Leveraging Conditional Linkage Models in Gray-box Optimization with the Real-Valued Gene-pool Optimal Mixing Evolutionary Algorithm

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

Stefanus C. Maree
Amsterdam UMC
University of Amsterdam
Amsterdam, The Netherlands
S.C.Maree@amsterdamumc.nl

Tanja Alderliesten
Leiden University Medical Center
Leiden, The Netherlands
T.Alderliesten@lumc.nl

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

ABSTRACT

Often, real-world problems are of the gray-box type. It has been shown that the Real-Valued Gene-pool Optimal Mixing Evolutionary Algorithm (RV-GOMEA) is in principle capable of exploiting such a Gray-Box Optimization (GBO) setting using linkage models that capture dependencies between problem variables, resulting in excellent performance and scalability on both benchmark and real-world problems that allow for partial evaluations. However, linkage models proposed for RV-GOMEA so far cannot explicitly capture overlapping dependencies. Consequently, performance degrades if such dependencies exist. In this paper, we therefore introduce various ways of using conditional linkage models in RV-GOMEA. Their use is compared to that of non-conditional models, and to VkD-CMA, whose performance is among the state of the art, on various benchmark problems with overlapping dependencies. We find that RV-GOMEA with conditional linkage models achieves the best scalability on most problems, with conditional models leading to similar or better performance than non-conditional models. We conclude that the introduction of conditional linkage models to RV-GOMEA is an important contribution, as it expands the set of problems for which optimization in a GBO setting results in substantially improved performance and scalability. In future work, conditional linkage models may prove to benefit the optimization of real-world problems.

CCS CONCEPTS

• **Mathematics of computing → Evolutionary algorithms**.

GECCO '20, July 8–12, 2020, Cancún, Mexico

ACM ISBN 978-1-4503-7128-5/20/07.
https://doi.org/10.1145/3377930.3390225

KEYWORDS

Gray-box optimization, GOMEA, linkage modeling

ACM Reference Format:
Anton Bouter, Stefanus C. Maree, Tanja Alderliesten, and Peter A.N. Bosman. 2020. Leveraging Conditional Linkage Models in Gray-box Optimization with the Real-Valued Gene-pool Optimal Mixing Evolutionary Algorithm. In *Genetic and Evolutionary Computation Conference (GECCO '20), July8–12, 2020, Cancún, Mexico.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3377930.3390225

1 INTRODUCTION

Traditionally, research in the field of Evolutionary Algorithms (EAs) has mainly been focused on the optimization of Black-Box Optimization (BBO) problems. In a BBO setting, no prior information of the objective function is assumed to be available, and only function evaluations can reveal the structure of the problem to be optimized. However, in various real-world problems [14, 24] the underlying structure, or even an exact definition, of the optimization problem is known. Despite this knowledge, such problems may still be difficult to optimize. As such, problem-specific knowledge has recently been used in EAs to greatly improve performance, e.g., through the use of problem-specific variation operators [14], partition crossover [13, 37], or partial evaluations [10].

Partial evaluations are used to efficiently calculate the objective value of a solution following the modification of a subset of its variables. The possibility to use partial evaluations is by no means an indicator of the difficulty of the problem, as partial evaluations may be possible for problems that are, e.g., non-separable, non-smooth, multi-modal, and/or multi-objective. It has been shown that partial evaluations can be efficiently leveraged in the Real-Valued Gene-pool Optimal Mixing Evolutionary Algorithm (RV-GOMEA) [10]. RV-GOMEA achieves excellent performance and scalability on single-objective and multi-objective benchmark problems [10, 11], but also on real-world problems such as medical deformable image registration [9], and treatment planning for prostate cancer [24]. However, the problems considered thus far are either separable or have relatively weak dependencies.

It is as of yet unclear if it is possible to efficiently leverage partial evaluations in case the optimization problem has strong overlapping dependencies, because partial evaluations are only beneficial when some form of selection is applied after the modification of a subset of variables of a solution. Applying variation not to all variables at once, but only to subsets of variables, may however be detrimental to the optimization of problems with strong overlapping dependencies. Such problems cannot be decomposed into disjoint sets of independent variables, and are therefore generally difficult to solve for optimization methods relying on decompositions if the dependencies themselves are strong, i.e., a rotated ellipsoid with high condition number [25, 35]. They can however be efficiently optimized by other decomposition methods like VkD-CMA [2, 3], a variant of the well-known Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [19], using a covariance matrix that is parameterized by a smaller number of parameters than the problem dimensionality. VkD-CMA can however not directly benefit from the possibility of partial evaluations, as variation is always applied to all problem variables at once.

In this paper, we aim to improve the performance of RV-GOMEA on problems with strong overlapping dependencies in a GBO setting. For this purpose, we explore the use of different novel linkage models, which are used by RV-GOMEA to model dependencies between problem variables. In particular, we explore linkage models that can capture conditional dependencies, as these models may benefit the optimization of problems with overlapping dependencies.

We first introduce the GBO setting in more detail in Section 2, followed by background on RV-GOMEA in Section 3. In Section 4 we discuss a number of conditional linkage models, and how they can be employed in RV-GOMEA. In Section 5 the performance and scalability of these conditional linkage models is compared to non-conditional models, and to VkD-CMA. The results are discussed in Section 6, followed by our conclusions in Section 7.

2 GRAY-BOX OPTIMIZATION

We consider the optimization of an objective function $f(\boldsymbol{x}) : \mathbb{R}^\ell \rightarrow \mathbb{R}$, subject to minimization. We refer to problem variables as $\boldsymbol{X} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_\ell\}$, which is indexed through the set $\mathcal{I} = [1, \ldots, \ell]$. A realization of the problem variables, i.e., a solution, is denoted $\boldsymbol{x} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell\}$.

In particular, we focus on a GBO setting (as previously defined [8]) that allows for partial evaluations, meaning that the objective value of a solution can be efficiently computed after the modification of a subset of its variables. Let $Y \subseteq \mathcal{I}$ refer to a subset of problem variables. We denote by $\boldsymbol{x}_Y$ the variables of $\boldsymbol{x}$ corresponding to the indices specified by Y. In this setting, the objective function is composed of q sub-functions $\boldsymbol{F} = \{f_1, f_2, \ldots, f_q\}$. A sub-function $f_j(\boldsymbol{x}_{\mathbb{I}_j}) \in \boldsymbol{F}$ takes problem variables $\boldsymbol{x}_i$ with $i \in \mathbb{I}_j \subseteq \mathcal{I}$ as input, where $\mathbb{I} = \{\mathbb{I}_1, \mathbb{I}_2, \ldots, \mathbb{I}_q\}$ is given by the problem definition. Each sub-function itself is treated as a black box. Moreover, a sub-function f_i is assumed not to be separable itself.

Problem variables $\boldsymbol{x}_u$ and $\boldsymbol{x}_v$ with $u, v \in \mathcal{I}$ are considered to be directly dependent, i.e., $\boldsymbol{x}_u \leftrightarrow \boldsymbol{x}_v$, when a sub-function $f_j(\boldsymbol{x}_{\mathbb{I}_j}) \in F$ exists with $\{u, v\} \subseteq \mathbb{I}_j$. Problem variables $\boldsymbol{x}_u$ and $\boldsymbol{x}_v$ are indirectly dependent when a set $\{u, \ldots, v\} \subseteq \mathcal{I}$ exists such that $\boldsymbol{x}_u \leftrightarrow \cdots \leftrightarrow \boldsymbol{x}_v$ but not $\boldsymbol{x}_u \leftrightarrow \boldsymbol{x}_v$.

An objective function in a GBO setting can be written as:

$$f(\boldsymbol{x}) = g\left(f_1(\boldsymbol{x}_{\mathbb{I}_1}) \oplus f_2(\boldsymbol{x}_{\mathbb{I}_2}) \oplus \cdots \oplus f_k(\boldsymbol{x}_{\mathbb{I}_q})\right), \tag{1}$$

with $\oplus$ a binary operator that has a known inverse $\ominus$ (e.g., addition or multiplication[8]), and $g : \mathbb{R} \rightarrow \mathbb{R}$ any (potentially non-linear) function.

A change in some variable $\boldsymbol{x}_u$ requires the evaluation of each sub-function $f_j(\boldsymbol{x}_{\mathbb{I}_j})$ for which $u \in \mathbb{I}_j$. If g is not the identity function, the intermediate result of $f_1(\boldsymbol{x}_{\mathbb{I}_1}) \oplus \cdots \oplus f_q(\boldsymbol{x}_{\mathbb{I}_q})$ must be stored in memory for each solution in the population.

We assume that the computational complexity of each sub-function is approximately equal. Therefore, given a total of q sub-functions, a partial evaluation of n sub-functions is counted as a fraction n/q of an evaluation in a GBO setting.

Given the GBO definition of an optimization problem, as in Equation (1), a Variable Interaction Graph (VIG) [37] can be created. A VIG is an undirected graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ where each vertex $v \in V$ corresponds to a problem variable $\boldsymbol{x}_v$, and each edge $(u, v) \in E$ denotes that problem variables $\boldsymbol{x}_u$ and $\boldsymbol{x}_v$ are directly dependent. Problem variables $\boldsymbol{x}_u$ and $\boldsymbol{x}_v$ are indirectly dependent if a path between vertices u and v exists in the VIG, but not an edge (u, v).

3 REAL-VALUED GENE-POOL OPTIMAL MIXING EVOLUTIONARY ALGORITHM

In this section we present an outline of RV-GOMEA. A more detailed description is provided in [10].

3.1 Linkage Model

In RV-GOMEA, dependencies between problem variables are explicitly modeled by a linkage model. Such linkage models are described by a Family of Subsets (FOS) $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_m\}$ that is a subset of the powerset of $\mathcal{I}$, i.e., $\mathcal{F}_i \subseteq \mathcal{I}$. Each FOS element $\mathcal{F}_i \in \mathcal{F}$ models a group of variables that is considered to be jointly dependent.

The univariate FOS $\mathcal{F} = \{\{1\}, \{2\}, \ldots, \{\ell\}\}$, which models all problem variables to be independent, is considered to be the simplest FOS model. A marginal product FOS models disjoint sets of dependent variables, i.e., $\mathcal{F}_i \cap \mathcal{F}_j = \emptyset$ for $\mathcal{F}_i, \mathcal{F}_j \in \mathcal{F}$. The full FOS models all variables as jointly dependent, i.e., $\mathcal{F} = \{\{1, \ldots, \ell\}\}$.

A linkage tree is a linkage model, which was first introduced for the discrete GOMEA [36], that is capable of modeling hierarchical dependencies. A linkage tree FOS firstly includes all singleton elements. All other FOS elements are the union of two smaller FOS elements. Formally stated, for each $\mathcal{F}_i \in \mathcal{F}$ there exist $\mathcal{F}_j, \mathcal{F}_k \in \mathcal{F}$ such that $\mathcal{F}_j \cap \mathcal{F}_k = \emptyset$ and $\mathcal{F}_j \cup \mathcal{F}_k = \mathcal{F}_i$. Building a linkage tree is often done using the Unweighted Pair Grouping Method with Arithmetic-mean (UPGMA) [18]. Therefore, pairs of FOS elements with the strongest dependencies are the first to be merged. The dependency strength of variables can either be determined based on problem-specific knowledge or based on mutual information.

The elements of a linkage tree may be restricted to a maximum size k. In this case, the linkage tree is generally learned offline based on problem-specific information, in which case we refer to it as the Bounded Fixed Linkage Tree (BFLT) [10].

3.2 Gene-pool Optimal Mixing

RV-GOMEA uses Gene-pool Optimal Mixing (GOM) as its variation operator. During GOM, variation is applied based on the dependencies defined by the linkage model $\mathcal{F}$. During each generation, one iteration of GOM is applied to each solution in the population for each FOS element $\mathcal{F}_i$. One iteration of GOM, applied to a parent solution in the population, consists of the sampling of new values for the problem variables contained in the FOS element in question. These values are inserted into the parent and this solution is evaluated. Because few variables change, as specified by $\mathcal{F}_i$, a partial evaluation can be performed. The modification of the parent is only accepted if it results in an improvement. Otherwise, the modification is accepted with probability p^{accept}. If the modification is rejected, the parent is returned to its previous state.

A sampling model is used to generate new problem variables. This can be taken from any EA that has an identifiable sampling model, such as CMA-ES [19]. Here, we build on the original version of RV-GOMEA that uses the sampling model of AMaLGaM [5].

Sampled values for problem variables are generated from a Multivariate Gaussian (MVG) that is estimated with maximum likelihood based on the selection $\mathcal{S}$, consisting of the $\lfloor \tau n \rfloor$ best solutions in the population $\mathcal{P}$, with n the population size. For each FOS element $\mathcal{F}_j$, an MVG $\mathcal{N}(\hat{\boldsymbol{\mu}}_j, \hat{\mathbf{C}}_j)$ is estimated for all problem variables in $\mathcal{F}_j$. Each such distribution is scaled by a distribution multiplier that is updated each generation based on the location of improvements found. This process is called Adaptive Variance Scaling (AVS).

Furthermore, RV-GOMEA includes a generational shift that is applied to n^{AMS} solutions in the population, named the Anticipated Mean Shift (AMS), and a procedure aimed at avoiding stagnation caused by spreading the population across multiple local minima, named the Forced Improvement (FI) procedure. The FI procedure is triggered when a solution has not been improved for a number of generations larger than the maximum No-Improvement Stretch (NIS). These procedures are described in more detail in [10]. High-level pseudo-code of RV-GOMEA is shown in Algorithm 1. The GOM procedure is shown in Algorithm 2.

Algorithm 1 RV-GOMEA

```
1:  procedure RV-GOMEA(n,τ)
2:      P ← InitializeAndEvaluatePopulation(n)
3:      F ← InitializeLinkageModel()
4:      while ¬TerminationCriterionSatisfied() do
5:          P_1 ← x^elitist
6:          for F_j ∈ F do                          ▷ Random order
7:              S ← ⌊τn⌋ best in P
8:              P({X_u : u ∈ F_j}) ← MaxLikelihoodEstimate(S)
9:              for x ∈ P_{2...n} do
10:                 GenepoolOptimalMixing(x, F_j)
11:             AdaptiveVarianceScaling(F_j)
12:         for x ∈ P_{2...n_AMS+1} do
13:             AnticipatedMeanShift(x)
14:         for x ∈ P_{2...n} do
15:             if NIS(x) > NIS^MAX then
16:                 ForcedImprovement(x)
```

Algorithm 2 Gene-pool Optimal Mixing

```
1: procedure GenepoolOptimalMixing(x, F_j)
2:     b ← x_{F_j}
3:     x_{F_j} ← P({X_u : u ∈ F_j})                  ▷ Sampling
4:     f_o ← PartialEvaluation(x, f_x, F_j)
5:     if f_o < f_x or U(0,1) < p^accept then f_x ← f_o
6:     else x_{F_j} ← b
```

Here, we deviate from the original RV-GOMEA definition in that selection is now performed at the start of each iteration of GOM, whereas previously, selection was only performed at the start of each generation [10]. This modification is important when the linkage model is not a marginal product. In such a case, variation is applied to a problem variable more than once per generation. The estimated distribution may then become outdated throughout a generation, making GOM less effective and efficient. Supporting experiments are included as supplementary material.

4 LINKAGE MODELING

Different methods for the modeling of dependencies between variables have previously been used for the design of Estimation of Distribution Algorithms (EDAs) [22]. Some of the models used by early EDAs include univariate [21, 28], tree-based [12], or multivariate [20] models. Marginal product models were however shown to be too restrictive to solve difficult problems [6]. Alternatively, Bayesian networks were later used in EDAs for discrete [26, 27] and real-valued optimization [1, 4, 7, 16]. Also Markov networks, or Markov random fields [34], have been proposed for use in (discrete) EDAs [33], for example in the Markovianity-based Optimization Algorithm (MOA) [32], the Distribution Estimation using Markov networks (DEUM) [30] algorithm, or the Markov Network Estimation of Distribution Algorithm (MN-EDA) [29]. Factorized Gaussian Markov networks were previously used in real-valued optimization in the Gaussian Markov Random Field EDA (GMRF-EDA) [23], though the latter method is focused on problems with disjoint sets of dependent variables. Here, we focus on the introduction of conditional linkage models, in particular based on Markov and Bayesian networks, because such models appear most suitable for problems with overlapping dependencies and integration with RV-GOMEA.

4.1 Non-conditional Linkage Models

A marginal product linkage model is an appropriate model for problems with disjoint groups of dependent variables. However, this is not efficient in case long chains of (pairwise) dependent variables exist, as such dependencies must be modeled by a single joint group. This requires the estimation of an ℓ-variate MVG, which becomes computationally expensive to sample from for large ℓ and requires a relatively large population to accurately estimate.

We illustrate the difficulty of optimizing problems with strong overlapping dependencies through the $f^{REBChain}(\boldsymbol{x}, c, \theta)$ function, defined in Equation (2), where $R^{\theta}(\boldsymbol{x})$ applies a rotation to $\boldsymbol{x}$ of θ degrees around the origin in a clockwise direction for each pair of variables. This function is the sum of a 2-dimensional (2-D) rotated ellipsoid function with condition number c (see Equation (3)) for each pair of consecutive variables. In Figure 1 we show

the number of evaluations required to solve the 20-D $f^{REBChain}$ function for varying rotation angles and condition numbers using RV-GOMEA with different marginal linkage models. In all cases, the Interleaved Multistart Scheme (IMS) [10] was used, which scales the population size over time by interleaving runs with gradually increasing population sizes, using an initial population size of 20.

Figure 1a clearly shows that a univariate linkage model is not adequate in case of a large condition number and large rotation angles, i.e., when dependencies are strong. Results for the BFLT, shown in Figure 1b, shows that a hierarchical model that does not include the joint distribution, leads to only minor improvements. The best performance for strongly dependent problems is obtained with the full linkage model, shown in Figure 1c. It is however known that the full linkage model does not scale well to very high-dimensional problems, and cannot benefit from a GBO setting. Furthermore, a full linkage model is often overkill, as most variables are not directly and/or strongly dependent in real-world problems.

$$f^{REBChain}(\boldsymbol{x}, c, \theta) = \sum_{i=1}^{\ell-1} f^{2DEllipsoid}(R^{\theta}(\{\boldsymbol{x}_i, \boldsymbol{x}_{i+1}\}), c), \tag{2}$$

$$f^{2DEllipsoid}(\boldsymbol{x}, c) = \boldsymbol{x}_1^2 + 10^c \boldsymbol{x}_2^2. \tag{3}$$

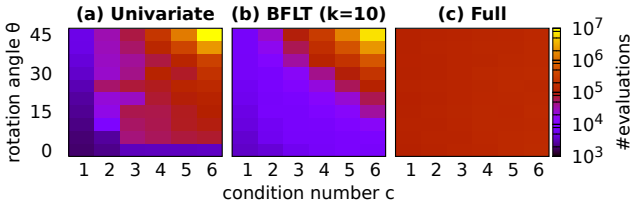

Figure 1: Median (30 runs) number of evaluations (color-coded) required to find the optimum with a precision of 10^{-10} of the 20-D $f^{REBChain}(\boldsymbol{x}, c, \theta)$, or the budget of 10^7 function evaluations was used, for RV-GOMEA using different marginal linkage models.

4.2 Bayesian Factorization

Updating variables $\mathbb{I}_j$ pertaining to a single sub-function f_j while taking variables pertaining to overlapping sub-functions into account, can be done by conditional sampling. To be precise, instead of sampling new values for problem variables $\boldsymbol{x}_{\mathbb{I}_j}$ using the multivariate marginal $P(\boldsymbol{X}_{\mathbb{I}_j})$, one could sample conditionally on the other variables of $\boldsymbol{x}$, i.e., $\boldsymbol{x}_{\neg\mathbb{I}_j}$ with $\neg\mathbb{I}_j = \mathcal{I} \setminus \mathbb{I}_j$, giving,

$$\boldsymbol{x}_{\mathbb{I}_j} \leftarrow P(\boldsymbol{X}_{\mathbb{I}_j} | \boldsymbol{X}_{\neg\mathbb{I}_j} = \boldsymbol{x}_{\neg\mathbb{I}_j}). \tag{4}$$

In this way, partial evaluations can still be performed.

As we use an MVG distribution to sample from, its conditional distribution is also an MVG, which is easy to sample from [15]. However, directly using this conditional distribution to sample from is also computationally expensive, as conditioning on all other variables has a computational complexity of $O(\ell^3)$. To reduce computational cost, we can exploit the VIG. The VIG $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ represents a Markov network (or field), and specifically a Gaussian Markov Field (GMF), between problem variables. In a GMF, the probability density function of a variable is completely defined in terms of a conditional probability function given its neighbors. Therefore, samples can be drawn from a GMF without modeling the joint probability distribution of all variables [30, 31]. Let $\mathrm{N}(\mathbb{I}_j)$ be the set of all neighbors of $\mathbb{I}_j$ in the VIG. Then, the conditional probability in Equation (4) simplifies to,

$$P\left(\boldsymbol{X}_{\mathbb{I}_j} | \boldsymbol{X}_{\neg\mathbb{I}_j} = \boldsymbol{x}_{\neg\mathbb{I}_j}\right) = P\left(\boldsymbol{X}_{\mathbb{I}_j} | \boldsymbol{X}_{N(\mathbb{I}_j)} = \boldsymbol{x}_{N(\mathbb{I}_j)}\right). \tag{5}$$

This has two advantages. First, it reduces computational cost of sampling from the conditional distribution, as sampling from $P(\boldsymbol{X})$ has a complexity of $O(\ell d^3)$ when the maximum degree of the VIG is bounded by a constant d. Second, the sample distribution explicitly takes conditional independencies into account. Estimating the sample distribution of a GMF, instead of the full MVG, requires fewer parameters to be estimated, which can therefore be done with a smaller population size, furthermore improving efficiency.

4.2.1 Partitioning. More in general, large groups of variables may be sampled from a joint MVG. Given a partitioning of variables C, the variables in each partition C_i can be jointly sampled conditioned on $\boldsymbol{X}_{N(C_i)}$. For a non-overlapping partitioning of variables $C \subset \mathcal{P}(\mathcal{I})$, i.e., with $C_i \cap C_j = \emptyset$ for $C_i, C_j \in C$, this results in,

$$P(\boldsymbol{X}) = \prod_{C_j \in C} P\left(\boldsymbol{X}_{C_j} | \boldsymbol{X}_{N(C_j)}\right). \tag{6}$$

The larger the clusters C_j, the less the independence structure imposed by the VIG is leveraged. This effect is smaller if the variables in C_j form a clique in the VIG, and guaranteed to be minimal if the clusters are set to be the singletons. If a univariate partitioning, i.e., $C = \{\{1\}, \ldots, \{\ell\}\}$, is used, Equation (6) represents a univariate conditional factorization. We refer to this factorization as UCond.

When each factor C_i may consist of multiple variables, Equation (6) represents a multivariate conditional factorization. We aim to have these multivariate factors correspond to the sub-functions, as all variables required for a sub-function may be jointly sampled. As such, these factors are constructed by traversing $\boldsymbol{G}$ in a breadth-first order, and finding maximal cliques. Given the current vertex u and its previously visited neighbors, a maximal clique including both u and these neighbors is constructed, if possible. A factor C_i is then defined as this clique, excluding the previously visited neighbors, and is added to C. This process continues until each variable (or node) is contained by a factor C_i. We refer to this factorization as MCond.

Note that the univariate and multivariate conditional factorizations may define the same probability distribution. Their distinction becomes important when a selection step is performed after sampling variables for some factor C_i, as discussed in Section 4.3.1.

4.2.2 Sampling. We use forward sampling [33] to sample from the distribution described in Equation (6). This requires a topological ordering of the GMF. As each partition C_i is jointly sampled, it is sufficient to direct the edges between vertices in different partitions. This is done by defining an order $\boldsymbol{O}$ in which the partitions of C are sampled. The order $\boldsymbol{O}$ is defined as a function mapping the index of a partition to its precedence. As such, C_u precedes C_v if $\boldsymbol{O}(u) < \boldsymbol{O}(v)$, indicating that each edge $(q, w) \in \boldsymbol{E}$ with $q \in C_u$ and $w \in C_v$ is directed as $w \leftarrow q$. Variables are then sampled conditioned on all variables from which an incoming edge exists.

Directing the edges of the GMF will generally result in a different joint probability distribution than represented by the GMF, and may break dependencies between variables due to d-separation

[17]. This is undesirable, because we aim to model a conditional dependency between each pair of variables between which a path exists in the VIG. To minimize the number of breaks, we traverse the network in a breadth-first manner, starting from the partition of a randomly selected vertex. This ensures that only the first partition is sampled independently from all others, and all remaining variables are sampled conditionally dependent. For each partition C_j, values are sampled for problem variables $\boldsymbol{X}_u$ with $u \in C_j$. These values are sampled from a normal probability distribution that is conditioned on the problem variables corresponding to the neighboring vertices of $u \in C_j$ that have already been sampled. The values of the starting partition are sampled independently of all other variables. Doing so simplifies the probability distribution in Equation (6) to,

$$P\left(\boldsymbol{X}_{C_j} | \boldsymbol{X}_{N(C_j)} = \boldsymbol{x}_{N(C_j)}\right) \approx P\left(\boldsymbol{X}_{C_j} | \boldsymbol{X}_{\pi(C_j)} = \boldsymbol{x}_{\pi(C_j)}\right), \quad (7)$$

with $\pi(C_j) = \{u \in N(C_j) : \boldsymbol{O}(u) < \boldsymbol{O}(v) \text{ for all } v \in C_j\}$ the neighbors of C_j that preceded C_j in the sampling order $\boldsymbol{O}$.

Figure 2 shows a VIG of the 13-D REB5SmallOverlap (Equation (15)) problem, given a UCond factorization. Figure 3 then shows the same VIG color coded according to the MCond factorization. An edge $u \rightarrow v$ indicates that $\boldsymbol{X}_v$ is sampled conditioned on $\boldsymbol{X}_u$. Edges are directed corresponding to an arbitrary (breadth-first) ordering starting from vertex 7.

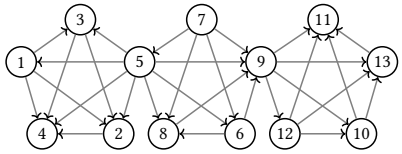

Figure 2: A VIG for the 13-D REB5SmallOverlap problem, showing a potential breadth-first ordering starting from randomly selected vertex 7.

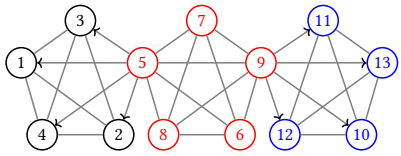

Figure 3: A VIG for the 13-D REB5SmallOverlap problem, showing a color-coded partitioning starting from randomly selected vertex 7.

4.3 GOM with Conditional Linkage Model

Because the application of GOM is the main strength of RV-GOMEA, and the way in which GOM is applied substantially influences performance and scalability, we explore different options of using GOM when variables are sampled from a conditional distribution.

4.3.1 FOS of Conditional Linkage Model. How GOM is applied using a conditional linkage model is still defined by a FOS, now indicated $\mathcal{F}^C$. However, each FOS element $\mathcal{F}_i^C$ now indicates which factors of C are resampled during one iteration of GOM. Applying GOM for a FOS element $\mathcal{F}_i^C$ to a solution $\boldsymbol{x}$ then amounts to sampling from the Bayesian factorization for each factor C_j with $j \in \mathcal{F}_i^C$, i.e.,

$$P\left(\boldsymbol{X}_{\mathcal{F}_i^C}\right) = \prod_{j \in \mathcal{F}_i^C} P\left(\boldsymbol{X}_{C_j} | \boldsymbol{X}_{\pi(C_j)}\right), \quad (8)$$

followed by the selection step of GOM. We consider three different options for the FOS $\mathcal{F}^C$ used in conditional linkage models.

First we consider the option where all variables are sampled during each iteration of GOM, i.e., $\mathcal{F}^C = \{\{1, \ldots, |C|\}\}$. The selection step of GOM is therefore only applied after all variables have been sampled. In this case RV-GOMEA reverts to a classic EDA. Because of the estimation and sampling used, it is similar to the Bayesian factorized version of AMaLGaM [5], but without learning the factorization structure each generation. As all variables are sampled during each iteration of GOM, partial evaluations do not benefit the optimization in this case. The benefit of a GBO setting is the fact that a Markov network can be constructed from the VIG. This option is referred to as Generational GOM (GG), as each solution in the population is subject to one iteration of GOM per generation.

In the second option we consider, we aim to benefit from partial evaluations by sampling only the variables for one factor C_i during each iteration of GOM, i.e., $\mathcal{F}^C = \{\{1\}, \ldots, \{|C|\}\}$. This option is referred to as Factorized GOM (FG). As a selection step is applied after sampling a subset of variables, optimization for strongly dependent problems may be hindered, because the selection step between these univariate variation operations prohibit moving a solution in a strongly correlated multi-dimensional cone of improvement.

Thirdly, we combine GG and FG to potentially combine their strengths. As such, GOM is first applied to each factor separately, followed by all factors jointly, i.e., $\mathcal{F}^C = \{\{1\}, \ldots, \{|C|\}, \{1, \ldots, |C|\}\}$. We refer to this option as Hybrid GOM (HG).

4.3.2 Application of GOM. A conditional linkage model is a combination of an underlying Bayesian factorization and a FOS $\mathcal{F}^C$ that describes how GOM is performed.

Algorithm 3 describes how GOM is applied to a solution $\boldsymbol{x}$ for a FOS element $\mathcal{F}_j^C$ of a conditional linkage model.

Algorithm 3 Conditional Gene-pool Optimal Mixing

procedure `GenepoolOptimalMixing`$(\boldsymbol{x}, \mathcal{F}_j^C, (\boldsymbol{V}, \boldsymbol{E}), \boldsymbol{O})$
  $\boldsymbol{b} \leftarrow \boldsymbol{x}_{\mathcal{F}_j^C}$
  for $u \in \mathcal{F}_j^C$ **do** ▷ In order $\boldsymbol{O}$
    $Y \leftarrow \{w \in \mathcal{I} : w \in \pi(C_u)\}$
    $\boldsymbol{x}_{C_u} \leftarrow P(\boldsymbol{X}_{C_u} \mid \boldsymbol{X}_Y = \boldsymbol{x}_Y)$ ▷ Eq. (7)
  $f_o \leftarrow$ `PartialEvaluation`$(\boldsymbol{x}, f_{\boldsymbol{x}}, \mathcal{F}_j^C)$
  if $f_o < f_{\boldsymbol{x}}$ **or** $\mathcal{U}(0,1) < p^{\text{accept}}$ **then** $f_{\boldsymbol{x}} \leftarrow f_o$
  else $\boldsymbol{x}_{\mathcal{F}_j^C} \leftarrow \boldsymbol{b}$

In Figure 4 we repeat the experiments shown in Figure 1 for a number of introduced conditional linkage models. This shows that UCond-FG and UCond-GG perform better than their non-conditional counterparts, i.e., univariate and full, respectively. Furthermore, the hybrid model UCond-HG achieves the best performance on strongly dependent problems, but also benefits from the

univariate steps on weakly dependent problems. The MCond models are not shown here, as they are almost identical to the UCond models for this problem.

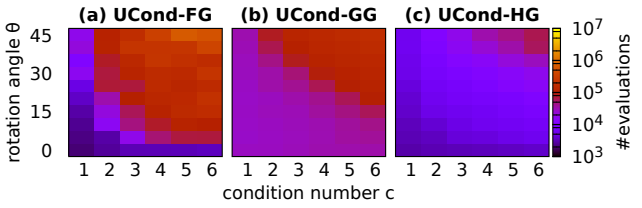

Figure 4: Median (30 runs) number of evaluations (color-coded) required to find the optimum with a precision of 10^{-10} of the 20-D $f^{REBChain}(\boldsymbol{x}, c, \theta)$ for RV-GOMEA using different conditional linkage models.

5 EXPERIMENTS

In this section, we analyze the performance of RV-GOMEA with conditional linkage models, and compare it to non-conditional linkage models. We only consider benchmark problems that are non-separable, because disjoint sets of variables in the VIG are independent, and can therefore be reduced to multiple non-separable problems that can be solved independently if g (see Equation (1)) is the identity function. All experiments are performed in a GBO setting that allows for partial evaluations.

We compare the univariate (Uni), full, UCond-GG, UCond-FG, UCond-HG, and MCond-HG linkage models. As the MCond-GG model is essentially identical to the UCond-GG model for our benchmark problems, and the MCond-FG was never found to be more effective than the UCond-FG, no results are shown for MCond.

We furthermore compare the performance of RV-GOMEA to VkD-CMA [2, 3]. VkD-CMA is among the current state of the art in continuous optimization, and is also capable of exploiting the fact that the number of dependencies is less than all possible dependencies, albeit through a means that is incompatible with exploiting partial evaluations.

5.1 Benchmark Problems

We firstly consider the Rosenbrock function, defined as:

$$f^{Rosenbrock}(\boldsymbol{x}) = \sum_{i=1}^{\ell-1} \left[100(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i^2)^2 + (1 - \boldsymbol{x}_i)^2\right]. \tag{9}$$

Each remaining benchmark problem we consider is constructed using rotated ellipsoid sub-functions, as defined in Equation (10). For a large condition number and rotation angle, the ellipsoid function is strongly dependent and cannot be solved efficiently using a linkage model that does not model the variables of the ellipsoid as jointly dependent. Furthermore, using rotated ellipsoid sub-functions allows us to analyze benchmark problems with different overlapping dependency structures.

Secondly, we introduce the *Rotated Ellipsoid Blocks* (REB) problem in Equation (11), defined as the sum of a number of rotated ellipsoid functions (Equation (10)) of size k with condition number 10^c, rotation angle θ, and stride $1 \leq s \leq k$. The REB function is a generalization of the Sum of Rotated Ellipsoid Blocks (SoREB) function [10]. The function $R^\theta(\boldsymbol{x})$ applies a counter-clockwise rotation to $\boldsymbol{x}$ of θ degrees around the origin for each pair of variables. The stride s determines the number of overlapping variables, as the starting indices of consecutive ellipsoid blocks are s apart. For $s = k$, each ellipsoid block is disjoint and therefore independent.

$$f^{Ellipsoid}(\boldsymbol{x}, c) = \sum_{i=1}^{|\boldsymbol{x}|} 10^{c(i-1)/(|\boldsymbol{x}|-1)} \boldsymbol{x}_i^2. \tag{10}$$

$$f^{REB}(\boldsymbol{x}, c, \theta, k, s) = \sum_{i=0}^{\frac{|\boldsymbol{x}|-k}{s}} f^{Ellipsoid}(R^\theta(\boldsymbol{x}[1+is : k+is]), c). \tag{11}$$

To have more flexibility in testing problems with overlapping dependencies, we define the *EllipsoidGraph* problem in Equation (12), of which the dependency structure is defined by the graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$. For each vertex $v \in \boldsymbol{V}$, a rotated ellipsoid of v and its neighboring vertices $N(v)$ is added, i.e.,

$$f^{REBGraph}(\boldsymbol{x}, c, \theta, (\boldsymbol{V}, \boldsymbol{E})) = \sum_{v \in \boldsymbol{V}} f^{Ellipsoid}(R^\theta(\{v\} \cup N(v)), c). \tag{12}$$

For convenience, we define the following functions:

$$f^{REBChainWeak}(\boldsymbol{x}) = f^{REB}(\boldsymbol{x}, c = 1, \theta = 5, k = 5, s = 4), \tag{13}$$

$$f^{REBChainStrong}(\boldsymbol{x}) = f^{REB}(\boldsymbol{x}, c = 6, \theta = 45, k = 5, s = 4), \tag{14}$$

$$f^{REB5SmallOverlap}(\boldsymbol{x}) = f^{REB}(\boldsymbol{x}, c = 6, \theta = 45, k = 5, s = 4), \tag{15}$$

$$f^{REB5LargeOverlap}(\boldsymbol{x}) = f^{REB}(\boldsymbol{x}, c = 6, \theta = 45, k = 5, s = 1), \tag{16}$$

$$f^{REBGrid}(\boldsymbol{x}) = f^{REBGraph}(\boldsymbol{x}, c = 6, \theta = 45, \boldsymbol{G}^{\mathtt{Grid}}), \tag{17}$$

$$f^{REBTorus}(\boldsymbol{x}) = f^{REBGraph}(\boldsymbol{x}, c = 6, \theta = 45, \boldsymbol{G}^{\mathtt{Torus}}), \tag{18}$$

$$f^{REBCube}(\boldsymbol{x}) = f^{REBGraph}(\boldsymbol{x}, c = 6, \theta = 45, \boldsymbol{G}^{\mathtt{Cube}}), \tag{19}$$

with $\boldsymbol{G}^{\mathtt{Grid}}$ a graph where the vertices are arranged into a $\sqrt{\ell} \times \sqrt{\ell}$ square grid, and an edge is added between each horizontally or vertically neighboring pair of vertices, with no wrap-around. The graph $\boldsymbol{G}^{\mathtt{Grid}}$ is therefore a planar graph with $2(\ell - \sqrt{\ell})$ edges, and the degree of each vertex is at least 2 and at most 4.

The graph $\boldsymbol{G}^{\mathtt{Torus}}$ is identical to $\boldsymbol{G}^{\mathtt{Grid}}$, but has wrap-around at the edges of the square grid. Therefore, the $\boldsymbol{G}^{\mathtt{Torus}}$ has 2ℓ edges and the degree of each edge is 4. We only consider the REBGrid and REBTorus problems for a square number of variables.

For the graph $\boldsymbol{G}^{\mathtt{Cube}}$, the vertices are arranged into a $\sqrt[3]{\ell} \times \sqrt[3]{\ell} \times \sqrt[3]{\ell}$ simple cubic lattice. An edge is added between pairs of vertices neighboring in the x, y, or z-direction, with no wrap-around. The total number of edges is therefore $3(\ell - \ell^{2/3})$, and each vertex has a degree of at least 3 and at most 6. We only consider the REBCube problem for a cubic number of variables.

5.2 Experimental Set-up

In each run, a budget of 10^7 function evaluations was used, and a time limit of 3 hours. In case of premature convergence before exceeding the budget, a run is restarted using the same population size. An implementation of RV-GOMEA [10] in C++, and an implementation of VkD-CMA [2, 3] in Python were used. Unless stated otherwise, default parameters were used for these algorithms. Source code for these algorithms is available on the web pages of the respective authors. The additions to RV-GOMEA introduced in this paper were also implemented in C++. Source code is publicly available on the web page of the last author[1].

[1] https://homepages.cwi.nl/~bosman/

5.3 Optimal Population Size

We first determine how the optimal population size scales with the increase of problem dimensionality for these linkage models, as this can be seen as a clear indicator of how well the linkage model matches the problem structure. For this purpose, we report the median population size of a number of bisections (1 for VkD-CMA, 5 for RV-GOMEA, due to substantial differences in required computation time) in Figure 5, with each data point in the bisection the median of 30 optimization runs. Estimated optimal population size for larger dimensions (dashed lines) were linearly extrapolated (on a log-log scale) with non-decreasing slope, as these populations were used to obtain results shown in Figure 6. Running bisections for all problem dimensionalities would require an exceptionally large amount of computation time.

5.4 Scalability

In Figure 6 we show the scalability of the number of function evaluations needed by RV-GOMEA with conditional linkage models, RV-GOMEA with non-conditional linkage models, and VkD-CMA. The population size is set to the value found through bisections or extrapolation, displayed in Figure 5.

In Figure 6 we generally observe the same trends as in Figure 5. The UCond-HG model is found to achieve the best performance on all problems with strong dependencies, except on the REB5SmallOverlap problem, where it performs slightly worse than the UCond-GG model, but seems to have better scalability. Furthermore, on the problems with weak overlapping dependencies, the UCond-HG was only a constant factor worse than the univariate models, which obtained the best performance.

Interestingly, the Uni and UCond-FG linkage models performed fairly well on the REB5LargeOverlap, REBGrid, REBTorus and REBCube problems, despite these problems consisting of 3-D to 6-D rotated ellipsoids with high condition number and rotation angle, indicating that the optimization problem becomes easier when many dependencies overlap. This is caused by the fact that the problem landscape of a subset of variables Y becomes less ill-conditioned when overlapping additive rotated ellipsoid sub-functions exist.

Furthermore, though the Uni and UCond-FG linkage models required over 10^6 function evaluations for even small dimensionalities of the REB5SmallOverlap function, this remained roughly constant for larger problem dimensionalities. This is caused by the fact that the sub-functions in this problem do not overlap much and are each relatively difficult to solve for univariate linkage models. Problem difficulty does however not increase with dimensionality if more such sub-functions are added, due to GOM.

6 DISCUSSION

Based on the results discussed in Sections 5.3 and 5.4, we argue that conditional linkage models have a clear benefit over non-conditional linkage models in RV-GOMEA. In particular, we find that using a conditional linkage model with Hybrid GOM (HG) achieves the best or close to the best performance on all problems, while also achieving the best scalability on all problems. The UCond-HG variant seems to be the best of the two conditional variants, as it was only slightly worse on the REB5SmallOverlap problem, and better on all other problems.

We find that VkD-CMA generally requires a smaller population size than RV-GOMEA, and performed better than the variants of RV-GOMEA that do not benefit from partial evaluations (full and UCond-GG). This is likely a result of the fact that variants of CMA-ES are known to obtain excellent performance on problems with quadratic surfaces, such as the benchmark problems considered here. Though RV-GOMEA here used a sampling model based on AMaLGaM [5], as it was originally introduced [10], it is possible to incorporate a sampling model based on CMA-ES into RV-GOMEA, potentially decreasing the required population size, and improving performance. This is left to future work.

In this work, we have only considered artificial benchmark problems, in particular benchmark problems constructed using rotated ellipsoids. In these problems, the dependency strength is equally strong for each rotated ellipsoid, though real-world problems may have varying degrees of dependency for different groups of variables, even if strictly speaking the problem is completely jointly dependent. For example, this is the case for HDR brachytherapy treatment planning for prostate cancer [24], where each variable is associated with a physical location, and the dependency strength of two variables correlates to the distance between their respective physical locations. To deal with such problems, it may be necessary to limit the linkage model to only the strongest dependencies, for example by removing edges corresponding to weak dependencies in the VIG. This does however require a notion of dependency strength with associated cut-off value, which we currently lack.

In this work, we limited our study to a GBO setting, because this is where RV-GOMEA mainly excels. The scalability of the UCond-HG linkage model will become worse in a BBO setting, because the univariate operations of GOM cannot benefit from partial evaluations. The UCond-GG linkage model does however not benefit from partial evaluations, and only benefits from the fact that the VIG is known a-priori, which would need to be learned online in a BBO setting. In contrast, the performance of VkD-CMA is identical in the GBO and BBO settings.

7 CONCLUSIONS

In this paper, we aimed to improve the performance of RV-GOMEA on problems with strong overlapping dependencies in a GBO setting that allows for partial evaluations. For this purpose, various new sorts of conditional linkage models capable of capturing conditional dependencies were introduced to be combined with RV-GOMEA.

The efficiency of RV-GOMEA with these new conditional models was compared to when non-conditional linkage models are used, and to the efficiency of VkD-CMA, which is considered to be among the state of the art for continuous optimization with EAs.

To compare performance, a number of benchmark problems with overlapping dependencies were were constructed to have different dependency structures and different dependency strength.

We observed that the UCond-HG linkage model with RV-GOMEA obtained the best overall performance, and scaled better than or equally good as VkD-CMA on all considered benchmark problems. Note that all benchmark problems were considered in a GBO setting that allows for partial evaluations, from which RV-GOMEA can benefit, unlike VkD-CMA. This GBO setting has previously

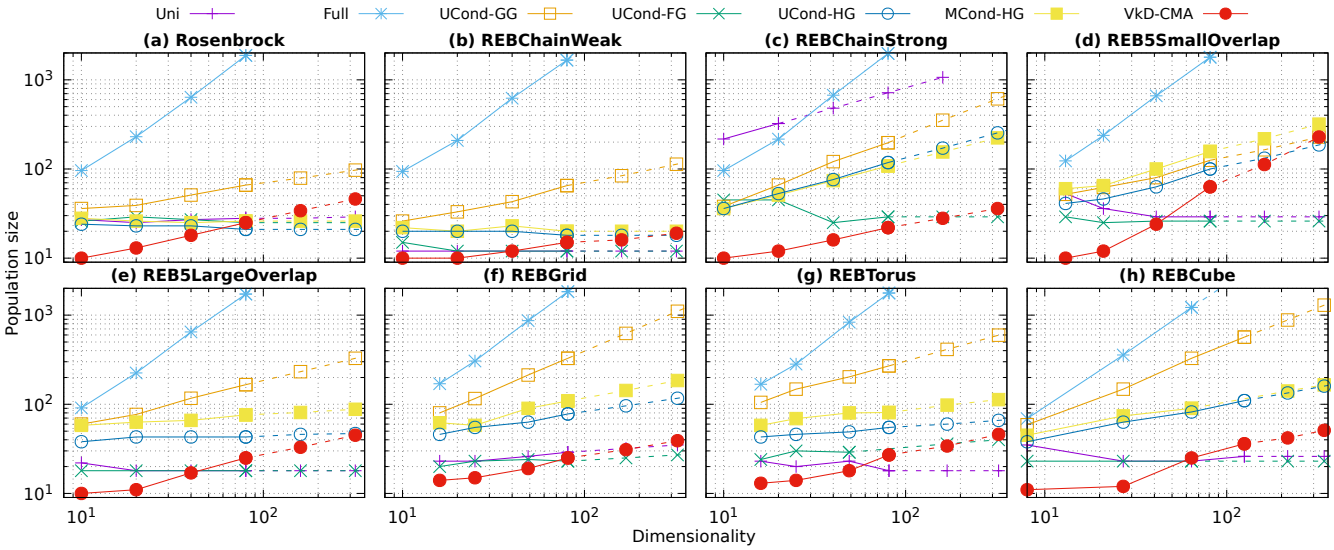

Figure 5: Estimated optimal population size for various linkage models for various benchmark problems, determined as median population size of a number (5 for RV-GOMEA, 1 for VkD-CMA) bisections, with each data point of a bisection the median number of evaluations of 30 runs.

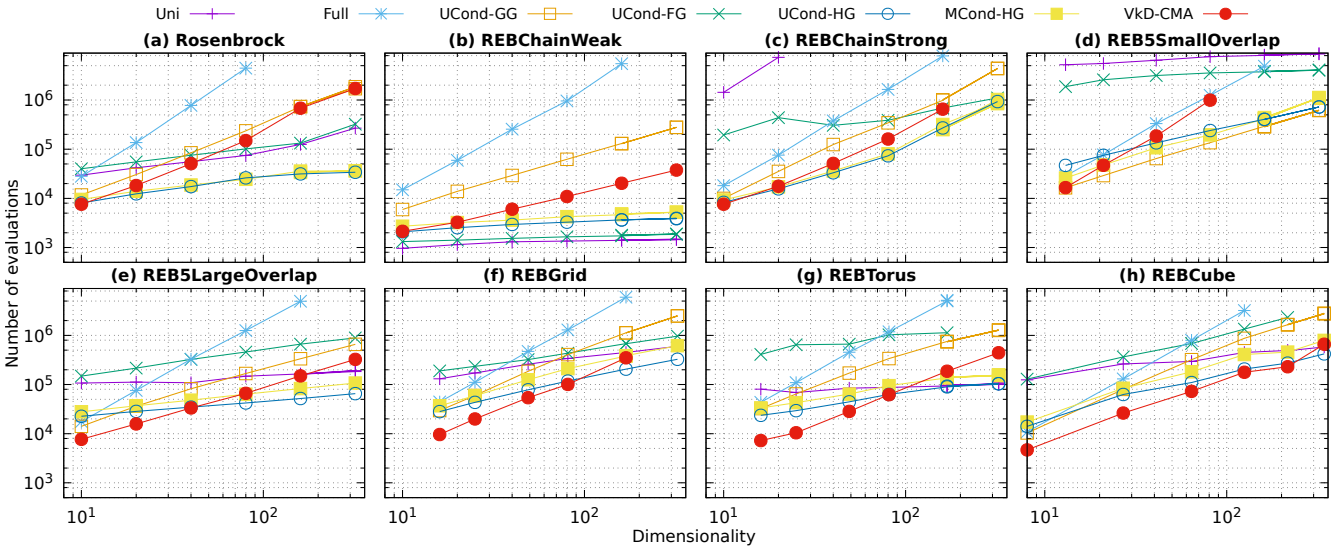

Figure 6: Median (30 runs) number of evaluations for various linkage models for various benchmark problems in a GBO setting using the population size shown in Figure 5.

appeared in various real-world problems [9, 24], which also have an overlapping dependency structure.

Furthermore, the UCond-HG linkage model was able to match the scalability of univariate linkage models on problems with weak overlapping dependencies, and that of full linkage models on problems with strong overlapping dependencies. For two benchmark problems, the UCond-HG linkage model scaled better than all other considered linkage models.

We conclude that the introduction of conditional linkage models can greatly benefit the performance of RV-GOMEA on problems with overlapping dependencies in a GBO setting. These results are promising and may benefit the optimization of real-world problems with similar dependency structures [9, 24] in future work.

8 ACKNOWLEDGMENTS

This work is part of the research programme IPPSI-TA with project number 628.006.003, which is financed by the Dutch Research Council (NWO) and Elekta. This work was partially carried out on the Dutch national e-infrastructure with the support of SURF Cooperative.

REFERENCES

[1] C.W. Ahn, R.S. Ramakrishna, and D.E. Goldberg. 2004. Real-coded Bayesian optimization algorithm: Bringing the strength of BOA into the continuous world. In *Genetic and Evolutionary Computation Conference*. Springer, 840–851.

[2] Y. Akimoto and N. Hansen. 2016. Online model selection for restricted covariance matrix adaptation. In *International Conference on Parallel Problem Solving from Nature*. Springer, 3–13.

[3] Y. Akimoto and N. Hansen. 2016. Projection-based restricted covariance matrix adaptation for high dimension. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. ACM, 197–204.

[4] P.A.N. Bosman. 2009. On empirical memory design, faster selection of Bayesian factorizations and parameter-free gaussian EDAs. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*. 389–396.

[5] P.A.N. Bosman, J. Grahl, and D. Thierens. 2013. Benchmarking parameter-free AMaLGaM on functions with and without noise. *Evolutionary Computation* 21, 3 (2013), 445–469.

[6] P.A.N. Bosman and D. Thierens. 1999. Linkage information processing in distribution estimation algorithms. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 1*. Morgan Kaufmann Publishers Inc., 60–67.

[7] P.A.N. Bosman and D. Thierens. 2000. Expanding from Discrete to Continuous Estimation of Distribution Algorithms: The IDEA. In *International Conference on Parallel Problem Solving from Nature*. Springer, 767–776.

[8] A. Bouter, T. Alderliesten, A. Bel, C. Witteveen, and P.A.N. Bosman. 2018. Large-scale parallelization of partial evaluations in evolutionary algorithms for real-world problems. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 1199–1206.

[9] A. Bouter, T. Alderliesten, and P.A.N. Bosman. 2017. A novel model-based evolutionary algorithm for multi-objective deformable image registration with content mismatch and large deformations: benchmarking efficiency and quality. *Proc. SPIE* 10133, 1013312.

[10] A. Bouter, T. Alderliesten, C. Witteveen, and P.A.N. Bosman. 2017. 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*. ACM, 705–712.

[11] A. Bouter, N.H. Luong, C. Witteveen, T. Alderliesten, and P.A.N. Bosman. 2017. The multi-objective real-valued gene-pool optimal mixing evolutionary algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 537–544.

[12] J.S. De Bonet, C.L. Isbell Jr, and P.A. Viola. 1997. MIMIC: Finding optima by estimating probability densities. In *Advances in Neural Information Processing Systems*. 424–430.

[13] O.Q. de Carvalho, R. Tinós, D. Whitley, and D.S. Sanches. 2019. A New Method for Identification of Recombining Components in the Generalized Partition Crossover. In *2019 8th Brazilian Conference on Intelligent Systems (BRACIS)*. IEEE, 36–41.

[14] K. Deb and C. Myburgh. 2016. Breaking the billion-variable barrier in real-world optimization using a customized evolutionary algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. ACM, 653–660.

[15] M.L. Eaton. 1983. Multivariate statistics: a vector space approach. *John Wiley & Sons, Inc.* (1983).

[16] C. Echegoyen, J.A. Lozano, R. Santana, and P. Larranaga. 2007. Exact Bayesian network learning in estimation of distribution algorithms. In *2007 IEEE Congress on Evolutionary Computation*. IEEE, 1051–1058.

[17] D. Geiger, T. Verma, and J. Pearl. 1990. Identifying independence in Bayesian networks. *Networks* 20, 5 (1990), 507–534.

[18] I. Gronau and S. Moran. 2007. Optimal implementations of UPGMA and other common clustering algorithms. *Information Processing Letters* 104, 6 (2007), 205–210.

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

[20] G.R. Harik. 1999. Linkage learning via probabilistic modeling in the ECGA. *IlliGAL report* 99010 (1999).

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

[22] M. Hauschild and M. Pelikan. 2011. An introduction and survey of estimation of distribution algorithms. *Swarm and Evolutionary Computation* 1, 3 (2011), 111–128.

[23] H. Karshenas, R. Santana, C. Bielza, and P. Larrañaga. 2012. Continuous estimation of distribution algorithms based on factorized Gaussian Markov networks. In *Markov Networks in Evolutionary Computation*. Springer, 157–173.

[24] N.H. Luong, T. Alderliesten, A. Bel, Y. Niatsetski, and P.A.N. Bosman. 2018. Application and benchmarking of Multi-Objective Evolutionary Algorithms on High-Dose-Rate brachytherapy planning for prostate cancer treatment. *Swarm and Evolutionary Computation* 40 (2018), 37–52.

[25] M.N. Omidvar, X. Li, Y. Mei, and X. Yao. 2013. Cooperative co-evolution with differential grouping for large scale optimization. *IEEE Transactions on Evolutionary Computation* 18, 3 (2013), 378–393.

[26] M. Pelikan. 2005. Hierarchical Bayesian optimization algorithm. In *Hierarchical Bayesian Optimization Algorithm*. Springer, 105–129.

[27] M. Pelikan, D.E. Goldberg, and E. Cantú-Paz. 1999. BOA: The Bayesian optimization algorithm. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 1*. Morgan Kaufmann Publishers Inc., 525–532.

[28] M. Pelikan and H. Mühlenbein. 1998. Marginal distributions in evolutionary algorithms. In *Proceedings of the International Conference on Genetic Algorithms Mendel*, Vol. 98. 90–95.

[29] R. Santana. 2005. Estimation of distribution algorithms with Kikuchi approximations. *Evolutionary Computation* 13, 1 (2005), 67–97.

[30] S. Shakya, A. Brownlee, J. McCall, F. Fournier, and G. Owusu. 2010. DEUM–A Fully Multivariate EDA Based on Markov Networks. In *Exploitation of Linkage Learning in Evolutionary Algorithms*. Springer, 71–93.

[31] S. Shakya and J. McCall. 2007. Optimization by estimation of distribution with DEUM framework based on Markov random fields. *International Journal of Automation and Computing* 4, 3 (2007), 262–272.

[32] S. Shakya and R. Santana. 2008. An EDA based on local Markov property and Gibbs sampling. In *Proceedings of the 10th annual Conference on Genetic and Evolutionary Computation*. ACM, 475–476.

[33] S. Shakya and R. Santana. 2012. *Markov Networks in Evolutionary Computation*. Springer.

[34] F. Spitzer. 1971. Markov random fields and Gibbs ensembles. *The American Mathematical Monthly* 78, 2 (1971), 142–154.

[35] Y. Sun, M. Kirley, and S.K. Halgamuge. 2017. A recursive decomposition method for large scale continuous optimization. *IEEE Transactions on Evolutionary Computation* 22, 5 (2017), 647–661.

[36] D. Thierens and P. A. N. Bosman. 2011. Optimal mixing evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 617–624.

[37] R. Tintos, D. Whitley, and F. Chicano. 2015. Partition crossover for pseudo-boolean optimization. In *Proceedings of the 2015 ACM Conference on Foundations of Genetic Algorithms XIII*. ACM, 137–149.