Solving multi-structured problems by introducing linkage kernels into GOMEA

Guijt, Arthur; Thierens, Dirk; Alderliesten, Tanja; Bosman, Peter A.N.

DOI
10.1145/3512290.3528828

Publication date
2022

Document Version
Final published version

Published in
GECCO 2022 - Proceedings of the 2022 Genetic and Evolutionary Computation Conference

Solving Multi-Structured Problems by Introducing Linkage Kernels into GOMEA

Arthur Guijt
Arthur.Guijt@cwi.nl
Centrum Wiskunde & Informatica
Amsterdam, The Netherlands

Dirk Thierens
D.Thierens@uu.nl
Utrecht University
Utrecht, The Netherlands

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

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

ABSTRACT

Model-Based Evolutionary Algorithms (MBEAs) can be highly scalable by virtue of linkage (or variable interaction) learning. This requires, however, that the linkage model can capture the exploitable structure of a problem. Usually, a single type of linkage structure is attempted to be captured using models such as a linkage tree. However, in practice, problems may exhibit multiple linkage structures. This is for instance the case in multi-objective optimization when the objectives have different linkage structures. This cannot be modelled sufficiently well when using linkage models that aim at capturing a single type of linkage structure, deteriorating the advantages brought by MBEAs. Therefore, here, we introduce linkage kernels, whereby a linkage structure is learned for each solution over its local neighborhood. We implement linkage kernels into the MBEA known as GOMEA that was previously found to be highly scalable when solving various problems. We further introduce a novel benchmark function called Best-of-Traps (BoT) that has an adjustable degree of different linkage structures. On both BoT and a worst-case scenario-based variant of the well-known MaxCut problem, we experimentally find a vast performance improvement of linkage-kernel GOMEA over GOMEA with a single linkage tree as well as the MBEA known as DSMGA-II.

CCS CONCEPTS

• **Computing methodologies** → **Discrete space search**; **Randomized search**; • **Theory of computation** → **Evolutionary algorithms**.

KEYWORDS

Evolutionary Algorithms, Linkage Learning, Kernels, Local Neighborhood

GECCO '22, July 9–13, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9237-2/22/07.
https://doi.org/10.1145/3512290.3528828

ACM Reference Format:
Arthur Guijt, Dirk Thierens, Tanja Alderliesten, and Peter A.N. Bosman. 2022. Solving Multi-Structured Problems by Introducing Linkage Kernels into GOMEA. In *Genetic and Evolutionary Computation Conference (GECCO '22), July 9–13, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3512290.3528828

1 INTRODUCTION

Recombination is most efficient when variables with a strong relationship, i.e., linkage, are recombined jointly [25, 27]. Compared to traditional Evolutionary Algorithms (EAs), Model-based EAs (MBEAs) aim to be significantly more scalable and reliable in solving problems that exhibit a certain type of exploitable problem structure by explicitly modelling aspects of this structure and using this when creating new solutions. Such models may be built based on problem-specific insights, if they are available, or, in case of a black-box optimization scenario, be learned during optimization based on previously performed function evaluations.

In this paper, we consider in particular MBEAs aimed at exploiting information about dependencies between problem variables. Various types of models exist in literature. Models capable of describing complex relationships, such as Bayesian networks [18], are however expensive to learn because they capture information about dependencies and explicitly estimate associated probability distributions, making optimization particularly inefficient if function evaluations themselves are not very expensive. Especially in such cases, computationally cheaper alternatives tend to be preferred and have become part of state-of-the-art algorithms in recent years. Examples thereof include the linkage tree used in the Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) [5, 26] and the Parameterless Population Pyramid (P3) [7], as the incremental linkage set used in Dependency Structure Matrix Genetic Algorithm (DSMGA-II) [10], in which only interactions between variables are modelled explicitly.

Despite many advances in recent years, especially when it comes to benchmark problems, most MBEA literature only consider problems in which a single linkage structure is clearly present. However, in practice, it is well possible a problem exhibits multiple linkage structures (in different parts of the search space). For instance, this can easily occur in Multi-Objective (MO) optimization, i.e., because the different objectives may have different linkage structures.

In early multi-objective MBEAs, as well as the more recent MO-GOMEA, objective-space clustering is used [14, 20]. While not originally introduced to capture different linkage structures along the front, the underlying rationale is similar: what is important in one extreme region of the Pareto front, is likely not important in another extreme region. I.e., solutions that maximize one objective may look very different from solutions that maximize another. Employing clustering and restricted mating therefore increases the chances of successful (local) variation. Similarly, clustering also offers a first remedy to multiple linkage structures, especially if linkage is learned in each cluster separately. However, the number of clusters is typically taken to be relatively small, because this has been observed to suffice to achieve effective optimization if solutions differ along the Pareto front, but the linkage information does not, as in the trap-inverse trap problem [14, 20] . However, if linkage information gradually changes along the front as well, which is to be expected if both objectives have *different* linkage structures, coarse-grained clusters may not suffice. However, this has, to the best of our knowledge, not been studied in detail.

The issue of multiple linkage structures in a single problem is not restricted to MO optimization. Similar issues can occur in a single-objective setting. For instance, single-objective optimization problems can be (highly) multi-modal, and each mode can have its own linkage structure. Such problems too, have, to the best of our knowledge, never been studied in detail (in the context of MBEAs). That is not to say that multi-modal problems have never been tackled before with EAs. Commonly, niching and mating restrictions are used here. For example, in [1] an estimation-of-distribution algorithm (EDA) is adaptively clustered and niched to optimize both a real-valued and discrete problem, learning multiple models over a population, showing improved performance over a conventional EDA. Similarly, including niching into CMA-ES was investigated in [21], which reports improvements on some multi-modal functions. Additionally, in [12] the applicability of niching on a real-valued multi-modal multi-objective problem is investigated, showing that to obtain good performance on such problems, diversity in both objective space and parameter space is necessary. Still, partially due to the use of particular benchmark problems, none of these studies address the explicit presence of different variables interactions *per niche* and the additional requirements this may bring to bear on model building in EAs, which is what we consider in this paper.

We name these problems with multiple linkage structures, *multi-structured problems*. In the context of MBEAs, one study considers a closely related phenomenon. Specifically, in [16] it is shown that multi-modality can be an issue for use of pairwise linkage, as commonly used in DSMGA-II and GOMEA. To resolve this issue, higher-order linkage was learned directly rather than through pairwise combinations. In practice, this can be significantly more costly than pairwise linkage; the cost of learning higher-order blocks grows exponentially with the order required. Furthermore, this approach and alternative linkage measures (e.g. [2, 19]), suffer from being unable to represent higher order varying linkage, as they generally build only a single linkage tree.

In this work, we propose a more scalable approach that is aimed at solving the more general notion of multi-structured problems, by introducing *linkage kernels*. Rather than learning a single linkage model, every solution learns their own linkage model over their own local neighborhood. In addition, to show the impact that the presence of multiple linkage structures has on both existing MBEAs and our newly proposed linkage kernels, we introduce a novel, scalable benchmark function called Best-of-Traps (BoT) that can be used both single- and multi-objectively. BoT is based on the well-known deceptive trap functions [3] in which the number of linkage structures and the size of key building blocks are parameterized.

The remainder of this work is organized as follows. In Section 2 we describe the recent variant of GOMEA, the approach that LK-GOMEA extends. In Section 3, the extensions that make up LK-GOMEA are described. In Section 4 the benchmark problems used in this work, Best-of-Traps and MaxCut, are described. Subsection 4.3 describes relevant aspects related to MO optimization, as both single- and MO experiments are performed. Section 5 contains the experimental setup related to the single-objective experiments, and the corresponding results, in which DSMGA-II, GOMEA and LK-GOMEA are compared. Section 6 similarly contains the experimental setup and results for the MO experiments. Finally, we end with the discussion in Section 7 and the conclusion in Section 8.

2 GOMEA & MO-GOMEA

We will focus on MBEAs that employ both characteristics of local search and EDAs. Of particular interest are GOMEA [5], DSMGA-II [10], and P3 [7], which have shown to perform well on a wide variety of problems and may be considered state of the art among (discrete) MBEAs. Given that GOMEA was previously also applied to MO problems, in this work we will extend upon GOMEA.

2.1 GOMEA

GOMEA, originally introduced in [26], is an MBEA that utilizes a Family Of Subsets (FOS) to describe (strong) dependencies between variables in terms of subsets of variables. These subsets are utilized in a recombination scheme named Gene-pool Optimal Mixing (GOM) which is applied to a copy of each solution in the population. This scheme iterates over the FOS elements in a random order, replacing the values in the solution at the problem variables corresponding to the FOS element with those of a random donor from the population. This change is immediately evaluated. If the change yields a worse fitness, it is reverted, otherwise it is accepted. If a solution has not been improved after a certain number of generations, forced improvements are applied: GOM is performed with the current elitist as donor until a single strict improvement is found. If no such improvement can be found for any FOS element, the solution is replaced with the current elitist solution.

2.2 MO-GOMEA

In MO-GOMEA [14], the multi-objective variant of GOMEA, an elitist archive is added to keep track of the current approximation front. Further, an overlapping variant of k-means-like clustering is applied, based on the distances in the objective space. For each cluster, a linkage tree is learned. Any cluster that has the largest mean value for objective i is considered to be a single-objective cluster. For such a cluster, single-objective GOM is used with respect to objective i. For the remaining clusters, GOM is still used, but a change is regarded an improvement if it is weakly Pareto-dominant over the previous state, or if it can be added to the elitist archive.

2.3 Enhancements

Recently, some performance enhancements were proposed for GOMEA [5]. Not all changes are applicable to MO-GOMEA, however. We therefore only employ a subset of the changes here.

Originally, a single linkage tree (LT) is learned from the population in GOMEA, and a separate LT is learned from each cluster in MO-GOMEA. Each LT is learned by first computing Normalized Mutual Information (NMI) between all pairs of variables, as opposed to MI in the original GOMEA version [26]. NMI was observed to lead to slightly better performance on a range of problems. The pairwise NMI is subsequently used in the UPGMA hierarchical clustering algorithm to construct the linkage tree. UPGMA iteratively merges sets of variables, starting from all univariate sets with one variable, until all sets are merged into one set containing all variables. Each node in the resulting clustering tree originally represents a subset of variables in the FOS. As in the latest version of GOMEA [5], we employ filtering. This removes all elements for which $NMI \approx 0$ and the subtrees of each element for which $NMI \approx 1$.

Unlike the latest version of GOMEA [5], we do not include Local Search (LS). Unlike single-objective search, there are multiple possible acceptation criteria for multi-objective optimization. Finding the best way to perform LS is problem-specific, especially for multi-objective optimization, and is considered outside the scope of this article. Moreover, LS can be added to our newly proposed LK-GOMEA in the same way as to GOMEA.

Further, we use the original Interleaved Multi-start Scheme (IMS) instead of the population pyramid introduced in [7], in part to allow comparison with DSMGA-II in the same population sizing setup. The IMS [5] is an online population sizing scheme, inspired by [8], that can also be used to control parameters like the number of clusters in MO-GOMEA [13]. Multiple simultaneous populations of increasing size are used. For each b generations of a smaller population, the next larger population undergoes one generation. This pattern recurses, e.g., the larger population after that undergoes one generation for every b^2 generations of the smaller population. Smaller populations with an average fitness lower than that of a larger population are stopped. In GOMEA, commonly $b = 4$ is used.

Of special note is donor search, originally introduced in P3 [7]. Instead of using a random donor, a donor is sought for which the variables in the subset do not all have the same value as what is currently in the solution. The effect of this is most notable when the population is close to converging. As we utilize locality in this work, the neighborhoods are much more likely to share variable values, increasing the likelihood of donor search being beneficial.

Decomposition-based methods in multi-objective optimization are known to provide better spread of solutions along the approximation front, and to be better suited for higher-dimensional objective spaces [11, 28]. Therefore, a second version of MO-GOMEA was introduced that leverages scalarizations by means of which each solution is assigned an improvement direction [13]. This is especially relevant for our work, since linkage kernels are similarly meant to associate a notion of local linkage relations to each solution that are likely important to exploit in order to achieve improvements. Aligning these notions of locality is likely of high value.

3 LINKAGE-KERNEL GOMEA

3.1 A Problem of 2 Modes

As outlined in the previous Sections, GOMEA (in a black-box optimization setting) employs (normalized) mutual information to detect (pairwise) linkages. The idea is that if two variables are dependent, after selection, some variable value combinations will occur more frequently then others, which can be measured in terms of their mutual information. While this has its limitations, leading to novel dependency detection methods being introduced that do not depend on statistical information [22], on most single-structure problems, the structure can be efficiently detected. Even so, this changes completely, also for other dependency-detection methods (unless they do a full Walsh decomposition as in [4]), in case of a multi-modal function with different dependencies in each mode. Then, some correlations between variables are undetectable when only looking at a pair of variables. An example of this is given in Figure 1. In this example, we have two modes for which a pair of variables v_0 and v_1 are strongly correlated, although differently for each mode. Especially at the start of optimization, we can assume the population is equally divided over the modes. That means, that when looking at pairs within the combined population however, the variables will seem to be uncorrelated, i.e. pairwise independence actually occurs. Consequently, models learning linkage from this will not create subsets with these variables together, making it harder to optimize for any mode. The more important the linkage information to optimize each mode, the bigger the problem.

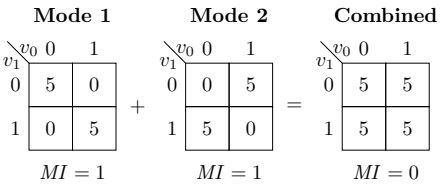

Figure 1: Count matrices and corresponding MI values within modes and within the entire search space, for two variables v_0 and v_1. High MI values for modes individually does not necessarily lead to high MI for the combination.

It is important to note that while this example concerns an example of only 2 variables, using higher-order linkage-learning alone would *not* resolve this issue. Firstly, if important linkages are completely different in each mode, but do overlap, the order of linkage learning required could be intractable from a classical single-population statistics point of view. Secondly, the values for high-quality solutions in each mode may very well be very different as well. When recombining during GOM, a subset is essentially sampled from the entire population. Even if the block is perfectly identified for one mode, poor exploitation cannot be avoided without also identifying the locality of the mode and employing restricted mating within a mode. Otherwise, inheriting from solutions from other modes will occur, which will likely still not be effective.

3.2 Splitting Modes

Based on the analysis above, it is clear that it is important to adeptly identify modes. If linkage learning works when the modes are separate, splitting up the population and learning separate linkage

models for each mode should solve the problem of multiple modes interfering. Luckily, similar to how linkage can be inferred using MI from a population that has experienced selection pressure, solutions belonging to the same mode tend to cluster together.

In and of itself, this concept is not new to EAs, as it is similar to diversity preserving measures, like mating restrictions and niching, that aim to distribute the population across different modes in the search space different parts of the search space. It has even previously be proposed to learn separate models for separate parts of the search space [1]. Similarly related are island topologies used in island model EAs. However, the best topology is known to vary from problem to problem [23, 24].

While these approaches offer the right direction to solve the outlined problem, there is an important catch: how well niching works depends fully on the quality of how the population is split. Including too many solutions from a different mode could have a large negative impact, for the same reasons as outlined in Subsection 3.1. Moreover, unlike the strict boundaries in the approaches above, linkage structures may be shared between different modes, or gradually change along the Pareto front in multi-objective optimization, necessitating many niches or fuzzy clustering boundaries. Finally, while a solution that has many variable values in common with another solution is likely part of the same mode, a solution with very dissimilar values is not necessarily part of another mode. Splitting the population, especially into a limited number of clusters, is hence a non-trivial problem.

3.3 Linkage Kernels

To account for this, we propose to associate a local neighborhood with each solution, from which both solution-specific linkage is learned and donors are sampled from when performing GOM. This enables many different types of linkage structure to be exploited throughout the population, with non-strict neighborhood boundaries, meeting some of the concerns raised above. We refer to this notion as *linkage kernels*.

In this paper, we use 2 definitions of neighborhood. Firstly, we define the neighborhood (or kernel) of a solution as its k-nearest neighbors (KNN) in the population, using Hamming distance as a distance measure, with ties in distance broken randomly. We refer to this neighborhood as the *asymmetric* neighborhood.

Using KNN as the neighborhood does come with a notable downside. When enough solutions converge to the same point, these solutions can only recombine with one another, and the niche pertaining to the linkage kernel becomes irreversibly converged. We note that KNN is asymmetric: while the converged solutions will only have each other as neighbor, a non-converged individual can still have one of the converged solutions in its neighborhood. Allowing recombination also with these solutions, would still allow for the introduction of new building blocks to the local neighborhood. Therefore, we secondly consider the *symmetric* KNN neighborhood, in which the neighborhood of a solution consists of the solutions of which it is a KNN, in addition to its own KNNs. Note that this means the neighborhood size is no longer exactly k for a solution, but can be larger, depending on the number of neighborhoods that solution occurs in.

What remains is the definition of the size of the neighborhood. This should neither be too big, to avoid including solutions from another mode, or too small, as linkage learning still requires a sufficiently large neighborhood to properly infer linkage. Partly based on preliminary experiments, we propose to use $k = \left\lceil \sqrt{|P|} \right\rceil$, where $|P|$ is the population size and k is the number of nearest neighbors. This provides a large enough pool to learn local linkage over, yet is small enough to be able to sustain niches. Furthermore, by making k relative to the population size, the IMS will make the size of the neighborhood (note: in terms of number of neighboring solutions in the population) relatively smaller as the population size increases: each doubling of the population size only increases k by a factor of $\sqrt{2}$ (ignoring rounding).

3.4 LK-GOMEA

With linkage kernels defined, we can now propose a novel version of GOMEA: Linkage-Kernel GOMEA (LK-GOMEA)[1]. In LK-GOMEA, the neighborhoods of each solution are first inferred, either using KNN or its symmetric variant, and then, an LT is learned for each solution based on this neighborhood. When improving a solution through GOM (or FI), the solution's FOS is used instead, and the donor pool is restricted to the neighborhood. Pseudocode can be found in the Supplementary material.

4 PROBLEMS

In this section we provide definitions of the problems we consider.

4.1 MaxCut & Worst-of-MaxCuts

The first problem is weighted MaxCut. In this problem, a graph $G = (V, E)$, with $|V| = \ell$ vertices and weighted edges $e = (v_i, v_j, w_{ij}) \in E$, where v_i and v_j are endpoints and w_{ij} is the corresponding weight, is given. The goal is to divide the vertices in two sets, such that the sum of weights corresponding to edges between the two sets is maximized. We encode a solution using a binary string, which determines for each vertex which of the two sets it is in. The objective value given a solution s is then:

$$f_{\text{maxcut}}(s) = \sum_{(v_i, v_j, w_{ij}) \in E} \begin{cases} w_{ij} & \text{if } s[v_i] \neq s[v_j] \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

MaxCut is a combinatorial optimization problem that GOMEA is known to solve well [5]. It is however a symmetric problem, swapping the sets (i.e., inverting the binary values of a solution) has no impact on the objective value. Semantically similar solutions are therefore not always close in Hamming distance, which could potentially be an issue. Furthermore, in general, weighted MaxCut is an NP-hard problem with many local optima. A single problem instance is used for each string length ℓ, and consists of a fully connected (dense) graph, with integer weights sampled from $[1, 5]$.

Although weighted MaxCut may have many local optima, the linkage structure of a single instance is still predominantly defined by the graph structure and the weights in a singular way. We therefore additionally consider a problem we call Worst-of-MaxCuts. Worst-of-MaxCuts is the worst-case scenario-based robust optimization variant of MaxCut. In real-world optimization, oftentimes

[1] Source code: https://github.com/8uurg/Multistructured-Problems-LK-GOMEA

there are uncertainties and multiple scenarios are then sampled in order to optimize for expected value or to hedge against the worst case. Worst-of-MaxCuts is representative of such problems. In Worst-of-MaxCuts, a set of problem instances is given that all have the same number of vertices. When evaluating a solution, the worst objective value out of the individual MaxCut objective values is returned. In this work, we combine two different instances, generated similarly as above.

4.2 Best-of-Traps

While weighted MaxCut and Worst-of-MaxCuts are interesting from a perspective of being a well-known problem and having a relation to real-world problems, to assess the impact of linkage kernels on multi-structured problems in a controlled way, a multi-structured benchmark function is needed. This benchmark function should be scalable, i.e., solvable for increasingly larger sizes, when structure is recognized properly. Furthermore, the degree to which a problem is multi-structured should be adjustable, as to be able to determine to what degree having multiple structures within a problem is an actual issue for scalable optimization.

The concatenated deceptive trap function [3] is an important problem in the linkage-learning community. Its deceptive nature makes the problem hard to solve in a scalable manner if the underlying structure, i.e., linkage between variables, is not recognized and accounted for [3, 27]. This makes it a much-used benchmark, used to assess if MBEAs are capable of recognizing higher-order problem structure. Yet, the issues raised around linkage learning have to do with a problem being multi-structured, and this is not the case for the deceptive trap function.

To this end, we here introduce a problem named Best-of-Traps (BoT), defined as the maximum over a number of the sub-problems. Each sub-problem is a *permuted* concatenated deceptive trap function in which we also redefine the unitation function to count not the number of ones, but the number of bits similar to a predefined string, so that the global and local attractor can be made different in the binary space as well (i.e., the optimum is not necessarily all ones for each sub-problem). By taking the maximum, and because each sub-function has the same optimal value, yet different binary encoding of the optimum, multiple modes are introduced. By changing the structure in each mode, multiple linkage structures are introduced that span the entire solution. Concordantly, this problem is multi-modal and has a controllable amount of linkage structure through the number of sub-problems used. This control will allow us to investigate to what extent and from what degree onward, multiple structures may pose an issue.

Each sub-problem in BoT is solvable individually by modern model-based EAs like GOMEA, DMSGA-II and P3. Hence, once a mode is localized, the problem should be effectively solvable. However, since we preserve all modes through the max function, the BoT function effectively contains all linkage structures of all sub-problems, creating situations as illustrated in Figure 1.

For string length ℓ, each BoT sub-problem consists of *fns* permuted trap functions t_a, $a \in \{0, 1, \ldots, fns - 1\}$, each with a predefined different optimal solution s_a^*, and permutation π_a, both of length ℓ. Given block size k, each sub-problem is defined to be:

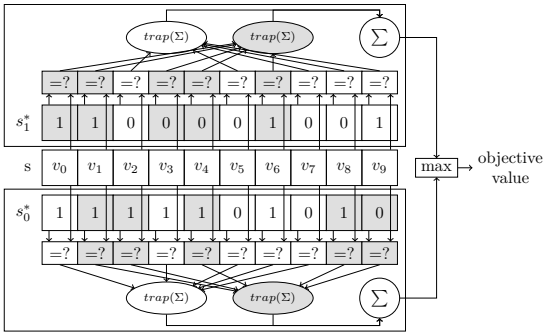

Figure 2: Illustration of Best-of-Traps evaluation for $\ell = 10$, $k = 5$, and $fns = 2$. Each sub-problem has their own optimum: s_0^* and s_1^*. As $\ell/k = 2$, each sub-problem has two blocks, indicated by white and gray. When a solution s is evaluated, the solution is compared with the optimum for each sub-problem. For each block (color), the number of matches is counted, over which the deceptive trap function is applied. The resulting values are summed together for each sub-problem. The resulting sums are finally combined through a max function.

$$t_a(s) = \sum_{i=0}^{\ell/k-1} T\left(\sum_{j=0}^{k-1} \begin{cases} 1 & \text{if } s[\pi_a[ik+j]] = s_x^*[\pi_a[ik+j]] \\ 0 & \text{otherwise} \end{cases}\right) \quad (2)$$

Where T is the deceptive trap function:

$$T(u) = \begin{cases} k & \text{if } u = k \\ k - u - 1 & \text{otherwise} \end{cases} \quad (3)$$

The BoT problem can now be defined as follows:

$$f_{\text{BoT}}(s) = \max_{a=0}^{fns-1} t_a(s) \quad (4)$$

Instances are generated by shuffling indices $(0, 1, \ldots, \ell - 1)$ to construct π_a, and uniformly randomly sampling a binary string s_a^* of length ℓ for each permuted trap function t_a, $a \in \{0, 1, \ldots, fns-1\}$. An example with $fns = 2$, $k = 5$ and $\ell = 10$ is given in Figure 2.

4.3 Multi-Objective Problems

We will furthermore consider multi-objective optimization problems in which the linkage structure gradually changes along the front by virtue of the individual objectives having different linkage structures. Moreover, we will consider situations in which the individual objectives themselves also have multiple linkage structures. Specifically, we propose to perform experiments with BoT vs BoT, BoT vs MaxCut and MaxCut vs MaxCut. For each problem, a different instance is used for each objective.

We will focus especially on the case in which both objectives are BoT. This configuration is interesting due to the combination of both large discrete changes in linkage within one objective, and the gradual change from one objective to another. Of particular interest is how difficult the resulting problem will be, and how well the kernel approach will adapt accordingly.

As these problems are non-trivial to solve, we do not know the optimum (i.e., the Pareto front). To still be able to obtain high-quality fronts for reference, we split up the BoT problem into its sub-problems. We then solved the MO problem for each of pair

of sub-problems individually and combined the resulting fronts afterwards. For more details, see the Supplementary material.

5 SINGLE-OBJECTIVE EXPERIMENTS

5.1 Experimental Setup

We consider Best-of-Traps (BoT), MaxCut and Worst-of-MaxCuts with problem sizes $\ell \in \{10, 20, 40, 80, 160, 320\}$ for BoT, and $\ell \in \{6, 12, 25, 50, 100, 200\}$ for MaxCut and Worst-of-MaxCuts. We compare LK-GOMEA, using both asymmetric and symmetric linkage kernels, with GOMEA and DSMGA-II [10]. GOMEA and LK-GOMEA use the LT as linkage structure (per kernel). In order to adapt the population size during a run, all approaches, including DSMGA-II, use the IMS as described in [5].

We measure the number of evaluations and number of milliseconds until the optimal solution is found. Limits are set to 100,000,000 evaluations and 6 hours of computational time per run. Each experiment is repeated 30 times. The experiments are performed on 2x AMD EPYC 7282 @ 2.8 GHz with 32 cores total with 252 GB of RAM. At most 30 experiments were running simultaneously, with each experiment being single-threaded.

Our goal is to compare scalability. We therefore perform pairwise statistical tests between GOMEA, DSMGA-II and LK-GOMEA (both variants) for the highest value of ℓ, and if applicable, for each value of *fns*. Statistical significance tests are performed using the Mann-Whitney U-test [6, 15]. Due to the multiplicity of comparisons we perform the Holm-Bonferroni [9] correction, with $\alpha = 0.05$.

5.2 Experimental Results & Discussion

A summary of the number of settings in which an EA is better (i.e., requires less time or evaluations) is provided in Table 1. Details and individual comparisons are in the Supplementary material.

Table 1: Number of times (corresponding rank) an approach is statistically significantly better over all single-objective experiments for number of evaluations-to-optimum (#eval) and time-to-optimum (time).

Approach	#eval	time
DSMGA-II	3 (4)	4 (3)
GOMEA	4 (3)	10 (1)
LK-GOMEA (Asymmetric)	9 (2)	7 (2)
LK-GOMEA (Symmetric)	12 (1)	10 (1)

The scalability results for Best-of-Traps (BoT) can be found in Figure 3. The reliability, i.e., the probability of a run successfully finding the desired optimum within the limits set, can be found in Supplementary material.

When BoT consists of a single sub-problem, BoT is essentially a concatenated deceptive trap function. All approaches can solve BoT in this case scalably and reliably. It is here that the overhead of learning many linkage models in LK-GOMEA is most noticeable, as it is superfluous. While the number of evaluations required increases slightly – within an order of magnitude – the amount of time increases by one or two orders of magnitude.

The gap closes as the number of sub-problems in BoT is increased. DSMGA-II fails to solve BoT with four functions (or more) and a problem size larger than 40 variables. Furthermore, at 80 variables and a similar number of sub-problems, GOMEA becomes less reliable. GOMEA is unable to solve problems larger than 40 variables with 8 functions reliably, failing on all runs except for one.

By stark contrast, LK-GOMEA can solve all BoT instances for all problem sizes tested. It is apparent that with eight functions with different linkage, learning multiple linkage models becomes a requirement. LK-GOMEA can solve these problems while maintaining similar scalability as the number of sub-problems grows.

The scalability results for MaxCut and Worst-of-MaxCuts can be found in Figure 4. Performance on MaxCut shows similar characteristics as the single-function BoT (i.e., the original permuted deceptive trap function). This is most likely due to the fact that a single MaxCut instance has little variation in linkage structure among its local optima, especially in case of fully connected graphs as utilized here. However, as the problem is more difficult than deceptive trap functions in general, the gap between GOMEA (the best performing EA) and LK-GOMEA is smaller.

For Worst-of-MaxCuts with 2 sub-problems, at $\ell = 100$ a substantial number of runs of GOMEA and DSMGA-II fail to find the optimum, while LK-GOMEA finds the optimum consistently (see Figure 4). This indicates the usefulness and larger problem-solving capacity of LK-GOMEA also in this worst-case problem. Though not tested, it is expected that the advantage of LK-GOMEA would only increase if the number of MaxCut sub-problems would increase.

6 MULTI-OBJECTIVE EXPERIMENTS

6.1 Experimental Setup

We again consider Best-of-Traps (BoT) and MaxCut in three configurations: BoT vs BoT, BoT vs MaxCut and MaxCut vs MaxCut. Each objective uses a different instance to ensure that the objectives are not strongly correlated. As the DSMGA-II code used does not support MO optimization, we only use MO-GOMEA [14]. Furthermore, in addition to using a domination-based acceptation criterion, we use the scalarization scheme described in [13]. Finally, for the LK variant, we test both approaches in 2 combinations: using asymmetric (Asym) and symmetric neighborhood (Sym) linkage kernels.

We measure the normalized hypervolume (HV) of the elitist archive over the number of evaluations. HV is the volume/area dominated between a front and a reference point [17, 29]. To compute the HV, the ranges of each objective are normalized by the ranges spanned by the reference front. The reference point is taken similarly to [17], placing it at an offset of 0.05 of the worst objective values present, ensuring that the extreme points can contribute HV. The normalized HV is then obtained by dividing the HV of the archive by the HV of the reference front.

Limits on the number of evaluations and time, the hardware used, and other experimental considerations are identical to the single-objective experiments. Statistical tests are also performed similarly as for the single-objective experiments, but now to compare the HV at the evaluation limit for instances where $\ell = 100$.

6.2 Experimental Results & Discussion

A summary, stating the number of cases in which an approach is statistically significantly better than another in a setting can be found

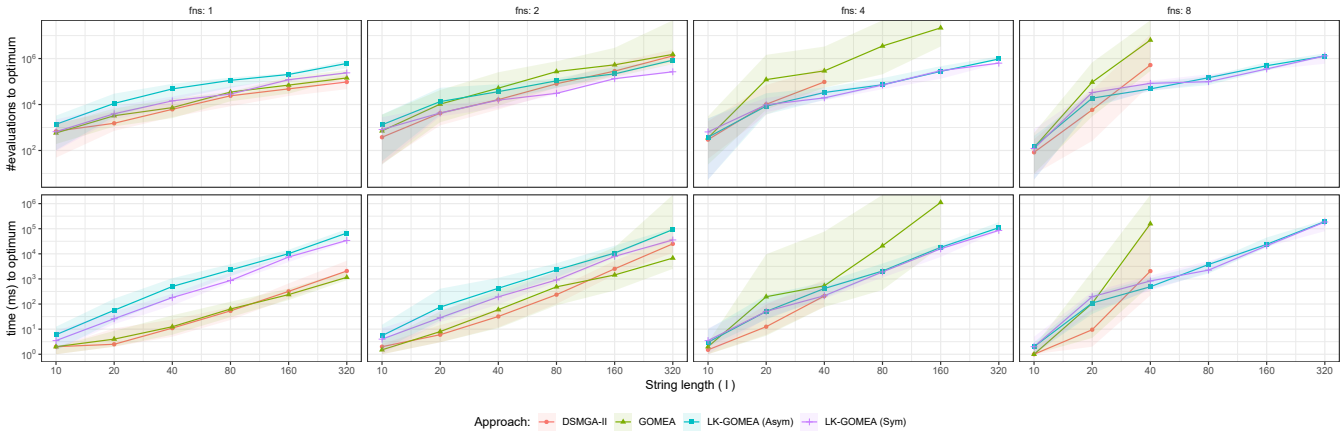

Figure 3: Scalability in number of evaluations (top) and time in ms (bottom) on BoT for various problem sizes (i.e. string lengths ℓ) and number of subfunctions *fns*. The line represents the median, whereas the shaded area ranges from the 95th percentile to the 5th percentile. If the median run was unable to find the optimum within the allotted budget, the point is left out.

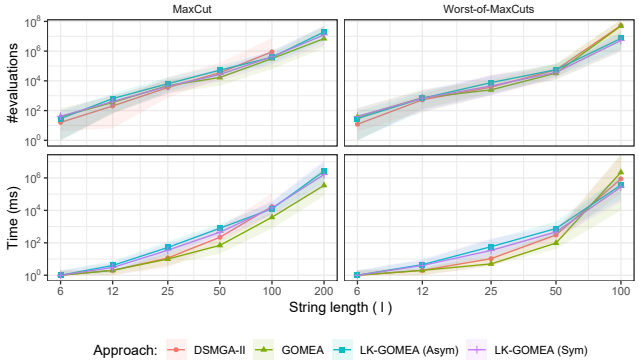

Figure 4: Scalability in evaluations to optimum (top) and time to optimum (bottom) on MaxCut (left) and Worst-of-Maxcuts (right) with 2 problem instances, both for various problem sizes (i.e., string lengths). No approach could consistently find the optimum on Worst-of-Maxcuts for $\ell = 200$. The line and shaded area are defined similarly to Figure 3.

Table 2: Number of times an approach is statistically significantly better on Best-of-Traps vs Best-of-Traps (and the corresponding rank).

Approach	Count (rank)
Objective / Domination	0 (6)
Objective / Scalarized	3 (5)
Asymmetric Kernel / Domination	5 (4)
Asymmetric Kernel / Scalarized	9 (2)
Symmetric Kernel / Domination	7 (3)
Symmetric Kernel / Scalarized	15 (1)

in Table 2. Detailed pairwise tests can be found in Supplementary material.

Convergence graphs for BoT vs BoT, with $\ell = 100$ and $fns \in \{1, 2, 4, 8\}$ can be found in Figure 5. Furthermore, final fronts corresponding to the median run of each approach can be found in Figure 6. Graphs and data for BoT vs MaxCut and MaxCut vs MaxCut can be found in the Supplementary material.

The graphs show that objective-space clustering is sufficient when multiple structures along the front are the result of each objective having a single different linkage structure, i.e., when $fns = 1$. However, objective-space clustering becomes increasingly insufficient as *fns* increases in both objectives. Both variants of LK-GOMEA then obtain better hypervolume values.

Of special note is that BoT seems to be hard to solve around the center part of the front, while the extremes (i.e., single-objective regions) of the front are easier: while the optima for the extremes are found in the median run by LK-GOMEA, in the central region to the front there is a gap to the best solution we found using the pairwise decomposition of the sub-problems as outlined in Section 4.3. Furthermore, although not extensively tested, from several additional runs we did, it appears that solving a weighted combination in single-objective is more difficult in this region, with the MO approach actually obtaining results much closer to the front. A reason for this may be that at the center of the front, search needs to account for the structure of both objectives. Since in our case, each of the two functions has *fns* modes, this results in up to fns^2 potential modes in the center of the Pareto front. As a multi-objective approach is inherently better at preserving diversity from the perspective of finding multiple solutions along the Pareto front, it is able to maintain and utilize building blocks required to succeed in solving individual functions much better. Additional research is needed however to look into this phenomenon further.

7 DISCUSSION

In our work, the neighborhood size has been set to $\sqrt{|P|}$ so as to ensure there was sufficient information to build a linkage model, while being small enough that the locality assumptions would likely hold. The best setting for this parameter is however dependent on the problem as well. A smaller neighborhood is a better fit for problems with a higher degree of multi-modality and multiple linkage structures, whereas a larger neighborhood performs better for problems with no multi-modality. Furthermore, as seen in the MO problem, the number of solutions in each niche can be different.

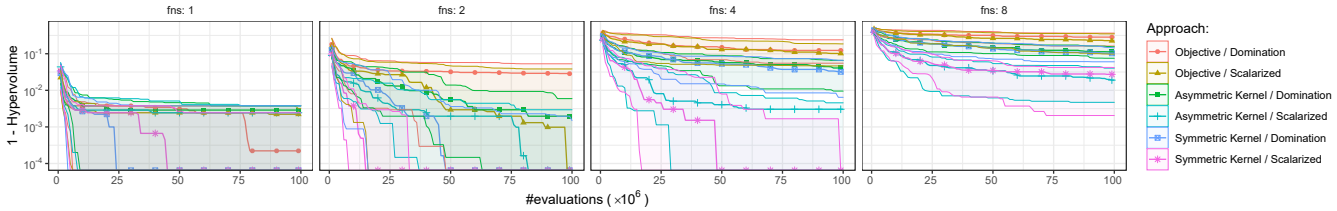

Figure 5: Convergence Graphs for $\ell = 100$ on BoT (Objective 1) vs BoT (Objective 2), for $\ell = 100$ and different numbers of subfunctions (i.e., $fns \in \{1, 2, 4, 8\}$). The points (and the solid line on which they sit) represent the median hypervolume over runs, whereas the shaded area and its bounded area represent the 5th and 95th percentile.

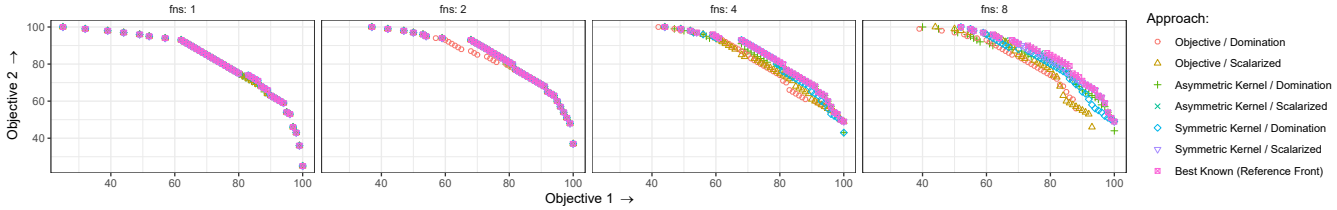

Figure 6: Approximation fronts for the run with the closest-to median obtained hypervolume for $\ell = 100$ on BoT (Objective 1) vs BoT (Objective 2).

This indicates that a singular value for the neighborhood size is insufficient. Certain modes can vary in their number of solutions, and the neighborhood size needs to be adapted accordingly. An adaptive scheme is therefore of interest for future research.

The fully connected instances for MaxCut do not lend themselves well to linkage learning, and usage of a local neighborhood does not improve this. It of interest whether MaxCut always has a more global structure, or if there are instances with a different graph structure or distribution of weights, in which linkage kernels help.

Furthermore, the FI operator performs recombination with the (global) elitist(s). While this is generally beneficial, a solution is not necessarily part of the same mode as the elitist. Applying this operator may therefore waste evaluations and lead to premature replacement of these solutions. An alternative scheme to perform global recombination, or an adaptation of FI that takes into account locality as well, may therefore be of interest for future work.

LK-GOMEA, the new variant of GOMEA evaluated in this work, spends more time on model-building per evaluation than prior versions of GOMEA. The added cost in time may be too great to be applicable on problems where evaluations are computationally very cheap. As the neighborhoods of neighboring individuals tend to be similar, reusing local models or learning linkage models over larger (combined) neighborhoods, could result in a notable improvement without a significant reduction in performance.

Overall, despite the potential possibilities for improvement, linkage kernels have shown to be a useful addition, providing a higher degree of robustness, allowing approaches like LK-GOMEA to solve more complex, multi-structured problems.

8 CONCLUSION

Model-Based Evolutionary Algorithms can be highly scalable by capturing a problem's structure in a model. In this work we have shown that significantly more useful linkage models can be obtained by introducing linkage kernels, i.e., learning linkage over a local neighborhood, in case of problems that exhibit multiple structures at once. In particular, we have introduced LK-GOMEA, a novel variant of GOMEA that utilizes search space locality by learning a separate linkage model for each solution in the population using a subset of solutions in the population deemed to be in its local neighborhood. While the definition of neighborhood is of special importance, and may well need to be defined differently for particular problems, we observed that a symmetric neighborhood was more robust and led to better performance by LK-GOMEA than using standard (asymmetric) KNN. We have shown LK-GOMEA to be capable of scalably solving a novel benchmark problem that has a tunable number of problem structures where current state-of-the-art model-based evolutionary algorithms such as GOMEA and DSMGA-II fail. Moreover, LK-GOMEA was found to also be superior in solving multi-objective problems that exhibit multiple linkage structures, especially at the problem length and the number of underlying problem structures increases. As this may well occur in complex real-world problems, we believe that we have provided a valuable and novel contribution to the body of work on MBEAs, moving the boundaries of the current state-of-the-art.

ACKNOWLEDGMENTS

This publication is part of the project "DAEDALUS - Distributed and Automated Evolutionary Deep Architecture Learning with Unprecedented Scalability" with project number 18373 of the research programme Open Technology Programme which is (partly) financed by the Dutch Research Council (NWO). Other financial contributions as part of this project have been provided by Elekta AB and Ortec Logiqcare B.V.. Special thanks go out to Arkadiy Dushatskiy for providing the source code for DSMGA-II with the Interleaved Multi-start Scheme, as well as the source code of the most recent version of GOMEA from [5].

REFERENCES

[1] Benhui Chen and Jinglu Hu. 2010. An Adaptive Niching EDA Based on Clustering Analysis. In *IEEE Congress on Evolutionary Computation*. 1–7. https://doi.org/10.1109/CEC.2010.5586387

[2] Ping-Lin Chen, Chun-Jen Peng, Chang-Yi Lu, and Tian-Li Yu. 2017. Two-Edge Graphical Linkage Model for DSMGA-II. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '17)*. Association for Computing Machinery, New York, NY, USA, 745–752. https://doi.org/10.1145/3071178.3071236

[3] Kalyanmoy Deb and David E. Goldberg. 1994. Sufficient Conditions for Deceptive and Easy Binary Functions. *Annals of Mathematics and Artificial Intelligence* 10, 4 (Dec. 1994), 385–408. https://doi.org/10.1007/BF01531277

[4] Arkadiy Dushatskiy, Tanja Alderliesten, and Peter A. N. Bosman. 2021. A Novel Approach to Designing Surrogate-assisted Genetic Algorithms by Combining Efficient Learning of Walsh Coefficients and Dependencies. *ACM Transactions on Evolutionary Learning and Optimization* 1, 2 (July 2021), 5:1–5:23. https://doi.org/10.1145/3453141

[5] Arkadiy Dushatskiy, Marco Virgolin, Anton Bouter, Dirk Thierens, and Peter A. N. Bosman. 2021. Parameterless Gene-pool Optimal Mixing Evolutionary Algorithms. (Sept. 2021). arXiv:2109.05259

[6] Michael P. Fay and Michael A. Proschan. 2010. Wilcoxon-Mann-Whitney or t-Test? On Assumptions for Hypothesis Tests and Multiple Interpretations of Decision Rules. *Statistics surveys* 4 (2010), 1–39. https://doi.org/10.1214/09-SS051

[7] Brian W. Goldman and William F. Punch. 2014. Parameter-Less Population Pyramid. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO '14)*. Association for Computing Machinery, New York, NY, USA, 785–792. https://doi.org/10.1145/2576768.2598350

[8] Georges R. Harik and Fernando G. Lobo. 1999. A Parameter-Less Genetic Algorithm. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation - Volume 1 (GECCO'99)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 258–265.

[9] Sture Holm. 1979. A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics* 6, 2 (1979), 65–70.

[10] Shih-Huan Hsu and Tian-Li Yu. 2015. Optimization by Pairwise Linkage Detection, Incremental Linkage Set, and Restricted / Back Mixing: DSMGA-II. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation (GECCO '15)*. Association for Computing Machinery, New York, NY, USA, 519–526. https://doi.org/10.1145/2739480.2754737

[11] Hui Li and Qingfu Zhang. 2009. Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation* 13, 2 (April 2009), 284–302. https://doi.org/10.1109/TEVC.2008.925798

[12] Yiping Liu, Hisao Ishibuchi, Yusuke Nojima, Naoki Masuyama, and Ke Shang. 2018. A Double-Niched Evolutionary Algorithm and Its Behavior on Polygon-Based Problems. In *Parallel Problem Solving from Nature – PPSN XV (Lecture Notes in Computer Science)*, Anne Auger, Carlos M. Fonseca, Nuno Lourenço, Penousal Machado, Luís Paquete, and Darrell Whitley (Eds.). Springer International Publishing, Cham, 262–273. https://doi.org/10.1007/978-3-319-99253-2_21

[13] Ngoc Hoang Luong, Tanja Alderliesten, and Peter A. N. Bosman. 2018. Improving the Performance of MO-RV-GOMEA on Problems with Many Objectives Using Tchebycheff Scalarizations. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '18)*. Association for Computing Machinery, New York, NY, USA, 705–712. https://doi.org/10.1145/3205455.3205498

[14] Ngoc Hoang Luong, Han La Poutré, and Peter A.N. Bosman. 2014. Multi-Objective Gene-Pool Optimal Mixing Evolutionary Algorithms. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO '14)*. Association for Computing Machinery, New York, NY, USA, 357–364. https://doi.org/10.1145/2576768.2598261

[15] H. B. Mann and D. R. Whitney. 1947. On a Test of Whether One of Two Random Variables Is Stochastically Larger than the Other. *The Annals of Mathematical Statistics* 18, 1 (March 1947), 50–60. https://doi.org/10.1214/aoms/1177730491

[16] Jean P. Martins and Alexandre C. B. Delbem. 2016. Pairwise Independence and Its Impact on Estimation of Distribution Algorithms. *Swarm and Evolutionary Computation* 27 (2016), 80–96. https://doi.org/10.1016/j.swevo.2015.10.001

[17] Krzysztof Nowak, Marcus Märtens, and Dario Izzo. 2014. Empirical Performance of the Approximation of the Least Hypervolume Contributor. In *Parallel Problem Solving from Nature – PPSN XIII (Lecture Notes in Computer Science)*, Thomas Bartz-Beielstein, Jürgen Branke, Bogdan Filipič, and Jim Smith (Eds.). Springer International Publishing, Cham, 662–671. https://doi.org/10.1007/978-3-319-10762-2_65

[18] Martin Pelikan, David E. Goldberg, and Erick Cantú-Paz. 1999. BOA: The Bayesian Optimization Algorithm. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation - Volume 1 (GECCO'99)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 525–532.

[19] Martin Pelikan, Mark W. Hauschild, and Dirk Thierens. 2011. Pairwise and Problem-Specific Distance Metrics in the Linkage Tree Genetic Algorithm. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO '11)*. Association for Computing Machinery, New York, NY, USA, 1005–1012. https://doi.org/10.1145/2001576.2001713

[20] Martin Pelikan, Kumara Sastry, and David E. Goldberg. 2005. Multiobjective hBOA, Clustering, and Scalability. In *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation (GECCO '05)*. Association for Computing Machinery, New York, NY, USA, 663–670. https://doi.org/10.1145/1068009.1068122

[21] Mike Preuss. 2010. Niching the CMA-ES via Nearest-Better Clustering. In *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO '10)*. Association for Computing Machinery, New York, NY, USA, 1711–1718. https://doi.org/10.1145/1830761.1830793

[22] Michal Witold Przewozniczek and Marcin Michal Komarnicki. 2020. Empirical Linkage Learning. *IEEE Transactions on Evolutionary Computation* 24, 6 (Dec. 2020), 1097–1111. https://doi.org/10.1109/TEVC.2020.2985497

[23] M. Ruciński, D. Izzo, and F. Biscani. 2010. On the Impact of the Migration Topology on the Island Model. *Parallel Comput.* 36, 10 (Oct. 2010), 555–571. https://doi.org/10.1016/j.parco.2010.04.002

[24] Mourad Sefrioui and Jacques Périaux. 2000. A Hierarchical Genetic Algorithm Using Multiple Models for Optimization. In *Parallel Problem Solving from Nature PPSN VI (Lecture Notes in Computer Science)*, Marc Schoenauer, Kalyanmoy Deb, Günther Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel (Eds.). Springer, Berlin, Heidelberg, 879–888. https://doi.org/10.1007/3-540-45356-3_86

[25] Dirk Thierens. 1999. Scalability Problems of Simple Genetic Algorithms. *Evolutionary Computation* 7, 4 (Dec. 1999), 331–352. https://doi.org/10.1162/evco.1999.7.4.331

[26] Dirk Thierens and Peter A.N. Bosman. 2011. Optimal Mixing Evolutionary Algorithms. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO '11)*. Association for Computing Machinery, New York, NY, USA, 617–624. https://doi.org/10.1145/2001576.2001661

[27] L. Darrell Whitley. 1991. Fundamental Principles of Deception in Genetic Search. *Foundations of Genetic Algorithms* 1 (Jan. 1991), 221–241. https://doi.org/10.1016/B978-0-08-050684-5.50017-3

[28] Q. Zhang and H. Li. 2007. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation* 11, 6 (Dec. 2007), 712–731. https://doi.org/10.1109/TEVC.2007.892759

[29] Eckart Zitzler and Lothar Thiele. 1998. Multiobjective Optimization Using Evolutionary Algorithms — A Comparative Case Study. In *Parallel Problem Solving from Nature — PPSN V (Lecture Notes in Computer Science)*, Agoston E. Eiben, Thomas Bäck, Marc Schoenauer, and Hans-Paul Schwefel (Eds.). Springer, Berlin, Heidelberg, 292–301. https://doi.org/10.1007/BFb0056872