SEARCH, BLACKBOX OPTIMIZATION, AND SAMPLE COMPLEXITY

H. Kargupta, D. E. Goldberg

Foundations of Genetic Algorithms (FOGA4)
San Diego, CA
August 1996
DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.
SEARCH, Blackbox Optimization, And Sample Complexity

Hillol Kargupta*
Computational Science Methods division
Los Alamos National Laboratory
Los Alamos, NM, USA.

David E. Goldberg
Department of General Engineering
University of Illinois at Urbana-Champaign
Urbana, IL, USA.

Abstract

The SEARCH (Search Envisioned As Relation & Class Hierarchizing) framework developed elsewhere (Kargupta, 1995; Kargupta & Goldberg, 1995) offered an alternate perspective toward blackbox optimization—optimization in presence of little domain knowledge. The SEARCH framework investigates the conditions essential for transcending the limits of random enumerative search using a framework developed in terms of relations, classes and partial ordering. This paper presents a summary of some of the main results of that work. A closed form bound on the sample complexity in terms of the cardinality of the relation space, class space, desired quality of the solution and the reliability is presented. This also leads to the identification of the class of order-k delineable problems that can be solved in polynomial sample complexity. These results are applicable to any blackbox search algorithms, including evolutionary optimization techniques.

1 Introduction

The SEARCH (Search Envisioned As Relation and Class Hierarchizing) framework introduced elsewhere (Kargupta, 1995) offered an alternate perspective of blackbox search (BBO) in terms of relations, classes and partial ordering. SEARCH is primarily motivated by the observation that searching for optimal solution in a BBO is essentially an inductive process (Michalski, 1983) and in absence of any relation among the members of the search space, induction is no better than enumeration (Watanabe, 1969). SEARCH decomposed BBO into three spaces: (1) relation, (2) class, and (3) sample spaces. SEARCH also identified the importance of searching for appropriate relations in BBO. No BBO algorithm can efficiently solve a reasonably general class of problems unless it searches for relations. Kargupta (1995) also showed that the class of order-k delineable problems can be solved in SEARCH with sample complexity polynomial in problem size, desired quality and reliability of the solution. These results are relevant for any BBO algorithm. Therefore evolutionary optimization algorithms must pay attention to them.

Section 2 presents a brief review of the related works. Section 3 describes the main components of SEARCH. Section 4 discusses the different decision makings in SEARCH. The main analytical results for ordinal decision making are presented in Section 5. Finally, Section 6 concludes this paper.

2 Background

Although optimization has been addressed in both theory and practice for several centuries, the methodology for solving optimization problems have often followed a pattern: Given a very specific class of problems with some known properties, design an algorithm for solving this class. Unfortunately, because of the ever-growing list of different optimization problems, the process of designing new problem-specific algorithms is unlikely to terminate. Designing algorithms for solving blackbox problems—optimization problems with little knowledge available about the problem domain—offers an alternate approach. By assuming little knowledge about the problem, algorithms designed using this approach aspire to solve a more general class of problems.

The purpose of this section is to introduce BBO and to review some earlier works. Section 2.1 introduces BBO and Section 2.2 reviews some existing works on BBO.
2.1 Blackbox optimization

Almost every discipline of engineering and science can make use of optimization algorithms. As a result, a large number of optimization algorithms have been developed and applied to different problems. For example, smooth convex functions can be efficiently optimized using gradient search techniques (Papadimitriou & Steiglitz, 1982) The simplex algorithm (Dantzig, 1963) performs well for a large class of linear programming problems. Dynamic programming techniques (Dreyfus & Law, 1977) work well when the optimization problems are stage decomposable. Several analyses have been done for local and global optimization of real functions that are Lipschitz continuous with a known Lipschitz constant (Törn & Žilinskas, 1989; Vavasis, 1991). This approach of operations research is characterized by a pattern: Given a class of problem, find an algorithm to solve it. Unfortunately, this approach of designing algorithms that work only for a specific class of problems does not ever seem to end, as the list of different types of optimization problems continues to grow. Moreover, determining the class of problems in which a real-world optimization problem belongs is often as difficult as finding a reasonable solution for the problem. The ever-increasing computing capability has also fueled the desire for solving large-scale problems with little prior knowledge about the objective functions.

This growing demand for algorithms to solve new classes of difficult optimization problems and the never-ending process of designing algorithms that work for a restricted class of problems suggest the need for an alternate approach. The applicability of the previously mentioned optimization algorithms is very restricted, because these algorithms make assumptions about the properties of the objective functions that are often too restrictive. Therefore, one step toward designing optimization algorithms that work for a large class of problems is to reduce assumptions about the objective function. Since these algorithms make little assumption about the objective function, they should be able to solve problems using as little domain knowledge as possible. These problems would fall into the general class of blackbox optimization (BBO) problems, where little knowledge about the objective function is assumed. In this model of optimization, the objective function is often available as a black box, i.e., for a given \( x \) in the feasible domain, it returns the function value \( \Phi(x) \). No local or global information about the function is assumed. Let us denote the finite input and the output spaces by \( \mathcal{X} \) and \( \mathcal{Y} \), respectively. The general blackbox optimization problem can be formally defined as follows. Given a blackbox that somehow computes \( \Phi(x) \) for an input \( x, \)

\[
\Phi : \mathcal{X} \rightarrow \mathcal{Y}
\]

The objective of a maximization problem is to find some \( x^* \in \mathcal{X} \) such that \( \Phi(x^*) \geq \Phi(x) \) for all \( x \in \mathcal{X} \). Performance of an optimization algorithm in this model depends on the information collected by sampling different regions of the search space. The following section presents a brief review of some previous studies related to the work presented in this section.

2.2 Brief review of previous works

By definition, a strict blackbox search algorithm must work without any prior information about the structure of the objective function. Although the field of global optimization has a rich volume of literature, many studies are severely restricted because of their assumptions about the properties of the objective function (Schoen, 1991), and therefore it can be questioned whether they can really be called BBO algorithms. The objective of this section is to present a brief account of some previously developed algorithms that make little use of domain information about the problem. First, we present a classification of BBO algorithms based on whether the algorithm is deterministic or non-deterministic. Next, we concentrate on the non-deterministic or stochastic methods. Finally, we present a brief description of some previous efforts to relate different BBO algorithms with one another and to understand them on common grounds.

Although there may be several ways to classify optimization algorithms from different points of view (Törn & Žilinskas, 1989), one natural candidate is classification based on the deterministic or non-deterministic nature of the search algorithm. Several earlier efforts (Archetti & Schoen, 1984; Dixon & Szegö, 1978; Gomulka, 1978) suggested classification of global optimization algorithms using this approach. BBO algorithms can be similarly classified as

- Deterministic approaches
- Stochastic approaches
  - blind random search methods
  - adaptive sampling search methods

Each of these approaches will be briefly described in the following.

Deterministic enumeration of members of the search space is one method. Unfortunately, for most of the interesting optimization problems, deterministic enumeration becomes practically impossible because of the growth in the search space.
On the other hand, the stochastic algorithms introduce some random elements into the algorithm and try to solve the problem by relaxing the guarantee of the deterministic enumerative search. This relaxed nature of stochastic search algorithms makes them more suitable for practical applications.

Blind random search (Schoen, 1991; Törn & Žilinskas, 1989) is probably the simplest class of algorithms within the family of stochastic BBO algorithms. The Monte Carlo and multistart algorithms are examples of this kind of algorithm. The Monte Carlo algorithm generates random samples from the search space according to a fixed distribution. Multistart methods make use of local search techniques in addition to the Monte Carlo sample generation process. Although algorithms of this class are simple in nature, they are likely to be suitable for the worst case when different regions of the search space cannot be qualified and when evaluating a particular member of the search space does not provide information about another member.

Adaptive sampling search techniques try to exploit the information gathered from samples taken from the search space. They try to qualify different regions of the search space in terms of the fitness values of their members and use that information to decide which region to explore next. Bayesian algorithms, clustering methods, simulated annealing (SA) and genetic algorithms (GAs) are examples of this class of algorithms. This dissertation mainly considers this class of algorithms.

Bayesian algorithms (Betrò, 1983) try to develop a statistical model of the objective function. These algorithms do not explicitly construct a function; instead, they use a random variable to minimize the expected deviation of the estimate from from the real global optimum. The expected value of the random variable is set to the best estimate of the function and the variance of the random variable capture the uncertainty about this estimate. The problem of Bayesian algorithms are that they are often complicated and involve fairly cumbersome computations, such as computing the inverse of the covariance matrix (Törn & Žilinskas, 1989).

Clustering methods (Rinnooy Kan & Timmer, 1984; Törn & Žilinskas, 1989) use a Monte Carlo sample generation technique. Cluster analysis algorithms are used to identify local minima. This is followed by a local search for each local optimum. Clustering methods have been found useful for many global optimization problems (Hart, 1994; Törn & Žilinskas, 1989). However they are likely to perform poorly when the objective function is multimodal and there are many local optima (Hart, 1994).

Since the early 80s, the simulated annealing (SA) algorithms (Kirpatrick, Gelatt, & Vecchi, 1983) and their variants have been used for solving blackbox problems. The natural motivation behind SA is the statistical behavior of molecules during the crystallization process in annealing. SA considers one sample at a time and this sample represents the state of the algorithm. A neighborhood generator is used to generate new samples. SA makes use of a probabilistic comparison statistic (the Metropolis criterion) for deciding whether the new sample should be accepted as the state of the algorithm. The Metropolis criterion dynamically changes along with a parameter known as temperature. The temperature takes a high value in the beginning and gradually decreases according to a chosen cooling schedule. The acceptance probability is often very high in the beginning, when the temperature is high. The acceptance probability decreases as the temperature reduces. SA has a proof for asymptotic convergence to the optimal solution (Kirpatrick, Gelatt, & Vecchi, 1983) SA has been applied to a wide range of blackbox problems. Many of them reported very promising results. However, in the recent past several negative results have also come out (Dueck & Scheuer, 1988; Ferreira & Žerovnik, 1993).

Genetic algorithms (GAs) (De Jong, 1975; Goldberg, 1989; Holland, 1975), evolutionary programming (Fogel, Owens, & Walsh, 1966), and evolutionary strategies (Rechenberg, 1973) are also getting increasing attentions for dealing with global optimization in blackbox problems. Design of the simple genetic algorithm (GA) is motivated by natural evolution. Unlike the SA, it emphasizes the role of representation and the interaction between the representation and perturbation operators. GAs use the representation to implicitly divide the search space into several non-overlapping classes often called schemata (Holland, 1975). Unlike SAs, GAs work from a population of samples, with each sample often represented as sequences. This population of sequences is used to evaluate different schemata. New samples are generated by crossover and mutation operators. Crossover also implicitly combines the better schemata while generating new samples. GAs have been successfully applied to different classes of problems (Goldberg, 1989). However, the simple GA suffers from several limitations. Although the simple GA realizes the role of representation that induces relations among members of the search space, the simple GA does not really search for appropriate relations. Moreover, the evaluation of schemata is also very noisy in the simple GA. These issues will be revisited and elaborated in Chapter 4.

With all these different BBO algorithms in our ar-
senal, it is quite natural to ask whether they can be studied on common grounds using common principles. Several previous efforts have been made to address this question. Although Holland's (1975) framework for adaptive search was primarily motivated by evolutionary computation, the underlying concepts of search based on schema processing and decision making are fundamental issues that are equally relevant in the context of any other adaptive BBO algorithms. In fact, Holland's work (1975) is the root of the current thesis. Davis (1987) made an effort to put literature on SAs and GAS under a common title. Unfortunately, this paper did not make any direct effort to link them; rather, it simply discussed them separately. Sirag and Weisser (1987) combined several genetic operators into a unified thermodynamic operator and used it to solve traveling salesman problems. However, this paper did not study the fundamental similarities and differences between SAs and GAS. Goldberg (1990) addressed this issue. He presented a common ground to understand the effects of the different operators of SAs and GAS. He also proposed the Boltzmann tournament selection operator, which attempts to achieve Boltzmann distribution over the population. Mahfoud and Goldberg (1992) introduced a parallel genetic version of simulated annealing called parallel recombinative simulated annealing. This algorithm attempted to harness the strengths of both SAs and GAS. Recently Rudolph (1994) developed a Markov chain formulation of SAs and GAS for analyzing their similarities and differences. Jones and Stuckman (1992) made an interesting effort to relate GAS with Bayesian approaches to global optimization. They noted the similarities and differences between these two approaches and concluded that they share many common grounds. They also developed hybrid algorithms that try to harness the strengths of both approaches. Recently Jones (1995) proposed a framework to study the correspondence between evolutionary algorithms and heuristic state space search of graph theory. In this approach the search domain of the objective function is viewed as a directed, labeled graph. Jones and Forrest (1995) also proposed the fitness-distance-correlation measure for quantifying search difficulty and applied this measure to several classes of objective functions.

Unfortunately, very few of the previous works actually made a quantitative effort to study the computational capabilities and limitations of BBO. Little attention has been paid to the role of relations in BBO, which is essential for transcending the limits of random enumerative search. We still lack any common framework that describes these different algorithms in terms of the basic concepts of theory of computation. The SEARCH framework, makes an attempt to do that. The following section presents a brief description of SEARCH.

3 SEARCH: An Informal Picture

SEARCH presents an alternate picture of blackbox optimization in terms of relations and classes that can be constructed among the members of the search space. SEARCH is also a formal framework that helps us quantify different aspects of BBO, such as sample complexity, problem difficulty, and many more. In this section we briefly review the framework and present the main analytical results without presenting the rigorous derivations given elsewhere (Kar-gupta, 1995). Section 3.1 presents the fundamental motivation behind SEARCH. Section 3.2 presents an overview of SEARCH. A more detailed picture is presented in Section 3.3.

3.1 Motivation

Some existing BBO algorithms try to find the optimal solution by directly searching the original domain of optimization variables. Samples are often used to estimate the best solution of the search space. In these approaches, a BBO algorithm always searches for a better solution compared to the current best solution. It takes one or more samples and then decides how to choose the next sample. Although the task is certainly non-trivial, the approach of finding the best solution by iteratively updating the best estimate has a fundamental problem. Sampling one particular point from the search domain does not necessarily tell us anything about another point. When a BBO algorithm makes a decision to sample another member from the domain, it is performing induction—the process of hypothesizing the premise from the consequences (Michalski, 1983). This is because we are first observing the objective function values for the members of the sample set and then trying to determine whether an unknown point should have a higher or lower objective function value. In other words, it is guessing; it is a proven fact that induction is impossible when no relation exists between the members (Mitchell, 1980; Watanabe, 1989). If no prior relation is assumed between them, there is little reason to choose one member over others, and the blackbox search will be no better than the random search unless the algorithm assumes and exploits some relations among the members of the search domain.

If assuming and exploiting relations among the members of a search space is essential, then it will be wise to isolate this possibility, study it, and see
how it can be used to the fullest. The SEARCH framework does that. Recall that SEARCH stands for Search Envisioned As Relation and Class Hierarchizing. Searching for better relations and better classes are the primary fronts emphasized in SEARCH. Relations classify the search space into different regions. Some relations classify the search space in such a way that it is relatively easier to detect the class containing the optimal solution. SEARCH tries to establish such relations among the members of the search space. Instead of directly searching for the best solution from the beginning, SEARCH tries to find these relations and then use them to locate the classes containing the optimal solution. The following section presents a brief overview of SEARCH.

3.2 Overview

The foundation of SEARCH is laid on a decomposition of the blackbox search problem into relation, class, and sample spaces. A relation is a set of ordered pairs. For example, in a set of cubes, some white and some black, the color of the cubes defines a relation that divide the set of cubes into two subsets—set of white cubes and set of black cubes. Consider a 4-bit binary sequence. There are $2^4$ such binary sequences. This set can be divided into two classes using the equivalence relation $f###$, where $f$ denotes position of equivalence; the # character matches with any binary value. This equivalence relation divides the complete set into two equivalence classes, 1### and 0###. The class 1### contains all the sequences with 1 in the leftmost position and 0### contains those with 0 in that position. In a BBO problem, relations among the search space members are often introduced through different means, such as representation, operators, heuristics, and others. The above example of relations in binary sequence can be viewed as an example of relation in the sequence representation. In a sequence space of length $l$, there are $2^d$ different equivalence relations. The search operators also define a set of relations by introducing a notion of neighborhood. For a given member in the search space, the search operator define a set of members that can be reached by one or several application of the operators. This introduces relations among the members. Heuristics identifies a subset of the search space as more promising than others often based on some domain specific knowledge. Clearly this can be a source of relations. Relations can sometimes be introduced in a more direct manner. For example, Perttunen and Stuckman (1990) proposed a Bayesian optimization algorithm that divides the search space into Delaunay triangles. This classification directly imposes a certain relation among the members of the search space. The same goes for interval optimization (Ratschek & Voller, 1991), where the domain is divided into many intervals and knowledge about the problem is used to compute the likelihood of success in those intervals. As we see, relations are introduced in any search problem either implicitly or explicitly. The role of relations in BBO is also very fundamental and important. Although, in a BBO, many relations can be introduced in different ways not all of the relations are appropriate from the optimization perspective. The objective of sampling based BBO is to detect regions of the domain that are most likely to contain the optimal solutions. In other words, a BBO algorithm tries to detect those classes defined by a relation which appear more promising. If a relation divides the search space in such a way that such detection is easier, then the relation is appropriate for that problem. We shall later formally define such relations as those which properly delineates the domain. Determining which relation is better requires first constructing a partial ordering among the classes defined by each of the relations. In a sampling based BBO all these decision making is done by taking a finite number of samples from the domain. Clearly, all the BBO algorithms often implicitly deal with the three distinct spaces: (1) relation space, (2) class space, and (3) sample space. SEARCH considers all of them explicitly in an effort to understand them rigorously. Figure 1 shows this fundamental decomposition in SEARCH. The major components of SEARCH can be listed as follows:

1. classification of the search space using a relation
2. sampling
3. evaluation, ordering, and selection of better classes
4. evaluation, ordering, and selection of better relations

5. resolution

Each component is discussed in more detail in the following paragraphs. To do so requires some notation that we shall use throughout the remainder of the section. A relation is denoted by \( r_i \), where \( i \) is the index of the set of all relations, \( \Psi_r \), under consideration of the algorithm. Let \( C_i \) be the collection of subsets, created by relation \( r_i \). The set of relations \( S_r \) actually used by an algorithm to solve the given BBO is a subset of \( \Psi_r \). Denote the members of \( C_i \) by \( C_{1,i}, C_{2,i}, \ldots, C_{N_i,i} \), where the cardinality of the class \( C_i \) is \( ||C_i|| = N_i \). Therefore, \( C_i \) is a collection of classes.

Once the relation is used to construct \( C_i \), the next step is to evaluate the classes in \( C_i \). To do that we need samples from the domain of optimization. A perturbation operator \( P \) is defined as an operator that generates new samples. This operator can be either a random sample generator or a smarter one that exploits information from the relation, class, and sample memory.

The next step is to construct an ordering among the classes in \( C_i \). To do so, we need a way to compare any pair of classes. A statistic \( T \) can be computed for each of the classes, and they may be compared on this basis. This statistic will be called a class comparison statistic. This class comparison statistic can be used for computing a tentative ranking among the classes in \( C_i \). For certain choices of \( T \), some classes may not be compared with other classes. This means that sometimes a total order may not be constructed. Therefore, in general, a statistic \( T \) can be used to construct a partial order on \( C_i \). Let us denote this partially ordered collection by \( C_{i[1]} \). Once the ordering is constructed, the next goal is to select some \( 1 \leq M_t \leq ||C_i|| \) top ranked classes from \( C_{i[1]} \). \( M_t \) represents the total number of top ranked classes that will be selected for future considerations. The exact choice of \( M_t \) depends on the decision error probability in choosing an appropriate relation and ordering construction among the classes. For example, if sampling is insufficient, the ordering of classes cannot be relied upon with high confidence, and drastic elimination of classes may not be appropriate. Therefore, a relatively larger value of \( M_t \) may be used. These \( M_t \) classes constitute the updated version of the class search space.

Next, this ordering among the classes is used to evaluate the relation \( r_i \) itself. Different kinds of statistics can be used to compare relations with one another. We denote this relation comparison statistic by \( T_r \) and call it a relation comparison statistic. This statistic for relation \( r_i \) is now computed. The set of all relations currently under consideration is ordered based on this statistic. Note that, again, this ordering does not have to be a total ordering. The top \( M_r \) relations are kept for future consideration and the rest are discarded, in a manner very similar to what we did for the classes.

Not all the classes defined by a relation need to be considered. As more and more relations are evaluated, the information gathered may be used to prune out different classes before evaluating a new relation. Let \( r_1 \) be a relation that is logically equivalent to \( r_1 \land r_2 \), where \( r_1 \) and \( r_2 \) are two different relations; the sign \( \land \) denotes logical AND operation. If either of \( r_1 \) or \( r_2 \) was earlier found to properly delineate the search space with certain value of \( M_t \), then the information about the classes that are found to be bad earlier can be used to eliminate some classes in \( r_2 \) from further consideration. Blackbox algorithms often implement a resolution-like process to take advantage of any such possible decomposability. If the chosen relation \( r_1 \) can be decomposed into a collection of different relations, denoted by \( U_{r_1 r_2} \), then resolution can eliminate some classes using the information collected from possible earlier evaluations of some relations in \( U_{r_1 r_2} \).

Repeated iterations of the above steps result in gradual focusing into those regions of the search space which look better using the chosen class and relation comparison statistics. The set of all these relations \( r_1, r_{i+1}, \ldots \) used to solve the problem is denoted by \( S_r \). Whether or not the algorithm approaches the globally optimal solution, depends on success in finding proper relations, better classes, and sufficient sampling.

The following section presents a formal description of the different aspects of the SEARCH framework.

3.3 SEARCH: The detailed picture

The objective of this section is to present a more quantitative picture of SEARCH and formalize the earlier descriptions. The definition of a better relation requires defining what we mean by better classes. Therefore, the decision making in the class space is considered first, in Section 3.3.1. Section 3.3.2 considers the class selection process. This is followed by Section 3.3.3 that discusses the relation search. Finally, Section 3.3.4 presents the resolution process of SEARCH.

3.3.1 Classification and ordering of classes

This section considers the decision-making process among the classes. Classification of the search space requires defining relations. A relation can be defined using different sources, such as operators and representation. In this section we assume no specific source of relations and simply consider \( \Psi_r \), a set of relations, as an abstract entity provided to the search process.
In a Hasse diagram, the vertices are the members of a poset (partially ordered set); \( C_{1,i} \) is drawn above \( C_{2,i} \) if and only if \( C_{1,i}, C_{2,i} \in C_i \) and \( C_{2,i} \leq_T C_{1,i} \). We can say that \( C_{1,i} \) covers \( C_{2,i} \) if \( C_{1,i}, C_{2,i}, C_{j,i} \in C_i \), \( C_{2,i} \leq_T C_{1,i} \), and no element \( C_{3,i} \in C_i \) satisfies \( C_{2,i} \leq_T C_{3,i} \leq_T C_{1,i} \). The depth of a node, \( C_{j,i} \), in a Hasse diagram is the minimum number of links that need to be traversed to reach \( C_{j,i} \) from any node at the highest level. Note that this ordering depends on the chosen class comparison statistic.

In a sampling-based search, the partial-order construction process is based on a finite set of samples taken from each of the subsets, \( C_{1,i}, C_{2,i}, \ldots, C_{N,i} \). Let us denote the approximate descriptions of these classes using the sample sets by \( \hat{C}_{1,i}, \hat{C}_{2,i}, \ldots, \hat{C}_{N,i} \). Let \( C_{[1]} \) be the ordering of classes from relation \( i \). Denote the class at rank \( b \) from the bottom of this ordering by \( C_{b,i} \). This means the top ranked class in this ordering is denoted by \( C_{[N],i} \). The partial ordering constructed using the sample estimates may be different from the actual ordering. Figure 2 (bottom) shows that the partial ordering constructed from sample estimates may differ from the actual ordering.

3.3.2 Selection of better classes

Once the classes are partially ordered based on \( \leq_T \), the next immediate objective is to select \( M \) "top" subsets. Since \( C_{[1]} \) is a partial order, the notion of "top" needs to be properly defined. This is an implementation-specific issue. One possible way to define this may be based on the depth of a subset in the Hasse diagram. For the current purpose, we assume that there exists a subroutine \( \text{TOP}(C_{[1]}, \mathcal{M}) \) which returns the set of "top" \( M \) subsets from the collection \( C_{[1]} \). Denote the particular subset that contains \( z^* \)—the globally optimal solution—by \( C_{*,i} \). If we denote the ordered collection of sample sets \( \hat{C}_{1,i}, \hat{C}_{2,i}, \ldots, \hat{C}_{N,i} \) by \( \hat{C}_{[1]} \), then we would like \( \hat{C}_{*,i} \) to be one among the collection of classes returned by \( \text{TOP}(\hat{C}_{[1]}, \mathcal{M}) \). Unfortunately, this is very unlikely, unless \( C_{*,i} \) itself is not within \( \text{TOP}(C_{[1]}, \mathcal{M}) \). This sets the stage for introducing the notion of inherently better or worse relations with respect to a given problem, a class comparison statistic, and memory size. This is considered in the following section.

3.3.3 Selection of appropriate relations: The delineation property

A relation is not appropriate with respect to the chosen class comparison statistic and the BBO problem if the class containing the optimal solution is not one
among some top-ranked classes, ordered based on this statistic. If the class \( C_{*,i} \) is not among the top \( M_i \) classes, the algorithm is not likely to succeed (neglecting any chance that may rank \( C_{*,i} \) higher than its actual ranking). Let us quantify this requirement of a relation to be appropriate by a function \( DC(r_i, T, M_i) \). This function returns a one if \( C_{*,i} \in \top(C_{T[i]}, M_i) \); otherwise, it returns a zero. This will be denoted by \( DC() \) in short (DC stands for Delineation Constraint), unless otherwise required.

**Definition 1 (Proper delineation):** For a given BBO problem, a relation \( r_i \), a class comparison statistic \( T \), and a memory size, \( M_i \). if \( DC(r_i, T, M_i) = 1 \), we say that \( r_i \) properly delineates the search space.

This delineation constraint plays an important role in SEARCH processes. It essentially qualifies or disqualifies a relation for a particular search problem. If a relation does not properly delineate the search space, there is very little chance that the class with the best solution will be detected. Therefore, for a given class comparison statistic, whether or not a relation is appropriate can be directly quantified based on this characteristic function. However, in reality the algorithm does not know this constraint. The algorithm has to decide whether or not a relation properly delineates the search space from the limited number of samples taken from the search space. Therefore, determining whether or not a relation properly delineates is again essentially a decision-making problem.

Given a finite set of samples from the search space, a class comparison statistic, \( T \), the memory size \( M_i \), and a relation \( r_i \), the goal is to determine whether a relation classifies the search space in such a way that \( C_{*,i} \) is in \( \top(C_{T[i]}, M_i) \). Since the problem is now reduced to a decision-making problem instead of the previous binary characteristic function, we can approach it using the same strategy that we took for selecting better classes. In other words, we can start comparing relations, estimate how well a relation would satisfy the delineation requirement compared to another relation, and choose the better relations. This problem is similar to the class selection problem; the only difference is that now we are trying to choose better relations instead of better classes. The first question is: How do we compare two relations? While comparing two classes, we needed a class comparison statistic, \( T \). The same thing can be done for relations. Let us denote a relation comparison statistic by \( T \). This statistic is used to compute an ordering among the relations. Denote this ordering relation by \( \leq_T \). The ordering among the relations in \( \Psi_T \) may not remain the same when relations are compared based on a limited number of samples. In other words, if \( r_j \leq_T r_i \), then it is not necessarily true that \( r_j \leq_T r_i \); we denote a relation \( r_i \) when compared based on limited sampling by \( \bar{r}_i \). This process of relation selection involves decision making in absence of complete knowledge and it is therefore susceptible to decision errors. The following section describes the resolution process.

### 3.3.4 Resolution of classes

Resolution plays an important role in SEARCH. Resolution takes advantage of possible delineability of relations. Classification of the search space defined by a relation is moderated by the resolution process. If possible, resolution eliminates classes that are not necessary to consider by using the information gathered by previous evaluations of some other relations. Let \( r_j \) be a relation that properly delineates the search space with memory size \( M_j \). Let \( r_i \) be the relation currently being evaluated, and \( r_i \) can be logically expressed as \( r_j \land r_k \), where \( r_k \) is a relation. Resolution of \( C_i \) with respect to \( r_j \) eliminates those classes of \( C_i \) that need not be considered using our knowledge about \( r_j \). This resolved set of classes in \( C_i \) can be formally defined as

\[
\bigcup_{b=N_j} \bigcup_{N_i-M_i} C_{a,b} \cap C_{[a,b]}
\]

where the index \( b \) varies over the all \( M_j \) top ranked classes of relation \( r_j \) and index \( a \) denotes the different \( N_i \) classes in \( C_i \). \( C_{[a,b]} \) is the rank \( b \) member of the ordered collection of classes in \( C_i \) and \( C_{a,b} \) is the a member of the unordered collection of classes \( C_i \). The following sections present a brief description of the analysis of the different decision makings in SEARCH.

### 4 Decision Making in SEARCH

The previous sections presented SEARCH from both informal and formal points of view. They also posed the class and relation selection processes as decision problems in absence of complete knowledge. In this section we analyze these two sources of decision error and combine them to develop an expression for the overall success probability. Two kinds of decision errors may make the selection of better classes erroneous:

1. The relation used to define collection \( C_i \) is such that for the chosen \( T \), the subset \( C_{*,i} \) is not in \( \top(C_{T[i]}, M_i) \). Therefore, despite how well the sampling is done, the selection process will always miss the subset containing \( x^* \), unless \( C_{*,i} \) is ranked higher by sampling error. A search algorithm needs to determine whether or not a relation does this from a finite number of samples.
Therefore, this could be a source of error. Let us call this error the relation selection error.

2. Even when \( C_{\ast, i} \) is in \( \text{TOP}(C_{\ast 1}, M_i) \), sampling error can produce a different partial order structure for \( \hat{C}_{1,i}, \hat{C}_{2,i}, \ldots \hat{C}_{N,i} \). As a result \( \hat{C}_{\ast, i} \) may not be in \( \text{TOP}(\hat{C}_{\ast 1}, M_i) \). The sampling error may result in incorrect ordering of the classes and we call this the class selection error.

These two dimensions of decision error in BBO determine the success probability. The following sections analyze the success probabilities associated with each of these dimensions. Finally, they are combined to develop an expression for the overall success probability.

4.1 Relation selection success

If an algorithm does not properly delineate the search space, it is not likely to select the class containing the optimal solution. Since, in the absence of knowledge, there is no way to know whether a relation satisfies this requirement or not a priori, this can only be estimated based on the sampling information. Relations are ordered based on the measure \( \tau_r \) and \( ||S_r|| \) top relations are selected. Since these top \( ||S_r|| \) relations are just the estimated relations that satisfy the delineation constraint, there is the possibility of decision error. If \( \hat{r}_i \) is actually in the top \( ||S_r|| \) relations, then the probability that \( \hat{r}_i \) will also be within the top \( ||S_r|| \) relations depends on correct decision making in the comparison with at least \( \Psi_r - ||S_r|| \) relations. Denote a relation which actually does not satisfy the delineation constraint by \( r_j \). If the probability that \( r_j \leq \hat{r}_i \) over all possible relations is denoted by, \( Pr(r_j \leq \hat{r}_i)_{\text{min}} \), the success probability that \( \hat{r}_i \) will be one among the top \( ||S_r|| \) relations is

\[
Pr(\text{CRS} \mid r_i) \geq Pr(\hat{r}_i \leq \tau_r \hat{r}_i)_{\text{min}}^{||S_r||},
\]

where \( \text{CRS} \) stands for correct relation selection. The following section considers the decision making in class selection process.

4.2 Class selection success

Let us now consider the class selection problem. The best solution is in any of the selected subsets will be denoted by \( Pr(\text{CCS}|r_i) \). \( \text{CCS} \) stands for correct class selection and conditional to \( r_i \), it reflects its association with relation \( r_i \). Let \( Pr(\hat{C}_{j,i} \leq \tau \hat{C}_{\ast, i}) \) denote the success probability given that \( C_{j,i} \leq \tau C_{\ast, i} \), and let \( Pr(\hat{C}_{j,i} \leq \tau \hat{C}_{\ast, i})_{\min} \) be the minimum value of \( Pr(\hat{C}_{j,i} \leq \tau \hat{C}_{\ast, i}) \) over every \( \hat{C}_{j,i} \) which has a depth greater than that of \( \hat{C}_{\ast, i} \) and there is a link connecting it to \( \hat{C}_{\ast, i} \). Now noting that \( M_i \) top classes are selected,

\[
Pr(\text{CCS} \mid r_i) \geq Pr(\hat{C}_{j,i} \leq \tau \hat{C}_{\ast, i})_{\min}^{N_{\text{min}} - M_i}.
\]

This gives the success probability for a particular relation \( r_i \).

4.3 Overall success

The overall success probability for all the considered relations in \( S_r \) then becomes

\[
Pr(\text{CS} \mid \forall r_i \in S_r) = \prod_{\forall r_i \in S_r} Pr(\text{CRS} \mid r_i)Pr(\text{CCS} \mid r_i).
\]

This equation captures the general idea that will be used in the following sections. As we see, at a higher level, the success of a blackbox search algorithm depends on

1. the success probability in finding relations that properly delineate the search space and
2. the success probability in detecting the class which actually contains the desired solution.

The following sections specialize the observations of this framework to a specific class comparison statistic and representation. First, we consider ordinal class and relation comparison statistic.

5 Ordinal Class and Relation Selection

Constructing a total order and selection of some \( M_i \) top subsets from that order have been studied using both parametric and non-parametric approaches (Gibbons, Sobel, & Olkin, 1977). If we are willing to make assumptions about the individual distributions of the members of \( C_i \), nice statistics can be formulated to solve this selection problem. However, in the following discussion, we adopt a non-parametric, ordinal approach (David, 1981) that allows a distribution-free analysis of the relation and class comparison process. The purpose of this section is to derive bounds on the success probability and sample complexity for a quite general ordinal relation and class comparison statistics.

Section 5.1 considers an ordinal class comparison statistic and the SEARCH framework is specialized for this statistic. Section 5.2 further specializes SEARCH for an ordinal relation comparison statistic. Section 5.3 combines the decision making for both better classes and relations; it also bounds the overall success probability. Finally, Section 5.4 derives the overall sample complexity and discusses its properties.
5.1 Ordinal class selection

As we argued in the previous section, BBO can be viewed as a combined process of search for better relations and better classes defined by each of these relations. Let us first consider the class comparison process from an ordinal perspective. In order statistics any two classes will be compared based on their $\alpha$ quantile of the cumulative distribution function (cdf). A quantile of order $\alpha$ can be defined as the number $G_\alpha$ such that $F(G_\alpha) = \alpha$, where $F(\Phi)$ is the cdf of $\Phi$. This definition of quantile is not fully satisfactory when the cdf is discrete and the $\alpha$ quantile may not be unique. In such cases, however, we can define $\Phi$ as any convex combination of points in the closure of the set $\{G_\alpha : F(G_\alpha) = \alpha\}$. To convey the main idea without unnecessary cluttering of symbols, let us assume that the $\alpha$ quantile is unique. We should note that such quantile-based class comparison will always produce a total order on the collection $C_j$.

Consider the comparison between two classes $C_{j,i}$ and $C_{k,i}$. Assume that we take $n$ samples from each of these classes. We shall denote $n$ samples from the class $C_{j,i}$ by $\hat{\Phi}_{1,j,i}, \hat{\Phi}_{2,j,i}, \ldots, \hat{\Phi}_{n,j,i}$, the corresponding objective function values by $\Phi_{1,j,i}, \Phi_{2,j,i}, \ldots, \Phi_{n,j,i}$. These $n$ samples can be totally ordered on the basis of their objective function values as follows:

$$\hat{\Phi}_{[1,j,i]} \leq \Phi_{[2,j,i]} \leq \cdots \leq \Phi_{[n,j,i]}$$

where, $\hat{\Phi}_{[w,j,i]} \leq \Phi_{[v,j,i]}$ if $\hat{\Phi}_{[w,j,i]} \leq \Phi_{[v,j,i]}$. $\hat{\Phi}_{[v,j,i]}$ denotes the $k$-th order statistic. The sample estimate of the $\alpha$ quantile for the class $j$ is denoted by $y_{\alpha,j}$. Define an integer $\tau = \alpha(n + 1)$; then, $y_{\alpha,j} = \hat{\Phi}_{[\tau,j,i]}$. If $\alpha(n + 1)$ is not an integer, we can set $\tau$ equal to the largest integer contained in $\alpha(n + 1)$ and compute $y_{\alpha,j}$ as follows:

$$y_{\alpha,j} = [\alpha(n + 1) - 1] \hat{\Phi}_{[\tau,j,i]} + \alpha(n + 1) - \tau \hat{\Phi}_{[\tau+1,j,i]}$$

This basically interpolates between two adjacent order statistics to approximate the point where the cdf is equal to $\alpha$. Again, to keep things simpler, we assume that $\alpha(n + 1)$ is an integer.

Figure 3 shows the cumulative distribution function $F'$ and $F$ of two arbitrary subsets $C_{j,i}$ and $C_{k,i}$, respectively. When these two classes are compared on the basis of the $\alpha$ quantile, then we say $C_{j,i} \leq_C C_{k,i}$ since $\Phi_{[\tau,j,i]} \leq \Phi_{[\tau,k,i]}$ and $\Phi_{[\tau,k,i]}$ and $\Phi_{[\tau,j,i]}$ are the solutions of $F'(\Phi_{[\tau,j,i]}) = \alpha$ and $F(\Phi_{[\tau,k,i]}) = \alpha$, respectively. Let us define

$$d = F(\Phi_{[\tau,k,i]}) - F(\Phi_{[\tau,j,i]})$$

The variable $d$ defines the zone of indifference, which is basically the difference in the percentile value of

![Figure 3: Fitness distribution function of two classes $C_{[j,i]}$ and $C_{[k,i]}$.](image)

$\Phi_{[\tau,k,i]}$ and that of $\Phi_{[\tau,j,i]}$ computed from the same cdf $F$. Figure 3 clearly explains this definition.

It can be easily shown that for $r$-th order statistics of set $C_{j,i}$ (David, 1981), $\Phi_{[r,j,i]}$.

$$Pr(\Phi_{[r,j,i]} \leq c') = \sum_{w=r}^{n} \binom{n}{w}(F(c'))^w(1 - F(c'))^{n-w}. \quad (4)$$

The probability of correct selection among these two classes can be written as (Kargupta, 1995)

$$Pr(\Phi_{[r,k,i]} \leq \Phi_{[r,j,i]}) \geq 1 - 2^n H(\alpha) (\alpha - d)^{\alpha n}. \quad (5)$$

where $H(\lambda)$ is the binary entropy function, $H(\lambda) = -\lambda \log_2 \lambda - (1 - \lambda) \log_2 (1 - \lambda)$. $H(0) = H(1) = 0$ and $H(\lambda)$ takes the maximum value for $\lambda = 0.5$. For relation $r$, if we denote the cdf of the class containing the optimal solution $x^*$. by $F(\Phi_{[x,i]})$, then define

$$d'' = \min\{F(\Phi_{[r,x,i]}) - F(\Phi_{[r,j,i]}); \forall j\},$$

the probability that the class $C_{*,i}$ will be within the top $M_i$ classes is

$$Pr(\text{CCS} \mid r_i) \geq 1 - 2^n H(\alpha) (\alpha - d'')^{\alpha n} M_i^{-M_i}. \quad (6)$$

Given relation $r_i$ that properly delineates the search space, Equation 6 can be used to compute the probability that $C_{*,i}$ will be within the top $M_i$ classes. Before we proceed toward computing the overall correct selection probability, we need to consider the search in the relation space.

5.2 Ordinal relation selection

A relation is appropriate if it properly delineates the search space. Determining whether or not a relation
satisfies this constraint with absolute certainty is not possible unless we completely enumerate the search space. Therefore, in reality, the characteristic function \( DC() \) is replaced by an estimator that measures how likely a relation satisfies delineation constraint. Let us define a measure \( \eta : \Psi_r \times 2^C \times 2^X \rightarrow \mathbb{R} \). \( 2^C \) denotes the collection of classes and \( 2^X \) denotes the sample set. For a given relation \( r_i \), the corresponding set of classes \( C_i \), and a sample set \( S \), this measure \( \eta(r_i,C_i,S) \) returns a real value that corresponds to the chances of \( r_i \) to satisfy the delineation constraint (i.e. \( C_{*,i} \) is a member of \( \text{TOP}(C_{i},M_{i}) \)). In short, \( \eta(r_i,C_i,S) \) will be written as \( \eta \). This measure will be used to order the equivalence relations \( r_i, r_j \in \Psi_r \). Let us again adopt an ordinal approach to compare different relations, just as we did for selection of better classes. For any two relations \( r_i \) and \( r_j \), the corresponding \( \eta_i \) and \( \eta_j \) can be treated as random variables.

In the class space the random variable was defined to be the objective function value of the samples. Unlike that, here in the relation space the random variable is the measure \( \eta \), which is defined over a collection of classes and a sample set for a given relation. Since, for a given \( r_i \), the computation of \( \eta_i \) depends on a tuple from \( (2^C \times 2^X) \), a collection of \( n_r \) such tuples will generate a distribution of different values of \( \eta_i \). Figure 4 shows the cdf of two competing relations \( r_i \) and \( r_j \). Let us say that \( r_i \) satisfies the delineation constraint and \( r_j \) does not.

If we compare these two relations on the basis of some \( \tau_r \)-th order statistic, the success probability can be computed in exactly the same way that we just did for class comparisons. If \( \alpha_r \) be the corresponding percentile,

\[
Pr(\tilde{r}_{r_i,j} \leq \alpha_r, \tilde{r}_{r_j,i}) \geq 1 - 2^{\tau_r \cdot H(\alpha_r)}(\alpha_r - d_r^* )^{n_r-\tau_r} \quad (7)
\]

where

\[
d_r^* = \min \{F(\eta_{[r_r,j]} - F(\eta_{[r_j,i]})) \forall j, \forall i\}
\]

where \( F \) is the cdf of the relation comparison statistic of relation \( r_i \). \( d_r^* \) is essentially similar to \( d'' \), except that this is for relation comparison instead of the previous case of class comparison. In the most general case, a relation needs to be chosen out of the all possible relations in \( \Psi_r \). However, in reality, it may be true that only a subset of \( \Psi_r \) is chosen at a time. In the following analyses we consider the general case, in which all relations in \( \Psi_r \) are under consideration. Let us assume that among these \( \Psi_r \) relations, the set \( \Psi_g \subseteq \Psi_r \) contains all the relations that properly delineate the search space. If relation \( r_i \in \Psi_g \), then the probability that \( r_i \) will be correctly identified is

\[
Pr(CRS \mid r_i \in \Psi_g) \geq \left[ 1 - 2^{n_r \cdot H(\alpha_r)}(\alpha_r - d_r^* )^{n_r-\tau_r} \right] \|\Psi_g\| - \|\Psi_r\|.
\]

This is the success probability in choosing one good relation. If we need \( S \subseteq \Psi_r \) relations to solve a problem, we can bound the overall success probability in the relation space as follows (Kargupta, 1995):

\[
n_r > \frac{\log(1 - q_r^{1/(\|\Psi_g\| - \|\Psi_r\|)})}{-d_r^*} \quad (8)
\]

Inequality 8 can be further rearranged. Define delineation-ratio,

\[
\Omega = \frac{\|\Psi_g\|}{\|\Psi_r\|} \quad (9)
\]

When this ratio is high, searching for appropriate relations is easier, since most of the members of the relation space are appropriate for properly classifying the search space. Using definition 9 and Inequality 8 we can write

\[
n_r > \frac{\log(1 - q_r^{1/(\|S_r\| - \|\Psi_r\| - (1-\Omega))})}{-d_r^*}. \quad (10)
\]

This bounds the overall computational complexity in the relation space. Inequality 10 can be further simplified using the approximation \( \log(1 - x) \approx -x \) for \( x \ll 1 \).

\[
n_r > q_r^{1/(\|S_r\| - \|\Psi_r\| - (1-\Omega))}. \quad (11)
\]

This clearly shows that \( n_r \) increases as \( q_r \) increases and that \( n_r \) increases when \( d_r^* \) is reduced. Since \( q_r \leq 1, n_r \) decreases as \( \Omega \) increases. As the number of relations needed to solve the problem, \( \|S_r\| \), increases, \( n_r \) also increases. The collection of relations \( \Psi_r \) defines the
complete search space for relations. The larger the number of relations in \( \Psi_r \), the more computation is required for searching for appropriate relations.

The decision making in the relation and class spaces are combined in the following section.

5.3 Overall selection success

Let us now combine the search for better relation and better classes together and compute the overall success probability. Define

\[
d' = \min \{ F(\Phi_{[r],j,i}) - F(\Phi_{[r],j,j}) \forall j, \forall i \}.
\]

\( d' \) is basically the minimum possible value of \( d \) over all classes (index \( j \)) which are compared with class containing the optimal solution and all relations (index \( i \)) in \( S_r \). Now let us consider the overall class selection success probability given by equation 3. Note that the relation \( \leq_\alpha \) imposes a total order onto \( C_i \). Define, \( N_{\text{max}} \) as the maximum possible value of \( N_1 \) over all relations in \( S_r \); Let \( M_{\text{min}} \) and \( q_r \) be the minimum value of memory size \( M_i \) and bound on success probability in choosing a relation respectively over all the relations in \( S_r \). In formal notation,

\[
N_{\text{max}} = \max \{ N_i \forall r_i \in S_r \}, \quad M_{\text{min}} = \min \{ M_i \forall r_i \in S_r \}.
\]

If \( d^* \) is a constant such that \( d' \geq d^* \), just like the previously defined \( d^*_r \), then the overall success probability can be bounded as follows (Kargupta, 1995):

\[
\left( \frac{q_r}{q_r} \right)^{\| S_r \| (N_{\text{max}} - M_{\text{min}})} \leq 1 \quad \text{and} \quad \log(1 - \frac{q_r}{q_r}) \left( \frac{q_r}{q_r} \right)^{\| S_r \| (N_{\text{max}} - M_{\text{min}})} \approx -d^* \quad \text{for} \quad z \ll 1.
\]

Inequality 15 presents a clear picture of the contributions of different parameters of the SEARCH framework into the sample complexity. Recall that \( q_r \) is the bound on the overall success probability in the relation and class spaces combined. Clearly, sample complexity \( SC \) grows polynomially with \( q_r \). On the other hand, \( q_r \) is the minimum bound in the success probability in choosing all \( \| S_r \| \) relations correctly. The cost of demanding higher success probability in the relation space shows up in inequality 10. However, as we increase our success probability in the relation space, the overall success probability in the combined relation and class spaces increases. The sample complexity should therefore decrease as success probability in the relation space increases. Inequality 15 clearly shows that \( SC \) decrease with increase in \( q_r \). Note that the ratio \( \left( \frac{q_r}{q_r} \right)^{\| S_r \| (N_{\text{max}} - M_{\text{min}})} \) approaches 1 in the limit as \( \| S_r \| (N_{\text{max}} - M_{\text{min}}) \) approaches infinity. Therefore, \( SC \) grows most linearly with the maximum index value \( N_{\text{max}} \) and the cardinality of the set \( S_r \). Recall that \( d^* \) defines the desired region of indifference; in other words, it defines a region in terms of percentile within which any solution will be acceptable. The sample complexity decreases as the \( d^* \) increases.

This bound on sample complexity establishes an insight introduced earlier in this section. In the beginning of Section 3, we argued that BBO can perform no better than random enumeration unless we try to exploit the relations among the members of the search space. Now that we have a closed-form bound on sample complexity, let us investigate the case when no relations are assumed among the members. Saying no relations are assumed essentially means that there exists only one relation in \( \Psi_r \) that basically divides the complete search space into a set of singleton classes. For our 4-bit problem representation, this could be the relation \( fff \). This relation divides the search space into 16 singleton classes, which is essentially the complete search space. From the definition of global optima, we know that such a relation always properly delineates the search space. Therefore, \( S_r = 1 \) and \( q_r = 1 \). The index of this relation is same as the cardinality of the search space. So, \( N_{\text{max}} = \| \mathcal{X} \| \),
where $|\mathcal{X}|$ denotes the size of the search space $\mathcal{X}$. Substituting these in Inequality 15 we get

$$SC \leq \frac{|\mathcal{X}|q^{T(d'-d^*)}}{d^*}.$$  \hspace{1cm} (16)

This inequality clearly tells us that the overall sample complexity becomes the size of the search space when we completely neglect all relations that put at least two members together in a class. The only advantage that we get comes from our relaxation in the desired solution quality ($d^*$) and the overall success probability ($q$). This confirms that although SEARCH provides one particular perspective of BBO, the importance on relations is fundamental, and it should be emphasized in all possible models of BBO that aspire to guide designing BBO algorithms that perform better than random enumerative search. No BBO algorithm can transcend the limit of random enumerative search without inducing relations among the members.

6 Conclusion

The SEARCH framework provided insightful quantitative results relevant to any BBO algorithms. Evolutionary search algorithms are no exceptions. The fundamental principles of SEARCH are equally applicable to them. SEARCH realizes the importance on searching for appropriate relations. Although genetic algorithms do not properly search for relations defined by the representation (Kargupta, 1995) Holland realized the importance and approached the design of GAs in terms of schemata and partitions, which are essentially equivalence classes and relations respectively. Relations however do not always have to be defined by representation. Search operators, heuristics can also do that job. Nevertheless, we must realize the importance for searching for relations and take a constructive approach to deal with the different decision making aspects of BBO. We hope that the SEARCH effort has taken one small step toward that direction.

7 Acknowledgment

This work was supported by AFOSR Grant F49620-94-1-0103 and the Illinois genetic Algorithm Laboratory. The first author also acknowledges the support from US. Department of Energy and the helpful comments from Jeffrey Horn and George Harik.

References


