Complex Network Analysis of a Genetic Programming Phenotype Network

Ting Hu[1(✉)], Marco Tomassini[2], and Wolfgang Banzhaf[3]

[1] Department of Computer Science, Memorial University, St. John's, NL, Canada
ting.hu@mun.ca

[2] Faculty of Business and Economics, Information Systems Department, University of Lausanne, Lausanne, Switzerland
marco.tomassini@unil.ch

[3] Department of Computer Science and Engineering, and BEACON Center, Michigan State University, East Lansing, MI, USA
banzhafw@msu.edu

Abstract. The genotype-to-phenotype mapping plays an essential role in the design of an evolutionary algorithm. Since variation occurs at the genotypic level but fitness is evaluated at the phenotypic level, this mapping determines how variations are effectively translated into quality improvements. We numerically study the redundant genotype-to-phenotype mapping of a simple Boolean linear genetic programming system. In particular, we investigate the resulting phenotypic network using tools of complex network analysis. The analysis yields a number of interesting statistics of this network, considered both as a directed as well as an undirected graph. We show by numerical simulation that less redundant phenotypes are more difficult to find as targets of a search than others that have much more genotypic abundance. We connect this observation with the fact that hard to find phenotypes tend to belong to small and almost isolated clusters in the phenotypic network.

Keywords: Complex networks · Genetic programming · Genotype-phenotype mapping · Phenotype networks · Evolvability

1 Introduction

In evolutionary algorithms, the quality of a candidate solution is assessed based on its phenotype, i.e., how well the phenotype is able to produce a desired outcome judged by a fitness measure. Yet, the actual EA search occurs in genotype space, where the encoding of candidate solutions is modified by mutation or recombination operations. Thus, how genotypes are mapped to phenotypes will substantially influence the search effectiveness of an evolutionary algorithm [1,2].

Redundant genotype-to-phenotype mappings are common in both natural [3,4] and computational evolution [5–8], where multiple genotypes can map to

L. Sekanina et al. (Eds.): EuroGP 2019, LNCS 11451, pp. 49–63, 2019.
https://doi.org/10.1007/978-3-030-16670-0_4

the same phenotype. Such a *redundancy* is often unevenly distributed among phenotypes, where some phenotypes are over-represented by many genotypes, and some are under-represented by only a few [7,9]. When the target phenotype is under-represented, its evolutionary search is often more difficult than having a genotypically over-represented target. This is intuitive since it can be more difficult to find one of the few genotypes that map to an under-represented target phenotype.

If the genotype-to-phenotype mapping is redundant, a mutation to a genotype may not change the phenotype it encodes, a phenomenon defined as *neutrality* [10,11], and such mutations are called neutral mutations [12–15]. Neutrality is facilitated by redundancy, but not guaranteed. For instance, there are cases where genotypes map to the same phenotype but are not mutationally connected, i.e., one genotype cannot be reached from the other through single point mutations, thus mutations that need to occur on the way from one to the other will need to alter the phenotype.

In contrast to neutral mutations, non-neutral mutations connect genotypes of distinct phenotypes. Such non-neutral mutational connections among phenotypes might also be heterogeneous [9,16], i.e., a phenotype may not have the same likelihood of mutating to other phenotypes and thus may tend to "prefer" some phenotypes over others. The difficulty of finding a target phenotype is thus influenced not only by its genotypic abundance, but also by how mutational connections are distributed among different phenotypes.

In this contribution, we quantitatively measure the genotypic redundancy of phenotypes and the mutational connections among them, and take a network approach to analyze how these properties correlate with the difficulty of finding a target phenotype. We use a linear genetic programming (LGP) algorithm for Boolean optimization, and numerically characterize its genotype, phenotype, and fitness space. Using random sampling and random walks, we construct a phenotype network to depict the mutational connections among different phenotypes. Once a specific target phenotype is chosen, this changes the connectivity of the phenotype network since only non-deleterious mutations, i.e. mutations that do not decrease fitness, are allowed. We show that such changes can significantly influence the difficulty of finding a target.

2 Methods

2.1 A Boolean Linear Genetic Programming Algorithm

We use a linear genetic programming (LGP) algorithm for our empirical analysis. LGP is a branch of genetic programming and employs a sequential representation of computer programs to encode an evolutionary individual [17]. Such an LGP program is often comprised of a set of imperative instructions, which are executed sequentially. Registers are used to either read input variables (input registers) or to enable computational capacity (calculation register). One or more registers can be designated as the output register(s) such that the final stored value(s) after the program is executed will be the program's output.

In this study, we use an LGP algorithm for a three-input, one-output Boolean function modeling application. Each instruction has one return, two operands and one Boolean operator. The operator set has four Boolean functions {AND, OR, NAND, NOR}, any of which can be selected as the operator for an instruction. Three registers $\mathtt{R}_1$, $\mathtt{R}_2$, and $\mathtt{R}_3$ receive the three Boolean inputs, and are write-protected in an LGP program. That is, they can only be used as an operand in an instruction. Registers $\mathtt{R}_0$ and $\mathtt{R}_4$ are calculation registers, and can be used as either a return or an operand. Register $\mathtt{R}_0$ is also the designated output register, and the Boolean value stored in $\mathtt{R}_0$ after an LGP program's execution will be the final output of the program. All calculation registers are initialized as FALSE before execution of a program. An LGP program can have any number of instructions, however, for the ease of simulation in this study, we determine that an LGP program has a fixed length of six instructions. An example LGP program is given as follows.

$$
\begin{aligned}
I_1 &: \mathtt{R}_4 = \mathtt{R}_2 \ \mathtt{AND} \ \mathtt{R}_3 \\
I_2 &: \mathtt{R}_0 = \mathtt{R}_1 \ \mathtt{OR} \ \mathtt{R}_4 \\
I_3 &: \mathtt{R}_4 = \mathtt{R}_4 \ \mathtt{NAND} \ \mathtt{R}_0 \\
I_4 &: \mathtt{R}_4 = \mathtt{R}_3 \ \mathtt{AND} \ \mathtt{R}_2 \\
I_5 &: \mathtt{R}_0 = \mathtt{R}_1 \ \mathtt{NOR} \ \mathtt{R}_1 \\
I_6 &: \mathtt{R}_0 = \mathtt{R}_3 \ \mathtt{AND} \ \mathtt{R}_0
\end{aligned}
$$

2.2 Genotype, Phenotype, and Fitness

The *genotype* in our evolutionary algorithm is a unique LGP program. Since we have a finite set of registers and operators, as well as a fixed length for all programs, the genotype space is finite. Specifically, considering an instruction, two registers can be chosen as the return, all five registers can be used as the two operands, and the operator is picked from the set of four possible Boolean functions. Thus, there are $2 \times 5 \times 5 \times 4 = 200$ unique instructions. Given the fixed length of six instructions for all LGP programs, we have a total number of $200^6 = 6.4 \times 10^{13}$ possible different programs. Although finite, the genotype space is enormous and is not amenable to exhaustive enumeration. Therefore, we conduct a simulation by randomly generating one billion LGP programs ($\approx$15.6 ppm = 0.00156% of the genotype space) to approximate the genotype space.

The *phenotype* in our evolutionary algorithm is a Boolean relationship that maps three inputs to one output, represented by an LGP program, i.e., $f : \mathbf{B}^3 \rightarrow \mathbf{B}$, where $\mathbf{B} = \{\mathtt{TRUE}, \mathtt{FALSE}\}$. There are thus a total of $2^{2^3} = 256$ possible Boolean relationships. Having 6.4×10^{13} genotypes to encode 256 phenotypes, our LGP algorithm must have a highly redundant genotype-to-phenotype mapping. We define the *genotypic redundancy* of a phenotype as the total number of genotypes that map to it.

The *fitness* of an LGP program is dependent on the target Boolean relationship, and it is defined as the dissimilarity of the presented and the target

Boolean relationships. Given three inputs, there are $2^3 = 8$ combinations of Boolean inputs. The Boolean relationship encoded by an LGP program can be seen as a 8-bit string representing the outputs that correspond to all 8 possible combinations of inputs. Fitness is defined as the Hamming distance of this 8-bit output and the target output. For instance, if the target relationship is $f(\mathtt{R}_1, \mathtt{R}_2, \mathtt{R}_3) = \mathtt{R}_1\ \mathtt{AND}\ \mathtt{R}_2\ \mathtt{AND}\ \mathtt{R}_3$, represented by the 8-bit output string of 00000001, the fitness of an LGP program encoding the FALSE relationship, i.e., 00000000, is 1. Fitness falls into the range of $[0, 8]$ where 0 is the perfect fitness and 8 is the worst, and is to be minimized.

2.3 Phenotype Networks

Point mutations to genotypes may change the encoded phenotypes from one to another. In the context of our LGP algorithm, a point mutation is to replace any one of the four elements, i.e., return, two operands, and operator, of an instruction in an LGP program. The mutational connections among pairs of phenotypes can be modeled using a *phenotype network*. In such a network, each node represents one of the 256 phenotypes that can be possibly encoded by the LGP genotypes. Two nodes (phenotypes) are directly connected by an edge if there exist at least one pair of underlying genotypes, one from each phenotype, that can be transitioned from one to the other through a single point mutation.

Since it is infeasible to enumerate all possible genotypes, sampling the mutational connections among phenotypes is also necessary. We assemble one million randomly generated LGP programs and allow each to take a 1000-step random walk in genotype space. All the phenotypes each random walker encountered are recorded in order to estimate the number of point mutations that can transition one phenotype to another. This random walk simulation yields a *undirected, weighted* phenotype network, where the weight of an edge is proportional to the number of sampled point mutations that can change the genotypes of one phenotype to that of the other phenotype.

Assigning a fitness to each phenotype and preventing deleterious mutations changes the reversible feature of point mutations and further transforms the weighted phenotype network into a *directed* graph. We pick two target phenotypes with a considerable difference in their genotypic redundancies, given the consideration that whether a target phenotype is over- or under-represented by genotypic encodings may influence the difficulty level of finding that target [7]. The first target is phenotype 11110000 (decimal 240) which has a genotypic redundancy of 46,729,920, i.e., 4.673% of the one billion sampled genotypes. The second target is phenotype 10110100 (decimal 180) with only a genotypic redundancy of 86. Setting such different targets will render the corresponding directed, weighted phenotype networks different. Thus, we investigate a variety of network properties to compare these two networks.

2.4 Complex Network Analysis

Since we need a few concepts and methods from the field of network science [18, 19], we here collect some useful definitions to be referenced and used later.

Strength. This term refers to the generalization of the vertex degree to weighted networks. It is defined as the sum of weights of the edges from node i to its neighbors $\mathcal{N}(i)$,

$$s_i = \sum_{j \in \mathcal{N}(i)} w_{ij},$$

where w_{ij} is the weight of the edge connecting nodes i and j.

Disparity. A given value of a node's strength can be obtained with very different values of edge weights. The contributing weights could be of about the same size or they could be very different. To measure the degree of heterogeneity of a node's edges *disparity* can be used. It is defined as follows:

$$Y_2(i) = \sum_{j \in \mathcal{N}(i)} \left(\frac{w_{ij}}{s_i} \right)^2.$$

If all the connections are of the same order then Y_2 is small and of order $1/k$ where k is the vertex degree. On the other hand, if there is a small number of high weight connections Y_2 is larger and may approach unity.

Average Shortest Paths. We use weighted and unweighted shortest paths between pairs of vertices. The average values of all two-point shortest paths in a graph give an idea of the typical distances between nodes.

Clustering Coefficient. The clustering coefficient $C(i)$ of a node i is defined as the ratio between the e edges that actually exist between the k neighbors of i and the number of possible edges between these nodes:

$$C(i) = \frac{e}{\binom{k}{2}} = \frac{2e}{k(k-1)}.$$

The clustering coefficient can be interpreted intuitively as the likelihood that two of node i's neighbors are also neighbors. The *average clustering coefficient* $\bar{C}$ is the average of $C(i)$ over all N vertices in the graph G, $i \in V(G)$: $\bar{C} = (1/N) \sum_{i=1}^{N} C(i)$.

Degree, Strength, and Weights Distribution Functions. These discrete distributions give, respectively, the frequency of a given node degree, node strength, or edge weight in the network. These distributions are useful for evaluating whether they are, for instance, homogeneous or heterogeneous, unimodal or multimodal.

3 Results

3.1 Sampled Genotype Space and Mutational Connections

When we decode the one billion randomly generated genotypes, we find that 17 of the total 256 phenotypes are never sampled. The distribution of the genotypic redundancy of the remaining 239 sampled phenotypes is highly heterogeneous (see Fig. 1a). The most over-represented phenotype is 0, i.e., `FALSE`, which has over 108 million genotypes, while phenotype 255, i.e., `TRUE`, its symmetric counterpart, is the second most abundant with over 93 million genotypes. The asymmetry in count is due to the initialization of calculation registers, including the output register $\mathtt{R}_0$, to `FALSE` in all LGP programs prior to execution. In addition to the 17 phenotypes never sampled, under-represented phenotypes include 105, 231, 24, 219, 189, 36, and 66, none of which has more than 40 sampled genotypic encodings.

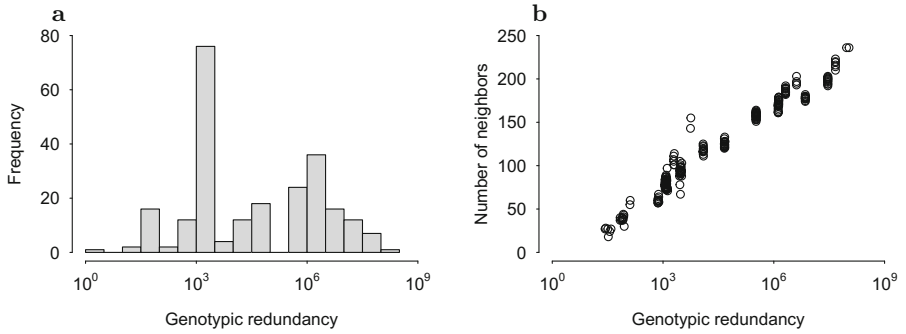

Fig. 1. (a) Distribution of the genotypic redundancy of sampled phenotypes using one billion randomly generated LGP programs. (b) Number of neighbors in relation to the genotypic redundancy of a phenotype. Their linear-log correlation has a coefficient of 0.9642 ($p < 2.2 \times 10^{-16}$).

Using the assembly of one million 1,000-step random walkers, the mutational connections among pairs of phenotypes can be approximated. 16 out of 256 phenotypes are never encountered, i.e., they are isolated nodes in the phenotype network, 15 of which belong to the 17 never-sampled phenotypes discussed previously. This also suggests that under-represented phenotypes are hard to reach by random walks. Figure 1b shows the correlation of node degree, i.e., the number of distinct phenotypes accessible from a phenotype through point mutations, and the genotypic redundancy of a phenotype. We observe a strong and highly significant positive correlation.

3.2 Properties of the Undirected Weighted Phenotype Network

We first construct an undirected, weighted phenotype network using the sampled mutational connections among pairs of phenotypes. The network has 240 nodes

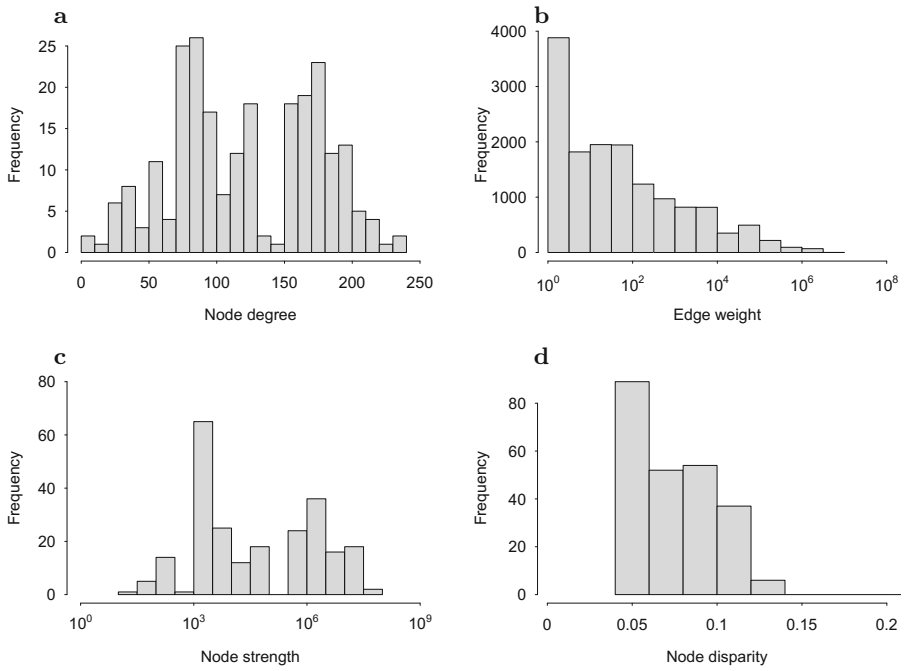

Fig. 2. Distribution of (a) node degree, (b) edge weight, (c) node strength, and (d) node disparity in the undirected weighted phenotype network.

representing 240 unique phenotypes, since 16 phenotypes were never encountered in the random walk sampling. See Sect. 2.4 for definitions of the various measures used. Ignoring edge weights, the network has 14,663 edges, which yields an average node degree of 122. There is only one connected component in this network. Its average shortest path is only 1.5 and its diameter is as short as 3, which means that any pair of phenotypes can be reached from one to another by point mutations through no more than 3 hops in the phenotype network. The clustering coefficient is high at 0.75; this is due to the fact that many nodes have neighbors that are themselves connected, giving rise to many closed triangles.

The degree and edge weight distributions are shown in Fig. 2(a) and (b). Phenotypes 0 (`FALSE`) and 255 (`TRUE`) have the highest node degree of 236, and phenotypes 22 and 104 only have a degree of 2. The distribution of edge weights is roughly monotonic, with the majority of edges having a weight of less than 50, while the edge connecting phenotypes 0 and 255 has the highest weight of 5,673,803.

Figure 2(c) and (d) show node strength and node disparity distributions, respectively. Strength, being a generalization of degree for weighted networks, has a shape that is qualitatively similar to the degree histogram, with a bimodal distribution. The node disparity shows that most phenotypes have a low disparity, i.e. their links tend to have similar weights, and the distribution decays

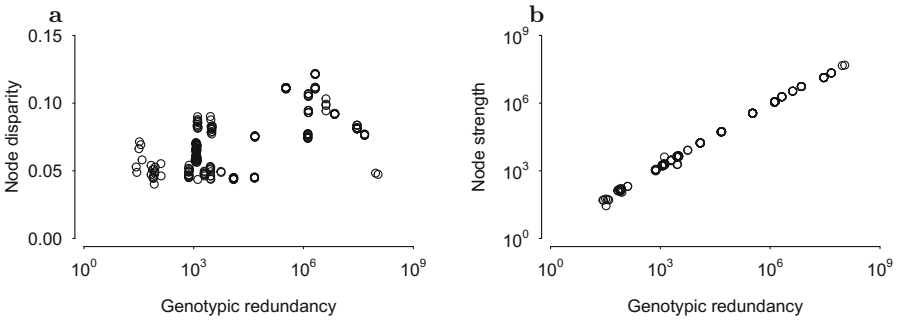

Fig. 3. Node (a) disparity and (b) strength in relation to the genotypic redundancy of a phenotype. The linear-log correlation in (a) has a coefficient $r^2 = 0.3568$ ($p < 2.2 \times 10^{-16}$), and the log-log correlation in (b) has a coefficient $r^2 = 0.9981$ ($p < 2.2 \times 10^{-16}$).

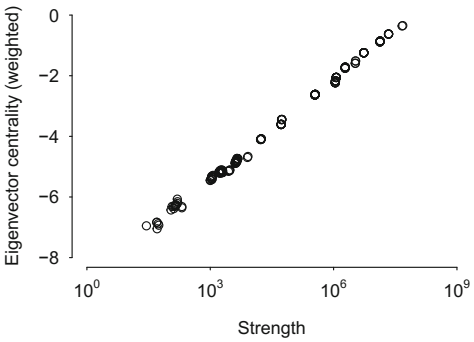

Fig. 4. Correlation of node strength and weighted eigenvector centrality. This positive correlation has a coefficient r^2=0.9982 ($p < 2.2 \times 10^{-16}$).

quickly. Since we were only interested in the qualitative aspects of these distributions, we did not attempt to fit any particular functions to them. Finally, Fig. 3 shows that genotypic redundancy is strongly correlated with node strength.

An important property of phenotypes is their *evolvability* which is the ability to generate novel and adaptive phenotypes. Evolvability can be defined quantitatively and has been studied in detail in previous work (see, e.g., [9,20,21]). In particular, it was found in [21] that a phenotypic network centrality measure called *weighted eigenvector centrality* was a good predictor for phenotypic evolvability. In that respect, we note that the simpler node strength is highly correlated with this centrality measure, as can be seen in Fig. 4. This means that a good proxy for phenotypic evolvability is the easily computable phenotype strength.

3.3 Communities in the Undirected Weighted Phenotype Network

Communities in a complex network can be loosely defined as collections of vertices that are more strongly linked among themselves than with the rest of the network. A precise and unique definition cannot be given, which makes community detection a hard and somewhat ill-defined task. Nevertheless, several community detection algorithms have been proposed that work well in practice.

Here we use the methods implemented in the *igraph* R package [22], which also cover weighted networks. Before submitting our phenotypic network G to a community detection algorithm some manipulations are necessary. In fact, the graph has a mean degree of about 122 which makes it a very dense network. Community detection algorithms typically do not work well, or at all, on such graphs. However, we note that edge weights in G span seven orders of magnitude (see Fig. 2b), which means that many links are comparatively very weak. Thus we have discarded weak network connections by cutting all edges with weights below a threshold of $w_{ij} < 10^5$. As a consequence, some of the original nodes also become disconnected but we have ensured that all edges of the target phenotypes are kept, especially for target node 180, which would have become isolated otherwise, since all its edges have weights lower than the threshold.

Modularity is a measure that estimates the cohesiveness of a partition found by a community detection algorithm with respect to a graph with the same degree distribution but with edges placed at random [23]. The community partition found with several community detection algorithms from *igraph* has a modularity value of about two, which is not very high but still significantly different from random. Figure 5 shows the communities found by the *Louvain* algorithm. It is important to note that the small community to which vertex 180 belongs is almost always found identically by all the different algorithms tried. Figure 5 clearly shows that node 180, together with its neighbors belonging to the same community, appears to be extremely difficult to reach, all the more taking into account that the intra-community and extra-community edges are weak. On the other hand, phenotype 240 is at the intersection of two bigger and well connected communities and thus it is intuitively reasonable that it should be easier to find. These indications will find a numerical confirmation in the next section where we shall use random walks to traverse the network.

3.4 Random Walks

Although the GP system searches the genotype space and not the much smaller phenotype space, it is still interesting to simulate random walk search in the latter to numerically confirm the above idea that some phenotypes are easy to find while others are hard. Random walks on networks are reviewed in [24]. In an unweighted network, the probability for going from node i to node j is $p_{ij} = a_{ij}/k_i$, where a_{ij} is the corresponding entry in the graph adjacency matrix being 1 if nodes i and j are connected and 0 if they are not. The random walk we are interested in is biased, since edges of the undirected phenotype network are weighted. So we have to modify the probabilities accordingly, but the changes are

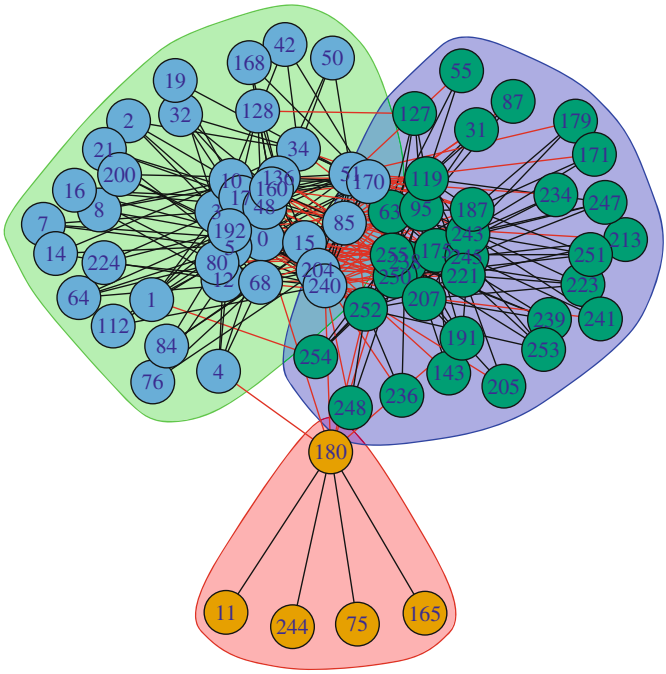

Fig. 5. Community structure of the edge-filtered undirected phenotypic network. Phenotype 180 clearly belongs to a small community that is very weakly connected to the rest of the network, while phenotype 240 is located in the center of the network and belongs to a larger and well connected community.

minor: the transition probability from node i to node j through an edge $\{ij\}$ with weight $w_{ij} \geq 0$ now becomes $p_{ij} = w_{ij}/s_i$, where s_i is the *strength* of node i and is defined as the sum of the weights of the edges from i to its neighbors $\mathcal{N}(i)$ (see definitions in Sect. 2.4). These probabilities are well behaved since a connected node must have a positive finite strength, $p_{ij} \geq 0$, and $\sum_{j \in \mathcal{N}(i)} p_{ij} = 1$.

For each of the two target phenotypes 180 and 240, we numerically simulate biased random walks in the original unfiltered network starting from all network nodes except the target nodes themselves. For each starting node we perform 10^5 random walk steps, for a total of $(N-1) \times 10^5 = 237 \times 10^5$ steps. For each of the phenotypes 180 and 240 we record the number of times it is found, i.e., the number of hits, and the mean number of steps to the first hit, when the node is found.

Results are shown in Table 1. From the number of hits and the first hit times, it is apparent that phenotype 180 is much harder to find than phenotype 240. Furthermore, if we exclude the first neighbors of the target node as starting nodes in the random walk, it becomes even more difficult, comparatively, to find phenotype 180 (figures after the comma in Table 1). We can also see that phenotype 240 is very often found directly from a starting node that is a first neighbor given that 240 has 223 neighbors while 180 only has 41 connections.

Table 1. Average number of hits and first hitting times for random walks having nodes 180 and 240 as targets. 237×10^5 random walks steps are performed in total. The figures after the commas refer to the same quantities when the first neighbors of nodes 180 and 240 are excluded as starting nodes for the walk.

	Target phenotype 180	Target phenotype 240
Number of hits	2, 0	97465, 5862
First hitting time	910306, –	10, 10

3.5 In-degree and Out-degree in the Directed Phenotype Networks

When a fitness is assigned to each phenotype based on its Hamming distance to the target phenotype, the phenotype network becomes oriented since we only allow non-deleterious point mutations. Note that we only consider simple graphs in the current study, i.e., self-loops are excluded in our network analysis, in order to focus on the mutational connections among distinct phenotypes. A phenotype/node now has edges with two directions, pointing to its neighbors (out edges) and being pointed from its neighbors (in edges). Subsequently, in-degree and out-degree can be used to depict how many unique phenotypes can access or can be reached from a reference phenotype.

Figure 6 shows the correlations of in- and out-degrees with the fitness of a phenotype in two directed phenotype networks with different targets. Using both targets, in-degrees are negatively correlated with fitness while out-degrees are positively correlated with fitness. Note that fitness is to be minimized. Phenotypes with better fitness will have less edges going out but more edges coming in, i.e., fitter phenotypes are easier to reach and harder to leave, which is intuitive and desirable since we hope reaching fitter phenotypes will be more likely leading to the path to the target. However, when we compare the correlations using different targets, it can be seen that using a relatively harder target (i.e., phenotype 180) results in weaker correlations of in-/out-degrees and fitness. This indicates that some targets are difficult to find not only because they under-represented by genotypes, but also because they render the guidance of the fitness gradient less effective. That is, reaching fitter phenotypes at a current stop does not necessarily lead to better paths to finding the target.

3.6 Fitness Correlation of Neighboring Phenotypes

Fitness correlation can give statistical information about the fitness assortativity of neighboring nodes in the network. A practical way for evaluating fitness correlation is given by the average fitness of neighbors $\bar{f}_{neighbor}(i)$ of a node i

$$\bar{f}_{neighbor}(i) = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} f_j,$$

where f_i is the fitness of phenotype/node i.

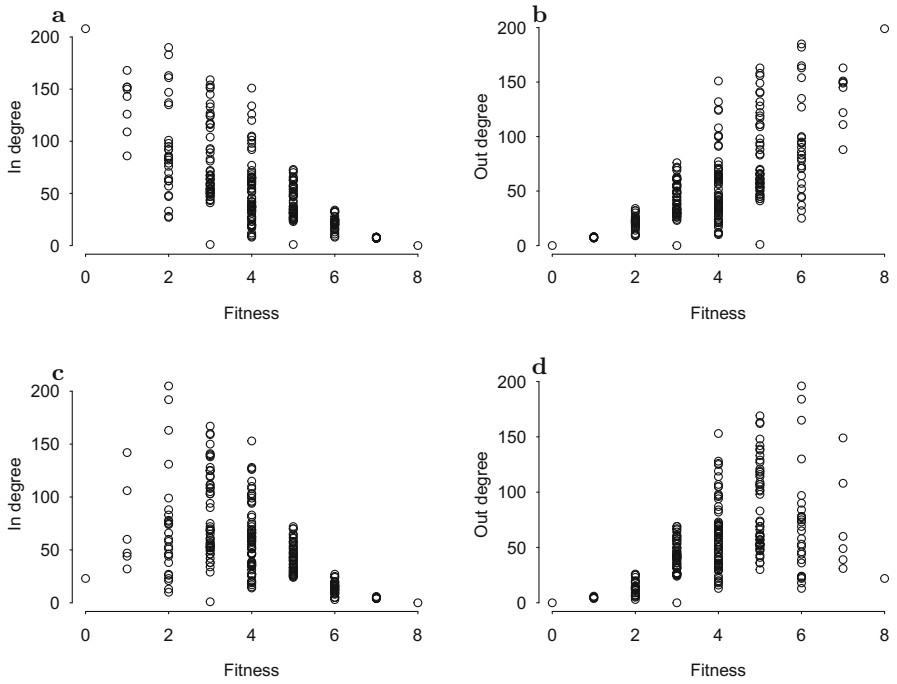

Fig. 6. Correlation of node in-degree (a, c) and out-degree (b, d) with fitness in the directed phenotype networks using an over-represented target 240 (a, b) and under-represented target 180 (b, d). In (a) and (c), the negative correlations are with a coefficient of $R^2 = 0.4623$ ($p < 2.2 \times 10^{-16}$), and $R^2 = 0.2519$ ($p < 4.6 \times 10^{-16}$), respectively. In (b) and (d), the positive correlations are with a coefficient of $R^2 = 0.4611$ ($p < 2.2 \times 10^{-16}$), and $R^2 = 0.2622$ ($p < 4.6 \times 10^{-16}$), respectively.

From this quantity one can compute the average fitness of the neighbors $\bar{f}_{neighbor}(f)$ for nodes of the phenotypic network having fitness value f which is a good approximation to the fitness-fitness correlation:

$$\bar{f}_{neighbor}(f) = \frac{1}{N_f} \sum_i \bar{f}_{neighbor}(i),$$

where N_f is the number of nodes with fitness f.

Remembering that a low fitness value is better in our context, one can see from Fig. 7 that the neighboring fitness correlations are quite different when different phenotypes are used as the search target. Specifically, when the over-represented phenotype 240 is set as the target (see Fig. 7a), the fitness values of neighboring phenotypes do not correlate, meaning that a phenotype can be connected to phenotypes with any fitness values. However, as shown in Fig. 7b, when the target is under-represented, phenotypes having fitnesses greater than four tend to have neighbors with lower (better) fitness, while good phenotypes

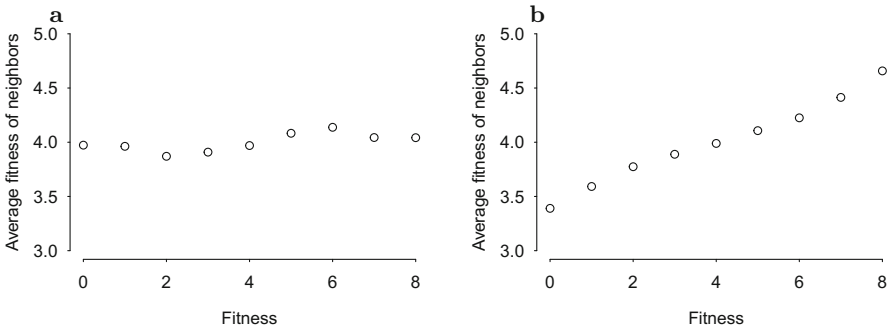

Fig. 7. Average neighbor fitness versus node fitness in the phenotypic network with the target phenotype (a) 240 and (b) 180.

below four tend to have as neighbors phenotypes with higher (worse) fitness. This situation indicates that, in the average, good phenotypes, i.e., those that have a fitness lower than the average of four, are surrounded by less good ones.

4 Discussion

The genotype-to-phenotype mapping plays a central role in enabling a system to be evolvable, since the variations characterizing the search occur in the genotype space, but the quality or behavior of a system can only be observed and evaluated at the phenotypic level. We argue that some target phenotypes are more difficult to find than others, not only because they are most likely under-represented in genotypic space, but also because the mutational connections between phenotypes are altered through setting different target phenotypes.

In this study, we took a network approach and quantitatively analyzed the distribution of mutational connections among phenotypes and how this distribution is changed with different phenotypes set as target. Using a Boolean LGP algorithm, we sampled the genotypic and phenotypic spaces and constructed a phenotype network to characterize the distribution of mutational connections among phenotypes. By setting two different phenotypes as the target, one genotypically over-represented and one under-represented, we compared the properties of the resulting directed, weighted phenotype networks.

Similar to many GP systems, our Boolean LGP algorithm has a highly redundant mapping from genotypes (6.4×10^{13}) to phenotypes (256). Such redundancy is heterogeneously distributed among phenotypes, with the most abundant phenotype possessing about 10% of the entire genotype space while some other phenotypes never appeared in our samples (Fig. 1a). By examining the undirected, weighted phenotype network, we found that more abundant phenotypes have more access to different phenotypes (Fig. 1b) and more tendency to mutate into certain neighboring phenotypes (Fig. 3a).

We chose two phenotypes, 180 and 240, as targets, and observed that in addition to having considerably different degrees, 41 and 233, the two phenotypes

have very different community structures (Fig. 5). This suggests that target 180 is much more difficult to find, not only because it was connected to fewer neighbors, but also because it is located in a small and distant community. It was also interesting to see that in the directed phenotype network resulting from setting 180 as target, fitness was less effective at guiding evolution, since fitness and in-/out-degree of a phenotype are less correlated (Fig. 6), i.e., reaching a fitter phenotype at a current stop would not necessarily lead to more promising paths to finding the target.

The search performance of an evolutionary algorithm can vary considerably with different problem instances. Our study provides a quantitative investigation into this issue using complex network analysis. That a specific target is hard to reach can have multiple explanations: (1) the target is under-represented in genotype space; (2) the target is connected to only a few phenotypes in phenotype space; (3) the target belongs to a small community distant from the rest of the phenotypes in that space; and (4) setting the target has wired the connections among phenotypes in a way that renders following fitter phenotypes in order to reach the target a less effective strategy. We hope our observations can be found useful to inspire more intelligent search mechanisms that are able to overcome these challenges.

Acknowledgements. This research was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada Discovery Grant RGPIN-2016-04699 to T.H., and the Koza Endowment fund provided to W.B. by Michigan State University.

References

1. Kell, D.B.: Genotype-phenotype mapping: genes as computer programs. Trends Genet. **18**(11), 555–559 (2002)
2. de Visser, J.A.G.M., Krug, J.: Empirical fitness landscapes and the predictability of evolution. Nat. Rev. Genet. **15**, 480–490 (2014)
3. Schaper, S., Louis, A.A.: The arrival of the frequent: how bias in genotype-phenotype maps can steer populations to local optima. PLoS One **9**(2), e86635 (2014)
4. Catalan, P., Wagner, A., Manrubia, S., Cuesta, J.A.: Adding levels of complexity enhances robustness and evolvability in a multilevel genotype-phenotype map. J. R. Soc. Interface **15**(138), 20170516 (2018)
5. Banzhaf, W.: Genotype-phenotype-mapping and neutral variation—a case study in Genetic Programming. In: Davidor, Y., Schwefel, H.-P., Männer, R. (eds.) PPSN 1994. LNCS, vol. 866, pp. 322–332. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-58484-6_276
6. Smith, T., Husbands, P., O'Shea, M.: Neutral networks and evolvability with complex genotype-phenotype mapping. In: Kelemen, J., Sosík, P. (eds.) ECAL 2001. LNCS (LNAI), vol. 2159, pp. 272–281. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44811-X_29
7. Rothlauf, F., Goldberg, D.E.: Redundant representations in evolutionary computation. Evol. Comput. **11**(4), 381–415 (2003)
8. Hu, T., Banzhaf, W., Moore, J.H.: The effect of recombination on phenotypic exploration and robustness in evolution. Artif. Life **20**(4), 457–470 (2014)

9. Hu, T., Payne, J., Banzhaf, W., Moore, J.H.: Evolutionary dynamics on multiple scales: a quantitative analysis of the interplay between genotype, phenotype, and fitness in linear genetic programming. Genet. Program. Evolvable Mach. **13**(3), 305–337 (2012)
10. Newman, M.E.J., Engelhardt, R.: Effects of selective neutrality on the evolution of molecular species. Proc. R. Soc. B **265**(1403), 1333–1338 (1998)
11. Wagner, A.: Robustness, evolvability, and neutrality. Fed. Eur. Biochem. Soc. Lett. **579**(8), 1772–1778 (2005)
12. van Nimwegen, E., Crutchfield, J.P., Huynen, M.A.: Neutral evolution of mutational robustness. Proc. Natl. Acad. Sci. **96**(17), 9716–9720 (1999)
13. Galvan-Lopez, E., Poli, R.: An empirical investigation of how and why neutrality affects evolutionary search. In: Cattolico, M. (ed.) Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1149–1156 (2006)
14. Hu, T., Banzhaf, W.: Neutrality and variability: two sides of evolvability in linear genetic programming. In: Proceedings of the 18th Genetic and Evolutionary Computation Conference (GECCO), pp. 963–970 (2009)
15. Hu, T., Banzhaf, W.: Neutrality, robustness, and evolvability in genetic programming. In: Riolo, R., Worzel, B., Goldman, B., Tozier, B. (eds.) Genetic Programming Theory and Practice XIV. GEC, pp. 101–117. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-97088-2_7
16. Nickerson, K.L., Chen, Y., Wang, F., Hu, T.: Measuring evolvability and accessibility using the Hyperlink-Induced Topic Search algorithm. In: Proceedings of the 27th Genetic and Evolutionary Computation Conference (GECCO), pp. 1175–1182 (2018)
17. Brameier, M.F., Banzhaf, W.: Linear Genetic Programming. Springer, Boston (2007). https://doi.org/10.1007/978-0-387-31030-5
18. Barábasi, A.L.: Network Science. Cambridge University Press, Cambridge (2016)
19. Barrat, A., Barthélemy, M., Vespignani, A.: Dynamical Processes on Complex Networks. Cambridge University Press, Cambridge (2008)
20. Hu, T., Payne, J.L., Banzhaf, W., Moore, J.H.: Robustness, evolvability, and accessibility in linear genetic programming. In: Silva, S., Foster, J.A., Nicolau, M., Machado, P., Giacobini, M. (eds.) EuroGP 2011. LNCS, vol. 6621, pp. 13–24. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20407-4_2
21. Hu, T., Banzhaf, W.: Quantitative analysis of evolvability using vertex centralities in phenotype network. In: Proceedings of the 25th Genetic and Evolutionary Computation Conference (GECCO), pp. 733–740 (2016)
22. Csardi, G., Nepusz, T.: The igraph software package for complex network research. InterJournal Complex Syst. **1695**, 1–9 (2006). http://igraph.org
23. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E **69**, 026113 (2004)
24. Masuda, N., Porter, M.A., Lambiotte, R.: Random walk and diffusion in networks. Phys. Rep. **716**, 1–58 (2017)