# A Study of Neutrality of Boolean Function Landscapes in Genetic Programming

Leonardo Vanneschi, Yuri Pirola, Giancarlo Mauri, Philippe Collard, Sébastien Verel

# A Study of Neutrality of Boolean Function Landscapes in Genetic Programming

Leonardo Vanneschi, Yuri Pirola, Giancarlo Mauri

*Dipartimento di Informatica Sistemistica e Comunicazione (D.I.S.Co.), University of Milano-Bicocca, Milan, Italy*

Marco Tomassini

*Information Systems Department, HEC, University of Lausanne, Switzerland*

Philippe Collard, Sébastien Verel

*I3S Laboratory, University of Nice–Sophia Antipolis, France*

**Abstract**

Neutrality of genetic programming Boolean function landscapes is investigated in this paper. Compared with some well known contributions on the same issue, (*i*) we first define new measures which help characterizing neutral landscapes; (*ii*) we use a new sampling methodology, which captures features that are disregarded by uniform random sampling; (*iii*) we introduce new genetic operators to define the neighborhood of tree structures; and (*iv*) we compare the fitness landscape induced by different sets of functional operators. This study indicates the existence of a relationship between our neutrality measures and the performance of genetic programming for the problems studied.

*Keywords:* Neutrality, Fitness Landscapes, Boolean Functions, Genetic Programming, Problem Difficulty, Negative Slope Coefficient.

## 1. Introduction

The role played by neutrality in determining the ability of evolutionary algorithms to find good quality solutions for a given problem has been a controversial issue in the last few years. A good introduction on the role of neutrality has been done by Reidys and Stadler in [1]. In [2], Collard et al. studied synthetic neutrality and its effects on the evolvability of Genetic Algorithms (GAs). They showed that a GA is able to explore new regions of the search space, and thus sometimes improve its performance, owing to this synthetic neutrality. Later, in [3], Toussaint and Igel claim the necessity of a certain degree of neutrality in fitness landscapes for self-adaptation. Consistently with the work of Collard et al., they showed that, in the absence of external control, neutrality allows a variation of the search distribution without the risk of fitness loss, and that this is beneficial for the effectiveness of the evolutionary process on a set of GA benchmarks. In the same year, Geard and coworkers [4] compared the neutrality of some binary landscapes, once again claiming a relationship between neutrality and performance of GAs. In particular, they consider the Kauffman's well-known NK landscape model, studying two variants of it, with significantly different structural properties: NKp and NKq. The fitness distributions of these neutral landscapes reveal that their levels of correlation with non-neutral landscapes are significantly different, as are the distributions of neutral mutations. They describe a series of simulations on NK, NKp and NKq landscapes with varying levels of epistatic interaction, claiming that these simulations demonstrate differences in the way that epistatic interaction affects the "searchability" of neutral landscapes. They conclude that neutrality has an impact on both the structure of the resulting landscape and on the performance of evolutionary search algorithms on these landscapes. Two years later, Collard and coworkers [5] proposed a new search heuristic using the scuba diving metaphor. This approach is based on the concept of evolvability and tends to exploit the neutrality that exists in many problems. Despite the fact that natural evolution does not directly select for evolvability, the basic idea behind the scuba search heuristic is to explicitly push evolvability to increase. This idea has been deepened by Galvan-Lopez in his PhD thesis [6]. In particular, he studied the relationship between neutrality and effectiveness of evolutionary search algorithms by measuring the "amount" of neutrality synthetically added to fitness landscapes with some well known difficulty measures such as fitness-distance correlation [7, 8]. This idea has, at least partially, inspired the present work, where we study the neutrality of some genetic programming (GP) Boolean function landscapes and we also estimate their difficulty by using another difficulty measure that we have recently introduced, called negative slope coefficient [9, 10, 11].

The study of neutrality for GP has had a slower start, probably because of the higher complexity of GP fitness landscapes. The first contribution is probably due to the work of Yu and Miller: in [12], they showed that artificially introducing neutrality can help Cartesian GP to navigate some restricted fitness landscapes. In particular, they claim that synthetically adding neutrality to "needle in haystacks" landscapes, it is possible to improve the effectiveness of Cartesian GP in finding the optimal solution. These results have been recently criticized by Collins and coworkers in [13]. In particular, they showed that the method of sampling used by the Cartesian GP is significantly less effective at locating solutions for "needle in haystacks" landscapes than the solution density of the corresponding formula space would warrant. They presented results indicating that the loss of performance is caused by the sampling bias of Cartesian GP, due to the neutrality-friendly representation. They also implemented a simple intron-free (i.e., without neutrality) random sampling algorithm, showing that it performs considerably better on the same problem, giving an interpretation of such performance. Even though the work of Collins and coworkers casts a shadow on the effectiveness of neutrality in evolutionary search, many other contributions on the importance of artificially introducing neutrality into fitness landscapes can be found (see for instance [14, 15, 16]), all of them showing that the approach is beneficial on some particular classes of problems.

We feel that, in controversial debate on the usefulness of neutrality for GP, what may often be misleading is *what kind* of neutrality is being considered: many different ways of intending and formalizing the concept of neutrality may exist and each one of them may lead to different, and in some cases conflicting, conclusions. For this reason, in this paper we take up a different point of view: first of all, we study Boolean function landscapes for standard tree-based GP [17] rather than Cartesian GP. Second, we study the landscapes without explicitly modifying their neutrality. Our only modification to the studied fitness landscapes consists in the use of a set of simple genetic operators, that have not been explicitly designed to alter neutrality. Third, we introduce some new neutrality measures, such as the *average neutrality ratio*, the *average* $\Delta$*-fitness* of neutral networks and the *ratio* of some particular solutions contained into the neutral networks, called *non-improvable* and *non-degradable* solutions (see Section 2 for the definitions of these measures). These measures are based on the concept of neutral network, that has been used in many contributions [18, 19, 20, 21], and that has been studied for GP Boolean spaces, for instance, by Banzhaf and Leier in [22]. These measures give a particular view of neutrality, which we believe is particularly useful to make inferences or diagnoses on the hardness of the underlying fitness landscape. The goal of the neutrality measures introduced here is to provide tools for investigating the shape and features of Boolean fitness landscapes, possibly allowing some inference on their difficulty. None of them is intended to be a "stand-alone", infallible hardness measure, since each one of them focuses only on one particular feature of the landscape; but considering them all together should allow us to have an interesting picture of fitness landscapes, especially those related to neutrality.

Given that these measures have been defined to help characterizing problem difficulty, it is natural to investigate the relationship between them and other hardness measures that can be found in literature. In this paper, we match the results obtained by these measures both with *Success Rate* statistics, obtained executing a set of GP experiments and with the results of a GP hardness measure called *Negative Slope Coefficient* (*NSC*) that we have recently introduced [9, 10, 11]. Finally, instead of using a fixed set of functions to build solutions, we compare the landscapes induced by different sets of Boolean operators ({NAND} and {XOR, NOT} for the even parity landscapes, and {NAND} and {IF} for multiplexer).

Boolean functions represent a very important set of benchmarks for GP, mainly because many application domains exist for which these types of landscapes are a good model, as for example applications of automatic synthesis of digital circuits, or applications in the fields of cryptography and telecommunications. Furthermore, some characteristics of these functions make them particularly suitable to study neutrality. For instance, they have a more restricted domain than other well known GP benchmarks like various forms of real-valued symbolic regression.

Boolean function landscapes have already been studied, among others, in [23]. In those contributions, either landscapes of small size have been studied exhaustively (i.e., taking into account all the possible solutions) or larger fitness landscapes have been studied by means of uniform random samplings. The shape and features of the Boolean function fitness landscapes make them hard to study by means of uniform random samplings (as we explain in Section 3) and thus more sophisticated sampling methods are needed. The first attempt to study them by means of some well known sampling techniques can be found in [11, 24]. In this paper we define a new, and more elaborate, sampling methodology to study Boolean function landscapes, but the techniques that we propose are general and can potentially be used for any GP space.

This paper is structured as follows: in Section 2 we give some preliminary definitions about fitness landscapes, neutrality and some hardness measures that will be used later. Section 3 introduces Boolean function landscapes and some of their most significant characteristics. In Section 4, we present our new sampling methodology. Section 5 contains experimental results for some *even parity* and *multiplexer* Boolean landscapes. Finally, in Section 6 we offer our conclusions and hints for future research.

## 2. Preliminary Definitions

In this section, we review some fundamental concepts used in the rest of the paper. In particular, in the first part we formally define the concept of fitness landscapes and neutrality, while in the second part we present a measure, called *Negative Slope Coefficient*, that has been proposed to characterize (and possibly predict) the problem difficulty for GP.

### 2.1. Fitness Landscapes and Neutrality

Using a landscape metaphor to gain insight about the workings of a complex system originates with the work of Wright on genetics [25]. A simple definition of fitness landscape in EAs is a plot where the points in the horizontal plane represent the different individual genotypes in a search space (placed according to a particular *neighborhood relationship*) and the points in the vertical direction represent the fitness of each one of these individuals [23]. Generally, the neighborhood relationship is defined in terms of the genetic operators used to "traverse" the search space [26, 23, 11]. This can be done easily for unary genetic operators like mutation, but it is clearly more difficult if binary or multi-parent operators, like crossover, are considered. Formal definitions of fitness landscape have been given (e.g. in [27]). Following these definitions, in this work a fitness landscape is a triple $\mathcal{L} = (\mathcal{S}, \mathcal{V}, f)$ where $\mathcal{S}$ is the set of all possible solutions, $\mathcal{V} : \mathcal{S} \to \mathcal{P}(\mathcal{S})$ is a neighborhood function specifying, for each $s \in \mathcal{S}$, the set of its neighbors $\mathcal{V}(s)$, and $f : \mathcal{S} \to \mathbb{R}$ is the fitness function. Given the set of variation operators, $\mathcal{V}$ can be defined as $\mathcal{V}(s) = \{s' \in \mathcal{S} \mid s'$ can be obtained from $s$ by a single variation$\}$. In some cases, as for the even parity problems, even though the size of the search space $\mathcal{S}$ is huge, $f$ can only assume a limited set of values (as we clarify in Section 3). Thus, a large number of different solutions have the same fitness. In this case, we say that the landscape has a high degree of neutrality [1]. Given a solution $s$, a particular subset of $\mathcal{V}(s)$ can be defined: the one composed of neighbor solutions that are also neutral. Formally, the *neutral neighborhood* of $s$ is the set $\mathcal{N}(s) = \{s' \in \mathcal{V}(s) \mid f(s') = f(s)\}$. If continuous fitness functions are used, considering exactly the same fitness values can be unrealistic. In those cases, a threshold $\delta$ is usually fixed such that a fitness difference between neighbors smaller than $\delta$ is considered neutral. In that case, the previous definition would become: $\mathcal{N}(s) = \{s' \in \mathcal{V}(s) \mid |f(s') - f(s)| < \delta\}$. Given that Boolean functions induce discrete fitness landscape, in this paper we only consider the first one of these definitions: two neighbors are considered as neutral if and only if the have exactly the same fitness value. The number of neutral neighbors of $s$ is called the *neutrality degree* of $s$ and the ratio between neutrality degree and cardinality of $\mathcal{V}(s)$ is the *neutrality ratio* of $s$. Given these definitions, we can imagine a fitness landscape as being composed of a set of (possibly large) *plateaus*. More formally, a *neutral network* [28] can be defined as a connected component of the graph $(\mathcal{S}, \mathcal{E}_{\mathcal{N}})$ where $\mathcal{E}_{\mathcal{N}} = \{(s_1, s_2) \in \mathcal{S}^2 \mid s_2 \in \mathcal{N}(s_1)\}$. We define the *fitness of a neutral network* (or *network fitness*) as the fitness value shared by all individuals of this neutral network. Finally, the *neutrality graph* of a fitness landscape is a graph $(N, A)$ such that $N$ is the set of neutral networks, and two neutral networks $n_i, n_j$ are connected by an edge $(n_i, n_j) \in A$ if there exists an individual $s_i$ of $n_i$ that has a neighbor $s_j \in \mathcal{V}(s_i)$ belonging to $n_j$.

### 2.2. Negative Slope Coefficient

Evolvability is a feature that is intuitively related to problem difficulty, although with much broader connotations. It has been defined as the ability of genetic operators to improve fitness quality [29]. Many other definitions of evolvability, with interesting differences compared to the one in [29] can be found in literature. For instance in [30], it is defined as the ability of a system to *continually* improve its fitness. One possible and very simple way of studying evolvability is to plot the fitness values of individuals against the fitness values of their neighbors, where a neighbor is obtained by applying one step of a genetic operator to the individual. We have called such a plot *fitness cloud* in [31]. Let $\mathcal{L} = (\mathcal{S}, \mathcal{V}, f)$ be a fitness landscape as defined above. The following set of points can be defined: $S = \{(f(s), f(s')) \mid s \in \mathcal{S},\ s' \in \mathcal{V}(s)\}$. The graphical representation of $S$ on a bidimensional plane, or fitness cloud, is the scatterplot of the fitness of all the individuals belonging to the search space against the fitness of all their neighbors.

In general, the sizes of the search space and of the neighborhoods do not allow one to consider all the possible individuals. Thus, samples are needed. Since the EA's selection algorithm is likely to eliminate bad individuals from the population, sampling techniques that assign a higher priority of being sampled to good individuals must to be used. In [11, 10, 9], the well-known *Metropolis-Hastings* technique [32] is used to sample the search space and the $k$-tournament selection algorithm [17] (with $k = 10$) is used to sample neighborhoods (see [11] for a detailed motivation of these choices). In this case, the fitness cloud is plotted for a sample of individuals (obtained by the Metropolis-Hastings technique) and a subset of their neighborhood (obtained by applying tournament selection).

The fitness cloud can be of help in determining some characteristics of the fitness landscape related to evolvability and problem difficulty. But the mere observation of the scatterplot is not sufficient to quantify these features. In [11, 10, 9] we have introduced an algebraic measure called *Negative Slope Coefficient* (*NSC*). It can be calculated as follows: the abscissae of a fitness cloud can be partitioned into $k$ segments $\{I_1, I_2, \dots, I_k\}$. The algorithms that can be used to suitably perform this partitioning are described in [9] and will not be discussed here. From those segments, one can obtain the set $\{J_1, J_2, \dots, J_k\}$, where each $J_i$ contains all the ordinates corresponding to the abscissas contained in $I_i$. Let $M_1, M_2, \dots, M_k$ be the averages of all the abscissa fitness values contained inside the segments $I_1, I_2, \dots, I_k$ and let $N_1, N_2, \dots, N_k$ be the averages of the corresponding ordinate values in $J_1, J_2, \dots, J_k$. Then, the set of segments $\{S_1, S_2, \dots, S_{k-1}\}$ can be defined, where each $S_i$ connects the point $(M_i, N_i)$ to the point $(M_{i+1}, N_{i+1})$. For each one of these segments $S_i$, the *slope* $P_i$ is defined as $P_i = (N_{i+1} - N_i)/(M_{i+1} - M_i)$. Finally, the Negative Slope Coefficient is defined as

$$NSC = \sum_{i=1}^{k-1} \min(0, P_i)$$

The hypothesis proposed in [10] is that *NSC* should classify problems in the following way: if *NSC*= 0, the problem should be easy; if *NSC*< 0 the problem should be difficult and the value of *NSC* should quantify its difficulty: the smaller its value, the more difficult the problem (in other words problem difficulty increases as *NSC* decreases further away from zero). The informal idea behind this hypothesis is that the presence of a segment with negative slope indicates a difficulty in improving evolvability for individuals having fitness values contained in that segment [11]. In [33], we have given a more formal justification of the *NSC*, while in [34], we have pointed out some drawbacks of this measure, among which the fact that it is not normalized into a given interval, and this makes it impossible to use it to categorize problems of different nature into difficulty classes. On the other hand, results shown in [11, 10, 9] report good predictions of the problem hardness for a large set of GP benchmarks, including various versions of symbolic regression, the artificial ant on the Santa Fe trail, the intertwined spirals problem and also Boolean problems, like the even parity and the multiplexer. Finally, in [35], we have pointed out the ability of the *NSC* to correctly quantify the difficulty of some real-life applications.

## 3. Boolean Function Landscapes

Boolean symbolic regression problems (BSRP) are a class of problems that are very often used as benchmarks in GP, given the difficulty of GP in finding optimal solutions for some of their instances, despite the high simplicity of their specifications. The goal of these problems is to find a Boolean expression that exactly represents a given truth table. In tree-based GP, the set of acceptable solutions is defined as the set of all the well formed trees having a depth less then or equal to a fixed value $h$ and that can be constructed using a set of operators $\mathcal{F}$ and a set of terminal symbols $\mathcal{T}$. From now on, a BSRP is called of *order* $k$ if $k = |\mathcal{T}|$, i.e. if the Boolean expressions can be built on a set of $k$ variables. The fitness function $f$ of an expression $E$ is calculated as the number of input data for which $E$ does return the same value as the target function. In this paper, the fitness values have always been normalized into the $[0, 1]$ range, by dividing them by $2^k$, where $k$ is the problem's order. Thus, from now on a solution with fitness equal to 0 represents an optimal solution, while 1 is the worst possible fitness value. To define a neighborhood structure, we have to choose a suitable set of variation operators. Standard crossover or subtree mutation [17] generate neighborhoods that are too wide and complex to be studied. In this paper, we consider a simplified version of the inflate and deflate mutation operators that we have introduced in [11, 26] (also called structural mutation operators in those works): (1) *Strict* *deflate mutation*, which transforms a subtree of depth 1 into a randomly selected leaf chosen among its children. (2) *Strict inflate mutation*, which transforms a leaf into a tree of depth 1, rooted in a random operator and whose children are a random list of variables containing also the original leaf in a random position. (3) *Point terminal mutation*, that replaces a leaf with another random terminal symbol. This set of genetic operators (that will be called *Strict-Structural*, or *StSt*, mutation operators from now on) is easy enough to study and provides enough exploration power to GP. For instance, *StSt* mutations present two important properties:

(i) each mutation has an inverse: let $M$ be the set of *StSt* mutation operators and let $S$ be the set of all the possible individuals (search space). For each pair of individuals $(i, j) \in S$, if an operator $m \in M$ exists such that $m(i) = j$, then an operator $m^{-1} \in M$ such that $m^{-1}(j) = i$ always exists. In other words, if an individual $i$ can generate an individual $j$ by one application of a *StSt* mutation operator, $i$ can always be generated from $j$ by one application of another *StSt* mutation operator;

(ii) for each pair of solutions, a sequence of mutations that transforms the first one into the second exists (not necessarily unique).

See [24, 36] for the formal proofs of these properties. Thus, the associated graph $(\mathcal{S}, \mathcal{V})$ of fitness landscape is an undirected (by Property (i)) and connected (by Property (ii)) graph.

In the rest of this section, we introduce the Boolean fitness landscapes that have been actually used in our study. In particular, we considered the *Even Parity* and the *Multiplexer* problems under different sets of Boolean operators. We empirically show that the different sets of operators induce problems of different hardness for the GP. Such a difference is immediate by considering the results of two problem difficulty measures that we computed on our landscapes, namely the *Success Rate* and the *Negative Slope Coefficient*.

## 3.1. The Even Parity Problem

The goal of the even-$k$ parity problem [17] is to find a Boolean function of $k$ variables that returns True if an even number of inputs are True and False otherwise. The set $\mathcal{T}$ is composed of $k$ variables (where $k$ is the order of the problem). Two different function sets are studied in this work: {XOR, NOT} and {NAND}. Obviously, both the sets can describe an exact solution for the even parity problem. However, the "minimal" exact solution (in term of size of the expression) that uses only XORS and NOTS is considerably smaller than the minimal exact solution that uses only NANDS. We chose these sets because it would be, intuitively, easier to construct an exact solution using the set {XOR, NOT} than using the set {NAND}. Notice that the set {XOR, NOT}, although it allows to "easily" describe an exact solution of the even parity problem, cannot represent all the Boolean functions (i.e., it is not *functionally complete* [37, 24, 36]) and should not used for other Boolean symbolic regression problems. Every Boolean function, instead, can be expressed as composition of NANDS (i.e., {NAND} is functionally complete [37]), and, therefore, the second function set can be used to solve other BSR problems. We have chosen these function sets because they are small enough to limit the cardinality of the search space but also rich enough to represent some perfect solutions. Furthermore, these function sets induce two fitness landscapes with different difficulties for GP [11]: the landscape induced by {XOR, NOT} is easy to search, while the one induced by {NAND} is generally hard. This fact is confirmed by the results shown in Table 1, where the values of the success rate (indicated by $SR$ in the table) for three different values of the mutation rate and *NSC* are reported for both landscapes. The success rate is the percentage of runs in which the global optimum was found within

| Set of operators | $SR_{(p_m=0.95)}$ | $SR_{(p_m=0.5)}$ | $SR_{(p_m=0.25)}$ | *NSC* |
|---|---|---|---|---|
| {XOR, NOT} | 1 | 1 | 1 | 0 |
| {NAND} | 0.03 | 0 | 0 | -0.14 |

Table 1: Values of the success rate for $p_m = 0.95$, $p_m = 0.5$ and $p_m = 0.25$ and of the *NSC* for the even-4 parity problem using two different sets of operators to build the individuals. The fitness landscapes induced by these two sets of operators clearly have different difficulties for GP.

200 generations (100 total runs have been executed in this paper). This definition is informal and prone to criticism: for example, in some cases, to calculate the success rate, one is forced to choose an error threshold, under which a fitness value is considered successful, and this is usually a rather arbitrary choice that can affect results. Nevertheless, good or bad success rate values, in particular when they are very different to each other, can correspond to our intuition of what

"easy" or "hard" means in practice. Thus $SR$ can be used as a broad measure to experimentally confirm our difficulty predictions. The success rate results reported in Table 1 have been obtained by executing 100 independent GP runs using the even-4 parity problem, maximum tree depth for the individuals equal to 8, population of size 100, ramped half-and-half population initialization, tournament selection of size 10, *StSt* mutations as genetic operators. Only one *StSt* mutation operator has been applied (100 runs have been executed with mutation probability equal to 0.95, 100 runs with mutation probability equal to 0.5 and 100 runs with mutation probability equal to 0.25) to each selected individual. The choice of the particular mutation operator has been done uniformly at random between the three *StSt* mutations. The results related to the *NSC* reported in Table 1 have been obtained by generating a sample of 40000 individuals by means of the Metropolis-Hastings algorithm and, for each of them, a neighbor has been generated by applying one *StSt* mutation. Once again, the choice of the particular mutation operator has been done uniformly at random between the three *StSt* mutations. Further results of the *NSC* for the even parity problem can be found in [10, 11].

The interpretation of the results in Table 1 is straightforward: the landscape induced by the $\{\textsc{xor}, \textsc{not}\}$ set of operators is easier than the one induced by $\{\textsc{nand}\}$ for GP (for all the mutation rates that we have used) and these results are also confirmed by the *NSC* values. Thus, we can compare the two landscapes (indicated by $\mathcal{L}^{\{\textsc{xor},\textsc{not}\}}_{(k,h)}$ and $\mathcal{L}^{\{\textsc{nand}\}}_{(k,h)}$ from now on, where $k$ is the problem order and $h$ is the prefixed tree depth limit; $k$ and $h$ will be omitted when not necessary), to find some interpretations of their different difficulties.

We recall some other interesting properties of the even parity fitness landscapes that we proved and discussed, among others, in [24, 36]. First of all, assuming that all fitness values have been normalized into the range [0, 1], if an expression does not contain at least one occurrence of each variable, then its fitness value is exactly equal to 0.5. For this reason, the wide majority of individuals in the even parity landscapes have fitness 0.5 [11]. Secondly, an expression in the $\mathcal{L}^{\{\textsc{xor},\textsc{not}\}}$ landscape can only have a fitness value equal to 0, 0.5 or 1. (see for instance [23, 24, 36] for the formal proofs of these properties). If we chose the *StSt* mutation operators, some other properties of the $\mathcal{L}^{\{\textsc{xor},\textsc{not}\}}$ landscape exist: (a) there is *only* one neutral network at fitness 0.5 (we call it the *central network*), (b) all the other networks (we call them the *peripheral networks*) are composed by *one* single individual (and thus we can call them *degenerate* networks) and (c) all the peripheral networks are adjacent to the central one (in the graph-theoretic sense). Property (c) can be easily proved by showing that, for each solution $s$, a single application of one *StSt* mutation on $s$ leads to an individual of fitness 0.5. Property (b) can be proved by showing that every *StSt* mutations of an individual of fitness 0 or 1 produce an offspring with a different fitness value. Finally, Property (a) can be proved by showing that for each pair of solutions with fitness 0.5 a sequence of neutral *StSt* mutations that transforms the former into the latter exists. The full proofs of these properties can be found in [24, 36].

## 3.2. *The Multiplexer Problem*

The goal of the $k$-multiplexer [17] problem is to design a Boolean function with $k$ inputs and one output. The first $x$ of the $k$ inputs can be considered as address lines. They describe the binary representation of an integer number. This integer chooses one of the $2^x$ $(= k - x)$ remaining inputs. The correct output for the multiplexer is the input on the line specified by the address lines. The terminals are the $k$ inputs to the function. In this paper, we have used two different sets of non-terminals: $\{\textsc{if}\}$ (where $\textsc{if}(x, y, z)$ is a ternary Boolean function that returns $y$ if $x$ is *true* and $z$ otherwise) and $\{\textsc{nand}\}$. As for the case of the even parity benchmark, we have chosen these two sets because they are small enough to limit the cardinality of the search space but rich enough to represent some perfect solutions. These two sets of Boolean operators induce two landscapes (indicated by $\mathcal{L}^{\{\textsc{if}\}}_{(k,h)}$ and $\mathcal{L}^{\{\textsc{nand}\}}_{(k,h)}$ from now on, where $k$ is the problem order and $h$ is the prefixed tree depth limit) with two different difficulties for GP. This fact is confirmed by the results shown in Table 2, where the values of the success rate ($SR$) for three different mutation rates and *NSC* are reported for both landscapes. The success rate results reported in Table 2 have been obtained by executing 100 independent GP runs using the

| Set of operators | $SR_{(p_m=0.95)}$ | $SR_{(p_m=0.5)}$ | $SR_{(p_m=0.25)}$ | *NSC* |
|---|---|---|---|---|
| $\{\textsc{if}\}$ | 1 | 0.98 | 0.71 | 0 |
| $\{\textsc{nand}\}$ | 0 | 0 | 0 | -0.21 |

Table 2: Values of the success rate for three different mutation rates and of the *NSC* for the 6-multiplexer problem using two different sets of operators to build the individuals. The fitness landscapes induced by these two sets of operators clearly have different difficulties for GP.

6-multiplexer problem, maximum tree depth for the individuals equal to 6 for the landscape induced by $\{\textsc{nand}\}$ and to 5 for the landscape induced by {IF} (this difference in the tree depths for the two landscapes is justified by the fact that these are the landscapes that will be studied later in this paper; the choice of these values for the tree depths are motivated in Section 5.2.2), population of size 100, ramped half-and-half population initialization, tournament selection of size 10, *StSt* mutations as genetic operators. As in the case of even parity problem, only one *StSt* mutation operator has been applied (with probability equal to 0.95, 0.5 and 0.25 in the three sets of runs) to each selected individual. The choice of the particular mutation operator has been done uniformly at random between the three *StSt* mutations. In this case, a run has been considered successful when an individual with a lower fitness than 0.15 has been found. The results related to the *NSC* reported in Table 2 have been obtained by generating a sample of 40000 individuals generated with the Metropolis-Hastings algorithm and, for each of them, a neighbor by applying one *StSt* mutation. Once again, the choice of the particular mutation operator to generate the neighbor has been done uniformly at random between the three *StSt* mutations. Further results of the *NSC* for the multiplexer problem can be found in [9]. The interpretation of the results in Table 2 is straightforward: $\mathcal{L}^{\{\text{NAND}\}}$ is an easier landscape than $\mathcal{L}^{\{\text{IF}\}}$ for GP.

## 4. Sampling Methodology

Boolean function fitness landscapes are in general very hard to sample. For instance, for the even-$k$ parity problem, the large majority of the individuals have fitness equal to 0.5 and as the order $k$ of the problem increases, the percentage of individuals with a fitness equal to 0.5 increases too. In [23] uniform random samplings for these spaces have been presented. In [11] sampling techniques such as Metropolis and Metropolis-Hastings [32] have been used. Even though the results obtained were satisfactory for the purposes of those works, still many individuals had fitness equal to 0.5 and too few ones with different fitness were considered. Thus, those samples did not capture some important characteristics of the fitness landscape, such as the number and size of the neutral networks at fitness values different from 0.5 and the connectivity of optimal solutions to these networks. In other words, those samplings did not offer a useful "view" of the fitness landscapes and did not allow us to completely understand the behavior of GP on them. In this paper, we present a new methodology to generate samples containing trees of many (possibly all) different fitness values and forming connected neutral networks, if possible. This technique is composed of three steps: we have called them *modified Metropolis*, *vertical expansion*, and *horizontal expansion*. Modified Metropolis generates a sample $S$ of individuals. The vertical expansion tries to enrich $S$ by adding to it some *non-neutral* neighbors of its individuals. Finally, the horizontal expansion tries to enrich $S$ by adding to it some *neutral* neighbors of its individuals. The meaning of names "vertical" and "horizontal" becomes apparent if we think of our sampling methodology as an exploration of a bi-dimensional plane. In fact, if we project the solution space to a plane where the $y$-axis represents the fitness values, then the vertical expansion is the step that tries to expand the initial sample along the $y$-axis (*i.e.*, non-neutral neighbors), while the horizontal expansion step tries to add solutions that lay on the same horizontal line of the previously found ones (*i.e.*, neutral neighbors).

### *4.1. Modified Metropolis Sampling*

Our sampling methodology has been inspired by the Metropolis technique. According to that technique, a solution is generated at random at the beginning and considered as the *current* solution $P$. Successively, a loop is executed. At each iteration of that loop, a new solution $T$ is generated at random and accepted (and thus inserted into the sample and considered as the new current solution $P$) or rejected according to a certain probability distribution. In the Metropolis technique, the distribution for accepting or rejecting individuals is equal to $\alpha_M(f(P), f(T)) = \min\left(1, \frac{f(P)}{f(T)}\right)$, where $f$ is the fitness function. In this way, the Metropolis technique favors fitter solutions but it does not penalize solutions at fitness 0.5. In our methodology, instead, we define a probability distribution $\alpha$ that rewards solutions with a different fitness than the previously accepted one. In this way, we intend to reward solutions with a different fitness than 0.5. Let $p_{min}$ be the minimum probability of accepting a solution, then our definition of the $\alpha$ function is:

$$\alpha(f(P), f(T)) = (1 - p_{min}) \cdot \log_{10}\left(\frac{9 \cdot |f(P) - f(T)|}{\max(f(P), 1 - f(P))} + 1\right) + p_{min} \tag{1}$$

In this way, if $|f(P) - f(T)|$ is equal to 0, the new solution $T$ gets a small probability (equal to $p_{min}$) of being accepted. If we set $p_{min} = 0$ and $f(P)$ is equal to 0.5, then the algorithm is likely to never terminate. Thus, a value of $p_{min}$ larger than zero, even though "as small as possible", has to be used. On the other hand, the larger the value of

$|f(P) - f(T)|$, the higher the probability. In particular, if $T$ has the most different possible fitness value from $P$, then $|f(P) - f(T)| = \max(f(P), 1 - f(P))$. In that case, the logarithmic term becomes $\log_{10}(9 + 1) = 1$, and thus $\alpha(f(P), f(T)) = 1$. We have chosen a logarithmic function because its returned value increases very quickly for small differences (a small delta in the value of $|f(P) - f(T)|$ results in a large change in the value of the logarithm) and thus it also rewards solutions $T$ with a slightly different fitness from $P$. Moreover, we have chosen the base-10 logarithm because we wanted the maximum value of $\alpha(f(P), f(T))$ to be equal to 1.

### *4.2. Vertical Expansion*

The vertical expansion of our methodology takes as input the sample $S$ generated by the modified Metropolis algorithm and enriches it by adding some new individuals. In synthesis, it works as follows: for each individual $i \in S$, $L$ different neighbors of $i$ are generated by means of $L$ *StSt* mutations. Each one of these neighbors can be accepted or rejected according to the probability distribution expressed by Equation 1. All accepted neighbors are finally inserted in $S$, that is returned as output of the vertical expansion phase. Since the value of $p_{min}$ is "small", there is a "small" probability of having neutral neighbors in $S$ at the end of the vertical phase.

### *4.3. Horizontal Expansion*

Let an *incomplete neutral network* be a sample $I_N$ of a neutral network $N$ such that at least one neutral neighbor $j$ of an individual $i \in I_N$ exists such that $j \notin I_N$. The horizontal expansion phase of our sampling technique takes as arguments the sample $S$ returned by the vertical expansion phase, the minimum admitted size of an incomplete neutral network $I_{min}$ and the maximum size of the sample that has to be generated $S_{max}$. These last two measures are parameters of our sampling technique and have to be manually defined. The horizontal phase returns a new sample $S$, possibly enriched with some individuals that are neutral neighbors of the individuals belonging to the sample returned by the vertical phase. Thus, the sample $S$ returned by the horizontal phase will hopefully contain larger neutral networks than the ones contained in the sample returned by the vertical phase. The horizontal expansion algorithm can be defined by the pseudo-code in Algorithm 1, where `rnd(0,1)` is a random number generated with uniform probability from

**Algorithm 1:** The pseudo-code describing the horizontal expansion of our sampling methodology.

```
$iter \leftarrow 1$ ;
while at least one incomplete neutral network exists in $S$ and $|S| < S_{max}$ do
    $\mathcal{N} \leftarrow$ set of incomplete networks in $S$ of size less than $I_{min}$ ;
    forall the $N \in \mathcal{N}$ do
        forall the $i \in N$ do
            forall the $j \in \mathcal{V}(i)$ do
                if rnd$(0, 1) < \beta(f(i), f(j), iter)$ and $|S| < S_{max}$ then
                    $S \leftarrow S \cup \{j\}$ ;
                end
            end
        end
    end
    $iter \leftarrow iter + 1$ ;
end
return $S$ ;
```

the range $(0, 1)$, *iter* is a variable containing the number of iterations that have been executed and $\beta(f(i), f(j), iter)$ is defined as follows:

$$\beta(f(i), f(j), iter) = \begin{cases} 1 & \text{if } f(i) = f(j), \\ k^{-iter} & \text{otherwise} \end{cases} \tag{2}$$

where $k$ is a constant that has to be chosen in such a way that probability $\beta$ decreases "quickly enough" with iterations (in this work, $k = 4$). Horizontal expansion stops when the sample reaches the maximum size $S_{max}$ or when an iteration

| | $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ | $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$ |
|---|---|---|
| No. of individuals | 1,446 | 5,552 |
| No. of optimal solutions | 8 | 660 |
| No. of neutral networks | 31 | 1,389 |
| Average network size | 46.64 | 3.99 |
| Median of network sizes | 9 | 1 |

Table 3: Some characteristics of the "small" fitness landscapes for the even-2 parity problem that we have exhaustively studied.

does not add any new individual. This algorithm is very useful to study neutrality, but it has some bias: for instance, a large neutral network could be represented in our sample by many smaller ones. It is the case, for instance, of the central network for the $\mathcal{L}^{\{\text{XOR,NOT}\}}$ even parity landscape (as we show in Section 5.1.2). However, in this particular case, this is not a problem, since we are aware of the unicity of the central network because of the theoretical results presented in Section 3.1 (Property (a)). Those theoretical results should contribute to the understanding of the real shape of the fitness landscapes under study.

## 5. Experimental Results

In this section, we present our new measures that describe various aspects related to the neutrality of fitness landscapes. The new measures are presented by discussing the distribution of their values on the fitness landscapes that have been introduced in Section 3. In particular, in Section 5.1, we studied 4 fitness landscapes of the *even parity* problem, while in Section 5.2, we studied 4 landscapes of the *multiplexer* problem.

### *5.1. Even-k Parity Problem*

The main aim of this section is to discuss the aspects related to neutrality of 4 fitness landscapes of the even parity problem and how they possibly determine the problem difficulty for the GP. Neutrality of the fitness landscapes is captured by a set of new measures that are here introduced and defined. In the first part (Section 5.1.1) we exhaustively studied 2 small fitness landscapes, while in the second part (Section 5.1.2) we investigated 2 larger fitness landscapes by means of samples obtained by our new sampling methodology. As discussed in the following, the results obtained for the "small" and "large" landscapes consistently support the existence of a relationship between neutrality and problem difficulty and, moreover, they empirically support the soundness of our sampling methodology.

#### *5.1.1. Exhaustive analysis of a "small" landscape*

The first step of our study is to investigate a fitness landscape of small size, in order to be able to exhaustively generate all the possible individuals contained in it. We have built it by considering the even-2 parity problem and trees with a maximum depth equal to 3. The resulting $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$ and $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ landscapes both contain at least one perfect solution. In Table 3 some characteristics of these fitness landscapes are reported. In agreement with the theoretical observations presented in Section 3.1, $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$ has a large number (1388) of neutral networks at fitness 0 and 1 composed by only one individual and one large (4164 individuals) central network at fitness 0.5. On the other hand, $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ has smaller size and it has few networks, all of them medium-sized. Figures 1 and 2 are a graphical representation of the neutrality graphs of $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ and $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$ respectively (see Section 2.1 for the definition of neutrality graph). Each square represents a neutral network, and its size is proportional to the logarithm of the network size. The node color indicates the fitness value of the network (from 1, black, to 0, which represents the best possible fitness, white).

The first parameter we study is the *average neutrality ratio*, $\bar{r}$. It is defined as the mean of the neutrality ratios (as defined in Section 2.1) of all the individuals in a network. High values $\bar{r}$ (near to 1) correspond to a large amount of neutral mutations. Figure 3 presents the scatterplot of $\bar{r}$ against fitness for each one of the neutral networks in the two landscapes. In this figure, as in all the subsequent ones, to guide the eye, a gray line is drawn, joining all the average points for each considered fitness value. These averages have been weighted according to the size of networks representing each point. Furthermore, points at the same coordinates have been artificially (slightly) displaced, so that they can be distinguished. In $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$, the central network (fitness equal to 0.5) has high values of $\bar{r}$, while for the

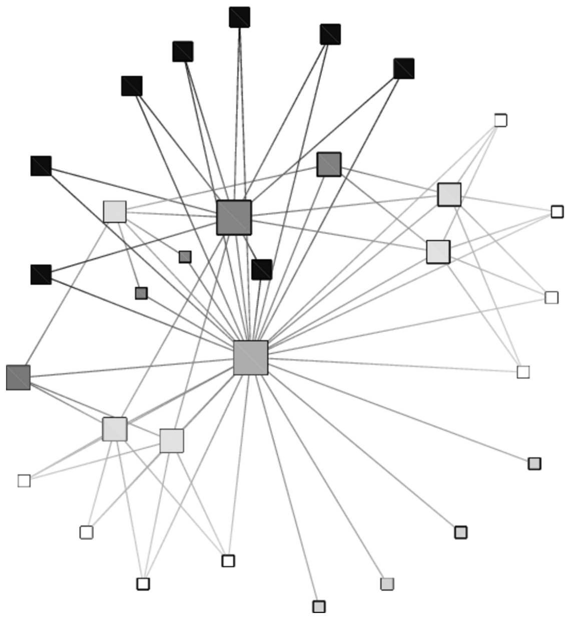
Figure 1: Graphical representation of the neutrality graph of $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$ for the even-2 parity problem.

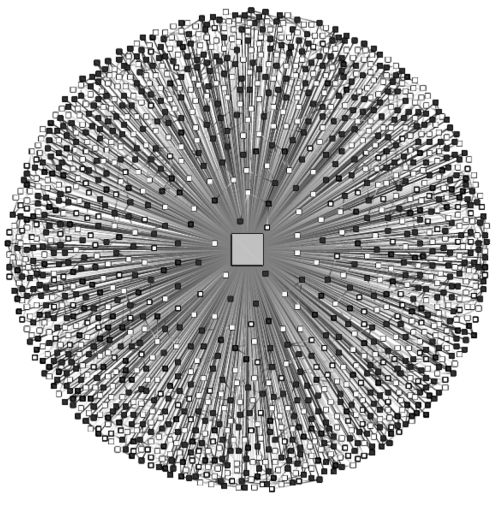
Figure 2: Graphical representation of the neutrality graph of $\mathcal{L}_{(2,3)}^{\{\text{XOR},\text{NOT}\}}$ for the even-2 parity problem.

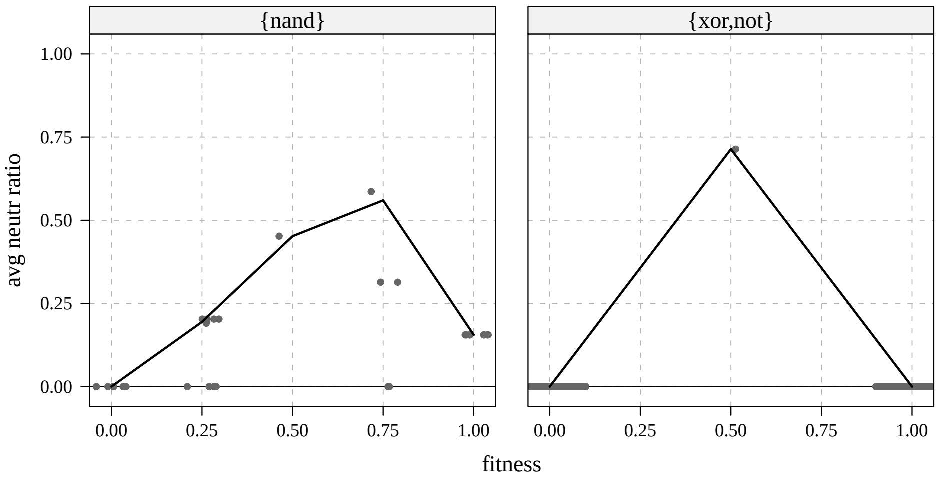

Figure 3: Scatterplot between fitness and average neutrality ratio in $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$ (left part) and $\mathcal{L}_{(2,3)}^{\{\text{XOR,NOT}\}}$ (right part) for the even-2 parity problem.

other networks $\bar{r} = 0$. Furthermore, the scatterplot is nearly symmetrical around fitness equal to 0.5. In $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$ $\bar{r}$ values are, on average, larger than 0.2 for some bad fitness values (fitness equal to 0.75) and smaller than 0.2 for good ones (fitness equal to 0 and 0.25): in general, in $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$ networks with bad fitness seem to be "more neutral" than networks with good fitness.

The second measure we study is the *average* $\Delta$*-fitness* of the neutral networks. This measure is the average fitness change (positive or negative) achieved after a mutation of the individuals belonging to the network. Formally, let $N$ be a neutral network, then its *average* $\Delta$*-fitness* can be defined as:

$$\Delta\bar{f}(N) := \frac{1}{|N|} \cdot \sum_{s \in N} \left[ \frac{\sum_{v \in \mathcal{V}(s)} (f(v) - f(s))}{|\mathcal{V}(s)|} \right]$$

This measure is clearly related to the notions of evolvability [29] and innovation rate [38]. A negative value of $\Delta\bar{f}$ corresponds to a fitness improvement (because it reduces the error) while a positive one corresponds to a worsening (because it increases the error). As Figure 4 shows, in $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$ the possible values of $\Delta\bar{f}$ are included into a narrower range than in $\mathcal{L}_{(2,3)}^{\{\text{XOR,NOT}\}}$. We conclude that in $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$ mutations cannot produce large fitness improvements (on average).

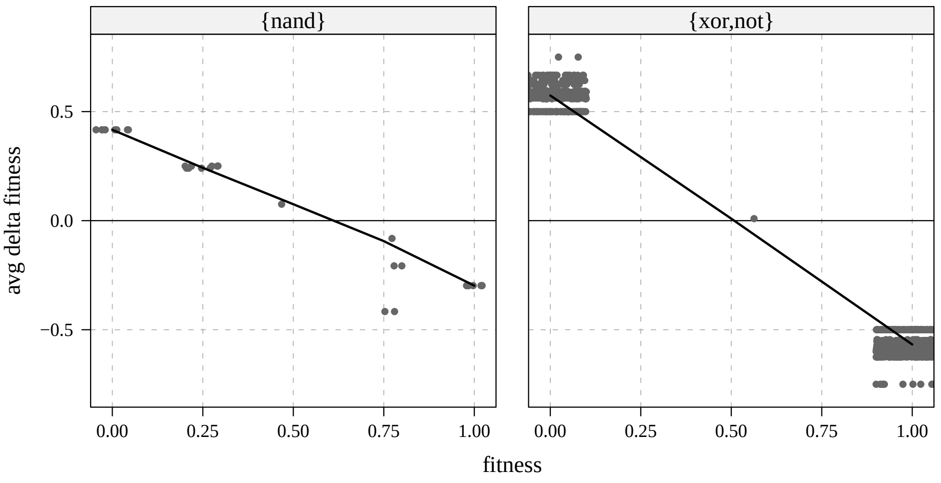

Figure 4: Scatterplot between fitness and average $\Delta$-fitness in $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$ (left part) and $\mathcal{L}_{(2,3)}^{\{\text{XOR,NOT}\}}$ (right part) for the even-2 parity problem.

Thus, to solve the problem, GP has to find individuals with many different fitness values. This is not the case for $\mathcal{L}_{(2,3)}^{\{\text{XOR,NOT}\}}$, where a mutation of an individual contained into the central network can produce an individual with a fitness equal to 0 (global optimum). Furthermore, in $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$ good fitness networks (fitness equal to 0.25 or 0.5) have positive values of $\Delta\bar{f}$. In other words, in $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$, it is unlikely that mutations of good individuals generate better offspring.

Now, we present two measures that we have called *Non-Improvable (NI) Solution ratio* ($r_{ni}$) and *Non-Degradable*[1] *(ND) Solution ratio* ($r_{nd}$). The first one is defined as the number of non-improvable solutions, or non-strict local optima (i.e., individuals $i$ that cannot generate offspring $j$ by applying a *StSt* mutation such that the fitness of $j$ is better than the fitness of $i$) that are contained in a network divided by the size of the network. The second one is the ratio of the individuals $i$ that cannot generate offspring $j$ (by applying a *StSt* mutation) such that the fitness of $j$ is worse than the fitness of $i$. Figures 5 and 6 present the scatterplots of $r_{ni}$ and $r_{nd}$ for each fitness value, respectively. *NI* solution ratio is

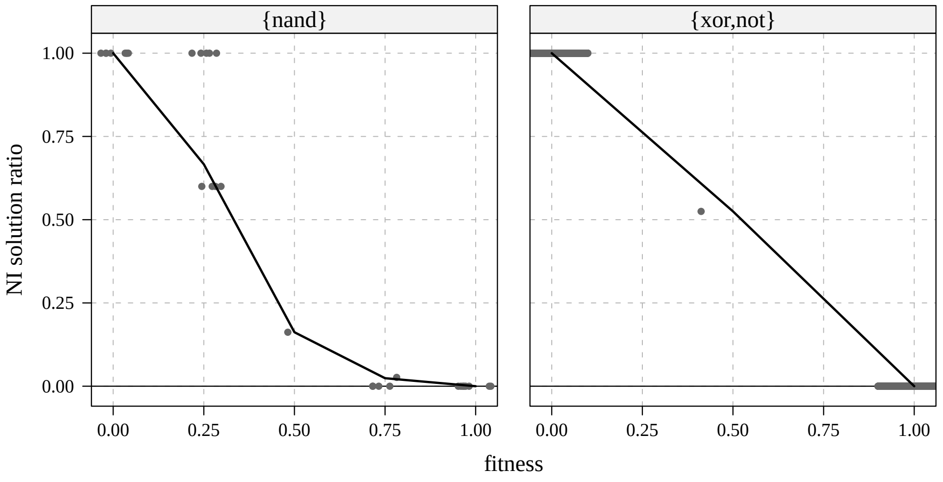

Figure 5: Scatterplot between fitness and *NI* solution ratio in $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$ (left part) and $\mathcal{L}_{(2,3)}^{\{\text{XOR,NOT}\}}$ (right part) for the even-2 parity problem.

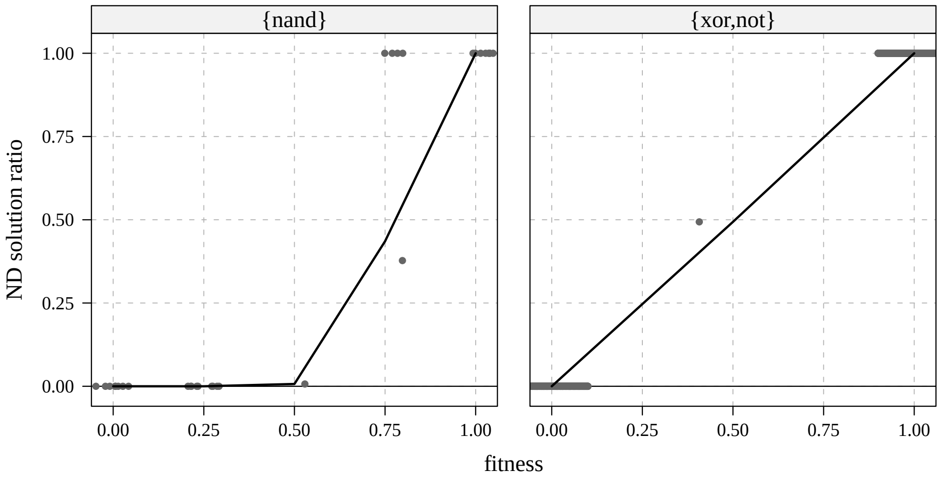

Figure 6: Scatterplot between fitness and *ND* solution ratio in $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$ (left part) and $\mathcal{L}_{(2,3)}^{\{\text{XOR,NOT}\}}$ (right part) for the even-2 parity problem.

1 in 0-networks (they are composed of optimal solutions, so they cannot further improve) and it is 0 in 1-networks.

[1]We are aware that the word "degradable" is normally used to indicate something different ("capable of being chemically degraded" from the English dictionary). Nevertheless we use it here as a contrary of "improvable", i.e. as something that cannot get worse.

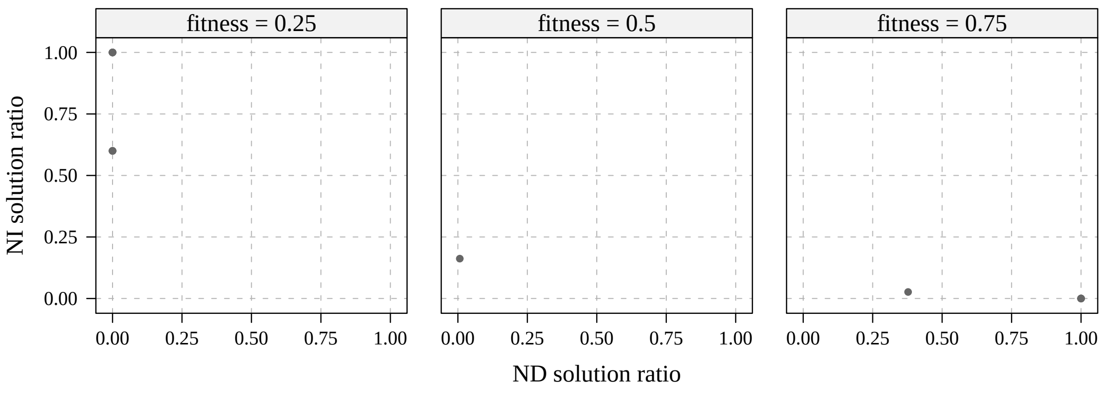

Figure 7: Scatterplot between *ND* and *NI* solution ratio in $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ for the even-2 parity problem.

| Fitness Value | *NI* solution ratio | *ND* solution ratio |
|---|---|---|
| 0 | 1 | 0 |
| 0.5 | 0.5156 | 0.4844 |
| 1 | 0 | 1 |

Table 4: *NI* solution ratios and *ND* solution ratios in $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$ for the three possible fitness values.

Analogously, *ND* solution ratio is 1 in 1-networks and it is 0 in 0-networks. In $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$, there are some good networks (low fitness) with high $r_{ni}$ values. At fitness 0.25, all the networks have a high value of $r_{ni}$ (larger than 0.6) and 5 of them (over a total of 9 networks) have a value of $r_{ni}$ equal to 1 and thus they are plateaus of non-strict local optima. We call these networks *trap networks*, since their individuals cannot generate better offspring and thus once GP has reached these networks, it cannot escape from them by means of a *StSt* mutation improving fitness. Trap networks do not exist in $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$.

Finally, we study the *ND* solution ratio against *NI* for relevant fitness values in $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ (Figure 7) and for all the possible fitness values in $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$ (Table 4). The scatterplot at fitness values equal to 1 and 0 for $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ are not reported to save space. However, they are obviously identical to the case of $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$ reported in Table 4.

In $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$, all the points are approximately placed along the segment $((1,0),(0,1))$. In $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$, the points are approximately positioned on the Cartesian axes and networks located at good fitness values have a large number of *NI* solutions. Thus, it is unlikely to mutate their individuals generating better offspring. This is not the case for $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$.

As a partial conclusion, we claim that none of the measures presented until now is able to completely justify the reason why the $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ landscape is hard for GP, while $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$ is easy. Nevertheless, each one of them gives a partial explanation and considering them all together, we can conclude that: (1) *StSt* mutations can produce larger fitness improvements in $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$ than in $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$; (2) Good individuals are harder to improve for $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ than for $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$; (3) $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ contains some "trap networks" at good fitness, which is not the case for $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$. Even though these are not formal proofs that $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ is harder than $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$, these are at least strong evidences, and all of them are based on the concept of neutrality. A bond between neutrality (expressed by our neutrality measures) and GP performance seems to exist. Below, we investigate the existence of that relationship for a larger landscape, studied by means of samples.

### *5.1.2. Analysis of larger landscapes by means of samples*

The largest even parity search spaces that we have been able to study correspond to the even-4 parity problems using trees of a maximum depth equal to 8. We indicate with $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(4,8)}$ and $\mathcal{L}^{\{\text{NAND}\}}_{(4,8)}$ the landscapes using {XOR, NOT} and {NAND} as function sets respectively. Both these spaces contain optimal solutions for the even-4 parity. Nevertheless, for $\mathcal{L}^{\{\text{NAND}\}}_{(4,8)}$ they are difficult to automatically generate (either by a Metropolis algorithm or by GP, as we have empirically shown in Section 3.1). Thus, if we want to sample all the possible fitness values, one feasible solution is to manually add one of them to the $S$ sample that is given as input to the vertical expansion phase. For the same reason, we have manually added to $S$ an individual with the worst possible fitness (fitness equal to 1). Table 5 summarizes the parameters used to generate the samples of the two landscapes (upper part) and some data about the samples that have been generated by our algorithm (lower part). It is particularly interesting to remark that our algorithms have generated

| | {NAND} | {XOR, NOT} |
|---|---|---|
| $p_{min}$ for Modified Metropolis | 0.005 | |
| $p_{min}$ for vertical expansion | 0.00005 | |
| $k$ for horizontal expansion | 4 | |
| Minimal size of an incomplete network | 2 | |
| Sample size of Modified Metropolis | 3 | 10 |
| $L$ of vertical expansion | 10 | 100 |
| Size of generated sample | 794,191 | 968,423 |
| No. of networks contained into the sample | 22,261 | 32,012 |
| Average network size | 35.68 | 30.25 |
| Median of network sizes | 35 | 31 |

Table 5: Parameters used to sample the $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(4,8)}$ and $\mathcal{L}^{\{\text{NAND}\}}_{(4,8)}$ landscapes for the even-4 parity problem.

a higher number of neutral networks for $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(4,8)}$ (32,012) than for $\mathcal{L}^{\{\text{NAND}\}}_{(4,8)}$ (22,261). This reflects the structures of the "small" landscapes studied in Section 5.1.1, where we have shown that the number of neutral networks in $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(2,3)}$ is higher than the one in $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ (see for instance Figures 1 and 2).

Figure 8 is an histogram of the fitness distributions and shows that our samples have covered the whole range of possible fitness values for the two landscapes. We remark that the principal aim of our sampling methodology is to cover a wide range of different fitness values, and it is not able to generate optimal solutions for landscapes that are more complex than the ones considered in this work. In particular, the sample of $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(4,8)}$ has 930 individuals with fitness 0 and 954 individuals of fitness 1. It is important to remind that each fitness landscape built using XOR and

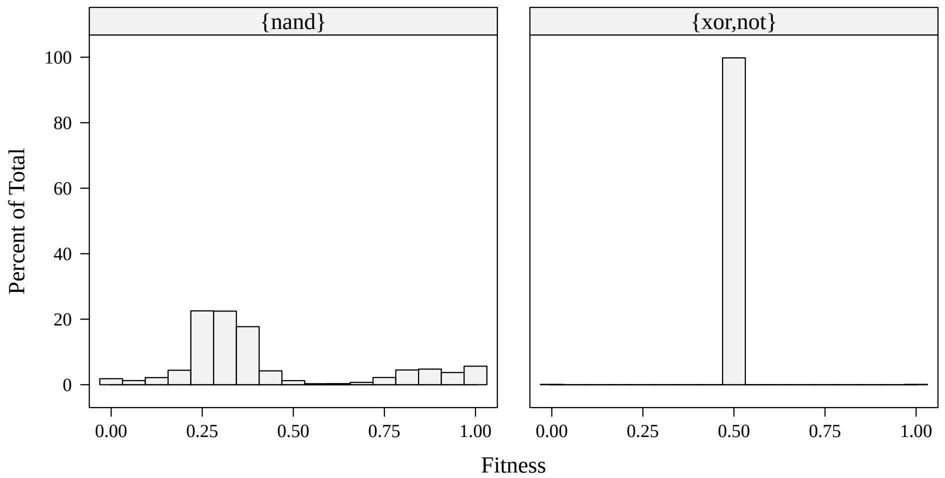

Figure 8: Frequency distribution of fitness in the sampled $\mathcal{L}^{\{\text{NAND}\}}_{(4,8)}$ (left part) and $\mathcal{L}^{\{\text{XOR,NOT}\}}_{(4,8)}$ (right part) for the even-4 parity problem.

NOT can only contain individuals of fitness 0, 0.5, or 1 (as formally proven in [24, 36]). Furthermore, we remark that many individuals of fitness different from 0.5 have been generated by means of our sampling technique. We would not have been able to study those individuals if we had used a uniform random sampling or a standard Metropolis-Hastings technique. A dissertation about the fact that with uniform random sampling and with standard Metropolis-Hastings techniques only a limited number of individuals with a different fitness from 0.5 can be generated is contained in [11].

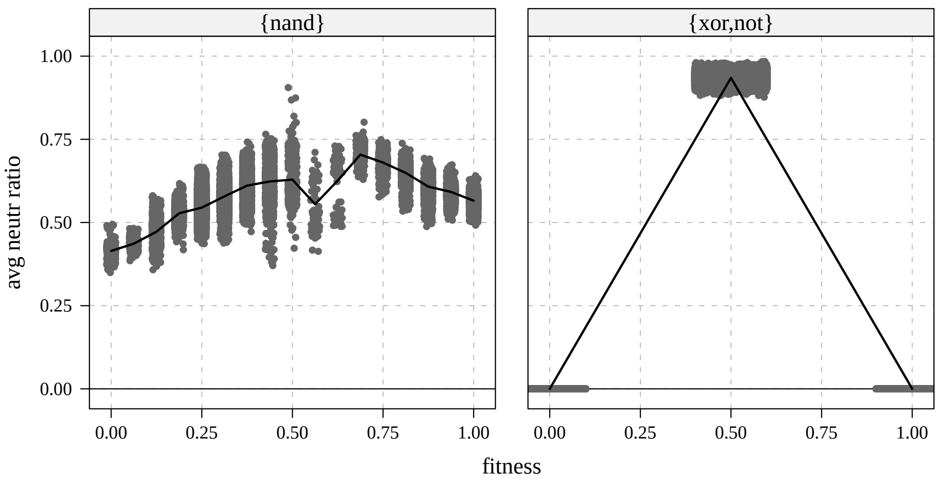

Figure 9: Scatterplot between fitness and average neutrality ratio in the sampled $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$ (left part) and $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$ (right part) for the even-4 parity problem.

In Figure 9, we present the *average neutrality ratios* ($\bar{r}$) results. The ratios calculated over the sample of $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$ are not affected by the presence of multiple 0.5-networks (caused by the bias of our sampling methodology) instead of having only one central network: all the ratios of these networks are close to the "large" single one observed for the even-2 parity (Figure 3). Furthermore, as for $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$ (Figure 3), also in $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$ the networks with good fitness values have a lower $\bar{r}$ than ones with bad fitness values. In other words, the networks with good fitness in $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$ seem to be "less neutral" than ones with bad fitness.

The scatterplot of the *average* $\Delta$*-fitness* is shown in Figure 10. In $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$ this scatterplot reflects the behavior

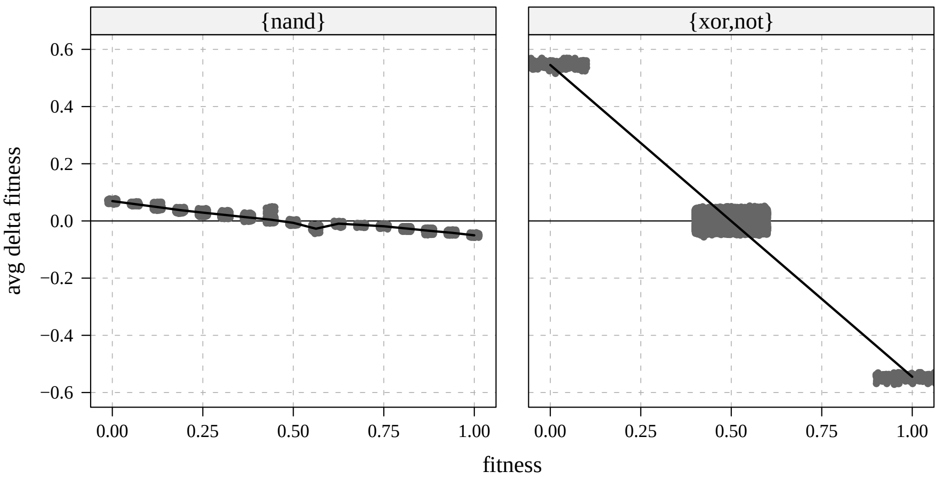

Figure 10: Scatterplot between fitness and average $\Delta$-fitness in the sampled $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$ (left part) and $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$ (right part) for the even-4 parity problem.

observed for the even-2 parity (Figure 4), whereas in $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$ it varies over a more limited range of average $\Delta$-fitness values. Our interpretation is that it is more difficult to significantly improve a solution in $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$ than in $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$ because the majority of the mutations generate solutions with similar fitness. Thus the optimum in $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$ can be found by GP only generating individuals of many different fitness values, i.e. GP cannot perform "large jumps" as in $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$. Remark that in the $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$ landscape, GP can *only* perform large jumps: the only possible fitness values are 0, 0.5 and 1, with no intermediate fitness values. The reason why GP search performs very efficiently for this landscape, even though the "central" network at fitness 0.5 contains the large majority of the individuals, is that the "central" network is very well connected to the other networks, at different fitness values, as we explained above. Thus, "large jumps" are frequent in $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$. Here, we have shown that they are not so frequent in $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$.

The scatterplot of *NI* and *ND* solution ratios (Figures 11 and 12 respectively) present some differences with respect to the ones observed for the landscape studied exhaustively (Figures 5 and 6), especially for $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$. Nevertheless, as

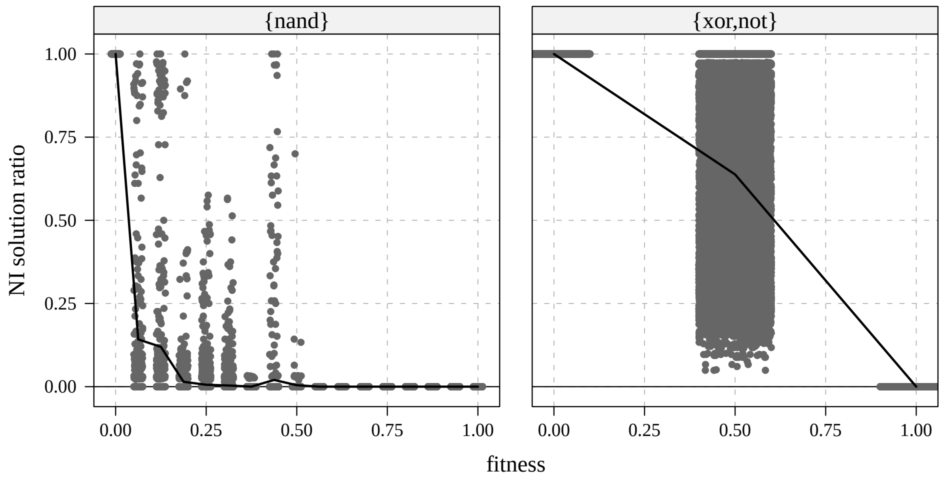

Figure 11: Scatterplot between fitness and *NI* solution ratio in the sampled $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$ (left part) and $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$ (right part) for the even-4 parity problem.

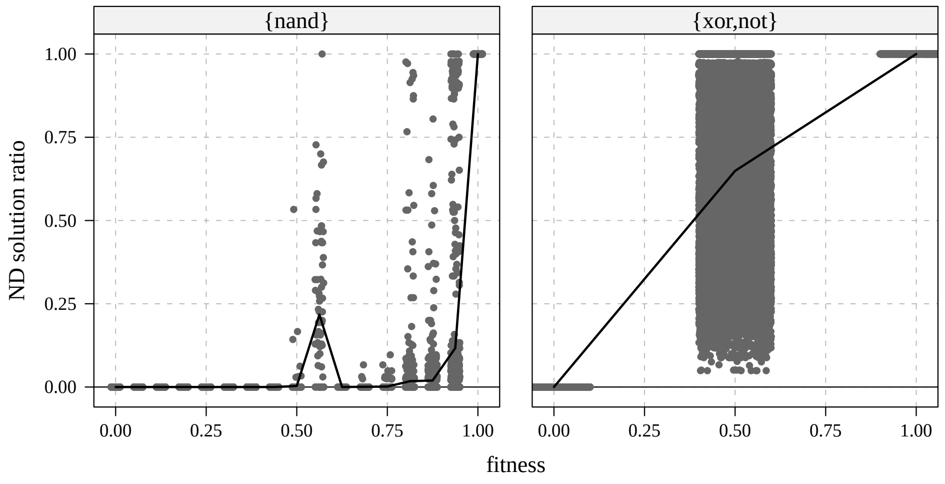

Figure 12: Scatterplot between fitness and *ND* solution ratio in the sampled $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$ (left part) and $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$ (right part) for the even-4 parity problem.

in $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$, networks with good fitness contain a large number of *NI* solutions (trap networks), which confirms that, in $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$, it is unlikely that mutating individuals belonging to good fitness neutral networks will generate better offspring. The differences for $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$ can likely be imputed to the sampling algorithm, that breaks down the central network in many smaller ones.

To save space, we do not show the mutual correlation scatterplots between *ND* and *NI* solution ratios for $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$ and $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$. Nevertheless, we can point out that these results are very similar to the ones obtained for the "small" landscapes shown in Figure 7 and Table 4. In particular, in the sample of $\mathcal{L}_{(4,8)}^{\{\text{XOR,NOT}\}}$, as in $\mathcal{L}_{(2,3)}^{\{\text{XOR,NOT}\}}$, the 0.5-networks are approximately placed above the segment $((0,1),(1,0))$. Furthermore, in the sample of $\mathcal{L}_{(4,8)}^{\{\text{NAND}\}}$, as in $\mathcal{L}_{(2,3)}^{\{\text{NAND}\}}$, the scatterplots of networks with fitness values smaller than 0.5 are approximately parallel to the $y$-axis and those of networks with larger fitness values are approximately parallel to the $x$-axis. Thus, as for $\mathcal{L}^{\{\text{NAND}\}}_{(2,3)}$ (Section 5.1.1), also in $\mathcal{L}^{\{\text{NAND}\}}_{(4,8)}$ networks with good fitness values have a large number of *NI* solutions and thus it is unlikely to escape from them mutating their individuals, which is not the case for $\mathcal{L}^{\{\text{XOR},\text{NOT}\}}_{(4,8)}$.

The conclusions of this section are exactly the same as the ones of Section 5.1.1: studying neutrality is helpful for understanding some characteristics of the fitness landscape and help us to make some inference on the difficulty of the problem. The proposed neutrality measures can be used for this goal.

### 5.2. *k-Multiplexer Problem*

In this section, as for the even parity problem, we first study two "small" fitness landscapes by considering the their whole sets of individuals (Section 5.2.1), and we then investigate "large" landscapes by means of samples (Section 5.2.2).

#### 5.2.1. *Exhaustive analysis of a "small" landscape*

As we have done for the even parity, also for the multiplexer problem, we study two fitness landscapes with some precise characteristics: (1) they have to be small enough to allow us to exhaustively generate all the possible solutions, at least as a first step; (2) they must contain at least one perfect solution for the problem; (3) their difficulties for GP must be different (generally speaking, one of them must be "easy" and the other one must be "hard"). To match all these requirements, we have initially set the problem order to $k = 3$ (in this way, expressions can be built using only three possible terminal symbols) and we have decided to choose the two sets of operators {NAND} and {IF}. Many other sets of operators for which some perfect solutions could be found might have been chosen. For instance, one might have studied the well known and mostly used set {AND, OR, NOT}, but in this way, the resulting fitness landscape would have been larger and studying it exhaustively would have been difficult. On the other hand, considering sets composed by one operator keeps the landscapes small enough while containing perfect solutions. In particular, if we consider the {IF} set (where $\text{IF}(x, y, z)$ returns $y$ is $x$ is equal to *true* and $z$ otherwise), the perfect solution for the 3-multiplexer is straightforward and one perfect solution for the 6-multiplexer can be found in a very simple way by composing some solutions to the 3-multiplexer (3 nested IF operations). On the basis of these considerations, and also of the experiments shown in Section 3.2 we can say that the multiplexer problem (in particular for reduced orders like $k = 3$ or $k = 6$) is easy to solve for GP if we consider $\mathcal{F} = \{\text{IF}\}$, while it is hard if we consider $\mathcal{F} = \{\text{NAND}\}$.

The largest fitness landscapes that we have been able to exhaustively study are composed of trees of a maximum depth equal to 3 when we have considered the {NAND} set of functions and equal to 2 for {IF} (the NAND operator is binary, while IF has three arguments, thus individuals built by IF are larger; this is the reason why we need to keep depth more limited). Table 6 reports some of the most important characteristics of the two landscapes induced by these operators, i.e. $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ and $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$. If we compare these results with the ones of Table 3 (exhaustive study of small landscapes for the

| | $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ | $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ |
|---|---:|---:|
| No. of individuals | 21,612 | 27,003 |
| No. of optimal solutions | 24 | 539 |
| No. of neutral networks | 39 | 17 |
| Average network size | 554.15 | 1,588.41 |
| Median of network sizes | 9 | 4 |

Table 6: Some characteristics of the "small" fitness landscapes of the 3-multiplexer problem that we have exhaustively studied.

even parity problem), the most apparent difference is that for the multiplexer problem the number of neutral networks is, in general, smaller than the one for the even parity. This observation can be explained by the fact that neighborhoods are larger for the multiplexer, since the cardinality of the set of terminals is larger (we have considered a problem order $k = 3$ for the multiplexer problem, against $k = 2$ for the even parity). Furthermore, IF has three arguments and this contributes to have even larger neighborhoods. As a consequence, it is easier (or more likely) to find at least one neutral neighbor. Thus neutral networks for the multiplexer are larger than for the even parity. We also remark that, even though the depth limit is equal to 2 for $\mathcal{L}^{\{\text{IF}\}}$ and to 3 for $\mathcal{L}^{\{\text{NAND}\}}$, the number of individuals in $\mathcal{L}^{\{\text{IF}\}}$ is larger than the one in $\mathcal{L}^{\{\text{NAND}\}}$. Finally, $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ has a much larger number of perfect solutions than $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$.

The distributions of fitness values in these two landscapes are reported in Figure 13. In $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ the majority of

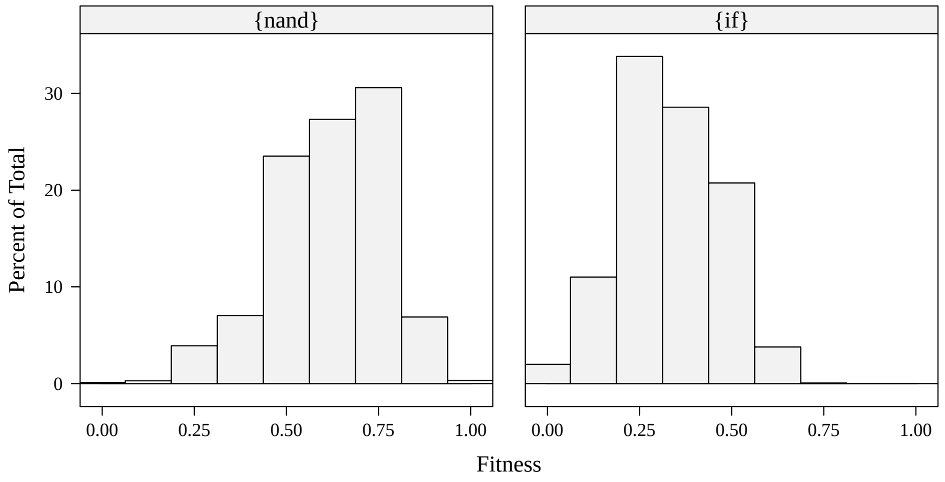

Figure 13: Frequency distribution of fitness in $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ (left part) and $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ (right part) for the 3-multiplexer problem.

individuals have a fitness value equal to 0.25 and many individuals have a fitness equal or near to zero (optimal fitness); no individuals with a fitness value larger (i.e., worse) than 0.75 can be found in $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$. On the other hand, in $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ the majority of the individuals have a fitness equal to 0.75, i.e. they have a bad fitness value, and only a very small portion of the individuals have a fitness less then or equal to 0.25.

Neutrality ratio scatterplots for $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ and $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ are shown in Figure 14. Neutrality ratio is high for neutral

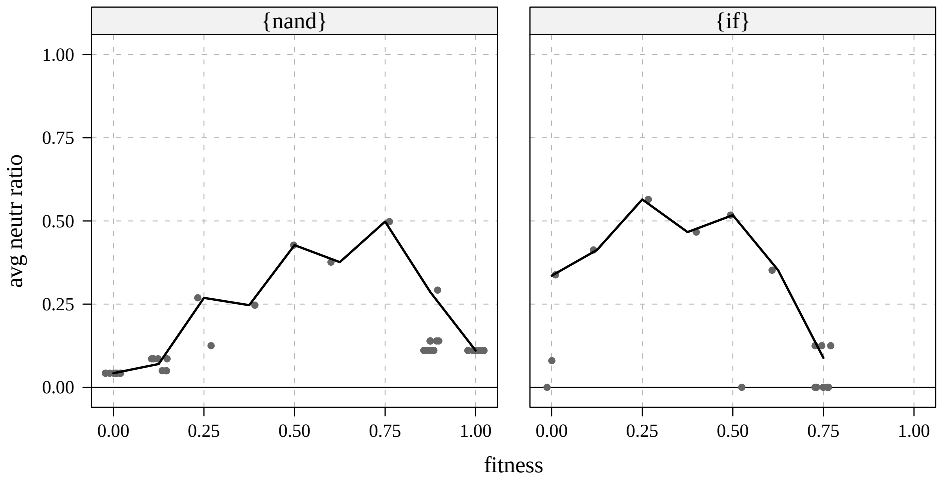

Figure 14: Scatterplot between fitness and average neutrality ratio in $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ (left part) and $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ (right part) for the 3-multiplexer problem.

networks at good fitness values and low for networks at bad ones in $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$, while the opposite holds for $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$.

Figure 15 shows the average Δ-fitness scatterplots. Even though for both landscapes individuals belonging to neutral networks at good fitness values are hard to improve (no negative values of the average Δ-fitness in correspondence to good fitness values), for $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ the values of Δ-fitness for fitness values near zero are larger than for $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$. Our interpretation is that for $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ improving good individuals is easier than for $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$.

Figure 16 shows the scatterplot of *NI* solution ratio. The most important difference between the scatterplots of the

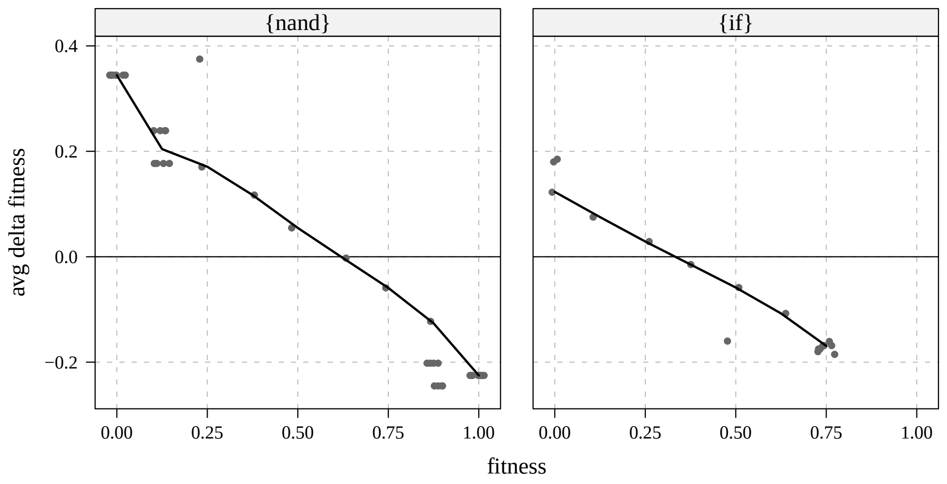

Figure 15: Scatterplot between fitness and average Δ-fitness in $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ (left part) and $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ (right part) for the 3-multiplexer problem.

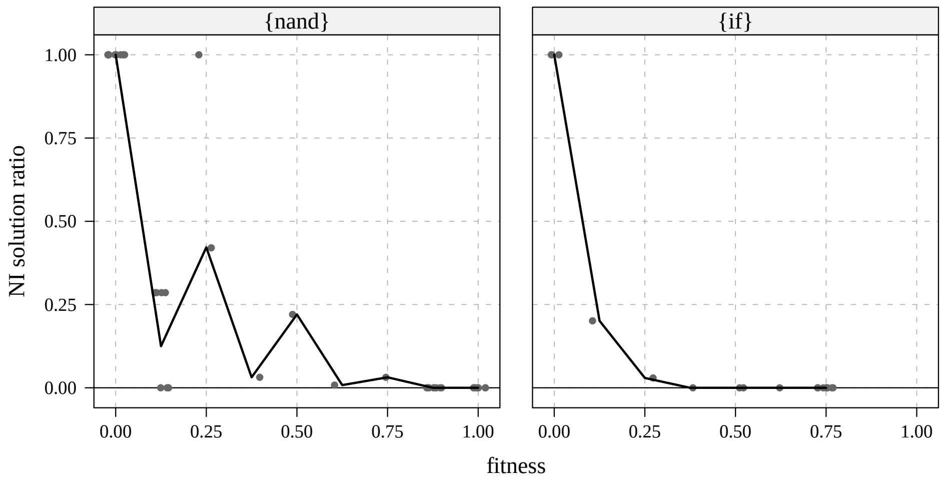

Figure 16: Scatterplot between fitness and *NI* solution ratio in $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ (left part) and $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ (right part) for the 3-multiplexer problem.

two fitness landscapes is that in the case of $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ there is a high number of *NI* solutions for fitness value equal to 0.25. This indicates the presence of some trap neutral networks at this fitness value. This is not the case for $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ where *NI* solution ratio at fitness 0.25 (and at good fitness values in general) is low (smaller than 0.2 for fitness equal to 0.125 and around 0.05 for fitness equal to 0.25).

We do not show here the scatterplots of *ND* solutions for $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ and $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$, even though we have generated and studied them. It is sufficient to point out that in $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ there are some good individuals that cannot generate worse offspring by means of mutation, which is not the case in $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$.

Figures 17 and 18 show the scatterplots of *NI* solution ratio against *ND* solution ratio for neutral networks at different fitness values for $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ and $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$, respectively. One different scatterplot is shown for three different fitness ranges for both landscapes: fitness between 0 and 0.5, fitness equal to 0.5 and fitness between 0.5 and 1. Since a large part of the individuals in the multiplexer problem share a fitness equal to 0.5, as discussed above, this fitness value deserves to be studied separately. On the other hand, we study the ranges of fitness values between 0 and 0.5 and between 0.5 and 1, meaning by them the ranges of good and bad fitness respectively. The fitness values 0 and 1 have not been included into these ranges, because their values of *NI* and *ND* solutions are obvious: for fitness equal to 0, both landscapes have only *NI* solutions (perfect solutions cannot be further improved) and for fitness equal to 1, both

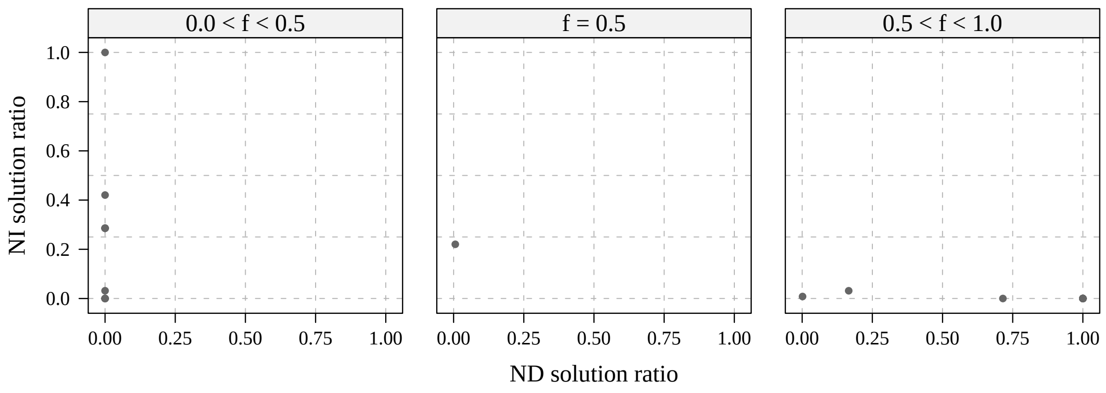

Figure 17: Scatterplot between *NI* and *ND* solution ratio in $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ for the 3-multiplexer problem.

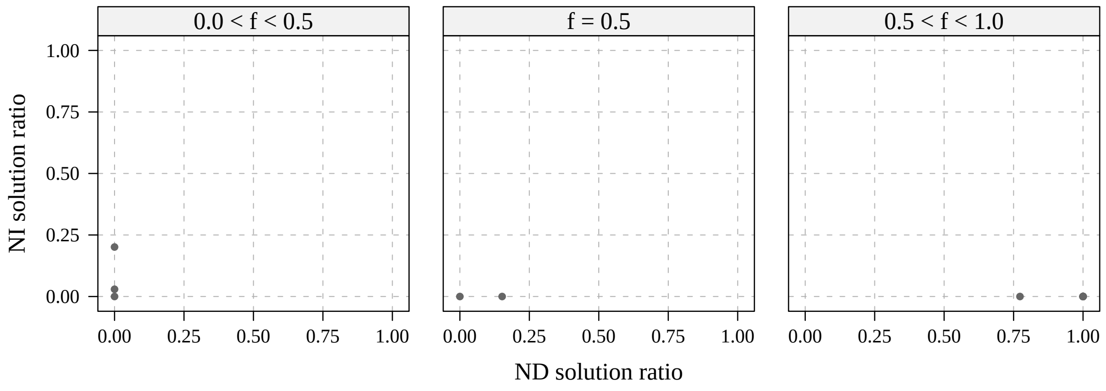

Figure 18: Scatterplot between *NI* and *ND* solution ratio in $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ for the 3-multiplexer problem.

landscapes have only *ND* solutions. For fitness values between 0 and 0.5 (that, we could informally say, represents the range of good, although not optimal fitness values), both landscapes have no *ND* solutions, but while $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$, for some fitness values, has a *NI* solution ratio equal to 1 (i.e. no solution can be improved), the maximum value of the *NI* solution ratio for $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ is approximately equal to 0.2. These results confirm that it is easier for GP to improve good solutions for $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ than for $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$. For fitness values between 0.5 and 1, no *NI* solution is present in the two landscapes, but while $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ has some networks with *ND* solution ratio equal to 0 and 0.25, $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ has only networks with *ND* solution ratio equal to 0.75 or 1.

Figure 19 shows the scatterplot of a new measure that we have called *profitable mutation ratio*: for each neutral network, we have calculated the number of mutations that generate better offspring and divided it by the total number of possible mutations of the individuals in that network. As Figure 19 clearly shows, the number of profitable mutations for $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ for fitness values ranging from 0 to 0.5 is much higher than for $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ for the same fitness range. Thus, it is much easier for GP to improve good solutions for $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ than for $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$.

To save space, we do not show the scatterplots of the profitable mutation ratios against the unprofitable ones for $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ and $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$; nevertheless, we point out that for fitness values between 0 and 0.5, $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ has a higher number of profitable mutations and a lower number of unprofitable ones than $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$.

All the results discussed in this section corroborate the hypothesis of the existence of a relationship between our neutrality measures and GP performance and give an indication of the fact that $\mathcal{L}^{\{\text{IF}\}}_{(3,2)}$ is easier than $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$ for GP.

### *5.2.2. Analysis of larger landscapes by means of samples*

In analogy with the study on the even parity problem (presented in Section 5.1.2), in this section we present a study of two landscapes of "large" size for the multiplexer problem. These two landscapes are respectively induced by the

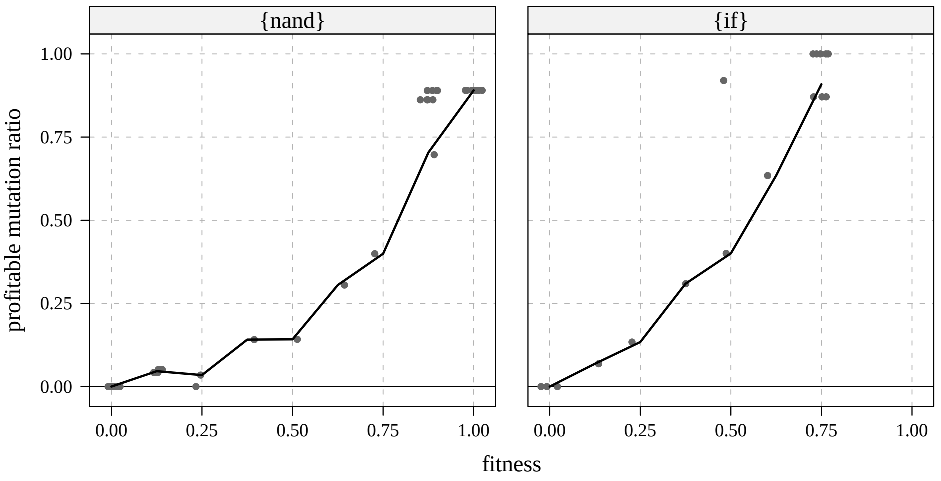

Figure 19: Scatterplot between fitness and profitable mutation ratio in $\mathcal{L}^{\{\textsc{nand}\}}_{(3,3)}$ (left part) and $\mathcal{L}^{\{\textsc{if}\}}_{(3,2)}$ (right part) for the 3-multiplexer problem.

{NAND} and {IF} sets of operators, as for the "small" landscapes that have been studied in Section 5.2.1. The sampling methodology we have used to study these landscapes is the same as the one presented in Section 4 and that has been used for the even parity problem. The larger search space induced by {NAND} that we have been able to study is the 6-multiplexer problem with a maximum depth equal to 6. In analogy with the terminology used above, we indicate with $\mathcal{L}^{\{\textsc{nand}\}}_{(6,6)}$ this landscape. The larger search space induced by {IF} that we have been able to study is the 6-multiplexer with a maximum tree depth equal to 5. This landscape will be indicated by $\mathcal{L}^{\{\textsc{if}\}}_{(6,5)}$. As for the limited size landscapes studied in Section 5.2.1, this difference in the tree depth limit for the two landscapes is due to the fact that NAND is an operator of arity 2 while IF is an operator of arity 3. Thus, given a fixed tree depth, the trees that can be built with IF are on average larger than the ones that can be built with NAND. Table 7 shows some of the most important characteristics of the samples of $\mathcal{L}^{\{\textsc{nand}\}}_{(6,6)}$ and $\mathcal{L}^{\{\textsc{if}\}}_{(6,5)}$. A remarkable difference between these two sampled landscapes is that $\mathcal{L}^{\{\textsc{nand}\}}_{(6,6)}$

| | {NAND} | {IF} |
|---|---|---|
| Tree-depth limit | 6 | 5 |
| $p_{min}$ for Modified Metropolis | $10^{-5}$ | |
| $p_{min}$ for vertical expansion | $10^{-7}$ | |
| $k$ for horizontal expansion | 4 | |
| Minimal size of an incomplete network | 5 | |
| Sample size of Modified Metropolis | 3 | 10 |
| $L$ of vertical expansion | 15 | 20 |
| Size of generated sample | 708,627 | 431,145 |
| No. of networks contained into the sample | 21,092 | 4,673 |
| Average network size | 33.60 | 92.26 |
| Median of network sizes | 32 | 50 |

Table 7: Parameters used to sample the $\mathcal{L}^{\{\textsc{if}\}}_{(6,5)}$ and $\mathcal{L}^{\{\textsc{nand}\}}_{(6,6)}$ landscapes for the 6-multiplexer problem.

has a larger number of neutral networks than $\mathcal{L}^{\{\textsc{if}\}}_{(6,5)}$. This happened also for the small sized landscapes that we have studied in Section 5.2.1 (see results reported in Table 6). Thus, this characteristic is probably not caused by a bias of our sampling methodology, but it is present in the actual (complete) $\mathcal{L}^{\{\textsc{nand}\}}_{(6,6)}$ and $\mathcal{L}^{\{\textsc{if}\}}_{(6,5)}$ landscapes. In other words, our sampling technique keeps the proportions of the number of neutral networks in $\mathcal{L}^{\{\textsc{nand}\}}_{(6,6)}$ and in $\mathcal{L}^{\{\textsc{if}\}}_{(6,5)}$ as they are in the real landscapes.

Figure 20 shows the fitness distributions for the two samples. For $\mathcal{L}^{\{\textsc{if}\}}_{(6,5)}$, all the sampled fitness values are included

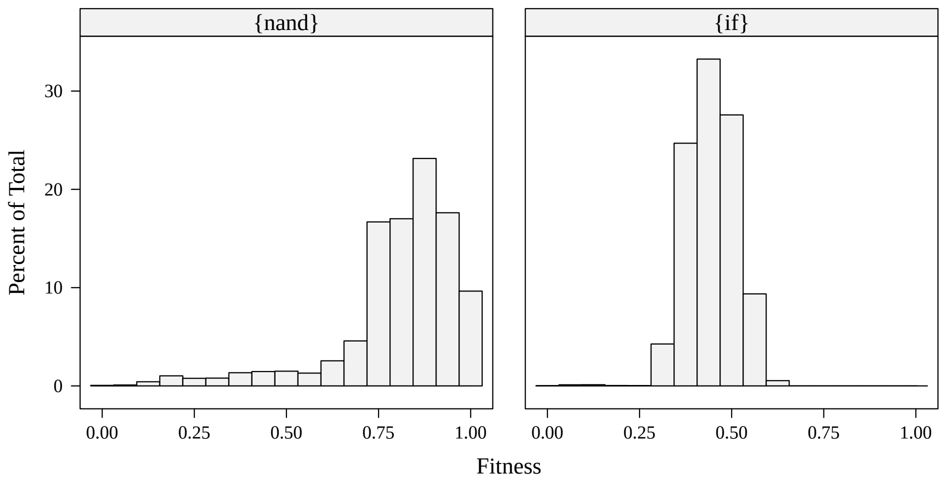

Figure 20: Frequency distribution of fitness in the sampled $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ (left part) and $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ (right part) for the 6-multiplexer problem.

into the range $[0, 0.625]$; in other words, no *bad* individual has been sampled. The same characteristic was present in the distribution of the "small" landscape shown in Figure 13, where no bad individuals (fitness larger than 0.75) exist and the number of individuals with a fitness value larger than 0.5 is low (smaller than 4% of the total number of individuals in the search space). Nevertheless, the shapes of the two distributions are not identical: our sampling technique differs from a uniform random sampling and has been designed to study some characteristics of the landscapes related to neutrality and not to exactly maintain the original distributions.

For $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$, a large part of the sampled individuals have a bad fitness value (included in the range $[0.75, 1]$), and this is similar to what happened in the histograms of the small size landscapes that we have studied exhaustively (see Figure 13), where the largest number of individuals had fitness equal to 0.75. Nevertheless, also in this case, the two distributions (Figures 13 and 20) are not identical; in particular, the sample of $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ generated by our technique contains fewer "good" individuals than the whole landscape $\mathcal{L}^{\{\text{NAND}\}}_{(3,3)}$. This is probably due to the fact that the hardness of the problem increases as the problem order increases, thus "good" individuals for the 6-multiplexer problem are harder to find (and to sample) than for the 3-multiplexer problem. Finally, we point out that with our sampling technique, as it was the case for the even parity problem, we have been able to generate individuals with many different fitness values. This would not have been possible if we had used a uniform random sampling or a standard Metropolis sampling.

Figure 21 reports the average neutrality ratio scatterplots for $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ (left part) and $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ (right part). As for the small landscape (results reported in Figure 14), also in this case $\mathcal{L}^{\{\text{IF}\}}$ has a higher neutrality ratio than $\mathcal{L}^{\{\text{NAND}\}}$ for networks at good fitness values. In particular for networks at fitness values around 0.25, the average neutrality ratio in $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ is approximately included between 0.3 and 0.5 and the average (gray line in figure) is around 0.45. For the same fitness values the average neutrality ratio in $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ is approximately included between 0.1 and 0.4 and the average value in about 0.2. If we consider networks with fitness values better than 0.25, the trend is even more marked: for networks at fitness 0.1, $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ has an average neutrality ratio around 0.4 while $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ has an average neutrality ratio approximately equal to 0.05. In conclusion, $\mathcal{L}^{\{\text{IF}\}}$ is "more neutral" than $\mathcal{L}^{\{\text{NAND}\}}$in good regions of the fitness landscape.

The average $\Delta$-fitness scatterplots are not reported here, but we have studied them and they confirm that improving good individuals for $\mathcal{L}^{\{\text{IF}\}}$ is easier than for $\mathcal{L}^{\{\text{NAND}\}}$, in fact for neutral networks at good fitness values the value of the average $\Delta$-fitness for $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ is positive and much larger than the one for $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ (this behavior is very similar to the one of the "small" landscapes shown in Figure 15).

The scatterplots of *NI* solution ratios are reported in Figure 22. $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ presents some *NI* solution ratios larger than 0.2 for some good fitness values (see for instance the peaks at fitness values approximately equal to 0.125, 0.375, and 0.5). This indicates the presence of some trap neutral networks at this fitness values. This is not the case for $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ where *NI* solution ratios are always equal to zero, except the obvious case of fitness equal to zero, where the *NI* solution ratio is, of course, equal to one.

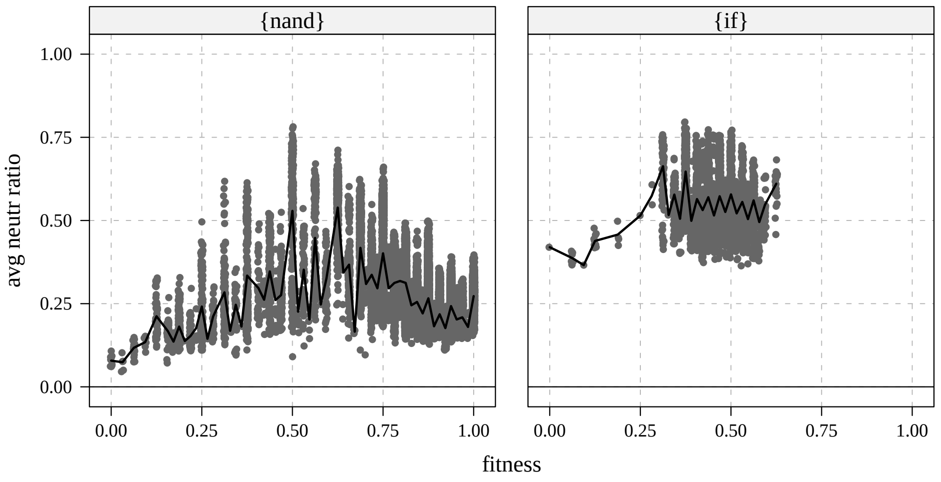

Figure 21: Scatterplot between fitness and average neutrality ratio in the sampled $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ (left part) and $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ (right part) for the 6-multiplexer problem.

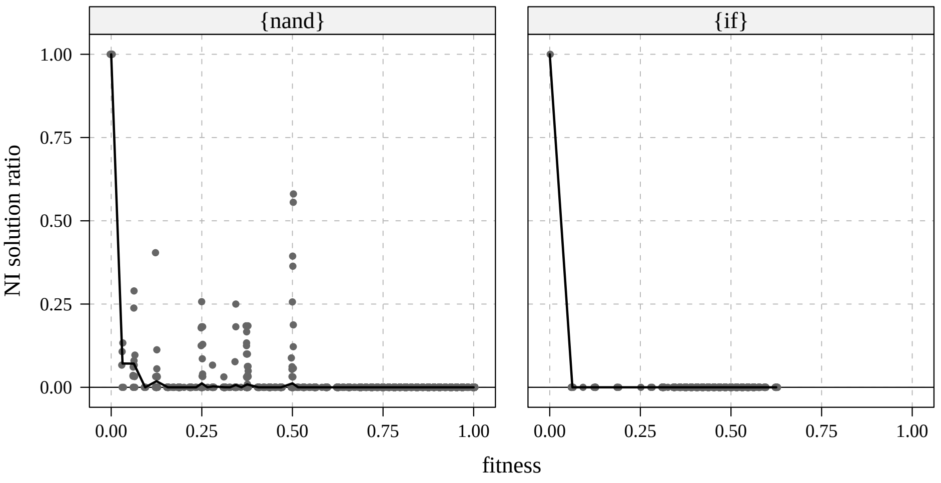

Figure 22: Scatterplot between fitness and *NI* solution ratio in the sampled $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ (left part) and $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ (right part) for the 6-multiplexer problem.

Figure 23 (Figure 24, respectively) shows the scatterplots of *NI* solution ratio against *ND* solution ratio for $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ ($\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$, respectively). As in Figures 17 and 18, one separate scatterplot is shown for three different fitness ranges for both landscapes: fitness values between 0 and 0.5, fitness equal to 0.5 and fitness values between 0.5 and 1. The fitness values 0 and 1 have not been included into these ranges. The most remarkable difference between the $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ and the $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ results can be seen in the respective scatterplots reporting results at fitness values between 0 and 0.5 (good, although not optimal fitness values). For these fitness values, both fitness landscapes contain no *ND* solution, but $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ has some neutral networks for which the *NI* solution ratio is higher than 0.25, while for $\mathcal{L}^{\{\text{IF}\}}_{(5,6)}$ all the neutral networks have an *NI* solution ratio equal to 0. In other words, for $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ some good individuals exist that cannot be improved by means of mutation, while this is not the case for $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$.

Figure 25 shows the scatterplot of unprofitable mutation ratios for $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ and $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$. Values of the unprofitable mutation ratios are higher for good fitness values in $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ than in $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$. In particular, for fitness values between 0 and 0.25, the majority of the possible mutations in $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ are unprofitable, while for $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ only about half of the possible mutations appear to be unprofitable. We do not show the scatterplots of profitable mutation ratios here, but they lead us

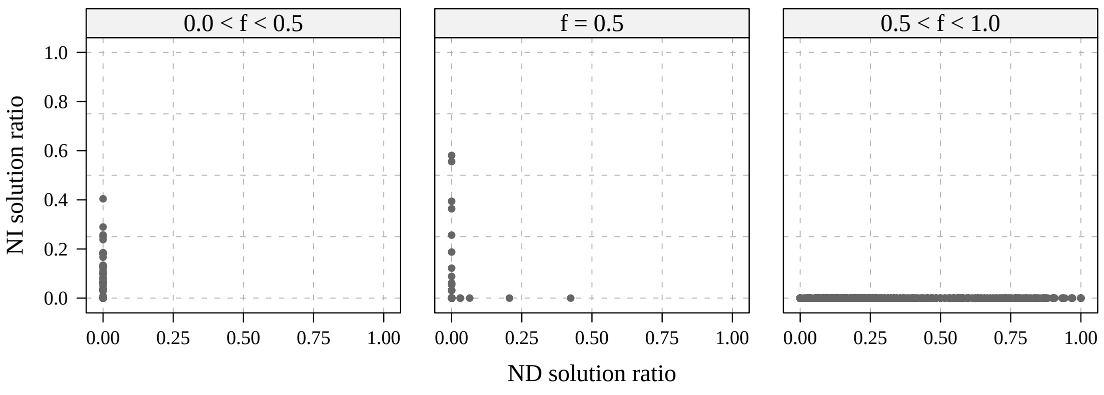

Figure 23: Scatterplot between *NI* and *ND* solution ratio in the sampled $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ for the 6-multiplexer problem.

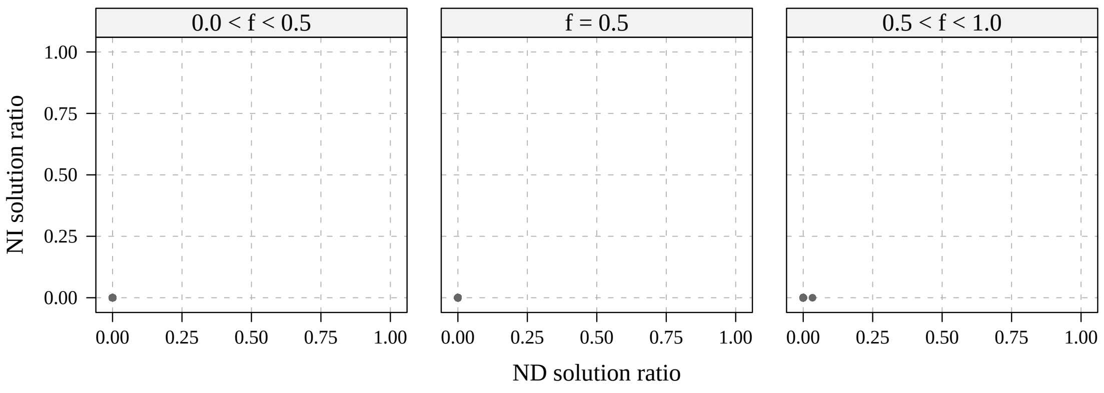

Figure 24: Scatterplot between *NI* and *ND* solution ratio in the sampled $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ for the 6-multiplexer problem.

to the same conclusions as the unprofitable ones.

Our conclusion is that all the neutrality measures that we have studied indicate that $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ is easier than $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ for GP.

### 5.3. *On the Generality of the Proposed Approach*

Neutrality within various landscapes can have complex properties and can deeply affect the performance of evolutionary algorithms. This complexity has contributed to the controversy surrounding neutrality and a proper study of such properties requires more nuanced tools. This paper is a contribution along this line. The proposed sampling technique is quite general. It can be used, in principle, for any fitness landscape and it has the property of sampling many different fitness values and to identify neutral networks at each different sampled fitness value. This property is important when using the proposed measures, which clearly work better if different fitness values are sampled. It is particularly useful for landscapes in which the set of possible fitness values is limited, like the Boolean ones. In this case, sampling many different possible fitness values is useful to have a view as wide as possible of the fitness landscape. In the case of continuous fitness functions, like real-valued symbolic regression, one should consider fixing a threshold to study neutrality, i.e. to consider two neighbors as neutral if the absolute value of their difference is smaller than a pre-fixed parameter.

The proposed neutrality measures are also quite general, but an important consideration needs to be done: in order to calculate the measures, one has basically to count some neighbors of the sampled individuals. The smaller the neighborhoods, the more the counted neighbors represent the whole neighborhood, and thus the more reliable the measures are. For this reason, it is suitable to use restricted mutation operators as the ones used in this work. If on the one side, it is possible to define such operators for any problem (as discussed in [11]), on the other hand it is also true

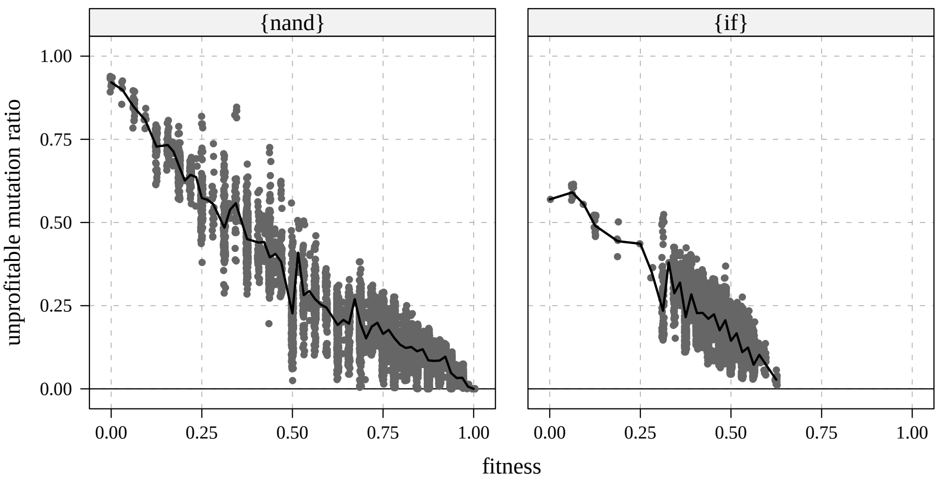

Figure 25: Scatterplot between fitness and unprofitable mutation ratio in the sampled $\mathcal{L}^{\{\text{NAND}\}}_{(6,6)}$ (left part) and $\mathcal{L}^{\{\text{IF}\}}_{(6,5)}$ (right part) for the 6-multiplexer problem.

that these operators are not the ones that are usually employed by GP practitioners and induce a different landscape. Attempts to formally define the neighborhood induced by standard GP subtree crossover and mutation have recently been done [39, 40, 41] and extending the present work using those results is one of our main research activities.

Once stated that these measures are general, in the sense that they can be calculated for any problem, the question remains open if they are informative for every possible fitness landscape. In particular, it is straightforward to remark that we have considered problem instances whose differences in the difficulties are rather marked. what happens when the differences in the difficulty of the landscape are finer? Are these measures precise enough to catch those small differences? Or similarly: how precise (fine) these measures are? The problem in answering to such a question is that the difficulty of a problem can depend on many factors, and not only on the limited characteristics bound to neutrality that we are interested in quantifying using our measures. For instance, we believe that if the cause of the different difficulties of two problems is the number of improvable solutions at good fitness levels, then our measure will be able to catch those differences even if they are small. If it is due to another cause, in particular if that cause is not bound to neutrality, like for instance a lack of correlation between fitness and distance to the goal, then our measures will probably fail to identify it (but this is not their target). What is important to remark here is that our measures are not intended to be difficulty measures, but, taken all together, their goal is just to offer a particular view of the fitness landscape. This view can be helpful in some cases.

## 6. Conclusions and Future Work

Some new characteristics of fitness landscapes related to neutrality have been defined in this paper and studied for different versions of the Boolean parity and multiplexer problems. In particular, we have defined: (*i*) the *average neutrality ratio* of a neutral network, which quantifies the amount of possible neutral mutations of its individuals; (*ii*) the *average* $\Delta$-*fitness* of a neutral network, which quantifies the average change in fitness achieved by mutating its individuals; (*iii*) the *non-improvable* (respectively "*non-degradable*") *solution ratio* of a neutral network, which quantifies the amount of solutions that cannot generate better (respectively worse) offspring in the network; (*iv*) the *profitable* (respectively *unprofitable*) *mutation ratio* of a neutral network, which quantifies the amount of mutations that generate better (respectively worse) offspring than their parents in the network.

Each measure, if considered separately, does not allow us to draw strong conclusions about fitness landscape difficulty because it only provides a narrow view of the landscape. But considered all together, they allowed us to have a clear picture of the characteristics of some particular, but important, tree-based GP Boolean function landscapes. Interestingly, all the studied measures always agree on the difficulty of the studied problems, motivating it from different viewpoints.

In synthesis, all these measures have made clear that the set of operators {XOR, NOT} induces an easier fitness landscape than {NAND} for the even parity problem and that the set of operators {IF} induces an easier fitness landscape than {NAND} for the multiplexer problem. This fact has also been experimentally demonstrated by executing 100 independent GP runs for each one of these problems and calculating the rate of successful runs. As a further confirmation, we have also calculated the value of another GP hardness indicator, called Negative Slope Coefficient.

Another interesting result that we have obtained with our measures is that the landscape induced by {XOR, NOT} for the even parity and by {IF} for the multiplexer appears to be "more neutral" than the corresponding ones induced by {NAND}. Thus, in these particular case studies, neutrality (as expressed by the average neutrality ratio measure) is helpful for GP, in particular when located in good regions of the fitness landscape. This fact, of course, does not allow us at all to conclude that more neutral landscapes are easier; nevertheless, studying the average neutrality ratios could be helpful to formulate some hypothesis on the difficulty of a given problem. All our results allow us to conclude that the proposed measures are helpful in establishing some good tools to study neutrality and to relate it to GP problem hardness, and to investigate some important features of fitness landscapes related to neutrality.

Results shown in this paper hold both for "small" even parity and multiplexer fitness landscapes, that we have been able to study by exhaustively generating all the individuals, and for "large" fitness landscapes, obtained by increasing the problem order and the maximum size of the individuals, and that we have sampled using a new methodology defined in this paper. This methodology is based on a modified version of the Metropolis algorithm (in which we have changed the probability of accepting a new individual into the sample), enriched by two extensions that we have called *vertical* and *horizontal* expansion. Vertical expansion has been introduced to enrich the sample with some non-neutral neighbors of its individuals, while the horizontal expansion has been introduced to enrich the sample with some neutral neighbors of its individuals. In this way, the resulting sample should be rich enough to describe both the characteristics of the fitness landscape related to neutrality and the ones that are not. All the results obtained using our sampling methodology may suggest its suitability both for the Boolean even parity and multiplexer problems. In particular, by means of our sampling strategy, it has been possible to generate and to study a large number of individuals that would not (or would very rarely) have been generated by means of a uniform random sampling or a standard Metropolis algorithm, like individuals with a fitness value different from 0.5 for the even parity. Furthermore, our sampling technique has allowed us to generate individuals with many different fitness values, thus giving us a wider view of the fitness landscapes.

Since our techniques are general and can be used for any GP program space, future work includes extending this kind of study to other problems and possibly defining new measures of problem hardness based on neutrality.

[1] C. M. Reidys, P. F. Stadler, Neutrality in fitness landscapes, Applied Mathematics and Computation 117 (2001) 321–350.

[2] P. Collard, M. Clergue, M. Defoin-Platel, Synthetic neutrality for artificial evolution, in: Artificial Evolution, pp. 254–265.

[3] M. Toussaint, C. Igel, Neutrality: A necessity for self-adaptation, in: Congress on Evolutionary Computation (CEC'02), IEEE Press, Piscataway, NJ, Honolulu, Hawaii, USA, 2002, pp. 1354–1359.

[4] N. Geard, J. Wiles, J. Hallinan, B. Tonkes, B. Skellett, A comparison of neutral landscapes – NK, NKp and NKq, in: Congress on Evolutionary Computation (CEC'02), IEEE Press, Piscataway, NJ, Honolulu, Hawaii, USA, 2002, pp. 205–210.

[5] P. Collard, S. Verel, M. Clergue, How to use the scuba diving metaphor to solve problem with neutrality?, in: R. Lopez, L. Saitta (Eds.), ECAI'04: European Conference on Artificial Intelligence, IOS Press, 2004, pp. 166–170.

[6] E. Galvan, An Analysis of the Effects of Neutrality on Problem Hardness for Evolutionary Algorithms, Ph.D. thesis, School of Computer Science and Electronic Engineering, University of Essex, United Kingdom, 2009.

[7] T. Jones, S. Forrest, Fitness distance correlation as a measure of problem difficulty for genetic algorithms, in: L. J. Eshelman (Ed.), Proceedings of the Sixth International Conference on Genetic Algorithms, Morgan Kaufmann, 1995, pp. 184–192.

[8] T. Jones, Evolutionary Algorithms, Fitness Landscapes and Search, Ph.D. thesis, University of New Mexico, Albuquerque, 1995.

[9] L. Vanneschi, M. Tomassini, P. Collard, S. Vérel, Negative slope coefficient. a measure to characterize genetic programming fitness landscapes, in: Collet, P., *et al.* (Ed.), Genetic Programming, 9th European Conference, EuroGP2006, Lecture Notes in Computer Science, LNCS 3905, Springer, Berlin, Heidelberg, New York, 2006, pp. 178–189.

[10] L. Vanneschi, M. Clergue, P. Collard, M. Tomassini, S. Vérel, Fitness clouds and problem hardness in genetic programming, in: K. Deb *et al.* (Ed.), Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'04, volume 3103 of *Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, New York, 2004, pp. 690–701.

[11] L. Vanneschi, Theory and Practice for Efficient Genetic Programming, Ph.D. thesis, Faculty of Sciences, University of Lausanne, Switzerland, 2004.

[12] T. Yu, J. F. Miller, Finding needles in haystacks is not hard with neutrality, in: J. A. Foster *et al.* (Ed.), Genetic Programming, Proceedings of the 5th European Conference, EuroGP 2002, volume 2278 of *LNCS*, Springer-Verlag, 2002, pp. 13–25.

[13] M. Collins, Finding needles in haystacks is harder with neutrality, in: H.-G. Beyer *et al.* (Ed.), GECCO 2005: Proceedings of the 2005 conference on Genetic and evolutionary computation, volume 2, ACM Press, Washington DC, USA, 2005, pp. 1613–1618.

[14] T. Yu, 'Six degrees of separation' in boolean function networks with neutrality, in: R. Poli *et al.* (Ed.), GECCO 2004 Workshop Proceedings, Seattle, Washington, USA.

[15] T. Yu, J. Miller, Neutrality and the evolvability of boolean function landscape, in: J. Miller *et al.* (Ed.), Proceedings of the Fourth European Conference on Genetic Programming (EuroGP-2001), volume 2038 of *LNCS*, Springer, Berlin, Heidelberg, New York, Lake Como, Italy, 2001, pp. 204–217. Lecture notes in Computer Science vol. 2038.

[16] T. Yu, J. F. Miller, The role of neutral and adaptive mutation in an evolutionary search on the onemax problem, in: E. Cantú-Paz (Ed.), Late Breaking Papers at the Genetic and Evolutionary Computation Conference (GECCO-2002), AAAI, New York, NY, 2002, pp. 512–519.

[17] J. R. Koza, Genetic Programming, The MIT Press, Cambridge, Massachusetts, 1992.

[18] M. Ebner, M. Shackleton, R. Shipman, How neutral networks influence evolvability, Complexity 7 (2001) 19–33.

[19] L. Barnett, Evolutionary Search on Fitness Landscapes with Neutral Networks, Ph.D. thesis, University of Sussex, 2003.

[20] T. Smith, P. Husbands, M. O'Shea, Neutral networks and evolvability with complex genotype-phenotype mapping, in: ECAL '01: Proceedings of the 6th European Conference on Advances in Artificial Life, Springer-Verlag, London, UK, 2001, pp. 272–281.

[21] S. Verel, Étude et Exploitation des Réseaux de Neutralité dans les Paysages Adaptatifs pour l'Optimisation Difficile, Ph.D. thesis, University of Nice – Sophia Antipolis, France, 2005. In French Language.

[22] W. Banzhaf, A. Leier, Evolution on neutral networks in genetic programming, in: T. Yu, R. L. Riolo, B. Worzel (Eds.), Genetic Programming Theory and Practice III, volume 9 of *Genetic Programming*, Springer, Ann Arbor, 2005, pp. 207–221.

[23] W. B. Langdon, R. Poli, Foundations of Genetic Programming, Springer, Berlin, Heidelberg, New York, Berlin, 2002.

[24] L. Vanneschi, Y. Pirola, P. Collard, M. Tomassini, S. Verel, G. Mauri, A quantitative study of neutrality in GP boolean landscapes, in: M. Keijzer *et al.* (Ed.), Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'06, volume 1, ACM Press, 2006, pp. 895–902.

[25] S. Wright, The roles of mutation, inbreeding, crossbreeding and selection in evolution, in: D. F. Jones (Ed.), Proceedings of the Sixth International Congress on Genetics, volume 1, pp. 356–366.

[26] L. Vanneschi, M. Tomassini, P. Collard, M. Clergue, Fitness distance correlation in structural mutation genetic programming, in: Ryan, C., *et al.* (Ed.), Genetic Programming, 6th European Conference, EuroGP2003, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, New York, 2003, pp. 455–464.

[27] P. F. Stadler, Fitness landscapes, in: M. Lässig, Valleriani (Eds.), Biological Evolution and Statistical Physics, volume 585 of *Lecture Notes Physics*, Springer, Berlin, Heidelberg, New York, 2002, pp. 187–207.

[28] P. Schuster, W. Fontana, P. F. Stadler, I. L. Hofacker, From sequences to shapes and back: a case study in RNA secondary structures, in: Proc. R. Soc. London B., volume 255, pp. 279–284.

[29] L. Altenberg, The evolution of evolvability in genetic programming, in: K. Kinnear (Ed.), Advances in Genetic Programming, The MIT Press, Cambridge, MA, 1994, pp. 47–74.

[30] J. Reisinger, K. O. Stanley, R. Miikkulainen, Towards an empirical measure of evolvability, in: GECCO '05: Proceedings of the 2005 workshops on Genetic and Evolutionary Computation, ACM Press, New York, NY, USA, 2005, pp. 257–264.

[31] S. Vérel, P. Collard, M. Clergue, Where are bottleneck in NK-fitness landscapes ?, in: CEC 2003: IEEE International Congress on Evolutionary Computation. Canberra, Australia, IEEE Press, Piscataway, NJ, 2003, pp. 273–280.

[32] N. Madras, Lectures on Monte Carlo Methods, American Mathematical Society, Providence, Rhode Island, 2002.

[33] R. Poli, L. Vanneschi, Fitness-proportional negative slope coefficient as a hardness measure for genetic algorithms, in: D. Thierens *et al.* (Ed.), Genetic and Evolutionary Computation Conference, GECCO'07, ACM Press, 2007, pp. 1335–1342.

[34] L. Vanneschi, A. Valsecchi, R. Poli, Limitations of the fitness-proportional negative slope coefficient as a difficulty measure., in: G. Raidl *et al.* (Ed.), GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation, ACM, Montreal, 2009, pp. 1877–1878.

[35] L. Vanneschi, Investigating problem hardness of real life applications, in: *et al.*. R. Riolo (Ed.), Genetic Programming Theory and Practice V, Springer US, Computer Science Collection, 2007, pp. 107–124. Chapter 7.

[36] Y. Pirola, Studio della neutralità degli spazi di ricerca booleani in Programmazione Genetica, Master's thesis, Università di Milano-Bicocca, Milano, 2006. In Italian language.

[37] H. Enderton, A mathematical introduction to logic, Academic Press, Boston, MA, 2 edition, 2001.

[38] M. Huynen, Exploring phenotype space through neutral evolution, J. Mol. Evol., 43:165–169 (1996).

[39] S. Gustafson, L. Vanneschi, Operator-based distance for genetic programming: subtree crossover distance, in: M. K. *et al.* (Ed.), Genetic Programming, 8th European Conference, EuroGP2005, volume 3447 of *Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, New York, 2005, pp. 178–189.

[40] L. Vanneschi, S. Gustafson, G. Mauri, Using subtree crossover distance to investigate genetic programming dynamics, in: Collet, P., *et al.* (Ed.), Genetic Programming, 9th European Conference, EuroGP2006, Lecture Notes in Computer Science, LNCS 3905, Springer, Berlin, Heidelberg, New York, 2006, pp. 238–249.

[41] S. Gustafson, L. Vanneschi, Operator-based tree distance in genetic programming, IEEE Transactions on Evolutionary Computation 12 (2008) 506–524.