Spurious Dependencies and EDA Scalability

Elizabeth Radetic, Martin Pelikan

MEDAL Report No. 2010002

January 2010

Abstract

Numerous studies have shown that advanced estimation of distribution algorithms (EDAs) often discover spurious (unnecessary) dependencies. Nonetheless, only little prior work exists that would study the effects of spurious dependencies on EDA performance. This paper examines the effects of spurious dependencies on the performance and scalability of EDAs with the main focus on EDAs with marginal product models and the onemax problem. A theoretical model is proposed to analyze the effects of spurious dependencies on the population sizing in EDAs and the theory is verified with experiments. The effects of spurious dependencies on the number of generations are studied empirically.

Keywords

Estimation of distribution algorithms, model accuracy, spurious dependencies, population sizing, scalability, gambler's ruin model, initial supply.

Missouri Estimation of Distribution Algorithms Laboratory (MEDAL)
Department of Mathematics and Computer Science
University of Missouri–St. Louis
One University Blvd., St. Louis, MO 63121
E-mail: `medal@cs.umsl.edu`
WWW: `http://medal.cs.umsl.edu/`

Spurious Dependencies and EDA Scalability

Elizabeth Radetic
Missouri Estimation of Distribution Algorithms Laboratory (MEDAL)
Dept. of Math and Computer Science, 321 CCB
University of Missouri at St. Louis
One University Blvd., St. Louis, MO 63121

Martin Pelikan
Missouri Estimation of Distribution Algorithms Laboratory (MEDAL)
Dept. of Math and Computer Science, 320 CCB
University of Missouri at St. Louis
One University Blvd., St. Louis, MO 63121
pelikan@cs.umsl.edu

January 30, 2010

Abstract

Numerous studies have shown that advanced estimation of distribution algorithms (EDAs) often discover spurious (unnecessary) dependencies. Nonetheless, only little prior work exists that would study the effects of spurious dependencies on EDA performance. This paper examines the effects of spurious dependencies on the performance and scalability of EDAs with the main focus on EDAs with marginal product models and the onemax problem. A theoretical model is proposed to analyze the effects of spurious dependencies on the population sizing in EDAs and the theory is verified with experiments. The effects of spurious dependencies on the number of generations are studied empirically.

Keywords: Estimation of distribution algorithms, model accuracy, spurious dependencies, population sizing, scalability, gambler's ruin model, initial supply.

1 Introduction

Estimation of distribution algorithms (EDAs) [20, 15, 5, 24] generate new candidate solutions by sampling an explicit probabilistic model constructed from promising solutions found so far. One of the key benefits of EDAs is that many of them are able to learn the structure of a problem with no prior knowledge of it [2, 1, 25, 8, 23, 22]. A potential problem, however, is that the problem structure, or some parts of it, may be learned incorrectly.

There are two main types of inaccuracies of EDA models in the discrete domain [18, 31, 12]: (1) The model may be missing some important dependencies, and (2) the model may contain unnecessary (spurious) dependencies. Both types of model inaccuracies are common in practice and several empirical studies have been published that provide insight into the accuracy of models learned by some multivariate EDAs [18, 31, 12, 17, 16]. Although it has been argued that both types of model inaccuracies hamper EDA performance [22, 32, 33, 17], most prior work on this topic focused on the learning of important dependencies and there has been practically no theoretical work on the effects of spurious dependencies in EDAs.

The purpose of this paper is to examine the effects of spurious dependencies on the performance and scalability of EDAs. The main focus is on the onemax problem and EDAs based on marginal product models. The effects of spurious dependencies on the population sizing are studied theoretically and the theory is verified with experiments. The impact of spurious dependencies on the number of generations until optimum is analyzed empirically. The results indicate that in most cases the effects of spurious dependencies on the population sizing are more significant than those on the number of generations until convergence and that assuming that the order of spurious dependencies is small, spurious dependencies should not significantly affect EDA scalability. However, the results show that when restricted tournament replacement is used to maintain population diversity, the effects of spurious dependencies on the number of generations until convergence become much more significant and the scalability suffers.

The paper is organized as follows. Section 2 provides a brief background on EDAs and outlines the onemax model for spurious dependencies. Section 3 summarizes the existing theoretical models of population sizing and time to convergence for EDAs, and it discusses how these models can be extended to account for spurious dependencies. Section 4 presents experimental results. Finally, section 5 summarizes and concludes the paper, and outlines important topics for future work.

2 Estimation of Distribution Algorithms

This section presents a brief background on EDAs with the main focus on the extended compact genetic algorithm (ECGA) and model accuracy in EDAs.

2.1 From Probability Vectors to Marginal Product Models

EDAs are distinguished from most other evolutionary algorithms by the method of generating new solutions from the population of selected promising solutions. Rather than using crossover and mutation, EDAs build an explicit probabilistic model of the selected population and then sample this model to generate new solutions. In this paper, the focus is on EDAs for problems defined over fixed-length binary vectors, although all EDAs discussed here can be easily modified to deal with strings over any finite alphabet.

Some of the simplest EDAs, such as the univariate marginal distribution algorithm (UMDA) [20] and the compact genetic algorithm (cGA) [11], treat each variable as independent. As the probabilistic model, both cGA and UMDA use the probability vector $p = (p_1, p_2, \ldots, p_n)$, which stores the probability p_i of a 1 in each position i. After selecting promising solutions, the probability vector is built by calculating the probabilities of 1s in each position of the selected set of promising solutions. New solutions are generated by sampling the probability vector, generating a 1 in each position with the probability encoded by the probability vector. Using single-bit probabilities to generate new candidate solutions provides excellent mixing of the individual bits [29]. However, since all bits are considered independent, statistical dependencies between different problem variables cannot be preserved and, consequently, even some relatively simple problems cannot be solved efficiently [30, 5, 22]. This problem is addressed by more advanced EDAs capable of solving problems in which variables interact and must be treated together [24, 22].

One of the EDAs capable of learning and using multivariate interactions among the variables is the the extended compact genetic algorithm (ECGA) [8, 19]. ECGA uses a marginal product model (MPM) for this purpose. In MPM the string positions are partitioned into disjoint subsets and for each subset, probabilities are stored for all combinations of bits in this subset. For example, an MPM model for 5-bit strings $(X_1, X_2, X_3, X_4, X_5)$ may represent 3 linkage groups $[X_1, X_4]$, $[X_2, X_5]$, and $[X_3]$. The new solutions are generated by sampling each of the groups using the probabilities stored for this group. This allows ECGA to preserve combinations of bits for variables that strongly interact.

To build an MPM, ECGA begins by assuming that each position is independent and considering each string position as an individual subset. In each iteration of the model building, models created by merging pairs of subsets are evaluated and the merge that improves model quality the most is executed. The model building terminates when no such merge improves model quality. For more details on model building in ECGA, see ref. [8]. The model-building procedure allows ECGA to deduce linkage between bit positions. This capacity for linkage learning makes ECGA an effective solver for problems composed of separable, non-overlapping subproblems, including many problems that cannot be efficiently tackled with simple, univariate models [8].

2.2 Linkage Learning and Model Accuracy

Along with ECGA, many other EDAs [2, 25, 8, 23, 22] are able to learn linkage. Linkage learning is important because it allows EDAs to discover and mix the optimal subsets of bits which contribute to the construction of the global optimum. The subsets which must be preserved and combined to construct the global optimum correspond to a decomposition of the problem into quasi-independent subproblems.

The ability to automatically discover an adequate problem decomposition is one of the main advantages of EDAs over more conventional evolutionary algorithms. Of course, one must make sure that the models represent an accurate decomposition of the problem. Although many EDAs use advanced machine learning techniques to learn models, because of the stochastic errors due to the use of finite solution samples, model inaccuracies may arise in practice. Specifically, the learned models may miss some necessary dependencies or introduce unnecessary, *spurious* dependencies.

Several studies have explored the issue of model accuracy in multivariate EDAs [18, 31, 12, 17, 16]. These studies examined several facets of model quality. They enumerated the conditions under which accurate models are created, in which the correct dependencies are discovered and spurious dependencies are omitted. They also examined the errors which are found in models, including the inclusion of spurious dependencies and the omission of necessary dependencies. Most of the experiments described used population sizes which were empirically found to be optimal. Though the population sizes were chosen to produce the best performance, the constructed models were still often found to have some inaccuracies. Thus even with a nearly optimal population size, inaccuracies such as missing or spurious dependencies often arise in practice.

Studies have also examined the impact model inaccuracies have on different aspects of EDA performance. Lima et al. [17] noted that inaccuracies in EDA models can negatively impact the performance of efficiency enhancement techniques which depend on model accuracy such as substructural hillclimbing [18]. Yu et al. [32] examined the effects of missing necessary dependencies on EDA convergence time. Later, Yu et al. [33] investigated the effects of spurious dependencies on convergence time in EDAs. These two studies noted the negative effects of both missing and spurious dependencies. The impact of missing dependencies on population sizing is discussed also in ref. [30], which concluded that population size increases exponentially when recombination does not respect an adequate decomposition for classes of problems decomposable into additively decomposable problems of fixed order $k > 1$. These studies all made important contributions to the understanding of the effects of missing and spurious dependencies on EDA performance.

Nonetheless, only little work exists that studies the effects of spurious dependencies on EDA scalability, which are the main topic discussed in this paper. Yet, understanding the effects of spurious dependencies represents one of the most important theoretical challenges in EDA research.

Spurious dependencies in EDA models arise due to a few different reasons. Sastry and Goldberg [28] noted spurious dependencies in ECGA models when solving the onemax problem. They explained it as the result of hitchhiking between bits due to the bias in the original population. The initial bias is due to the finite population size and the effects of the random number generator used to generate the initial population. Although increasing population size increased model accuracy in this instance, overall performance of the algorithm is decreased and additional spurious dependencies are discovered [26] when the population size becomes too large. This is the result of linkage disequilibrium, in which selection itself introduces statistical dependencies even when there are none in the underlying problem. The statistical dependencies lead to spurious dependencies in EDA models [26, 29].

The remainder of this section discusses the model for studying the effects of spurious dependencies, which is used as the starting point in this paper. The effects of spurious dependencies on EDA performance will be discussed next.

2.3 Onemax Model for Spurious Dependencies

In onemax, the task is to maximize the following objective (fitness) function:

$$onemax(X_1, X_2, \ldots, X_n) = \sum_{i=1}^{n} X_i,$$

where X_i denotes the ith bit in the input binary string.

To solve onemax, the model does not have to include any dependencies, because the fitness contribution of each bit is fully independent of others. Furthermore, to maximize mixing [29, 5] and minimize the population-sizing requirements due to the initial supply [7, 5], the number of dependencies should be minimized. Thus, the probability vector should provide a perfect model for onemax. This hypothesis is confirmed with the theory presented in the next section.

To study the effects of spurious dependencies, we use MPM models of ECGA. The order of linkage groups is tuned by specifying their size using the parameter $k_{spurious}$. For simplicity, we assume that all linkage group are of the same size. Ideally, the order of the linkage groups should be $k_{spurious} = 1$, in which case the model is simply the probability vector. The larger the value of $k_{spurious}$, the greater the order of spurious dependencies and the worse the performance of an EDA should become. Since in onemax the function of each bit is the same, it does not matter what bits are in what linkage group; the only thing that matters is how large the linkage groups are.

Of course, a more complex model may be used for modeling spurious linkage. For example, linkage groups of different sizes may be used or Bayesian network models may be used as the underlying class of models. Nonetheless, the simplicity of the model used in this paper allows a straightforward use of the scalability theory for genetic algorithms to theoretically predict the effects of spurious dependencies.

3 EDA Scalability

Scalability is an important topic in the study of EDAs and their computational requirements because study of computational complexity on problems of limited size can often be misleading.

Specifically, the key issue is the growth of the computational requirements with problem size and other problem properties. For EDAs and other population-based evolutionary algorithms, scalability depends mainly on the number of evaluations until the global optimum is reached, and the number of evaluations is given as the product of the population size and the number of generations. Therefore, the two most important measures of EDA scalability are (1) the population size required to solve the problem and (2) the number of generations until the population converges. The following sections will discuss these aspects of EDA scalability. Some of the presented scalability theory is extended to the onemax model for spurious linkage.

3.1 Population Sizing

For most EDAs and other population-based evolutionary algorithms, the population size is the most important parameter to set in order to ensure reliable convergence to the global optimum or its accurate approximation. Most of the studies of EDA population sizing are based on the theoretical models for standard genetic algorithms [9, 5]. These studies were further extended to account for the specific features of EDAs [22, 26]. Essentially, there are three main factors that influence population sizing for EDAs:

(i) Initial supply (see section 3.1.1),

(ii) decision making (see section 3.1.2), and

(iii) model building (see section 3.1.3).

The remainder of this section discusses these three factors and relates them to the onemax model for spurious dependencies in EDAs based on marginal product models.

3.1.1 Initial Supply

Assuming an adequate decomposition of the problem, it is important that with a high probability the initial population contains at least one copy of each partial solution in this decomposition. Assuming that all partial solutions are properly represented, the partial solutions contained in the optimum can be combined to form the optimum using the methods for learning and sampling probabilistic models or more conventional recombination operators.

Assuming that the order of partial solutions is upper bounded by k and the total number of subproblems (partitions) is m, the initial-supply population-sizing model [7, 5] provides a lower bound on the population size of

$$N = \chi^k (k \ln \chi + \ln m), \tag{1}$$

where χ is the cardinality of the alphabet. The above model assumes that on average all but one partial solution will be fully represented in the initial population. It is of note that this initial-supply model can be extended to other scenarios in a straightforward manner, but for the purposes of this paper the above bound suffices. The initial-supply bound grows at a slower rate than the remaining two population-sizing factors and that is why the initial-supply requirements are relatively easy to satisfy [22, 26].

With respect to the onemax model for spurious dependencies, the population size to ensure a sufficient initial supply of partial solutions is given by

$$N = 2^{k_{spurious}} \left(k_{spurious} \ln 2 + \ln \frac{n}{k_{spurious}} \right). \tag{2}$$

The growth of the population size in terms of the order $k_{spurious}$ of spurious dependencies can also be expressed as the ratio of the population size required for for an arbitrary value of $k_{spurious} \geq 1$ and that for the perfect model (probability vector):

$$\gamma_{is} = 2^{k_{spurious}-1} \frac{k_{spurious} \ln 2 + \ln \frac{n}{k_{spurious}}}{\ln 2 + \ln n} \tag{3}$$

It is easy to see that $\gamma_{is} \geq 1$ for $k_{spurious} \geq 1$, and $\gamma_{is} > 1$ for $k_{spurious} > 1$. The greater the value of γ_{is}, the greater the initial-supply population-sizing requirements compared to the case with the probability vector.

3.1.2 Decision Making

The search for the optimum in a GA or EDA can be viewed as many simultaneous competitions between competing partial solutions in different partitions of the problem decomposition (subproblems) [13, 14, 4, 6]. The decision making between competing partial solutions in any partition is stochastic in nature, because the solution quality is affected also by the remaining partitions. The influence of the remaining problem partitions or subproblems can be modeled as collateral noise [13, 14, 6]. It is important to use large enough populations to alleviate the effects of collateral noise and to ensure that the correct partial solutions increase in number over time.

Assuming that the problem can be exactly or approximately decomposed into independent subproblems of bounded order k, the decision making can be modeled using the gambler's ruin model [3, 9, 5]. Let us assume that the difference between the contribution of the best and the second best partial solutions is at least d and that the variance of fitness contributions of one subproblem is upper bounded by σ_{bb}^2. To ensure that the probability of obtaining a correct partial solution for every subproblem is at least $(1-\alpha)$, the gambler's ruin population-sizing model provides a lower bound on the population size [9]

$$N = -2^{k-1} \ln(\alpha) \frac{\sigma_{bb}\sqrt{\pi m'}}{d}, \tag{4}$$

where k is the size of one subproblem, α is the error (probability of missing a partial solution), σ_{bb} is the square root of the fitness variance of one subproblem, $m' = m - 1$, m is the total number of subproblems, and d is the signal [9]. The model assumes that the solutions are represented by binary strings and that the initial population is generated at random using the uniform distribution over all binary strings of the given length [9].

For onemax and EDAs based on probability vectors, the population size bound based on the gambler's ruin becomes

$$N = -\frac{1}{2} \ln \alpha \sqrt{\pi(n-1)}. \tag{5}$$

Assuming onemax and an MPM model with the linkage group size $k_{spurious}$, the model yields

$$N = -2^{k_{spurious}-2} \ln \alpha \sqrt{\pi(n-1)}. \tag{6}$$

The ratio γ_{dm} by which the decision-making population size bound increases with $k_{spurious}$ can thus be estimated as

$$\gamma_{dm} = 2^{k_{spurious}}. \tag{7}$$

It is of note that while the initial supply factor requires population sizes that grow as $\Omega(2^k \ln n)$, the gambler's ruin model leads to a lower bound of $\Omega(2^k \sqrt{n})$. The lower bound based on the decision making therefore grows more quickly than that based on the initial supply and the decision making thus takes precedence over the initial supply in population sizing for both GAs and EDAs. This observation has been highlighted in a number of past population-sizing studies for GAs and EDAs [5, 26, 22].

3.1.3 Model Building

Having a sufficiently large population of solutions is one of the keys to building accurate models. In fact, model building is the most important factor in population sizing for EDAs that use advanced probabilistic models capable of linkage learning [26, 22].

To estimate an adequate population size for building accurate models, Pelikan et al. [26] demonstrated that, for uniformly-scaled problems of length n and building blocks of size k, the optimal population size to discover necessary dependencies is $\Omega(2^k n^{1.05})$ while the size to discover unnecessary dependencies is $\Omega(2^k n^{2.1})$. Yu et al. [34] further refined the model for population sizing in entropy-based model building EDAs as $\Omega(2^{2k} n \ln n)$. Clearly, the estimates which take model building into consideration grow faster than both the $\Omega(2^k \ln n)$ bound for the initial supply and the $\Omega(2^k \sqrt{n})$ bound for the decision making. Therefore, model building is expected to take precedence over both the initial supply and the decision making in EDA population sizing.

Nonetheless, for the purposes of this paper, the population sizing for the model building is not relevant. This is because the goal of this paper is to study the effects of spurious dependencies discovered due to the use of a finite sample size. On the other hand, regarding spurious dependencies, the goal of the population sizing models for the model building is to investigate the reasons for the discovery of spurious dependencies or methods to alleviate this problem.

3.2 Convergence Time

Another important aspect of EDA scalability is the number of generations until convergence (also called time to convergence). For EDAs with the probability vector on onemax, the convergence time can be estimated by [21]

$$G = \left(\frac{\pi}{2} - \arcsin(2p-1)\right) \frac{\sqrt{n}}{I} \tag{8}$$

where p is the initial proportion of building blocks, n is the problem length and I is the selection intensity. For many selection methods, including the tournament and truncation selection, I is constant [5]. This model shows that, assuming a constant I, the growth of convergence time is expected to be $O(\sqrt{n})$. Thus under constant selection, convergence time grows relatively slowly with respect to the problem size n. The growth of the convergence time may become somewhat faster—specifically, it may become $O(n)$—if the fitness contributions of the different subproblems are nonuniformly scaled [29].

We have not extended the above theoretical convergence model to estimate the effects of spurious dependencies on time to convergence. Nonetheless, it can be hypothesized that the number of generations until convergence for the onemax model for spurious dependencies will still be upper bounded by eq. 8. The main reason for this is that the key difference between the two scenarios is that a model with spurious dependencies does not enforce statistical independence of pairs of string positions and that because of this, the populations may lose diversity faster than with the probability vector. This should in turn lead to even shorter convergence times. This hypothesis is in agreement with experimental results, which show that the number of generations decreases slightly with $k_{spurious}$, although the effects of $k_{spurious}$ are nearly negligible. The only scenario that contradicts this hypothesis is that with niching.

3.3 Discussion

The estimates presented thus far in this section provide a conservative bound on the impact of spurious linkage on population sizing and time to convergence. The main reason for this is that the onemax model for spurious dependencies assumes that all bits included in one linkage group should be in fact treated independently; in practice, at least some of the bits in each linkage group can be expected to depend on each other. Furthermore, in the assumed model the linkage groups do not change over time, whereas in practice, many of the spurious dependencies can be expected to be eliminated over time. For example, Hauschild et al. [12], found that most spurious dependencies are typically eliminated by the middle of an EDA run. This would further reduce the impact of spurious dependencies on EDA performance.

Finally, it is important to remember that model building itself is usually the dominant factor in determining population sizing for EDAs. Thus when model building is performed in a typical EDA, this factor will take precedence in population sizing, over both initial supply and decision making. As a result, the population size is still expected to be $O(2^{2k} n \ln n)$ [34], which outweighs the effects of spurious linkage. The model presented in this paper provides a theoretical support for this fact.

The next section presents experiments to empirically investigate the effects of spurious linkage on EDA scalability.

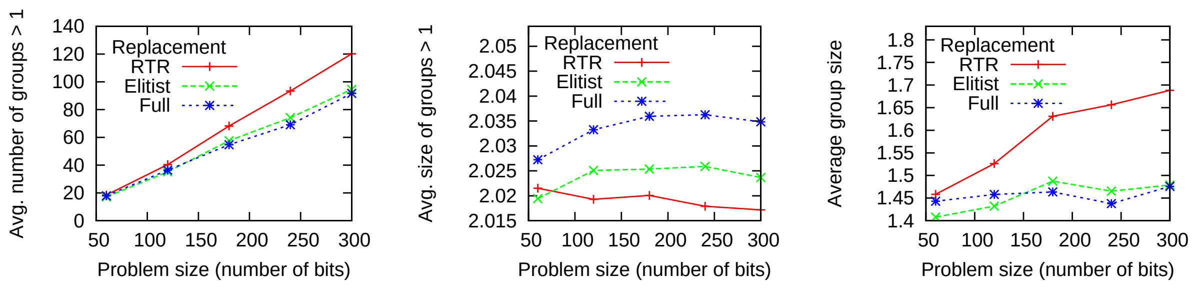
4 Experiments

Thus far, we have proposed a onemax model for spurious dependencies and presented theory that estimates the effects of spurious dependencies on the population size for the proposed model. This section presents experimental results. There are two main goals of the experiments in this paper: (1) Establish the types of spurious dependencies in practice, and (2) analyze the effects of spurious dependencies and verify the theory presented in section 3. Since several studies have been published on the accuracy of EDA models [12, 17, 16], the main focus is on the second goal.

4.1 Methodology

Two sets of experiments were conducted. In the first set, standard ECGA was used to solve onemax, and its models were analyzed to determine the types and quantities of spurious dependencies discovered. In the second set of experiments, an EDA with a fixed model structure was used to solve onemax, and the effects of spurious linkage on performance with the special focus on the population size were measured. As a base case for these tests, an EDA with the probability vector [20] was used, since this is the perfect model for onemax. Then, for comparison, MPM models were used with spurious linkage groups of size $k_{spurious} > 1$.

The experiments involving fixed models with spurious dependencies were conducted in two parts. In the first part, MPM models with a fixed size of linkage groups from $k_{spurious} = 2$ to 5 were used with the total number $n/k_{spurious}$ of linkage groups. In the second part, only linkage groups of size 1 and 2 were used and the number of linkage groups of size 2 was varied from 0 (probability vector) to $n/2$ (all bits are paired). The scenario with most linkage groups of size 1 or 2 corresponds closely to practice.



(a) Number of spurious linkage groups (b) Avg. size of spurious linkage groups (c) Average linkage group size

Figure 1: The average number of spurious linkage groups (groups of size ≥ 2), the average size of linkage groups of size ≥ 2, and the average linkage group size (including all linkage groups) for ECGA on onemax. Three replacement strategies are considered: full replacement, elitist replacement and RTR. For each problem size and replacement strategy, the results represent an average over 100 runs (10 bisections of 10 runs each).

Problems of size $n = 60$ to 300 bits in increments of 60 were tested. Population sizes were determined empirically using the bisection method [27, 22] to ensure 10 successful consecutive runs. For each problem size and each test scenario, 10 independent bisections were performed, for a total of 100 independent runs. A run was considered successful when a string was found with at most one suboptimal linkage group (with the linkage groups depending on the used model). For the base cases with no spurious linkage and the experiments with ECGA, at most 1 bit was allowed to be incorrect. Full population convergence was not required and each run was terminated when one solution of the desired quality had been found. This allowed the same stopping criterion for all tested replacement methods including those with niching, which prevents full convergence.

Binary tournament selection without replacement was used to select parent populations. To incorporate the offspring into the population, three replacement strategies were tested: (1) Full replacement, where the child population completely replaces the parent population; (2) elitist replacement, where the worst 50% of the parent population was replaced with the child population; and (3) restricted tournament replacement (RTR) [10, 22], where for each offspring w solutions were randomly selected from the parent population and the one genotypically closest to the offspring was replaced if its fitness is worse. The window size in RTR was set to $\min(n, 0.05N)$ as suggested by ref. [22].

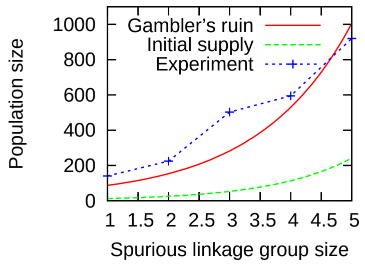
The maximum number of generations allowed was set to $5n$ for most runs, but this limit was increased to $10000n$ when RTR was used with spurious dependencies. The need for this increase was due to the effects of niching, as will be discussed in section 4.2.3.

4.2 Results

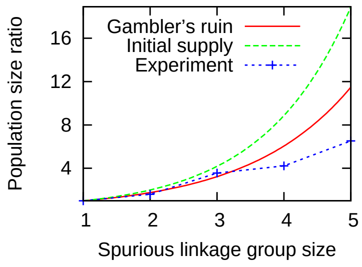
This section presents experimental results. First, the results that depict the accuracy of ECGA models on onemax are presented. The effects of spurious dependencies on the population size and the time to convergence are then shown.

4.2.1 Spurious linkage for ECGA on onemax

The results in figure 1 provide an insight into the number and size of spurious dependencies discovered by ECGA on onemax using the population size obtained with bisection. Specifically, figure

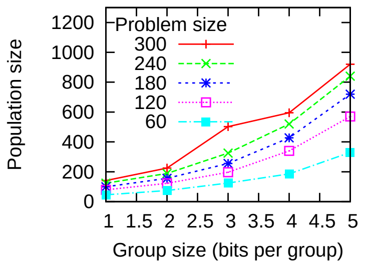


(a) Population size

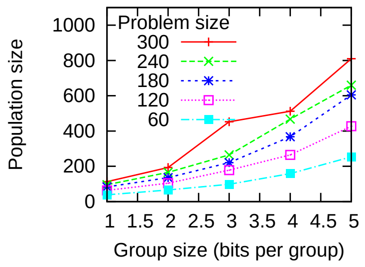


(b) Population size ratio

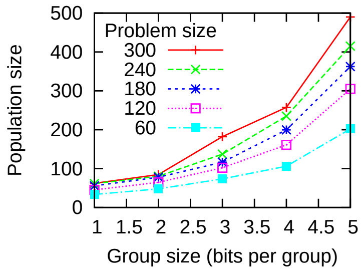
Figure 2: Growth of the population size with respect to the group size for a problem of 300 bits. The left-hand side shows the actual population sizes compared to the theoretical model, whereas the right-hand side shows the ratio of the population sizes with spurious linkage and the population sizes with no spurious linkage.



(a) Full replacement



(b) Elitist replacement



(c) RTR

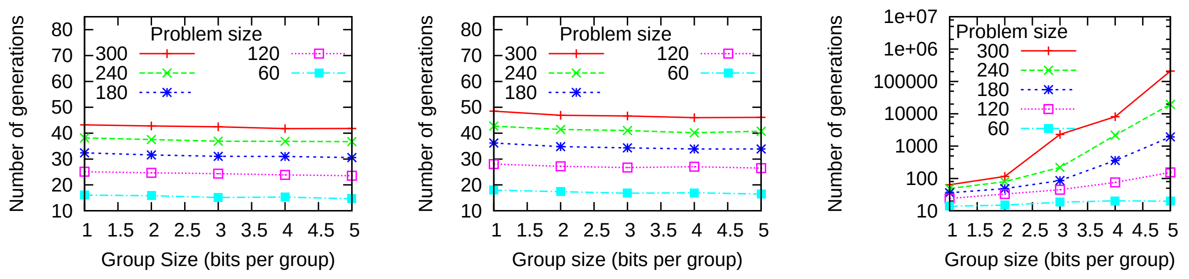
Figure 3: Growth of the population size with respect to the spurious linkage group size.

1(a) shows the average number of spurious linkage groups (groups of size at least 2) for each problem size. The results indicate that the number of such groups increases approximately linearly with problem size. Figure 1(b) shows the average size of spurious linkage groups. For each problem size, the size of spurious linkage groups is close to two, indicating that larger linkage groups were created only rarely. Finally, figure 1(c) shows the average linkage group size when both spurious linkage groups and independent bits are taken into account. The average group size is between 1 and 2 and it increases slightly with problem size. Similar results for ECGA on onemax were reported in ref. [28]. The results presented here thus reaffirm the need for studying spurious dependencies and their effects.

4.2.2 Population sizing with spurious linkage

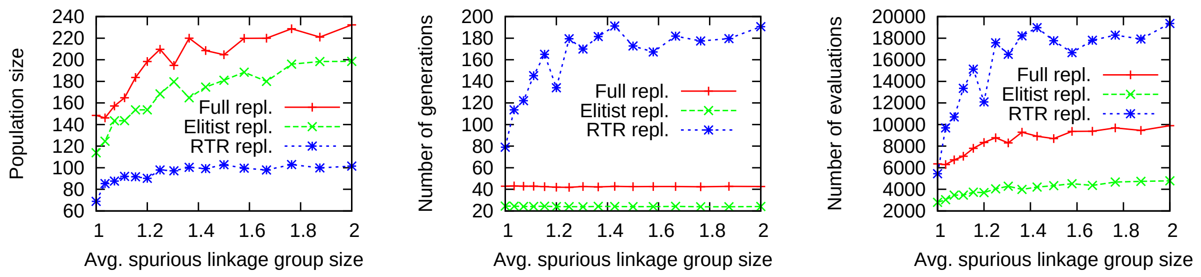
This section presents the results of using fixed MPM models with spurious dependencies on onemax. To confirm the theory presented in section 3, figure 2 compares the experimental results for the population size to the predictions made by the initial supply and gambler's ruin population sizing models. As expected, the gambler's ruin model matches the experimental results more closely than that for the initial supply. The gambler's ruin model can therefore help to determine the impact of spurious dependencies on EDA population sizing. Since the population size is one of the key factors that affect EDA performance, this can provide guidelines for predicting the overall impact of spurious dependencies on EDA performance.

Additional results on the effects of spurious dependencies on the population size for all problem sizes and replacement methods are shown in figure 3. These results illustrate that, in each case, the population size grows approximately exponentially with the spurious linkage group size. Further-



(a) Full replacement (b) Elitist replacement (c) RTR

Figure 4: Growth of the number of generations with respect to the spurious linkage group size.



(a) Population size (b) Number of generations (c) Number of evaluations

Figure 5: The population size, the number of generations, and the number of evaluations for the 300-bit onemax with varying numbers of spurious linkage groups of size 2 (the remaining groups are of size 1).

more, elitist replacement requires somewhat smaller populations than full replacement, and RTR outperforms other replacement strategies in terms of the population size. However, in each case the relative increase of population sizes with the size of spurious linkage groups is similar.

The results for ECGA presented earlier (figure 1) indicated that typically the models in ECGA combined only a fraction of the bits into spurious linkage groups, most of which were of size 2. The results of figure 5 illustrate the effects of having different proportions of such linkage groups. More specifically, figure 5 illustrates the results obtained when using a variable number of spurious linkage groups of size 2 on a 300-bit onemax (the remaining groups are of size 1). From these results, it is clear that the required population size increases with the number of spurious linkage groups. An upper bound for this scenario should be of course provided by EDAs with MPMs with the fixed order of spurious dependencies $k_{spurious} = 2$.

In summary, the results presented in this subsection demonstrate that spurious dependencies tend to increase the required population size for EDAs and that the theory from section 3 provides an accurate estimation of the effects of spurious dependencies on the population size.

4.2.3 Number of generations with spurious linkage

According to the results in figures 4 and 5, the number of generations until optimum does not seem to be strongly affected by the order of spurious dependencies when full and elitist replacement methods are used. In fact, figure 4 shows that the number of generations required to find an accurate solution decreases slightly with the size of spurious linkage groups. Figure 5 shows similar results on a smaller scale.

The number of generations until optimum is affected differently, however, when niching is introduced, as is revealed in the results for RTR (figure 4c). The purpose of niching is to preserve useful diversity so that (1) the population does not converge before a solution of the desired quality is found and (2) the chances of obtaining multiple alternative solutions increase. As stated in section 4.1, full population convergence was not required in these experiments. Instead, only one optimal solution was required to be found so that the same stopping criteria could be used in all runs, regardless of the replacement strategy. Typically, this prevents RTR from excessively extending the number of generations that the algorithm runs. With spurious linkage groups, however, the time to find even one optimal solution was extended substantially anyway. Figure 4 reveals that the number of generations to find a solution of sufficient quality grew faster than exponentially with the size of spurious linkage groups. This is dramatically different compared to the results for full and elitist replacement, for which the number of generations decreased slightly with $k_{spurious}$.

Yu et al. [33] suggested that spurious linkage undermines the ability of EDAs to effectively mix partial solutions. The results presented in figures 4 and 5 show that the detrimental effects of spurious dependencies become much more substantial in the presence of niching. Since it has been suggested that niching is important for solving difficult hierarchical and nearly decomposable problems [22], the results presented in this paper strongly reaffirm the importance of studying the effects of spurious linkage in EDAs and finding ways to alleviate these effects in practice.

5 Summary and Conclusions

Prior work on advanced EDAs showed that models discovered by EDAs often contain spurious or unnecessary dependencies, which negatively affect EDA performance. Yet, there has been practically no prior work that would analyze the effects of spurious dependencies theoretically.

The study presented here examined the effects of spurious dependencies on the performance and scalability of EDAs. A model for spurious dependencies based on the simple onemax problem was proposed and the effects of spurious dependencies on the required population size were theoretically analyzed. In summary, spurious dependencies were shown to increase the population size required to obtain a solution of sufficient quality. It was also argued that the effects of spurious dependencies on the number of generations until optimum should be negligible, although in the presence of niching, the negative effects of spurious dependencies were empirically shown to be substantial. This emphasizes the need for understanding the effects of spurious dependencies and finding ways for alleviating these effects in practice.

Although this paper shed some light on the effects of spurious dependencies on EDA performance and proposed an approach to study the effects of spurious dependencies theoretically, many research challenges remained for future work. First of all, a rigorous convergence time model should be developed to estimate the effects of spurious dependencies on the number of generations. Furthermore, one should examine the effects of spurious dependencies for models in which the subproblems overlap; in this case, Bayesian networks can be used as the base class of models. It would also be interesting to combine the results on the effects of spurious dependencies with those on model building to gain better understanding of the tradeoffs involved. Finally, this paper showed that the negative effects of spurious dependencies become much more substantial in the presence of niching; this indicates that the study of the interaction between models with spurious dependencies and niching remains an important topic for future work.

Acknowledgments

This project was sponsored by the National Science Foundation under CAREER grant ECS-0547013, by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant FA9550-06-1-0096, and by the Univ. of Missouri in St. Louis through the High Performance Computing Collaboratory sponsored by Information Technology Services, and the Research Award and Research Board programs. The U.S. Government is authorized to reproduce and distribute reprints for government purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, the Air Force Office of Scientific Research, or the U.S. Government. Some experiments were done using the hBOA software developed by M. Pelikan and D. E. Goldberg at the Univ. of Illinois at Urbana-Champaign and most experiments were performed on the Beowulf cluster maintained by ITS at the Univ. of Missouri in St. Louis.

References

[1] S. Baluja and S. Davies. Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space. *Proceedings of the International Conference on Machine Learning*, pages 30–38, 1997.

[2] J. S. De Bonet, C. L. Isbell, and P. Viola. MIMIC: Finding optima by estimating probability densities. *Advances in neural information processing systems (NIPS-97)*, 9:424–431, 1997.

[3] W. Feller. *An introduction to probability theory and its applications.* Wiley, New York, NY, 1970.

[4] D. E. Goldberg. *Genetic algorithms in search, optimization, and machine learning.* Addison-Wesley, Reading, MA, 1989.

[5] D. E. Goldberg. *The design of innovation: Lessons from and for competent genetic algorithms*, volume 7 of *Genetic Algorithms and Evolutionary Computation.* Kluwer, 2002.

[6] D. E. Goldberg, K. Deb, and D. Thierens. Genetic algorithms, noise, and the sizing of populations. IlliGAL Report No. 91010, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 1991.

[7] D. E. Goldberg, K. Sastry, and T. Latoza. On the supply of building blocks. IlliGAL Report No. 2001015, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 2001.

[8] G. Harik. Linkage learning via probabilistic modeling in the ecga. IlliGAL Report No. 99010, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 1999.

[9] G. Harik, E. Cantú-Paz, D. E. Goldberg, and B. L. Miller. The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation*, 7(3):231–253, 1999.

[10] G. R. Harik. Finding multimodal solutions using restricted tournament selection. *Proceedings of the International Conference on Genetic Algorithms (ICGA-95)*, pages 24–31, 1995.

[11] G. R. Harik, F. G. Lobo, and D. E. Goldberg. The compact genetic algorithm. *Proceedings of the International Conference on Evolutionary Computation (ICEC-98)*, pages 523–528, 1998.

[12] M. Hauschild, M. Pelikan, C. Lima, and K. Sastry. Analyzing probabilistic models in hierarchical boa on traps and spin glasses. MEDAL Report No. 2007001, Missouri Estimation of Distribution Algorithms Laboratory, University of Missouri–St. Louis, St. Louis, MO, 2007.

[13] J. Holland. Genetic algorithms and the optimal allocation of trials. *SIAM Journal of Computing*, 2(2):88–105, 1973.

[14] J. H. Holland. *Adaptation in natural and artificial systems*. University of Michigan Press, Ann Arbor, MI, 1975.

[15] P. Larrañaga and J. A. Lozano, editors. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Kluwer, Boston, MA, 2002.

[16] C. Lima, F. G. Lobo, and M. Pelikan. From mating pool distributions to model overfitting. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2008)*, pages 431–438, 2008.

[17] C. Lima, M. Pelikan, D. E. Goldberg, F. G. Lobo, K. Sastry, and M. Hauschild. Influence of selection and replacement strategies on linkage learning in boa. MEDAL Report No. 2007005, Missouri Estimation of Distribution Algorithms Laboratory, University of Missouri–St. Louis, St. Louis, MO, 2007.

[18] C. Lima, M. Pelikan, K. Sastry, M. Butz, and D. E. Goldberg. Substructural neighborhoods for local search in the bayesian optimization algorithm. MEDAL Report No. 2006007, Missouri Estimation of Distribution Algorithms Laboratory, University of Missouri–St. Louis, St. Louis, MO, 2006.

[19] F. G. Lobo, K. Sastry, and G. R. Harik. Extended compact genetic algorithm in c++ version 1.1. IlliGAL Report No. 2006012, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 2006.

[20] H. Mühlenbein and G. Paaß. From recombination of genes to the estimation of distributions I. Binary parameters. *Parallel Problem Solving from Nature*, pages 178–187, 1996.

[21] H. Mühlenbein and D. Schlierkamp-Voosen. Predictive models for the breeder genetic algorithm: I. Continuous parameter optimization. *Evolutionary Computation*, 1(1):25–49, 1993.

[22] M. Pelikan. *Hierarchical Bayesian optimization algorithm: Toward a new generation of evolutionary algorithms*. Springer-Verlag, 2005.

[23] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. BOA: The bayesian optimization algorithm. IlliGAL Report No. 990003, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 1999.

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

[25] M. Pelikan and H. Mühlenbein. The bivariate marginal distribution algorithm. *Advances in Soft Computing—Engineering Design and Manufacturing*, pages 521–535, 1999.

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

[27] K. Sastry. Evaluation-relaxation schemes for genetic and evolutionary algorithms. Master's thesis, University of Illinois at Urbana-Champaign, 2001.

[28] K. Sastry and D. E. Goldberg. On extended compact genetic algorithm. IlliGAL Report No. 2000026, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 2000.

[29] D. Thierens. *Analysis and design of genetic algorithms.* PhD thesis, Katholieke Universiteit Leuven, Leuven, Belgium, 1995.

[30] D. Thierens. Scalability problems of simple genetic algorithms. *Evolutionary Computation*, 7(4):331–352, 1999.

[31] H. Wu and J. L. Shapiro. Does overfitting affect performance in estimation of distribution algorithms. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2006)*, pages 433–434, 2006.

[32] T.-L. Yu and D. E. Goldberg. Quality and efficiency of model building for genetic algorithms. IlliGAL Report No. 2004004, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 2004.

[33] T.-L. Yu, K. Sastry, and D. E. Goldberg. Linkage learning, overlapping building blocks, and systematic strategy for scalable recombination. IlliGAL Report No. 2005016, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 2005.

[34] T.-L. Yu, K. Sastry, D. E. Goldberg, and M. Pelikan. Population sizing for entropy-based model building in estimation of distribution algorithms. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2007)*, pages 601–608, 2007.