binary-string, permutation, syntactic-trees), and even for a given representation, usually, there is more than one way to define recombination (e.g., for binary strings we have one-point, two-points, uniform crossover, etc.).

Moraglio and Poli [13] introduced the notion of *topological crossovers* as a class of representation independent operators

that are well-defined once a notion of distance over the solution set is defined. Simply stated, topological crossovers produce offspring on the line segment between their parents. Moraglio and Poli showed how topological crossover generalizes the notion of crossover for binary strings [13], permutations [14] and syntactic trees [15].

Formally, a line segment between x and y is defined as:

$$[x;y] = \{z | d(x,z) + d(z,y) = d(x,y)\}$$

A topological uniform crossover UX is defined as the probability distribution that a solution z will be the offspring of the parents x and y:

$$p_{UX}(z|x,y) = \frac{\delta(z \in [x;y])}{|[x;y]|}$$

The crossover is defined as a function of the segment $[x;y]$ which in turn, is a function of the distance. Therefore, *by definition*, the sampling probability distribution is a function of distances only. So, again $p(S(x, I(t)), t)$ is sufficient to model any kind of uniform topological crossover. It follows that evolutionary algorithms that use uniform topological crossover are instances of our structural black box algorithm as well.

We showed that our algorithm can describe explicitly SA – a representative trajectory algorithm – and a family of EAs – representative population based algorithms. The ever increasing number of different metaheuristics does not allow us to prove that our model is applicable to all of them. However, we strongly believe that, in a similar way, the algorithm defined in table 2 can account for many other metaheuristics. For example, it must definitely apply to local search. Also, we believe it may apply to tabu search and particle swarm optimisers. But what is the advantage of using this definition? It follows from our model that any algorithm that can be described by our definition has *exactly* the same performance on any function which belongs to the same isometry group.

Our argument is true by the definition of the structural-search-algorithm. However, we will make it even more general:

THEOREM 1. *Let (X, d) be a metric space, σ^d a distance preserving permutation, $f : X \to Y$ a function and A, a structural search algorithm. Let $P(A|f)$ denotes any performance measure defined as a function of I^Y. The following holds:*

$$\forall_{\sigma^d} P(A|f) = P(A|\sigma^d(f))$$

PROOF. A problem in the black box scenario is defined as a function of the triple $\{X, d, f\}$. However, by construction, the multi-set $S(x, I(t))$ depends only on distance and fitness. Since f and $\sigma^d f$ are isometrically isomorphic, the expected performance (i.e., the expected set of f_i's) of a structural search algorithm for both is the same. □

6. DISCUSSION

We aim at giving the most precise definition of a black box algorithm that is general enough to account for as many metaheuristics as possible. Naturally, it is possible to consider alternative models – in particular, some might claim that our model is not general enough. We cannot possibly *prove* that it is (there are simply too many metaheuristics around). However, our model fits the view of many researches who explore the properties of the fitness landscapes – rather than any particular metaheuristics. Furthermore, the only restriction that our model has w.r.t. existing models is the requirement for it to be *unbiased* to any region of the search space. We believe that this restriction, given that a black box algorithm does not have a priori knowledge about the problem, is quite reasonable.

This paper has formalized some intuitive results which many might be aware of. The objective, however, is to provide a new perspective in which search algorithms in the black box scenario can be studied. In this section we summarize two possible directions which we intend to follow in *future research*.

6.1 Classes of problems for black box algorithms

Classic algorithm theory is concerned with algorithms which are (1) designed to solve *particular problems* and (2) are proved to be efficient on these problems. Outside the context of a problem, a problem-specific algorithm has no meaning.

Black box algorithms, on the other hand, are believed to be generic. They are defined out of the context of any particular problem. This is indeed one of the main obstacles a theory for black-box algorithms has to overcome. The basic notion of a problem (or at least, the target problems) is not well defined.

As a consequence, the NFLTs consider for example, what many believe to be *irrelevant* problems. On the other hand, since black-box algorithms are not designed for any specific problem, it is tempting to think about what Droste et al [6] call the *one-shot-scenario*: in real-life, a black box algorithm needs to solve only one instance (or one fitness function) on which, however, it has no knowledge about. From the perspective of classical algorithm theory, however [6], this is absurd – on theory, it is possible to design an algorithm which sample, in the first step, the global optimum.

The main point that we would like to make is that *classes of problems for the black box scenario are well defined*, albeit, not in the usual way: any function f can be associated with a *structural class* of functions on which the algorithm is expected to perform the same. While this might have been known for some time, it had no real impact. For example, having this in mind, the one-shot-scenario, does not exist. Using the *xor* operator as an isometry permutation (assuming a binary-Hamming metric space), the global optimum, for different functions which belong to the same structural class, can be any solution in X. Since the algorithm is expected to have the same performance over the entire class, it has to be more sophisticated than just picking, on the first trial, the global optimum.

This has some implication even when trying to analyze the performance of a black-box algorithm on well-studied problems. It is common in computational complexity to distinguish between a problem (e.g., TSP) and an instance of a problem (e.g., any configuration of cities and distances). The theory is concerned with the expected performance of the algorithm on instances of the problem. Its performance, naturally, on any other problems is of no relevance. While this is certainly the case for problem-specific algorithms it is not the case for black box algorithms.

For each (relevant) instance of a problem, a black box algorithm is designed to perform exactly the same on the *structural group* of this instance. This is the case irrespec-

tively of whether the functions from the structural class are instances of the problem or not. When analyzing the performance of a black box algorithm on a certain problem, one must take into account, not only all the possible instances, but also, the set of all possible structural classes of these instances.

As mentioned before, different approaches to the study of black-box algorithms focus on different aspects. The fitness landscape approach studies the structural properties of a problem, without considering any particular algorithm. On the other hand, formal models tend to hide the structure inside a probability distribution. Our model can bridge between the two approaches. For example, Droste at el.[7] used Yao's minimax principle to prove lower bounds for black box algorithms. The basic idea is to use the expected run time (w.r.t. a probability distribution on problem instances) of a deterministic search algorithm in order to derive lower bounds for randomized algorithms. Using our model, it is possible to study the effect of the notion of *structural classes* (which connects to the study of fitness landscape) on the probability distribution defined over the problem instances. We believe that this can give tighter bounds to the performance of black box algorithms. We plan to investigate this in future research.

As a final remark, for the NFLTs, any algorithm, the so called black-box algorithms included, has to make an a priori assumption about the problem in order to perform better than random search. It is often claimed that black box algorithms make the right assumption about real-world problems. We would like to emphasize, that we do not make any such claim. That is, we do not argue that black box algorithms are particulary good for the real world scenario. Moreover, while we argue that the performance over any structurally identical function is similar – we do not, at this stage, distinguish between easy structure and hard one. We do not know whether the "assumption" black box algorithms make is good or bad – we simply state that they make an assumption, and that they should be studied accordingly[1]. To give additional examples, one can ask how the size of the isometry group changes for different representations. This is equivalent to the number of instances (i.e., isometry permutations of a function) a problem (i.e., a function with unique structural properties) in the black box scenario has. Similarly, one can investigate how this size changes as the size of the search space increases.

6.2 Entropy and Search

The simple intuition which we formalized in this paper suggests that the key strategy of any black box algorithm is (a) focus on solutions with high fitness and (b) when sampling a new point, try to select one according to its distance from such solutions. This strategy will fail in either of the two scenarios – (1) solutions, more often than not, have similar fitness (e.g., the NIAH) and hence most of the choices, as to which solutions to select, are done randomly, and (2) solutions have different fitness values, however, the notion of proximity is not *selective* enough (e.g., in fully connected graph all the solutions in the search space have the same distance from any point. Proximity in that case is not a good method to select which point to sample next).

[1]This is naturally studied in the context of fitness landscapes. However, in this paper we suggest a different approach, outside of that context.

It is possible to characterize, formally, how random the search is expected to be w.r.t. these two scenarios. At this stage we cannot go into full, formal, details (it is still work in progress), however, main intuition is this: Let us restrict our attention to search algorithms that, at each stage, have a population of size k, of which, in the selection phase, s are selected, and accordingly k new solutions (i.e., a new population) are generated.

The entropy, or uncertainty in the way the algorithm searches can be defined, separately for the two stages. In the first stage, given the fitness of the k solutions (i.e., $\{f(x_1), ..., f(x_k)\}$), let the probability distribution P_f^k define the probability of each solution (x_i) to be selected. The entropy of the particular set can be defined as follows: $S_s = \sum_{i=1}^{k} P_f^k(x_i) log(P_f^k(x_i))$. If the solutions are selected uniformly at random, the entropy is maximal and the search will be essentially random. The expectation of this for all possible sets of k fitness values, will give us the entropy of the selection phase (that is, the extent to which solutions are picked randomly – i.e., irrespectively of their fitness value).

Similarly, once s points are selected, one can calculate the probability distribution of picking any other point in the search space. The probability of a point x to be selected depends on its distances from the s selected points, and hence, similarly to the previous step, the entropy for a particular set of s points is defined (assuming that $I(t)$ represents our set of s selected points): $S_e = \sum_{i=1}^{|X|} p_{S(x,I(t))}(x_i) log(p_{S(x,I(t))}(x_i))$. The entropy of the exploration phase is, similarly, the expectation of this for all possible sets of s points (that is, the extent to which the algorithm chooses to sample new solutions in a random way).

It is not trivial to define in a generic way such sets or, moreover, given such sets, to find the corresponding probability distributions $(P_f^k, p_{S(x,I(t))})$. However, once these issues are resolved it will be possible to compare different algorithms accordingly. Moreover, while S_e depends solely on the metric-space (i.e. is the same irrespectively of a particular fitness function), S_s depends also on the fitness-distribution of the function. Considering the NIAH, for example, irrespectively of the selection mechanism (or, selection intensity [1]) the entropy will be maximal (i.e. given a set of solutions with the same fitness – the algorithm will select one of them, uniformly at random). A similar approach, i.e. measuring the information content of a fitness-function was suggested in [4].

7. CONCLUSION

The notion of isometry, isometric isomorphism or congruence mapping was studied as a natural consequence, in mathematics and geometry to the definition of metric spaces. The notions of isometric transformation captures the many symmetry properties of all the ways to combine a metric space with a fitness functions. While the NFLTs consider $|Y|^{|X|}$ possible fitness functions – it is clear that there are much less functions with a unique *structure*. This is the only reasonable way to think of structure.

In this paper we formally defined a notion of a *structural class of problems*. We defined accordingly a model of structural search algorithms. We proved, by construction, that any structural search algorithm is expected to have identical performance on any function which belong to the same isometry group (i.e. the same structural class). We showed

(using simulated annealing and evolutionary algorithms as examples) that our model is likely to represent many existing metaheuristics. Thus, we have unified the nowadays main approaches to the study of meta-heuristics: on the one hand, the model considered in the NFLTs [25] and similarly by Droste, Jansen and Wegener [23, 24, 7] and on the other hand, the one implicitly considered by the study of fitness landscapes [17, 16].

Establishing a none-trivial model of metaheuristics allows an extensive exploration (which is one of our immediate objectives) of the generic behavior of such algorithms. Some of the immediate areas of future research were considered in section 6. The class of structural problems is of particular interest in the context of combinatorial optimization (CO). The representations of many CO problems is relatively well understood. Drawing the line between classes of CO problems and the class of structural problems defined here, may shed important light on the potential of metaheuristics for combinatorial optimization . We make a first step towards this direction in [3].

8. ACKNOWLEDGMENT

This work was supported by the Anglo-Jewish-Association and the Harold Hyam Wingate Foundation. The authors would like to thank Alberto Moraglio and Jonathan Row for useful comments.

9. REFERENCES

[1] T. Blickle and L. Thiele. A comparison of selection schemes used in evolutionary algorithms. *Evolutionary Computation*, 4(4):361–394, 1996.

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

[3] Y. Borenstein and R. Poli. Structure and metaheuristics. GECCO 2006.

[4] Y. Borenstein and R. Poli. Information landscapes. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 1515–1522, New York, NY, USA, 2005. ACM Press.

[5] J. C. Culberson. On the futility of blind search: An algorithmic view of "no free lunch". *Evolutionary Computation*, 6(2):109–127, 1998.

[6] S. Droste, T. Jansen, and I. Wegener. Optimization with randomized search heuristics - the (a)nfl theorem, realistic scenarios, and difficult functions. *Theor. Comput. Sci.*, 287(1):131–144, 2002.

[7] S. Droste, T. Jansen, and I. Wegener. Upper and lower bounds for randomized search heuristics in black-box optimization. *Electronic Colloquium on Computational Complexity (ECCC)*, (048), 2003.

[8] T. M. English. Evaluation of evolutionary and genetic optimizers: No free lunch. In *Evolutionary Programming*, pages 163–169, 1996.

[9] T. M. English. On the structure of sequential search: Beyond "no free lunch". In J. Gottlieb and G. R. Raidl, editors, *EvoCOP 2004, Coimbra, Portugal, April 5-7, 2004, Proceedings*, volume 3004 of *Lecture Notes in Computer Science*, pages 95–103. Springer, 2004.

[10] P. Grunwald and P. Vitanyi. Shannon information and kolmogorov complexity. *IEEE Transactions on Information Theory*, 2004. In Review.

[11] C. Igel and M. Toussaint. A no-free-lunch theorem for non-uniform distributions of target functions. *Journal of Mathematical Modelling and Algorithms*, 3(4):313–322, 2004.

[12] T. Jones and S. Forrest. Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 184–192, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[13] A. Moraglio and R. Poli. Topological interpretation of crossover. In K. Deb et al. editors, *Genetic and Evolutionary Computation - GECCO 2004, Genetic and Evolutionary Computation Conference, Seattle, WA, USA, June 26-30, 2004, Proceedings, Part I*, volume 3102 of *Lecture Notes in Computer Science*, pages 1377–1388. Springer, 2004.

[14] A. Moraglio and R. Poli. Abstract geometric crossover for the permutation representation. *IEEE Trans. Evolutionary Computation*, page submitted, 2005.

[15] A. Moraglio and R. Poli. Geometric landscape of homologous crossover for syntactic trees. In *CEC (1)*, pages 427–434, 2005.

[16] P.F.Stadler and C.R.Stephens. Landscapes and effective fitness. *Comments Theori. Biol.*, 8:389–431, 2002.

[17] C. M. Reidys and P. F. Stadler. Combinatorial landscapes. *SIAM Rev.*, 44(1):3–54, 2002.

[18] J. Rowe, D. Whitley, L. Barbulescu, and J.-P. Watson. Properties of gray and binary representations. *Evol. Comput.*, 12(1):47–76, 2004.

[19] J. E. Rowe, M. D. Vose, and A. H. Wright. Group properties of crossover and mutation. *Evol. Comput.*, 10(2):151–184, 2002.

[20] J. E. Rowe, M. D. Vose, and A. H. Wright. Structural search spaces and genetic operators. *Evol. Comput.*, 12(4):461–493, 2004.

[21] C. Schumacher, M. D. Vose, and L. D. Whitley. The no free lunch and problem description length. In L. Spector et al. editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 565–570, San Francisco, California, USA, 7-11 July 2001. Morgan Kaufmann.

[22] M. D. Vose. *The Simple Genetic Algorithm: Foundations and Theory*. MIT Press, Cambridge, MA, USA, 1998.

[23] I. Wegener. Towards a theory of randomized search heuristics. In B. Rovan and P. Vojtás, editors, *MFCS*, volume 2747 of *Lecture Notes in Computer Science*, pages 125–141. Springer, 2003.

[24] I. Wegener. Randomized search heuristics as an alternative to exact optimization. In W. Lenski, editor, *Logic versus Approximation, Essays Dedicated to Michael M. Richter on the Occasion of his 65th Birthday*, volume 3075 of *Lecture Notes in Computer Science*, pages 138–149. Springer, 2004.

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