How Partition Crossover Exposes Parallel Lattices and the Fractal Structure of k-Bounded Functions

Darrell Whitley
whitley@colostate.edu
Colorado State University
Fort Collins, Colorado, USA

Gabriela Ochoa
gabriela.ochoa@stir.ac.uk
University of Stirling
Stirling, Scotland, UK

Francisco Chicano
chicano@uma.es
ITIS Software, University of Malaga
Malaga, Spain

ABSTRACT

A combination of recombination and local search can expose the existence of an exponential number of parallel lattices that span the search space for all classes of k-bounded pseudo-Boolean functions, including MAX-kSAT problems. These "parallel" lattices sometimes have identical evaluations shifted by a constant. We use Partition Crossover to aid in the discovery of lattices, which are sets of 2^q possible offspring from recombination events, organized into q-dimensional hypercubes, where q is the number of recombining components given two parents. Finally, we show that recursively embedded subspace lattices display a fractal structure, which can be captured using rewrite rules based on a Lindenmayer system that accurately model how local optima are distributed across different size lattices.

CCS CONCEPTS

• **Mathematics of computing → Combinatorial optimization**; • **Computing methodologies → Genetic algorithms**; • **Theory of computation → Scheduling algorithms**.

KEYWORDS

partition crossover, lattices, genetic algorithms, combinatorial optimization

ACM Reference Format:
Darrell Whitley, Gabriela Ochoa, and Francisco Chicano. 2025. How Partition Crossover Exposes Parallel Lattices and the Fractal Structure of k-Bounded Functions. In *Genetic and Evolutionary Computation Conference (GECCO '25), July 14–18, 2025, Malaga, Spain.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3712256.3726340

1 INTRODUCTION

Understanding how recombination induces neighborhoods and landscapes in search spaces has long been a challenging problem in evolutionary computation and related fields. For example, the pioneering work of Forrest and Jones [13] used fitness distance correlation as a measure of problem difficulty. But fitness distance correlation is most sensitive to more traditional definitions of neighborhood such as the bit-flip Hamming neighborhood. Metrics based on recombination neighborhoods have been more elusive.

This paper builds on the idea that recombination induces subspace hypercube "lattices" in the search space. A lattice corresponds to a set of 2^q possible offspring from one recombination event, where q is the number of recombining components, organized into a hypercube. Recombination lattices were first described in association with the Travelling Salesman Problem [28]. However, lattices also occur in k-bounded pseudo-Boolean functions, such as MAX-kSAT [12, 21], NK-Landscapes [15, 16] and spin glass functions [19]. It is important to emphasize that the existence of lattices does not depend on recombination, but lattices are efficiently found using Partition Crossover for some NP-hard problems. And while it may be less efficient, lattices can also be discovered using 1-point or 2-point crossover operators, particularly in a Gray-box context where the nonlinear interactions between variables are known [23].

We present new results that show that many lattices belong to a family of "parallel" lattices. It is possible to construct different types of parallel lattices, but for the simplest constructions, we prove two results. First, a single lattice can be associated with an exponential number of parallel lattices. An initial lattice serves as the "seed" lattice. A closure operator can then generate members in a family of parallel lattices. Second, every offspring (e.g. string) which is a member of a "perfect" parallel lattice closure can be evaluated using exactly the same linear equation (shifted by a constant) that was used to evaluate the offspring in the original seed lattice. An exponential number of solutions in the search space will often be related by membership in the same family of parallel lattices and all of these solutions can be evaluated by a single linear equation. This result holds for classic NP-hard problems such as MAX-kSAT.

Partition Crossover uses a Variable Interaction Graph (VIG) to efficiently find recombining components that can yield a lattice of offspring. It constructs a Recombination Graph from the VIG so as to remove nonlinear interactions between recombining components in the recombination graph. However, when constructing parallel lattices, it is necessary to also consider the *complement* of the recombination graph, which includes all of the vertices of the VIG that are dropped out of the recombination graph.

How can these results improve search algorithms for k-bounded pseudo-Boolean functions? Partition Crossover can "tunnel" between local optima in $O(n)$ time, potentially finding improved solutions as well as previously unknown local optima. In special cases, Partition Crossover can be executed in $O(1)$ time, particularly for MAX-kSAT problems. But the existence of parallel lattices provides new ways that recombination and local search can be used together. Recently, Canonne and Derbel [6, 7] have exploited improving moves that are revealed after Partition Crossover is applied. They show that these improving moves can be used to improve the DRILS algorithm (Deterministic Recombination with Iterated Local Search) which uses both Partition Crossover and local search [9]. We prove that different solutions in parallel lattices can share exactly the same improving moves, and sometimes yield exactly the same change in evaluation, because they appear in parallel lattices that can be evaluated using the same basic linear equation. There

GECCO '25, July 14–18, 2025, Málaga, Spainl

ACM ISBN 979-8-4007-1465-8/2025/07
https://doi.org/10.1145/3712256.3726340

are $O(1)$ methods to find improving moves in individual solutions [27] but these may not always be efficient for finding improving moves in a lattice that might be exponential in size. However, the location of improving moves in parallel lattices is not random, and in some cases the location of new improving moves can be predicted or even guaranteed. We can also prove that certain bit flips can *never* yield an improving move after Partition Crossover.

Finally, we also show that recursively embedded subspace lattices display a fractal structure, which can be captured by using a rewrite system based on a Lindenmayer system. This result give us a way to model the distribution of the number of local optima in lattices.

2 PARTITION CROSSOVER, LATTICES AND LINEAR EQUATIONS

Developing an intuitive understanding of parallel lattices becomes easier by considering a class of simple k-bounded pseudo-Boolean functions: adjacent NK-Landscapes [14, 15, 25]. This is because nonlinear interactions are adjacent, and recombining components are also adjacent. However, our theoretical results hold for all k-bounded pseudo-Boolean functions.

All k-bounded pseudo-Boolean functions have the form

$$f(x) = \sum_{i=1}^{m} f_i(x)$$

where each subfunction $f_i(x)$ is computed over exactly k variables in Boolean string x [5]. The total number of variables is n. The output of subfunction $f_i(x)$ can be any real-valued number. There are also k-bounded Boolean functions (e.g. MAX-kSAT problems).

For an adjacent NK Landscape with k=3:

$$f(x) = \sum_{i=1}^{n} f_i(x_i, x_{i+1}, x_{i+2})$$

where addition is mod n on the index (e.g. $n + 1 = 1$). We will need to construct the Variable Interaction Graph (VIG) for these functions. The vertices of a VIG are the variables, and there is an edge in the VIG between variables if there is a nonlinear interaction between those variables. All Adjacent NK Landscapes of size N (where $N = n$) have exactly the same *heuristic* Variable Interaction Graph. A *heuristic* VIG assumes that two variables which appear together in the same subfunction will have a nonlinear interaction. But nonlinear interactions in different subfunctions can also cancel, particularly in the case of MAX-kSAT functions. The exact VIG can be found by computing a Walsh polynomial, which has $O(n)$ complexity since the Walsh transform can be applied to each subfunction in constant time, assuming that $m = O(n)$.

We start by using Partition Crossover to find a lattice. When applying Partition Crossover, we start with two parents, P1 and P2.

Without loss of generality, we can assume that exclusive-or is used to remap the search space so that parent P2 is the string of all zeros, 0^n. (This remapping is not used in practice, of course). Parent 1 is a mixture of 1s and 0s. However, all Boolean variables with shared assignments are 0 in both parents. We first remove all variables with shared assignment 0 from the Variable Interaction Graph to produce a Recombination Graph. For Partition Crossover to be successful, the Recombination Graph must break into two or more subgraphs. This "shattering" of the Recombination Graph also decomposes the nonlinear interactions between variables.

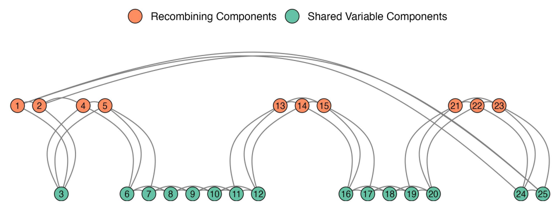

Figure 1: Variable interaction graph for $P1$ and $P2$, where $P1$ = 1101100000001110000011100 and P2 = 0^{25}. This new representation of the VIG shows both the Recombining Graph as well as the Shared Variable Graph. This graph is "shattered" by cutting all edges between the Recombination Graph (above) and the Shared Variable Graph (below). The Shared Variable Graph also decomposes into separable components.

In an NK-Landscape we might observe the following bit strings in parents P1 and P2 (bits are in groups of 5 to aid the reader).

$$P1 = 11011\ 00000\ 00111\ 00000\ 11100$$
$$P2 = 00000\ 00000\ 00000\ 00000\ 00000$$

An example of the heuristic VIG for this Adjacent NK-Landscape with N=25 is shown in Figure 1. In an Adjacent NK-Landscape with $K = 2$ (with $k = K+1$ variables per subfunction), the variable x_i only has potential interactions with $x_{i-2}, x_{i-1}, x_{i+1}, x_{i+2}$. Thus, we easily see the valid recombining components in Figure 1. Variables are in different recombining components if the variables are separated by K or more zero bits. We denote a recombining component by RC_i. For Figure 1, numbering variables left to right, starting at 1, the recombining components are:

$$RC_1 = \{x_1, x_2, x_4, x_5\},\ RC_2 = \{x_{13}, x_{14}, x_{15}\},\ RC_3 = \{x_{21}, x_{22}, x_{23}\}$$

But, we can also decompose the variables with shared bit assignments. Let SC_i denote a component of **shared bits.** Like recombining components, a "shared component" SC_i has no direct nonlinear interaction with a different shared component SC_j in the Shared Variable Graph. This also limits nonlinear interactions between different improving moves in different shared components. Thus, improving moves are often also "localized" in scope.

The shared components in our example are

$$SC_1 = \{x_3\},\ SC_2 = \{x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}\}$$

$$SC_3 = \{x_{16}, x_{17}, x_{18}, x_{19}, x_{20}\},\ SC_4 = \{x_{24}, x_{25}\}$$

Note that shared variable x_3 only has nonlinear interactions with the variables in the recombining component RC_1.

But we can also make another observation: the shared variables x_8, x_9, x_{10} as well as variable x_{18} have no nonlinear interactions with *any* of the recombining components. Thus, variables x_8, x_9, x_{10} and x_{18} cannot be the source of an improving move after recombination. Furthermore, the variables x_8, x_9, x_{10} only have nonlinear interactions with other variables in shared component SC_2.

Tinos et al. [24] prove that improving moves can only be found in the Shared Variable Graph. But this observation can be made more precise: only shared variables that have an edge connected to at least one recombining component can become an improving move after recombination.

Definition: Unaffiliated Variable. An "unaffiliated variable" is a variable in a Shared Component that has no connection to any variable which appears in a Recombination Component.

Unaffiliated variables play a key role in proving that there exists one class of **identical parallel lattices** (shifted by a constant) which hold over similar but distinct sets of variable assignments.

2.1 A Linear Equation for Lattices

We denote the number of recombining components by q. Next, let C_i^1 denote an offspring produced by recombining parents P1 and P2 such that C_i^1 inherits all 1 bits from parent P1 only in one recombining component RC_i. Thus, both C_i^1 and RC_i are defined from $i = 1$ to q. For each recombining component RC_i we define components of a weight vector α as follows:

$$\alpha_i = f(C_i^1) - f(P2)$$

where α_i denotes the change in evaluation when the bits in the i^{th} recombining component are inherited from parent P1.

Let b denote an auxiliary binary string of length q. Let $b_i = 0$ denote that the child inherits the bits from Parent 2 in the i^{th} recombining component. By symmetry, $b_i = 1$ denotes that the child inherits the bits from Parent 1 in the i^{th} recombining component. In effect, b is a recombination mask over the set of q recombining components. Inheritance of all recombining components from P1 is represented by $b = 1^q$ and inheritance from P2 is represented by $b = 0^q$. The child $C_{i=1}^1$ has the recombination mask $b = 10^{q-1}$. Whitley et al. [29] prove the following theorem. We offer a new more concise proof.

Theorem 2.1. *Under Partition Crossover, all of the 2^q children of parents P1 and P2 can be evaluated using the following linear equation and the auxiliary binary recombination mask b:*

$$f(x) = \alpha_0 + \sum_{i=1}^{q} \alpha_i b_i \tag{1}$$

where $\alpha_0 = f(P2)$ and $\alpha_i = f(C_i^1) - f(P2)$.

Proof. Recombination mask $b_i = 0$ denotes that all of the variables in RC_i have variable assignments inherited from Parent P2, which is evaluated by $\alpha_0 = f(P2)$. When $b_i = 1$ all of the variables in RC_i have variable assignments inherited from Parent P1. Thus $\alpha_i = f(C_i^1) - f(P2)$ yields the change in the evaluation function when all bits in RC_i are flipped to the assignment in Parent P1. By construction and by definition the recombining components are linearly separable, and the result holds. □

We next generalize another prior result [29] by showing that local optima which appear together in lattices can be grouped into pairs, such that each pair has the same combined fitness.

Theorem 2.2. **The Lattice Constraint Theorem:** *Let C_b denote a child produced by Partition Crossover using mask b. Denote its complement by $C_{\bar{b}}$. The following equalities hold for all b:*

$$f(P1) + f(P2) = f(C_b) + f(C_{\bar{b}}) \tag{2}$$

Assume we fix any bit in mask b^q to obtain mask b^{q-1}. The following equalities also hold for all b^{q-1}:

$$f(C_{0^{q-1}}) + f(C_{1^{q-1}}) = f(C_{b^{q-1}}) + f(C_{\overline{b^{q-1}}}). \tag{3}$$

This result holds recursively for all embedded sublattices.

Proof. In Theorem 2.1 inheritance from P1 is represented by $b = 1^q$ and from P2 is represented by $b = 0^q$. It follows from Theorem 2.1 that Eqn. (2) must hold because each component makes an independent contribution to the linear equation Eqn. (1). Also assume we fix any bit in mask b to obtain mask b^{q-1}. It also follows from Eqn. (1) that Eqn. (3) must also hold. The result also holds recursively by induction. □

This is a powerful constraint that must hold over all lattices of dimension q, as well as all embedded hypercube subspace lattices and key parallel lattices. The following subspaces are just a sample of those that hold for a size 8 lattice. (Integers represent bit strings: e.g., 6 = 110).

$$f(C_0) + f(C_7) = f(C_1) + f(C_6) = f(C_2) + f(C_5) = f(C_3) + f(C_4)$$

but also (high order bit = 0): $f(C_0) + f(C_3) = f(C_1) + f(C_2)$

and (low order bit = 1): $f(C_1) + f(C_7) = f(C_3) + f(C_5)$

as well as (middle bit = 1): $f(C_2) + f(C_7) = f(C_3) + f(C_6)$

2.2 Parallel Lattices

We next construct **parallel lattices** from an arbitrary seed lattice. A seed lattice can be found by successfully recombining two local optima. The seed lattice determines both q as well as which variables are members of each recombining component.

Definition: Perfect Parallel Lattices. Parallel lattices are **perfect** if they use all or a subset of exactly the same α_i values in the linear equation Eqn. (1), except the constant α_0 can change.

2.2.1 Lattice Class PL1: Perfect Parallel Subspace Lattices. This first class of parallel lattices are lattices derived from a larger lattice. The decomposition of the larger lattice form subsets of **perfect parallel lattices.**

We can illustrate this with an example where the dimension of the lattice is $q = 4$, but we will follow-up with a real world MAX-SAT example where $q = 1374$. When $q = 4$, there are 4 recombining components: RC_1, RC_2, RC_3, RC_4. A plot of a 4-D lattice taken from a random NK-Landscape is shown in Figure 2.

Assume we decide not to use two of the recombining components, namely, we do not use RC_3 and RC_4. We are left with 2 recombining components, RC_1 and RC_2. Furthermore, assume we set RC_3 and RC_4 to match parent P2 for a subset of children $C_b'^{\,q-2}$. In this special case, α_0 does not change, and the linear equation changes from

$$f(C_b) = \alpha_0 + \alpha_1 b_1 + \alpha_2 b_2 + \alpha_3 b_3 + \alpha_4 b_4$$

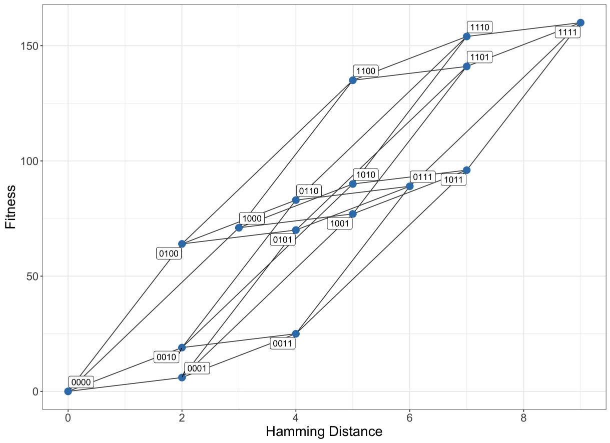

Figure 2: A lattice where all of the members are local optima. These local optima are plotted in terms of evaluation on the y-axis, and Hamming distance from a known global optima on the x-axis. Thus, this is part of a "Big-Valley" plot [3, 4] A linear equation passes through the lattice, providing the evaluation of every local optimum in the lattice.

to just a shorter form:

$$f(C_b'^{\,q-2}) = \alpha_0 + \alpha_1 b_1 + \alpha_2 b_2$$

If we instead fix RC_3 and RC_4 such that $b_3 = 1, b_4 = 1$ we only need to update the constant α_0. Thus, in a lattice of size 16, we get 4 subsets (where $b_3, b_4 = 00, 01, 10, 11$) of perfect parallel lattices of size 4. However, there are (4 choose 2) = 6 ways we could choose 2 of the 4 components. Therefore, we can generate 6 groups of 4 perfect parallel lattice of size 4. In Figure 3 two groups of 4 perfect parallel lattices are illustrated. We now generalize this result.

LEMMA 2.3. *Given a lattice with $q \geq 3$ components , if we do not use $z \leq q-2$ of the recombining components, we can fix their assignments in 2^z ways. Over all possible assignments of z this results in 2^z lattices of size 2^{q-z} of perfect parallel lattices of dimension $q-z$.*

PROOF. The constraints $q \geq 3$ and $z \leq q-2$ ensures there will be at least two active recombining components left. The original mask b now will only use $q-z$ values of α_i. Since the new subspace lattices use exactly the same recombining components extracted from P1 and P2, evaluating a corresponding subspace lattice must use exactly the same subset of α_i values, except that the reduced linear equation can be shifted by the new constant α_0. □

We next show empirically that there can exist an exponential number of subsets of large subspace lattices in real-world MAX-SAT applications. Chen [8] presents data on 26 problems from the SAT Competitions. For each problem a well known Iterated Local Search solver for MAX-SAT, the AdaptG2WSAT solver [18], was used to find 10 candidate solutions. Partition Crossover was used to recombine the 10 solutions for each instance in 45 different pairing of parents. When working with MAX-SAT a more relaxed definition of local optima is used. Any solution in a neighborhood without an improving move is accepted as "locally optimal."

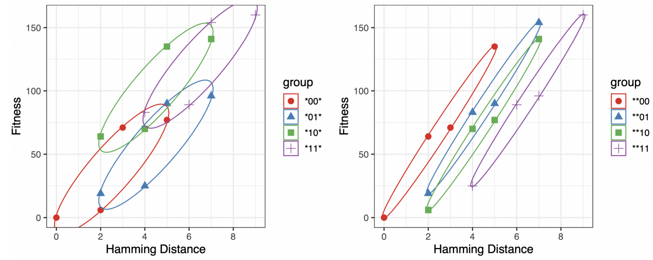

Figure 3: Two sets of Perfect Parallel Lattices where there are four members in each Parallel Lattice set. The lattices are perfect because they are subsets (and subspaces) of a larger lattice. The subset of lattices on the right use the linear equation $f(x) = \alpha_0' + \alpha_1 b_1 + \alpha_2 b_2$, where α_0' must be recomputed, except in hyperplane **00. The shift in fitness is due to α_0'.

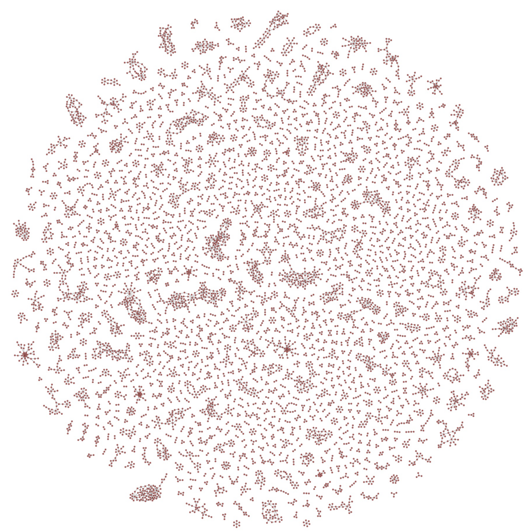

Figure 4: Recombination graph for a large industrial MAX-SAT instance AProVE09-06 from the Industrial track of the 2014 SAT competition. There are n=37,726 variables. There are q=1374 recombining components; this is the median number of recombining components found for this instance.

Figure 4 is a recombination graph from MAX-SAT instance AProVE09-06 from the industrial track of a recent SAT Competition, with n=37,726 variables. There are $q = 1374$ recombining components. MAX-SAT problems have exponentially large plateaus, where all of the solutions on a plateau have the same evaluation [10]. Determining if a plateau is "closed" with no improving move can be intractable.

The search space for MAX-SAT instance AProVE09-06 has an exponential number of parallel subspace lattices. Assume we fix 374 of the 1374 recombining components in Figure 4. We can do this in (1374 choose 374) $\approx 5.3 \cdot 10^{347}$ ways. The 374 fixed components can be fixed in 2^{374} different ways and $q = 1000$ for each of the corresponding subspace lattices. This results in approximately $5.3 \cdot$

10^{347} groups of 2^{374} subspace lattices of size 2^{1000}, all of which are perfect parallel subspace lattices using coefficients from the same original linear equation.

2.2.2 Lattice Type PL2: Perfect Parallel Lattices. We next demonstrate the existence of another type of **perfect parallel lattice.** The parallel lattices continue to have dimension q. Recall that an "unaffiliated variable" in the Shared Variable Graph has no connection to any recombining component in the Recombination Graph. We start by flipping one arbitrary unaffiliated bit in all of the members of a lattice. This include the parents P1 and P2 as well as the children in the set C^1 such that C_i^1 inherits all 1 bits from parent P1 only in recombining component RC_i.

LEMMA 2.4. *Assume any unaffiliated bit is flipped in all of the members of a lattice. In the linear equation that evaluates the children of the original lattice, evaluation of P2 changes and thus α_0 changes. Every other α_i value remains identical to the original α_i values in the seed lattice.*

PROOF. For all k-bounded pseudo-Boolean functions, the evaluation function can be expressed as a Walsh polynomial. Assume that any unaffiliated variable assignment is flipped. By definition, none of the Walsh coefficients that are connected to variables in the Recombination Graph are also connected to any unaffiliated variable. Because the shared unaffiliated variables can be replaced by a constant, flipping the assignment of an unaffiliated variable only changes α_0. Thus, $\alpha_i = f(C_i^1) - f(P2)$ cannot change after any unaffiliated variable is flipped. □

Assume there are z number of unaffiliated variables. We can flip all of the unaffilated variables, and (in theory) enumerate all of the possible 2^z Perfect Parallel Lattices.

2.2.3 Lattice Type PL3: Self-Similar Parallel Lattices. The principle of "self-similarity" is based on the idea that basic structures are repeated at different scales [17]. It is often associated with recursive constructions such as fractals, but hypercubes and lattices have self-similar recursive decompositions. Weinberger and Stadler point out that some fitness landscapes are fractal [26]; their work builds on the work of Sorkin [22] on fractal properties in energy landscapes.

We define "self-similar" parallel lattices to be lattices that shared some (but not all) of the same identical recombining components. Self-similar parallel lattice can share subspace lattices that are perfect parallel lattices (subspaces where all of the recombining components are identical). But a small number of recombining components can be somewhat different, or several recombining components can be very slightly different. The result is that "different" lattices can look extremely similar. This "self-similarity" also has a recursive structure due to the embedding of lower dimensional lattices.

Discovering lattices that look very similar turns out to be extremely easy. We discovered parallel lattices accidentially by discovering that many lattices, that seem unrelated, look very similar.

Constructing Self-Similar Lattices: One simple way to construct self-similar lattices uses the "affiliated bits" in the Shared Variable Graph. The affiliated bits have a connection to one or more variables that also appear in one or more of the recombining components. Flipping one or more of these "affiliated bits" transforms the evaluation of the children in a lattice without changing the physical structure of the lattice. And it is often the case that changing an affiliated bit will only interact with a small number of recombining components. It is also important to recognize that the Shared Variable Graph may also be disconnected (see Figure 1).

If flipping a bit only changes the contribution of one recombining component, RC_i, then only one coefficient will change in the corresponding linear equation in Theorem 2.1, namely α_i. And if the change to α_i is small, the change in the original linear equation will also be small. And if only a small fraction of the recombining components change by a small amount, the change in the original linear equation will also be small.

Our data shows that flipping affiliated bits sometimes moves an entire lattice from one group of local optima to different group of local optima in a different region of the search space. Figure 5 shows 6 self-similar lattices which were discovered by accident. These lattices have similar, but not identical recombining components. The existence of self-similar lattices appears to be relatively common.

3 EMBEDDED LATTICES AS FRACTALS

The Empirical Data: To explore Parallel Lattices and embedding of lattices, we generate 30 instances of 3 classes of problems. The problem classes are Random and Adjacent NK-Landscapes, as well as random MAX-3SAT instances generated with a power-law distribution of variables [1, 2]. We use Goldman's "hyperplane elimination" algorithm [11] to enumerate all local optima, and then recombine every local optimum with every other local optimum.

Figure 6 shows the number of lattices of different size across all 3 problem domains. Figure 7 shows the total number of lattices as a function of the number of local optima. More local optima yield more lattices, which is consistent with the fact that we only recombine local optima. Adjacent NK-Landscapes are somewhat more "decomposable" under Partition Crossover than Random NK-Landscapes. However, the power-law distribution MAX-3SAT instances yield more lattices than the Adjacent NK-Landscapes. Under the power law distribution of the MAX-SAT instances, two variables that appear together with higher frequency are more likely to appear together more than once in two or more different subfunctions, resulting in fewer total variable interactions. Also, for MAX-SAT problems, any solution x which has a neighborhood $N(x)$ without an improving move will be classified as locally optimal. Thus, a solution on a large plateaus can also be classified as a local optimum, thus yielding a larger number of "local optima."

One key way in which lattices are different is the number of local optima that are found in the set of offspring. In our data, the parents P1 and P2 are always local optima. But sometimes the children are not. For example, in a lattice of size 4, there can be 2, or 3 or 4 offspring that are local optima. We will use LO to denote the count of offspring that are local optima. We use SO (for SubOptima) for children that are not local optima, and thus can be improved by local search. A priori, we might assume that the distribution of locally optima (LO) offspring and suboptima (SO) offspring are uniformly distributed. But this is clearly not the case.

Figure 8 presents aggregrate data over all of the 30 MAX-3SAT instances. Each boxplot presents the specific number of local optima (LO) on the x-axis, and the percentage of samples for that number of local optima (LO) on the y-axis. Results are shown for lattices

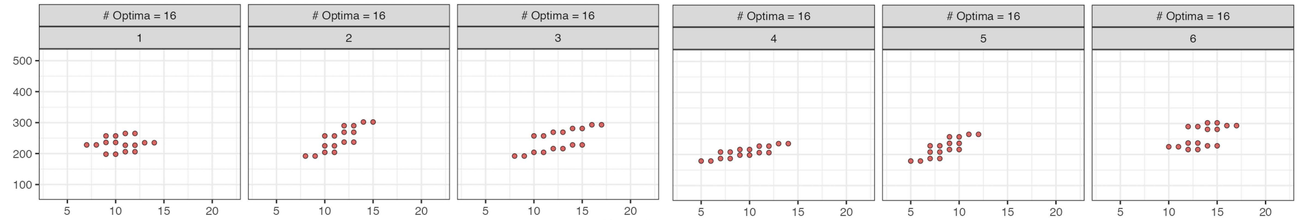

Figure 5: Self-Similar Lattices of size 16 sampled from a Random NK Landscape (N=30, k=3). These were not expected to be similar. All members of the lattices are local optima. Lattices 2 and 5 have similar recombining components. For lattices 3 and 4, a larger change in RC_3 shifted the embedded similar upper and lower sub-lattices closer together. Lattice 1 and 6 are also similar. Lattice 1 and 5 are identical in the top half of the lattice; but the bottom half has been shift on the x-axis in Hamming distance.

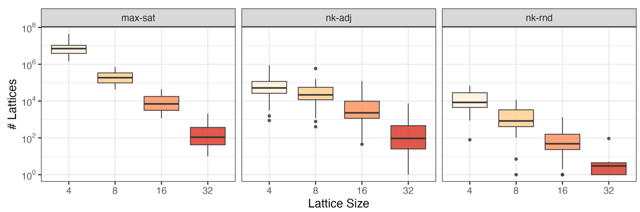

Figure 6: Distribution of the number of lattices of different sizes (Sizes 4, 8, 16 and 32) for Power-Law MAX-3SAT instances (max-sat), Adjacent NK-Landscapes (nk-adj) and Random NK-Landscapes (nk-rnd).

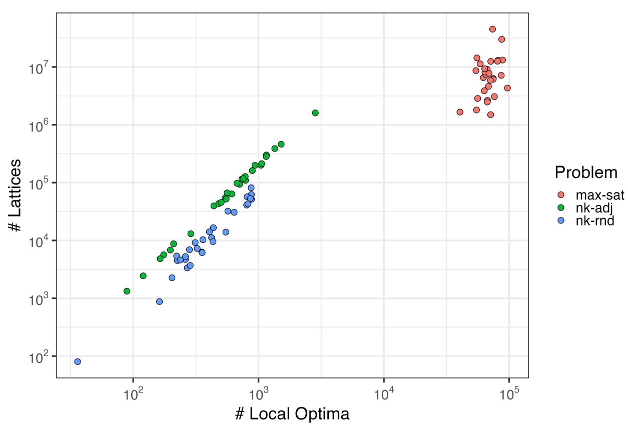

Figure 7: log-log scatter plot of the number of lattices in relation to the number of local optima. The NK-Landscapes (nk-adj and nk-rnd) display a strong correlation, with Adjacent NK-landscapes having more lattices. The power-law MAX-3SAT instances have dramatically more lattices.

of size 4, 8, and 16. The whisker boxplot format shows the average number of lattices, as well as the first and third quartile, and max and min, except for points determined to be outliers.

First, there is a strong "odd" and "even" pattern in the data where the number of local optima is most likely to be even. Odd and even counts are shown in different colors. We will mathematically justify

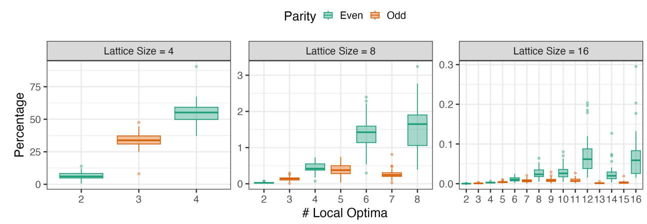

Figure 8: Distribution of the number of local optima over lattices of size 4 to 16 for all MAX-3SAT instances. For lattices of size 4, there can be 2 or 3 or 4 local optima in the lattice. For the size 8 and size 16 lattices, there is a strong bias toward more children being local optima, but also a bias toward an even number of local optima. This is due to how smaller lattices are embedded into higher dimensional lattices.

this choice. When looking at the data for individual problems, a few rare instances have more size 4 lattices with 3 local optima than 4 local optima. All instances across all domains have a very small number of lattices with 2 local optima. The boxplot for Random NK Landscapes look extremely similar to the MAXSAT instances. The Adjacent NK landscapes have a larger number of solutions that are local optima and fewer "odd" patterns in the data.

Lattices are also hypercubes. Hypercubes at dimension n have sub-hypercubes at dimension $n-1$. This helps to explain the existence of perfect parallel subspace lattices (lemma 2.3). Thus, lattices of size 16 are largely composed of sub-lattices of size 8, and lattices of size 8 are largely composed of sub-lattices of size 4. Interaction between variables found in lattices of size 4 "reappear" in the lattices of size 8 and 16.

Understanding the constraints on lattices of size 4 is important for understanding the distribution of local optima in high-dimensional lattices. "Embedding" 2D lattices into 3D lattices creates new constraints because of variable interactions between the Recombining Components and the Shared Variable Components.

3.1 Foundational lattices of size 4

We next define necessary conditions that will explain when a lattice of size 4 will have 2 or 3 or 4 local optima. Lattices of size

4 are the "base case" that provide the foundation for the recursive embedding (and inductive expansion) of all larger lattices.

First, consider a lattice of size 4 with 1 improving move, so that there is one suboptimal solution in the lattice (thus, LO = 3, SO = 1). Denote the members of the lattice by $P1$, $P2$, $C1$, $C2$, where $P1$ and $P2$ are the parents, which are local optima. From the Lattice Constraint Theorem 2.2, we know that for lattices of size 4, $f(P1) + f(P2) = f(C1) + f(C2)$.

Lemma 3.1. *In any lattice of size 4, only 1 of the 2 new children can be a suboptimal solution that can be improved by a single bit flip when Partition Crossover is applied to local optima.*

Proof. Without loss of generality, assume $C1$ is suboptimal and there is an associated improving bit flip. Flip the improving bit in **all** four strings in the lattice. This creates a parallel lattice because the recombining components and shared components stays exactly the same. Denote members of the new lattice as $P1'$, $P2'$, $C1'$, $C2'$.

Because the members of the new lattice is a parallel lattice:

$$f(P1') + f(P2') = f(C1') + f(C2')$$

However, the parents P1 and P2 are local optima. So (maximizing):

$$f(C1)+f(C2) = f(P1)+f(P2) \geq f(P1')+f(P2') = f(C1')+f(C2')$$

Which yields: $f(C1) + f(C2) - f(C1') \geq f(C2')$

Note that $f(C1') > f(C1)$ because $C1$ was improved by local search (1 bit flip). It follows that $f(C2') < f(C2)$. Thus, $f(C2')$ is not an improving move when flipping the same shared bit. □

Therefore, in any lattice of size 4 where the two offspring can both be improved by a bit flip, there must exist two distinct **shared** variables (e.g. k and z) that yield improving moves.

3.2 Embedded Lattices

Consider a lattice of size 4 with two recombining components, RC_1 and RC_2. Assume one child is a local optimum and the other child is suboptimal. Since there is one improving move, some interaction occurs between RC_1 and RC_2 and the improving variable in the Shared Variable Graph. If an improving move exists that is connected to a single recombining component, then both P1 and P2 cannot be local optima.

Next assume there exists a lattice of size 8 that uses exactly the same two recombining components RC_1 and RC_2, but also a third recombining component RC_3. Some shared bits might have different assignments, but the Shared Variable Graph stays the same *except* some shared variables can be given new complementary assignments to create component RC_3.

Assume that RC_3 has no interaction with any variable in the Shared Variable Graph that also interacts with either RC_1 and RC_2. Let the recombination mask $b = 01$ yield suboptimal solution C_{01} in the size 4 lattice. Add a bit in the 3rd position in recombination mask b for RC_3. Then $b = 010$ and $b = 011$ must correspond to suboptimal solutions in the size 8 lattice; strings in hyperplane $b = 01*$ map onto the size 4 subspace lattice. Thus, there must be **two** improving moves in the size 8 lattice. This happens because the variable assignments in RC_3 have no interaction with the improving move found in the subspace induced by RC_1 and RC_2.

We can generalize this concept. Again, assume we start with two recombining components, RC_1 and RC_2. And assume in the corresponding lattice of size 4 there is one SO (corresponding to one improving move). Next, introduce $q - 2$ additional recombining components, represented by $RC_3 \ldots RC_q$. Assume recombining components $RC_3 \ldots RC_q$ introduce no new improving moves, and have no shared variable interactions with RC_1 and RC_2. Again, let mask $b = 01$ identify the improving move found in the size 4 lattice. Then every solution in the hyperplane $b = 01*^{q-2}$ will include one improving move, and the number of local optima will be $2^q - 2^{q-2}$.

If there are zero interactions, then all improving moves will be perfectly preserved, and the change in fitness is identical in every case. Therefore, for $q = 4$ the change in evaluation for masks $b = 01^{**} = \{b = 0100, b = 0101, b = 0110, b = 0111\}$ will be *identical*; such cases exist in our data. But it is more common for only subsets of improving moves in a larger lattice to display an identical change in evaluation. This is explained by introducing the concept of a *subthreshold change in fitness*. Assume $RC_3 \ldots RC_q$ have interactions with shared variables that also interact with the recombining components RC_1 and RC_2. However, the change in evaluation resulting from these variable interactions is too small to disrupt the improving move found in the size 4 lattices (with respect to RC_1 and RC_2) and too small to introduce new improving moves in the 2^{q-2} size lattice induced by the $q-2$ new recombining components.

Definition: The nonlinear interactions between any two groups of recombining components are **insignificant** if the interactions are too small to change the number of improving moves found in the original two groups of recombining components.

Lemma 3.2. *Assume there is one improving move found in the 2D sublattice associated with RC_1 and RC_2. Assume there are also $q - 2$ additional recombining components $RC_3 \ldots RC_q$ and that the sublattice associated with $RC_3 \ldots RC_q$ has no suboptimal solutions. Assume the recombining components $RC_3 \ldots RC_q$ have insignificant interactions with recombining components RC_1 and RC_2. Then the number of local optima in the lattice of size 2^q is given by $2^q - 2^{q-2}$.*

Proof. This follows by definition, because the recombining components $RC_3 \ldots RC_q$ have **insignificant** interactions with the recombining components RC_1 and RC_2 relative to $RC_3 \ldots RC_q$. □

This result is important because in our data we see strong spikes where the number of local optima is given by $2^q - 2^{q-2}$. In particular, for lattice sizes 8, 16, 32, 64 we see strong spikes at 6, 12, 24 and 48. We also see strong spikes at 8, 16, 32 and 64 (e.g., 2^q). See Figure 8 (and Figure 9). This happens because there are no **significant interactions** between most of the recombining components.

Why is there a strong pattern of even parity in the data? Assume there is one additional recombining component RC_q added to a lattice of size 2^{q-1}. Assume RC_q has no **significant interactions** with any of the other recombining components. Then the set of recombination masks $b = *^{q-1}0$ and $b = *^{q-1}1$ will exactly double the number of local optima in the size q lattice. This doubling results in an even number of local optima. This often happens when lower dimensional lattices are embedded in higher dimensional lattices.

3.3 A Simple Fractal Rewrite System

We next explicitly show how the embedding of lower dimensional lattices in higher dimensional lattices yields a fractal structure. We construct a simple **rewrite system** that is based on a Lindenmayer system (L-system) [20]. The rewrite system uses an inductive (embedding) construction to closely approximate the distribution of local optima in lattices for a specific problem instance.

The rules used by our rewrite system are intentionally simple. Let $d(q, p)$ denote the number of lattices that have $LO = p$ local optima in a lattice of size 2^q. The operation of the rewrite system is as follows. Let $1 - r$ denote a fraction indicating what percentage of lattices of size 2^q will be embedded in some lattice of size 2^{q+1}. Variable x is a fraction indicating what percentage of local optima (LO) in the 2^q lattice are "no longer local optima" after embedding in the size 2^{q+1} lattice. For example, assume a size lattice 4 (with RC_1 and RC_2) has 4 local optima. But when a new recombining component RC_3 yields a size 8 lattice, nonlinear interactions between variables can occasionally result in 1 suboptimal offspring. Thus, there are now 7 local optima (an odd number) in the size 8 lattice.

In our model, local optima can be "destroyed" by the embedding, but no new local optima are created. The rules of our rewrite system are given in **Algorithm 1,** where variable $r = 0.5$ and variable $x = 0.12$. Variables x and r could be changed as the embedding dimension q changes, but we use constants in our simple model.

Algorithm 1 Inductive Rewrite System for Lattice LO Distributions

Input Seed distribution $d(q = 2, p)$ for a size 4 lattice
while (Still Embedding into a Larger Lattice) **do**
/*** Embed the distribution to next lattice size $q + 1$ *******/
/*** Reduce LO count by r percent: some embeddings fail **/
  for $p = 2$ to 2^q **do**
    $d(q + 1, 2p) := (1 - r) * d(q, p)$
  end for
/*** Redistribute lattice counts in size $q + 1$ lattice: ********/
/*** Initially $d(q + 1, 2p - 1) = 0$ because $2p - 1$ is odd. ***/
  for $p = 2$ to 2^q **do**
    $d(q + 1, 2p - 2) := d(q + 1, 2p - 2) + x * d(q + 1, 2p)$
    $d(q + 1, 2p - 1) := x * d(q + 1, 2p)$
    $d(q + 1, 2p) := (1 - 2x) * d(q + 1, 2p)$
  end for
$q = q + 1$
end while=0

Figure 9 (top-most) shows bar graphs for a specific MAX-3SAT instance. The graphs show the exact local optima counts for lattices of size 4, 8 and 16. Figure 9 (bottom) shows a synthetically created distribution of local optima generated using our rewrite system. The rewrite system uses a similar size 4 lattice distribution from the MAX-3SAT instance as seed. The results are surprisingly similar, despite the simple nature of our model. In both distributions, the LO counts pike at 6 and 8 for lattice size 8, and at 12 and 16 for lattice size 16.

The rewrite system could be made more precise (for example with more parameters). But it's simplicity is a testament to the importance that embedding plays in the distribution of local optima in higher dimensional lattices.

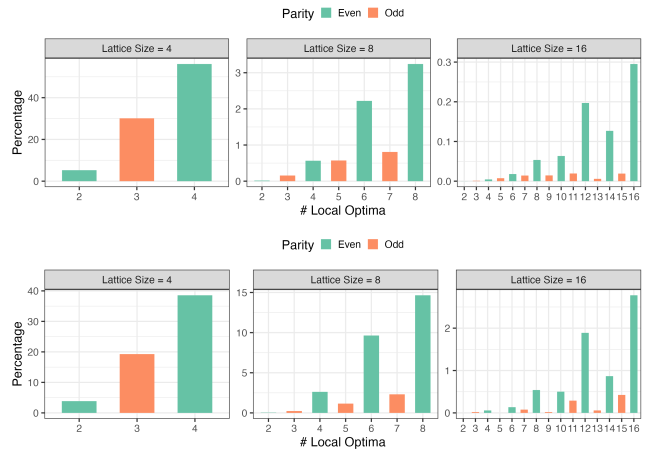

Figure 9: The top most set of bar graphs shows the distribution of local optima (LO) across all lattices of size 4, 8 and 16 for a single MAX-3SAT instance ($n = 30$). The bottom set of bar graphs shows a synthetically created distribution of local optima generated using our simple rewrite system with the distribution of the size 4 lattices as input.

4 CONCLUSIONS

It is surprising that local optima which appear in the same lattice can be evaluated using the same linear equation (Theorem 2.1). We prove that higher dimensional lattices can be broken into many lower dimensional lattices. We prove that an exponential number of lattices and local optima will share one identical linear equation when there exists **perfect parallel** lattices and sublattices (lemmas 2.3 and 2.4). This result proves that there can exist structure that is repeated again and again across different parts of the search space.

We also introduce the idea that some lattices are self-similar, but not perfect replicas of each other. Small changes can result in low level variable interactions between the recombining components that in turn result in small changes in the linear equations that correspond to these self similar lattices. Thus, self-similar lattices are made up of offspring that display similar evaluations.

How is this useful? Our work shows that the location of improving moves in perfect parallel lattices will be identical. And in self-similar lattices, the location of some improving moves will be the same, while others might be slightly different. This tells us where to look in a lattice for improving moves. Parallel sublattices can have parallel improving moves. Brute force methods for finding the best improving move after Partition Crossover will have exponential cost in an exponentially large lattice. Quickly identifying improving moves is extremely useful.

The appearance of local optima in higher dimensional lattices can be predicted in various ways by looking at the distribution of local optima in size 4 lattices. We show that subspace lattices which are recursively embedded in larger lattices display a fractal structure, which can be captured by using a rewrite system based on a Lindenmayer system. Our rewrite system can accurately model the distribution of the number of local optima in lattices.

ACKNOWLEDGMENTS

This research was supported by US National Science Foundation Grants, Award Number:1908866 and CCF Award Number (FAIN): 2426840 to Darrell Whitley. The research was also supported by the University of Malaga under project PAR 4/2023.

REFERENCES

[1] Carlos Ansótegui, Maria Luisa Bonet, and Jordi Levy. 2009. On the structure of industrial SAT instances. In *International Conference on Principles and Practice of Constraint Programming*. Springer, 127–141.

[2] Carlos Ansótegui, Maria Luisa Bonet, and Jordi Levy. 2009. Towards Industrial-Like Random SAT Instances. In *IJCAI*, Vol. 9. 387–392.

[3] K. D. Boese, A. B. Kahng, and S. Muddu. 1993. *On the big valley and adaptive multi-start for discrete global optimizations.* Technical Report. UCLA CS Department.

[4] Kenneth D. Boese, Andrew B. Kahng, and Sudhakar Muddu. 1994. A new adaptive multi-start technique for combinatorial global optimizations. *Operations Research Letters* 16 (1994), 101–113.

[5] E. Boros and P.L. Hammer. 2002. Pseudo-Boolean Optimization. *Discrete applied mathematics* 123, 1 (2002), 155–225.

[6] Lorenzo Canonne and Bilel Derbel. 2022. DRILS Revisited: Designing Perturbation in Graybox Optimization Techniques. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Boston, Massachusetts) *(GECCO '22)*. Association for Computing Machinery, New York, NY, USA, 204–212. https://doi.org/10.1145/3512290.3528774

[7] Lorenzo Canonne, Bilel Derbel, Francisco Chicano, and Gabriela Ochoa. 2023. To Combine or not to Combine Graybox Crossover and Local Search?. In *GECCO '23: Genetic and Evolutionary Computation Conference*. ACM, Lisbon Portugal, Portugal, 257–265. https://doi.org/10.1145/3583131.3590466

[8] W. Chen. 2018. *Discovering and Harnessing Structures in Solving Application Satisfiability Instances.* Ph. D. Dissertation. University Tech.

[9] F. Chicano, D. Whitley, G. Ochoa, and R. Tinós. 2017. Optimizing one million variable NK landscapes by hybridizing deterministic recombination and local search. In *Genetic and Evolutionary Computation Conference (GECCO)*. ACM, 753–760.

[10] Jeremy Frank, Peter Cheeseman, and John Stutz. 1997. When Gravity Fails: Local Search Topology. *Journal of Artificial Intelligence Research* 7 (1997), 249–281. citeseer.ist.psu.edu/article/frank97when.html

[11] B. Goldman and W. Punch. 2016. Hyperplane Elimination for Quickly Enumerating Local Optima. In *European Conference on Evolutionary Computation in Combinatorial Optimization*. Springer LNCS.

[12] H.H. Hoos and Th. Stützle. 2004. *Stochastic Local Search: Foundations and Applications.* Morgan Kaufman.

[13] Terry Jones and Stephanie Forrest. 1995. Fitness Distance Correlation as a Measure of Problem Difficulty for Genetic Algorithms. In *International Conference on Genetic Algorithms (ICGA 95)*.

[14] S.A. Kauffman. 1989. Adaptation on Rugged Fitness Landscapes. In *Lectures in the Science of Complexity*, D.L. Stein (Ed.). Addison-Wesley, 527–618.

[15] Stuart Kauffman and Simon Levin. 1987. Towards a general theory of adaptive walks on rugged landscapes. *Journal of Theoretical Biology* 128 (1987), 11–45.

[16] Stuart A. Kauffman. 1993. *Origins of order: Self-organization and selection in evolution.* Oxford University Press.

[17] Flip Korn, B-U Pagel, and Christos Faloutsos. 2001. On the" dimensionality curse" and the" self-similarity blessing". *IEEE Transactions on Knowledge and Data Engineering* 13, 1 (2001), 96–111.

[18] C Lin, W Wei, and H Zhang. 2007. Combining adaptive noise and look-ahead in local search for SAT.. In *SAT 2007*. Springer, 121–133.

[19] Marc Mezard, Giorgio Parisi, and Miguel A Virasoro. 1987. *Spin glass theory and beyond: An introduction to the replica method and its applications.* Vol. 9. World Scientific.

[20] Przemyslaw Prusinkiewicz and Aristid Lindenmayer. 2012. *The Algorithmic Beauty of Plants.* Springer Science & Business Media.

[21] B. Selman, H. Levesque, and D. Mitchell. 1992. A New Method for Solving Hard Satisfiability Problems. In *The National Conference on Artificial Intelligence (AAAI)*. San Jose, CA, 440–446.

[22] Gregory B Sorkin. 1991. Efficient simulated annealing on fractal energy landscapes. *Algorithmica* 6 (1991), 367–418.

[23] R. Tinós, D. Whitley, and F. Chicano. 2015. Partition Crossover for Pseudo-Boolean Optimization. In *Foundations of Genetic Algorithms, (FOGA-15)*. 137–149.

[24] R. Tinos, D. Whitley, and F. Chicano. 2015. Partition Crossover for Pseudo-Boolean Optimization. In *Foundations of Genetic Algorithms*. ACM Press, 137–149.

[25] E. Weinberger. 1990. Correlated and Uncorrelated Landscapes and How to Tell the Difference. *Biological Cybernetics* (1990).

[26] Edward D. Weinberger and Peter F. Stadler. 1993. Why some fitness landscapes are fractal. *Journal of Theoretical Biology* 163 (1993), 255–275.

[27] D. Whitley and W. Chen. 2012. Constant Time Steepest Descent Local Search with Lookahead for NK-Landscapes and MAX-kSAT. In *Genetic and Evolutionary Computation Conference (GECCO)*. ACM, 1357–1364.

[28] D. Whitley and G. Ochoa. 2022. Local optima organize into lattices under recombination: an example using the traveling salesman problem. In *Genetic and Evolutionary Computation Conference (GECCO)*. ACM, 757–765.

[29] D. Whitley, G. Ochoa, and F. Chicano. 2023. Partition Crossover can Linearize Local Optima Lattices of k-bounded Pseudo-Boolean Functions. In *Foundations of Genetic Algorithms*. ACM.