Approximating Pareto local optimal solution networks

Shoichiro Tanaka, Gabriela Ochoa, Arnaud Liefooghe, Keiki Takadama, Hiroyuki Sato

Approximating Pareto Local Optimal Solution Networks

Shoichiro Tanaka
tanaka-shoichiro@fukuchiyama.ac.jp
The Univerisity of Fukuchiyama
Fukuchiyama, Kyoto, Japan

Gabriela Ochoa
gabriela.ochoa@stir.uk
University of Stirling
Stirling, FK9 4LA, UK

Arnaud Liefooghe
arnaud.liefooghe@univ-littoral.fr
Université du Littoral Côte d'Opale
Calais, France

Keiki Takadama
keiki@inf.uec.ac.jp
The Univerisity of Electro-communications, Chofu, Tokyo, Japan

Hiroyuki Sato
h.sato@uec.ac.jp
The Univerisity of Electro-communications, Chofu, Tokyo, Japan

ABSTRACT

The design of automated landscape-aware techniques requires low-cost features that characterize the structure of the target optimization problem. This paper approximates network-based landscape models of multi-objective optimization problems, which were constructed by full search space enumeration in previous studies. Specifically, we propose a sampling method using dominance-based local search for constructing an approximation of the Pareto local optimal solution network (PLOS-net) and its variant, the compressed PLOS-net. Both models are valuable to visualize and compute features on the distribution of Pareto local optima. We conduct experiments with multi-objective nk-landscapes and compare the features of full-enumerated PLOS-nets with that of approximate PLOS-nets. We analyze the correlation between landscape features and the performance of well-established multi-objective evolutionary and local search algorithms. Our results show that approximated networks can predict algorithm performance and provide recommendation for algorithm selection with the same level of accuracy, even though they are much more computationally affordable compared to full-enumerated networks. We finally illustrate how the approximate PLOS-net scale to large-size instances.

CCS CONCEPTS

• **Theory of computation** → Optimization with randomized search; Discrete optimization; • **Applied computing** → Multi-criterion optimization and decision-making.

KEYWORDS

Multi-objective optimization, Landscape analysis, NK-landscapes.

ACM Reference Format:
Shoichiro Tanaka, Gabriela Ochoa, Arnaud Liefooghe, Keiki Takadama, and Hiroyuki Sato. 2024. Approximating Pareto Local Optimal Solution Networks. In *Genetic and Evolutionary Computation Conference (GECCO '24), July 14–18, 2024, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3638529.3653999

GECCO '24, July 14–18, 2024, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-0494-9/24/07.
https://doi.org/10.1145/3638529.3653999

1 INTRODUCTION

Over the past thirty years, many evolutionary multi-objective optimization (EMO) algorithms and related search heuristics have been proposed [5, 6]. However, it remains unclear which algorithm to choose given a particular multi-objective optimization problem instance. In response to this, much attention has been devoted to automated algorithm selection and configuration based on problem features [10, 12]. In such approaches, a set of features characterizing the problem is provided as input to a pre-trained statistical or machine learning model in order to predict algorithm performance. In black-box scenarios, however, problem-specific features might not be available. This is where landscape analysis takes on its full meaning by providing general-purpose features from a high-level perspective [13, 22, 23, 28]. It is therefore essential to design features that are both meaningful and computationally inexpensive in landscape-aware technologies. Landscape analysis for multi-objective optimization has made recent progress, including studies of algorithm selection using multi-objective landscape features [2, 18, 21, 32].

The global structure of landscapes is known to strongly impact the performance of optimization methods [14, 22]. Local optima networks (LONs) [25] have been extensively used to study the global structure of single-objective landscapes in terms of the distribution and connectivity pattern of local optima [24, 26, 30]. The PLOS-net [19] and its variant the C-PLOS-net [20] extend LONs to multi-objective optimization, and were shown to provide valuable features for algorithm selection. However, these models were constructed from a full enumeration of the search space for small instances. In order to scale up these models to problems with realistic size, sampling mechanisms are required to approximate the underlying landscapes. In this paper, we construct approximate PLOS-net and C-PLOS-net models from a carefully-designed sampling method. Our contributions can be summarized as follows:

(1) We propose a new sampling method based on Pareto local search (PLS) [27], a local search heuristic based on dominance, and compare it with two traditional sampling methods from landscape analysis: random and adaptive walks;
(2) We compare approximate PLOS-nets against complete PLOS-nets to clarify the impact of sampling on these models;
(3) We analyze correlations between network features and the performance of established multi-objective algorithms.
(4) We begin to test the power of the new sampling method to construct PLOS-net models for large instances.

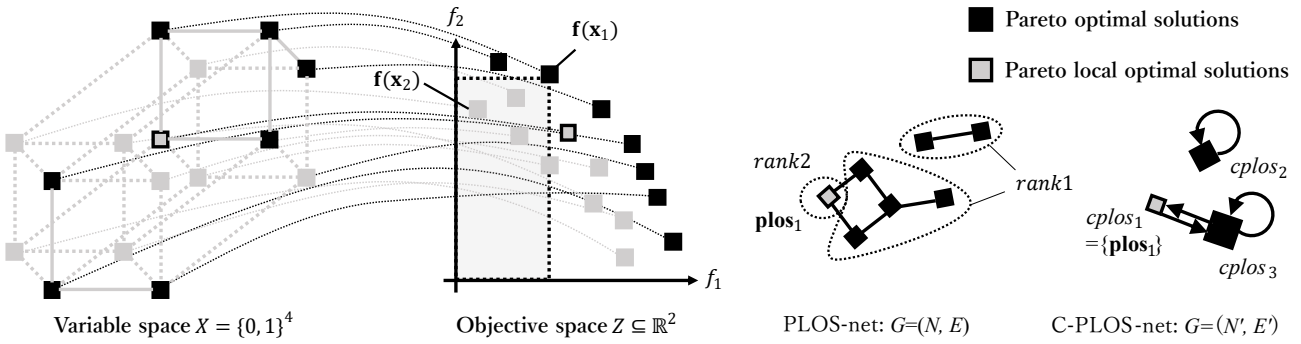

Figure 1: PLOS-net and C-PLOS-net construction on a small example instance. From left to right: four-dimensional variable space $X = \{0,1\}^4$, two-dimensional objective space $Z \subseteq \mathbb{R}^2$, PLOS-net, and C-PLOS-net.

We conduct experiments on ρmnk-landscapes, a family of pseudo-Boolean optimization problems with tunable properties.

The paper is organized as follows. Section 2 provides the necessary background. Section 3 describes the sampling techniques. Section 4 gives the experimental design and features. Section 5 shows the impact of sampling on (C-)PLOS-nets. Section 6 concludes the paper and discusses further research.

2 BACKGROUND

2.1 Multi-objective Optimization

We are interested in solving a multi-objective optimization problem with n variables and m objectives to be maximized, such that $\mathbf{f}: X \rightarrow Z$. Each solution $\mathbf{x} = (x_1, \dots x_n)$ in the variable space X is assigned a vector $\mathbf{z} = (z_1, \dots z_m)$ in the objective space $Z \subseteq \mathbb{R}^m$, and $z_i = f_i(\mathbf{x})$ for all $i \in (1, \dots, m)$. Given two solutions $\mathbf{x}, \mathbf{x}' \in X$ and their objective vectors $\mathbf{z} = \mathbf{f}(\mathbf{x}), \mathbf{z}' = \mathbf{f}(\mathbf{x}') \in Z$, $\mathbf{x}$ dominates $\mathbf{x}'$ iff $z_i \geqslant z'_i$ for all $i \in \{1, \dots, m\}$, and there exists a $j \in \{1, \dots, m\}$ such that $z_j > z'_j$. A solution $\mathbf{x} \in X' \subseteq X$ is non-dominated in X' if there is no solution $\mathbf{x}' \in X'$ such that $\mathbf{x}'$ dominates $\mathbf{x}$. Let $NDS(X')$ be the set of non-dominated solutions in X'; the set of Pareto optimal solutions (POS) is $NDS(X)$.

There exist different ways of ranking solutions to a multi-objective optimization problem. One of them is based on non-dominated sorting [8] used, e.g., in NSGA2 [7]. Given a set $X' \subseteq X$, solutions are sorted into distinct layers of mutually non-dominated solutions. These non-dominated solutions $NDS(X')$ are $rank1$ solutions: $rank_1(X')$. Non-dominated solutions from X' excluding $rank_1(X')$ are $rank2$ solutions, and so on.

2.2 ρmnk-Landscapes

A ρmnk-landscape [31] is multi-objective pseudo-Boolean function where the number of objectives, variables, co-variables and the correlation between the objectives can be tuned by means of benchmark parameters. It is an extension of mnk-landscapes, where each objective is an independent nk-landscape [1, 15]. A ρmnk-landscape can be formulated as follows:

$$\mathbf{f}_{\rho mnk}(\mathbf{x}) = (f_{nk,1}(\mathbf{x}), \dots, f_{nk,m}(\mathbf{x})) \tag{1}$$

such that:

$$f_{nk,j}(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} g_{i,j}(\text{mask}_i(\mathbf{x})) \tag{2}$$

where $mask_i: \{0,1\}^n \rightarrow \{0,1\}^{k+1}$ maps the solution to the variable x_i and its $k < n$ co-variables other than x_j. Component-function $g_{i,j}: \{0,1\}^{k+1} \rightarrow [0,1]$ assigns a real-valued contribution associated with x_i to the j-th objective value. Following [11], those k co-variables are set uniformly at random among the $n-1$ variables other than x_i. As k increases, the nk-landscape gradually changes from a smooth uni-modal landscape to a rugged multi-modal landscape. In ρmnk-landscapes, the contribution of each variable is generated following a multivariate uniform distribution with a parameter $\rho > \frac{-1}{m-1}$ defining the correlation among objectives [31]. We use the same correlation coefficient ρ among all pairs of objectives and the same variable interactions for all the objectives. The source code of the ρmnk-landscapes generator is available at the following URL: https://gitlab.com/aliefooghe/mocobench.

2.3 Multi-objective Landscapes and Networks

2.3.1 Multi-objective Landscapes. The fitness landscape of a multi-objective optimization problem is defined as a triplet $(X, \mathcal{N}, \mathbf{f})$ such that X is the variable space, $\mathcal{N}: X \mapsto 2^X$ is a neighborhood structure, $f: X \rightarrow Z$ is an objective function vector. An example variable space $X = \{0,1\}^4$ is shown in **Figure 1** (far-left). We consider the neighborhood structure $\mathcal{N}$ based on Hamming distance: two solutions $\mathbf{x}, \mathbf{x}' \in X$ are neighbors if $\text{dist}(\mathbf{x}, \mathbf{x}') = 1$. Finally, the objectives $\mathbf{f}$ are as described in Section 2.2.

A solution $\mathbf{x} \in X$ is a *Pareto local optimal solution* (PLOS) if there is no neighbor $x' \in \mathcal{N}(x)$ which dominates $\mathbf{x}$ [27]. POS are colored and PLOS are framed in black in **Figure 1**. By definition, any Pareto optimal solution is a PLOS.

2.3.2 PLOS-nets. In [19], the Pareto local optimal solutions network (PLOS-net) is defined as an unweighted undirected graph $G = (N, E)$ where nodes N are the PLOS. Additionally, there is an edge $e(\mathbf{plos}, \mathbf{plos}') \in E$ from **plos** to **plos′** iff **plos** and **plos′** are mutual neighbors. A PLOS-net example is shown in **Figure 1** (middle-right).

2.3.3 C-PLOS-nets. A compressed version of the PLOS-net model, referred to as C-PLOS-net, was recently proposed in [20]. It aims to address several shortcomings, including the vast number of nodes and edges. A C-PLOS-net is a weighted and directed graph $G' = (N', E')$ where nodes N' are compressed PLOS (C-PLOS). C-PLOS correspond to connected components of each PLOS-net's sub-graph, composed of nodes with the same rank. There is an edge $e(cplos, cplos') \in E'$ if a neighbor of at least one solution in $cplos$ is included in $cplos'$. The *weight* of a compressed edge is the normalized count of uncompressed edges it encompasses; see [20] for details. A C-PLOS-net example is shown in **Figure 1** (far-right).

3 APPROXIMATE (C-)PLOS-NETS

Although (C-)PLOS-nets provide insightful information, it should be noted that they have been constructed by full-enumeration of the search space so far. To the best of our knowledge, no study has constructed a (C-)PLOS-net from sampled solutions, nor investigated the effect of sampling on its characteristics. Most landscape analysis metrics are computed from a sample of solutions [22, 23]. Two sampling techniques are generally used: random walks and adaptive walks [11, 33]. In addition, we propose a new sampling method using dominance-based local search, which is more suitable for constructing local optima networks.

3.1 Random and Adaptive Walks

In landscape analysis, sampling is often performed by means of a *walk* over the landscape. A walk is an ordered sequence of solutions $X_{\text{sample}} = (\mathbf{x}_1, \dots \mathbf{x}_\ell)$ such that $\mathbf{x}_1 \in X$ is selected at random and $\mathbf{x}_t \in \mathcal{N}(\mathbf{x}_{t-1})$ for all $t \in \{2, \dots, \ell\}$. During a *random walk* [33], there is no particular criterion to pick the neighboring solution at each step: a random neighbor is selected, even though the best neighbor is known. By contrast, an *adaptive walk* [11] selects an improving neighbor at each step — similar to hill climbing. In multi-objective optimization, improving neighbors are those that dominate the current search point. We follow the Pareto adaptive walk from [31]: the current solution is replaced by a random *dominating* neighbor. If no such neighbor exists, the adaptive walk terminates.

In general, walks do not necessarily evaluate all neighbors from the current solution. However, we recall that the sampling in this paper aims at identifying PLOS. We thus evaluate the objectives of all neighbors. This change enables us to flag PLOS from non-PLOS in X_{sample}. All solutions evaluated during the walk and their objective values are stored in an archive A. This archive does not only contain non-dominated solutions, since *all* visited solutions are included. This archive A is necessary to approximate the rank of solutions without the full-enumeration of the search space.

A random walk usually terminates after a pre-defined length ℓ, while an adaptive walk terminates when it converges to a PLOS. In this paper, we simply use the archive size $|A|$ as the termination criterion in order to fairly compare the different sampling techniques. In both walks, once a solution is chosen as a search point, it is not chosen again. This is to prevent evaluating its neighbors again in future iterations. If a walk converges before reaching the upper limit on the archive, it simply restarts from a randomly-selected solution $\mathbf{x} \in X$, taking over sample X_{sample} and archive A. As such, a user-defined parameter sets the upper bound on the archive size FE_{max}, that is the number of evaluated solutions among which the sampled PLOS belong.

Algorithm 1: Approximate PLOS-net construction with PLS-sampling.

```
Procedure construct_PLOS-net(f, FE_max):
    Initialize nodes and edges: PLOS_sample, E ← ∅
    Initialize archive and search points A, X_sample ← ∅
    while |A| < FE_max do
        x is randomly selected from X
        A, X_sample ← PLS(x, A, X_sample, f, FE_max)
    PLOS_sample ← {x ∈ X_sample | ∄x' ∈ N(x) s.t. x' dominates x}
    E ← connect_neighbors(PLOS_sample, E)
    return G_PLOS = (PLOS_sample, E)

Procedure PLS(x, A, X_sample, f, FE_max):
    Initialize candidate of search points X_c ← {x}
    X_c−s ← X_c \ X_sample
    while X_c−s ≠ ∅ ∧ |A| < FE_max do
        x is selected randomly from X_c−s
        X_sample ← X_sample ∪ {x}
        foreach neighbor x' ∈ N(x) do
            A ← A ∪ {(x', f(x'))}
        X_c ← NDS(X_c ∪ N(x))
        X_c−s ← X_c \ X_sample
    return A, X_sample

Procedure connect_neighbors(PLOS_sample, E):
    foreach plos ∈ PLOS_sample do
        foreach plos' ∈ PLOS_sample do
            if plos ∈ N(plos') then
                E ← E ∪ (plos, plos')
    return E
```

3.2 PLS Sampling

We further propose a sampling technique based on Pareto local search (PLS) [27]. The pseudo-code for constructing the approximate PLOS-net using PLS-sampling is given in **Algorithm 1**. The middle part of the pseudo-code is where solutions are sampled by PLS (procedure PLS). PLS is a dominance-based multi-objective local search that maintains an unbounded set of candidate search points X_c. At each iteration, the current solution is selected randomly from the candidate set X_c, and its neighbors are evaluated. Dominated solutions are discarded from X_c and non-dominated solutions $NDS(X_c \cup \mathcal{N}(\mathbf{x}))$ are updated candidates. The search point $\mathbf{x}$ is flagged as *visited* and is added to X_{sample}. When all solutions from X_c are flagged as *visited*, PLS has converged and the process restarts from another random solution. The whole process continues until the archive size reaches the upper bound FE_{max}. The archive A and *visited* search points X_{sample} are kept, but they no longer take part in the PLS selection to prevent premature convergence.

In the construct_PLOS-net procedure, PLOS are extracted from the search points X_{sample} obtained by PLS after sampling. To check whether a solution $\mathbf{x} \in X_{\text{sample}}$ is a PLOS, the objective values of $\mathbf{x}$ and of its neighbors $\mathcal{N}(\mathbf{x})$ are required. However, by construction, they are all stored in the archive A for *visited* search points. Therefore, no additional evaluation is required to extract $PLOS_{\text{sample}}$ from X_{sample}. At last, in the connect_neighbors procedure, when

Table 1: Parameters for the considered ρmnk-landscapes.

parameter	considered values
number of variables	$n = 16$
number of co-variables	$k \in \{0, 1, 2, 3, 4\}$
number of objectives	$m \in \{2, 3\}$
objectives correlation	$\rho \in \{0.0, \mp 0.2, \mp 0.4, \mp 0.7\}$ s.t. $\rho > \frac{-1}{m-1}$

any two solutions $\mathbf{plos}, \mathbf{plos}' \in PLOS_{\text{sample}}$ are neighbors, an undirected edge $(\mathbf{plos}, \mathbf{plos}')$ is added to the approximate PLOS-net.

4 EXPERIMENTAL SETUP

We provide below the experimental setup of our analysis, including the benchmark problems and the considered algorithms with their parameters. This section also describes the considered landscape features extracted from (C-)PLOS-nets.

4.1 Benchmark Problems

Regarding problem instances, we randomly generate 10 independent ρmnk-landscapes (Section 2.2) for each combination of parameters listed in **Table 1**, for a total of 650 instances. This setting allows us to compare the full-enumerated (C-)PLOS-net and approximate (C-)PLOS-nets for problems with two and three objectives, uncorrelated, independent and correlated objectives, as well as different numbers of co-variables.

4.2 Multi-objective Algorithms

We experiment with the following well-established multi-objective approaches, often used for ρmnk-landscapes [18, 20].

4.2.1 PLS. Pareto local search (PLS) [27] is the algorithm on which the PLS sampling is based, although there are a few differences between search and sampling. For search, we do not consider the archive size $|A|$ for the number of evaluations, but the product of the number of search points and of the neighborhood size n. This is a reasonable setting, given that PLS does not normally keep track of all evaluated solutions.

4.2.2 G-SEMO. The global simple evolutionary multi-objective optimizer (G-SEMO) is a an elitist steady-state multi-objective evolutionary algorithm [16]. Similar to PLS, it holds unbounded candidate search points set X_c and selects one solution $\mathbf{x} \in X_c$ randomly at each iteration. It then produces a single offspring $\mathbf{x}'$ by means of the standard stochastic bit-flip mutation. Each binary variable of $\mathbf{x}$ is independently flipped with a rate of $1/n$. The candidate set X_c is then updated with non-dominated solutions from $NDS(X_c \cup \{\mathbf{x}'\})$.

4.2.3 NSGA2. The non-dominated sorting genetic algorithm 2 (NSGA2) is a multi-objective evolutionary algorithm using dominance for selection [7]. At each generation t, the current population X_c is merged with its offspring X_c', and the merged population is split in non-dominated fronts $\{rank_1(X_c \cup X_c'), rank_2(X_c \cup X_c'), \dots\}$ based on non-dominated sorting. Offspring are generated following a standard setting with uniform crossover followed by stochastic bit-flip mutation. In NSGA2, selection is based on rank-values, and crowding distance is used as a tie breaker. The process of survival selection involves populating X_c with solutions that have the lowest rankings, while parent selection for reproduction involves binary tournaments between random solutions.

4.2.4 Parameter Setting. We conduct 30 independent runs of each algorithm per instance. For PLS, we consider the (natural) convergence of the algorithm as a termination criterion: We count the total number of evaluations performed by PLS before it gets stuck into a Pareto local optimum set [27]. For G-SEMO and NSGA2, the termination criterion is set to 10 000 evaluations. NSGA2 uses a population of size 100. This means that G-SEMO terminates after 10 000 generations and NSGA2 after 100 generations. For all algorithms, we return all non-dominated solutions found at each trial. Algorithm performance is measured as the proportion of Pareto optimal solutions identified (reso) and as the relative hypervolume [34]. The relative hypervolume is simply $hv/hv^\star$, such that $hv^\star$ is the best hypervolume calculated by full-enumeration. A higher relative hypervolume value is better, and hv = 1 (or reso = 1) actually means that all Pareto optimal solutions were found. Objective values are normalized in the range $[0, 1]$. The hypervolume reference point is set to the origin.

4.3 Network Construction

This section describes the features used to predict algorithm performance as well as the sampling parameters.

4.3.1 Network Metrics. Based on graph theory and complex networks, the PLOS-net and C-PLOS-net models are analyzed using informative structural metrics and statistics. We analyze a number of network metrics — the features — considered in previous studies [20]. They are presented in **Table 2**, where we first list metrics relevant to both models, followed by metrics relevant to the compressed model only. The metrics associated to nodes are their proportion, their ratio to Pareto optimal solutions and dominance ranks, their degree, and whether they are *isolated* nodes (i.e. without incident edges). For consistency, we use the same feature names than previous studies [20]. However, we emphasize that a sampling is used for approximate (C-)PLOS-nets. For example, node_pareto_n is the size of the Pareto set $rank_1(X)$ in the full-enumerated network. However, in the approximate network, node_pareto_n corresponds to the size of $rank_1(X_{\text{sample}})$ since there is no guarantee to sample (exact) Pareto optimal solutions.

Metrics related to edges are their density, the likelihood of nodes to connect with other nodes that share similar degrees (assortativity), as well as statistics reflecting the graph connectivity (starting with cc_) and pathways between nodes (starting with path_).

Finally, metrics that only apply to the compressed model include the mean node width (i.e. the number of compressed solutions in a node) and the mean rate of compression. Regarding the edges, we compute the strength (i.e. weighted degree) of all nodes and Pareto nodes ($rank1$ nodes), the average edge weights, as well as path distance metrics among nodes, and to Pareto nodes. These metrics were calculated using the `NetworkX` package from `Python` [9].

4.3.2 Network Construction. We construct networks with four algorithms: full-enumeration, random walk, adaptive walk, and the proposed PLS sampling. In full-enumeration, all PLOS are obtained. The ranks are determined by the objective values of all solutions.

Table 2: Considered network metrics. We use the same names and descriptions as [20] for consistency.

	metric	description
uncompressed and compressed networks	node_n	proportion of nodes
	node_pareto_n	proportion of Pareto nodes (nodes with rank 1)
	node_adj_pareto_n	proportion of nodes adjacent to a Pareto node
	node_rank_worst	maximum (worst) node rank
	degree_avg	average degree of nodes
	rank_degree_cor	node rank-vs-degree correlation
	isolated_n	proportion of isolated nodes
	pareto_isolated_n	proportion of Pareto nodes that are isolated
	isolated_rank_avg	average rank of isolated nodes
	edge_density	density of edges
	assort_degree	assortativity by degree
	cc_n	proportion of connected components (cc)
	cc_max	size of largest cc
	cc_avg	average size of cc
	cc_max_pareto	size of largest cc that contains a Pareto node
	cc_pareto_max	(average) size of cc with most Pareto nodes
	cc_pareto_avg	average number of Pareto nodes per cc
	cc_rank_avg_avg	mean of average rank per cc
	cc_rank_best_avg	mean of best rank per cc
	path_length_avg	average path length
	path_length_max	longest path length (diameter)
	path_pareto_exist	number of nodes connected to a Pareto node
	path_pareto_avg	avg. nb. of Pareto nodes a node is connected to
	path_length_pareto_avg	avg. (existing) path length to a Pareto node
compressed networks	node_width_avg	average node width
	node_cmpr	compression rate over nodes
	strength_avg	average node strength
	strength_pareto	sum of strengths of Pareto nodes
	rank_strength_cor	node rank-vs-strength correlation
	edge_weight_avg	average edge weight
	edge_cmpr	compression rate over edges
	dist_avg	average distance
	dist_max	longest distance
	dist_pareto_avg	avg. dist. to Pareto nodes (existing paths)

In contrast to full-enumeration, the three sampling algorithms output an archive of all visited solutions, together with the solution set X_{sample} for which the objective values of all neighbors have also been evaluated. We can get $PLOS_{\text{sample}}$ out of X_{sample} without additional evaluation. The cost of sampling is the upper limit of the archive size, which is the termination criterion used for sampling. We set this upper limit to 5% of the variable space, that is $FE_{max} = 3\,276$ evaluated solutions for problems with $n = 16$. The rank values required to construct a C-PLOS-net are computed according to the objective values recorded in the archive.

5 EXPERIMENTAL ANALYSIS

In this section, we visualize examples of PLOS-nets under different sampling techniques. We investigate the correlation between algorithm performance and network metrics of both full-enumerated and approximate (C-)PLOS-nets. We also challenge their ability to predict algorithm performance and to provide reliable recommendations for algorithm selection. We conclude the section by experimenting the ability of the PLS sampling method to construct the PLOS-net for large landscapes.

5.1 PLOS-net Visualization

Let us begin by examining the results of sampling on the construction of PLOS-nets. The ratios of PLOS obtained by each sampling technique (random walk, adaptive walk and PLS sampling) are shown in **Figure 2** for a subset of instances. We observe that the proportion of PLOS found with random walk is almost constant and not much affected by the benchmark parameters. By contrast, the larger ρ and the lower k, the larger the proportion of PLOS identified by the adaptive walk and the PLS sampling. Anyhow, they both consistently find much more PLOS than random walk. Since the number of PLOS obtained by random walk is almost zero, it is unlikely to construct a meaningful network with such sampling. Therefore, the results for random walk are omitted hereafter.

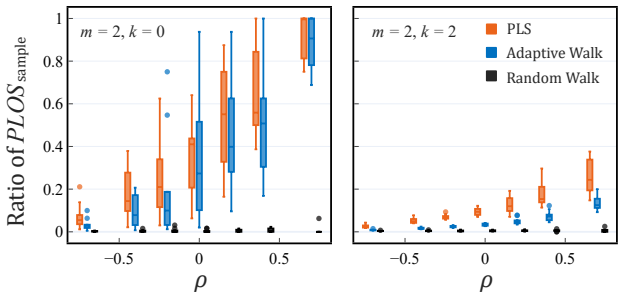

Figure 2: Ratio of $PLOS_{\text{sampled}}$ by each sampling technique for ρmnk-landscapes with $m = 2$ and $k \in \{0, 2\}$.

We continue by visually inspecting the difference of PLOS-net structures constructed by adaptive walk and PLS sampling. In **Figure 3**, all sub-plots correspond to a selected instance with $n = 16$, $\rho = -0.4$ and $k = 2$. The top-left plot shows in different colors the PLOS found by each approach in the objective space. The three other plots are the constructed PLOS-nets visualized under the rank layout from [20]. In order to highlight the difference between sampling techniques, the networks are visualized with a fixed layout computed for the full-enumerated model. The y-axis is the rank of each node with respect to the whole variable space, while the x-axis is determined by a force-directed layout. The PLOS from PLS sampling shows a bias towards nodes with better ranks — close to one. PLS being a multi-objective search algorithm, it naturally emphasizes solutions with a better rank. With adaptive walks, the rank of identified PLOS is more balanced. Furthermore, we remark that networks built from adaptive walks have extremely few connections between nodes. A possible explanation is that an adaptive walk converges to a single PLOS and then restarts from another random solution. As such, adaptive walks are not inherently designed to search for neighboring PLOS.

5.2 Impact of Metrics on Performance

Let us now assess the effect of PLOS-net and C-PLOS-net metrics on algorithm performance. **Figure 4** shows the Spearman rank correlation between each feature and the hypervolume (hv) obtained by the three algorithms. There is one color per sampling technique.

Apart from a few exceptions, the full-enumerated and PLS sampling network features follow the same trend overall. Features associated with connected components (cc_) were found to be important predictors for algorithm performance in [20]. For the number (cc_n) and the size (cc_max, cc_avg) of connected components, the PLS sampling actually strengthens the correlation. Conversely, for features related to the ranks of components (cc_rank_) and to the shortest path length (path_length_), the PLS sampling slightly weakens

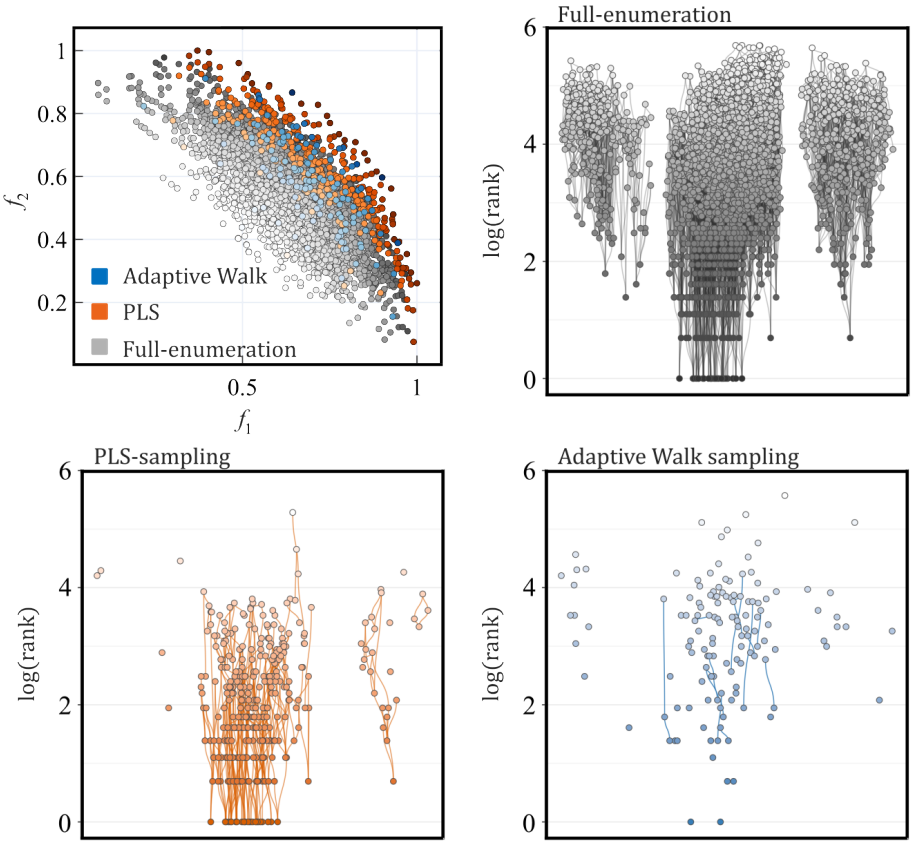

Figure 3: PLOS-nets for a ρmnk-landscape with $\rho = -0.4, m = 2, n = 16, k = 2$: Sampled PLOS in the objective space (top left), PLOS-net by full-enumeration (top-right), PLOS-net by PLS-sampling (bottom left), PLOS-net by adaptive walk sampling (bottom right). The color of nodes shows a gradient with higher intensity for lower (i.e. better) ranks.

the correlation. When looking at the adaptive walk sampling, the features have a different correlation trend compared against the features of the other two networks. They are rarely correlated with algorithm performance, especially for the PLS algorithm.

5.3 Predicting Performance

Let us continue by analyzing the impact of sampling on the feature-based prediction of algorithm performance. We build a regression model to predict the performance of the different algorithms using the network metrics as predictors. We not only include the relative hypervolume (hv) of PLS, G-SEMO and NSGA2, but also their Pareto resolution (reso) as well as the number of evaluations performed by PLS. We use random forests [3, 17], a well-established ensemble learning method for regression, with default parameters. We assess the prediction accuracy of the regression models using 30 replicates of 10-fold cross-validation. The results are reported in **Figure 5**. We first observe that the prediction accuracy with network features constructed by PLS sampling is almost the same as that with fully enumerated network features. These results suggest that appropriate sampling is sufficient to predict algorithm performance with (C-)PLOS-nets. In [20], it was reported that the metrics from the compressed model lead to a better prediction accuracy than metrics from the uncompressed model, with the exception of PLS (eval) and G-SEMO (hv). The same trends are confirmed by our results on full-enumerated network metrics. By contrast, we observe no improvement in prediction accuracy due to compression in networks constructed with PLS sampling. We attribute this to the use of approximate ranks when sampling. This is consistent with the observation that PLS sampling does not obtain quite the same pattern for rank-related features. The C-PLOS-net model compresses the sub-graphs with respect to rank values and is therefore sensitive to ranking. However, despite these slight differences, we emphasize that the prediction accuracy remains satisfactory. Interestingly, using PLS for sampling does not appear to negatively bias the performance prediction of algorithms other than PLS.

Compared against the features from the full-enumerated and PLS-sampled networks, the ones from the network constructed by means of adaptive walks significantly deteriorate the prediction accuracy. Indeed, with adaptive walks, we end up with extremely sparse networks with few connections between nodes. This drop in performance suggests that the number of PLOS, but also their distribution and connection are crucially important for explaining algorithm performance. Considering the results obtained so far, we choose to discard adaptive walk sampling and to focus on PLS sampling in order to construct approximate (C-)PLOS-nets below.

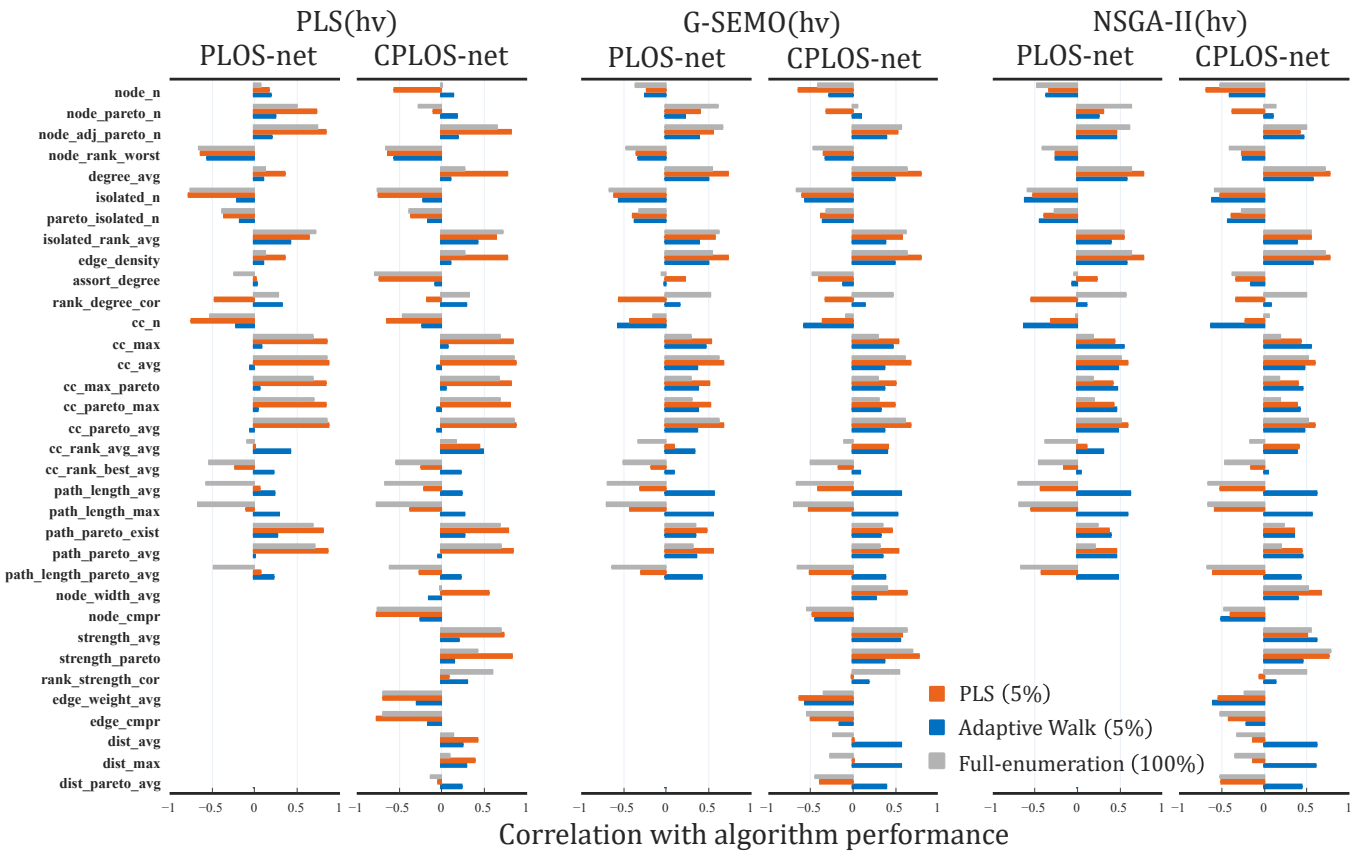

Figure 4: Correlation between network metrics and algorithm performance (hypervolume) for the different algorithms.

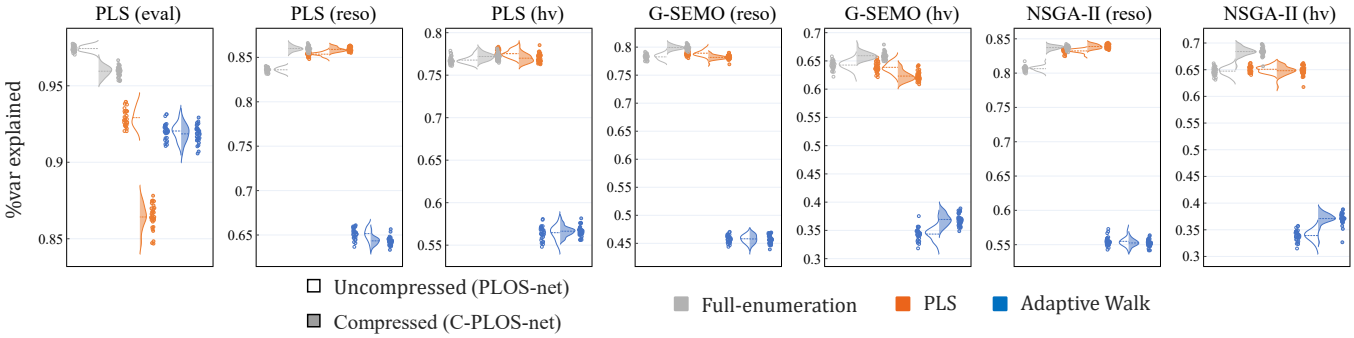

Figure 5: Prediction accuracy of regression models trained by algorithm performance using network metrics from the PLOS-net and the C-PLOS-net constructed by full-enumeration, PLS-sampling and adaptive walk sampling. Markers represent 30 replicates of 10-fold cross-validation.

5.4 Algorithm Selection

We train a CART decision tree [4, 29] for recommending the algorithm that obtained the best average hypervolume for a given instance using PLOS-net and C-PLOS-net metrics as predictors. This approach tackles a classification task where the classes represent the three considered algorithms. Out of the 650 instances from our analysis, we selected 144 instances where there was a substantial difference in performance between algorithms; i.e. a difference larger than 0.01 in relative hypervolume between the best and second-best algorithms. The obtained decision tree is shown in **Figure 6** for metrics from both the compressed and uncompressed networks constructed by PLS sampling. The values beneath each node of the tree represents the number of instances where PLS (left), G-SEMO (center) and NSGA2 (right) performs best, respectively.

The 10-fold cross-validated classification accuracy is 90.5% for networks constructed by PLS sampling. This accuracy is slightly higher than for full-enumerated networks (87.5%). We can therefore firstly conclude that the features from approximate (C-)PLOS-nets

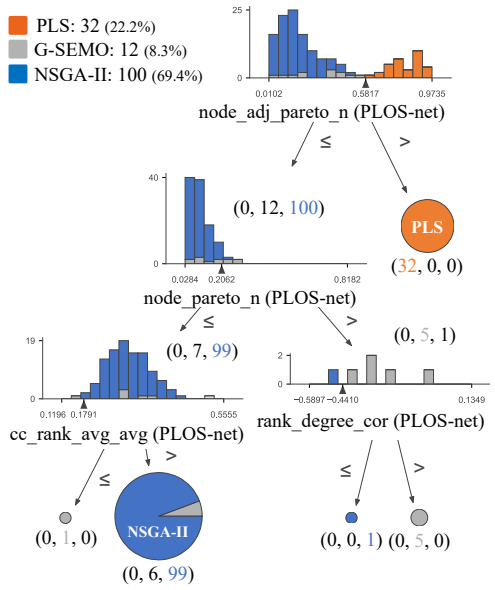

Figure 6: Hypervolume-based algorithm selection using metrics from both the approximate PLOS-net and C-PLOS-net constructed by PLS sampling.

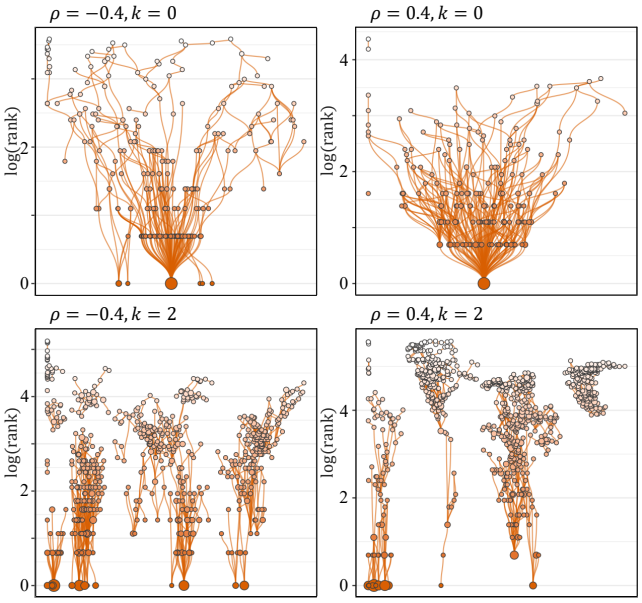

Figure 7: C-PLOS-net graphics for ρmnk-landscapes with $n = 100$, $\rho \in \{-0.4, 0.4\}$, $m = 2$, and $k \in \{0, 2\}$. The size of nodes and the darkness of edges are proportional to their sampling frequency. Node colors show a gradient with higher intensity for lower (i.e. better) ranks.

prove to be useful indicators for problem instances where an appropriate algorithm must be selected. We further observe that the first branch of the decision tree, corresponding to the proportion of PLOS adjacent to a rank 1 solution (pareto_adj_node_n), distinguishes NSGA2 and G-SEMO from PLS. The distribution of pareto_adj_node_n is clearly bimodal. A high value indicates that, in PLS sampling, solutions labeled as rank 1 were actually *visited* from many solutions. This feature reflects the dense connectivity between good-quality solutions, which appears to be beneficial for the PLS algorithm whose search process relies on a more local exploration than G-SEMO and NSGA2.

5.5 Approximate PLOS-nets for Large Landscapes

A key goal behind approximating PLOS-nets is to be able to scale up the model to larger landscapes. We here construct models for $n = 100$ variables and several values of the other benchmark parameters. Due to space constraints, a full analysis of these larger instances is not feasible in this article. However, we show in **Figure 7** some example landscapes for $n = 100$, $\rho \in \{-0.4, 0.4\}$ and $k \in \{0, 2\}$, using the C-PLOS-net model with the rank layout. The plots show that the models can differentiate the structure given by the landscape parameters k and ρ. Moreover, they show some similar trends as those observed for small landscapes. Specifically, the number of nodes and the number of separated components increases when we move from no ruggedness to larger ruggedness, $k = 0$ (top row) to $k = 2$ (bottom row). For $k = 0$ and $\rho = 0.4$, a single structure resembling a *funnel* is observed. This is consistent with the expectation of a smooth landscape. Interestingly, for both values of k, the number of nodes decreases with increasing ρ: correlated objectives tend to reduce the number of PLOS.

6 CONCLUSION

In this paper, we approximated the PLOS-net (and its compressed variant), a network-based fitness landscape model for multi-objective optimization, from sampled solutions. Besides common sampling methods in landscape analysis, random and adaptive walks, we proposed a new sampling technique using Pareto local search (PLS sampling) and examined their effectiveness.

We also investigated the effectiveness of features derived from approximate models in predicting algorithm performance and algorithm selection. Some features were found to correlate more strongly with algorithm performance under the PLS sampling. Interestingly, this approximate model, despite being smaller, maintains the same level of prediction accuracy as the original full-enumerated model. We confronted the approximate network metrics for algorithm selection. Decision trees with approximate network features were able to recommend the appropriate algorithm with about 90% accuracy in instances where performance substantially differs between algorithms.

To assess the sampling-based approximate model, we compared it to the full-enumerated model on small problem instances. We also started to explore how our approach scales with the search space size by visualizing the PLOS-net of larger problems. Future works will consider benchmarks of realistic problem sizes, alongside examining additional algorithms under different budgets, especially according to the number of objectives. We also emphasize the importance of investigating how the sample size impacts the meaningfulness of features, as well as its effect on algorithm prediction and recommendation models.

REFERENCES

[1] Hernán Aguirre and Kiyoshi Tanaka. 2007. Working principles, behavior, and performance of MOEAs on MNK-landscapes. *European Journal of Operational Research* 181, 3 (2007), 1670–1690.

[2] Hanan Alsouly, Michael Kirley, and Mario Andrés Muñoz. 2023. An Instance Space Analysis of Constrained Multiobjective Optimization Problems. *IEEE Transactions on Evolutionary Computation* 27, 5 (2023), 1427–1439.

[3] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (2001), 5–32.

[4] Leo Breiman, Jerome Friedman, Charles J. Stone, and R. A. Olshen. 1984. *Classification and regression trees.* Taylor & Francis, Andover, UK.

[5] Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen. 2007. *Evolutionary Algorithms for Solving Multi-Objective Problems* (second ed.). Springer.

[6] Kalyanmoy Deb. 2001. *Multi-Objective Optimization using Evolutionary Algorithms.* John Wiley & Sons.

[7] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197.

[8] David E. Goldberg. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Boston, MA, USA.

[9] Aric Hagberg, Pieter Swart, and Daniel S Chult. 2008. *Exploring network structure, dynamics, and function using NetworkX.* Technical Report. Los Alamos National Lab (LANL), Los Alamos, NM, USA.

[10] Serdar Kadioglu, Yuri Malitsky, Meinolf Sellmann, and Kevin Tierney. 2010. ISAC –Instance-Specific Algorithm Configuration. In *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010).* IOS Press, 751–756.

[11] Stuart A. Kauffman. 1993. *The Origins of Order.* Oxford University Press.

[12] Pascal Kerschke, Holger H. Hoos, Frank Neumann, and Heike Trautmann. 2019. Automated Algorithm Selection: Survey and Perspectives. *Evolutionary Computation* 27, 1 (2019), 3–45.

[13] Pascal Kerschke and Mike Preuss. 2019. Exploratory landscape analysis. In *Genetic and Evolutionary Computation Conference Companion, GECCO 2019.* ACM, Prague, Czech Republic, 1137–1155.

[14] Pascal Kerschke, Mike Preuss, Simon Wessing, and Heike Trautmann. 2015. Detecting Funnel Structures by Means of Exploratory Landscape Analysis. In *Genetic and Evolutionary Computation Conference (GECCO 2015).* ACM, Madrid, Spain, 265–272.

[15] Joshua D. Knowles and David Corne. 2007. Quantifying the Effects of Objective Space Dimension in Evolutionary Multiobjective Optimization. In *Evolutionary Multi-Criterion Optimization Conference, EMO 2007 (Lecture Notes in Computer Science, Vol. 4403).* Springer, Matsushima, Japan, 757–771.

[16] Marco Laumanns, Lothar Thiele, and Eckart Zitzler. 2004. Running time analysis of evolutionary algorithms on a simplified multiobjective knapsack problem. *Natural Computing* 3, 1 (2004), 37–51.

[17] Andy Liaw and Matthew Wiener. 2002. Classification and Regression by randomForest. *R News* 2, 3 (2002), 18–22.

[18] Arnaud Liefooghe, Fabio Daolio, Sébastien Verel, Bilel Derbel, Hernán Aguirre, and Kiyoshi Tanaka. 2020. Landscape-Aware Performance Prediction for Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation* 24, 6 (2020), 1063–1077.

[19] Arnaud Liefooghe, Bilel Derbel, Sébastien Verel, Manuel López-Ibáñez, Hernán Aguirre, and Kiyoshi Tanaka. 2018. On Pareto Local Optimal Solutions Networks. In *Parallel Problem Solving from Nature (PPSN XV).* Springer, Coimbra, Portugal, 232–244.

[20] Arnaud Liefooghe, Gabriela Ochoa, Sébastien Verel, and Bilel Derbel. 2023. Pareto local optimal solutions networks with compression, enhanced visualization and expressiveness. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2023).* ACM, Lisboa, Portugal, 713–721.

[21] Arnaud Liefooghe, Sébastien Verel, Benjamin Lacroix, Alexandru-Ciprian Zavoianu, and John A. W. McCall. 2021. Landscape Features and Automated Algorithm Selection for Multi-objective Interpolated Continuous Optimisation Problems. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2021).* ACM, Lille, France, 421–429.

[22] Katherine M. Malan. 2021. A Survey of Advances in Landscape Analysis for Optimisation. *Algorithms* 14, 2 (2021), 40.

[23] Olaf Mersmann, Bernd Bischl, Heike Trautmann, Mike Preuss, Claus Weihs, and Günter Rudolph. 2011. Exploratory Landscape Analysis. In *Genetic and Evolutionary Computation Conference (GECCO 2011).* ACM, Dublin, Ireland, 829–836.

[24] Gabriela Ochoa, Francisco Chicano, and Marco Tomassini. 2020. Global Landscape Structure and the Random MAX-SAT Phase Transition. In *Parallel Problem Solving from Nature (PPSN XVI).* Springer, Leiden, The Netherlands, 125–138.

[25] Gabriela Ochoa, Marco Tomassini, Sébastien Verel, and Christian Darabos. 2008. A study of NK landscapes' basins and local optima networks. In *Genetic and Evolutionary Computation Conference (GECCO 2008).* ACM, Atlanta, GA, USA, 555–562.

[26] Gabriela Ochoa and Nadarajen Veerapen. 2018. Mapping the global structure of TSP fitness landscapes. *Journal of Heuristics* 24, 3 (2018), 265–294.

[27] Luís Paquete, Tommaso Schiavinotto, and Thomas Stützle. 2007. On local optima in multiobjective combinatorial optimization problems. *Annals of Operations Research* 156, 1 (2007), 83–97.

[28] Hendrik Richter and Andries Engelbrecht (Eds.). 2014. *Recent Advances in the Theory and Application of Fitness Landscapes.* Springer.

[29] Terry Therneau and Beth Atkinson. 2022. *rpart: Recursive partitioning and regression trees.* R package version 4.1.16.

[30] Nadarajen Veerapen and Gabriela Ochoa. 2018. Visualising the global structure of search landscapes: genetic improvement as a case study. *Genetic Programming and Evolvable Machines* 19, 3 (2018), 317–349.

[31] Sébastien Verel, Arnaud Liefooghe, Laetitia Jourdan, and Clarisse Dhaenens. 2013. On the structure of multiobjective combinatorial search space: MNK-landscapes with correlated objectives. *European Journal of Operational Research* 227, 2 (2013), 331–342.

[32] Aljoša Vodopija, Tea Tušar, and Bogdan Filipič. 2022. Characterization of constrained continuous multiobjective optimization problems: A feature space perspective. *Information Sciences* 607 (2022), 244–262.

[33] E. D. Weinberger. 1990. Correlated and uncorrelatated fitness landscapes and how to tell the difference. *Biological Cybernetics* 63, 5 (1990), 325–336.

[34] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Grunert Da Fonseca. 2003. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation* 7, 2 (2003), 117–132.