Sensibility of Linkage Information and Effectiveness of Estimated Distributions

Chung-Yao Chuang cychuang@nclab.tw
Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan

Ying-ping Chen* ypchen@nclab.tw
Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan

Abstract

The probabilistic model building performed by estimation of distribution algorithms (EDAs) enables these methods to use advanced techniques of statistics and machine learning for automatic discovery of problem structures. However, in some situations, it may not be possible to completely and accurately identify the whole problem structure by probabilistic modeling due to certain inherent properties of the given problem. In this work, we illustrate one possible cause of such situations with problems consisting of structures with unequal fitness contributions. Based on the illustrative example, we introduce a notion that the estimated probabilistic models should be inspected to reveal the effective search directions and further propose a general approach which utilizes a reserved set of solutions to examine the built model for likely inaccurate fragments. Furthermore, the proposed approach is implemented on the extended compact genetic algorithm (ECGA) and experiments are performed on several sets of additively separable problems with different scaling setups. The results indicate that the proposed method can significantly assist ECGA to handle problems comprising structures of disparate fitness contributions and therefore may potentially help EDAs in general to overcome those situations in which the entire problem structure cannot be recognized properly due to the temporal delay of emergence of some promising partial solutions.

Keywords

Sensible linkage, effective distribution, linkage sensibility, probabilistic model, model pruning, estimation of distribution algorithm, extended compact genetic algorithm, evolutionary computation.

1 Introduction

Estimation of distribution algorithms (EDAs; Mühlenbein and Paaß, 1996; Larrañaga and Lozano, 2001; Pelikan, Goldberg, et al., 2002) are a class of evolutionary algorithms that replace the traditional variation operators, such as mutation and crossover, by building a probabilistic model on promising solutions and sampling the built model to generate new candidate solutions. Using probabilistic models for exploration enables these methods to automatically capture the likely structure of promising solutions and exploit the identified problem regularities to facilitate further search. It is presumed that EDAs can detect the structure of the problem by recognizing the regularities within the promising solutions. However, for certain problems, EDAs are unable to identify the

*To whom correspondence should be addressed.

entire structure of the problem at a given time because the set of selected solutions on which the probabilistic model is built contains insufficient information regarding some parts of the problem and renders EDAs incapable of processing these parts accurately.

This paper starts by observing the evolutionary process of an EDA when dealing with an exponentially scaled problem, and recognizing that the population on which the probabilistic model is built does not necessarily contain sufficient information for all problem structures to be detected completely and accurately. Based on the observation, this study proposes a general concept that estimated probabilistic models should be inspected to reveal the effective search directions, and we provide a practical approach that utilizes a reserved set of solutions to examine the built model for the fragments that may be inconsistent with the actual problem structure. Furthermore, the proposed approach is implemented on the extended compact genetic algorithm (ECGA; Harik, 1999) and experimented on several sets of additively separable problems with different scaling difficulties (Goldberg, 2002) to demonstrate the applicability.

The following section briefly reviews the research topics concerning this study. Section 3 then demonstrates the interaction between the scaling difficulty and probabilistic model building performed by EDAs. More specifically, we will investigate how the scaling difficulty shadows the ability of EDAs to recognize problem structures and causes inaccurate processing on the part of some solutions. Accordingly, a general approach will be proposed in Section 4 to resolve this issue and enforce accurate processing during the optimization process. In Section 5, an implementation of the proposed approach on the extended compact genetic algorithm will be detailed. Section 6 presents the empirical results, followed by discussion and analysis in Section 7. Finally, Section 8 concludes the paper.

2 Background

Genetic algorithms (GAs; Holland, 1992; Goldberg, 1989) are search techniques loosely based on the paradigm of natural evolution, in which species of creatures tend to adapt to their living environments through mutation and inheritance of useful traits. Genetic algorithms mimic this mechanism by introducing artificial selections and genetic operators to discover and recombine partial solutions. By properly growing and mixing promising partial solutions, which are often referred to as building blocks (BBs; Goldberg, 2002), GAs are capable of efficiently solving a variety of problems. The ability to implicitly process a large number of partial solutions has been recognized as an important source of the computational power of GAs. According to the Schema theorem (Holland, 1992), short, low-order, and highly fit subsolutions increase their share in the final combined solution. Further, as stated in the building block hypothesis (Goldberg, 1989), GAs implicitly decompose a problem into subproblems by processing building blocks. This decompositional bias is a good strategy for tackling many real-world problems, because real-world problems can oftentimes be reliably solved by combining the pieces of promising solutions in the form of problem decomposition.

However, proper growth and mixing of building blocks are not always achieved. GAs in the simplest form employ fixed representations and problem-independent recombination operators, which often breaks promising partial solutions while performing crossovers. This can cause crucial building blocks to vanish, thus leading to a convergence to local optima. In order to overcome this building block disruption problem, various techniques have been proposed. In this study, we focus on one line of effort often called the estimation of distribution algorithm (EDA; Mühlenbein and Paaß, 1996;

Larrañaga and Lozano, 2001; Pelikan, Goldberg, et al., 2002). These methods construct probabilistic models of promising solutions and utilize the built models to generate new solutions. Ideally, by detecting dependencies among variables through probabilistic modeling, these approaches can capture the structure of the problem and thus avoid the disruption of identified partial solutions. Early EDAs, such as population-based incremental learning (PBIL; Baluja, 1994) and the compact genetic algorithm (cGA; Harik et al., 1999), assume no interaction between decision variables. That is, decision variables are assumed to be independent of each other. Subsequent studies progressed from capturing pairwise interactions, such as mutual-information-maximizing input clustering (MIMIC; De Bonet et al., 1997), Baluja's dependency tree approach (Baluja and Davies, 1997), and the bivariate marginal distribution algorithm (BMDA; Pelikan and Mühlenbein, 1999), to modeling multivariate interactions, such as the extended compact genetic algorithm (ECGA; Harik, 1999), the Bayesian optimization algorithm (BOA; Pelikan et al., 1999), the estimation of Bayesian network algorithm (EBNA; Etxeberria and Larrañaga, 1999), the factorized distribution algorithm (FDA; Mühlenbein and Mahnig, 1999), and the learning version of FDA (LFDA; Mühlenbein and Höns, 2005). Along this line of research, questions arose naturally regarding the ability of EDAs to solve problems and the probabilistic models employed to learn the problem structures. Early studies recognized that solving problems composed of higher order building blocks is not expected to be accomplished by using just any probability density structure. Bosman and Thierens (1999) demonstrated that even when the set of variables forming a building block is linked and expressed by the best possible MIMIC-like chain structure, directly sampling that chain to generate new solutions is not a good strategy for reliable optimization. More recently, Echegoyen et al. (2007) compared the behavior of EBNA with approximate and exact Bayesian network learning. In another vein, Hauschild et al. (2007) analyzed the structure and complexity of learned probabilistic models and attempted to facilitate the model building process by incorporating the knowledge acquired from previous models (Hauschild et al., 2008).

Another topic relevant to this study is the impact of disparate *scale* among different building blocks on the behavior and performance of evolutionary algorithms. It is commonly observed that building blocks with higher marginal fitness contributions—salient building blocks—converge before those with lower marginal fitness contributions. This sequential convergence behavior is referred to as domino convergence (Thierens et al., 1998). In real-world applications, it is often the case that some parts of the problem are more prominent and contribute more to the fitness than the other parts.[1] Such a situation can pose two types of difficulties. Firstly, because the processing on the population is statistical in nature, building block scaling can cause inaccurate processing of less fit building blocks (Goldberg et al., 1992; Goldberg and Rudnick, 1991). The second difficulty arises because the lower fitness of a building block generally causes it to be processed at a later time compared to those of higher fitness. This delay on timeline can cause the building block to converge under random pressure, instead of proper selective pressure. Previous studies on this topic include the explicit role of scale in a systematic experimental setting (Goldberg et al., 1990), a theoretical

[1]The reader may note that this statement cannot be formally proved nor disproved because we do not know nor even have a way to estimate the distribution of all real-world problems. However, this intuition can be better articulated by the explanation provided in Goldberg (2002): differences in scale are likely to be common across the space of likely problems, that is, the chance that we encounter differences in scale may be much larger than encountering equivalence in scale.

model on the convergence behavior of exponentially scaled problems (Thierens et al., 1998), an extension of that model to building blocks more than one variable long (Lobo et al., 2000), and a convergence model of linkage learning genetic algorithms (LLGAs; Harik, 1997) on problems with different scaling setups (Chen and Goldberg, 2005).

Although the aforementioned scaling difficulty exists in a number of problems and degrades the performance of many evolutionary algorithms (EAs), there are scant investigations concerning the behavior of EDAs in the presence of scaling difficulties. Therefore, this study attempts to explore how the scaling difficulty affects EDAs, and proposes a practical countermeasure to assist EDAs on problems with different scalings. Specifically, we propose the notion that the estimated probabilistic models should be examined to enforce accurate processing of building blocks and prevent random drift from taking place. In the remainder of this paper, our approach will be demonstrated and evaluated on the test problems constructed by concatenating several trap functions. A k-bit trap function is a function of unitation[2] which can be expressed as

$$f_{\text{trap}_k}(s_1 s_2 \cdots s_k) = \begin{cases} k, & \text{if } u = k \\ k - 1 - u, & \text{otherwise} \end{cases},$$

where u is the number of ones in the binary string $s_1 s_2 \cdots s_k$. The trap functions were used pervasively in the studies concerning EDAs and other evolutionary algorithms because they provide well-defined structures among variables, and the ability to recognize intervariable relationships is essential to solve the problems consisting of traps (Deb and Goldberg, 1993, 1994).

3 Linkage Sensibility

The ability of EDAs to handle the building block disruption problem comes primarily from the explicit modeling of selected promising solutions using probabilistic models. The model construction algorithms, though they differ in their representative power, capture the likely structures of good solutions by processing the population-wise statistics collected from the selected solutions. By reasoning the dependencies among different parts of the problem and the possible formations of good solutions, reliable mixing and growing of building blocks can be achieved. As noted by Harik (1999), learning a good probability distribution is equivalent to learning linkage, where linkage refers to the dependencies among variables. Bosman and Thierens (1999) further recognized that in order to achieve reliable optimization, linkage information should be utilized in a way such that each corresponding building block can be identified and used as a whole.

In most studies on EDAs, it is presumed that EDAs can detect linkage and recognize building blocks according to the information contained in the set of selected solutions. However, in this study, we argue that in some situations, accurate and complete linkage information cannot be acquired by distribution estimation because the selected set of solutions on which the model is built contains insufficient information on the lower fitness parts of the problem. For example, consider a 16-bit maximization problem

[2] A function of which the function value depends only on the number of ones in the binary input string.

Table 1: Marginal product models built by ECGA when solving an exponentially scaled problem. Each group of variables represents a marginal model in which a marginal distribution resides. The converged variables are crossed out.

Generation	Marginal product model
1	$[\boldsymbol{s_1\ s_2\ s_3\ s_4}]\ [s_5\ s_{10}\ s_{16}]\ [s_6\ s_7]\ [s_8\ s_9\ s_{12}]\ [s_{11}\ s_{14}\ s_{15}]\ [s_{13}]$
2	$[\cancel{s_1}]\ [\cancel{s_2}]\ [\cancel{s_3}]\ [\cancel{s_4}]\ [\boldsymbol{s_5\ s_6\ s_7\ s_8}]\ [s_9\ s_{13}\ s_{16}]\ [s_{10}\ s_{14}\ s_{15}]\ [s_{11}s_{12}]$
3	$[\cancel{s_1}]\ [\cancel{s_2}]\ [\cancel{s_3}]\ [\cancel{s_4}]\ [\cancel{s_5}]\ [\cancel{s_6}]\ [\cancel{s_7}]\ [\cancel{s_8}]\ [\boldsymbol{s_9\ s_{10}\ s_{11}\ s_{12}}]\ [s_{13}\ s_{16}]\ [s_{14}\ s_{15}]$
4	$[\cancel{s_1}]\ [\cancel{s_2}]\ [\cancel{s_3}]\ [\cancel{s_4}]\ [\cancel{s_5}]\ [\cancel{s_6}]\ [\cancel{s_7}]\ [\cancel{s_8}]\ [\cancel{s_9}]\ [\cancel{s_{10}}]\ [\cancel{s_{11}}]\ [\cancel{s_{12}}]\ [\boldsymbol{s_{13}\ s_{14}\ s_{15}\ s_{16}}]$

formed by concatenating four 4-bit trap functions as subproblems,

$$f(s_1 s_2 \cdots s_{16}) = \sum_{i=0}^{3} (5^{3-i} f_{\text{trap}_4}(s_{4i+1} s_{4i+2} s_{4i+3} s_{4i+4})) \ ,$$

where $s_1 s_2 \cdots s_{16}$ is a solution string. Note that in contrast to other studies of EDAs, in which the test problems are scaled uniformly, that is, the subproblems are of equal fitness, in this problem, each elementary trap function is scaled exponentially. This scaling is an abstraction for problems of distinguishable prominence or solving priority among the constitutive subproblems. Suppose that we choose ECGA (Harik, 1999), which uses a class of multivariate probabilistic models called marginal product models (MPMs) to tackle this problem.[3] By observing subsequent generations of the optimization process, a series of models built by ECGA can be obtained like those listed in Table 1. In this table, the variables enclosed by the same pair of brackets are considered dependent and are modeled jointly. Each group of variables represents a marginal model in which a marginal distribution resides, and the converged variables are crossed out.[4]

It can be observed that the models shown in Table 1 are only partially correct in each generation. More specifically, in each generation, only the most fit building block on which the population has not converged is correctly modeled. This is due to the fact that some part of the problem contributes much more than all the others combined. If one part of the problem is worth more than the others, then this part of the solution solely determines the chance regarding whether or not the solution will be selected. As a consequence, only the most fit building block can provide sufficient information to be modeled correctly, since the model searching is performed based on the selected solutions. The remaining parts of the model are primarily the result of low fitness partial solutions "hitchhiking" on the more fit building blocks.

From the above example, we can see that not all building blocks can be detected from a given set of selected solutions by probabilistic model building. Model building algorithms cannot "see" the entire structure of the problem from the selected set of solutions because the disparate scale among different building blocks prevents complete linkage information from being included in the selected population. In this work, we will refer to this concept as *linkage sensibility* and those problem structures that can be identified properly using the given set of solutions are called *sensible linkage*. Based on this notion, we reexamine EDAs on the building block disruption problem. It is clear

[3]See Section 5.1 for a more detailed description of ECGA and marginal product models.

[4]The convergence of a variable is defined as all solutions in the population possessing the same value for that variable, that is, no further changes for that variable will occur.

that the disruption problem still exists in the insensible portion of the problem because that part of the problem cannot be modeled properly. Although the above example is an extreme case of scaling, in that each subproblem is exponentially scaled, in real-world problems, it is often the case that the constitutive subproblems are weighted significantly differently, which implies that the linkage might be only partially sensible. In addition to the building block disruption problem, the random drift of the less salient parts of the problem mentioned in Section 2 further worsens the situation. These situations and issues are usually handled by increasing population size when EDAs are adopted. However, we may gain a new way to deal with these situations if it is possible to distinguish a sensible linkage from an insensible linkage.

4 Effective Distributions

The idea of sensible linkage can be closely mapped into another notion called *effective distributions*. By effective distributions, we mean that by sampling these distributions, the solution quality can be reliably advanced. Hence, the crucial criteria for effective distributions are the consistency with building blocks and the provision of good directions for further search. If it is possible to extract effective marginal distributions from the built probabilistic model, we can perform partial sampling using only these marginal distributions and leave the remaining parts of the solutions unchanged. Thus, the diversity is maintained and we are free from the building block disruption and random drift problems. For instance, returning to the earlier 16-bit optimization problem, if it is possible to identify those partial models that are built on the sensible linkage like $[s_1\ s_2\ s_3\ s_4]$ in the first generation and $[s_5\ s_6\ s_7\ s_8]$ in the second generation, we can sample only the corresponding marginal distributions which are, in this case, effective. That is, in the first generation, for each solution string, we resample only $s_1s_2s_3s_4$ according to the marginal distribution and keep $s_5s_6 \cdots s_{16}$ unchanged. In the second generation, we resample only s_1 to s_8 according to the marginal distributions and keep $s_9s_{10} \cdots s_{16}$ with the same values (note that $s_1s_2s_3s_4$ converged). In this way, we do not have to resort to increasing the population size to deal with the problems caused by the disparate building block scaling.

The above thoughts leave us one complication: the identification of effective distributions. However, the direct identification of effective distributions may be a difficult if not impossible task. It may be wise to adopt a complementary approach—to identify those marginal distributions that are *not* likely to be effective. If there is a way to identify the ineffective distributions, we can bypass them and use only the rest of the probabilistic model, and thus approximate the result of knowing effective distributions. Our idea is that we can split the entire population into two subpopulations, use only one of the subpopulations for building the probabilistic model, and utilize the other subpopulation to collect some statistics for possible indications of ineffectiveness of certain marginal distributions in the probabilistic model built on the first subpopulation. That is, with some appropriate heuristics or criteria, we can prune the likely ineffective portions of the model.

In the next section, our implementation in ECGA of the proposed concept will be detailed. More specifically, a judging criterion will be proposed to detect the likely ineffective marginal distributions of a given marginal product model.

5 ECGA with Model Pruning

This section starts with a brief review of the ECGA (Harik, 1999). Based on the idea of detecting the inconsistency of statistics gathered from two subpopulations of the

Table 2: An example of a marginal product model that defines a probability distribution over four variables. The variables enclosed in the same brackets are modeled jointly, and each variable subset is considered independent of the other variable subsets.

$[s_1]$	$[s_2\ s_4]$	$[s_3]$
$P(s_1 = 0) = 0.4$	$P(s_2 = 0, s_4 = 0) = 0.2$	$P(s_3 = 0) = 0.5$
$P(s_1 = 1) = 0.6$	$P(s_2 = 0, s_4 = 1) = 0.1$	$P(s_3 = 1) = 0.5$
	$P(s_2 = 1, s_4 = 0) = 0.1$	
	$P(s_2 = 1, s_4 = 1) = 0.6$	

same source, a mechanism is devised to identify the possibly ineffective parts of the built probabilistic model. Finally, an optimization algorithm incorporating the proposed technique is described in detail.

5.1 Extended Compact Genetic Algorithm

ECGA uses a product of marginal distributions on a partition of the variables. This kind of probability distribution belongs to a class of probabilistic models known as marginal product models (MPMs). In this kind of model, subsets of variables are modeled jointly, and each subset is considered independent of the other subsets. In this work, the conventional notation is adopted that variable subsets are enclosed in brackets. Table 2 presents an example of MPM defined over four variables: s_1, s_2, s_3, and s_4. In this example, s_2 and s_4 are modeled jointly and each of the three variable subsets ($[s_1]$, $[s_2\ s_4]$, and $[s_3]$) is considered independent of the other subsets. For instance, the probability that this MPM generates a sample $s_1s_2s_3s_4 = 0101$ is calculated as follows,

$$\begin{aligned} P(s_1s_2s_3s_4 = 0101) &= P(s_1 = 0) \times P(s_2 = 1, s_4 = 1) \times P(s_3 = 0) \\ &= 0.4 \times 0.6 \times 0.5\ . \end{aligned}$$

In fact, as its name suggests, a marginal product model represents a distribution that is a "product" of the marginal distributions defined over variable subsets.

In ECGA, both the structure and the parameters of the model are searched and optimized in a greedy fashion to fit the statistics of the selected set of promising solutions. The measure of a good MPM is quantified based on the minimum description length (MDL) principle (Rissanen, 1978), which states that any regularity in a given set of data can be used to compress that data, and the success of a model in capturing those regularities can be measured by the cost of expressing the model and the length of the data compressed according to the model. The MDL principle thus penalizes both inaccurate and complex models, thereby leading to a descriptive yet not overly complicated distribution. Specifically, the search measure is the MPM complexity which is quantified as the sum of model complexity, C_m, and compressed population complexity, C_p. The greedy MPM search first considers all variables as independent and each of them forms a separate variable subset. In each iteration, the greedy search merges two variable subsets that yield the greatest reduction in $C_m + C_p$. This process continues until there is no mergence that can further decrease the combined complexity.

The model complexity, C_m, quantifies the model representation in terms of the number of bits required to store all the marginal distributions. Suppose that the given problem is of length ℓ with binary encoding, and the variables are partitioned into m

subsets each of size k_i, $i = 1 \ldots m$, such that $\ell = \sum_{i=1}^{m} k_i$. Then the marginal distribution corresponding to the ith variable subset requires $2^{k_i} - 1$ frequency counts to be completely specified. Taking into account that each frequency count is of length $\log_2(n+1)$ bits, where n is the population size, the model complexity, C_m, can be defined as

$$C_m = \log_2(n+1) \sum_{i=1}^{m} \left(2^{k_i} - 1\right).$$

The compressed population complexity, C_p, quantifies the suitability of the model in terms of the number of bits required to store the entire selected population (the set of promising solutions picked by the selection operator) under an ideal compression scheme. The compression scheme is based on the partition of the variables. Each subset of the variables specifies an independent "compression block" on which the corresponding partial solutions are optimally compressed. Theoretically, the optimal compression method encodes a message of probability p_i using $-\log_2 p_i$ bits. Thus, taking into account all possible messages, the expected length of a compressed message is $\sum_i -p_i \log_2 p_i$ bits, which is optimal. In information theory (Cover and Thomas, 1991), the quantity $-\log_2 p_i$ is called the *information* of that message and $\sum_i -p_i \log_2 p_i$ is called the *entropy* of the corresponding distribution. Based on information theory, the compressed population complexity, C_p, can be derived as

$$C_p = n \sum_{i=1}^{m} \sum_{j=1}^{2^{k_i}} -p_{ij} \log_2 p_{ij},$$

where p_{ij} is the frequency of the jth possible partial solution to the ith variable subset observed in the selected population.

Note that in the calculation of C_p, it is assumed that the jth possible partial solution to the ith variable subset is encoded using $-\log_2 p_{ij}$ bits. This assumption is fundamental to our technique of identifying the likely ineffective marginal distributions. More precisely, the information of the partial solutions, $-\log_2 p_{ij}$, is a good indicator of inconsistency of statistics gathered from two separate subpopulations.

5.2 Model Pruning

Our technique of identifying the possibly ineffective fragments of a marginal product model is based on the notion that ECGA uses compression performance to quantify the suitability of a probabilistic model for a given set of solutions. The degree of compression is a quite representative metric to the fitness of modeling, because all good compression methods are based on capturing and utilizing the relationships among data (Grünwald, 2007). Thus, if the compression scheme of the MPM built on one set of solutions is incapable of compressing another set of solutions produced under the same condition,[5] then we can speculate that some of the constitutive marginal models observed in the first set of solutions are likely inconsistent with the distribution of the corresponding partial solutions observed in the second set of solutions. Such inconsistency can be seen

[5]For example, if all individuals are produced by sampling the same probabilistic model and selected using the same selection technique under the same pressure.

as a disagreement on the direction of further search. However, under the premise that these two sets of solutions are produced under the same condition, they are supposed to reveal similar directions of further search. Thus, we can reasonably speculate that proper selection pressures were not applied on these partial solutions (causing them to drift toward two different directions), and the true linkage structures on these parts of the problem are not sensible under this condition. Recalling our definition in Section 4, an effective distribution should be capable of providing a good direction for further search and consistent with the linkage structure. Thus, if the aforementioned inconsistency is found, we can expect that with a high probability,[6] the inconsistent marginal models are ineffective. Based on the reasoning, we can perform a systematic checking on the given MPM for the likely ineffective portions.

Suppose that the population of solutions, P, is split into two subpopulations S and T. The model searching is performed on S', the set of promising solutions selected from S. Then we can use the statistics collected from T', the set of solutions selected from T, to examine the built probabilistic model, M. Since each marginal model functions independently, they can be inspected separately. Recall the former description that a variable subset, which specifies a marginal model, is viewed as a "compression block" that encodes each possible partial solution according to the marginal distribution. The jth possible partial solution to the ith variable subset is encoded using $-\log_2 p_{ij}$ bits, where p_{ij} is the frequency of the jth possible partial solution to the ith variable subset observed in S'. Assuming that the given problem is of length ℓ with binary encoding, and there are m variable subsets with each of size k_i, $i = 1 \ldots m$, in the built model M, for the ith marginal model, $i = 1 \ldots m$, we can check whether or not

$$\sum_{j=1}^{2^{k_i}} q_{ij}(-\log_2 p_{ij}) > k_i,$$

where q_{ij} is the frequency of the jth possible partial solution to the ith variable subset collected from T'. If the inequality holds, then the compression scheme employed in the ith marginal model is not a good one for compressing the corresponding partial solutions in T' because it encodes a k_i-bit partial solution to a bit string with an expected length of more than k_i bits. Based on the earlier reasoning, such a condition indicates that the marginal model is likely ineffective because T' does not agree on this part of the model. Otherwise, the scheme should be able to compress the partial solutions in T'.

Further explained from a machine learning perspective (Mitchell, 1997), a good model should generalize well to unseen instances. Otherwise, it captures coincidental regularities among the training data or what it has observed. If model building is performed on the portion where linkage is not sensible from the given set of solutions, it will "overfit" these partial solutions (i.e., take on hitchhikers) that were not subject to proper selection pressures. Consequently, the regularities captured by this part of modeling tend to be inconsistent with the true problem structure. Furthermore, the partial solutions that were not subject to proper selection pressure appear to be random, and such a situation brings about the phenomenon of random drift mentioned in Section 2. By its nature, drift is random, and two different subpopulations tend to drift in two different directions. Thus, we can use the statistical inconsistency between S' and

[6]Because the solutions are generated probabilistically, we cannot be absolutely sure.

Algorithm 1 ECGA with Model Pruning

```
Initialize a population P with n solutions of length ℓ.
while the stopping criteria are not met do
    Evaluate the solutions in P.
    Divide P into two subpopulations S and T at random.
    S' ← apply t-wise tournament selection on S.
    T' ← apply t-wise tournament selection on T.
    M ← build the MPM on S' with greedy search.
    M' ← prune M based on the inconsistency with T'.
    for each remaining marginal distribution D in M' do
        for each solution s = s1 s2 ··· sℓ in P do
            Change the values in s partially by sampling D.
        end for
    end for
end while
```

T' to locate the possible drift portions of the solutions and identify the likely ineffective parts within the whole model. By removing these likely ineffective parts, we can forge a partial but more effective model.

An issue in practice concerning the calculation of the inequality is that sometimes one or more possible partial solutions are absent in the set of selected solutions, leaving $-\log_2 p_{ij}$ undefined because $p_{ij} = 0$. In the present work, we handle this technical problem by assigning a very small value, smaller than $1/n$, to the p_{ij}'s that are zero and normalizing them such that p_{ij}'s sum to 1 (i.e., $\sum_j p_{ij} = 1$).

5.3 Integration

In this section, the optimization process incorporating ECGA and the proposed technique is described. This combination helps ECGA to achieve better performance when a disparate scale exists among different parts of the problem.

The procedure is presented in Algorithm 1. This process starts with initializing a population of solutions. After initialization, the solutions are evaluated, and then the entire population is randomly split into two subpopulations. Selection operations are performed on the two subpopulations separately with the same operator and selection pressure. Model building is performed on one of the subpopulations. The other subpopulation is used to prune the built model using the technique described previously. Finally, all solutions in the population are altered by sampling the remaining marginal distributions, which are considered effective, in the pruned model. These steps are repeated until the stopping criteria are satisfied.

A prominent difference between the above process and the regular EDAs is that the sampling might not include all variables. As introduced in Section 4, the existing solutions are altered by sampling only the marginal distributions surviving the model pruning process. Thus, a solution string might not be entirely modified in an iteration. This technique hence avoids random drift and inaccurate processing of low-fitness building blocks by postponing the processing until sufficient linkage information is available. Similar to the concept proposed by Bosman and Thierens (1999) that linkage information estimated from the selected solutions has to be utilized to recognize

Table 3: Marginal product models before and after pruning when solving a 16-bit exponentially scaled problem with the proposed approach.

Generation	Marginal product model (before and after pruning)	
1	Before	$[s_1\ s_2\ s_3\ s_4]\ [s_5\ s_{13}\ s_{16}]\ [s_6\ s_7\ s_{12}]\ [s_8\ s_{11}]\ [s_9\ s_{10}]\ [s_{14}\ s_{15}]$
	After	$[s_1\ s_2\ s_3\ s_4]$
2	Before	$[\cancel{s_1}]\ [\cancel{s_2}]\ [\cancel{s_3}]\ [\cancel{s_4}]\ [s_5\ s_6\ s_7\ s_8]\ [s_9\ s_{14}]\ [s_{10}\ s_{15}]\ [s_{11}\ s_{13}\ s_{16}]\ [s_{12}]$
	After	$[\cancel{s_1}]\ [\cancel{s_2}]\ [\cancel{s_3}]\ [\cancel{s_4}]\ [s_5\ s_6\ s_7\ s_8]$
3	Before	$[\cancel{s_1}]\ [\cancel{s_2}]\ [\cancel{s_3}]\ [\cancel{s_4}]\ [\cancel{s_5}]\ [\cancel{s_6}]\ [\cancel{s_7}]\ [\cancel{s_8}]\ [s_9\ s_{10}\ s_{11}\ s_{12}]\ [s_{13}\ s_{14}]\ [s_{15}\ s_{16}]$
	After	$[\cancel{s_1}]\ [\cancel{s_2}]\ [\cancel{s_3}]\ [\cancel{s_4}]\ [\cancel{s_5}]\ [\cancel{s_6}]\ [\cancel{s_7}]\ [\cancel{s_8}]\ [s_9\ s_{10}\ s_{11}\ s_{12}]$
4	Before	$[\cancel{s_1}]\ [\cancel{s_2}]\ [\cancel{s_3}]\ [\cancel{s_4}]\ [\cancel{s_5}]\ [\cancel{s_6}]\ [\cancel{s_7}]\ [\cancel{s_8}]\ [\cancel{s_9}]\ [\cancel{s_{10}}]\ [\cancel{s_{11}}]\ [\cancel{s_{12}}]\ [s_{13}\ s_{14}\ s_{15}\ s_{16}]$
	After	$[\cancel{s_1}]\ [\cancel{s_2}]\ [\cancel{s_3}]\ [\cancel{s_4}]\ [\cancel{s_5}]\ [\cancel{s_6}]\ [\cancel{s_7}]\ [\cancel{s_8}]\ [\cancel{s_9}]\ [\cancel{s_{10}}]\ [\cancel{s_{11}}]\ [\cancel{s_{12}}]\ [s_{13}\ s_{14}\ s_{15}\ s_{16}]$

building blocks, we further address that the validity of the linkage information should be confirmed beforehand. In this way, better performance in terms of function evaluations can be achieved if a disparate scale exists among different parts of the problem.

In order to confirm that the proposed method meets its design purpose, Table 3 lists the models before and after pruning when the earlier exponentially scaled problem is solved by Algorithm 1. It can be seen that the proposed approach appropriately removes the ineffective parts during each stage of the optimization process. In order to further illustrate the behavior and effect of the proposed approach, the algorithm is applied to another problem with a different scaling called *overloaded scaling*[7]

$$f(s_1 s_2 \cdots s_{16}) = \sum_{i=0}^{1} f_{\text{trap}_4}(s_{4i+1}s_{4i+2}s_{4i+3}s_{4i+4}) + \sum_{i=2}^{3} \frac{1}{5} f_{\text{trap}_4}(s_{4i+1}s_{4i+2}s_{4i+3}s_{4i+4}) \ ,$$

where $s_1 s_2 \cdots s_{16}$ is a solution string. The overloaded cases are those with two scales, where some subproblems are at the high level and the rest are at the low one. The models before and after pruning when such a problem is solved are shown in Table 4. It can be observed that the proposed method works as expected in splitting the solving process according to the scaling structure. The two subproblems of higher fitness are handled first, and the two subproblems of lower fitness are solved later.

6 Experiments

The experiments are designed to reveal the behavior of the proposed approach in handling sets of problems with different scaling difficulties. Because ECGA is limited in handling overlapped building blocks, we use only test problems that are additively separable. In this study, three bounding models of scalings (Goldberg, 2002) are considered: exponential, power law, and uniform. While the uniform and exponential cases

[7] As mentioned by Goldberg (2002), the word "overloaded" is a reference to the application of this idea in the early messy GA work (Goldberg et al., 1990), where such distributions were used to try to overload or overwhelm the ability of the messy GA to keep all building blocks present through all phases of the process.

Table 4: Marginal product models before and after pruning when solving a 16-bit problem of the overloaded scaling with the proposed approach.

Generation	Marginal product model (before and after pruning)	
1	Before	$[s_1\ s_2\ s_3\ s_4]\ [s_5\ s_6\ s_7\ s_8]\ [s_9\ s_{16}]\ [s_{10}\ s_{14}\ s_{15}]\ [s_{11}\ s_{13}]\ [s_{12}]$
	After	$[s_1\ s_2\ s_3\ s_4]\ [s_5\ s_6\ s_7\ s_8]$
2	Before	$[s_1\ s_2\ s_3\ s_4]\ [s_5\ s_6\ s_7\ s_8]\ [s_9\ s_{13}\ s_{14}]\ [s_{10}\ s_{12}]\ [s_{11}\ s_{15}]\ [s_{16}]$
	After	$[s_1\ s_2\ s_3\ s_4]\ [s_5\ s_6\ s_7\ s_8]$
3	Before	$[s_1]\ [s_2]\ [s_3]\ [s_4]\ [s_5]\ [s_6]\ [s_7]\ [s_8]\ [s_9\ s_{10}\ s_{11}\ s_{12}]\ [s_{13}\ s_{14}\ s_{15}\ s_{16}]$
	After	$[s_1]\ [s_2]\ [s_3]\ [s_4]\ [s_5]\ [s_6]\ [s_7]\ [s_8]\ [s_9\ s_{10}\ s_{11}\ s_{12}]\ [s_{13}\ s_{14}\ s_{15}\ s_{16}]$
4	Before	$[s_1]\ [s_2]\ [s_3]\ [s_4]\ [s_5]\ [s_6]\ [s_7]\ [s_8]\ [s_9\ s_{10}\ s_{11}\ s_{12}]\ [s_{13}\ s_{14}\ s_{15}\ s_{16}]$
	After	$[s_1]\ [s_2]\ [s_3]\ [s_4]\ [s_5]\ [s_6]\ [s_7]\ [s_8]\ [s_9\ s_{10}\ s_{11}\ s_{12}]\ [s_{13}\ s_{14}\ s_{15}\ s_{16}]$

bound the scaling performance of an algorithm at two extremes, the power law cases enable us to see the behavior in between. Based on the different scalings, three sets of test functions are constructed using f_{trap_k} as the elemental function:

$$\text{Exponential: } \sum_{i=0}^{m-1}(k+1)^i f_{\text{trap}_k}(s_{k\times i+1}s_{k\times i+2}\cdots s_{k\times i+k})$$

$$\text{Power law: } \sum_{i=0}^{m-1}(i+1)^3 f_{\text{trap}_k}(s_{k\times i+1}s_{k\times i+2}\cdots s_{k\times i+k})$$

$$\text{Uniform: } \sum_{i=0}^{m-1} f_{\text{trap}_k}(s_{k\times i+1}s_{k\times i+2}\cdots s_{k\times i+k})$$

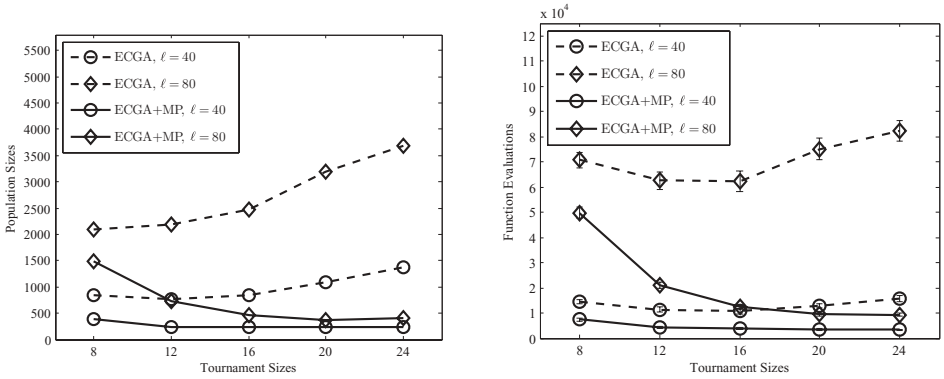
By adopting different scaling setups, we can compare the original ECGA with our approach under different degrees of linkage sensibilities. By varying k and m, we can observe the behavior of the proposed method with respect to different problem and subproblem sizes in a controlled manner. Furthermore, various selection pressures are also taken into consideration to make a more thorough observation.

The purpose of the following experiments is to understand the impact of the proposed method on the *computational resource* (population size and function evaluations) required to solve a problem. Thus, we do not use solution quality as a measure of comparison but treat it as a minimum requirement. More precisely, we use a bisection method (Sastry, 2001) to bound the minimum population size capable of achieving reliable convergence to the optimum. Of course, solution quality can be an important indicator for evaluating a newly invented approach. However, the primary goal of this study is to design a more *economic* approach for solving problems, and the experiments are designed to evaluate the ability of the proposed approach in this aspect.

6.1 Effect of Selection Pressure

This section describes the experiments designed for observing the effect of selection pressure on both the original ECGA and the ECGA combined with the proposed approach. The purpose of these experiments is twofold.

- First, we want to determine the range of selection pressure with which the proposed approach works as we designed. Appropriate selection pressure is quite



(a) Population Sizes (b) Function Evaluations

Figure 1: Empirical results of the proposed method and original ECGA on 40- and 80-bit ($k = 4, m = 10$ and 20) exponentially scaled problems. Five tournament sizes ranging from 8 to 24 were used to observe the behavior of the algorithms under different selection pressures.

important to the proper functioning of our approach because the pruning mechanism is designed according to the statistical inconsistencies between the two subpopulations.

- Second, because the proposed approach will be compared with the original ECGA in the subsequent experiments, in order to make a fair and meaningful comparison, the selection pressure must be set to an appropriate value for the original ECGA to work under good conditions.

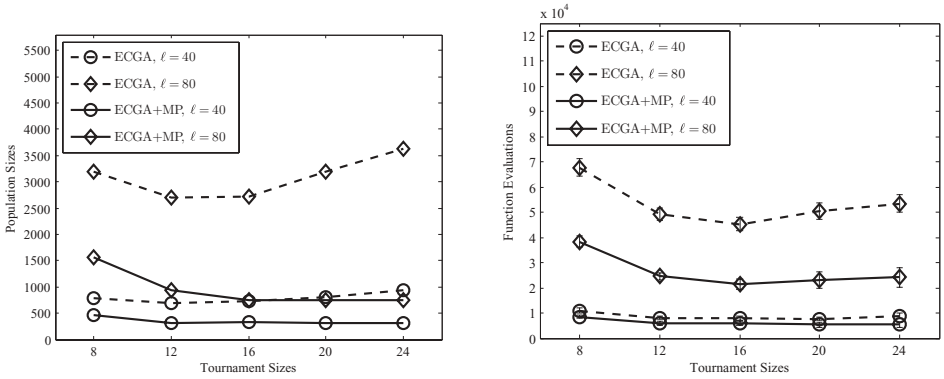
6.1.1 Experimental Settings

Because tournament selection is adopted, the selection pressure is altered by changing the tournament size. We consider tournament sizes ranging from 8 to 24, and the problem instances used to make the observations are of length 40 bits and 80 bits with 4-bit trap functions as subproblems ($k = 4$, $m = 10$ and 20, respectively).

For simplicity, the splitting of population is performed in the way that the two resulting subpopulations are disjoint and of equal size. The stopping criterion is set such that a run is terminated when all solutions in the population converge to the same fitness value. For each tournament size, the minimum required population size is determined by a bisection method (Sastry, 2001) such that on average, $m - 1$ building blocks converge to the correct values in 50 runs for each of the two problem instances.

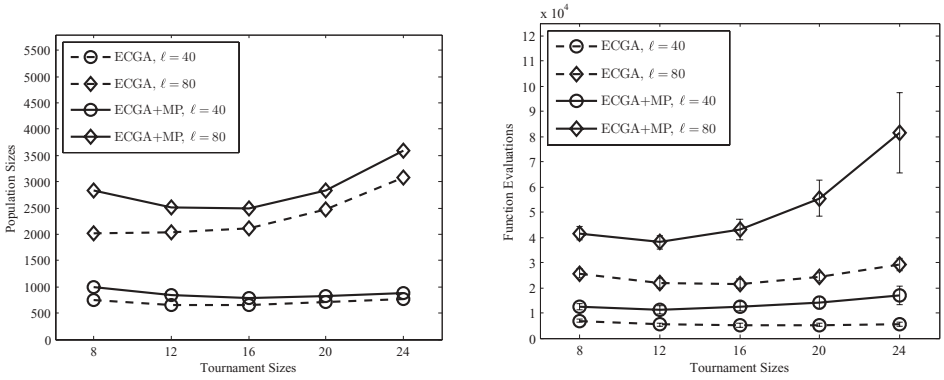
6.1.2 Results and Observations

The results for exponentially, power law, and uniformly scaled problems are presented in Figures 1, 2, and 3, respectively. It can be observed from Figures 1(b), 2(b), and 3(b) that for all three scalings, the original ECGA works best (in terms of the number of function evaluations) under tournament size 12 or 16. Based on that, we will use these two tournament sizes in the following sets of experiments to ensure that the improvement of our approach over the original ECGA is not a result of improper selection pressure. In fact, we also performed experiments using a tournament size of 4, of which the results



(a) Population Sizes (b) Function Evaluations

Figure 2: Empirical results of the proposed method and original ECGA on 40- and 80-bit ($k = 4, m = 10$ and 20) power law scaled problems. Five tournament sizes ranging from 8 to 24 were used to observe the behavior of the algorithms under different selection pressures.



(a) Population Sizes (b) Function Evaluations

Figure 3: Empirical results of the proposed method and original ECGA on 40- and 80-bit ($k = 4, m = 10$ and 20) uniformly scaled problems. Five tournament sizes ranging from 8 to 24 were used to observe the behavior of the algorithms under different selection pressures.

are listed in Table 5. This demonstrates that adopting a lower selection pressure does not yield better performance for ECGA or for our approach.

The results of these experiments give some insights into the pruning mechanism. It can be observed that the appropriateness of a particular selection pressure is related to the linkage sensibility of the problem at hand. This property could cause inconvenience in choosing selection pressure for the algorithm because when dealing with black box optimization, we usually do not have any information about the problem at hand. Fortunately, Figures 1(b), 2(b), and 3(b) also suggest that under tournament sizes ranging

Table 5: Empirical results of the proposed method and original ECGA using a tournament size of 4. Experiments were conducted on 40- and 80-bit problems formed by concatenating 4-bit trap functions with three different scalings. The symbols ℓ, n, and f_{ev} denote problem size, population size, and function evaluations, respectively.

		ℓ	n	f_{ev}	SD f_{ev}
Exponential	ECGA	40	1,719	44,487.72	2,682.02
		80	3,748	187,549.92	5,912.06
	ECGA+MP	40	1,405	37,373.00	2,027.11
		80	4,221	210,881.16	8,568.54
Power law	ECGA	40	1,604	32,946.16	2,105.37
		80	5,507	163,557.90	6,017.21
	ECGA+MP	40	1,248	27,755.52	1,929.44
		80	4,361	141,034.74	5,884.63
Uniform	ECGA	40	1,346	17,228.80	1,489.44
		80	3,479	58,308.04	3,411.61
	ECGA+MP	40	2,181	30,446.76	2,411.81
		80	5,598	100,540.08	5,535.96

from 8 to 16, our approach works better than the original ECGA in the exponentially and power law scaled cases. Under this range of tournament sizes (8 to 16), the behavior of the proposed approach in uniformly scaled cases is relatively stable compared to that under higher selection pressure. This observation demonstrates that for a broad range of selection pressure, the improvement obtained by using the pruning mechanism can be expected in cases of limited linkage sensibility, while in cases for which linkage information is completely sensible, the overhead is relatively stable.

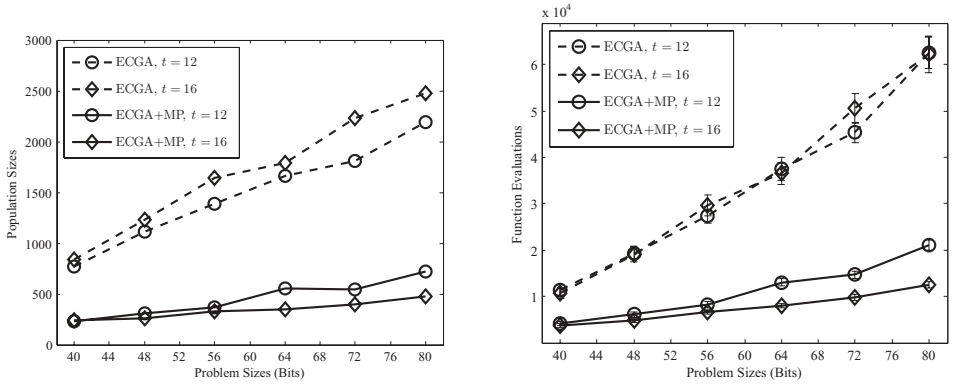
6.2 Impact on Population Requirement with Increasing *m*

This section describes experiments designed to reveal the behavior of the proposed approach when the number of subproblems within a problem is growing (i.e., increasing m while fixing k). In order to illustrate the effectiveness and benefit of adopting the pruning mechanism and to estimate the overhead when it is not needed, the proposed approach will be compared with the original ECGA on three sets of problems with different scaling setups.

6.2.1 Experimental Settings

The problem instances used in this set of experiments are composed of 4-bit trap functions and range from 40 to 80 bits ($k = 4$, $m = 10 \ldots 20$). Two selection pressures are adopted by setting tournament size t to 12 and 16. The reason for using these two tournament sizes is because our approach is compared with the original ECGA, which seems to perform better with $t = 12$ or $t = 16$ according to the previous set of experiments. Otherwise, a question might arise as to whether or not the inferior performance of the original ECGA under some scaling difficulties comes from the inappropriate setting of selection pressure.

As in the previous experiment, the splitting of population is also performed in the way that the two resulting subpopulations are disjoint and of equal size. The stopping criterion is set such that a run is terminated when all solutions in the population converge to the same fitness value. For each problem instance, the minimum required



(a) Population Sizes (b) Function Evaluations

Figure 4: Empirical results of the proposed method compared to the original ECGA on exponentially scaled problems with tournament sizes $t = 12$ and $t = 16$. Problem sizes ranging from 40 to 80 bits ($k = 4$, $m = 10 \ldots 20$) were used to observe the performance of the algorithms.

population size is determined by a bisection method such that on average, $m - 1$ building blocks converge to the correct values in 50 runs.

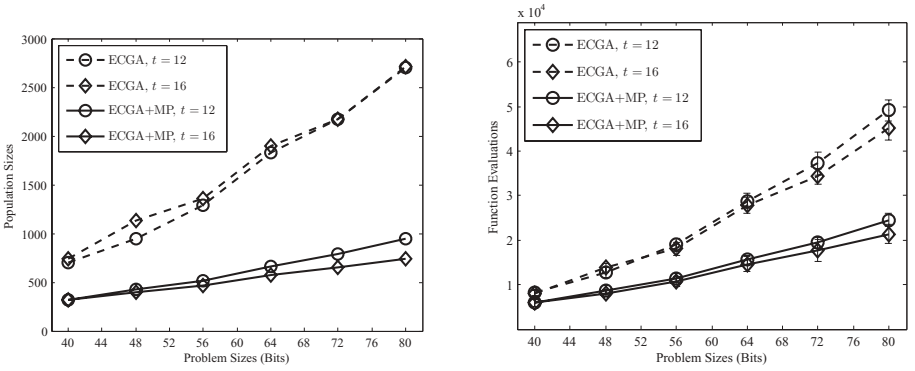
6.2.2 Results and Observations

The empirical results for exponentially scaled problems are shown in Figure 4. The minimum population sizes required by the proposed method are much smaller than the sizes needed by the original ECGA and grow at a relatively slow rate. The same situation is also observed in the function evaluations for which our approach performed remarkably well. This improvement can be explained by the previous discussion on random drift and linkage sensibility presented in earlier sections. If simultaneous detection and processing of all building blocks cannot be achieved, additional costs have to be paid for the inaccurate processing and random drift of subsolutions. By adopting the pruning mechanism, we can save these costs by detecting possibly ineffective partial models and postponing the changes on them until accurate processing can be made.

Figure 5 shows the results for power law scaled problems. The results of the minimum population sizes are similar to those obtained in the previous set of experiments. The proposed method still uses fewer function evaluations, but the differences are reduced. This is because the linkage sensibility of the power law scaled problems is less limited compared to that of the exponentially scaled problems.

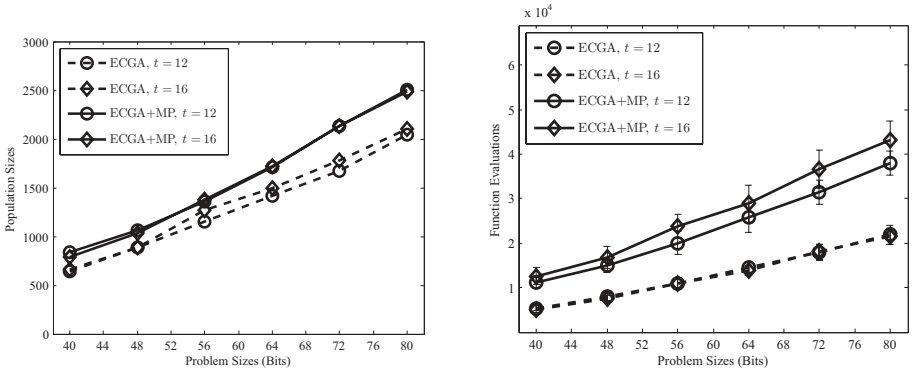
The empirical results for uniformly scaled problems are presented in Figure 6. As expected, the proposed method requires larger population sizes than those needed by the original ECGA. Due to the fact that for uniformly scaled problems, the model building process can correctly identify all building blocks, the verification on the built model may just be useless and wasteful. The results also suggest that the function evaluations used by the proposed method are about twice the number of those needed by the original ECGA.

In order to support the significance of the observations, we have also performed Welch's t-test on the results. For each problem size, a t-test of the null hypothesis that the



(a) Population Sizes (b) Function Evaluations

Figure 5: Empirical results of the proposed method compared to the original ECGA on power law scaled problems with tournament sizes $t = 12$ and $t = 16$. Problem sizes ranging from 40 to 80 bits ($k = 4$, $m = 10 \ldots 20$) were used to observe the performance of the algorithms.



(a) Population Sizes (b) Function Evaluations

Figure 6: Empirical results of the proposed method compared to the original ECGA on uniformly scaled problems with tournament sizes $t = 12$ and $t = 16$. Problem sizes ranging from 40 to 80 bits ($k = 4$, $m = 10 \ldots 20$) were used to observe the performance of the algorithms.

number of function evaluations spent by ECGA and the number of function evaluations spent by the proposed approach are with equal means (against the alternative that the means are not equal) was performed. The significance level was set to 5%, and the respective statistics are listed in Table 6. The resulting statistics suggest that the outcomes of the proposed approach are significantly different from those of the original ECGA for all three scaling setups.

6.3 Impact on Population Requirement with Varying *k*

This section describes the experiments that accompany the previous ones to further demonstrate the performance of the proposed approach. The experiments were designed

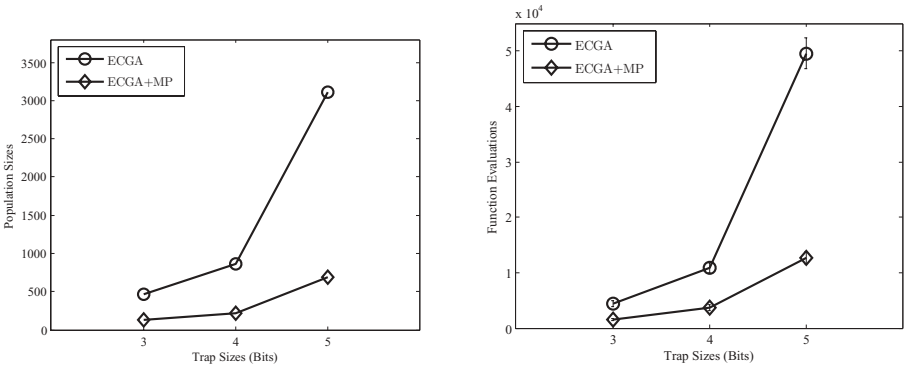
Table 6: Welch's t-test on empirical results presented in Figures 4, 5, and 6. The null hypothesis is that the number of function evaluations spent by ECGA and the number of function evaluations spent by ECGA-MP are with equal means (against the alternative that the means are not equal). The first three rows indicate whether the null hypothesis is rejected, the p-value, and the t-statistics from the tests, respectively. The last row lists whether the number of average function evaluations needed by ECGA-MP is smaller ($<$) or larger ($>$) than the number needed by the original ECGA.

Problem size	40	48	56	64	72	80
	(a) Exponentially scaled cases with tournament size 12					
Reject null	True	True	True	True	True	True
p-value	4.409×10^{-48}	2.137×10^{-60}	2.031×10^{-69}	1.102×10^{-56}	2.053×10^{-70}	5.563×10^{-63}
t-statistics	41.9194	64.0567	82.6409	66.3387	97.2660	82.4819
Comparison	$<$	$<$	$<$	$<$	$<$	$<$
	(b) Exponentially scaled cases with tournament size 16					
Reject null	True	True	True	True	True	True
p-value	1.458×10^{-48}	6.409×10^{-53}	2.149×10^{-54}	1.847×10^{-58}	4.341×10^{-60}	8.518×10^{-58}
t-statistics	40.7834	60.9651	71.7845	81.9204	89.1136	87.8952
Comparison	$<$	$<$	$<$	$<$	$<$	$<$
	(c) Power law scaled cases with tournament size 12					
Reject null	True	True	True	True	True	True
p-value	1.094×10^{-25}	1.208×10^{-33}	5.515×10^{-48}	4.294×10^{-61}	6.05×10^{-53}	1.608×10^{-72}
t-statistics	14.5298	18.4542	29.4004	48.5933	44.0576	63.2243
Comparison	$<$	$<$	$<$	$<$	$<$	$<$
	(d) Power law scaled cases with tournament size 16					
Reject null	True	True	True	True	True	True
p-value	1.582×10^{-22}	6.032×10^{-50}	1.047×10^{-47}	1.717×10^{-59}	3.383×10^{-56}	7.91×10^{-69}
t-statistics	12.7581	30.2641	28.5023	37.5145	38.8386	49.2693
Comparison	$<$	$<$	$<$	$<$	$<$	$<$
	(e) Uniformly scaled cases with tournament size 12					
Reject null	True	True	True	True	True	True
p-value	3.356×10^{-35}	1.23×10^{-39}	4.264×10^{-33}	3.399×10^{-33}	4.006×10^{-45}	4.903×10^{-53}
t-statistic	–25.4683	–26.7928	–23.7365	–22.9905	–29.0505	–33.4524
Comparison	$>$	$>$	$>$	$>$	$>$	$>$
	(f) Uniformly scaled cases with tournament size 16					
Reject null	True	True	True	True	True	True
p-value	1.126×10^{-34}	1.776×10^{-33}	8.802×10^{-40}	6.649×10^{-32}	3.684×10^{-38}	4.062×10^{-44}
t-statistic	–25.7066	–25.9356	–31.5460	–25.0263	–28.6037	–34.0405
Comparison	$>$	$>$	$>$	$>$	$>$	$>$

to observe the behavior of the proposed approach when the size of the constitutive subproblem changes (i.e., varying k while fixing m). As in the previous set of experiments, the original ECGA will also be tested for comparison.

6.3.1 Experimental Settings

In contrast to the previous set of experiments, we use trap functions of different sizes to form our test problems. While the size of constituting subproblem varies, the number of the subproblems remains fixed. The problem instances are constructed by concatenating



(a) Population Sizes (b) Function Evaluations

Figure 7: Empirical results of the proposed method compared to the original ECGA for exponentially scaled problems composed of subproblems of sizes 3, 4, and 5 ($k = 3$, 4, and 5). In this experiment, tournament size $t = 16$ was used and the number of subfunctions forming the test problems was fixed at 10 (i.e., $m = 10$).

10 trap functions of size 3, 4, or 5 ($k = 3$, 4, or 5, $m = 10$). Tournament size $t = 16$ is used in this set of experiments.

As in the previous experiments, the splitting of the population is also performed so that the two resulting subpopulations are disjoint and of equal size. The stopping criterion is set such that a run is terminated when all solutions in the population converge to the same fitness value. For each problem instance, the minimum required population size is determined by a bisection method such that on average, $m - 1$ building blocks converge to the correct values in 50 runs.

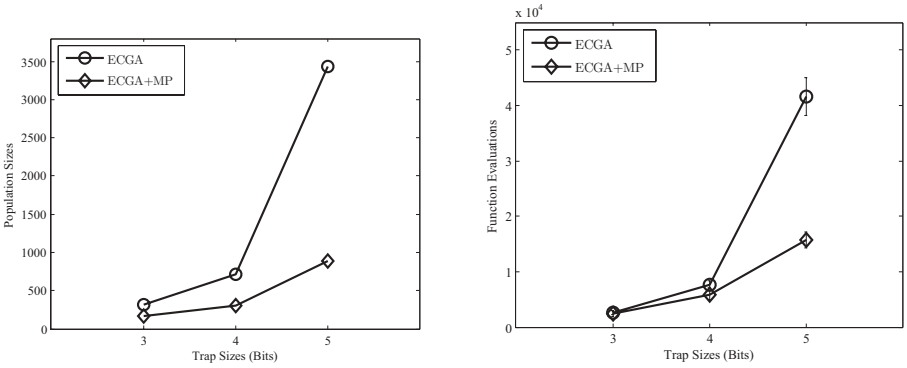
6.3.2 Results and Observations

The results for exponentially and power law scaled problems are presented in Figures 7 and 8, respectively. It can be observed that for these three different subproblem sizes, the proposed approach uses smaller population sizes and fewer function evaluations to solve the test problems. Furthermore, the degree of improvement over the original ECGA seems to increase with the size of the constituting subproblems. As can be seen in the problems composed of 5-bit trap functions, the pruning mechanism achieves great savings in function evaluations compared to the original ECGA.

On the other hand, for the uniformly scaled problems, our approach still requires larger population sizes than what was needed by the original ECGA. This result is expected, as it can be conjectured that in solving uniformly scaled problems, the verification on the built model may be useless and wasteful. A further observation is that these results seem to be consistent with what we observed in the previous set of experiments in which the function evaluations used by the proposed method are about twice the number needed by the original ECGA.

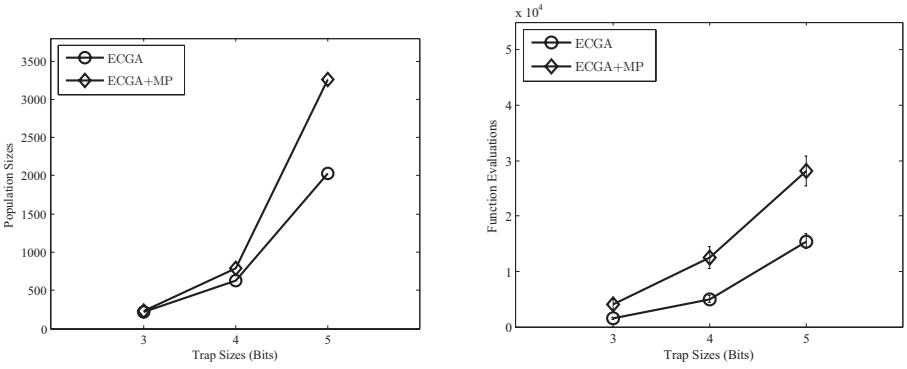
6.4 Building versus Verifying

This section describes the sets of experiments on the proposed method to reveal the change in performance when different splitting ratios of the two subpopulations are



(a) Population Sizes (b) Function Evaluations

Figure 8: Empirical results of the proposed method compared to the original ECGA for power law scaled problems composed of subproblems of sizes 3, 4, and 5 ($k = 3$, 4, and 5). In this experiment, tournament size $t = 16$ was used and the number of subfunctions forming the test problems was fixed at 10 (i.e., $m = 10$).



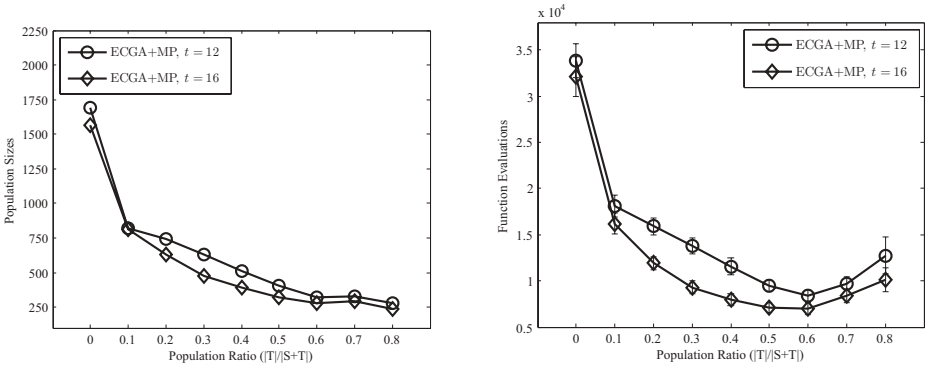
(a) Population Sizes (b) Function Evaluations

Figure 9: Empirical results of the proposed method compared to the original ECGA for uniformly scaled problems composed of subproblems of sizes 3, 4, and 5 ($k = 3$, 4, and 5). In this experiment, tournament size $t = 16$ was used and the number of subfunctions forming the test problems was fixed at 10 (i.e., $m = 10$).

adopted. It presents the experimental results to illustrate the behavior under different scalings. The purpose for performing these experiments is twofold:

- First, we would like to observe how the splitting ratio is related to the scaling or linkage sensibility of a problem.
- Second, we wish to empirically study the change in performance obtained from decreasing or increasing the proportion of population for checking the model.

It is important in practice to spend function evaluations wisely. Since using too large a proportion of the population for pruning may result in a waste of resources, it should



(a) Population Sizes

(b) Function Evaluations

Figure 10: Empirical results of the proposed method for a 60-bit exponentially scaled problem with different splitting ratios between the two subpopulations. The splitting ratio ($|T|/|S+T|$) ranging from 0.0 (ECGA without pruning) to 0.8 was used to observe the change in performance of the proposed approach.

be estimated to what degree the expense on checking the built model yields savings, and how the scaling of the problem is related to this matter.

6.4.1 Experimental Settings

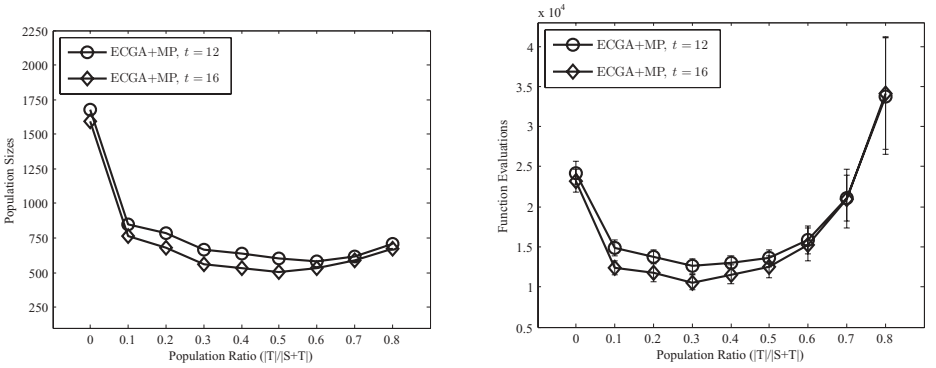
The problem instances used in this set of experiments were of 60 bits formed by concatenating 4-bit trap functions ($k = 4$, $m = 15$). The splitting ratio ($|T|/|S+T|$) ranged from 0.0 to 0.8. The ratio 0.0 represents the result of running the original ECGA (without pruning), which serves as a baseline. Two selection pressures were adopted by setting tournament size t to 12 and 16.

As in the previous experiments, the stopping criterion is set such that a run is terminated when all solutions converge to the same fitness value. For each splitting ratio, the minimum required population size was determined by a bisection method such that on average, $m - 1$ building blocks converge to the correct values in 50 runs.

6.4.2 Results and Observations

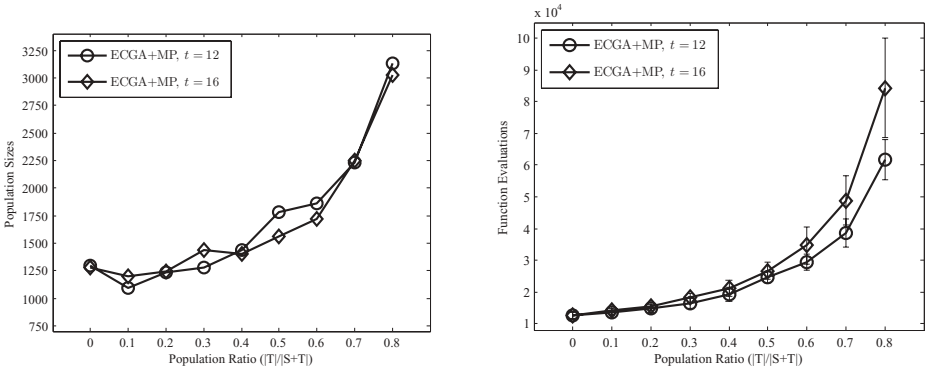
The empirical results for exponentially scaled problems are presented in Figure 10. For both tournament sizes, the required population size decreases as the splitting ratio increases. However, the number of generations increases with the splitting ratio. The combined effect is that the minimum required function evaluation is obtained when the splitting ratio is 0.6, and the required function evaluation grows when the splitting ratio either increases or decreases.

Figure 11 shows the results for power law scaled problems. In contrast to the previous case, the required population size does not strictly decrease with the increment of the splitting ratio. The population size first decreases as the splitting ratio grows and then hits a turning point at 0.5 ($t = 16$) or 0.6 ($t = 12$). Similar to the exponentially scaled case, the number of generations increases with the splitting ratio. The combined effect is that the number of function evaluations first decreases and then increases. For both tournament sizes, the minimum is obtained when the splitting ratio equals 0.3.



(a) Population Sizes (b) Function Evaluations

Figure 11: Empirical results of the proposed method for a 60-bit power law scaled problem with different splitting ratios between the two subpopulations. The splitting ratio ($|T|/|S+T|$) ranging from 0.0 (ECGA without pruning) to 0.8 was used to observe the change in performance of the proposed approach.



(a) Population Sizes (b) Function Evaluations

Figure 12: Empirical results of the proposed method for a 60-bit uniformly scaled problem with different splitting ratios between the two subpopulations. The splitting ratio ($|T|/|S+T|$) ranging from 0.0 (ECGA without pruning) to 0.8 was used to observe the change in performance of the proposed approach.

Figure 12 shows the results for uniformly scaled problems. As expected, Figures 12(a) and 12(b) both share a common pattern in which the population size and the number of function evaluations increase with the splitting ratio. This is because in the uniformly scaled case, the linkage is always completely sensible, and there is no need to verify or prune the built probabilistic model.

These experimental results demonstrate that under different scaling setups, the behavior of the proposed approach corresponding to the splitting ratio varies differently. The empirical results suggest that if the given problem is evidently with distinguishable prominence among the constituting subproblems, using higher splitting ratios will yield

better performance. Lower ratios are more suitable if the problem at hand is composed of subproblems with roughly equal salience.

Another insight provided by this set of experiments is that reducing the size of the proportion of population spent on the pruning mechanism can considerably improve the performance. As shown in Figures 10(b) and 11(b), compared to the original ECGA (splitting ratio = 0.0 in the figures), significant performance gain can be obtained by using a mere 10% of the population to validate the built model. On the other hand, Figure 12(b) also demonstrates that using this small percentage of population on the pruning mechanism will not bring serious overhead to the overall performance.

6.5 Splitting Ratio versus Subproblem Size

This section describes the experiments extending the previous set of experiments for observing the interaction between the splitting ratio and the performance. The focus of this set of experiments is to study the effect of different splitting ratios when the size of the constitutive subproblem changes (i.e., varying k while fixing m). Our main purpose is to see whether the result of adopting a particular splitting ratio changes significantly when the complexity of the problem varies. Furthermore, we want to empirically examine whether or not the improvement of using just 10% of the population to validate the built model is still prominent for different sizes of the constitutive subproblems.

6.5.1 Experimental Settings

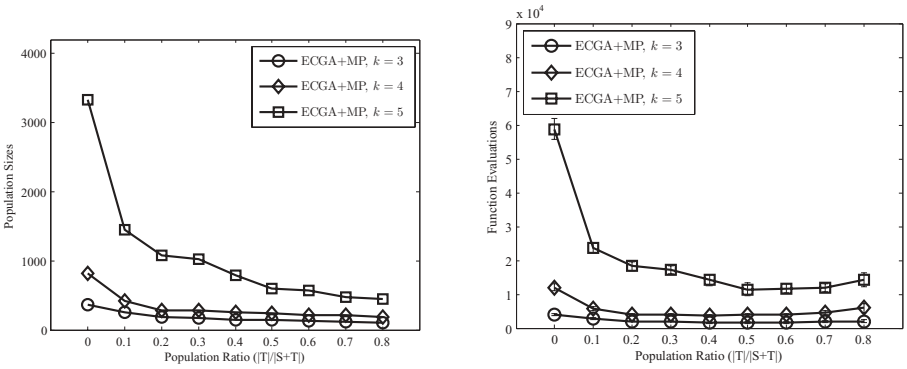
In this set of experiments, we use trap functions of different sizes to construct our test problems. While the size of constitutive subproblems varies, the number of the subproblems forming the test problems remains the same. The problem instances are built by concatenating 10 trap functions of sizes 3, 4, or 5 ($k = 3$, 4, or 5, $m = 10$). Tournament size $t = 16$ is adopted in this set of experiments.

As in the previous set of experiments, the splitting ratio ($|T|/|S + T|$) ranges from 0.0 to 0.8. The ratio 0.0 represents the result of running the original ECGA (without pruning), which serves as a baseline. The stopping criterion is set such that a run is terminated when all solutions in the population converge to the same fitness value. For each splitting ratio, the minimum required population size was determined by a bisection method such that on average, $m - 1$ building blocks converge to the correct values in 50 independent runs.

6.5.2 Results and Observations

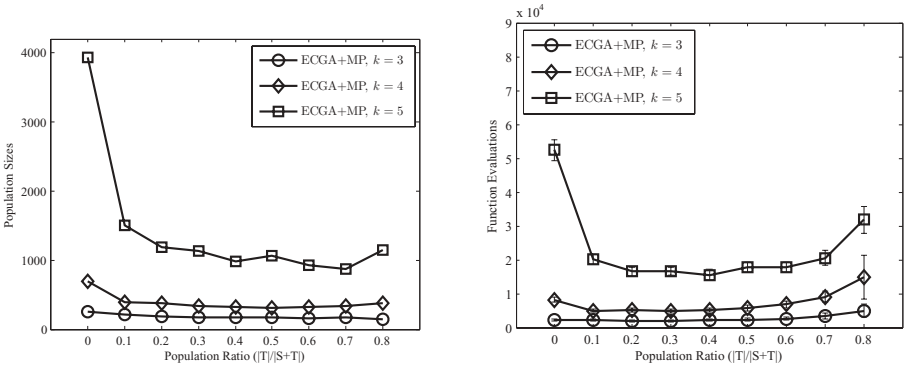
The results for exponentially, power law, and uniformly scaled problems are presented in Figures 13, 14, and 15, respectively. We can see that the result of adopting a particular splitting ratio does not change significantly relative to other splitting ratios for all three subproblem sizes. It can also be observed that several kinds of behavior similar to what we have seen in the previous experiments are presented in these results. For the uniformly scaled problems, the results presented in Figure 15(b) show a similar pattern to what is observed in the previous set of experiments in which the number of function evaluations increases with the splitting ratio. In addition, similar to the results from the previous set of experiments, we can see that using a small percentage (10%) of population on the pruning mechanism does not bring serious overhead to the overall performance for all three subproblem sizes.

On the other hand, for exponentially and power law scaled problems, the greatest improvements are obtained when using 10% of the population to validate the built



(a) Population Sizes (b) Function Evaluations

Figure 13: Empirical results of the proposed method using different splitting ratios ($|T|/|S+T|$) for exponentially scaled problems composed of subproblems of sizes 3, 4, or 5 ($k = 3$, 4, or 5). In this experiment, tournament size $t = 16$ was used and the number of subfunctions forming the test problems was fixed at 10 (i.e., $m = 10$)



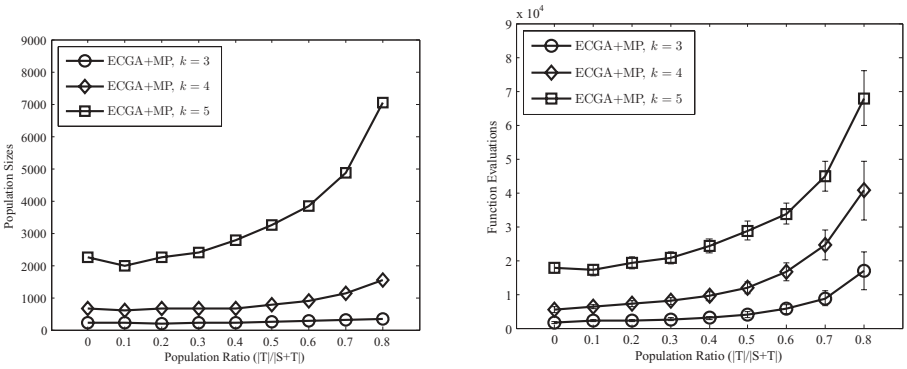
(a) Population Sizes (b) Function Evaluations

Figure 14: Empirical results of the proposed method using different splitting ratios ($|T|/|S+T|$) for power law scaled problems composed of subproblems of sizes 3, 4, or 5 ($k = 3$, 4, or 5). In this experiment, tournament size $t = 16$ was used and the number of subfunctions forming the test problems was fixed at 10 (i.e., $m = 10$)

model. Furthermore, similar to what we have observed in the experiments described in Section 6.3, the degree of improvement over the original ECGA (splitting ratio = 0.0) increases with the size of the constitutive subproblem.

7 Discussion

We utilized the existence of disparate scales in problems to create a controlled experimental environment in order to study the situation in which complete, accurate linkage information may or may not be available for the estimation of distribution algorithms.



(a) Population Sizes

(b) Function Evaluations

Figure 15: Empirical results of the proposed method using different splitting ratios ($|T|/|S+T|$) for uniformly scaled problems composed of subproblems of sizes 3, 4, or 5 ($k = 3, 4$, and 5). In this experiment, tournament size $t = 16$ was used and the number of subfunctions forming the test problems was fixed at 10 (i.e., $m = 10$)

According to the obtained results shown in Figures 4(b) and 5(b), the proposed approach does improve the original ECGA for the test problems where disparate scales exist among building blocks. In this section, we discuss some interesting aspects of the proposed approach and possible extensions of this work.

7.1 Overhead in Uniformly Scaled Problems

The empirical results presented in Figure 6(b) show that for the uniformly scaled cases, the proposed approach uses nearly twice as many function evaluations as the original ECGA does. We speculate that this double expenditure is a general property of the proposed approach when dealing with uniformly scaled problems.

This speculation can be explained through a reverse thinking on a hypothetical situation described as follows. Suppose that given a uniformly scaled problem, the original ECGA with appropriate selection pressure needs a population of size n to handle that problem properly. Now, consider adopting the proposed approach to handle the same problem. If we use a population of size $2n$, then in our algorithm, the entire population will be divided into two subpopulations of size n, assuming that the splitting of population is disjoint and of equal size. If the original ECGA is capable of detecting the accurate problem structure with a population of size n, then in our algorithm, a subpopulation of size n will also do the job. In the ideal case, there will be no statistical inconsistency between the built model and the set of promising solutions selected from the second subpopulation. As a result, we waste half of the population for the use of pruning which causes the extra cost compared to the original ECGA.

In order to support the inference, we performed an experiment based on the scenario just described. Table 7 lists some of the empirical results obtained from the experiments described in Section 6.2. This table shows that for 40-bit and 80-bit uniformly scaled problems formed by concatenating 4-bit trap functions, the original ECGA needs populations of sizes 646 and 2,042, respectively, to solve the given problem. Based on these results, we used population sizes that are twice that to run our approach. The results are

Table 7: Empirical results of the original ECGA using tournament size 12. Experiments were conducted on 40-bit and 80-bit uniformly scaled problems formed by concatenating 4-bit trap functions. The symbols ℓ, n, g, and f_{ev} denote problem size, population size, generation, and function evaluations, respectively.

	ℓ	n	g	$SD\ g$	f_{ev}	$SD\ f_{ev}$
ECGA	40	646	8.36	0.92	5,400.56	594.65
	80	2,042	10.72	1.01	21,890.24	2,064.38

Table 8: Empirical results of the proposed approach using a tournament size of 12. Experiments were conducted on 40-bit and 80-bit uniformly scaled problems formed by concatenating 4-bit trap functions. The symbols ℓ, $2n$, g, and f_{ev} denote problem size, twice the population size required by the original ECGA, generation, and function evaluations, respectively.

	ℓ	$2n$	g	$SD\ g$	f_{ev}	$SD\ f_{ev}$
ECGA+MP	40	1,292	9.24	0.89	11,938.08	1,154.42
	80	4,084	10.58	0.70	43,208.72	2,868.90

listed in Table 8. It can be observed that the function evaluations spent by the proposed approach for 40-bit and 80-bit problems are about twice the amount of the original ECGA needed in each case.

Although the inference together with the empirical validation can serve as an intuitive explanation, it cannot fully explain the results presented in Section 6.2. As illustrated in Figure 6(a), the minimum population sizes needed by the proposed method is not exactly twice that required by the original ECGA. In fact, the numbers are much lower than twice what is needed by the original ECGA. On the other hand, our approach uses more generations compared to the original ECGA because the subpopulation for model building was not sufficiently large for all problem structures to be detected properly in the beginning of the process. In this situation, the processing was slowed down because the pruning mechanism removed certain parts of the model exhibiting statistical inconsistencies. As a consequence, the originally expected simultaneous processing of building blocks was not fully achieved and delay of convergence occurred. Nevertheless, spending more generations seems to yield an equivalent use of function evaluations as the hypothetical case described above. We think that the pruning mechanism introduces an additional interaction between population size and generations. Further empirical or theoretical studies are needed to investigate such an interaction.

7.2 A Deeper Look at the Pruning Criterion

This section provides a more detailed elaboration on the adequacy of the proposed pruning metric. To start this discussion, let

$$\lambda_i = \sum_{j=1}^{2^{k_i}} q_{ij}(-\log_2 p_{ij})$$

which is the quantity to be examined by the pruning criterion (i.e., whether $\lambda_i \geq k_i$). Based on λ_i, we can reformulate the issue of adequacy more concisely as "is it possible

that p_i is not effective but $\lambda_i \leq k_i$ or p_i is effective but $\lambda_i \geq k_i$?" To elaborate on this, we have to separate the discussion into two cases:

1. p_i is not effective and $\lambda_i \leq k_i$, and
2. p_i is effective and $\lambda_i \geq k_i$.

For the first case, if p_i is not effective and its ineffectiveness is caused by drift, it is possible that $\lambda_i \leq k_i$ if the set of solutions on which q_i is estimated also drifts in the same direction. However, by its nature, drift is random, and two different sets of solutions tend to drift in two different directions. Thus, we can expect the chances of this situation to be small and our empirical results also support this conjecture.

For the second case, if p_i is effective, that is, it provides a good direction for further search, then the (sub)solutions on which p_i is estimated must be subject to proper selection pressure. Based on the premise that these two sets of solutions are produced under the same condition, we can expect that the (sub)solutions on which q_i is estimated should also be subject to the same pressure. In this case, if these two sets of solutions are produced under the same condition and the selection pressure is properly applied (i.e., no drifting), it would be unreasonable to see inconsistencies between p_i and q_i (i.e., $\lambda_i \geq k_i$.) However, the above discussion is based on the assumption that the population size is sufficiently large. If the population size if not sufficiently large, inconsistencies tend to be observed because there are too few samples to reveal the true statistical property.

Using the above discussion, we can further analyze what would happen if we use more than one set of solutions to prune the built model. This kind of technique is used frequently in machine learning research to assess the performance of a learning algorithm, in which multiple reserved subsets of testing instances are examined. Extending from the above discussion, let P be the probability of the above case 1 (p_i is not effective and $\lambda_i \leq k_i$) and use r sets of solutions for validating the built model. Then the probability that we cannot detect the ineffectiveness of a marginal model will be P^r, for it is tested independently on r different sets, which is smaller than the probability of using only one set of solutions (i.e., P). However, in this paper, we focused on the baseline behavior of the proposed approach, since we know that employing a larger r should yield better performance and should also incur higher costs.

7.3 Pruning Network-Based Probabilistic Models

In this work, we have introduced a technique to prune a given marginal product model based on the statistics collected from a reserved set of solutions. It is possible to extend the fundamental idea and concept to design pruning mechanisms for other EDAs. For example, consider the EDAs that use network-based probabilistic models with the Bayesian information criterion (BIC; Schwarz, 1978) as the model scoring metrics, such as EBNA (Etxeberria and Larrañaga, 1999) and a variant of BOA (Pelikan et al., 2001). In the binary case, BIC assigns a given network structure B of ℓ variables a score

$$\begin{aligned} S(B) &= \sum_{i=1}^{\ell} \left(-n \times H(X_i|\Pi_i) - 2^{|\Pi_i|} \frac{\log_2 n}{2} \right) \\ &= -\sum_{i=1}^{\ell} n \times H(X_i|\Pi_i) - \sum_{i=1}^{\ell} 2^{|\Pi_i|} \frac{\log_2 n}{2} , \end{aligned}$$

where X_i, $i = 1 \ldots \ell$, are variables, $H(X_i|\Pi_i)$ is the conditional entropy of X_i given its parent Π_i in the network, and n is the population size. The conditional entropy $H(X_i|\Pi_i)$ is given by

$$H(X_i|\Pi_i) = -\sum_{x_i,\pi_i} p(x_i,\pi_i)\log_2 p(x_i|\pi_i),$$

where $p(x_i,\pi_i)$ is the probability of instances with $X_i = x_i$, $\Pi_i = \pi_i$, and $p(x_i|\pi_i)$ is the conditional probability of instances with $X_i = x_i$ given that $\Pi_i = \pi_i$.

The term $\sum_{i=1}^{\ell} n \times H(X_i|\Pi_i)$ provides the same functionality as the compressed population complexity (C_p) in ECGA because $H(X_i|\Pi_i)$ denotes the average number of bits required to store a value of X_i with compression given the information of Π_i. Thus, we can check whether or not variable X_i should be pruned away by using the following inequality

$$-\sum_{x_i,\pi_i} q(x_i,\pi_i)\log_2 p(x_i|\pi_i) > 1,$$

where $q(x_i,\pi_i)$ is the frequency of $X_i = x_i$, and $\Pi_i = \pi_i$ is observed in the set of solutions selected from the reserved subpopulation. Using the idea described in Section 5.2, if this inequality holds, X_i should be removed because it encodes a one-bit partial solution to a bit string with an expected length of more than one bit.

However, despite the similarities in ideas, some technical complications remain to be overcome before we can finish the design of a pruning mechanism for network-based probabilistic models. For instance, what if a variable which we intend to prune is a parent node of some other variables? In summary, pruning network-based probabilistic models is potentially feasible but requires further investigation.

8 Summary and Conclusions

This paper reviewed previous studies on EDAs and scaling difficulties. It then illustrated how the scaling difficulty shadows the EDA ability in recognizing building blocks. Following that, a notion called *linkage sensibility* was introduced to describe the observation, and we used the term *sensible linkage* to refer to the problem structures that can be extracted by inspecting only the set of selected solutions. Based on this concept, we briefly defined the effectiveness of distributions estimated by probabilistic model building and proposed a general approach to achieve more effective modeling. Finally, an implementation of the proposed approach on ECGA was introduced and experiments were done using several test functions of different scaling difficulties. In this section, we briefly summarize the major results derived from this work and outline the possible future extensions of this research.

8.1 Contributions

In this work, we have shown that the underlying facilities for EDAs to solve problems efficiently and reliably do not work as expected when the problem at hand is composed of subparts of unequal fitness contributions. More specifically, under this situation, the model built from the selected solutions cannot fully reflect the true problem structures. Although there are previous studies and discussion on the parameter selection (Pelikan et al., 2002; Lima and Lobo, 2004; Pelikan and Lin, 2004; Yu et al., 2007), selection

mechanisms (Lima et al., 2007, 2008), and model building algorithms (Echegoyen et al., 2007) related to the model accuracy, we consider the conditions discussed in this paper to be more fundamental and closer to the problem nature than those other factors. This is because in our discussion, the condition that suppresses modeling accuracy is embedded in the problem inherently. For some situations, we can reasonably fine-tune algorithmic parameters or select between alternative model building approaches; however, in general we do not have a way to remove a property (e.g., scaling) that exists inherently within the problem to improve modeling accuracy.

Alongside the modeling inaccuracy is the phenomenon of random drift. In a finite population, the selection process can cause convergence to some subsolutions for reasons other than the fitness contribution of these subsolutions. The converged subsolutions might be hitchhikers that appear with other high quality building blocks in selected solutions, or just a result of stochastic errors of sampling due to small population accumulated over generations. As demonstrated in the earlier sections with problems having disparate scalings among subparts, a problem property (e.g., scaling) can cause drift in population as well as making some parts of the problem structure undetectable to the model building process. This situation is usually resolved by increasing the population size to prevent convergence caused by random drift. In contrast, our approach handles this situation by relating these two co-occurring events and by using a pruning mechanism to avoid building models on, and sampling from, the possible drift portions. In this way, we effectively save the cost that we originally have to pay for the maintenance of diversity by using larger populations.

Empirical results show that our approach improves the original ECGA in cases where disparate scales exist among constitutive subproblems and in the uniformly scaled problems (i.e., all the constitutive subproblems have the same fitness contribution), the overhead of using the proposed pruning mechanism is about the amount of function evaluations spent by the original ECGA. The experimental results further suggest that this constant overhead in uniformly scaled cases is not affected by the size of the subfunction (i.e., k) forming the problem, and the improvement in nonuniformly scaled cases seems to increase with the size of the problem. Moreover, we also demonstrated through experiments that in the nonuniformly scaled cases, a small proportion (10%) of population spent on the pruning mechanism can greatly reduce the amount of required function evaluations compared to that spent by the ECGA without pruning.

The experiments with different scaling setups also led to another consideration: that whether uniformly or near-uniformly scaled problems adopted by many previous studies are suitable to fully test the performance of an algorithm designed for solving black box optimization problems. In our humble opinion, presuming a black box optimization problem to be handled is uniformly scaled is too strong an assumption, because there will be no information to confirm this assumption prior to the application of the algorithm. Thus, we believe that in order to generalize beyond the assumption that all subproblems are uniformly scaled, the constant-time overhead for solving the uniformly scaled cases is a reasonable tradeoff.

In addition, several efficiency enhancement techniques for EDAs (Sastry and Goldberg, 2004; Sastry et al., 2004, 2005, 2006; Lima et al., 2005, 2006) rely on the structure information delivered from the probabilistic models. Their good functioning crucially depends on the structural accuracy of the built models. Thus, it is conceivable that if the built model does not properly capture the true structure of the underlying problem, the model-based enhancement mechanism will not fully work as expected. Furthermore, as we demonstrated in this paper, the condition that hinders the model building

algorithm from constructing models that truly reflect the problem structure may be an inherent property of the underlying problem (e.g., different scales among constitutive subproblems). Thus, we think that adapting pruning mechanisms will provide a more appropriate circumstance for the model-based enhancement techniques to work.

8.2 Future Work

In this paper, we demonstrated a pruning mechanism design and its integration into ECGA. It may also serve as a basis for developing other techniques for more efficient and robust optimization. Some possible extensions of this work are outlined as follows.

First of all, the immediate direction is to design pruning mechanisms for other EDAs. As illustrated in Section 7.3, we can extend the pruning metric described in this paper to handle network-based models with a Bayesian information criterion. However, a pruning mechanism for network-based models requires more than that. We also need to consider the possible disruption of variable dependencies after pruning a particular variable. The simplest solution is to consider only those variables that are not depended upon by other variables as possible candidates for pruning. However, the validity of such an approach requires further investigation. A more promising yet more sophisticated approach is to first identify the tightly related components (e.g. cliques or strongly connected subgraphs) in the model and then process each component as a unit which is similar to how we process the marginal product models in this work.

Another direction for future research is to assist efficiency enhancement techniques that use the information contained in the built model. As described previously in Section 8.1, some model-based efficiency enhancement techniques for EDAs crucially rely on the structural accuracy of the probabilistic models. However, most of those studies implicitly assume the information contained in the given population is sufficient for learning accurate model structures. As demonstrated in the previous sections by nonuniformly scaled problems, this assumption does not always hold. From this perspective, incorporating pruning mechanisms to preprocess the built model for these enhancement techniques is a promising direction for designing more robust approaches.

From an abstract point of view, this work also demonstrates an instance of a new class of techniques operating on built models to control, adapt, or regulate the optimization process. Another example based on this viewpoint is the termination criterion proposed by Ocenasek (2006) which uses an entropy-based measurement to evaluate the built model for detecting an appropriate stopping point. According to the information collected in the model, we can gain better control over the process compared to the conventional evolutionary algorithms. Such an idea may be carried over to other designs of EDAs so that more robust and efficient optimization can be realized.

Acknowledgments

The work was supported in part by the National Science Council of Taiwan under Grant NSC-98-2221-E-009-072. The authors are grateful to the National Center for High-performance Computing for computer time and facilities.

References

Baluja, S. (1994). Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Tech. Rep. CMU-CS-94-163, Carnegie Mellon University, Pittsburgh, Pennsylvania.

Baluja, S., and Davies, S. (1997). Using optimal dependency-trees for combinational optimization. In *Proceedings of the 14th International Conference on Machine Learning (ICML '97)*, pp. 30–38.

Bosman, P. A. N., and Thierens, D. (1999). Linkage information processing in distribution estimation algorithms. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference 1999 (GECCO-99)*, Vol. 1, pp. 60–67.

Chen, Y.-p., and Goldberg, D. E. (2005). Convergence time for the linkage learning genetic algorithm. *Evolutionary Computation*, 13(3):279–302.

Cover, T. M., and Thomas, J. A. (1991). *Elements of information theory*. New York: Wiley-Interscience.

De Bonet, J., Isbell, C., and Viola, P. (1997). MIMIC: Finding optima by estimating probability densities. In M. C. Mozer, M. I. Jordan, and T. Petsche (Eds.), *Advances in Neural Information Processing Systems*, Vol. 9, pp. 424–430.

Deb, K., and Goldberg, D. E. (1993). Analyzing deception in trap functions. In *Foundations of Genetic Algorithms 2*, pp. 93–108.

Deb, K., and Goldberg, D. E. (1994). Sufficient conditions for deceptive and easy binary functions. *Annals of Mathematics and Artificial Intelligence*, 10(4):385–408.

Echegoyen, C., Lozano, J. A., Santana, R., and Larrañaga, P. (2007). Exact Bayesian network learning in estimation of distribution algorithms. In *Proceedings of the 2007 IEEE Congress on Evolutionary Computation (CEC 2007)*, pp. 1051–1058.

Etxeberria, R., and Larrañaga, P. (1999). Global optimization using Bayesian networks. In A. O. Rodriguez, M. S. Ortiz, and R. S. Hermida (Eds.), *Proceedings of the Second Symposium on Artificial Intelligence (CIMAF-99)*, pp. 332–339.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Reading, MA: Addison-Wesley.

Goldberg, D. E. (2002). *The design of innovation: Lessons from and for competent genetic algorithms*. Dordrecht, The Netherlands: Kluwer Academic Publishers.

Goldberg, D. E., Deb, K., and Clark, J. H. (1992). Genetic algorithms, noise, and the sizing of populations. *Complex Systems*, 6(4):333–362.

Goldberg, D. E., Deb, K., and Korb, B. (1990). Messy genetic algorithms revisited: Studies in mixed size and scale. *Complex Systems*, 4(4):415–444.

Goldberg, D. E., and Rudnick, M. (1991). Genetic algorithms and the variance of fitness. *Complex Systems*, 5(3):265–278.

Grünwald, P. (2007). *The minimum description length principle*. Cambridge, MA: MIT Press.

Harik, G. R. (1997). *Learning gene linkage to efficiently solve problems of bounded difficulty using genetic algorithms*. PhD thesis, University of Michigan, Ann Arbor.

Harik, G. R. (1999). Linkage learning via probabilistic modeling in the ECGA. IlliGAL Report No. 99010, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign.

Harik, G. R., Lobo, F. G., and Goldberg, D. E. (1999). The compact genetic algorithm. *IEEE Transactions on Evolutionary Computation*, 3(4):287–297.

Hauschild, M., Pelikan, M., Lima, C. F., and Sastry, K. (2007). Analyzing probabilistic models in hierarchical BOA on traps and spin glasses. In *Proceedings of ACM SIGEVO Genetic and Evolutionary Computation Conference (GECCO-2007)*, pp. 523–530.

Hauschild, M. W., Pelikan, M., Sastry, K., and Goldberg, D. E. (2008). Using previous models to bias structural learning in the hierarchical BOA. In *Proceedings of ACM SIGEVO Genetic and Evolutionary Computation Conference (GECCO-2008)*, pp. 415–422.

Holland, J. H. (1992). *Adaptation in natural and artificial systems*. Cambridge, MA: MIT Press.

Larrañaga, P., and Lozano, J. A. (2001). *Estimation of distribution algorithms: A new tool for evolutionary computation*, Vol. 2 of *Genetic Algorithms and Evolutionary Computation*. Dordrecht, The Netherlands: Kluwer Academic Publishers.

Lima, C. F., and Lobo, F. G. (2004). Parameter-less optimization with the extended compact genetic algorithm and iterated local search. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2004)*, pp. 1328–1339.

Lima, C. F., Lobo, F. G., and Pelikan, M. (2008). From mating pool distributions to model overfitting. In *Proceedings of ACM SIGEVO Genetic and Evolutionary Computation Conference (GECCO-2008)*, pp. 431–438.

Lima, C. F., Pelikan, M., Goldberg, D. E., Lobo, F. G., Sastry, K., and Hauschild, M. (2007). Influence of selection and replacement strategies on linkage learning in BOA. In *Proceedings of 2007 IEEE Congress on Evolutionary Computation (CEC 2007)*, pp. 1083–1090.

Lima, C. F., Pelikan, M., Sastry, K., Butz, M. V., Goldberg, D. E., and Lobo, F. G. (2006). Substructural neighborhoods for local search in the Bayesian optimization algorithm. In *Proceedings of the 9th International Conference on Parallel Problem Solving from Nature (PPSN IX)*, pp. 232–241.

Lima, C. F., Sastry, K., Goldberg, D. E., and Lobo, F. G. (2005). Combining competent crossover and mutation operators: A probabilistic model building approach. In *Proceedings of ACM SIGEVO Genetic and Evolutionary Computation Conference (GECCO-2005)*, pp. 735–742.

Lobo, F. G., Goldberg, D. E., and Pelikan, M. (2000). Time complexity of genetic algorithms on exponentially scaled problems. In D. Whitley, D. Goldberg, E. Cantú-Paz, L. Spector, I. Parmee, and H.-G. Beyer (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, pp. 151–158.

Mitchell, T. M. (1997). *Machine learning*. New York: McGraw-Hill Higher Education.

Mühlenbein, H., and Höns, R. (2005). The estimation of distributions and the minimum relative entropy principle. *Evolutionary Computation*, 13(1):1–27.

Mühlenbein, H., and Mahnig, T. (1999). FDA: A scalable evolutionary algorithm for the optimization of additively decomposed functions. *Evolutionary Computation*, 7(4):353–376.

Mühlenbein, H., and Paaß, G. (1996). From recombination of genes to the estimation of distributions. I. Binary parameters. In *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature (PPSN IV)*, pp. 178–187.

Ocenasek, J. (2006). Entropy-based convergence measurement in discrete estimation of distribution algorithms. In J. A. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea (Eds.), *Towards a new evolutionary computation. Advances in estimation of distribution algorithms*, Vol. 192 of *Studies in Fuzziness and Soft Computing* (pp. 39–50). Berlin: Springer.

Pelikan, M., Goldberg, D. E., and Cantú-Paz, E. (1999). BOA: The Bayesian optimization algorithm. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99*, pp. 525–532.

Pelikan, M., Goldberg, D. E., and Lobo, F. G. (2002). A survey of optimization by building and using probabilistic models. *Computational Optimization and Applications*, 21(1):5–20.

Pelikan, M., Goldberg, D. E., and Sastry, K. (2001). Bayesian optimization algorithm, decision graphs, and Occam's razor. In L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pp. 519–526.

Pelikan, M., and Lin, T.-K. (2004). Parameter-less hierarchical BOA. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2004)*, pp. 24–35.

Pelikan, M., and Mühlenbein, H. (1999). The bivariate marginal distribution algorithm. In R. Roy, T. Furuhashi, and P. K. Chawdhry (Eds.), *Advances in soft computing—Engineering design and manufacturing* (pp. 521–535). Berlin: Springer.

Pelikan, M., Sastry, K., and Goldberg, D. E. (2002). Scalability of the Bayesian optimization algorithm. *International Journal of Approximate Reasoning*, 31(3):221–258.

Rissanen, J. (1978). Modelling by shortest data description. *Automatica*, 14:465–471.

Sastry, K. (2001). Evaluation-relaxation schemes for genetic and evolutionary algorithms. Master's thesis, University of Illinois at Urbana-Champaign. Also IlliGAL Report No. 2002004.

Sastry, K., Abbass, H. A., Goldberg, D. E., and Johnson, D. D. (2005). Sub-structural niching in estimation of distribution algorithms. In *Proceedings of ACM SIGEVO Genetic and Evolutionary Computation Conference (GECCO-2005)*, pp. 671–678.

Sastry, K., and Goldberg, D. E. (2004). Designing competent mutation operators via probabilistic model building of neighborhoods. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2004)*, pp. 114–125.

Sastry, K., Lima, C. F., and Goldberg, D. E. (2006). Evaluation relaxation using substructural information and linear estimation. In *Proceedings of ACM SIGEVO Genetic and Evolutionary Computation Conference (GECCO-2006)*, pp. 419–426.

Sastry, K., Pelikan, M., and Goldberg, D. E. (2004). Efficiency enhancement of genetic algorithms via building-block-wise fitness estimation. In *Proceedings of the 2004 IEEE Congress on Evolutionary Computation (CEC 2004)*, pp. 720–727.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464.

Thierens, D., Goldberg, D. E., and Pereira, Â. G. (1998). Domino convergence, drift and the temporal salience structure of problems. In *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation*, pp. 535–540.

Yu, T.-L., Sastry, K., Goldberg, D. E., and Pelikan, M. (2007). Population sizing for entropy-based model building in discrete estimation of distribution algorithms. In *Proceedings of ACM SIGEVO Genetic and Evolutionary Computation Conference (GECCO-2007)*, pp. 601–608.