Randomness in Local Optima Network Sampling

Sarah L. Thomson
University of Stirling
Stirling, United Kingdom
s.l.thomson@stir.ac.uk

Nadarajen Veerapen
Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL
Lille, France
nadarajen.veerapen@univ-lille.fr

Gabriela Ochoa
University of Stirling
Stirling, United Kingdom
gabriela.ochoa@stir.ac.uk

Daan van den Berg
VU University
Amsterdam, Netherlands
d.van.den.berg@vu.nl

ABSTRACT

We consider statistical randomness in the construction of *local optima networks* (LONs) and conduct a preliminary and exploratory study into this. LONs capture a fitness landscape into network format: the nodes are local optima, and edges are heuristic search transitions between them. Problem instances from the benchmark quadratic assignment problem library are used in the analysis. LONs are constructed using an iterated local search (ILS) and several different random seeds. Metrics are computed from the networks and visualised to assess the effect of randomness. Algorithm performance models for ILS runtime are built using metrics of LONs constructed using different seeds and the results compared. The results show that some LON metrics seem consistent across seeds, while others vary substantially. Additionally, the quality of algorithm performance models using LON metrics as predictors can differ depending on randomness. Finally, LON metrics associated with separate seeds can lead to different algorithm configuration recommendations for the same instance.

CCS CONCEPTS

• **Mathematics of computing** → **Graph algorithms**; *Combinatorial algorithms*; • **Theory of computation** → **Evolutionary algorithms**.

KEYWORDS

Fitness Landscapes, Quadratic Assignment Problem (QAP), Local Optima Networks (LONs), Iterated Local Search

ACM Reference Format:
Sarah L. Thomson, Nadarajen Veerapen, Gabriela Ochoa, and Daan van den Berg. 2023. Randomness in Local Optima Network Sampling. In *Genetic and Evolutionary Computation Conference Companion (GECCO '23 Companion), July 15–19, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3583133.3596309

GECCO '23 Companion, July 15–19, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0120-7/23/07.
https://doi.org/10.1145/3583133.3596309

1 INTRODUCTION

Local optima networks (LONs) [21] model local optima connectivity in fitness landscapes and can serve as a vehicle for understanding or predicting algorithm-problem interactions [1, 6]. Although several studies use small search spaces so that exhaustive enumeration of a LON can be conducted [21, 23, 25, 28, 32, 40], there has been an emergent appetite for larger spaces and therefore LON sampling in recent years [8, 20, 27, 31, 35, 39, 41]. The sampling algorithms contain elements of randomness such as starting location; however, it is not known how much of an impact this randomness has. Previous literature shows that evolutionary algorithm (EA) performance can be affected (sometimes to a substantial degree) by randomness [3, 11, 16, 17]; it therefore follows that LON construction — which is done by evolutionary and heuristic search — may partially depend on random factors. We argue that this is an important consideration, especially as the popularity of LONs for understanding and predicting search is growing. The purpose of this paper is to investigate the variation seen in LONs when providing different random seeds to an iterated local search algorithm used for constructing them. Instances from the quadratic assignment problem (QAP) library are used. Although the QAP is a case study here, the consideration of randomness in LONs would be relevant for other problem domains. We consider the nature of the LONs themselves, visualising and analysing several extracted features. Additionally, the predictive potency of LON metrics used in algorithm performance models is studied, with iterated local search (ILS) runtime as the response variable. Throughout the study, the effect of randomisation is given particular attention. The contributions of the work are as follows:

(1) An preliminary, exploratory investigation into the effect of randomness in LON construction
(2) Insights about LON metrics which show varying degrees of stability across different random seeds
(3) The finding that different algorithm configurations can be recommended, depending on the seed provided to LON construction

2 BACKGROUND

2.1 Fitness Landscapes

A fitness landscape [29] is composed of three parts: (S, N, f) : S is the full set of possible solutions; $N : S \longrightarrow 2^S$ is the neighbourhood function, which assigns a set of adjacent solutions $N(s)$ to every $s \in S$; and f is a fitness function $f : S \longrightarrow \mathbb{R}$ that provides a

mapping from solution to associated fitness. That fitness can be conceptualised as the solution *height* within the landscape metaphor. The *local optima network* (LON) model [21] was introduced as a tool for studying the connectivity of local optima in a fitness landscape, and has subsequently shown proficiency in helping with explaining metaheuristic search behaviour [1, 6]. In this work, we consider LONs which are sampled using iterated local search. We now formally define this type of LON.

Local optima. We assume a search space S with a fitness function $f(S)$ and a neighbourhood function $N(s)$. A local optimum, which in the QAP is a minimum, is a solution l such that $\forall s \in N(l)$, $f(l) \leq f(s)$.

Monotonic perturbation edges. There is an edge from local optimum l_1 to local optimum l_2, if l_2 can be obtained after applying a random perturbation (in the present study, this is k-exchange) to l_1 followed by local search, and $f(l_2) \leq f(l_1)$. In LON terminology, these are called *escape* edges [38]. The edges are called *monotonic* because they record only non-deteriorating, directed transitions between local optima. Edges are weighted with the frequency of transition. The set of edges is denoted by E.

Compound local optima. A compound local optimum is a set of connected LON nodes with the same fitness value. Two nodes are connected if there is a monotonic perturbation edge between them. The set of connected optima with the same fitness, denoted by CL, corresponds to the set of nodes in the compound monotonic LON model.

Compound monotonic LON. Is the directed network $CMLON = (CL, CE)$, where nodes are compounded local optima CL, and the edges CE are aggregated from the monotonic edge set E by summing up the edge weights.

Monotonic sequence. A monotonic sequence is a path of connected nodes $MS = \{cl_1, cl_2, \ldots, cl_s\}$ where $cl_i \in CL$. By definition of the edges, $f(cl_i) \leq f(cl_{i-1})$. There is a natural end to every monotonic sequence, cl_s, when no improving transitions can be found. This node, cl_s, is called a *sink* as it does not have improving outgoing edges[1] The global optimum will always be a sink, and there can be sub-optimal sinks as well. Note that cl_s can also be a compounded node; that is, the combination of more than one connected nodes of equivalent fitness.

Funnel. A funnel in the CMLON comprises the aggregation of all monotonic sequences ending at the same sink (which can be a compound node) [23] and is a subset of the CMLON. Conceptually, funnels can be viewed as being basins of attraction at the level of local optima. The funnel whose sink is the global optimum can be called the "optimal" funnel (there can be more than one optimal funnel); all other funnels are sub-optimal.

2.2 LON Construction

For a few years after local optima networks were proposed, exhaustive enumeration of the complete network was standard [21, 25, 40]; indeed, some later works also took this approach [23, 28, 32]. In these works, the node set is constructed by mapping every solution to its local optimum through local search. Edges are then defined according one of two approaches: basin transition edges [21], which capture the probability of moving between two basins of attraction under the chosen neighbourhood operator, and escape edges [38]: these connect nodes if the destination local optimum can be reached from the source by carrying out a defined operation (usually perturbation) followed by local search. The nature of exhaustive enumeration necessarily constrains the studied instances to small sizes. In response to this, sampling methodologies have been proposed to construct partial local optima networks; these implement the escape edge model. Algorithms include iterated local search (ILS) [20]. ILS is a metaheuristic which repeatedly combines large and random mutations (*perturbations*, for the purpose of diversification) with local search (intensification); it navigates through the local optima space, and is therefore suited for extracting a LON. Other LON sampling approaches include snowball sampling [39], recombination approaches [8, 41], and local search based methods [1, 7]. Literature has suggested that snowball-sampled LON metrics correlate well with those of a complete LON [39]. For LON sampling by ILS, the number of nodes — and to a lesser degree, the number of edges — were found to be related to the exhaustively-enumerated quantity [35]. The issue with this kind of comparative analysis is that it is necessarily constrained to problem sizes small enough that LONs can be completely enumerated; it is not clear how the perceived correlations scale to larger problem dimension.

In terms of connection between LONs and their predictive potency, authors have shown that LONs sampled using ILS seem to be able to explain search performance better than enumerated LONs [35]. This might be in line with the argument from network sampling literature that sampling algorithms can provide a kind of "positive" bias [14], in that they may encourage the samples to contain representative information. In general, questions surrounding sampling bias and the effect of randomness in LON construction algorithms are still open. Iterated local search based sampling seems to be the most prevalent in the literature [22, 27, 31, 34, 37]; we therefore focus our attention on it for the purposes of this study.

2.3 Randomness and Sampling

Statistical randomness, otherwise known as *pseudorandomness*, is obtained from a pseudorandom number generator [9]. This behaviour does not imply or guarantee true randomness. We consider the effect of statistical randomness in this study, and refer to it with the term "randomness" for the purpose of concision. In statistical language, a partial local optima network is essentially a sample from a larger population: the exhaustively-enumerated sets of nodes and edges. For random samples, it has been argued that 30-50 are sufficient [18]; this aligns with the Central Limit Theorem, which stipulates that from around 30 observations, sample means begin to resemble a normal distribution [12].

Literature has indicated that evolutionary algorithm performance can be affected by the choice of random number generator [3, 11, 16, 17]. It has been argued that it is possible to remove randomness from EAs while maintaining algorithm proficiency [42]. Some authors have proposed a "quasi-random" approach as an alternative to random population instantiation [13]; a quasi-random generator

[1] In directed networks, a node without outgoing edges is called a *sink*.

places points in a space such that the points are maximally distant from each another. The construction of sampled LONs is influenced by randomness in the heuristic algorithms used to identify the nodes and edges. Authors use varying numbers of independent algorithm runs for iterated local search LON construction to try and mitigate the effects of randomness. There seems to be no agreed value: as few as 30-50 runs have been employed [22, 27, 31]; other works use in the region of 100-200 [19, 34–36]; as many as 1000 have also been considered [37]. This lack of agreement is one reason why an assessment for the influence of randomness on LON construction is important.

The random mechanism which is used during LON construction in this work is the pseudorandom number generator, Mersenne Twister (MT) [15]. The quality of a pseudorandom number generator can be defined by its period, p — how many numbers before the generator repeats itself. Mersenne Twister has p=2^{19337}-1. Randomness is found in a few places in the LON construction algorithm. The most consequential of these may be the starting solution, but there is also the order of neighbour exploration in the first-improvement local search and the random perturbations to consider.

3 METHODOLOGY

3.1 Quadratic Assignment Problem

Definition. A solution to the QAP is generally written as a permutation s of the set $\{1, 2, ..., n\}$, where s_i gives the location of item i. Therefore, the search space is of size $n!$. The cost associated with a permutation s is a quadratic function of the distances between the locations, and the flow between the facilities, $f(s) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{s_i s_j}$, where n denotes the number of facilities/locations and $A = \{a_{ij}\}$ and $B = \{b_{ij}\}$ are the distance and flow matrices, respectively.

Instances. We consider instances from the QAPLIB[2] [2] with between 25 and 50 facilities; these are of moderate size, and yet are not always trivial to solve. There are 47 instances in this size range, but we remove the **esc** group (which are real-world QAP occurrences for the testing of sequential circuits in computer hardware): their local optima networks are uninteresting to study because there is a very high degree of LON neutrality [33]. The reduction results in 40 QAPLIB instances for the study, ranging in size between 25 and 50 facilities and locations. Eleven of the instances in this group have not been solved to optimality; for those, we use their best-known fitness as the surrogate global optimum. In the rest of this paper, for simplicity we refer to these as the global optimum.

3.2 LON Metrics

Funnel-like organisation of local optima has been shown to be related to search difficulty for the travelling salesman problem [4], number partitioning problem [23] and also for the QAP [33]. We therefore concentrate our focus towards measurements relating to funnels. Unless otherwise specified, all metrics are computed on the CMLON. To calculate funnel depth measurements, we first compute all finite shortest paths in the CMLON from origin nodes (local optima at the beginning of ILS runs) towards sink nodes (termination points of ILS runs). A sink is the bottom of a funnel. This process is completed separately for optimal and suboptimal sinks. From the resultant sets of shortest paths (which are monotonic in nature) we can extract metrics. For both optimal and suboptimal funnels, we consider the mean depth and the maximum depth of them; that is, the mean and maximum of the shortest paths which reach the end of the funnel from an origin point.

Other LON metrics are included too: the incoming strength to optimal sinks, and the relative size, in nodes, of the optimal funnel(s). The former is calculated as the weighted in-degree to optimal sinks in CMLON — as a proportion of the total weighted incoming degree to *all* sinks. The size of optimal funnel(s) is defined to be the number of nodes which can reach the optimal sinks using a monotonic path; the size is normalised by the total number of nodes. Also considered are the number of global optima and local optima, respectively, in the un-compounded LON; the number of nodes in the CMLON; the mean size of compounded nodes; and the cardinality for optimal and sub-optimal funnels. Abbreviated names for LON features are used in the results section; we therefore provide a list with shortnames and descriptions now, in the order they first appear in a Figure.

(1) *cnodes_size_avg*: the mean size of compounded LON nodes (nodes which are a combination of more than one connected nodes of equal fitness)
(2) *depth_gfunnel_avg*: the mean depth (in edges) of optimal funnels
(3) *depth_gfunnel_max*: the maximum depth (in edges) of optimal funnels
(4) *depth_sofunnel_avg*: the mean depth (in edges) of sub-optimal funnels
(5) *depth_sofunnel_max*: the maximum depth (in edges) of sub-optimal funnels
(6) *num_cnodes*: number of compounded nodes
(7) *num_gfunnel*: number of optimal funnels
(8) *num_global*: number of global optima
(9) *num_nodes*: number of local optima
(10) *num_sofunnel*: number of sub-optimal funnels
(11) *size_gfunnel*: proportional size, in nodes, of optimal funnel(s)
(12) *strength_gfunnel*: proportional strength of optimal funnel(s), measured as weight of incoming edges to the funnel sink(s)

4 EXPERIMENTAL SETUP

Throughout the experiments, a random swap of facilities in the QAP permutation is considered as the neighbourhood structure. All experiments are run on an 8-core MacBook Pro equipped with the Apple M1 chip and using OS X Ventura. We use Stützle's iterated local search (ILS) for both gathering performance data and as the foundation of LON construction [30]. The local search stage uses a first improvement hill-climbing, and the perturbation strength is $k = \frac{N}{2}$. This setting was selected because large perturbation strengths of around 50% of N are generally advantageous in this problem domain [20, 30].

4.1 LON Construction

For each instance, 30 LONs are constructed — with a different random seed provided to the Mersenne Twister (MT) mechanism each

[2] https://coral.ise.lehigh.edu/data-sets/qaplib/qaplib-problem-instances-and-solutions/

time. The C implementation of MT is used. For each instance and seed pair, 100 runs of ILS are amalgamated to form the LON. Only local optima with equal or improving fitness are accepted. Worsening local optima are never accepted. Runs terminate after 10 000 ILS iterations with no improvement. This termination condition has been used in the literature to estimate sinks and funnel structure [24].

4.2 Algorithm Performance

We consider runtime over 100 independent executions (with different random seeds from those used for the LON construction runs) to summarise ILS performance on the instances. Each of the three ILS variants have the same seeds provided to them. Runs terminate when *either* the known best fitness is found or after 10 000 ILS iterations with no improvement. To give an indication of how many iterations this may result in: the instance **tai50a** was associated with 12 178 iterations in an example run. Performance metrics were computed using three distinct local optima acceptance strategies for the ILS, detailed now.

Better or equal. Only local optima which have improved or equal fitness to the current local optimum are accepted. Worsening local optima are never accepted.

Annealing. Improving and equal local optima are always accepted. Worsening local optima are accepted according to probability. This is governed by an annealing-like cooling schedule termed *Large Step Markov Chain* (LSMC) in the work proposing ILS for QAP. We set the LSMC parameters to mirror those used in Stützle's work [30]: that is, an initial temperature of 2.5% of the current solution fitness — a solution which is 2.5% lower fitness is allowed with the probability $\frac{1}{e}$; the number of steps at a temperature is 10. Temperature is reduced according to a scaling factor, $\alpha = 0.9$.

Restart. Improving and equal local optima are always accepted. If iterations since an improvement in local optima quality have exceeded $3N$ (where N is the problem dimension), a restart happens.

4.3 Regression Model Setup

LON metrics serve as the independent variables in the algorithm performance models. The dependent variable is runtime of iterated local search; the modelling is therefore a regression task. As the learning algorithm, Random Forest in R is used. For each individual model, the observations include metrics from LONs constructed using one of the 30 random seeds, combined with ILS runtime for the problem instance as the dependent variable. To be clear, the same ILS performance data is used regardless of the LON random seed: for example, the runtime value for *bur26a* is mapped to all 30 of its LONs. Because we use one row per QAP instance to build the models, there are limited number of observations in the datasets: 40. The *one-in-ten* rule [5] stipulates that roughly ten observations are required per independent variable. Owing to data splitting during cross-validation, our training sets can be as small as 32. We compute 12 LON metrics and then conduct recursive feature elimination to obtain the top three features. The feature selection bases its decisions on the minimisation of RMSE and executes 100 bootstrap replications. One hundred bootstraps is a quantity backed up by literature [26]. Feature selection is conducted separately for each of the 30 set of LONs (one set per seed), and the elite features stored. For the final modelling stage, features which appeared the most at the top three positions were included. Each feature selection process includes 100 bootstrapping iterations as cross-validation.

For both feature selection and final modelling, default hyperparameters for the Random Forests are employed; these are: 500 trees, $\frac{np}{3}$ variables sampled [where np is the number of predictors], subsampling of observations with replacement, and a sample size of n [where n is the number of the whole sample]. We conduct Monte Carlo cross-validation (random repeated sub-sampling) to estimate model quality metrics. This is done with an 80-20 training-validation split and for 100 iterations; we consider the R^2 and root mean squared error (RMSE) as model metrics. These are computed on the validation sets and averaged over 100 cross-validation iterations.

5 RESULTS

5.1 Network Comparison

Figure 1 shows, for each QAP instance, the distribution of LON features across different random seeds provided to the LON construction process. There are 30 seeds for each instance. Each horizontal line on a box is for a particular feature, as indicated on the left of the overall Figure. Recall the descriptions for features from Section 3.2. Colours represent the normalised value of the feature: dark is closer to zero, with yellow closer to one. Instance names are the labels for individual boxes.

Notice from the Figure that particular features, such as the metricsx
size_gfunnel and *strength_gfunnel*, are often consistent across seeds. This can be observed with the solid blocks of either yellow or dark blue at the bottom of several of the instance boxes. Other features which display little variation for many of the instances include *size_cnodes_avg*, *num_global*, and *num_sofunnel*. There are, however, features which show less stability across seeds. We notice that metrics relating to funnel depth, particularly *depth_gfunnel_avg* and *depth_gfunnel_max*, often depend on the seed. This can be seen by checking for the rows labelled with those metric names and observing the variation in colour there.

With all that said, there seems to be no universal rule about which metrics are dependable across seeds. Even *size_gfunnel* and *strength_gfunnel* display lack of uniformity in some instances (for example, **kra30a** and **kra30b**); we therefore stipulate that stability of metrics depends on the problem instance.

Figure 2 presents distributions of LON metrics across seeds as parallel coordinate plots. Notice that the x-axis now measures the normalised value for features. Each line traces feature values for a particular seed. Colour is used to identify an instance as belonging to an instance group, as labelled in the legend. Surveying the Figure, we can see that the eight instances comprising the **bur** category are each associated with LONs which are remarkably constant across different seeds. This can be seen by considering the tightness of the line groups in these instances when compared to instances from other instance types. In the **tainna** category, there is a lot of variation in metrics relating to depth of the optimal funnel(s): *depth_gfunnel_avg* and *depth_gfunnel_max*. The number of local optima, as well as features which capture neutrality at the local

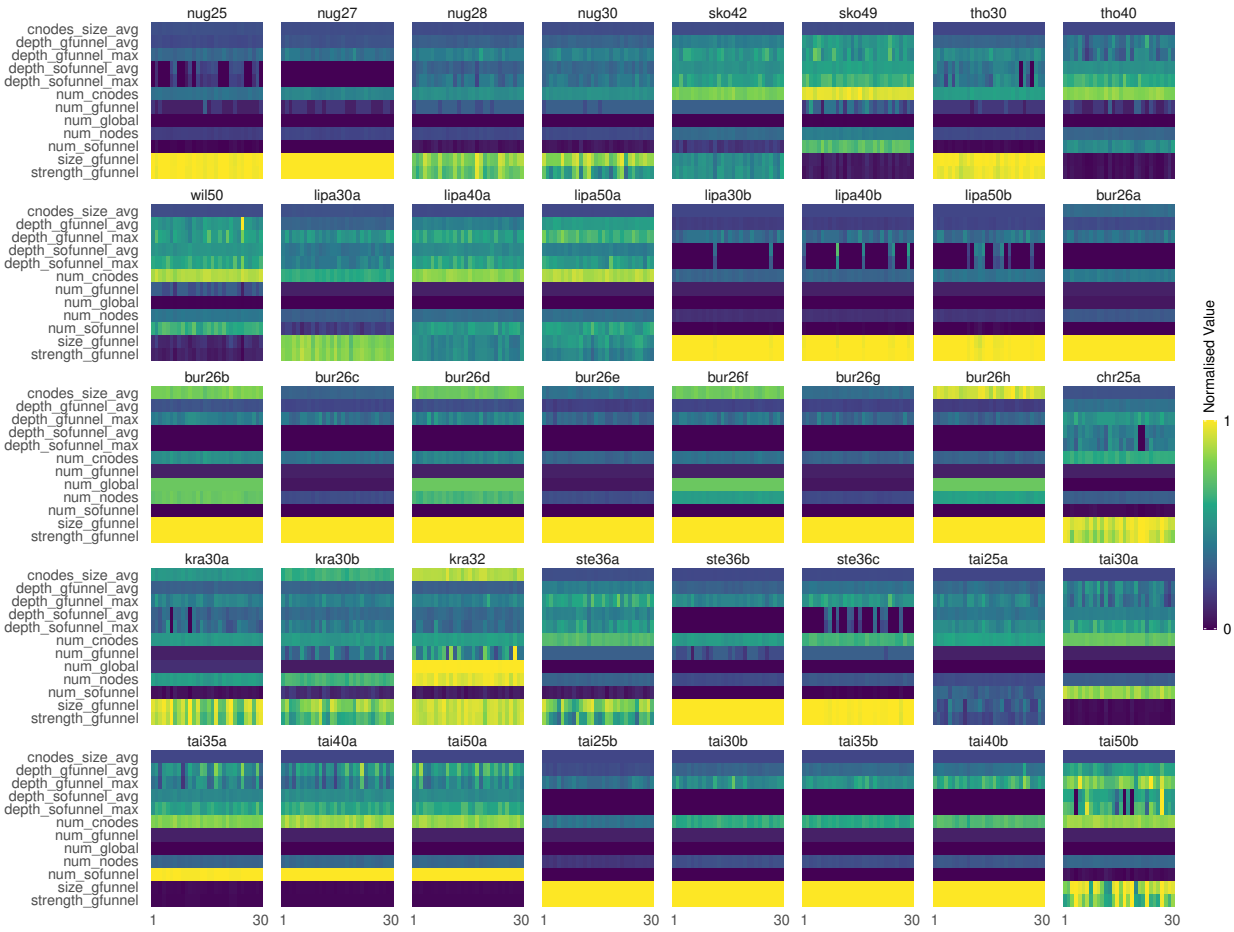

Figure 1: For each instance, the distribution of LON metrics across different random seeds provided to LON construction

optima level — *cnodes_size_avg* and *num_cnodes* — show stability across seeds almost universally for problem instances (note the lowest part of the individual boxes). The **tainnb** category each display the same "skeleton" of feature patterns. The first four instances have extremely similar plots, while **tai50b** shows more diversity of funnel depth metrics across different seeds.

To take a closer look into possible differences between QAP instance groups, we show the distributions for selected features as boxplots in Figures 3 and 4. The diagrams represent the features *num_nodes* and *depth_gfunnel_avg*, respectively. Those two were selected because one of them displays relative consistency across groups (and across random seeds), while the other shows different behaviour: the amount of variation across seeds seems to be somewhat related to instance group. Due to space constraints, we cannot show these plots for every feature.

Figure 3 tells us that the number of nodes (local optima) captured by LON sampling is usually quite consistent even with differing random seeds. We can see this from the narrow distributions throughout most of the plot. These are present in every instance group, although some instances within **real** have more local optima and slightly more varied distributions. Turning our attention to Figure 4, the pattern is unlike that of Figure 3. Here, there are several distributions which exhibit high variation. In some cases, this seems like it might be related to instance group: compare **grid** and **tainna** to the other categories. For those two, the sampled depth of optimal funnel is often highly diverse depending on seed. For the **real** instances, however, the depth is quite stable across seeds.

5.2 Feature Selection

For feature selection, the set of observations used to build a model comprise metrics computed on a set of 40 LONs — one per QAP instance — constructed using a particular random seed. There are therefore sets 30 models, differentiated by the seed provided to the LONs. Each of the three ILS variants serve as a separate response variable, so each has an associated group of 30 model configurations. We investigate in this Section how consistent the top-selected feature was for these models across the different random seeds. Figure 5, presents the number of times particular features ranked first place for the three response variables: ILS with default, annealing-like, and restart acceptance. Notice that some features from the full set (described in Section 3.2) do not appear in Figure 5; this means that they did not appear as the top feature in any of the models.

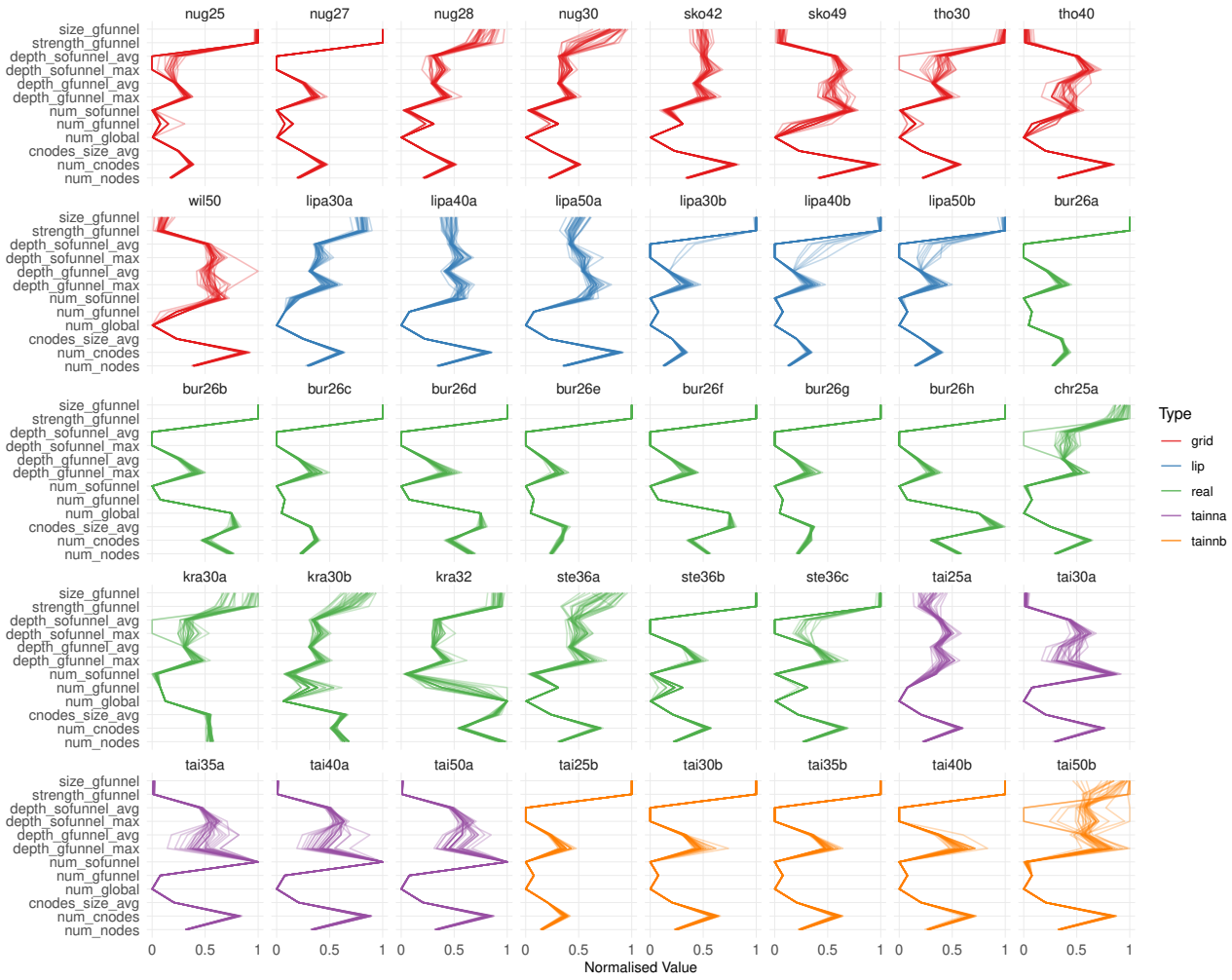

Figure 2: For each instance, the parallel coordinate representation of the distribution of LON metrics across different random seeds provided to LON construction

Comparing the distributions in the three sub-plots, we can see that the top feature for the *default* and *annealing* ILS variants was rather consistent across different LON seeds. In the case of *default*, the *strength_gfunnel* feature ranked as first for 17 of the 30 possible seeds (observe the lowest bar of the left-most plot). The most-common top feature for *annealing* ILS, *depth_gfunnel_avg*, is even more dominant across seeds: around 22 out of 30. With *restart* ILS as response variable, it is a different story: a few different features have several occurrences, but none prevail as being distinctly the most salient. These results suggest that when it pertains to explaining and predicting *default* and *annealing* ILS, the LON features which are most potent are consistent across random seeds provided to the LON construction algorithm. This does not appear to be the case for *restart* ILS, however.

5.3 Modelling

The final models contain the top three selected features for each ILS variant. Different LON seeds sometimes resulted in different features ranking as the best. We took a pooling approach to feature choice for the final models: selecting the features which most often ranked at places one, two, and three across the 30 seeds. For default acceptance, these were *strength_gfunnel*, *size_gfunnel*, and *depth_sofunnel_avg*; in the case of annealing ILS, they were *depth_gfunnel_avg*, *num_cnodes*, and *depth_sofunnel_avg*; and for restart acceptance: *strength_gfunnel*, *depth_gfunnel_avg*, and *size_gfunnel*. Figures 6 and 7 present the distributions across 30 LON seeds of model R^2 and RMSE, respectively, on validation sets — averaged over 100 Monte Carlo cross-validation iterations.

One outlier has been removed for visual clarity: a model in the *restart* set which had an R^2 of -13.44. For random forest regression, a negative R^2 like this can occur when the model prediction is less accurate than simply predicting the mean of the response

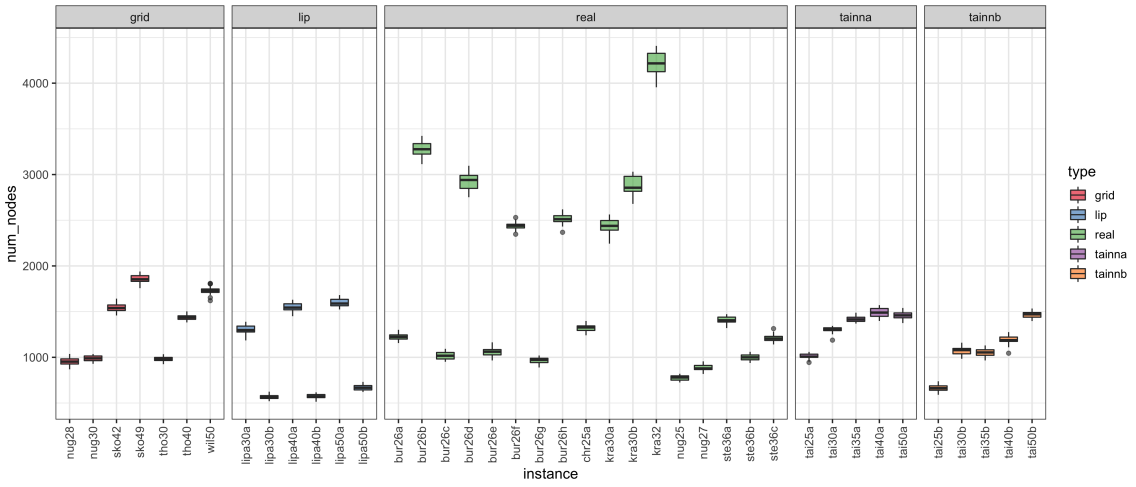

Figure 3: For each instance, the distribution of the number of local optima in the LON across different random seeds provided to LON construction

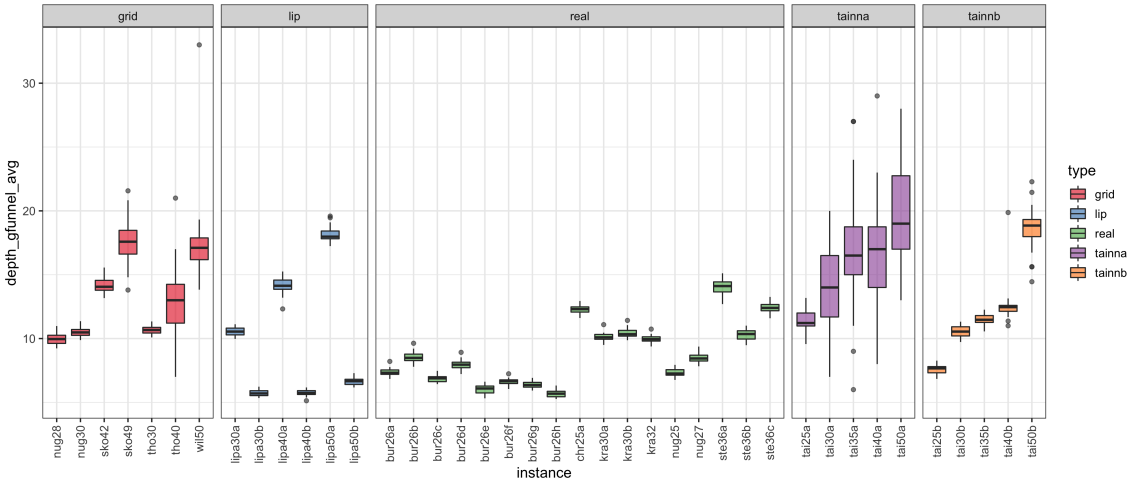

Figure 4: For each instance, the distribution of the average depth of optimal funnel(s) in the LON across different random seeds provided to LON construction

variable. Looking at Figure 6 and comparing the three boxes, it can be observed that models with *default* ILS as response variable have the narrowest range of R^2 across seeds. Even so, there are values as far apart as ~0.5 and ~0.97. This is a substantial difference. The ranges for *annealing* and *restart* are even more vast; this is particularly the case for the latter. Depending on the random seed provided to the LON construction algorithm, the subsequent model R^2 can vary from below zero — indicating a model which is worse than predicting the mean response variable value — to ~0.96 (almost all variance explained). Recalling the variation of top feature across seeds during the feature selection process (Figure 5, it is possible that the low R^2 values arise because the most powerful metrics of a LON constructed with a specific seed were not used (we used the same three features to build models regardless of seed).

Figure 7 contains RMSE values. RMSE is in the same units as the response variable, which is ILS runtime, measured in iterations. The Figure indicates that the lowest RMSE values, and indeed the narrowest distribution, arises from models with *default* ILS as the response variable. This communicates that these models are the strongest, and also that the dispersion of model error is relatively low across LON seeds. The *annealing* and *restart* models have larger errors and wider distributions. The consistency of model error depending on LON seeds seems quite low.

5.3.1 Algorithm Configuration. We now report an algorithm configuration decision experiment. The intention was to investigate whether different recommendations might be made depending on the seed provided to LON construction.

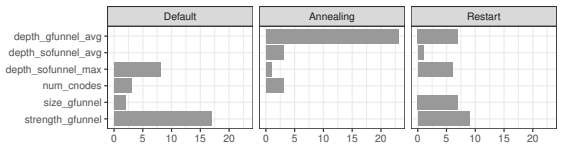

Figure 5: Distribution across 30 seeds of the top feature selected with each ILS variant as response variable

For all three ILS variants, we set their predictor features to be the best three for the default ILS. This decision was made for simplicity and consistency. Thereafter, we obtain the predicted runtimes for each QAP instance and for the three algorithm configurations. These are produced by executing 100 iterations of Monte Carlo cross-validation, with the relevant instance always excluded from the training set and included in the validation. The models predict the runtime. Then each instance is mapped to the ILS variant with associated with the lowest predicted runtime.

In Table 1 we report statistics describing the consistency of algorithm configuration recommended across different seeds: the column labelled *#config.* contains values between 0 and 1 — the number of recommended algorithm configurations as a proportion

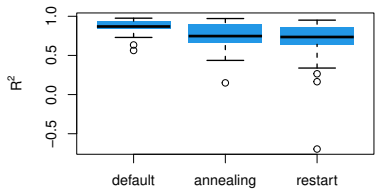

Figure 6: Distribution across 30 LON seeds of model R^2 on validation data (mean over 100 cross-validation iterations) with three ILS variants as the response variable. Predictors are the top three LON metrics obtained from feature selection

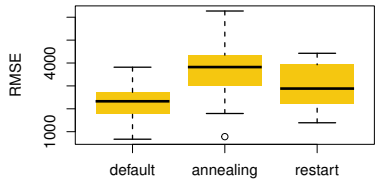

Figure 7: Distribution across 30 LON seeds of model *RMSE* on validation data (mean over 100 cross-validation iterations) with three ILS variants as the response variable. Predictors are the top three LON metrics obtained from feature selection

of the available number (three). In column *freq. prevailing config.* the proportions are out of 30, and represent for how many seeds the most prevalent algorithm configuration was recommended.

Table 1: Algorithm configuration distributions: *#config.* captures how many different ILS variants were recommended for the instance, depending on LON seed (as a proportion of three total variants); *freq. prevailing config.* is how often the most common recommendation for an instance appears (presented as a proportion of 30 models)

statistic	#config.	freq. prevailing config.
minimum	0.667	0.500
1st quartile	0.667	0.567
median	0.667	0.633
mean	0.825	0.700
3rd quartile	1.000	0.833
maximum	1.000	0.967

Let us consider the *#config.* column initially. The minimum proportion is 0.667, which means that at least two algorithm configurations were recommended for each instance. Both the 3rd quartile and the maximum have proportional values of 1, which tells us that some instances had all three ILS variants recommended. In the *freq. prevailing config.* column, notice that the minimum proportional representation of the dominant recommended configuration is 0.5. The maximum is 0.967, which represents 29 out of the 30 seeds agreeing on the best algorithm variant. A mean and median of 0.700 and 0.633 respectively indicate that, for most instances, the dominant variant is recommended for a reasonable majority of LON seeds.

6 CONCLUSIONS

We have conducted a preliminary study on statistical randomness in a local optima network (LON) construction algorithm: iterated local search (ILS). LONs were extracted for a set of quadratic assignment problem (QAP) instances and with a range of different random seeds provided to the construction algorithm. Metrics were computed on the obtained LONs. These were analysed with the aid of visualisation. Algorithm performance prediction models were built using the metrics as features and performance of ILS as the response variable. The results showed that some LON metrics seem consistent across seeds, while others vary substantially. Additionally, the quality of algorithm performance models using LON metrics as predictors can differ depending on randomness. Finally, LON metrics associated with separate seeds can lead to different algorithm configuration recommendations for the same instance. For future work, we will consider a larger QAP instance set. In particular, an instance generator (such as [10]) could be used to obtain an expansive set of similar problems. This should remove the variation seen between instances from different classes. We would also like to further remove noise from the performance modelling pipeline; consider the study of other problems, including pseudo-boolean spaces; and include other heuristics or alternative LON sampling methods such as snowballing [39] and those relating to recombination [8, 41]. The data from this work are available at https://doi.org/10.5281/zenodo.7900303.

REFERENCES

[1] Wojciech Bożejko, Andrzej Gnatowski, Teodor Niżyński, Michael Affenzeller, and Andreas Beham. 2018. Local optima networks in solving algorithm selection problem for tsp. In *International Conference on Dependability and Complex Systems*. Springer, 83–93.

[2] Rainer E. Burkard, Stefan E. Karisch, and Franz Rendl. 1997. QAPLIB – A Quadratic Assignment Problem Library. *Journal of Global Optimization* 10, 4 (1997), 391–403.

[3] Miguel Cárdenas-Montes, Miguel A Vega-Rodríguez, and Antonio Gómez-Iglesias. 2011. Sensitiveness of evolutionary algorithms to the random number generator. In *International Conference on Adaptive and Natural Computing Algorithms*. Springer, 371–380.

[4] D R Hains, L D Whitley, and A E Howe. 2011. Revisiting the big valley search space structure in the TSP. *Journal of the Operational Research Society* 62, 2 (2011), 305–312.

[5] Frank E Harrell Jr, Kerry L Lee, Robert M Califf, David B Pryor, and Robert A Rosati. 1984. Regression modelling strategies for improved prognostic prediction. *Statistics in medicine* 3, 2 (1984), 143–152.

[6] Sebastian Herrmann, Gabriela Ochoa, and Franz Rothlauf. 2016. Communities of local optima as funnels in fitness landscapes. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference*. ACM, 325–331.

[7] David Iclanzan, Fabio Daolio, and Marco Tomassini. 2014. Data-driven local optima network characterization of QAPLIB instances. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*. 453–460.

[8] Melike D Karatas, Ozgur E Akman, and Jonathan E Fieldsend. 2021. Towards population-based fitness landscape analysis using local optima networks. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 1674–1682.

[9] Maurice G Kendall and B Babington Smith. 1938. Randomness and random sampling numbers. *Journal of the royal Statistical Society* 101, 1 (1938), 147–166.

[10] Joshua Knowles and David Corne. 2003. Instance generators and test suites for the multiobjective quadratic assignment problem. In *Evolutionary Multi-Criterion Optimization: Second International Conference, EMO 2003, Faro, Portugal, April 8–11, 2003. Proceedings 2*. Springer, 295–310.

[11] Pavel Krömer, Václav Snáel, and Ivan Zelinka. 2013. Randomness and chaos in genetic algorithms and differential evolution. In *2013 5th International Conference on Intelligent Networking and Collaborative Systems*. IEEE, 196–201.

[12] Sang Gyu Kwak and Jong Hae Kim. 2017. Central limit theorem: the cornerstone of modern statistics. *Korean journal of anesthesiology* 70, 2 (2017), 144–156.

[13] Heikki Maaranen, Kaisa Miettinen, and Marko M Mäkelä. 2004. Quasi-random initial population for genetic algorithms. *Computers & Mathematics with Applications* 47, 12 (2004), 1885–1895.

[14] Arun S Maiya and Tanya Y Berger-Wolf. 2011. Benefits of bias: Towards better characterization of network sampling. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 105–113.

[15] Makoto Matsumoto and Takuji Nishimura. 1998. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation (TOMACS)* 8, 1 (1998), 3–30.

[16] Markus Maucher, Uwe Schöning, and Hans A Kestler. 2011. Search heuristics and the influence of non-perfect randomness: examining Genetic Algorithms and Simulated Annealing. *Computational Statistics* 26, 2 (2011), 303–319.

[17] Mark M Meysenburg and James A Foster. 1999. Randomness and GA performance, revisited. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 1*. 425–432.

[18] Christopher Z Mooney, Christopher F Mooney, Christopher L Mooney, Robert D Duval, and Robert Duvall. 1993. *Bootstrapping: A nonparametric approach to statistical inference*. Number 95. sage.

[19] Gabriela Ochoa and Francisco Chicano. 2019. Local optima network analysis for MAX-SAT. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 1430–1437.

[20] Gabriela Ochoa and Sebastian Herrmann. 2018. Perturbation strength and the global structure of QAP fitness landscapes. In *International Conference on Parallel Problem Solving from Nature*. Springer, 245–256.

[21] Gabriela Ochoa, Marco Tomassini, Sebástien Vérel, and Christian Darabos. 2008. A study of NK landscapes' basins and local optima networks. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*. 555–562.

[22] Gabriela Ochoa and Nadarajen Veerapen. 2022. Neural Architecture Search: A Visual Analysis. In *International Conference on Parallel Problem Solving from Nature*. Springer, 603–615.

[23] Gabriela Ochoa, Nadarajen Veerapen, Fabio Daolio, and Marco Tomassini. 2017. Understanding phase transitions with local optima networks: number partitioning as a case study. In *European Conference on Evolutionary Computation in Combinatorial Optimization*. Springer, 233–248.

[24] Gabriela Ochoa, Nadarajen Veerapen, Darrell Whitley, and Edmund K Burke. 2016. The multi-funnel structure of TSP fitness landscapes: a visual exploration. In *Artificial Evolution: 12th International Conference, Evolution Artificielle, EA 2015, Lyon, France, October 26-28, 2015. Revised Selected Papers 12*. Springer, 1–13.

[25] Gabriela Ochoa, Sébastien Verel, Fabio Daolio, and Marco Tomassini. 2014. Local optima networks: A new model of combinatorial fitness landscapes. In *Recent advances in the theory and application of fitness landscapes*. Springer, 233–262.

[26] Nicholas D Pattengale, Masoud Alipour, Olaf RP Bininda-Emonds, Bernard ME Moret, and Alexandros Stamatakis. 2010. How many bootstrap replicates are necessary? *Journal of computational biology* 17, 3 (2010), 337–354.

[27] Lucas Marcondes Pavelski, Marie-Éléonore Kessaci, and Myriam Delgado. 2021. Local Optima Network Sampling for Permutation Flowshop. In *2021 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 1131–1138.

[28] Isak Potgieter, Christopher W Cleghorn, and Anna S Bosman. 2022. A Local Optima Network Analysis of the Feedforward Neural Architecture Space. In *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[29] Peter F. Stadler. 2002. Fitness landscapes. *Biological Evolution and Statistical Physics. Lecture Notes in Physics* 585 (2002), 183–204.

[30] Thomas Stützle. 2006. Iterated local search for the quadratic assignment problem. *European Journal of Operational Research* 174, 3 (2006), 1519–1539.

[31] Sara Tari and Gabriela Ochoa. 2021. Local search pivoting rules and the landscape global structure. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 278–286.

[32] Sarah L Thomson, Fabio Daolio, and Gabriela Ochoa. 2017. Comparing communities of optima with funnels in combinatorial fitness landscapes. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 377–384.

[33] Sarah L Thomson and Gabriela Ochoa. 2022. On funnel depths and acceptance criteria in stochastic local search. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 287–295.

[34] Sarah L Thomson, Gabriela Ochoa, and Sébastien Verel. 2020. The fractal geometry of fitness landscapes at the local optima level. *Natural Computing* (2020), 1–17.

[35] Sarah L Thomson, Gabriela Ochoa, Sébastien Verel, and Nadarajen Veerapen. 2020. Inferring future landscapes: sampling the local optima level. *Evolutionary computation* 28, 4 (2020), 621–641.

[36] German Treimun-Costa, Elizabeth Montero, Gabriela Ochoa, and Nicolás Rojas-Morales. 2020. Modelling parameter configuration spaces with local optima networks. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*. 751–759.

[37] Nadarajen Veerapen, Fabio Daolio, and Gabriela Ochoa. 2017. Modelling genetic improvement landscapes with local optima networks. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 1543–1548.

[38] Sébastien Verel, Fabio Daolio, Gabriela Ochoa, and Marco Tomassini. 2012. Local optima networks with escape edges. In *Artificial Evolution: 10th International Conference, Evolution Artificielle, EA 2011, Angers, France, October 24-26, 2011, Revised Selected Papers 10*. Springer, 49–60.

[39] Sébastien Verel, Fabio Daolio, Gabriela Ochoa, and Marco Tomassini. 2018. Sampling local optima networks of large combinatorial search spaces: The qap case. In *International Conference on Parallel Problem Solving from Nature*. Springer, 257–268.

[40] Sébastien Verel, Gabriela Ochoa, and Marco Tomassini. 2010. Local optima networks of NK landscapes with neutrality. *IEEE Transactions on Evolutionary Computation* 15, 6 (2010), 783–797.

[41] Darrell Whitley and Gabriela Ochoa. 2022. Local optima organize into lattices under recombination: an example using the traveling salesman problem. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 757–765.

[42] Ivan Zelinka, Mohammed Chadli, Donald Davendra, Roman Senkerik, Michal Pluhacek, and Jouni Lampinen. 2013. Do evolutionary algorithms indeed require random numbers? Extended study. In *Nostradamus 2013: Prediction, Modeling and Analysis of Complex Systems*. Springer, 61–75.