TU Delft

Delft University of Technology

Fitness-based Linkage Learning in the Real-Valued Gene-pool Optimal Mixing Evolutionary Algorithm

Olieman, Chantal; Bouter, Anton; Bosman, Peter A.N.

DOI
10.1109/TEVC.2020.3039698

Publication date
2020

Document Version
Accepted author manuscript

Published in
IEEE Transactions on Evolutionary Computation

Citation (APA)
Olieman, C., Bouter, A., & Bosman, P. A. N. (2020). Fitness-based Linkage Learning in the Real-Valued Gene-pool Optimal Mixing Evolutionary Algorithm. *IEEE Transactions on Evolutionary Computation*, *25*(2), 358-370. Article 9265419. https://doi.org/10.1109/TEVC.2020.3039698

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

Fitness-based Linkage Learning in the Real-Valued Gene-pool Optimal Mixing Evolutionary Algorithm

Chantal Olieman
Delft University of Technology
Delft, The Netherlands
chantalolieman@outlook.com

Anton Bouter
Centrum Wiskunde & Informatica
Amsterdam, The Netherlands
anton.bouter@cwi.nl

Peter A. N. Bosman
Centrum Wiskunde & Informatica
Amsterdam, The Netherlands
Delft University of Technology
Delft, The Netherlands
peter.bosman@cwi.nl

***Abstract*—The recently introduced Real-Valued Gene-pool Optimal Mixing Evolutionary Algorithm (RV-GOMEA) has been shown to be among the state-of-the-art for solving grey-box optimization problems where partial evaluations can be leveraged. A core strength is its ability to effectively exploit the linkage structure of a problem, which often is unknown a priori and has to be learned online. Previously published work on RV-GOMEA however demonstrated excellent scalability when the linkage structure is pre-specified appropriately. A mutual-information-based metric to learn linkage structure online, as commonly adopted in EDA's and the original discrete version of GOMEA, did not lead to similarly excellent results, especially in a black-box setting. In this article, the strengths of RV-GOMEA are combined with a new fitness-based linkage learning approach that is inspired by differential grouping that reduces its computational overhead by an order of magnitude for problems with fewer interactions. The resulting new version of RV-GOMEA achieves scalability similar to when a predefined linkage model is used, outperforming also, for the first time, the EDA AMaLGaM upon which it is partially based in a black-box setting where partial evaluations can not be leveraged.** [1]

***Index Terms*—Genetic Algorithm, Linkage Learning, Fitness**

I. Introduction

A key strength of many state-of-the-art model-based evolutionary algorithms (EA's) lies in the effective exploitation of a problem's linkage structure [5], [11], [29], [33]. When the linkage structure of a problem is known, this information can be used to solve the optimization problem more effectively. If a problem is fully decomposable into sub-problems, these lower-dimensional sub-problems can be solved independently to achieve better efficiency [17]. Conversely, if a problem is (partially) inseparable and its variables are strongly dependent, trying to solve the problem with a model that wrongly assumes decomposability is very inefficient [31]. This is known to hold for problems with discrete (binary) variables, e.g., the deceptive trap function [37], as well as real-valued variables, e.g., the rotated ellipsoid function [21].

A well-known approach that effectively exploits the linkage structure of a problem in the discrete domain is Gene-pool Optimal Mixing (GOM) [39]. In GOM, variables modeled in the same linkage set will be affected by recombination together, ensuring that no valuable information captured in the specific combination of variables is lost. The recombination operator applies recombination to partial solutions by iterating over all linkage sets in a linkage model. For every linkage set, recombination is executed only on the variables represented in the current linkage set, thereby exchanging partial solutions between individuals. If this recombination leads to improved fitness of a solution, the changes to that solution are accepted. The Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) [3] randomly chooses a donor solution for different linkage sets, thereby using the entire gene-pool in search for optimal sub-solutions.

In many real-world applications, the optimization problem is not a black-box. It may be treated as such if exploiting specific properties is difficult, but generally some additional knowledge is available. This may include the possibility to use partial evaluations, which are used to evaluate the impact of variation on a solutions' objective value in a fraction of $\frac{k}{\ell}$ time of a full fitness evaluation if only k variables are changed. Most literature, especially in case of real-valued variables is however focused on black-box optimization [13], [14].

Because OM works specifically by changing subsets of variables in existing solutions for which the fitness of the solution is known, partial evaluations can be leveraged excellently, also in case of real-valued variables. This makes GOMEA a well-suited method for solving real-world grey-box problems. Indeed, recent results show that superior results can be obtained over taking a black-box approach for real-world problems, including brachytherapy treatment planning [18], [20] and deformable image registration [6], [41]. These results were obtained with the recently introduced Real-Valued GOMEA (RV-GOMEA) [5] that leverages the strengths of GOMEA for the real-valued domain.

The linkage learning method employed by GOMEA has shown excellent performance and scalability in the discrete domain, but some issues have been encountered when applying the same approach to the continuous search spaces of the real-valued domain. A fundamental drawback of the currently used mutual information-based approach lies in its inability to correctly recognize fully decomposable sub-components. At the root of this lies the fact that this method has problems in identifying independent variables because selection causes the solutions to align with the fitness contours in the search

[1] This article is extended from the MSc thesis of Chantal Olieman, available at https://repository.tudelft.nl/ [24]

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

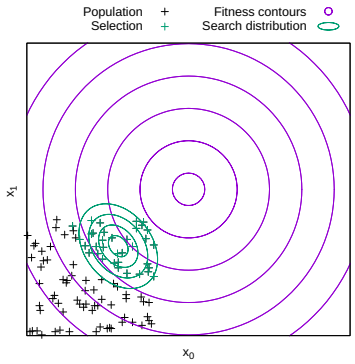


Fig. 1: Non-decomposable Gaussian search distribution (green iso-lines) learned (with maximum-likelyhood estimation) from a population of selected solutions on a 2D version of the decomposable sphere problem (purple iso-lines).

space as illustrated in Figure 1. Additional mechanisms such as the anticipated mean shift in AMaLGaM [2] and the evolution path in CMA-ES [9] cause the solution and Gaussian model of variation to align with the joint direction of improvement (i.e., the gradient in smooth problems), similar to how momentum is used in gradient descent algorithms when training neural networks [36]. In either case, the mutual information of the Gaussian model will indicate that dependencies exist, even if this is not the case, e.g., on the sphere function, especially when the population is initialized far away and not bracketing the optimum. Moreover, the method is based on the spread of the population and often many generations are needed for the linkage structure to be properly exposed by a population. Other methods used to identify linkage such as delta grouping and the random grouping scheme [26], [42] are also unable to correctly detect independent variables.

An alternative approach to identifying the linkage structure of a problem is based on measuring the changes in fitness values by perturbing certain variables. This method was first introduced in combination with the greedy linkage learning approach known as Differential Grouping [25], generally used with cooperative co-evolution [19], [30]. Whilst this method is able to correctly identify independent variables, it does not allow for overlapping linkage sets nor does it define a comparable measure on the dependence of variables. Lastly, when the problem consists mainly of decomposable sub-components, learning the linkage model is unnecessarily computationally expensive, as all of the $\ell(\ell-1)$ possible pairs have to be checked, even for completely decomposable problems.

In this paper, we try to overcome the earlier stated drawbacks of existing linkage learning methods by using the fitness-based dependency strengths to build an adapted linkage model based on the Linkage Tree as used in RV-GOMEA. Two different linkage model building methods are proposed. Both methods separate decomposable sub-problems as much as possible without separating non-decomposable variables that are strongly dependent. The resulting linkage models are integrated into RV-GOMEA, which has been proven to perform excellently on real-valued benchmark problems when correct linkage models are provided [5]. The introduced methods will be compared with existing linkage learning methods in combination with RV-GOMEA for a variety of benchmark problems. The hypothesis is that the proposed method is able to scale almost identically to offline learned linkage models but without the need of problem-specific knowledge. In the black-box domain, we expect a performance similar to that of AMaLGaM, something that has not been achieved before.

As for various real-world applications of (RV-)GOMEA like brachytherapy treatment planning [18], [20], deformable image registration [6], [41] and more [15], the optimal linkage model is not known and strong dependencies are imposed through geometry, for example, deformation vector field nodes or potential windmill locations that are near each-other are strongly dependent, but those far apart are weakly dependent. The provable added value of RV-GOMEA for these real-world problems could be increased even further if correct linkage models could be learned efficiently online.

The remainder of this article is structured as follows. In Section II we elaborate on the existing RV-GOMEA. Existing methods to model the dependencies of an optimization problem are discussed in Section III. In Section IV our newly proposed incremental approach for learning the linkage structure of a problem is introduced. The benchmark problems used to validate the performance of our method are introduced in Section V, and Section VI shows scalability results on these problems. The implications of our work and further challenges ahead are discussed in Section VII. Lastly, we summarize and draw conclusions about our findings in Section VIII.

II. RV-GOMEA

One of the key elements of GOMEA is its variation operator: the GOM method. This method uses a so-called Family of Subsets (FOS) to exploit the linkage structure of a problem. The current version of RV-GOMEA [5] is a combination of the existing GOMEA [3], which performs excellently in the discrete domain, extended with a continuous sampling model as employed in the state-of-the-art for numerical optimization, the Adapted Maximum-Likelihood Gaussian Model Iterated Density-Estimation Evolutionary Algorithm (AMaLGaM) [1]. This section will explain how linkage is modeled in RV-GOMEA by using a Family of Subsets (Sub-section II-A) and how a new population is generated by using GOM (Sub-section II-B). Then the application of RV-GOMEA in the grey-box domain is justified in Sub-section II-C.

A. Family of subsets

The linkage structure of a problem is modeled in GOMEA using a FOS, denoted as $\mathcal{F}$. The set $S = \{0, 1, ..., \ell-1\}$ contains all problem variables and every set $\mathcal{F}_i \in \mathcal{F}$ is a subset of S. The complete FOS $\mathcal{F}$ is a subset of the power-set $P(S)$ of S. Lastly, the FOS $\mathcal{F}$ is complete, meaning that every problem variable is represented in at least one subset of $\mathcal{F}$. There are various methods for constructing a FOS. Different

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

FOS structures have proven to work best on different problems [39]. Here we partially focus on two types:

1) Marginal product FOS: A marginal product FOS is defined as a set $\mathcal{F}$ where for every $\mathcal{F}_i$, $\mathcal{F}_j \in \mathcal{F}$ it holds that $\mathcal{F}_i \cap \mathcal{F}_j = \emptyset$. The univariate FOS is a special case of a marginal product FOS with $|\mathcal{F}| = \ell$ and thus $|\mathcal{F}_i| = 1$ for all $\mathcal{F}_i \in \mathcal{F}$.

2) Linkage tree FOS: The linkage tree FOS is most commonly used and shown to be the most universal in discrete optimization [38], [39]. The defining property of a linkage tree FOS is that every set $\mathcal{F}_i \in \mathcal{F}$ that contains more than one set is the union of two other sets in $\mathcal{F}$. Conceptually the linkage tree is built by iteratively merging the two FOS elements with the highest dependence $\mathcal{F}_i$ and $\mathcal{F}_j$ to form a new FOS element $\mathcal{F}_k$ that is added to the FOS, thus $\mathcal{F}_i \cup \mathcal{F}_j = \mathcal{F}_k$. When only pairwise-dependencies are used an implementation exists that allows for an LT FOS to be built in $O(n\ell^2)$ time [12]. The process of iteratively merging linkage sets is repeated until no more merges are possible, i.e., the full set of variables is added or the maximum linkage set size $|\mathcal{F}_k| = 100$ is reached.

B. Gene-Pool Optimal Mixing

The core principle of GOMEA, Gene-pool Optimal Mixing (GOM), mixes the population following the FOS subsets represented in the linkage model. Variables that are represented in the same FOS subset will be recombined together. In the real-valued domain, recombination alone does not suffice as one needs to sample values not currently present in the population and thus a continuous model is needed. In the recently introduced RV-GOMEA [5] solutions in the space represented by a FOS subset are sampled from a multivariate Gaussian that is estimated with maximum-likelihood based on the selection, as is done in AMaLGaM [1]. During one generation of RV-GOMEA, for every $\mathcal{F}_i \in \mathcal{F}$ of size $|\mathcal{F}_i| = k$ a $k-$dimensional multivariate Gaussian is estimated from the $n \cdot \tau$ best individuals in the population $\boldsymbol{P}$ (τ being the fraction of solutions selected from the population, in this case 0.35 as used in original AMaLGaM [1]). To create new offspring, all linkage sets $\mathcal{F}_i$ are considered in random order. For every individual in the population, $|\mathcal{F}_i|$ new values are sampled from the multivariate Gaussian and inserted into the existing individual. If the algorithm is run in a grey-box setting, partial evaluations can be leveraged to evaluate this new solution. In a black-box setting, a complete evaluation is required to determine whether the fitness of the solution has improved. If the change resulted in an increase in fitness, the changes are accepted. If not, the change is accepted with probability $p^{accept} = 0.05$. The next linkage set is then considered. If an individual does not improve for a certain number of generations, a method called forced improvement is applied to alter the individual following a convex linear combination of the parent solution and the elitist solution of the population, i.e. by moving it closer to the elitist solution.

To obtain good performance in an EA it is often important to correctly set the population size parameter. However, the best-suited population size is problem dependent and can thus not be set without any problem specific knowledge. To avoid needing to tune the population size parameter, RV-GOMEA uses an interleaved multi-start scheme (IMS) that runs multiple independent EA instances with growing population sizes. As smaller population sizes converge quicker but get stuck in local optima, an instance will be terminated once it is outperformed by another instance that has a larger population size.

C. Grey-box domain

Most of the research done on EAs is aimed at black-box optimization problems where no knowledge about the problem or its underlying structure is known. Other existing research that does operate in the grey-box domain such as [8], [10], [40] is fundamentally restricted to discrete optimization. In this article, we consider a domain of grey-box optimization problems where partial problem evaluations can be performed as this is directly applicable to real-valued optimization [5]. Whilst partial evaluations can be applied, that does not impose that the optimal problem structure is known and thus in both domains, effective linkage learning plays an important part in optimization. In a grey-box setting, partial evaluations allow for the recalculation of fitness when there are only few, e.g. k, modified variables in $O(g(k))$ time, rather than incurring the $O(g(\ell))$ overhead of full evaluations since these are only needed when all variables are changed. With $g()$ typically being a polynomial function e.g. $g(\ell) = \ell$ or $g(\ell) = \ell^2$. The cost of one partial function evaluation is therefore counted as $\frac{k}{\ell}$ with k as the number of changed variables. Since the optimal mixing phase of RV-GOMEA makes almost exclusively partial modifications to existing solutions, RV-GOMEA can very effectively leverage partial evaluations which makes it an effective algorithm for grey-box problems.

D. RV-GOMEA

The earlier introduced RV-GOMEA [5] combines the previously explained components into a real-valued evolutionary algorithm that exploits the linkage structure of a problem by iteratively applying GOM to different (sub)sets of problem variables in every generation.

III. Related Work

Multiple methods have been proposed to identify the structure of an unknown optimization function. In this section two of these methods will be discussed. Firstly, a population-based method that uses the spread of a population in the search-space to model dependencies is discussed. Secondly, a fitness-based method where the fitness values are directly used to measure linkage between variables is discussed. Both of these methods focus on pairwise dependencies. Once these dependencies are known, dependencies between subsets of variables are extrapolated from them.

A. Distribution-based methods

One of the methods used to extract dependencies based on the distribution of a population is the Mutual Information (MI) method [16]. The mutual information MI_{ij} of variables x_i

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

and x_j defines how much information about x_i we can derive by knowing x_j and vice versa. The MI is computed based on the probability distribution associated with the variables. In the case of real-valued variables, a parametric distribution is often used. In our case, as the AMaLGaM model is essentially a normal distribution, to calculate the MI between two variables x_i and x_j, the Pearson product-moment correlation coefficient r_{ij} can be used with $r_{ij} \in [-1, 1]$. A high absolute value of r corresponds to a high linear correlation between x_i and x_j. The mutual information between x_i and x_j is defined as follows:

$$\mathrm{MI}_{ij} = \log\left(\sqrt{\frac{1}{1-(r_{ij})^2}}\right) \tag{1}$$

$$\text{where } r_{ij} = \hat{\Sigma}_{ij} / \left(\hat{\sigma}_i \hat{\sigma}_j\right) \in [-1, 1] \tag{2}$$

Where $\hat{\Sigma}_{i,j}$ and $\hat{\sigma}_i$ are obtained from a covariance matrix that is estimated with maximum likelihood based on the selection. $r_{i,j}$ is only computed for pairs of variables. A FOS can be built from these pairwise dependencies as described in Section II-A2.

B. Fitness-based methods

The second method we consider to define whether two variables interact is directly based on fitness values and classifies a pair of variables as either separable or non-separable specifically by comparing the difference in fitness whilst making the exact same perturbation for x_i for different values of x_j [22], [23]. A more recent application of this method is leveraged in Differential Grouping [25]. Four points in the solution space are picked by combining all possible points that can be created by picking two different values for each x_i and x_j. The differences in fitness values for those points are used to calculate the interaction between x_i and x_j by determining if the change in fitness caused by a modification to x_i is affected by a modification to x_j. Variables x_i and x_j are said to interact when $|\Delta_i - \Delta_{i,j}| \geq \epsilon$ for some user-defined small ϵ, where Δ_i and $\Delta_{i,j}$ are defined as follows:

$$\Delta_i = (f(\boldsymbol{x})|x_i = a_i, x_j = a_j) - \tag{3}$$

$$(f(\boldsymbol{x})|x_i = a_i + b_i, x_j = a_j) \tag{4}$$

$$\Delta_{i,j} = (f(\boldsymbol{x})|x_i = a_i, x_j = a_j + b_j) - \tag{5}$$

$$(f(\boldsymbol{x})|x_i = a_i + b_i, x_j = a_j + b_j) \tag{6}$$

where a_i and b_i can be any real value as long as x_i and x_j remain within the function bounds. In this method a_i and b_i are selected randomly such that for every x_i, a_i and $a_i + b_i$ fall within the bound for x_i inside the current population. In our experiments, ϵ is set to 0 where we rounded to the smallest possible machine precision to ignore calculation errors.

IV. Scaled fitness-based linkage learning

To learn a linkage tree FOS, it is necessary to define a notion of linkage, or dependency, strength between pairs of variables. We denote $d_{i,j}$ as the pairwise dependency strength between x_i and x_j where:

$$d_{i,j} = \begin{cases} 1 - \Delta_{i,j}/\Delta_i & \text{if } \Delta_i \geq \Delta_{i,j} \\ 1 - \Delta_i/\Delta_{i,j} & \text{otherwise} \end{cases} \tag{7}$$

From $d_{i,j}$ a matrix D of size $\ell \times \ell$ can be constructed, storing all pairwise dependency strengths of an ℓ-dimensional problem. By definition of equation 7 the values of this matrix will lie within $[0, 1)$ with 0 for independent variables and $d_{i,j} > 0$ indicating some interaction between x_i and x_j. It is worth noting that even though $d_{i,j}$ does not represent an absolute dependency strength between variables, it can be used to compare the relative pairwise dependency strength by comparing $d_{i,j}$ and $d_{i,k}$. This property makes it possible to learn a linkage tree FOS based on the information stored in D as described in [5].

A. Analysis of overhead

Filling matrix D requires $\frac{\ell(\ell-1)}{2}$ dependency checks. For each of these dependency checks, four evaluations are needed, which results in a total number of $2\ell(\ell-1)$ evaluations. It is shown in [28] that it is possible to decrease the number of evaluations to $1 + \ell + \frac{\ell(\ell-1)}{2}$ by using the same a_i and b_i for every check and storing $f(x)|_{x_i = a_i + b_i}$ for every $x_i \in x$. When partial evaluations can be leveraged, the overhead can be decreased even further. Since only one variable is changed for the evaluations done to compute $d_{i,j}$, the number of changed variables $k = 1$ and therefore it is possible to decrease the total number of evaluations even further to $1 + \frac{1}{\ell}(\ell + \frac{\ell(\ell-1)}{2})$ = $2 + \frac{\ell-1}{2}$.

1) Picking a_i and b_i: As described in Section III-B, a_i and b_i can be picked randomly. However, for this method to work, a_i and b_i should be set and should remain unchanged. The values for a_i and b_i are estimated based on the current population:

$$a_i = min(x_i) + ((max(x_i) - min(x_i)) \cdot 0.35) \tag{8}$$

$$b_i = (max(x_i) - min(x_i)) \cdot 0.35 \tag{9}$$

The value of 0.35 has been empirically found to work well for setting a_i and b_i but every value > 0 and ≤ 0.5 is acceptable as the resulting values for x_i will be within the current population.

Employing the values of a_i and b_i for every pair of variables will give us $1 + \ell + \frac{\ell(\ell-1)}{2}$ new solutions with their associated fitness values. However, these solutions are heavily centered in one area of the solution space with ℓ solutions containing a_i in all but one of the problem dimensions and $\frac{\ell(\ell-1)}{2}$ solutions with a_i in all but two of the problem dimensions. For this reason these solutions will only be added to the population if their objective value is lower than any other solution in the population, i.e, if the solution is an elitist solution.

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

B. Incremental dependency updating

It is plausible to assume that real-world high-dimensional problems tend not to be fully dependent with strong dependencies (e.g., a 1000-dimensional rotated ellipsoid), but to be non-decomposable only to some extent. We therefore aim to slightly bias our method toward the assumption of less-than-fully dependent problems by decreasing the number of evaluations needed on sparsely dependent problems. To this end, to spread the computational load of filling the dependency matrix, every d_{ij} is initially set to the default value of 0 (independent). In each generation, ℓ random pairs are evaluated and a new FOS is learned. Since the right linkage model is not dependent on the population size, the same linkage structure is used for every population that is maintained with respect to the IMS explained in Section II-B. We call this process incremental dependency updating. The pseudo-code for this algorithm can be found in Algorithm 1. The process of checking all $\frac{\ell(\ell-1)}{2}$ pairs will be called a dependency cycle. When all pairs have been checked, no checks will happen for 2^k generations where k is the number of cycles that have taken place already. To better allocate the computational budget during optimization of a problem that is suspected to be independent, the dependency cycle is stopped prematurely and started again after 2^k generations with new random pairs. Specifically, the dependency cycle is stopped whenever no dependencies are found in one iteration (ℓ checks) and if the average number of found dependencies over all dependence cycles so far is smaller than some minimum value. We used $\frac{2}{\ell}$ as this was empirically found to work well on a variety of problems. Since the matrix is initialized with 0's the computational resources are geared more toward univariate optimization which results in a slight bias to decomposable problems. Yet, by restarting the dependency cycle every 2^k generations, changes in function landscapes can be captured and the dependencies can be updated accordingly by estimating a_i and b_i again as described in section IV-A1. This can result in a different $d_{i,j}$ for the same pair of variables x_i and x_j. As a result of the initialization of the dependency matrix, all variables are assumed to be independent and will only be considered to be dependent once an actual dependency is detected. Stopping the dependency detection when no dependencies are found is therefore not expected to significantly change the outcome of the dependency checks.

C. Pruning

Based on the information obtained from the scaled fitness-based dependency detection, a pruning method for the Linkage Tree FOS described in Section II-A2 can be used. The goal of pruning is to eliminate unnecessary linkage sets. A smaller FOS reduces the number of function evaluations and time spent on GOM per generation. If only the linkage sets that best capture the dependency between variables are correctly maintained, the efficiency of GOMEA may very well improve.

Consider the moment during the learning of the FOS, that two linkage sets $\mathcal{F}_i$ and $\mathcal{F}_j$ are to be merged to create $\mathcal{F}_k = \mathcal{F}_i \cup \mathcal{F}_j$. If all variables in $\mathcal{F}_k$ are pairwise dependent, $\mathcal{F}_i$ and $\mathcal{F}_j$ are removed from the FOS. Since all variables in $\mathcal{F}_k$ are dependent, mixing these variables together (which in RV-GOMEA entails sampling from a joint Gaussian distribution) will likely yield better results than separately mixing the variables from $\mathcal{F}_i$ or $\mathcal{F}_j$. Similarly, if there is no pairwise dependence between any variable in $\mathcal{F}_i$ and any other variable in $\mathcal{F}_j$, $\mathcal{F}_k$ is not added to the newly learned FOS.

In case that there are some pairwise dependencies between subsets, but not every variable in $\mathcal{F}_i$ is dependent on every variable in $\mathcal{F}_j$, the problem consists of non-decomposable overlapping sub-components. For this case we present two different pruning approaches, resulting in different FOS structures:

1) (Partial) linkage Tree: In this case, the two subsets $\mathcal{F}_i$ and $\mathcal{F}_j$ are merged together into $\mathcal{F}_k$ and all linkage FOS sets are kept in the FOS. This approach will result in a (partial) linkage tree where the biggest linkage sets are the size of the biggest non-decomposable sub-components.

2) Marginal product: The second approach ignores the subset of dependencies between $\mathcal{F}_i$ and $\mathcal{F}_j$ and keeps only the fully dependent linkage sets $\mathcal{F}_i$ and $\mathcal{F}_j$ in the FOS without merging any more sets. Combined with the other pruning steps this will always create a marginal product FOS, containing every variable exactly once.

D. FOS-based population size

With the problem-specific knowledge obtained by our linkage learning method, we can project a minimally required population size needed for RV-GOMEA to work well. If γ is the size of the biggest linkage set in $\mathcal{F}$ then following [2] the minimal population size n_{base} needed can be calculated as $n_{base} = 17 + 3\gamma\sqrt{\gamma}$. We combine this baseline with the IMS described in Section II-B. Across all populations in the IMS, one FOS is maintained since the linkage structure of a problem is not dependent on the population size. The incremental dependency updating as described in Section IV-B thus counts every generation equally, i.e., if due to IMS multiple populations are maintained simultaneously, incremental dependency updating is performed during every generation (regardless of the population index). Since a single solution from one population is used to perform the fitness difference testing for all populations, the population size of each population in the IMS is irrelevant. Every time a new FOS is built and the size of the biggest fully dependent linkage set has increased, n_{base} is recalculated and the populations with a population size smaller than n_{base} are stopped.

A second use of population sizing is if the linkage tree FOS is built, and not all variables in one linkage set are pairwise dependent, e.g., if $\mathcal{F}_i = \{1, 2, 3\}$ and $d_{1,2} = 0.5, d_{1,3} = 0, d_{2,3} = 0.5$. The population size is then not updated as described earlier, but the FOS set $\mathcal{F}_i$ is only added to $\mathcal{F}$ if $|\mathcal{F}_i| \leq \gamma_{max}$, with $\gamma_{max} = (\frac{n-17}{3})^{\frac{2}{3}}$ the maximal acceptable linkage set size for a population in the IMS of size n.

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

Algorithm 1 Incremental dependency updating

$pairs \leftarrow shufflePairs()$ ▷ All possible pairs, randomly ordered
$waitingCycles \leftarrow 0$ ▷ The number of waiting cycles
$k \leftarrow 0$ ▷ The number of cycles that have take place
$passedGenerations \leftarrow 0$ ▷ The generations that have taken place
$totalDependencies \leftarrow 0$ ▷ The total dependencies found
while not terminated **do**
  if $waitingCycles = 0$ **then**
    $Dependencies \leftarrow evaluateLpairs(pairs)$ ▷ Number of dependencies found
    $totalDependencies \leftarrow totalDependencies + Dependencies$
    $passedGenerations \leftarrow passedGenerations + 1$
    if ($Dependencies = 0$ and $totalDependencies \leq 2 \cdot passedGenerations$) or all pairs are evaluated **then**
      $waitingCycles \leftarrow 2^k$
      $k \leftarrow k + 1$
      $pairs \leftarrow shufflePairs()$
      $totalDependencies, Dependencies \leftarrow 0$
  else
    $waitingCycles \leftarrow waitingCycles - 1$
  continue RV-GOMEA
  ...

Linkage model	Description
RV-GOMEA-FBLT	RV-GOMEA with a learned fitness-based linkage tree
RV-GOMEA-FBMP	RV-GOMEA with a learned fitness-based marginal product model
RV-GOMEA-UNI	RV-GOMEA with a predefined univariate linkage model
RV-GOMEA-UNI5	RV-GOMEA with a predefined linkage model using blocks of 5 consecutive variables.
RV-GOMEA-FULL	RV-GOMEA with a predefined full linkage model
RV-GOMEA-LT	RV-GOMEA with a learned linkage tree model, based on Mutual Information as described in [5]
RV-GOMEA-DG	RV-GOMEA with a learned linkage structure based on differential grouping as proposed in [25]
AMaLGaM-UNI	AMaLGaM with a predefined univariate linkage model
AMaLGaM-FB	AMaLGaM with a learned fitness based linkage model

TABLE I: All algorithms used for our experiments.

V. Experiments

A. Benchmark algorithms

To conduct our experiments we compare the performance of RV-GOMEA in combination with our two proposed methods for fitness-based linkage learning, described in Section IV, to existing versions of RV-GOMEA and AMaLGaM. Table I gives an overview of all versions of RV-GOMEA and AMaLGaM that are used for our experiments, differing only in how the linkage model is defined or learned.

For our newly proposed methods, we make a distinction between a fitness-based linkage tree (RV-GOMEA-FBLT) and a fitness-based marginal product linkage structure (RV-GOMEA-FBMP) of which the differences between the resulting models are described in Section IV-C.

Because the second pruning approach of the fitness-based linkage learning method will always create a marginal product linkage structure, this method can also be used to create a linkage structure for AMaLGaM, equipping it with a linkage learning method for the first time. The MP linkage model is used to restrict the covariance matrix of AMaLGaM, i.e., the covariance is assumed to be 0 for variables in different FOS elements. AMaLGaM-FB will be compared to the previously described versions of RV-GOMEA. [2]

B. Benchmark problems

To study the impact of different types of linkage learning on the performance of RV-GOMEA, we first consider a set of six optimization problems. Whilst some of these problems are not decomposable, none of the used benchmark problems are fully dependent, aligned with the idea that real-world high-dimensional optimization problems are highly unlikely to have a linkage structure where each variable is dependent on every other variable. The problems we will consider are Sphere, Michalewicz, Rastrigin, Rosenbrock, Sum of Rotated Ellipsoid Blocks (SoREB) and an overlapping version of SoREB.

The first three benchmark functions exhibit no dependencies. First, we consider the Sphere function which is a widely-used benchmark for real-valued optimization. The Sphere function has a smooth landscape and no local minima.

$$f_{\text{Sphere}}(\boldsymbol{x}) = \sum_{i=0}^{\ell-1} \boldsymbol{x}_i^2$$

Second, we consider the Michalewicz function. In comparison with the smooth Sphere function, it contains $\ell!$ local optima that are unevenly distributed throughout the search space. The definition of the Michalewicz function is as follows, with $x_i \in [0, \pi]$:

[2]The C source code for these algorithms can be found at https://github.com/chantal-olieman/rv-gomea

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

$$f_{\text{Michalewicz}}(\boldsymbol{x}) = \sum_{i=0}^{\ell-1} \left[-\sin\left(\boldsymbol{x}_i\right) \cdot \sin\left(\left(i+1\right)\boldsymbol{x}_i^2/\pi\right)^{20} \right]$$

Next, the Rastrigin function is also a non-linear and multi-modal function but its local minima are evenly spread and superimposed on the Sphere function.

$$f_{\text{Rastrigin}}(\boldsymbol{x}) = 10\ell + \sum_{i=0}^{\ell-1} \left[\boldsymbol{x}_i^2 - 10\cos\left(2\pi\boldsymbol{x}_i\right) \right]$$

The fourth benchmark function we consider is the Rosenbrock function that contains a parabolic valley with one global optimum and one local optimum for $4 \leq \ell \leq 100$ [32]. Finding the global optimum in this valley is considered relatively hard. Finding the valley is trivial, but converging to the global minimum requires a search through this parabolic valley that requires differently oriented covariance matrices at different points during the search. By design, every consecutive pair of optimization variables in this function is dependent which results in $\ell - 1$ overlapping dependent components. The definition of the Rosenbrock function is as follows:

$$f_{\text{Rosenbrock}}(\boldsymbol{x}) = \sum_{i=0}^{\ell-2} \left[100\left(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i^2\right)^2 + \left(1 - \boldsymbol{x}_i\right)^2 \right]$$

The SoREB function uses a rotation function R_θ that defines the counterclockwise rotation of a vector around the origin by an angle of θ and an ellipsoid function $f_{\text{Ellipsoid}}$. Due to the construction of the SoREB function, the variables in every block of k consecutive optimization variables have strong dependencies, but are independent from any other optimization variables outside of their block. For our benchmark we use a block size of $k = 5$ and a rotation of $\theta = 45°$. Partial evaluations can only be performed by recalculating the fitness for all k variables present in the same block as x_i. The ellipsoid function and the SoREB function are defined as follows:

$$f_{\text{Ellipsoid}}(\boldsymbol{x}) = \sum_{i=0}^{\ell-1} \left[10^{\frac{6\ell}{\ell-1}} x_i^2 \right]$$

$$f_{\text{SoREB}}(vecxx, k) = \sum_{i=0}^{\ell/k-1} \left[f_{\text{Ellipsoid}}\left(R_\theta\left(\left[x_{ki}, \ldots, x_{k(i+1)-1}\right]\right)\right) \right]$$

The SoREB function is a problem containing only non-overlapping non-decomposable sub-components of size k. We define an overlapping version of this problem as OSoREB. In addition to the original SoREB problem, a second set of SoREB blocks is used with blocks of length 2 for every pair of consecutive parameters in successive blocks of SoREB with $k = 5$ (e.g., for x_4, x_5 and x_9, x_{10}). For partial evaluations every k variables in the same block as x_i need to be recalculated (at a cost of k/l). If x_i is either the first or last variable of a block, and is thus part of a pair of consecutive parameters in successive blocks, then that block of size 2 needs to be recalculated as well at a cost of $2/l$. The definition of OSoREB is as follows:

$$f_{\text{OSoREB}}(x, k) = f_{\text{SoREB}}(x, k) + \sum_{i=1}^{\ell/k-1} \left[f_{\text{Ellipsoid}}\left(R_\theta\left(\left[x_{ki-1}, x_{ki}\right]\right)\right) \right]$$

C. Setup

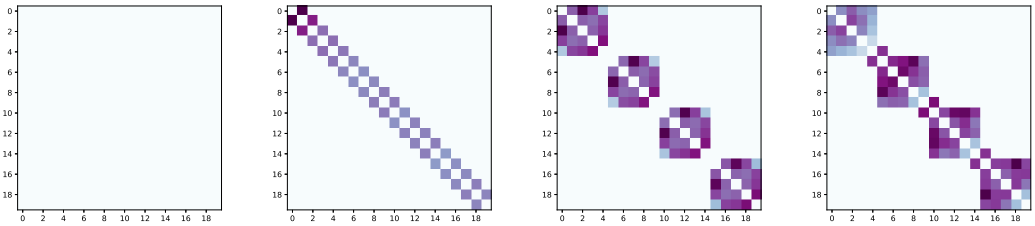
1) Evaluating linkage learning: We employ different means to verify the validity and impact of different linkage learning algorithms. The dependency matrices produced by our fitness-based method is compared to the matrices produced by the existing mutual information method. The dependency matrices give us valuable insight into the pairwise dependencies found during optimization, which is used to create the FOS structures used for GOM. For all benchmark problems, the pairwise dependencies are known and can thus be easily compared to the learned dependency matrices. Average dependency matrices are computed over 30 independent runs with $\ell = 50$ for all benchmark problems. As our linkage learning approach builds a model on the relative dependencies between variables, the heatmaps shown are normalized according to min-max feature scaling, such that all values range between $[0, 1]$ without loss of information. For all non-overlapping benchmark problems, we also verify whether the learned linkage sets corresponds to the combination of optimal linkage sets that capture all existing dependencies but do not combine independent variables.

2) Evaluating scalability: Another important aspect of our evaluation is the scalability analysis of RV-GOMEA on a subset of the benchmark functions. Scalability graphs are commonly used to benchmark the performance of optimization algorithms because they summarize the most important aspects of the algorithm's performance as well as provide a prediction regarding the performance on higher-dimensional problems. We compare the scalability of RV-GOMEA when using different linkage learning models as described in Section V-A. For a broader comparison of the previously existing versions of RV-GOMEA with different state-of-the-art EAs, we refer to [5]. For visibility we have only plotted RV-GOMEA-FBMP on non-overlapping problems since RV-GOMEA-FBLT both produce the same FOS structure and thus have the same scalability.

For every benchmark problem, 30 independent runs are performed with a time limit of 10^4 seconds (roughly 2 hours and 45 minutes). All experiments are performed on a 64-core (4 x 16-core AMD Opteron(tm) Processor 6386 SE) server running Fedora 28 where each run is performed on a single core. In every run the population is randomly initialized between $[-115, -100]$ for every variable, i.e., definitely not bracketing the optimum, except for $f_{\text{Michalewicz}}$, where we initialize between $[0, \pi]$, which is also its constrained range. A problem is considered to be sufficiently minimized if the elitist solution reaches a value to reach (VTR) of 10^{-10} if the optimum value is 0 (which is the case for all problems except $f_{\text{Michalewicz}}$) and 95% of the optimum for $f_{\text{Michalewicz}}$).

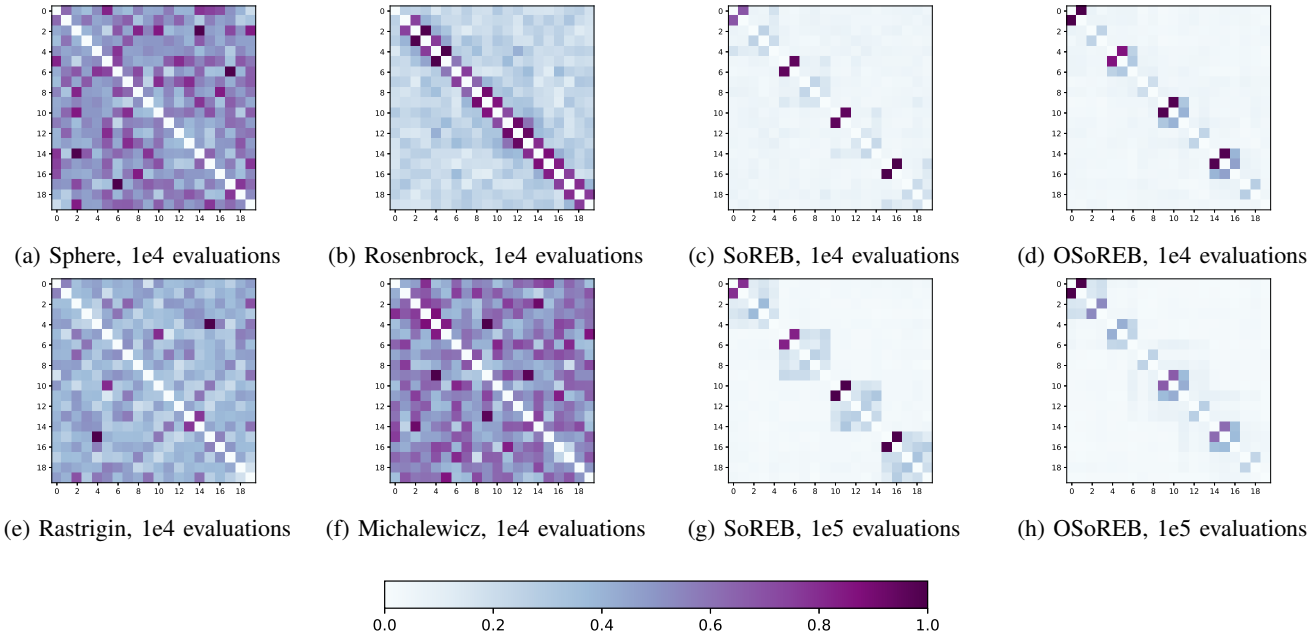
1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation



(a) Sphere, Rastrigin and Michalewicz (b) Rosenbrock (c) SoREB (d) OSoREB

Fig. 2: Heatmaps over 30 runs for $\ell = 50$ with the Fitness-Based measure



(a) Sphere, 1e4 evaluations (b) Rosenbrock, 1e4 evaluations (c) SoREB, 1e4 evaluations (d) OSoREB, 1e4 evaluations

(e) Rastrigin, 1e4 evaluations (f) Michalewicz, 1e4 evaluations (g) SoREB, 1e5 evaluations (h) OSoREB, 1e5 evaluations

Fig. 3: Heatmaps over 30 runs for $\ell = 50$ with the Mutual Information measure

If all runs are solved within the time limit, the problem size is doubled, until the maximum dimensionality of 10^4 is reached. Since various real-world optimization problems, as well as all our benchmark problems allow for partial evaluations, we have decided to focus mainly on grey-box optimization in order to obtain a realistic view of the performance of our algorithm on most real-world optimization problems.

VI. Results

A. Dependency matrices

Figure 2 shows heatmaps of the dependency matrices for all benchmark problems as calculated by RV-GOMEA-FBLT. For the mutual information measure shown in Figure 3, the dependency matrices for SoREB and OSoREB become more specific and less noisy after more evaluations, this becomes apparent in Figures 3h and 3g. All heatmaps shown are zoomed in to the first 20 dimensions, making it easier to inspect the dependencies, whilst still optimizing a 50-dimensional problem. On the fully decomposable problems: Sphere, Rastrigin and Michalewicz the mutual information measure is not able to correctly identify the independence of the variables. Even if variables are independent, a correlation is measured. In combination with the normalization of the dependency matrices used to create a FOS we can conclude that this measure encounters high amounts of noise for decomposable problems. The fitness-based method is able to correctly identify two independent variables, which results in a matrix containing only 0's for all three decomposable benchmark problems.

It is worth noting that the results obtained by our fitness-based method cannot be compared to the MI-based results without careful consideration. Whilst our fitness-based method can easily detect separability of two variables (whilst ignoring machine precision) it is close to impossible to detect separability whilst using the mutual information measure because selection causes the solutions to align with the fitness contours in the search space (Figure 1.). Further, the Mutual Information needs to be normalized because the absolute values have no real meaning. This makes the actual values computed

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

for the fitness-based and MI-based approaches incomparable. However, the way our algorithm uses information extracted from these two measures can be used to compare the measures themselves. Specifically, both measures are used to build a dependency structure (in our case FOS) for the problem at hand by iteratively merging the variables that are most closely linked according to the normalized two measures are used. Thus only relative values are important within one measure as this drives which variables are merged first.

By definition of the SoREB and OSoREB problems, the first two variables of a block exercise the highest influence on the total sum and thus show the strongest dependencies. This strong dependency can be seen in all Figures 2c, 2d, 3c, 3d, 3g and 3h, but only the fitness-based method (2c, 2d) and the mutual information method on SoREB after 10^5 evaluations have extracted the correct block structure without displaying noise between decomposable variables. One of the main drawbacks of the mutual information method now becomes immediately clear, because even though the dependencies will eventually be found, RV-GOMEA is able to already solve this instance of SoREB within 1e5 evaluations if a proper FOS is provided [5].

B. FOS structures

We can divide the benchmark problems into non-overlapping and overlapping optimization problems. Sphere, SoREB, Michalewicz, and Rastrigin are non-overlapping. The Rosenbrock problem and the OSoREB problem contain overlapping components. To better visualize the FOS structures produced by our methods, Figures 4 and 5 show the elements captured in a single FOS structure. In these figures, the horizontal axis represents the index of the optimization variables. Every linkage set is represented in one row of the figure, the highlighted x values mark the presence of that optimization variable in that one linkage set. Colors are used to improve visibility but do not contain any additional information.

1) Non-overlapping benchmarks: For the non-overlapping problems, the optimal FOS structures are known and can be compared with the FOS structures generated by RV-GOMEA-FB. We will look at the algorithm's FOS structures created for Sphere and SoREB. Rastrigin and Michalewicz are not considered here since for these problems the dependency matrix and thus FOS structure is equal to that of Sphere. The FOS structures found and used by RV-GOMEA-FB for Sphere and SoREB that can be seen in Figures 4a and 4b are as expected, considering the dependency matrices discussed in Section VI-A. For Sphere, it holds that a fully decomposable problem can best be represented by a univariate FOS consisting of exactly ℓ subsets, each containing a single optimization variable. As stated in Section V-B the SoREB function is rotated in blocks of k consecutive optimization variables with $k = 5$ in this case. Thus the dependencies of SoREB should be represented by a marginal product FOS containing blocks of size k as is the case in Figure 4b.

2) Overlapping benchmarks: The optimal FOS structures for overlapping benchmarks are unknown because no marginal product FOS can describe all dependencies without combining independent variables or leaving out dependencies. For these overlapping problems, a distinction is made between RV-GOMEA-FBLT and RV-GOMEA-FBMP. The latter linkage learning method creates a marginal product FOS, whereas the former continues to build a linkage tree, eventually containing all non-decomposable linkage sets. Figure 5 shows the FOS structures created for Rosenbrock and OSoREB by RV-GOMEA-FBLT and RV-GOMEA-FBMP where $l = 20$.

(b) SoREB

(a) Sphere

Fig. 4: FOS structures for $\ell = 50$ with every block representing a single linkage set

(a) Rosenbrock MP (b) Rosenbrock LT

(c) OSoREB MP (d) OSoREB LT

Fig. 5: Linkage structures for $\ell = 20$ with every block representing a single linkage set

C. Scalability analysis

Figure 6 shows the performance of different linkage learning methods in combination with RV-GOMEA on all six benchmark problems.

1) (Partially) Decomposable problems: For the fully decomposable problems Sphere, Rastrigin and Michalewicz, we can observe that RV-GOMEA-FBLT and RV-GOMEA-FBMP scale as well as RV-GOMEA-UNI and better than AMaLGaM-FB. Whilst RV-GOMEA is partially based on AMaLGaM,

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

the optimal mixing employed in RV-GOMEA has not been shown to outperform the model based EDA approach used by AMaLGaM on all benchmark problems before. In this paper suitable comparison has been made between RV-GOMEA and AMaLGaM as both algorithms have been provided with the same linkage learning method and better results have been obtained by RV-GOMEA, implying that the optimal mixing of (RV-)GOMEA has significant added value in the real-valued domain.

The incremental dependency updates have minimal overhead on the overall scalability as opposed to the original differential grouping (RV-GOMEA-DG) where the number of dependency checks needed to build a linkage model scales quadratically with the problem size.

On SoREB, a non-univariate linkage model is used as RV-GOMEA baseline, containing blocks of 5 consecutive optimization variables (RV-GOMEA-UNI5). RV-GOMEA-FBLT and RV-GOMEA-FBMP find the same structure (fig. 4b), but in the grey-box setting a small overhead is noticeable as the problem size increases. This overhead is caused by a decrease in the ratio of dependent to independent pairs as the problem size increases. By the definition of SoREB, every variable is dependent on the other $k-1$ variables in its block and has no dependency on all other $\ell - k$ problem variables. If the dimensionality of SoREB increases, the number of blocks increases, but the size of the blocks will remain equal to k. In other words, as ℓ grows larger, the number of independent variables $\ell - k$ becomes larger and the number of dependency checks that result in a measured dependency of 0 increases. Eventually, every possible pair will be checked, which ultimately still causes a quadratic overhead compared to the baseline RV-GOMEA-UNI5, which uses a predefined structure, but still scales better than the original fitness-based linkage learning used in RV-GOMEA-DG.

2) Non-decomposable problems: As the Rosenbrock and OSoREB problem contain non-decomposable sub-components, RV-GOMEA-FBLT and RV-GOMEA-FBMP generate different linkage models and their scalability should be evaluated independently. As shown in Section VI-B2, a full linkage tree FOS is built in RV-GOMEA-FBLT to capture the dependence between single parameters in different sub-components, whereas a marginal product structure is used by RV-GOMEA-FBMP. The former results in slightly better scalability on OSoREB implying that there is indeed added value in bigger FOS elements that can capture the linkage over partially dependent (sub)-components. On the Rosenbrock problem, RV-GOMEA-FBLT and RF-GOMEA-FBMP show similar scalability but are outperformed by RV-GOMEA-UNI, implying that even though this problem contains dependencies, it can still be efficiently solved by a univariate linkage model.

VII. Discussion

The method introduced in this paper is able to learn pairwise dependencies between variables online. There are, however, certain issues left unaddressed in finding the optimal linkage structure to any optimization problem. One of the key questions left unanswered is how to deal with problems containing overlapping sub-components. The optimal linkage structure to solve these benchmark problems is unknown, nor do we know whether a universally optimal linkage structure for these kinds of problems exists for RV-GOMEA. Whilst two of our benchmark problems contain overlapping components, the results between our two proposed linkage models did not vary much and we did not manage to find a definitive optimal structure for these benchmark problems. It is of value to note that recent efforts to move from marginal dependency models to conditional dependency models are a very likely candidate to overcome this issue [7]. The work in this article can readily be combined with these novel dependency modeling techniques.

While RV-GOMEA was shown to perform well on noise-free (real-world) problems [6], [18], [20], [41], future work remains to study how well these algorithms fare in case of noisy real-world problems and to find a suitable value for ϵ when determining dependencies in noisy (real-world) problems.

While our proposed approach defines relative dependencies on pairs of variables, it does not provide an absolute or relative minimal value to define non-separability of sub-components. Moreover, there is no guarantee that variables that have detectable linkage with our approach should always be considered inseparable during optimization. More research is required to determine whether it is possible to find a measure that allows for the further decomposition of weakly dependent sub-components.

When focusing on the pairwise dependency checks done on the Rosenbrock problem, it occurs that whilst the results obtained from these checks are as expected, pairwise dependencies might not be suitable to model the higher order dependence of this non-decomposable problem. Even though two variables x_i and x_{i+2} are not pairwise dependent, their optimal values both depend on the value of x_{i+1} and vice versa, making them dependent to some extent. These higher order dependencies cannot be captured by the pairwise dependency checks done by the approach introduced in this article.

It would furthermore be interesting to explore the use of more recent adaptations of differential grouping [27], [34], [35] in combination with RV-GOMEA. Also, it was recently shown that the mechanisms that drive the estimation and sampling of the Gaussian distributions as taken from AMaLGaM, can be replaced by that of the leading Gaussian-based ES, CMA-ES. This gives a different variant of RV-GOMEA that scales better on selected problems such as SoREB [4] we expect all results and comparisons between RV-GOMEA and AMaLGaM to straightforwardly extend to that variant of RV-GOMEA and CMA-ES with similar conclusions, but it would be interesting to immediately combine this also with the different recent adaptations of differential grouping.

Lastly, the order in which pairs are currently checked is random. As a consequence, our approach as proposed here is invariant to permutations of coordinates. If problem-specific knowledge indicates that certain pairs of variables are more

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

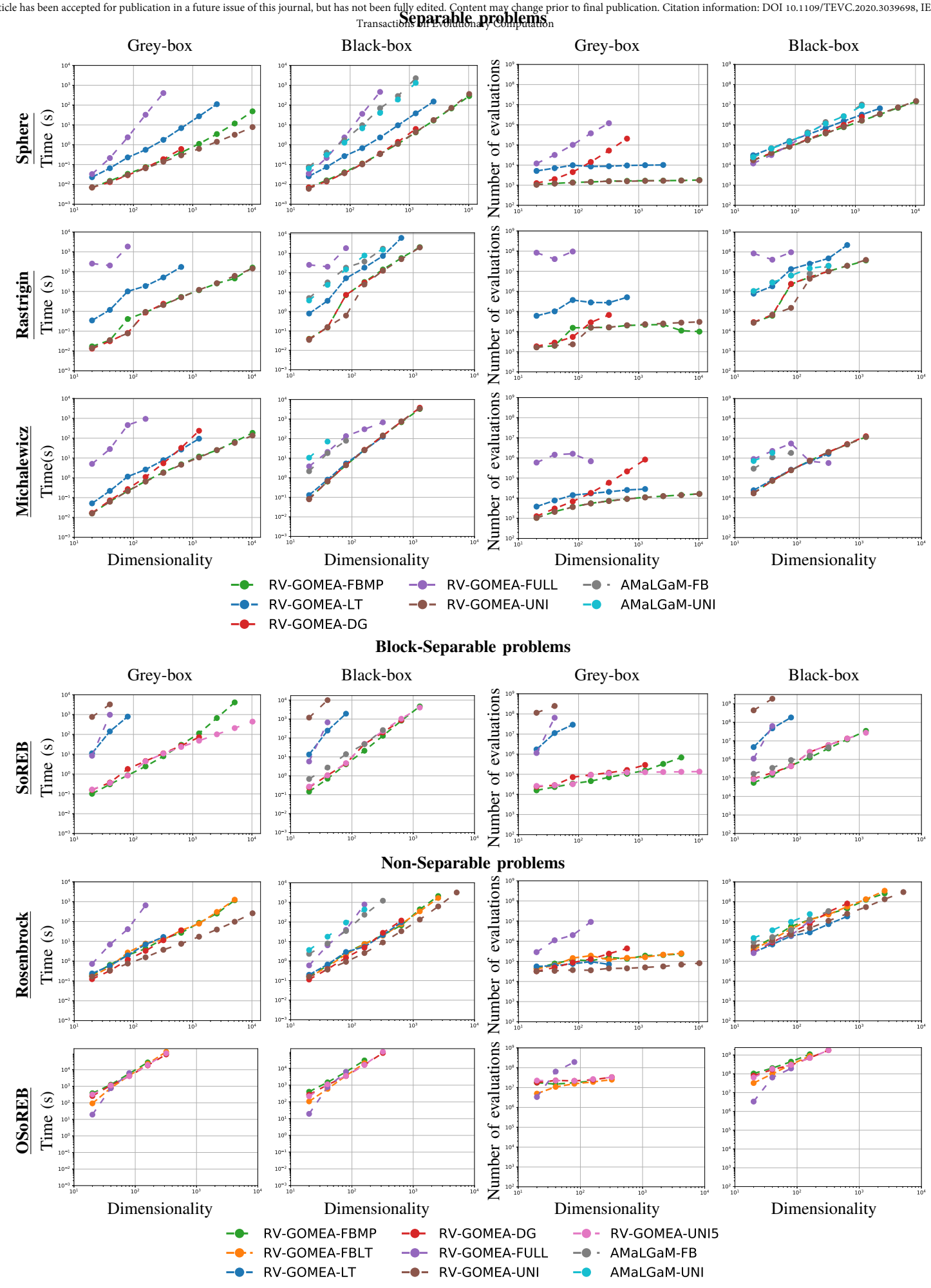


Fig. 6: Medians of scalability experiments with each data point being the median of 30 successful runs.

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

likely to be dependent than others, these pairs can be evaluated first, which is likely to improve RV-GOMEA-FBLT's performance even more on specific problems. However, this will likely disrupt the coordinate permutation invariance property, one of the currents strengths of RV-GOMEA-FBLT.

VIII. Conclusion

We have introduced a fitness-based linkage learning approach that can find pairwise dependencies between variables and build a linkage structure online without the need for any problem specific knowledge. The proposed method has been evaluated on different well-known benchmark problems and has proven to be efficient in determining the correct pairwise dependencies between variables. Two different methods have been proposed based on estimated pairwise dependencies to model the linkage structure of a problem. These methods have been integrated into RV-GOMEA and AMaLGaM and resulted in both algorithms being able to exploit important dependencies online. RV-GOMEA-FBMP and RV-GOMEA-FBLT have shown to outperform a state-of-the-art (for black-box scenarios) EA known as AMaLGaM upon which RV-GOMEA was based, whilst leveraging the same linkage model. Because a comparison between RV-GOMEA and AMaLGaM has never before resulted in better performance of one algorithm on all benchmark problems, this is the first time we can conclude that RV-GOMEA has outperformed the model-based EA that it was partially based on which clearly shows the added value of the optimal mixing employed by RV-GOMEA.

Whilst the two methods used for RV-GOMEA have equal performance on (partially) decomposable benchmarks, the generated linkage models differ for benchmark problems with overlapping sub-components. The difference lies in whether a (partial) linkage tree is built that captures all possible dependencies, or a marginal product linkage model is used. Whilst both methods have their strengths, RV-GOMEA-FBLT scales better on problems with strong dependencies and overlapping sub-components, thus proving to be a more robust method for large-scale real-world optimization problems with unknown problem structures.

Given the overall scalability of RV-GOMEA-FBLT, we conclude that the proposed algorithm is able to learn a linkage model online and scale as well as RV-GOMEA provided with the optimal structure, the current state-of-the-art for grey-box optimization.

References

[1] Peter A. N. Bosman, Jörn Grahl, and Dirk Thierens. Enhancing the performance of maximum–likelihood gaussian EDAs using anticipated mean shift. *International Conference on Parallel Problem Solving from Nature*, pages 133–143, 01 2008.

[2] Peter A. N. Bosman, Jörn Grahl, and Dirk Thierens. Benchmarking parameter-free AMaLGaM on functions with and without noise. *Evolutionary Computation*, 21(3):445–469, 2013.

[3] Peter A.N. Bosman and Dirk Thierens. Linkage neighbors, optimal mixing and forced improvements in genetic algorithms. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, GECCO '12, pages 585–592, New York, NY, USA, 2012. ACM.

[4] Anton Bouter, Tanja Alderliesten, and Peter Bosman. Achieving highly scalable evolutionary real-valued optimization by exploiting partial evaluations. *Evolutionary Computation*, pages 1–27, 06 2020.

[5] Anton Bouter, Tanja Alderliesten, Cees Witteveen, and Peter A. N. Bosman. Exploiting Linkage Information in Real-valued Optimization with the Real-valued Gene-pool Optimal Mixing Evolutionary Algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '17, pages 705–712, New York, NY, USA, 2017. ACM.

[6] Anton Bouter, Peter A. N. Bosman, and Tanja Alderliesten. A novel model-based evolutionary algorithm for multi-objective deformable image registration with content mismatch and large deformations: benchmarking efficiency and quality. *Proceedings of the SPIE Medical Imaging Conference 2017*, 10133, 2017.

[7] Anton Bouter, Stef Maree, Tanja Alderliesten, and Peter Bosman. Leveraging conditional linkage models in gray-box optimization with the real-valued gene-pool optimal mixing evolutionary algorithm. pages 603–611, 06 2020.

[8] Wenxiang Chen, Darrell Whitley, Renato Tinós, and Francisco Chicano. Tunneling between plateaus: Improving on a state-of-the-art maxsat solver using partition crossover. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, page 921–928, New York, NY, USA, 2018. Association for Computing Machinery.

[9] Charles Consel. Program adaptation based on program transformation. *ACM Computing Surveys*, 28, 02 2001.

[10] Kalyanmoy Deb and Christie Myburgh. Breaking the billion-variable barrier in real-world optimization using a customized evolutionary algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, GECCO '16, pages 653–660, New York, NY, USA, 2016. ACM.

[11] Brian Goldman and William Punch. Parameter-less population pyramid. *GECCO 2014 - Proceedings of the 2014 Genetic and Evolutionary Computation Conference*, 07 2014.

[12] Ilan Gronau and Shlomo Moran. Optimal implementations of UPGMA and other common clustering algorithms. *Inf. Process. Lett.*, 104:205–210, 12 2007.

[13] N. Hansen, S. D. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary Computation*, 11(1):1–18, 2003.

[14] Nikolaus Hansen, Anne Auger, Raymond Ros, Steffen Finck, and Petr Pošík. Comparing results of 31 algorithms from the black-box optimization benchmarking bbob-2009. In *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '10, page 1689–1696, New York, NY, USA, 2010. Association for Computing Machinery.

[15] Hoang Luong, Han La Poutré, and Peter Bosman. Exploiting linkage information and problem-specific knowledge in evolutionary distribution network expansion planning. *Evolutionary Computation*, 26(3):471–505, 2018.

[16] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 69:066138, 07 2004.

[17] Yong Liu, Xin Yao, Qiangfu Zhao, and Tetsuya Higuchi. Scaling up fast evolutionary programming with cooperative coevolution. *Proceedings of the IEEE Conference on Evolutionary Computation, ICEC*, 2, 07 2004.

[18] Hoang Luong, Tanja Alderliesten, Arjan Bel, Yury Niatsetski, and Peter Bosman. Application and benchmarking of multi-objective evolutionary algorithms on high-dose-rate brachytherapy planning for prostate cancer treatment. *Swarm and Evolutionary Computation*, June 2018.

[19] X. Ma, X. Li, Q. Zhang, K. Tang, Z. Liang, W. Xie, and Z. Zhu. A survey on cooperative co-evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 23(3):421–441, June 2019.

[20] Stef Maree, Hoang Luong, Ernst Kooreman, Niek van Wieringen, Arjan Bel, Karel Hinnen, Henrike Westerveld, Bradley Pieters, Peter Bosman, and Tanja Alderliesten. Evaluation of bi-objective treatment planning for high-dose-rate prostate brachytherapy—a retrospective observer study. *Brachytherapy*, February 2019.

[21] Marcin Molga and Czesław Smutnicki. Test functions for optimization needs. *Comput. Inform. Sci.*, pages 1–43, 01 2005.

[22] M. Munetomo and D. E. Goldberg. A genetic algorithm using linkage identification by nonlinearity check. In *IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.99CH37028)*, volume 1, pages 595–600 vol.1, 1999.

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2020.3039698, IEEE Transactions on Evolutionary Computation

[23] Masaharu Munetomo and David E. Goldberg. Linkage identification by non-monotonicity detection for overlapping functions. *Evolutionary Computation*, 7(4):377–398, 1999.

[24] Chantal Olieman. Fitness-based linkage learning in the real-valued gene-pool optimal mixing evolutionary algorithm. *Delft University of Technology, education repository*, 06 2019.

[25] M. N. Omidvar, X. Li, Y. Mei, and X. Yao. Cooperative co-evolution with differential grouping for large scale optimization. *IEEE Transactions on Evolutionary Computation*, 18(3):378–393, June 2014.

[26] M. N. Omidvar, X. Li, and X. Yao. Cooperative co-evolution with delta grouping for large scale non-separable function optimization. In *IEEE Congress on Evolutionary Computation*, pages 1–8, July 2010.

[27] M. N. Omidvar, M. Yang, Y. Mei, X. Li, and X. Yao. Dg2: A faster and more accurate differential grouping for large-scale black-box optimization. *IEEE Transactions on Evolutionary Computation*, 21(6):929–942, 2017.

[28] Mohammmad Nabi Omidvar, Ming Yang, Yi Mei, Xiaodong Li, and Xin Yao. Dg2: A faster and more accurate differential grouping for large-scale black-box optimization. *IEEE Transactions on Evolutionary Computation*, 21(6):929–942, Dec 2017.

[29] Martin Pelikan, Kumara Sastry, and Erick Cantu-Paz. *Scalable optimization via probabilistic modeling. From algorithms to applications*, volume 33. 01 2006.

[30] Mitchell Potter and Kenneth De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary computation*, 8:1–29, 02 2000.

[31] Ralf Salomon. Reevaluating genetic algorithm performance under coordinate rotation of benchmark functions - a survey of some theoretical and practical aspects of genetic algorithms. *BioSystems*, 39:263–278, 1995.

[32] Yun-Wei Shang and Yu-Huang Qiu. A note on the extended rosenbrock function. *Evolutionary Computation*, 14:119–126, 03 2006.

[33] Yi-En Su and Tian-Li Yu. Use model building on discretization algorithms for discrete EDAs to work on real-valued problems. *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014*, pages 2491–2498, 07 2014.

[34] Yuan Sun, Michael Kirley, and Saman Halgamuge. A recursive decomposition method for large scale continuous optimization. *IEEE Transactions on Evolutionary Computation*, PP:1–1, 11 2017.

[35] Yuan Sun, Mohammmad Nabi Omidvar, Michael Kirley, and Xiaodong Li. Adaptive threshold parameter estimation with recursive differential grouping for problem decomposition. *GECCO '18: Proceedings of the Genetic and Evolutionary Computation Conference*, 07 2018.

[36] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, page III–1139–III–1147. JMLR.org, 2013.

[37] Dirk Thierens. Scalability problems of simple genetic algorithms. *Evolutionary computation*, 7:331–52, 02 1999.

[38] Dirk Thierens. The linkage tree genetic algorithm. In *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature: Part I*, PPSN'10, page 264–273, Berlin, Heidelberg, 2010. Springer-Verlag.

[39] Dirk Thierens and Peter A.N. Bosman. Optimal mixing evolutionary algorithms. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, GECCO '11, pages 617–624, New York, NY, USA, 2011. ACM.

[40] Renato Tinós, Darrell Whitley, and Francisco Chicano. Partition crossover for pseudo-boolean optimization. In *Proceedings of the 2015 ACM Conference on Foundations of Genetic Algorithms XIII*, FOGA '15, pages 137–149, New York, NY, USA, 2015. ACM.

[41] Medha V. Wyawahare, Pradeep M. Patil, and Hemant K. Abhyankar. Image registration techniques: An overview. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, pages 11–39, 2009.

[42] Zhenyu Yang, Ke Tang, and Xin Yao. Large scale evolutionary optimization using cooperative coevolution. *Inf. Sci.*, 178:2985–2999, 08 2008.

1089-778X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: TU Delft Library. Downloaded on December 15,2020 at 12:09:46 UTC from IEEE Xplore. Restrictions apply.