Swarm and Evolutionary Computation 27 (2016) 80–96

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Regular Paper

Pairwise independence and its impact on Estimation of Distribution Algorithms

Jean P. Martins *, Alexandre C.B. Delbem

University of São Paulo, Institute of Mathematical and Computer Sciences, São Carlos — SP 13566-590, Brazil

ARTICLE INFO

Article history:
Received 24 January 2015
Received in revised form
3 September 2015
Accepted 5 October 2015
Available online 22 October 2015

Keywords:
EDAs
Pairwise independence
Linkage-learning
LTGA
BOA
NK-landscapes

ABSTRACT

Estimation of Distribution Algorithms (EDAs) were proposed as an alternative for traditional evolutionary algorithms in which reproduction operators could rely on information extracted from the population to enable a more effective search. Since information is usually represented as a probabilistic graphic model, the effectiveness of EDAs strongly depends on how accurately such models represent the population. In this sense, models of increasing complexity have been employed by EDAs, with the most successful ones being able to encode multivariate factorizations of joint probability distributions. However, some studies have shown that even multivariate EDAs fail to build accurate models for problems in which there is an intrinsic *pairwise independence* between variables. This study elucidates how pairwise independence impacts the linkage learning procedures of multivariate EDAs and affects their accuracy. First, the necessary conditions for learning additively separable functions are assessed, from which it is shown that extreme multimodality can induce pairwise independence. Second, it is demonstrated that in the presence of pairwise independence the approximate linkage learning procedures employed by many EDAs are not able to retrieve high-order dependences. Finally, in an attempt to infer how likely pairwise independence occur in practical problems, the case of non-separable functions is empirically investigated. For this purpose, the *NK*-model and the Linkage-Tree Genetic Algorithm (LTGA) were used as a study case and a range of usefulness for the LTGA was estimated according to N (problem size) and K (degree of interactions among variables and multimodality). The results indicated that LTGA linkage learning is probably more useful for $K \leq 6$ on instances with random linkages (this range grows with N), and for $K \leq 9$ on instances with nearest-neighbor linkages (this range is stable with N). Outside these ranges, pairwise independence is more likely to occur, which deteriorates models accuracy and impairs LTGA performance.

1. Introduction

Nearly-decomposable systems were conceptually interpreted by Holland as composing the rationale for the performance of simple Genetic Algorithm (GAs) [1,2]. From such ideas the schema theory and the Building-Block (BB) hypothesis were proposed and further extended by Goldberg [3]. In a general sense, Holland and Goldberg's theories rely on reductionist ideas, in which complex optimization problems would be composed of interacting substructures that could increase the efficiency of the search if effectively exploited by GAs [1,2].

The schema theory formalized such ideas and provided an explanation for GAs effectiveness [2–4]. It argues that certain common features shared by high-quality solutions are privileged during the selection step. Therefore, in order to guarantee an efficient search, such features should not be frequently destroyed during reproduction [2]. These ideas grounded the so-called BB hypothesis, from which much has been investigated and also many controversies have arisen [4,5].

Estimation of Distribution Algorithm (EDAs) emerged in that context as an alternative for traditional evolutionary algorithms, in which reproduction operators could rely on explicit probability distributions estimated from the population in an attempt to provide a more efficient search [6–8]. The relative success of early EDAs promoted a more intense interaction between machine learning and evolutionary computation [9], which resulted in EDAs of increasing complexity.

The field progressed with the assumption that the effectiveness of EDAs would mainly depend on how accurately the probabilistic models employed could encode the information contained in the

* Corresponding author.
E-mail addresses: jean@icmc.usp.br (J.P. Martins), acbd@icmc.usp.br (A.C.B. Delbem).

http://dx.doi.org/10.1016/j.swevo.2015.10.001
2210-6502/

population. Therefore, simple models based on univariate factorizations of the joint probability distribution [10–12] were put aside for the sake of more complex bivariate [13,14] and multivariate factorizations [15–18]. However, due to the high computational cost associated with the building (linkage learning) and sampling of multivariate probabilistic graphical models, their appropriate use in the context of EDAs is still an intense area of research [19].

In order to obtain efficient model-building procedures, most EDAs restrict the generality of their models and/or employ approximate algorithms. The extended Compact Genetic Algorithm (eCGA), for example, assumes that the joint probability distribution can be factorized as the product of disjoint multivariate factors [18]. More general models, such as Bayesian networks, have also been applied, but since the building of optimal Bayesian networks is an NP-complete problem [20], EDAs as the Bayesian Optimization Algorithm (BOA) [17] and the Estimation of Bayesian Network Algorithm (EBNA) [16] rely on greedy approximations of optimal Bayesian networks. Due to these constraints, the limits and difficulties of learning accurate probabilistic models has also become a research focus [21].

The accuracy of the probabilistic models employed by EDAs determines how effectively the search can be performed. Therefore, by understanding the required conditions which enable an accurate learning of probability distributions, the robustness of EDAs could also be assessed. One of the first discussions in this direction was proposed by Coffin et al. [22,23], who showed that for concatenated parity functions (CPFs) the hierarchical BOA [24] requires an exponential number of fitness evaluations (in average) to find an optimal solution. The studies that followed indicated *pairwise independence* as the main cause for the difficulty of learning CPFs and the bad performance of EDAs.

Although being hard for learning, CPFs are easy to solve, Chen et al. [25], for example, showed that the eCGA [26] can solve them in polynomial time, although not being able to learn accurate models. Echegoyen et al. [27], on the other hand, showed that optimal Bayesian networks could correctly model the high-order dependences of CPFs, evidencing that such functions do not limit EDAs applicability but only challenges current approximate model-building procedures. Iclanzan [28] proposed a high-order learning procedure, from which the eCGA could model high-order dependences correctly and he also investigated more difficult hierarchical functions composed of pairwise independent variables [29]. The main criticism on these results stands on the artificiality of CPFs and the fact such results do not explain the causes of pairwise independence.

Regarding non-separable functions, Echegoyen et al. [30,31] showed that the degree of interactions between decision variables was also a cause for inaccurate learning. It was shown that as the number of interactions among variables increases, the capacity of EBNA to learn the correct dependences decreases. At a certain degree of interactions a phase transition occurs in which the models produced stop to help the search. These results clearly indicate limits for EDAs applicability in terms of degree of interactions, however, only artificial functions were employed in the investigation and it is not evident how likely such extreme situations occur in practice.

In an attempt to close the gap between the conclusions drawn for artificial functions and their implications when solving more practical problems, some studies have been reported in the literature. Liaw and Ting [32], for example, showed that univariate and bivariate EDAs could outperform more complex algorithms in some instances of the *NK*-model. Martins et al. [33,34], on the other hand, investigated the accuracy of the linkage-tree models produced by the Linkage-Tree Genetic Algorithm (LTGA) and the performance of other EDAs on the Multidimensional Knapsack Problem (MKP). The authors also compared the LTGA performance when employing linkage-trees and random linkage-trees, with no evidence indicating that linkage-tree learning helped to solve the MKP [35]. Sadowski et al. [36] faced similar difficulties when applying the LTGA to the MAX-SAT. According to the literature, such bad results could be caused by a high degree of interactions [30] or the non-uniformity of fitness contributions associated with subsets of variables [37]. However, few studies have followed in such a direction [38,39].

Martins et al. [40,41] investigated the connections between multimodality and pairwise independence. Considering additively separable functions, the increase in multimodality was shown to increase the difficulty of learning, with negative impacts on the accuracy of the models produced. It was argued that in those functions pairwise independence emerge as a consequence of extreme multimodality. Such results contributed to the understanding of how multimodality affects model-building and EDAs in general, a long-term issue in the field that many approaches have tried to circumvent [42–44].

The research on EDAs initially progressed on the perspective of proving the concept, and a large number of experimental comparisons with traditional evolutionary algorithms were performed to show the validity of EDAs. We believe that in order to keep the progress of the field it is important to understand the limitations of current EDAs to: (1) circumvent the limitations with new proposals or (2) identify the characteristic of the problems that can be properly solved by current EDAs.

This study focus on the second alternative and hypothesizes that there is a certain class of optimization problems in which contemporary EDAs excel. Following from the results of Echegoyen et al. [30] and Martins et al. [40,35], we suppose such problems are defined by the degree of interactions among variables and multimodality. Since the degree of interactions was already investigated, we focus on the theoretical relation between multimodality and pairwise independence on additively separable functions. Additionally, in order to infer a range for the effectiveness of EDAs, we investigate experimentally the influence of N and K on the emergence of pairwise independence in *NK*-landscape instances.

The paper is organized as follows. Section 2 formally defines EDAs and reviews some concepts from information theory which are used along the paper. Section 3 summarizes and updates the assumptions made by Martins and Delbem [40], which define conditions for accurate bivariate learning and describes how extreme multimodality can induce pairwise independence. Section 4 analyzes some multivariate EDAs and proves that their model-building procedures are not able to learn high-order dependences in the presence of pairwise independence. Section 5 generalizes the results of the previous sections for non-separable *NK*-landscapes instances, in order to estimate the range of K (degree of interactions and multimodality) in which EDAs excel. Section 6 discusses the results and shows that the LTGA performed significantly better than its randomized version only for instances with small K which indicates that pairwise independence might appear with the increase of K. Section 7 summarizes the paper and discusses general implications.

2. Background

This section introduces methods, concepts and the notation used along the paper. Section 2.1 provides a general overview of EDAs and their classification in terms of the probabilistic models they employ, namely: univariate, bivariate and multivariate. Section 2.2 describes concepts from information theory, such as entropy and mutual information, which are fundamental to the analysis reported in Section 3.

2.1. Estimation of Distribution Algorithms

EDAs are an alternative for traditional evolutionary algorithms that provide a more effective way to search the solution space and at the same time exempt the practitioner of the task of developing problem-specific reproduction operators. Such goals are pursued by the use of probabilistic graphic models (PGMs) that encode the empirical probability distribution underlying the solutions within the population. The PGMs are then sampled to generate new solutions. If accurate models are available such a process should be less disruptive than common crossover and mutation operators [45,6].

Given an unconstrained optimization problem over a solution space composed of N dimensional binary vectors, an optimal solution is defined as

$$\boldsymbol{x}^{\star} = \arg\max_{\boldsymbol{x} \in \{0,1\}^N} f(\boldsymbol{x}). \tag{1}$$

A population $P \subset \{0,1\}^N$ composed of $\lambda \geq 2$ candidate solutions can be described by its underlying probability distribution. Let $\boldsymbol{X} = (X_1, \ldots, X_N)$ be a vector of binary random variables, where assignments $\boldsymbol{x} = (x_1, \ldots, x_N)$ are candidate solutions. Therefore, $p_{\boldsymbol{X}}(\boldsymbol{x})$ is the joint probability distribution, $p_{X_i X_j}(x_i, x_j)$ is a bivariate distribution whereas $p_{X_i}(x_i)$ is the marginal distribution of X_i, for $\forall i,j \in \{1, \ldots, N\}$.

EDAs estimate $p_{\boldsymbol{X}}(\boldsymbol{x})$ from the samples $\boldsymbol{x} \in P$. However, due to the exponential time complexity associated with the estimation of multivariate (N-order) statistics, EDAs limit the generality of their models by using low-order factorizations of the joint probability distribution [21].

A factorization is the product of marginal (possibly multivariate) probability distributions (factors). Let π be a subset of $\{1, \ldots, N\}$, with $|\pi| \leq \sigma$ ($\sigma \ll N$). The subset of random variables referenced by π and denoted $\boldsymbol{X}_\pi$, compose a factor $p_{\boldsymbol{X}_\pi}(\boldsymbol{x}_\pi)$. Therefore, the factorized joint probability distribution is the product of the factors

$$p_{\boldsymbol{X}}(\boldsymbol{x}) = \prod_{\pi \in \mathcal{F}} p_{\boldsymbol{X}_\pi}(\boldsymbol{x}_\pi). \tag{2}$$

The complexity of the EDAs proposed in the literature relate to the PGMs employed and the order (σ) of the largest factors supported by the factorized joint probability distribution they encode. According to σ, EDAs can be classified as univariate ($\sigma = 1$), bivariate ($\sigma = 2$) or multivariate ($\sigma > 2$) [46,19].

2.2. Information theory

One of the fundamental concepts in information theory is *entropy*, which measures the uncertainty associated with a random variable. Consider, for example, a random variable X (shorthand for X_i) that can assume two distinct values $\xi = \{0, 1\}$, each with probability $p_X(x), x \in \xi$.

Definition 2.1 (*Entropy, Kelbert and Suhov [47]*). The entropy $H(X)$ of X measures the expected amount of information gained from the observation of X and is defined on $\mathcal{S}_X = \{x \in \xi : p_X(x) > 0\}$

$$H(X) := -\sum_{x \in \mathcal{S}_X} p_X(x) \log_2 p_X(x). \tag{3}$$

By this definition, the information gain is proportional to the uncertainty. Therefore, the entropy increases as the probability distribution of X tends to uniform, reaching its maximum value when $p_X(x) = |\xi|^{-1}$, $\forall x \in \xi$. For a binary alphabet, the entropy is maximum if $p_X(x) = 1/2$ and minimum if $p_X(x) = 1$.

In the context of linkage learning it is also useful to consider the joint entropy of two random variables (X,Y) (shorthand for any pair X_i, X_j, $\forall i,j \in \{1, \ldots, N\}$).

Definition 2.2 (*Joint entropy, Kelbert and Suhov [47]*). The joint entropy of a binary random vector (X,Y) with joint probability distribution $p_{XY}(x,y)$ is defined as

$$H(X,Y) := -\sum_{x \in \mathcal{S}_X} \sum_{y \in \mathcal{S}_Y} p_{XY}(x,y) \log_2 p_{XY}(x,y). \tag{4}$$

Again, the joint entropy increases as the joint probability distribution of (X,Y) tends to uniform. Therefore, it reaches its maximum value when $p_{XY}(x,y) = |\xi|^{-2}$, $\forall x,y \in \xi$.

Eqs. (3) and (4) allow us to measure the amount of information a random variable conveys about another. For any variables X and Y

$$H(X,Y) \leq H(X) + H(Y).$$

The difference between the left-hand and right-hand sides is called *mutual information*.

Definition 2.3 (*Mutual Information, Kelbert and Suhov [47]*). Given binary random variables X and Y, their mutual information is given by

$$I(X;Y) := H(X) + H(Y) - H(X,Y). \tag{5}$$

By definition, $I(X;Y) \geq 0$, with equality holding if X and Y are independent, i.e. $p_{XY}(x,y) = p_X(x) p_Y(y)$. The mutual information is a measure of statistical dependence but it is not strictly a metric. In the cases a metric is required, the *variation of information* is usually employed [48].

Definition 2.4 (*Variation of information*). Given binary random variables X and Y, the variation of information is

$$VI(X;Y) : = H(X,Y) - I(X;Y) = 2\,H(X,Y) - H(X) - H(Y). \tag{6}$$

The variation of information is $VI(X;Y) \geq 0$ and decreases as the mutual information $I(X;Y)$ increases, i.e. $VI(X;Y) \approx 0$ indicates X and Y are highly dependent variables. In other words, the variation of information measures the independence between random variables, with minimum independence meaning maximum dependence.

Although $VI(X;Y)$ is preferable as a metric, it is easily derived from the mutual information and the joint entropy. For the purpose of our theoretical analysis it is sufficient to use $I(X;Y)$ and $H(X,Y)$ (Section 3) but for the experimental analysis of the range of usefulness of linkage learning, the intensity of linkages is measured by the $VI(X;Y)$ (Section 5).

3. Bivariate linkage learning

Most EDAs (except those with univariate models) rely on measurements of bivariate statistics in some step of their model-building procedures. This section describes how the increase of multimodality impacts such measurements, leading to *pairwise independence* in the extreme cases. The analysis is restricted to additively separable pseudo-Boolean functions and reviews/summarizes some previous results [40].

3.1. Additively separable functions

A function is said additively separable if it can be written as the sum of m sub-functions, each of them acting on a subset $\pi \subset \{1, \ldots, N\}$. The set of all subsets is denoted $\mathcal{F}$.

$$f(\boldsymbol{x}) = \sum_{\pi \in \mathcal{F}} f_\pi(\boldsymbol{x}_\pi), \boldsymbol{x} \in \{0,1\}^N. \tag{7}$$

NK-landscape and MAX-SAT are well-known examples of problems with additively separable objective functions [49].

If, additionally, the sets $\pi \in \mathcal{F}$ are mutually disjoint, optimal solutions for f are composed of the optimal solutions for sub-functions f_π. Deceptive functions are examples of such functions that are commonly used to benchmark EDAs [4].

3.2. Necessary conditions for learning

Let $f : \{0,1\}^N \mapsto \mathbb{R}$ be an additively separable function and $\mathcal{F}$ a set of disjoint subsets of $\{1,\ldots,N\}$. Therefore, f can be decomposed in $m = (N/k)$ sub-functions $f_\pi : \{0,1\}^k \mapsto \mathbb{R}$. An effective EDA for such functions should be able to estimate a joint probability distribution closely related to the separability of the function f, i.e. $p_{\boldsymbol{X}}(\boldsymbol{x}) = \prod_{\pi \in \mathcal{F}} p_{\boldsymbol{X}_\pi}(\boldsymbol{x}_\pi)$. This section describes the necessary conditions for learning the correct factorization of the joint probability distribution for such functions.

Consider the simple case with $m=2$ and $\mathcal{F} = \{\pi_1, \pi_2\}$, which leads to $f(\boldsymbol{x}) = f_{\pi_1}(\boldsymbol{x}_{\pi_1}) + f_{\pi_2}(\boldsymbol{x}_{\pi_2})$. Additionally, assume that X and Y are random variables indexed by π_1, whereas Z is a random variable indexed by π_2. Any linkage learning procedure will only be able to detect the correct linkage between X and Y, instead of X and Z, if the following condition is met:

$$I(X;Y) > I(X;Z). \tag{8}$$

By Definition 5, it follows

$$\begin{aligned} H(X)+H(Y)-H(X,Y) &> H(X)+H(Z)-H(X,Z) \\ H(Y)-H(X,Y) &> H(Z)-H(X,Z). \end{aligned} \tag{9}$$

In the simplest scenario, in which all sub-functions f_π are equal, the entropies H(Y) and H(Z) cancel each other, therefore

$$H(X,Y) < H(X,Z). \tag{10}$$

In a linkage learning procedure based on measurements of bivariate statistics, it is clear that Eqs. (9) and (10) are necessary conditions for a correct modeling, i.e. the pairwise joint entropy should reflect the separability of the objective function. To a certain extent, they also define valid necessary conditions for any linkage learning procedure that builds upon bivariate statistics.

3.3. Multimodality and pairwise independence

In a population generated uniformly at random, inequalities (9) and (10) do not hold. The joint probability distribution of (X,Y) is uniform, therefore the joint entropy is maximum and there is no mutual information. As selection and reproduction are performed, only individuals in regions of better fitness survive, which changes the distribution of solutions in the population. Such a decrease of diversity might decrease the joint entropy between random variables, hence increasing the mutual information and evidencing statistical dependences.

However, for some functions, even after selection and reproduction the decrease of the joint entropy might be too small. In such cases, linkage learning would not identify useful linkages and their use would not improve the search [25,22]. This section suggests multimodality of sub-functions f_π as one of the characteristics responsible for such a difficulty.

Fig. 1 shows two continuous fitness landscapes used to illustrate the effect of multimodality on learning the dependence between the random variables X and Y. Fig. 1(a) shows an example in which the dependence between X and Y can be easily observed from statistics: there are only five local attractors and after selection a large part of the search space becomes uninhabited. Since many pairs of values do not occur in the population anymore, the mutual information between X and Y is increased. Fig. 1(b) illustrates a more difficult situation, with a larger number of local attractors: even after selection the pairs of values are well distributed, which implies a small increase of the mutual information and a more difficult observation of statistical dependences.

These small examples illustrate the intuition underlying the hypothesis that multimodality imposes limitations for linkage learning due to its impact on the difficulty of observing useful linkages. As the degree of multimodality increases, bivariate linkage learning is expected to become less accurate and even infeasible in extreme cases.

In order to evaluate such hypothesis, the next sections deal with the analysis of different separable functions so as to understand the consequences of multimodality on the difficulty of learning. We show that extreme multimodality can lead to pairwise independence and illustrate how difficult common benchmark functions are in comparison to CPFs.

3.4. Measures of difficulty

To understand the impact of multimodality on linkage learning, we must provide ways to measure linkage learning difficulty and compare the measurements obtained for functions of different degrees of multimodality. The linkage learning difficulty can be estimated from conditions (9) and (10).

If function f is composed of sub-functions f_π with different characteristics, the difficulty of learning the separability of the function can be derived from the general condition (9)

$$\begin{aligned} &I(X;Y) > I(X;Z) \\ &0 < I(X;Y) - I(X;Z) \\ &\delta_I = I(X;Y) - I(X;Z). \end{aligned} \tag{11}$$

If, otherwise, function f is composed of equal sub-functions f_π, the difficulty of learning the separability of the function can be

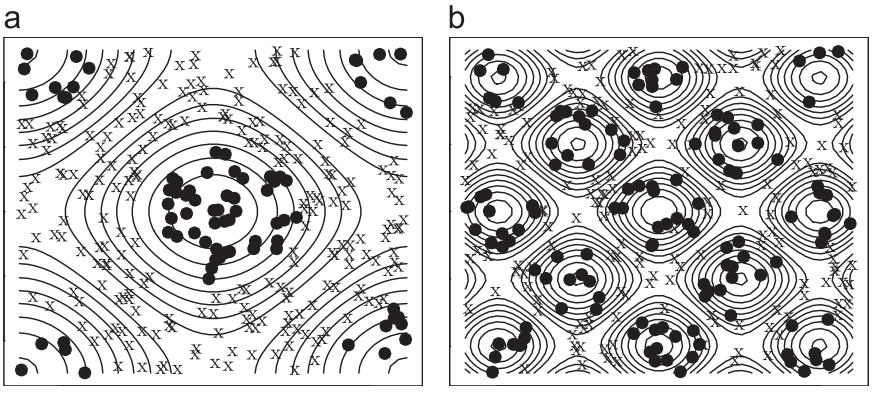


Fig. 1. Population before (crosses) and after selection (circles). According to the number of local attractors, the distribution of values after selection might become closer to uniform and statistical dependences harder to detect. (a) A few local attractors. (b) Many local attractors.

derived from the restricted condition (10)

$$H(X,Y) < H(X,Z)$$
$$0 < H(X,Z) - H(X,Y)$$
$$\delta_H = 1 - H(X,Y)/H(X,Z). \quad (12)$$

We say a function is easy for linkage learning if δ_H or δ_I are large (e.g. $\delta_H \approx 1$).

For the computation of δ_H or δ_I, the probability distribution of pairs (X,Y) (intra-block variables) and (X,Z) (inter-block variables) must be known beforehand. However, before introducing more details on how to infer such distributions let us describe the benchmark functions used as reference.

3.5. Benchmark functions

The functions described in this section are examples of additively separable functions which are defined on mutually disjoint subsets $\pi \in \mathcal{F}$. In order to simplify the theoretical analysis we also assume that π_i refer to a subsequence of $(1,\ldots,N)$ starting at $(ki-k+1)$ and ending at (ki). For example, $\pi_1 = \{1,\ldots,k\}$, $\pi_2 = \{k+1,\ldots,2k\}$ until $\pi_m = \{N-k+1,\ldots,N\}$.

All sub-functions $f_\pi : \mathbb{N} \mapsto \mathbb{R}$ have input produced by an unitation function $u(\boldsymbol{x}_\pi) = \sum_{j \in \pi} x_j$. For the analysis described in next sections, we have chosen four well-known sub-functions: k-trap, k-bipolar, k-parity and k-parity/trap, whose outcomes are illustrated Fig. 2.

mk-traps: The mk-trap function is obtained from the concatenation of k-trap functions. Each k-trap contains one global- and one local-optimum solutions (all-ones and all-zeros, respectively). The fitness of a whole solution $\boldsymbol{x} \in \{0,1\}^N$ is given by the sum of the results produced by each k-trap, which is defined as

$$k\text{-trap}(u) = \begin{cases} k & \text{if } u = k, \\ k-1-u & \text{otherwise.} \end{cases}$$

Fig. 2(a) illustrates the possible outcomes of k-trap (u) for $k=5$. Solutions generated uniformly at random are likely to be closer (in Hamming distance) to suboptimal than the optimal configuration.

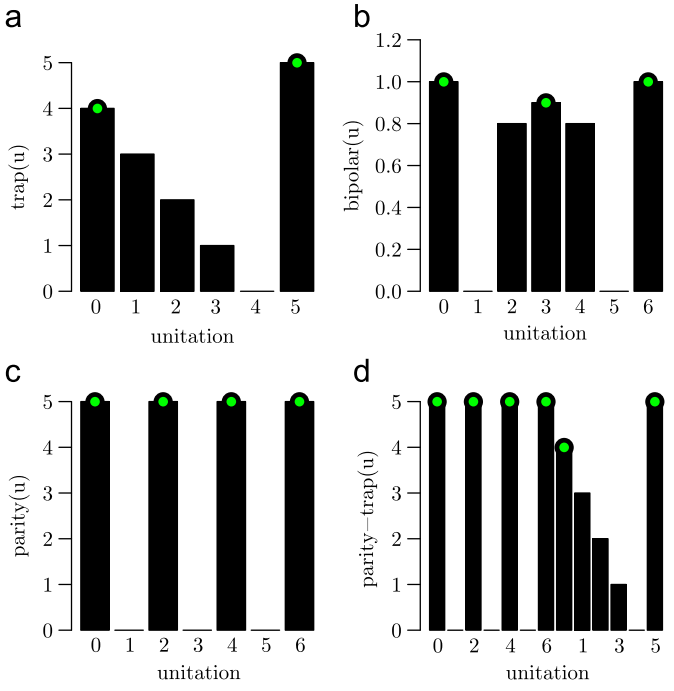


Fig. 2. All possible outcomes of a trap function with $k=5$ (a), a bipolar function with $k=6$ (b), a parity function with $k=6$ and a parity/trap function with $k=6/k=5$ (see [40]).

mk-bipolar: The mk-bipolar is obtained from the concatenation of k-bipolar functions. Each k-bipolar function has two globally optimal configurations and $\binom{k}{k/2}$ locally optimal configurations. For parameters $m=5$ and $k=6$, there are 32 globally optimal solutions (combinations of all-one and all-zero components) and approximately three million locally optimal solutions. The k-bipolar function is defined as follows:

$$k\text{-bipolar}(u) = \begin{cases} 1, & \text{if } |u-k/2| = k/2, \\ 0.9, & \text{if } |u-k/2| = 0, \\ 0.8, & \text{if } |u-k/2| = 1. \end{cases}$$

Fig. 2(b) shows the possible outcomes for a 6-bipolar function.

mk-parity: The mk-parity is obtained from concatenation of m k-parity functions. Differently from the previous functions, it is not deceptive. Each k-parity function contains 2^{k-1} globally optimal configurations and it is defined as follows:

$$k\text{-parity}(u) = \begin{cases} C_{\text{even}}, & \text{if } u \text{ is even,} \\ C_{\text{odd}}, & \text{otherwise, with } C_{\text{even}} > C_{\text{odd}}. \end{cases}$$

Fig. 2(c) shows the possible outcomes for a 6-parity function.

mk-parity/trap: This function is composed of the interleaved concatenation of k-parity and k-trap functions. It mixes two different characteristics, the high multimodality of parity functions and the easier bimodal traps. Fig. 2(d) illustrates the outcomes of two concatenated 6-parity and 5-trap functions.

3.6. Estimates of the difficulty

As previously mentioned, in order to compute the δ_H and δ_I values for the functions defined Section 3.5, the probability distribution underlying the population $P \sim D$ must be known. In general, the estimation of D from P is a hard task, fortunately, for our purposes it is sufficient to consider the distribution D at a certain stage in which linkage learning is essential. This approach is a simpler alternative to the analysis of selection schemes and reproduction operators, however, it also has its limitations, since it can only consider worst case scenarios.

Regarding additively separable deceptive functions, such a stage occurs when solutions for all sub-functions are either in globally- or locally-optima configurations. At this point, the average fitness of the population tends to stop to increase due to the disruptive effects of crossover and mutation. Therefore, linkage learning would highly benefit the progress of the search by increasing crossover effectiveness. Martins and Delbem [40] employed a deterministic Hill-Climbing (dHC) to lead the population directly to such stage and measure the δ_H and δ_I values.

Here, we only show estimates for such measurements based on upper bounds for the entropy. At expense of some accuracy, such upper bounds provide sufficient information to elucidate the relation between multimodality and pairwise independence. Entropy upper bounds are obtained from Jensen's inequality [50]:

$$\left[H(X) = \sum_{x \in \mathcal{S}_X} p_X(x) \log_2 \left(\frac{1}{p_X(x)} \right) \right] \leq \log_2(|\mathcal{S}_X|)$$

where the right-hand side is an upper bound for the entropy. Therefore, to compute an upper bound only the size of the support sets $\mathcal{S}_{XY}$ and $\mathcal{S}_{XZ}$ are needed. Assuming the population is in the stage t previously mentioned, the support sets contains only elements from optimal (local and global) configurations.

Next, δ_I values are derived for the mk-parity/trap and δ_H values are derived for all the other functions. To make clear they are approximations, we refer them by $\tilde{\delta}_H$ and $\tilde{\delta}_I$.

$\tilde{\delta}_{H_t}$ (k-trap): For mk-trap functions, there are only two optimal configurations, e.g. blocks 000000 (local) and 111111 (global) ($k=6$). Therefore, $|\mathcal{S}_{XY}| = 2$. Applying Jensen's inequality to intra-

block variables (*X*,*Y*), we find

$$H(X,Y) \leq [\log_2(2) = 1].$$

Concerning inter-block variables (*X*,*Z*), all pairs of values can occur, therefore, $|\mathcal{S}_{XZ}| = 4$ and Jensen's inequality results in the upper bound:

$$H(X,Z) \leq [\log_2(4) = 2].$$

Using these values we can compute $\tilde{\delta}_{H_t}$, which results in

$$\tilde{\delta}_{H_t} = 1 - H(X,Y)/H(X,Z) = \mathbf{0.5},$$

which, indeed is a tight upper bound for δ_{H_t} [40].

$\tilde{\delta}_{H_b}$ (*k-bipolar*): In this case, there are two globally optimal and $\binom{k}{k/2}$ locally optimal configurations for intra-block variables. Since (*X*,*Y*) can assume any pair of values, $|\mathcal{S}_{XY}| = 4$. Inter-block variables (*X*,*Z*) can also assume any pair of values, therefore, $|\mathcal{S}_{XZ}| = 4$, thus

$$[H(X,Y) = H(X,Z)] \leq [\log_2(4) = 2],$$
$$\tilde{\delta}_{H_b} = 1 - H(X,Y)/H(X,Z) = \mathbf{0}.$$

In this case, $\tilde{\delta}_{H_b}$ only approximates the true value $\delta_{H_b} = 0.005$ shown in [40]. Aside this fact, the large difference between $\tilde{\delta}_{H_t}$ and $\tilde{\delta}_{H_b}$ confirms a well-known result that the *mk*-bipolar is harder for linkage learning than the *mk*-traps [4].

$\tilde{\delta}_{H_p}$ (*k-parity*): The parity function is an extreme example of multimodality, although there are well-defined blocks, they are not identifiable by low-order statistics. There are 2^{k-1} different possible configurations (all optimal, with even unitation). Therefore, there is no *mutual information* or way to distinguish between statistically dependent and independent variables. Using Jensen's inequality, the joint entropy between any pair of variables and the $\tilde{\delta}_{H_p}$ are approximated by

$$[H(X,Y) = H(X,Z)] \leq [\log_2(4) = 2],$$
$$\tilde{\delta}_{H_p} = 1 - H(X,Y)/H(X,Z) = \mathbf{0},$$

which is a tight upper bound and means there is no observable pairwise dependences between *X* and *Y*. However, there are high-order dependences, as shown next.

Assume that $k-1$ dependent variables have been correctly discovered and stored in a set *A*. We only need to decide which variable *Y* is also dependent to *A*. The cardinality of the support set in an optimal configuration equals the number of configurations with even unitation. Therefore, if the correct variable *Y* is chosen, $|\mathcal{S}_{A\cup Y}| = 2^{k-1}$, and

$$H(A \cup Y) \leq [\log_2(2^{k-1}) = k-1]$$

If a wrong variable *Z* is chosen instead, the exact number of possible configurations in $\mathcal{S}_{A\cup Z}$ might contain odd unitations. Therefore, it will be surely greater than $\mathcal{S}_{A\cup Y}$ by a positive integer ϵ, i.e. $|\mathcal{S}_{A\cup Z}| = (2^{k-1} + \epsilon)$ and

$$H(A \cup Z) \leq [\log_2(2^{k-1} + \epsilon)].$$

Since the logarithmic function is monotonically increasing, then $H(A \cup Y) < H(A \cup Z)$, and $\tilde{\delta}_{H_p}$ is greater than zero, which shows the existence of *k*-order dependences.

$\tilde{\delta}_{I_{pt}}$ (*k-parity/trap*): The difficulty of concatenated parity/trap function is very similar to the difficulty of its components. Assuming that *X*, *Y* from a *k*-trap and *Z* from a *k*-parity function, there would be $|\mathcal{S}_{XY}| = 2$ possible configurations, therefore,

$$H(X,Y) \leq [\log_2(2) = 1].$$

For inter-block variables, the support set has size $|\mathcal{S}_{XZ}| = 4$ and the joint entropy is

$$H(X,Z) \leq [\log_2(4) = 2].$$

Since sub-functions are different, δ_I must be used instead of δ_H. Using Jensen's inequality, and considering the support sets of size $|\mathcal{S}_Y| = 2$ and $|\mathcal{S}_Z| = 2$, we have entropies $H(Y) \leq 1$ and $H(Z) \leq 1$, therefore

$$\tilde{\delta}_{I_{pt}} = I(X;Y) - I(X;Z),$$
$$\tilde{\delta}_{I_{pt}} = H(Y) - H(X,Y) - H(Z) + H(X,Z) = \mathbf{1}.$$

Since $\tilde{\delta}_{I_{pt}}$ is maximum, we can conclude that it is easy to identify statistically dependent variables from trap functions. However, if we had considered *X* and *Y* from a parity function and *Z* from a trap function, the situation would have been the opposite. In summary, in a problem with *k*-parity and *k*-trap sub-functions, each of them keep their original linkage learning difficulty, *k*-trap variables are identified as linkages while *k*-parity variables are not, due to pairwise-independence.

Fig. 3 shows the true δ_H values found by Martins and Delbem [40] considering different values of *k*. The difficulty of the *mk*-trap is exactly the same obtained by Jensen's inequality; besides, it does not grow with *k*. The *mk*-bipolar, on the other hand, it is slightly larger than zero for $k = 6 (\delta_{H_b} \approx 0)$, but approaches zero when *k* grows. The learning of *mk*-parity from bivariate statistics is infeasible for any *k* ($\delta_{H_p} = 0$).

This section showed how multimodality impacts the difficulty of accurately learning bivariate linkages and can, in extreme cases, lead to pairwise independence, making linkage learning from bivariate statistics infeasible. Next section analyzes how pairwise independence impacts more complex EDAs in which multivariate statistics are also taken into consideration.

4. Multivariate linkage learning

The limitations of bivariate linkage learning have been constantly reported in the literature as motivation for the use of more complex probabilistic graphic models, in which multivariate dependences could be modeled. However, many studies have shown that some multivariate EDAs, which use Bayesian networks as probabilistic models, have their performance undermined by the presence of pairwise independence.

This section demonstrates how the approximation algorithms usually employed to build multivariate probabilistic models have their accuracy hampered by the presence of pairwise independence. Although the probabilistic graphical models support multivariate dependences, most common linkage learning procedures make a strong assumption that does not hold in the presence of pairwise independence and limit the variety of models that can be produced. This assumption can be stated as:

"High-order dependences imply low-order dependences".

Next, we analyze the impact of such assumption in the linkage learning procedures of different multivariate EDAs in the absence of pairwise dependences. It is assumed the same notation of the previous section, but now *X* and *Y* are *pairwise independent* random variables that compose a high-order linkage, whereas *Z* represents any other random variable from a different sub-function.

4.1. Extended Compact Genetic Algorithm

The extended Compact Genetic Algorithm (eCGA) [51] was one of the first EDAs in which high-order dependencies among decision variables could be modeled. Its approach factorizes the joint probability distribution in the product of multivariate marginal distributions. Assume that $\mathcal{F}_{\text{eCGA}}$ is a set of subsets (linkages), in which every $\pi \in \mathcal{F}_{\text{eCGA}}$ contains $|\pi|$ variables. The factorized joint probability distribution is a multivariate marginal product model

$$p_{\boldsymbol{X}}(\boldsymbol{x}) = \prod_{\pi \in \mathcal{F}_{\text{eCGA}}} p_{\boldsymbol{X}_\pi}(\boldsymbol{x}_\pi). \tag{13}$$

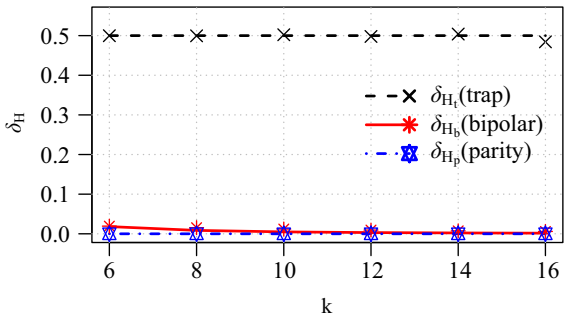


Fig. 3. δ_H shows the k-trap linkage learning difficulty is stable as k grows. For k-bipolar, whose degree of multimodality also depends on k, δ_{H_b} approximates zero as k increases (see [40]).

The eCGA linkage learning procedure relies on two measures: (1) the *Model Complexity* (MC) and (2) the *Compressed Population Complexity* (CPC). The MC quantifies the model representation size in terms of the number of bits required to store all the marginal probabilities

$$MC = \log_2(\lambda+1) \sum_{\pi \in \mathcal{F}_{eCGA}} (2^{|\pi|}-1), \tag{14}$$

The CPC, on the other hand, quantifies the data compression in terms of entropy of the marginal distribution over all partitions, where λ is the population size and $H(\pi)$ is the joint entropy of the variables in π

$$CPC = \lambda \sum_{\pi \in \mathcal{F}_{eCGA}} H(\pi). \tag{15}$$

The linkage learning in eCGA works as follows: (1) Insert each variable in a cluster, (2) compute $CCC = MC + CPC$ of the current linkage sets, (3) verify the increase on CCC obtained by joining every pairs of clusters, (4) effectively joins the clusters with highest CCC improvement. This procedure is repeated until no CCC improvements are possible.

Let us investigate the impact of *pairwise independence* on eCGA's linkage learning.

Theorem 4.1. *In the presence of pairwise independence, the eCGA greedy linkage learning procedure cannot correctly learn high-order linkages.*

Proof. Let us compare the CCC values when: (1) X and Y are joined, (2) X and Z are joined. The high-order linkage set can be identified only if X and Y are joined instead of X and Z. In the beginning $\mathcal{F}_{eCGA}$ contains N singleton sets π, then

$$MC = \log_2(\lambda+1)N.$$

Since $|\pi_i| = 1, \forall \pi_i \in \mathcal{F}_{eCGA}$, then $H(\pi_i) = H(X_i)$, hence:

$$CPC = \lambda \sum_{i=1}^{N} H(X_i).$$

Without loss of generality, assume $X_1 = X, X_2 = Y$ and $X_N = Z$. MC_{XY} and CPC_{XY} (MC_{XZ} and CPC_{XZ}) are the measurements obtained by joining the variables X and Y (X and Z). After one joining, there are $(N-2)$ singletons and one subset of size two, therefore

$$MC_{XY} = \log_2(\lambda+1)(N+1).$$

The $MC_{XY} = MC_{XZ}$ because it only accounts for the size of the linkage sets. The CPC, on the other hand, depends of the subsets being joined. Let us consider first the pair X, Y. In this case, the CPC_{XY} value changes to account for the joint entropy $H(X,Y)$,

$$CPC_{XY} = \lambda \left(H(X,Y) + \sum_{i=3}^{N} H(X_i) \right).$$

After joining the pair X, Z, the CPC_{XZ} can be written as

$$CPC_{XZ} = \lambda \left(H(X,Z) + \sum_{i=2}^{N-1} H(X_i) \right).$$

By definition $H(X,Y) \leq H(X) + H(Y)$, with equality holding only if X and Y are independent. Since we are assuming *pairwise independence*, $H(X,Y) = H(X) + H(Y)$ and $H(X,Z) = H(X) + H(Z)$, therefore

$$CPC_{XY} = CPC_{XZ} = \lambda \sum_{i=1}^{N} H(X_i).$$

Now, compare the CCC before and after the possible joining. Since CPC did not change with the new clusters, CCC_{XY} and CCC_{XZ} only depends on the MC values, which are equal $MC_{XY} = MC_{XZ}$. Both cases are equal and it follows:

$$\begin{aligned} &MC < MC_{XY}, \\ &\log_2(\lambda+1)N < \log_2(\lambda+1)(N+1), \\ &N < (N+1). \end{aligned}$$

This result indicates that extended Compact Genetic Algorithm will not join neither $\{X, Y\}$ or $\{X, Z\}$, because both situations make the CCC worst than the previous model (initial univariate factorization). Therefore, the multivariate eCGA cannot correctly model high-order linkages in the presence of *pairwise independence*.□

4.2. Bayesian networks models

Bayesian networks are directed acyclic graphs, with nodes representing variables and edges representing conditional probabilities between pair of variables. The value assumed by a variable X_i can be conditioned on $|\pi_i|$ other variables, where π_i refer to the "parents" of X_i. Bayesian networks encode the following factorized joint distribution:

$$p_{\boldsymbol{X}}(\boldsymbol{x}) := \prod_{i=1}^{N} p(X_i | \pi_i), \tag{16}$$

in which the structure (edges) and the parameters (conditional probabilities) are estimated from the population.

The problem of finding an optimal Bayesian network is known to be NP-complete [20], therefore, approximation algorithms must be used in practice. In the context of EDAs, such algorithms are usually defined by: (1) a scoring metric, (2) a search procedure [19]. The search is performed on the space of direct acyclic graphs with the scoring metric indicating the quality of candidate graphs.

A common search procedure starts with a empty network, containing nodes but no edges, and, at every step, adds the edge which better improves the scoring metric. Although such approach supports operations like edge deletion and reversal, they are usually neglected for the sake of time efficiency. Next, the scoring metrics used in the Bayesian Optimization Algorithm (BOA) [17] and Estimation of Bayesian Network Algorithm (EBNA) [16] are evaluated.

4.2.1. Bayesian Optimization Algorithm

The BOA [17,52], as originally proposed, uses a Bayesian-Dirichlet metric with no *prior* information as scoring metric, which is called K2 [53]. The S_{K2} is computed as the product of the

score-contributions, $g_{K2}(X_i,\pi_i)$, of all variables X_i:

$$S_{K2}(B,P)=p(B)\prod_{i=1}^{N} g_{K2}(X_i,\pi_i) \tag{17}$$

Given a variable X_i, its score-contribution $g_{K2}(X_i,\pi_i)$ consider all $q_i \le 2^{|\pi_i|}$ instantiations of its parents and all $r_i \le 2$ instantiations of X_i in the population P, as follows:

$$g_{K2}(X_i,\pi_i)\coloneqq \prod_{j=1}^{q_i}\frac{(r_i-1)!}{(\lambda_{ij}+r_i-1)!}\prod_{t=1}^{r_i}(\lambda_{ijt})! \tag{18}$$

where λ_{ijt} is the number of solutions $\boldsymbol{x}\in P$ in which $X_i=v_t$ and its parents $\boldsymbol{X}_{\pi_i}=w_{ij}$, with $v_t\in\{0,1\}$ and $w_{ij}\in\{0,1\}^{q_i}$. Lastly, for each parental configuration j there are r_i configurations for X_i, $\lambda_{ij}=\sum_{t=1}^{r_i}\lambda_{ijk}$ counts them all.

Since π_i might be greater than two, S_{K2} correctly accounts for multivariate statistics. However, the greedy and sequential addition of edges does not allow a recovery from bad decisions during the structural learning of the Bayesian network. In other words, after one edge is added to the graph it cannot be further changed (removed or reversed). In such a scenario, if the greedy decision concerning initial edges (Y,X) and (Z,X) is incorrect, the accuracy of the whole network is compromised.

Theorem 4.2. *In the presence of pairwise independence, the greedy and sequential addition of edges (no deletions or reversals) based on S_{K2} cannot guarantee the correct learning of high-order linkages.*

Proof. It is assumed that X and Y compose a higher-order linkage (but are pairwise independent) whereas X and Z are independent. To evaluate the impact of pairwise independence on the network produced by BOA we compare the score-contributions obtained by the addition of both edges (Y,X) and (Z,X).

At the first step of the greedy edge addition, there is no edge at the graph, therefore, the addition of an edge introduce only one parent. In the case of binary variables ($r_i\le 2$) this implies $q_i\le 2$. Therefore, λ_{ijt} is the number of solutions in which $X_i=v_t$, with $v_t\in\{0,1\}$ and its unique parent $\boldsymbol{X}_{\pi_i}=w_j$, with $w_j\in\{0,1\}$. Since there is no concrete population P in this analysis, λ_{ijt} must be computed from its expected value.

Assume that $X_i=X$, $\boldsymbol{X}_{\pi_i}=Y$ and a population size λ. Due to pairwise independence, $p_{XY}(x|y)=p_X(x)$, then

$$\mathbb{E}(\lambda_{ijt})=\lambda p_{XY}(v_t|w_j)=\lambda p_X(v_t) \tag{19}$$

$$\mathbb{E}(\lambda_{ij})=\sum_{t=1}^{r_i}\mathbb{E}(\lambda_{ijt})=\lambda \tag{20}$$

The algorithm will only produce an accurate model if (Y,X) is chosen instead of (Z,X). Let us assume, Y and Z are the possible parents of X. The addition of (Y,X) and (Z,X) have score-contributions $g(X,Y)$ and $g(X,Z)$, respectively.

$$g_{K2}(X,Y)=\prod_{j=1}^{q}\frac{(r-1)!}{(\mathbb{E}(\lambda_{ij})+r-1)!}\prod_{t=1}^{r}\mathbb{E}(\lambda_{ijt})!=\prod_{j=1}^{2}\frac{1}{(\lambda+1)!}\prod_{t=1}^{2}(\lambda p_X(v_t))! \tag{21}$$

$$g_{K2}(X,Z)=\prod_{j=1}^{2}\frac{1}{(\lambda+1)!}\prod_{t=1}^{2}(\lambda p_X(v_t))! \tag{22}$$

The correct edge should produce a higher score-contribution, i.e. $g_{K2}(X,Y)>g_{K2}(X,Z)$. However, since the condition is not met, there is no guarantee that the correct edge (Y,X) would be chosen and no guarantees that the high-order linkage would be found by BOA. In summary, in the presence of pairwise independence, if the search procedure used to build the Bayesian network only consider greedy edge additions, the score-contributions are not sufficient to guarantee the retrieving of high-order dependences.□

4.2.2. Estimation of Bayesian Network Algorithm

The EBNA [16] uses the Bayesian information criterion (BIC) [54] to score the quality of the networks. Differently from BOA, the EBNA do not imposes a limit for the number of parents. Simpler models are favored by introducing a penalty term into the scoring metric.

In summary, the S_{BIC} is computed from the sum of all score-contributions $g_{BIC}(X_i,\pi_i)$, and a penalty term $\rho(B)$.

$$S_{BIC}(B,P)=\left(\sum_{i=1}^{N} g_{BIC}(X_i,\pi_i)\right)-\rho(B), \tag{23}$$

where $\rho(B)=\frac{\log\lambda}{2}\sum_{i=1}^{N}(r_i-1)q_i$ is a penalty term which privileges less complex networks [54]. Each score-contribution is defined as follows:

$$g_{BIC}(X_i,\pi_i)=\sum_{j=1}^{q_i}\sum_{t=1}^{r_i}\lambda_{ijt}\log\left(\frac{\lambda_{ijt}}{\lambda_{ij}}\right). \tag{24}$$

Theorem 4.3. *In the presence of pairwise independence, the greedy and sequential addition of edges (no deletions or reversals) based on S_{BIC} cannot guarantee the correct learning of high-order linkages.*

Proof. Following the same approach of the previous section, the score-contributions can be rewritten according to the expected values $\mathbb{E}(\lambda_{ijt})$ and $\mathbb{E}(\lambda_{ij})$. The addition of the edge (Y,X) has a score-contribution

$$g_{BIC}(X,Y)=\sum_{j=1}^{q}\sum_{t=1}^{r}\mathbb{E}(\lambda_{ijt})\log\left(\frac{\mathbb{E}(\lambda_{ijt})}{\mathbb{E}(\lambda_{ij})}\right) \tag{25}$$

$$g_{BIC}(X,Y)=\sum_{j=1}^{q}\sum_{t=1}^{r}\lambda p_X(v_t)\log p_X(v_t). \tag{26}$$

Analogously, the addition of edge (Z,X) leads to a score-contribution

$$g_{BIC}(X,Z)=\sum_{j=1}^{q}\sum_{t=1}^{r}\lambda p_X(v_t)\log p_X(v_t). \tag{27}$$

For a correct linkage identification, the following condition: $g_{BIC}(X,Y)>g_{BIC}(X,Z)$ should be satisfied. However, it does not occur. Therefore, in the presence of pairwise independence, there is no way to guarantee that the correct edge (Y,X) will produce a higher score-contribution than other edge (Z,X). As a result, the high-order linkage containing $\{X,Y\}$ is not guaranteed to be found using S_{BIC}.□

It is important to make clear that these results holds for the sequential greedy addition of edges, which provides only an approximate to the optimal Bayesian network and it is sometimes referred to as B algorithm [55]. Indeed, Echegoyen et al. [27,56] have shown that, if globally optimum Bayesian networks are used, then correct linkages can be found even in the presence of pairwise independence.

In summary, although Bayesian networks are able to model multivariate factorizations of the joint probability distribution, the approximate algorithms used to build these networks, in the context of EDAs, can significantly limit the accuracy of the models produced. Unfortunately, the use of more robust algorithms would incur in a considerable increase of running-time for EDAs, a topic that should be further investigated.

4.3. Linkage-tree Genetic Algorithm

The LTGA is an EDA that uses a hierarchical linkage model called linkage-tree. Differently from most EDAs, the LTGA does not rely on model sampling to generate new solutions. Instead, it performs a search in the hierarchical neighborhoods defined by its linkage-tree model using multi-parent crossover as basic operator.

If an accurate linkage-tree is available, the LTGA have been shown to perform very well and it can still progress with poor linkage-trees due to its reproduction operator. Such a robustness is not common in EDAs that rely only in model sampling, indeed the LTGA has shown to outperform many of the previous state-of-the-art EDAs [57,48,58].

Linkage-trees are built using an agglomerative hierarchical clustering algorithm called Unweighted Pair Grouping Method with Arithmetic-mean (UPGMA) [59,60]. The UPGMA requires a distance matrix containing all pairwise distances between random variables. In the context of the LTGA, the metric is usually based on variations of the mutual information [61]. Without loss of generality, let us assume that $d(X,Y) = I(X;Y)$ if X and Y are singletons.

Initially, each variable is a singleton cluster. The first step joins the clusters with highest mutual information, producing a new cluster $C = \{X, Y\}$. The distance between C and the remaining clusters is approximated by the *average linkage* of their elements. For example, the distance between C and another variable Z is defined as (see Ref. [59])[1]:

$$d_{\text{avg}}(C,\{Z\}) = \frac{|X|\,d(X,Z) + |Y|\,d(Y,Z)}{|C| + |Z|}.$$

Such algorithm produce *linkage-trees* whose subtrees should contain subsets of statistically dependent variables. Fig. 4 illustrates a linkage-tree linking four variables, which can be represented by the subsets in $\mathcal{F}_{\text{LTGA}}$.

The root node is not part of the $\mathcal{F}_{\text{LTGA}}$.

$$\mathcal{F}_{\text{LTGA}} = \{\{1\},\{2\},\{3\},\{4\},\{1,2\},\{3,4\}\}.$$

By construction, a subset $\pi \in \mathcal{F}_{\text{LTGA}}$ should refer to statistically dependent variables. However, this is not guaranteed in the presence of pairwise independence.

Theorem 4.4. *In the presence of pairwise independence, the LTGA linkage learning procedure cannot guarantee the learning of high-order linkages.*

Proof. A linkage tree can only be correctly produced if condition $I(X;Y) > I(X;Z)$ is met. However, in the presence of pairwise independence

$$I(X;Y) > I(X;Z)$$
$$H(X)+H(Y)-H(X,Y) > H(X)+H(Z)-H(X,Z)$$
$$H(Y)-H(X,Y) > H(Z)-H(X,Z)$$

Due to pairwise independence $H(X,Y) = H(X)+H(Y)$, then

$$H(Y)-H(X)-H(Y) > H(Z)-H(X)-H(Z)$$
$$-H(X) > -H(X).$$

Since the initial condition leads to a contradiction, the LTGA cannot guarantee the learning of high-order linkages in the presence of pairwise independence.□

The LTGA reproduction operator (Algorithm 1) requires a family of subsets $\mathcal{F}_{\text{LTGA}}$ and a population P. Each solution $\boldsymbol{x}^i \in P$ is modified as follows: for each subset $\pi \in \mathcal{F}_{\text{LTGA}}$, a random parent $\boldsymbol{x}^j \in P$ is chosen as donor for the genetic material $\boldsymbol{x}^j_\pi$, i.e. $\boldsymbol{x}^i_\pi \leftarrow \boldsymbol{x}^j_\pi$. If the new $\boldsymbol{x}_i$ has better fitness than its previous version, the algorithm continues to the next subset in $\mathcal{F}_{\text{LTGA}}$, otherwise the exchange is undone.

[1] The UPGMA implementation used in this study was based on the `C clustering library`, which can be found at ⟨http://bonsai.hgc.jp/~mdehoon/software/cluster/software.htm⟩.

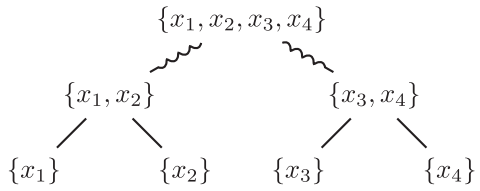


Fig. 4. A linkage tree and the possible subsets that can be obtained.

Algorithm 1. LTGA reproduction operator.

Require: a family of subsets $\mathcal{F}$ and a population P
Ensure: an updated population P
for all $\boldsymbol{x}^i$ in P **do**
  for all π in $\mathcal{F}_{\text{LTGA}}$ **do**
    choose a random $\boldsymbol{x}^j \in P : \boldsymbol{x}^i \neq \boldsymbol{x}^j$
    $f_{\boldsymbol{x}^i} \leftarrow f(\boldsymbol{x}^i)$ {Store the current $f(\boldsymbol{x}^i)$}
    $t_{\boldsymbol{x}^i_\pi} \leftarrow \boldsymbol{x}^i_\pi$ { Temporarily store $\boldsymbol{x}^i_\pi$}
    $\boldsymbol{x}^i_\pi \leftarrow \boldsymbol{x}^j_\pi$
    if $f(\boldsymbol{x}^i) < f_{\boldsymbol{x}^i}$ **then**
      $\boldsymbol{x}^i_\pi \leftarrow t_{\boldsymbol{x}^i_\pi}$ {If $\boldsymbol{x}_i$ has worsened, undo}
return P

Only non-deteriorating moves are performed by the LTGA, which makes it similar to a local search over the neighborhood defined by $\mathcal{F}_{\text{LTGA}}$. Therefore, the LTGA is able to improve solutions even if poor linkage-trees are available, or, in an extreme case, if random linkage-trees are employed.

The Random Linkage-Tree Genetic Algorithm (rLTGA) is a version of the LTGA which uses the same reproduction operator but relies on random pairwise distances to build its linkage-trees. It is straightforward to see that the LTGA will outperform the rLTGA only in cases where linkage learning can produces accurate models to guide the search. Therefore a comparison between the LTGA and rLTGA can provide evidences for the limits of current linkage learning procedures (based on bivariate statistics) in practical optimization problems.

5. Empirical analysis

In the previous sections we have analyzed two main points. Section 3 showed that, regarding additively separable functions, multimodality may considerably impact the observation of statistical dependences between random variables that should indeed be seen as dependent. In other words, multimodality may be one of the causes of pairwise independence. Such a characteristic may limit the usefulness of linkage learning to problems in which the multimodality degree does not exceed a certain limit. Section 4 analyzed the impact of pairwise independence on multivariate EDAs. It was shown that the approximate linkage learning procedures implemented by some state-of-the-art multivariate EDAs cannot find high-order linkages in the presence of pairwise independence, a fact that explains previous bad results concerning parity functions [25]. In summary, multimodality contributes to the difficulty of learning and most multivariate EDAs are not robust to pairwise independence.

Other characteristics of a function might contribute to its difficulty of learning and induce pairwise independence, for example, the degree of interactions per variable [31]. Considering multimodality and the degree of interactions, there must be a range of usefulness in which linkage learning produce useful models.

Outside such a "range of usefulness" we expect EDAs not to perform well. Although it is not possible to find such a range for the general case, it is possible for instances of the *NK*-model, in which K defines both multimodality and degree of interactions and N defines the problem size.

This section investigates the range of usefulness for instances of the *NK* model with increasing multimodality and degree of interactions (K) and increasing dimensionality (N). The experiments compare the benefits of using, or not, linkage learning in the context of the LTGA and the rLTGA, which uses random linkage-trees. The methodology relies on the fact that, if linkage learning were always useful, the LTGA would never be outperformed by the rLTGA. Therefore, by identifying the range of K and N in which the LTGA performs better than rLTGA we are at the same time estimating the range of usefulness of linkage learning in the context of the *NK* model.

5.1. The NK-landscape model

The *NK*-landscape is a well-known model for building unconstrained pseudo-Boolean functions with parameterized difficulty (degree of interactions, multimodality) [49]. The instances are defined over binary search spaces $\boldsymbol{x} \in \{0,1\}^N$, where every decision variable x_i interacts with K other variables:

$$\pi_i \subset \{1,\ldots,N\}\backslash\{i\}.$$

NK-landscape instances can be randomly generated for fixed parameters N and K. At first, for each variable x_i, a set π_i is randomly chosen from $\{1,\ldots,N\}$. The subset of variables $i \cup \pi_i$ has size $K+1$, therefore, in the binary case, it can assume 2^{K+1} different configurations. For each configuration, a contribution $f_i(x_i,\pi_i)$ is randomly chosen within the $(0,1]$ interval. The average contribution of all variables provides the fitness of a candidate solution $f_{NK}(\boldsymbol{x}) = (\sum_{i=1}^{N} f_i(x_i,\pi_i))/N$. Globally optimum solutions are defined as

$$\boldsymbol{x}^{\star} = \arg\max_{\boldsymbol{x}\in\{0,1\}^N} f_{NK}(\boldsymbol{x}) \tag{28}$$

According to the aspects discussed in this paper, K is the most important parameter, as it defines the degree of interactions among decision variables and the multimodality of the fitness landscape. In summary, as K grows, the algorithm effectiveness in finding global optima solutions should decrease considerably [49]. Since we have associated pairwise independence to multimodality and degree of interactions, we expect the LTGA and rLTGA to perform similarly as K grows.

Another relevant aspect commonly explored in the literature is the choice of linked variables π_i. In some studies π_i variables are not randomly chosen, but set to be subsequences of $(1,\ldots,N)$, e.g. if $\pi_i = \{i+1,\ldots,i+K\}$ then $\pi_{i+1} = \{i+2,\ldots,i+K+1\}$ [62]. This approach is commonly referred to as *nearest-neighbor linkages*. Instances with nearest-neighbor linkages maintain some similar characteristics to those with random linkages but, in fact, are easier to solve [49].

5.2. Settings of the experiments

The performances of LTGA and rLTGA were compared on *NK*-landscape instances with $K = 2, 5, 10, 15$, problem sizes $N = 100, 500, 1000$ and population sizes $\lambda = 100, 500, 1000$. Additionally, three other algorithms were evaluated:

1. GA (with uniform crossover and mutation);
2. LTGA;
3. LTGA+ (LTGA with mutation);
4. rLTGA (LTGA with random linkage-models)
5. rLTGA+ (rLTGA with mutation).

In all cases, mutation flips each variable with probability $1/N$. The insertion of duplicate solutions in the population was prohibited so as to maintain the diversity of solutions (preliminary experiments showed it improves performance in long-runs). The GA was implemented in a steady-state fashion, with uniform crossover and binary tournament selection (see Chu and Beasley GA [63]).

For each triple (K,N,λ), the algorithms were allowed to perform $\theta = N \times 10^3$ fitness evaluations. Since θ does not depend on the population size λ, it is possible to assess if LTGA performs better with large or small populations. For each configuration, one hundred runs were performed with different seeds. The resulting sample averages were compared by the Wilcoxon–Mann–Whitney hypothesis test and we conclude the LTGA has outperformed the rLGA only if p-value $< \alpha$, with $\alpha = 0.001$.[2]

Although five algorithms were evaluated, we provide p-values only for the comparison between the LTGA and the rLTGA. The other algorithms were included for the following reasons: the GA is a standard reference, whereas the LTGA+ and rLTGA+ evidence the impact of mutation. The results are described according to the following criteria: (1) Which algorithm found the best solution: LTGA or rLTGA? (2) How did mutation impact on the results? (3) How did population size impact on the results?

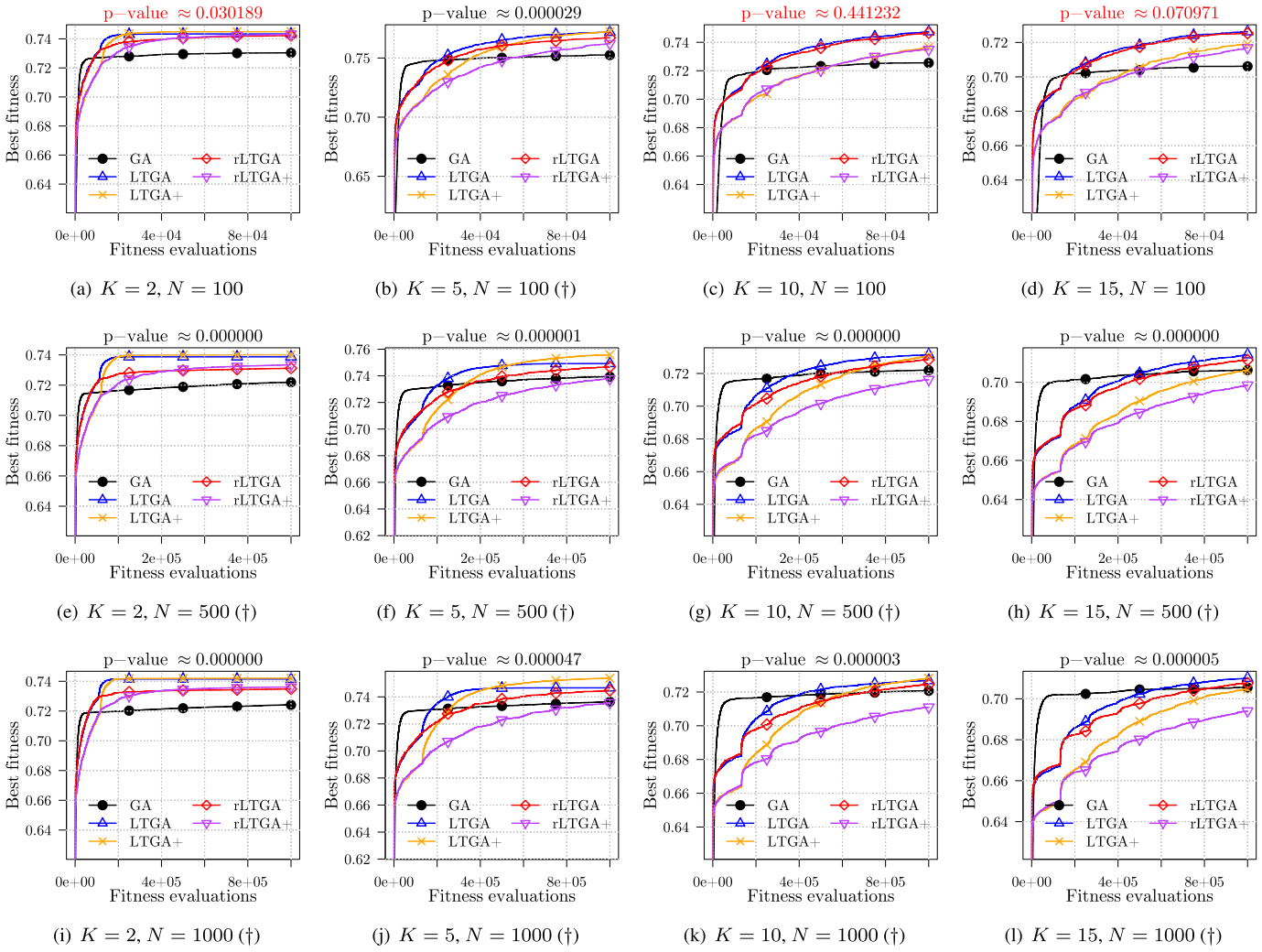
5.3. Effectiveness results

This section describes the results for different instances of the *NK*-model. The p-values obtained from one-sided Wilcoxon–Mann–Whitney hypothesis tests are shown above each figure and a symbol † is written in the respective caption if p-value $< \alpha$, the LTGA has outperformed the rLTGA.

Fig. 5 shows the results for a population size $\lambda = 100$. For the smaller instances (N=100) [Fig. 5(a)–(d)], the LTGA significantly outperformed the rLTGA only for K=5, but both found considerably better solutions than GA. Mutation had positive effects only for small $K \leq 5$. The GA was the fastest algorithm to reach high-quality solutions, but it stuck in suboptimal regions and solution improvements ceased to occur. For the medium-sized instances (N=500) [Fig. 5(e)–(h)], the LTGA outperformed the rLTGA in all cases, although with a small margin for $K \geq 10$. The positive effects of mutation ceased to occur for $K \geq 10$. The GA reached high-quality solutions very fast but ceased to progress very early. For highly multimodal instances (K=10,15), GA's results were similar to those achieved by the other algorithms. For the largest instances (N=1000) [Fig. 5(i)–(l)], the pattern of the results was very similar, the LTGA outperformed the rLTGA in all cases. Mutation impacted positively for $K \leq 5$ but it lost effectiveness for $K \geq 10$. The GA behaved similarly as in the previous case (N=500), but its results were much closer to the LTGA results for $K \geq 10$. The LTGA lost some power as N scaled, while the GA did not suffer much in this aspect (see their best fitness).

In general, for a population size $\lambda = 100$, the benefits of using linkage learning and mutation decreased considerably with the increase of K. While the GA performed stably and mostly independently of problem size, all LTGA variants showed a decrease in the solution quality as the problem size increased (their advantage in relation to the GA decreased). Such a decrease was more evident for $K \geq 10$ [Figs. 5(d), (h) and (l)]. It is interesting to note the differences between the evolutionary patterns produce by GA and LTGA. While the first relies mostly on small improvements, the second relies on large jumps of fitness to evolve solutions.

[2] The p-values above each figure were rounded to six decimal places.



(a) $K = 2, N = 100$ (b) $K = 5, N = 100$ (†) (c) $K = 10, N = 100$ (d) $K = 15, N = 100$

(e) $K = 2, N = 500$ (†) (f) $K = 5, N = 500$ (†) (g) $K = 10, N = 500$ (†) (h) $K = 15, N = 500$ (†)

(i) $K = 2, N = 1000$ (†) (j) $K = 5, N = 1000$ (†) (k) $K = 10, N = 1000$ (†) (l) $K = 15, N = 1000$ (†)

Fig. 5. Average trajectory of the best fitness of each algorithm along $\theta = N \times 10^3$ fitness evaluations. All experiments with $K = 2, 5, 10, 15$ and $N = 100, 500, 1000$ were performed with a fixed $\lambda = 100$.

Fig. 6 show the results for a population size $\lambda = 500$. For the smaller instances (N=100) [Figs. 6(a)–(d)], the LTGA outperformed the rLTGA for $K \leq 5$ but lost its benefits for $K \geq 10$. Mutation did not exert positive effects even for small K and both LTGA+ and rLTGA+ found low-quality solutions. GA had difficulties in finding high-quality solutions for $K \geq 10$. For the medium-sized instances (N=500) [Figs. 6(e)–(h)], the LTGA outperformed the rLTGA, except for K=15. Mutation, again, did not provide positive results. The great surprise was the performance of the GA, which surpassed all LTGA variants for $K = 5, 10, 15$. For the largest instances (N=1000) [Figs. 6(i)–(l)], the same situation occurred, the LTGA outperformed the rLTGA in all cases, except for K=15. Mutation was of no help, except for K=2 and the GA achieved the best solutions for all instances but K=2.

In general, for a population size $\lambda = 500$, mutation did not help the search, except for the easier instances (K=2). Although the LTGA outperformed the rLTGA in most cases (except for K=15), both algorithms found very low-quality solutions in comparison to the results of GA. The LTGA showed again its stairway-like evolutionary fitness pattern, but due to the increase in the population size did not had the time reach the high fitness increases ("jumps"). Since only a few "jumps" occurred, very poor solutions were found [Figs. 6(c) and (d)].

From these results, it is reasonable to conclude LTGA variants do not work well with large populations, which is a surprising fact among EDAs [64]. In summary, with a limited number of fitness evaluations ($\theta = N \times 10^3$), LTGA performed better with small populations, whereas the GA was less sensitive to λ and found solutions of similar quality in both configurations $\lambda = 100, 500$, being the best overall with $\lambda = 500$.

Fig. 7 shows the results for a population size $\lambda = 1000$. As expected from the previous results, the relative performance of the LTGA and rLTGA was even worse. First, there was no case in which both algorithms differed significantly. Second, the GA outperformed all LTGA variants by a considerable margin. As in the previous experiment ($\lambda = 500$), mutation did not help the search in any configuration, i.e. LTGA+ and rLTGA+ behaved similarly and worse than versions without mutation. We may argue that, with large populations, there is no need for additional diversity, therefore instead of improving the search mutation only slows down the progress. The LTGA's stairway-like pattern of fitness increase did not appear, consequently the LTGA variants did not found high-quality solutions. We suppose the waiting time for reaching a sharp increase of fitness with the LTGA is proportional to the population size, therefore, θ should be increased in order to see it happen.

This section described the effectiveness of the algorithms. It was shown that linkage learning usefulness seems to decrease as K increases, with LTGA and rLTGA performing similarly for large K. The next section describes results concerning the efficiency

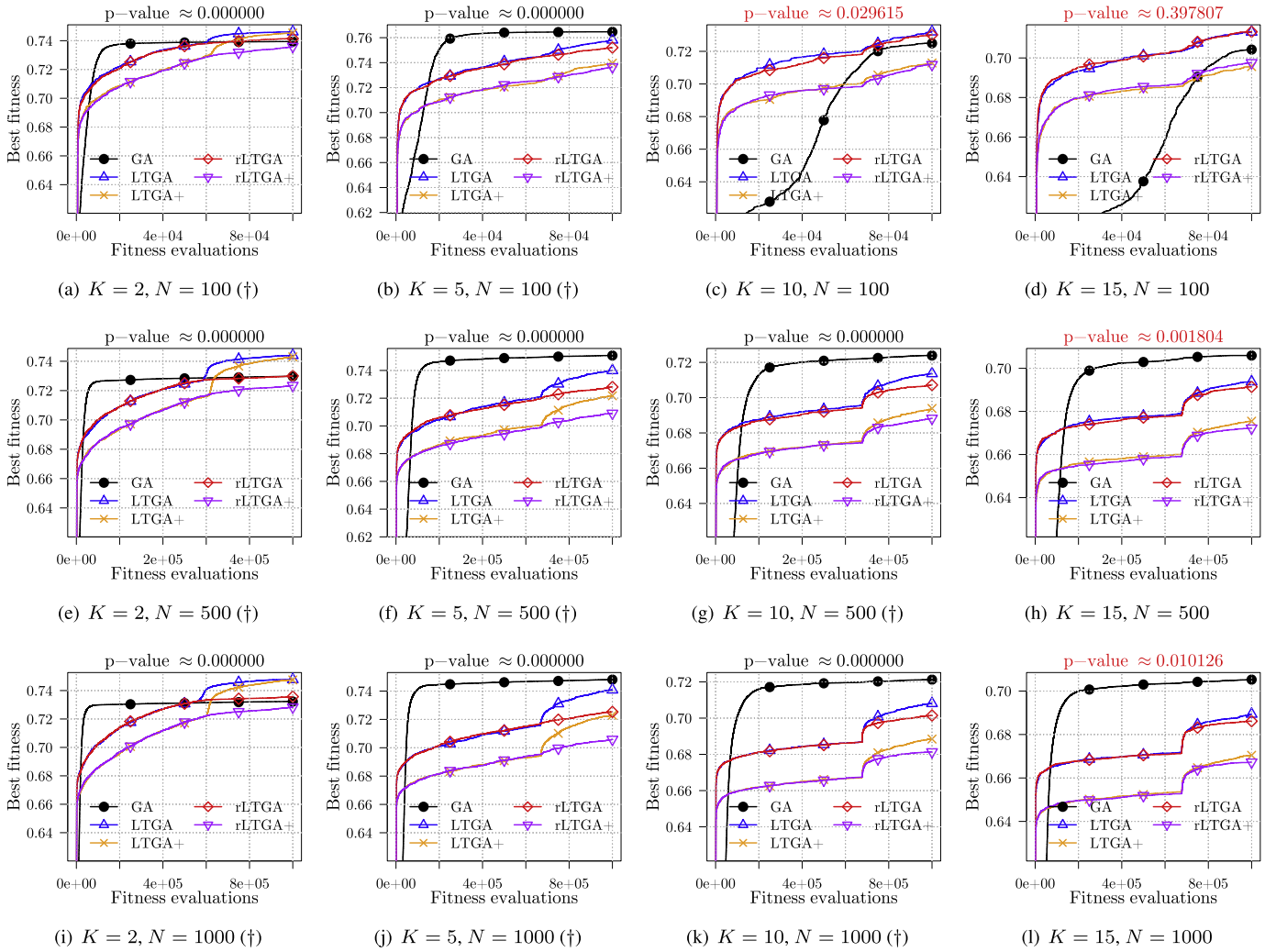


Fig. 6. Average trajectory of the best fitness of each algorithm along $\theta = N \times 10^3$ fitness evaluations. All experiments with $K = 2, 5, 10, 15$ and $N = 100, 500, 1000$ were performed with a fixed $\lambda = 500$.

(running-time) of the algorithms. The conjunction of both effectiveness and efficiency should provide reliable indicators to estimate the range of usefulness of linkage learning.

5.4. Efficiency results

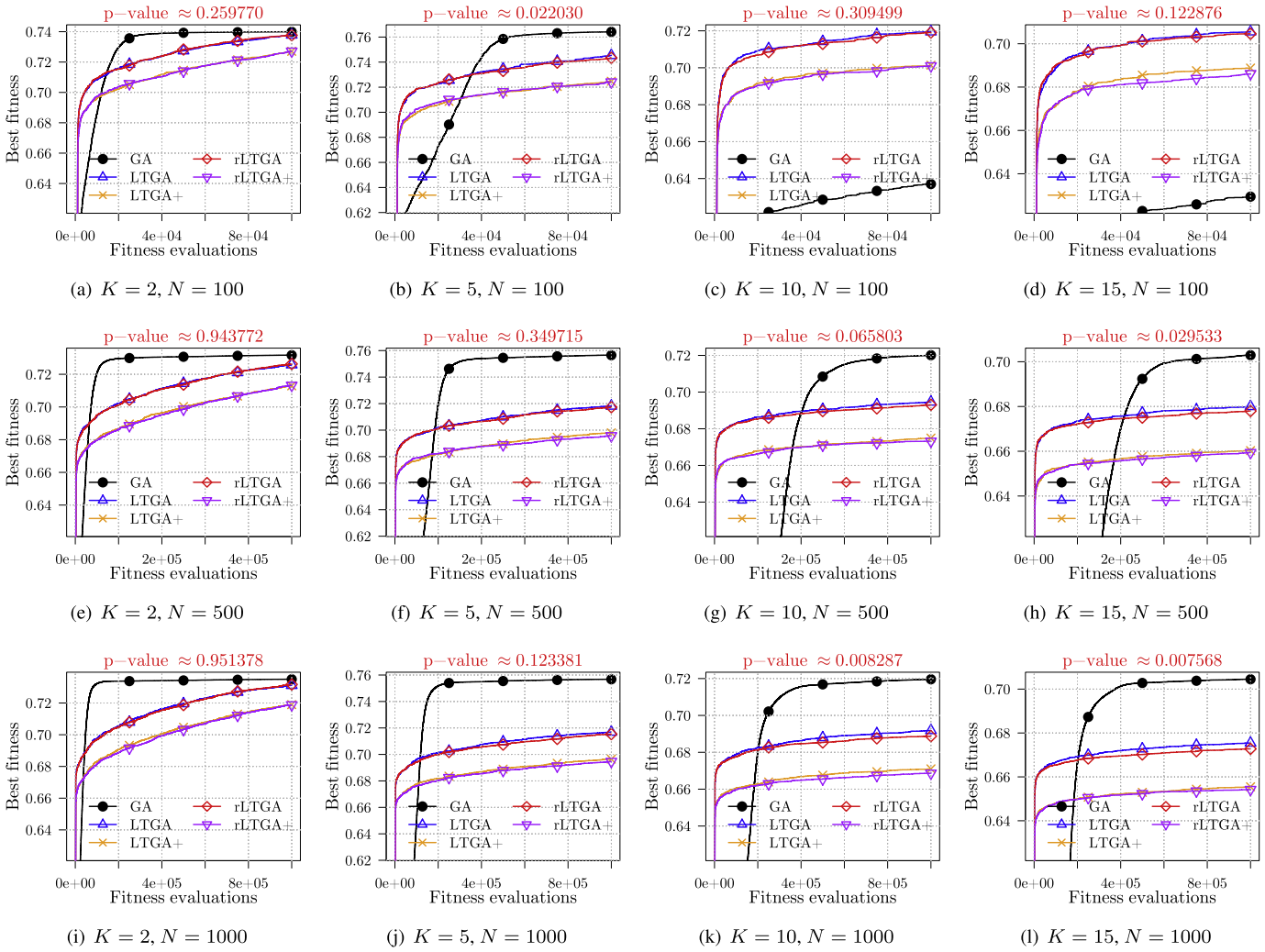
This section reports the results for the same instances of the previous section, but now regarding the efficiency of the algorithms in reaching their best solutions (first-hit time). A symbol ‡ has been added to the captions of experiments whose p-value $< \alpha$ the LTGA outperformed the rLTGA. According to previous results, where a † symbol represented the LTGA found better solutions, now a pair of symbols †‡ means the LTGA found better solutions faster than the rLTGA. Additionally, we have colored the boxes where p-value $\geq \alpha$ to facilitate the identification of experiments in which LTGA and rLTGA first-hitting times did not differ significantly. Since the LTGA was not effective with population size $\lambda = 1000$, the efficiency results for this case were suppressed.

Fig. 8 shows the distribution of the first-hitting time of each algorithm obtained with a population size $\lambda = 100$. For smaller instances (N=100) [Figs. 8(a)–(d)], the LTGA was slower than the rLTGA (except for K=2). Since building a linkage-tree is much more expensive than building a random linkage-tree, such a result was expected. It took the LTGA variants with mutation longer to find their final solutions, which was also expected, due to the additional diversity imposed by mutation. The GA was the most unstable among the algorithms and its first-hit time distribution showed a large standard deviation.

For medium-sized instances (N=500) [Figs. 8(e)–(h)], the results are better. The LTGA was the fastest with K=2,10, hence the most effective and efficient in these instances (note the †‡ in the caption). The increase of the problem size (N) led to a considerable reduction of variance in the LTGA and rLTGA results. GA and LTGA+ were very unstable with small and large standard deviations in different situations. Mutation exerted a much more subtle effect on the rLTGA than on the LTGA.

For the largest instances (N=1000) [Figs. 8(i)–(l)], the results were even better. The LTGA outperformed the rLTGA in all cases and found better solutions faster (note the †‡ in the captions). LTGA+ and GA were the most unstable and showed large variances in some cases.

In general, for a population size $\lambda = 100$, the LTGA was better in large instances, N=500,1000. In these cases, large linkage-trees are produced, which also lead to large sets $\mathcal{F}_{\text{LTGA}}$. As the number of subsets $\pi \in \mathcal{F}_{\text{LTGA}}$ grows, the number of fitness evaluations performed by the LTGA variants in each generation also increases. As a consequence, within the limit of θ fitness evaluations, the number of generations decreases and fewer model-building steps are performed, which lead to a decrease in the running-time. The comparison between LTGA and rLTGA suggests that, in terms of



(a) $K = 2, N = 100$ (b) $K = 5, N = 100$ (c) $K = 10, N = 100$ (d) $K = 15, N = 100$

(e) $K = 2, N = 500$ (f) $K = 5, N = 500$ (g) $K = 10, N = 500$ (h) $K = 15, N = 500$

(i) $K = 2, N = 1000$ (j) $K = 5, N = 1000$ (k) $K = 10, N = 1000$ (l) $K = 15, N = 1000$

Fig. 7. Average trajectory of the best fitness of each algorithm along $\theta = N \times 10^3$ fitness evaluations. All experiments with $K = 2, 5, 10, 15$ and $N = 100, 500, 1000$ were performed with a fixed $\lambda = 1000$.

solution quality and running-time, linkage learning is more useful in large instances [Figs. 5(i)–(l) and 8(i)–(l), respectively].

Fig. 9 shows the distribution of the first-hitting time with a population size $\lambda = 500$. As mentioned Section 5.3, the LTGA variants make progresses by large increases of the best fitness. However, with large populations such a pattern requires more time to occur and progresses in fitness slow down. With a population size $\lambda = 500$ this fact was evident and in most cases the LTGA did not outperform the rLTGA (p-values ≈ 1).

For instances with K=2,10,15 and problem sizes N=100,500,1000, the differences in the running-time between the LTGA and the rLTGA did not support the use of linkage learning. Moreover, such differences increased with the increase of multimodality and rLTGA became even faster than LTGA. Mutation slowed down the first-hitting time in all cases and the GA showed the highest variance in its first-hitting time.

For instances of moderate multimodality (K=5) the LTGA was the fastest among the algorithms in all problem sizes. Furthermore, it was also more effective than the rLTGA [Figs. 6(i)–(l)]. In such cases we can conclude the LTGA is better than rLTGA, i.e. both more effective and more efficient, which means linkage learning plays an important role in the solution of instances with $K \approx 5$. The next section summarizes the results concerning such aspects for the estimation of a range of usefulness for linkage learning in terms of K and N.

6. Discussions

The experiments aimed at verifying the existence of a range of usefulness for linkage learning, delimited by K, outside which pairwise independence is likely to occur. The results have shown the validity of our hypothesis and the performance of the LTGA deteriorated, in relation to the rLTGA, as K increased. Now, we have evidences to conclude linkage learning is more effective in instances with $K \approx 5$. Coincidentally, most of the previous studies that evaluated EDAs in NK-landscapes used values of K around this threshold [65,62,66,67,60,48,68]. This section summarizes and generalizes such evidences by analyzing results for all $1 \leq K \leq 15$. Since NK-landscapes with nearest-neighbor linkages have also been used to benchmark EDAs, they are also considered here.

The algorithms were run 100 times for each triple (N, K, λ). General information about linkage learning usefulness was collected by counting how many times the LTGA outperformed the rLTGA in terms of solution quality. Sign test statistics were obtained by subtracting from the total the number of cases in which the rLTGA found the best solutions. Therefore, for every pair (N, K) there is a value that summarizes Fig. 10(a) shows the values (normalized within the [0,1] interval) concerning random linkages Fig. 10(b) shows the values concerning nearest-neighbor linkages.

The values on the map can be interpreted as the probability of the LTGA outperforming the rLTGA in each configuration of (N,K)

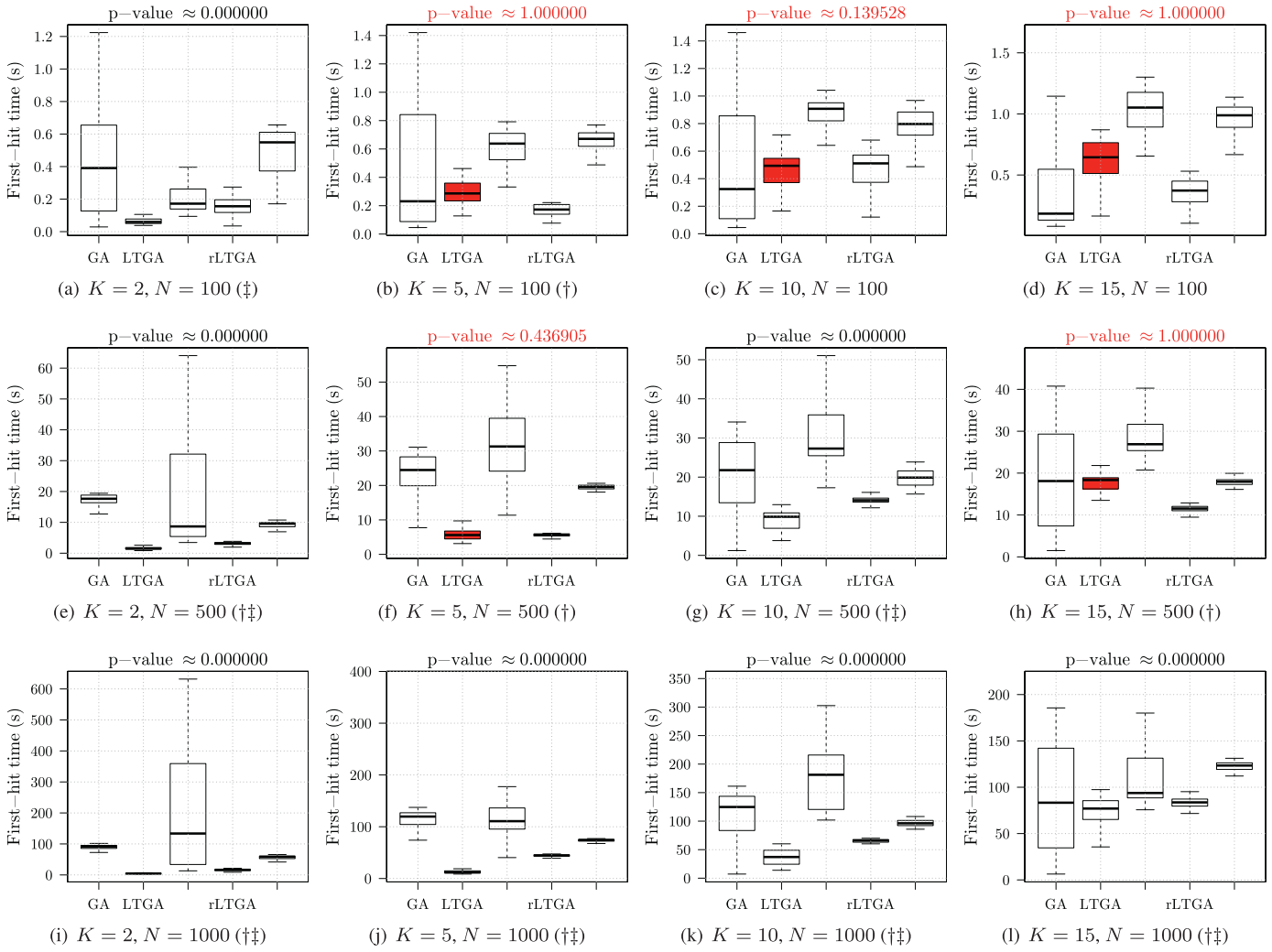


Fig. 8. Distribution of the first-hitting time for each algorithm, with $K = 2, 5, 10, 15$, $N = 100, 500, 1000$ and $\lambda = 100$.

and considering all population sizes λ=100,500,1000 (darker colors meaning lower probabilities). Regarding random linkages [Fig. 10(a)], the range in which linkage learning provided the best benefits encompasses the very restricted region defined by the $1 \leq K \leq 6$ interval. On the other hand, there is a large region in which linkage learning produced reasonable benefits. For example, for large instances (N=1000), linkage learning was reasonably useful in the large $1 \leq K \leq 13$ interval. However, for small instances (N=100), linkage learning was useful only in the small region delimited by the $1 \leq K \leq 5$ interval. Regarding nearest-neighbor linkages [Fig. 10(b)], regions were well delimited and, independently of the problem size (N), linkage learning was of great utility for $K \leq 9$.

Although these color maps elucidate the limits of linkage learning, they do not quantify how well LTGA outperformed the rLTGA. To obtain such information we use a different sign test. If the LTGA finds a much better solution (higher fitness) than the rLTGA, a large positive contribution is accumulated (difference between the fitness of solutions found by each algorithm). If, instead, the rLTGA finds a better solution, the difference will be negative and subtracted from the total. Through this procedure, we obtain information about how intensely linkage learning has contributed to the search.

Fig. 11 shows the color map resulting from the second test. Since it is more restrictive than the first, the regions where linkage learning was more effective have shrunk for both random and nearest-neighbor linkages. On the other hand, the agreement between such regions has increased considerably and, in both cases, the sweet-spot was delimited by the $2 \leq K \leq 6$ interval. Similarly to the first color map, the linkage learning usefulness for random linkages [Fig. 11(a)] was shown to depend on the problem size, whereas for nearest-neighbor linkages [Fig. 11(b)] the range of usefulness was mostly independent of the problem size N.

7. Conclusions

EDAs were proposed as an alternative for traditional evolutionary algorithms in which reproduction operators could rely on probabilistic models learned from the population to enable a more effective search. In the pursuit for effective EDAs, many different linkage learning procedures and probabilistic graphic models have been investigated. However, due to the algorithmic complexity associated with the learning of representative models, some concessions had to be made in order to obtain useful models in a reasonable time. The two main concessions relate to limiting the complexity of the model and/or using approximation algorithms to build the models. Such concessions made EDAs practical but at the same time introduced some limitations.

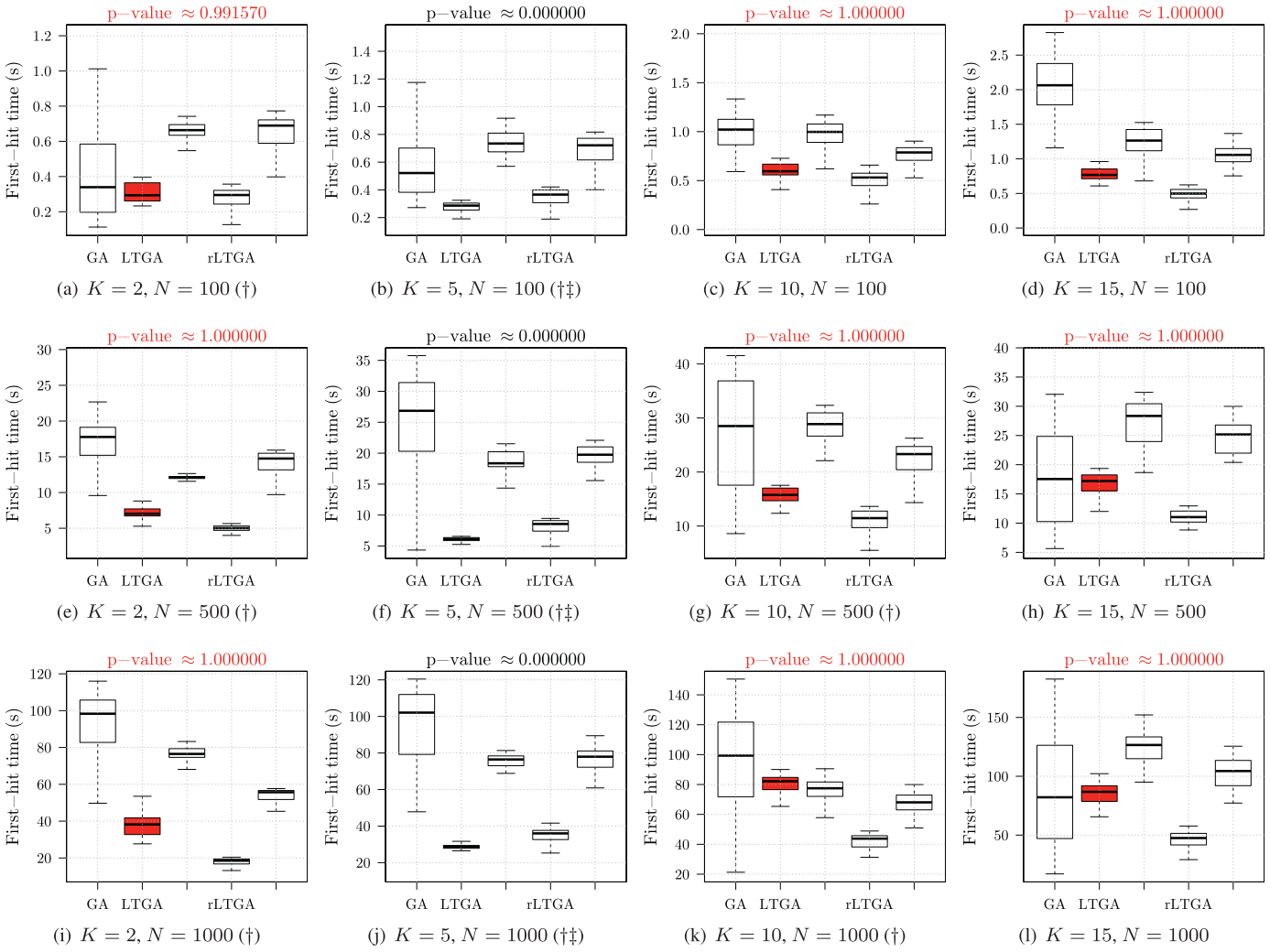


Fig. 9. Distribution of the first-hitting time for each algorithm, with $K = 2, 5, 10, 15$, $N = 100, 500, 1000$ and $\lambda = 500$.

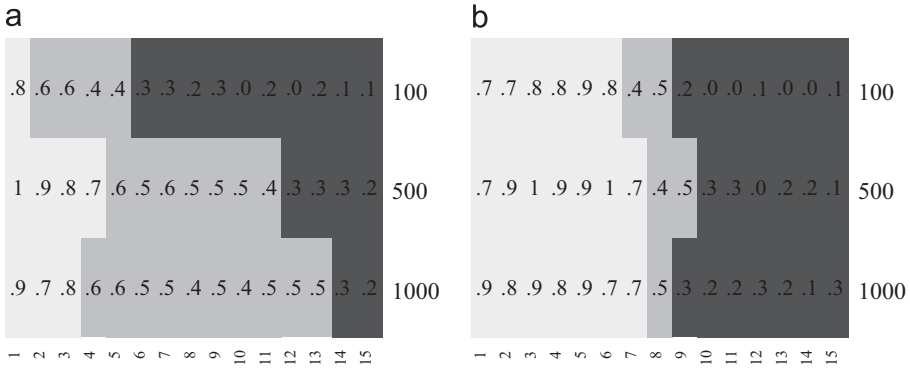


Fig. 10. For each pair (*K*,*N*) the values indicate the probability of the LTGA outperforming the rLTGA (according to the experiments). Linkage-learning was more useful in light-colored regions. (a) Random linkages. (b) Nearest-neighbor linkages.

Pairwise independence is one example of such limitations, which most contemporary EDAs cannot overcome. In the presence of pairwise independence, bivariate statistics do not provide information about high-order statistics, a situation not predicted by multivariate EDAs. Many studies have reported the bad performance of state-of-the-art EDAs in these contexts. This study investigated the causes of pairwise independence in order to understand how likely problems with such characteristic appear in practice. This main goal was tackled in three fronts.

In the first part, we reviewed the theoretical analysis proposed by the authors in Ref. [40], in which the conditions for accurate learning of additively separable functions (from bivariate statistics) were defined. From these conditions, we measured the influence of multimodality in the linkage learning difficulty of some well-known

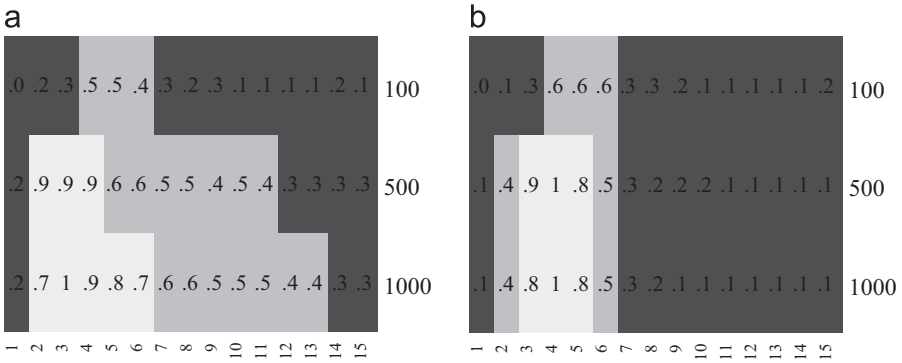


Fig. 11. For each pair (*K*,*N*), the values indicate how intensely the LTGA has outperformed the rLTGA (according to experiments). Light-colored regions indicate values of *K* in which significant benefits were obtained. (a) Random linkages. (b) Nearest-neighbor linkages.

additively separable functions: *mk*-trap, *mk*-bipolar, *mk*-parity and *mk*-parity/trap. The analysis demonstrated a causality between multimodality and linkage learning difficulty. In other words, as multimodality grows it becomes harder to learn accurate models and, in extreme cases, pairwise independence might emerge.

In the second part, we investigated the ability of some state-of-the-art multivariate EDAs to learn additively separable functions in the presence of pairwise independence. It was analytically proved that EDAs like the eCGA and LTGA are intrinsically unable to learn high-order dependences in the presence of pairwise independence. EDAs like the BOA and EBNA are also unable to learn the high-order dependences, but in this case due the limitations imposed by their approximate model-building, which can be circumvented by the use of more robust procedures.

In the third part, we investigated the impact of pairwise independence on non-separable functions. In such cases, multimodality and degree of interactions per variable are both possible causes of pairwise independence. Fortunately, both of these characteristics can be evaluated simultaneously by the parameter *K* on the *NK*-model. Therefore, by a comparison between the LTGA and rLTGA, regarding effectiveness and efficiency, in instances of different sizes ($N=100,500,1000$) and degrees of multimodality ($1 \leq K \leq 15$) we had a method to assess the limitations of linkage learning and the emergence of pairwise independence. Since the rLTGA uses random linkage-trees during the search, it is expected to find solutions of quality similar to the LTGA only in cases which linkage learning do not produce useful models. The results of such comparison confirmed our hypothesis and the LTGA and rLTGA performed very similarly on instances with large *K*. In general, most of the LTGA best results were found for $K < 10$, more specifically for $K \approx 5$.

Since pairwise independence was shown to impact similarly many multivariate EDAs, we expect other algorithms to have a range of usefulness similar to the one found for the Linkage-Tree Genetic Algorithm. From this perspective, we conjecture that performance differences among eCGA, BOA, EBNA, LTGA and other similar EDA, should occur within a common range of *K*, whose instances do not induce pairwise independence.

These conclusions shed some light on promising research directions for EDAs in binary search spaces. One of the alternatives is the development of EDAs capable of circumvent the limitations imposed by pairwise independence, which in turn depends on efficient multivariate learning procedures. A second alternative is the identification of the characteristics defining the problems in which current EDAs would excel, in order to exploit EDAs in such a context. In both cases, it is clear that linkage learning should not be seen as general tool applicable to any optimization problem. In fact, according to the results EDAs would be indicated to large scale optimization problems with moderate multimodality.

Acknowledgments

The authors are indebted to the FAPESP for the financial support provided for this research (2011/07792-4).

References

[1] H. Simon, The Sciences of the Artificial, The MIT Press, Cambridge, MA, USA, 1996.
[2] J.H. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press, Ann Arbor, 1975.
[3] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Professional, Boston, MA, USA, 1989.
[4] D.E. Goldberg, The Design of Innovation: Lessons from and for Competent Genetic Algorithms, Kluwer Academic Publishers, Norwell, MA, USA, 2002.
[5] C.R. Reeves, J.E. Rowe, Genetic Algorithms: Principles and Perspectives: A Guide to GA Theory, Kluwer Academic Publishers; Norwell, MA, USA, 2002.
[6] P. Larrañaga, J. Lozano, Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, vol. 2, Springer, Netherlands, 2002.
[7] J.A. Lozano, P. Larrañaga, I.n. Inza, E. Bengoetxea, Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms, Studies in Fuzziness and Soft Computing, Springer, Secaucus, NJ, USA, 2006.
[8] M. Pelikan, K. Sastry, E. Cantú-Paz, Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications, Springer Verlag, Secaucus, NJ, USA, 2006.
[9] J. Zhang, Z.-H. Zhan, Y. Lin, N. Chen, Y.-J. Gong, J.-h. Zhong, H.-H. Chung, Y. Li, Y.-h. Shi, Evolutionary computation meets machine learning: a survey, IEEE Comput. Intell. Mag. 6 (4) (2011) 68–75, http://dx.doi.org/10.1109/MCI.2011.942584.
[10] H. Mühlenbein, G. Paaß, From recombination of genes to the estimation of distributions I. Binary parameters, in: Parallel Problem Solving from Nature, PPSN IV, Lecture Notes in Computer Science, vol. 1141, Springer, 1996, pp. 178–187; London, UK. http://dx.doi.org/10.1007/3-540-61723-X_982.
[11] S. Baluja, Population-based incremental learning: a method for integrating genetic search based function optimization and competitive learning, Technical Report, Carnegie Mellon University, USA, 1994.
[12] G. Harik, F.G. Lobo, D. Goldberg, The compact genetic algorithm, IEEE Trans. Evol. Comput. 3 (4) (1999) 287–297, http://dx.doi.org/10.1109/4235.797971.
[13] J. De Bonet, C. Isbell, P. Viola, MIMIC: finding optima by estimating probability densities, Adv. Neural Inf. Process. Syst. (1997) 424–430.
[14] M. Pelikan, H. Müehlenbein, The bivariate marginal distribution algorithm, in: Advances in Soft Computing, Springer, London, 1999, pp. 521–535. http://dx.doi.org/10.1007/978-1-4471-0819-1_39.
[15] H. Mühlenbein, T. Mahnig, FDA—a scalable evolutionary algorithm for the optimization of additively decomposed functions, Evol. Comput. 7 (4) (1999) 353–376, http://dx.doi.org/10.1162/evco.1999.7.4.353.
[16] R. Etxeberria, P. Larranaga, Global optimization using Bayesian networks, in: Second Symposium on Artificial Intelligence (CIMAF-99), Habana, Cuba, 1999, pp. 332–339.
[17] M. Pelikan, D. Goldberg, E. Cantú-Paz, Linkage problem, distribution estimation, and Bayesian networks, Evol. Comput. 8 (3) (2000) 311–340, http://dx.doi.org/10.1162/106365600750078808.
[18] G.R. Harik, F.G. Lobo, K. Sastry, Linkage learning via probabilistic modeling in the Extended Compact Genetic Algorithm (ECGA), in: Scalable Optimization via Probabilistic Modeling, Studies in Computational Intelligence, vol. 33, Springer, Berlin Heidelberg, 2006, pp. 39–61. http://dx.doi.org/10.1007/978-3-540-34954-9_3.
[19] P. Larraãga, H. Karshenas, C. Bielza, R. Santana, A review on probabilistic graphical models in evolutionary computation, J. Heuristics 18 (5) (2012) 795–819, http://dx.doi.org/10.1007/s10732-012-9208-4.

[20] D. Chickering, Learning bayesian networks is np-complete, in: D. Fisher, H.-J. Lenz (Eds.), Learning from Data, Lecture Notes in Statistics, vol. 112, Springer, New York, 1996, pp. 121–130. http://dx.doi.org/10.1007/978-1-4612-2404-4_12.
[21] R. Santana, P. Larrañaga, J.A. Lozano, Research topics in discrete estimation of distribution algorithms based on factorizations, Memet. Comput. 1 (1) (2009) 35–54, http://dx.doi.org/10.1007/s12293-008-0002-7.
[22] D.J. Coffin, R.E. Smith, The limitations of distribution sampling for linkage learning, in: Proceedings of the Congress on Evolutionary Computation, CEC'07, IEEE, 2007, pp. 364–369; Singapore. http://dx.doi.org/10.1109/CEC.2007.4424494.
[23] D. Coffin, R.E. Smith, Linkage learning in estimation of distribution algorithms, in: Linkage in Evolutionary Computation, Studies in Computational Intelligence, vol. 157, Springer, Berlin Heidelberg, 2008, pp. 141–156. http://dx.doi.org/10.1007/978-3-540-85068-7_7.
[24] M. Pelikan, Hierarchical bayesian optimization algorithm, in: Hierarchical Bayesian Optimization Algorithm, Studies in Fuzziness and Soft Computing, vol. 170, Springer, Berlin Heidelberg, 2005, pp. 105–129. http://dx.doi.org/10.1007/978-3-540-32373-0_6.
[25] S.-C. Chen, T.-L. Yu, Difficulty of linkage learning in estimation of distribution algorithms, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'09, ACM, 2009, pp. 397–404. http://dx.doi.org/10.1145/1569901.1569957.
[26] G. Harik, D. Goldberg, Linkage learning through probabilistic expression, Comput. Methods Appl. Mech. Eng. 186 (2–4) (2000) 295–310, http://dx.doi.org/10.1016/S0045-7825(99)00388-6.
[27] C. Echegoyen, R. Santana, J. Lozano, P. Larraaga, The impact of Exact Probabilistic Learning Algorithms in EDAs based on bayesian networks, in: Linkage in Evolutionary Computation, Studies in Computational Intelligence, vol. 157, Springer, Berlin Heidelberg, 2008, pp. 109–139. http://dx.doi.org/10.1007/978-3-540-85068-7_6.
[28] D. Iclanzan, Higher-order linkage learning in the eCGA, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '12, ACM, 2012, pp. 265–272; Philadelphia, Pennsylvania, USA. http://dx.doi.org/10.1145/2330163.2330202.
[29] D. Iclanzan, Hierarchical allelic pairwise independent functions, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'11, ACM, 2011, pp. 633–640; Dublin, Ireland. http://dx.doi.org/10.1145/2001576.2001663.
[30] C. Echegoyen, Q. Zhang, A. Mendiburu, R. Santana, J. Lozano, On the limits of effectiveness in estimation of distribution algorithms, in: Proceedings of the Congress on Evolutionary Computation, CEC'11, IEEE, 2011, pp. 1573–1580; New Orleans, LA. http://dx.doi.org/10.1109/CEC.2011.5949803.
[31] C. Echegoyen, A. Mendiburu, R. Santana, J. Lozano, Toward understanding EDAs based on bayesian networks through a quantitative analysis, IEEE Trans. Evol. Comput. 16 (2) (2012) 173–189, http://dx.doi.org/10.1109/TEVC.2010.2102037.
[32] R.-T. Liaw, C.-K. Ting, Effect of model complexity for estimation of distribution algorithm in nk landscapes, in: Proceedings of the Symposium on Foundations of Computational Intelligence, FOCI'13, IEEE, 2013, pp. 76–83; Singapore. http://dx.doi.org/10.1109/FOCI.2013.6602458.
[33] J.P. Martins, A.C.B. Delbem, The influence of linkage-learning in the linkage-tree GA when solving multidimensional knapsack problems, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'13, ACM, 2013, pp. 821–828; Amsterdam, The Netherlands. http://dx.doi.org/10.1145/2463372.2463476.
[34] J.P. Martins, C. Bringel Neto, M.K. Crocomo, K. Vittori, A.C.B. Delbem, A Comparison of linkage-learning-based genetic algorithms in multidimensional knapsack problems, in: Proceedings of the Congress on Evolutionary Computation, CEC'13, IEEE, 2013, pp. 502–509; Cancun. http://dx.doi.org/10.1109/CEC.2013.6557610.
[35] J.P. Martins, C.M. Fonseca, A.C. Delbem, On the performance of linkage-tree genetic algorithms for the multidimensional knapsack problem, Neurocomputing 146 (2014) 17–29, http://dx.doi.org/10.1016/j.neucom.2014.04.069.
[36] K.L. Sadowski, P.A. Bosman, D. Thierens, On the Usefulness of Linkage Processing for Solving MAX-SAT, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'13, ACM, 2013, pp. 853–860; Amsterdam, The Netherlands. http://dx.doi.org/10.1145/2463372.2463474.
[37] C.-Y. Chuang, Y.-p. Chen, Sensibility of linkage information and effectiveness of estimated distributions, Evol. Comput. 18 (4) (2010) 547–579, http://dx.doi.org/10.1162/EVCO_a_00010.
[38] C. Echegoyen, A. Mendiburu, R. Santana, J.A. Lozano, On the taxonomy of optimization problems under estimation of distribution algorithms, Evol. Comput. 21 (3) (2013) 471–495, http://dx.doi.org/10.1162/EVCO_a_00095.
[39] C. Echegoyen, R. Santana, A. Mendiburu, J.A. Lozano`, Comprehensive characterization of the behaviors of estimation of distribution algorithms, Theor. Comput. Sci. 598 (2015) 64–86. http://dx.doi.org/10.1016/j.tcs.2015.04.015.
[40] J.P. Martins, A.C. Delbem, Multimodality and the linkage-learning difficulty of additively separable functions, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'14, ACM, 2014, pp. 365–372; Vancouver, BC, Canada. http://dx.doi.org/10.1145/2576768.2598281.
[41] J.P. Martins, Analysis of linkage learning in evolutionary optimization (Ph.D. thesis), University of Sao Paulo, Brazil, 2015.
[42] B. Yuan, M. Gallagher, On the importance of diversity maintenance in estimation of distribution algorithms, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'05, ACM, 2005, pp. 719–726; Washington DC, USA. http://dx.doi.org/10.1145/1068009.1068129.
[43] J. Peña, J. Lozano, P. Larrañaga, Globally multimodal problem optimization via an estimation of distribution algorithm based on unsupervised learning of bayesian networks, Evol. Comput. 13 (1) (2005) 43–66, http://dx.doi.org/10.1162/1063656053583432.
[44] K. Sastry, D. Goldberg, M. Pelikan, Limits of scalability of multiobjective estimation of distribution algorithms, in: Proceedings of the Congress on Evolutionary Computation, CEC'05, vol. 3, IEEE, 2005, pp. 2217–2224; Edinburgh, Scotland. http://dx.doi.org/10.1109/CEC.2005.1554970.
[45] M. Pelikan, D. Goldberg, F. Lobo, A survey of optimization by building and using probabilistic models, Comput. Optim. Appl. 21 (1) (2002) 5–20, http://dx.doi.org/10.1023/A:1013500812258.
[46] M. Hauschild, M. Pelikan, An introduction and survey of estimation of distribution algorithms, Swarm Evol. Comput. 1 (3) (2011) 111–128, http://dx.doi.org/10.1016/j.swevo.2011.08.003.
[47] M. Kelbert, Y.M. Suhov, Information Theory and Coding by Example, Cambridge University Press, New York, NY, USA, 2013.
[48] D. Thierens, P.A. Bosman, Optimal mixing evolutionary algorithms, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'11, ACM, 2011, pp. 617–624; Dublin, Ireland. http://dx.doi.org/10.1145/2001576.2001661.
[49] S.A. Kauffman, The Origins of Order: Self Organization and Selection in Evolution, Oxford University Press; Oxford, 1993.
[50] J. Jensen, Sur les fonctions convexes et les ingalits entre les valeurs moyennes, Acta Math. 30 (1) (1906) 175–193, http://dx.doi.org/10.1007/BF02418571.
[51] G. Harik, D. Goldberg, Learning Linkage, in: Foundations of Genetic Algorithms, FOGA'97, Morgan Kaufmann Publishers, 1997, pp. 247–262.; San Diego, CA, USA.
[52] M. Pelikan, K. Sastry, D. Goldberg, Scalability of the Bayesian optimization algorithm, Int. J. Approx. Reason. 31 (3) (2002) 221–258, http://dx.doi.org/10.1016/S0888-613X(02)00095-6.
[53] G.F. Cooper, E. Herskovits, A Bayesian method for the induction of probabilistic networks from data, Mach. Learn. 9 (1992) 309–347, http://dx.doi.org/10.1007/BF00994110.
[54] G. Schwarz, Estimating the dimension of a model, Ann. Stat. 6 (2) (1978) 461–464.
[55] W. Buntine, Theory refinement on bayesian networks, in: Proceedings of the Conference on Uncertainty in Artificial Intelligence, UAI'91, Morgan Kaufmann Publishers Inc., 1991, pp. 52–60; Los Angeles, California, USA.
[56] C. Echegoyen, J. Lozano, R. Santana, P. Larranaga, Exact Bayesian network learning in estimation of distribution algorithms, in: Proceedings of the Congress on Evolutionary Computation, CEC'07, IEEE, 2007, pp. 1051–1058; Singapore. http://dx.doi.org/10.1109/CEC.2007.4424586.
[57] D. Thierens, The linkage tree genetic algorithm, in: Parallel Problem Solving from Nature, PPSN XI, Lecture Notes in Computer Science, vol. 6238, Springer, Berlin Heidelberg, 2010, pp. 264–273. http://dx.doi.org/10.1007/978-3-642-15844-5_27.
[58] D. Thierens, P.A. Bosman, Hierarchical problem solving with the linkage tree genetic algorithm, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'13, ACM, 2013, pp. 877–884; Amsterdam, The Netherlands. http://dx.doi.org/10.1145/2463372.2463477.
[59] I. Gronau, S. Moran, Optimal implementations of UPGMA and other common clustering algorithms, Inf. Process. Lett. 104 (6) (2007) 205–210, http://dx.doi.org/10.1016/j.ipl.2007.07.002.
[60] M. Pelikan, M.W. Hauschild, D. Thierens, Pairwise and problem-specific distance metrics in the linkage tree genetic algorithm, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'11, ACM, 2011, pp. 1005–1012; Dublin, Ireland. http://dx.doi.org/10.1145/2001576.2001713.
[61] P. Bosman, D. Thierens, On measures to build linkage trees in LTGA, in: Parallel Problem Solving from Nature, PPSN XII, Lecture Notes in Computer Science, vol. 7491, Springer, Berlin Heidelberg, 2012, pp. 276–285. http://dx.doi.org/10.1007/978-3-642-32937-1_28.
[62] M. Pelikan, NK landscapes, problem difficulty, and hybrid evolutionary algorithms, in: Proceedings of the Conference on Genetic and Evolutionary Computation, GECCO'10, ACM, 2010, pp. 665–672; Portland, Oregon, USA. http://dx.doi.org/10.1145/1830483.1830606.
[63] P. Chu, J. Beasley, A genetic algorithm for the multidimensional knapsack problem, J. Heuristics 4 (1998) 63–86, http://dx.doi.org/10.1023/A:1009642405419.
[64] T.-L. Yu, K. Sastry, D.E. Goldberg, M. Pelikan, Population sizing for entropy-based model building in discrete estimation of distribution algorithms, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'07, ACM, 2007, pp. 601–608; London, England. http://dx.doi.org/10.1145/1276958.1277080.
[65] M. Pelikan, Analysis of estimation of distribution algorithms and genetic algorithms on NK landscapes, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'08, ACM, 2008, pp. 1033–1040; Atlanta, GA, USA. http://dx.doi.org/10.1145/1389095.1389287.
[66] M.W. Hauschild, M. Pelikan, Network crossover performance on NK landscapes and deceptive problems, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'10, ACM, 2010, pp. 713–720. http://dx.doi.org/10.1145/1830483.1830612.
[67] M. Pelikan, Analysis of epistasis correlation on NK Landscapes with nearest-neighbor interactions, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'11, ACM, 2011, pp. 1013–1020; Dublin, Ireland. http://dx.doi.org/10.1145/2001576.2001714.
[68] D. Thierens, P. Bosman, Evolvability analysis of the linkage tree genetic algorithm, in: Parallel Problem Solving from Nature, PPSN XII, Lecture Notes in Computer Science, vol. 7491, Springer, Berlin Heidelberg, 2012, pp. 286–295. http://dx.doi.org/10.1007/978-3-642-32937-1_29.