Finding Needles in Haystacks is Not Hard with Neutrality

Tina Yu[1] and Julian Miller[2]

[1]ChevronTexaco Information Technology Company, San Ramon CA 94583, U.S.A.
tiyu@chevrontexaco.com, http://www.improvise.ws

[2]School of Computer Science, University of Birmingham, Birmingham B15 2TT, U. K.
j.miller@cs.bham.ac.uk, http://www.cs.bham.ac.uk/~jfm

Abstract. We propose building neutral networks in needle-in-haystack fitness landscapes to assist an evolutionary algorithm to perform search. The experimental results on four different problems show that this approach improves the search success rates in most cases. In situations where neutral networks do not give performance improvement, no impairment occurs either. We also tested a hypothesis proposed in our previous work. The results support the hypothesis: when the ratio of adaptive/neutral mutations during neutral walk is close to the ratio of adaptive/neutral mutations at the fitness improvement step, the evolutionary search has a high success rate. Moreover, the ratio magnitudes indicate that more neutral mutations (than adaptive mutations) are required for the algorithms to find a solution in this type of search space.

1 Introduction

Many optimization problems can be defined as search problems: all possible solutions constitute the search space; the objective is to find the one that best solves the given problem. In many search problems, there is sufficient correlation between search points to allow heuristic search techniques to find good solutions faster than random search. However, there is one kind of search space, needle-in-haystack, which is difficult for heuristic search algorithms to outperform random search.

In a needle-in-haystack type of search space, a solution is either a needle or a piece of hay. In other words, a search algorithm either finds a perfect solution (the needle) or otherwise (the hay). No knowledge about the location of the needles can be obtained from examining the hays. In this kind of situation, a heuristic search algorithm works like a random search algorithm. When the number of solutions in the search space is small, finding a good solution is difficult, no matter what search algorithm one uses.

Evolutionary algorithms are heuristic search algorithms where the search process is directed by fitness: only solutions that have competitive fitness are considered in the solution population pool. An evolutionary algorithm selects solutions with good fitness and uses them to find solutions with better fitness.

What if the search space only has two possible fitness values (one for the needles and the other for the hay)? Evolutionary algorithms seem to become helpless in this kind of situation. In this study, we investigate building a network within the "hay" to provide a trail for the search process. In this way, the discovery of the "needle" solutions may become easier. Since the network connects solutions with the same

fitness (within the hays), it is called "neutral network". Moreover, an evolutionary algorithm utilizing such a network for search is said to support neutrality, a term borrowed from evolutionary biology.

Neutral theory [3] in evolutionary biology has inspired our work in incorporating neutrality in evolutionary algorithms (Section 2). Previously, we have devised a Genetic Programming (GP) system that utilizes neutrality to search for problem solutions [12]. When applied to a Boolean function problem, the solutions were found with a higher success rate. We analyzed the results and identified that high success rates are associated with a particular adaptive/neutral mutation ratio pattern. This work extends that finding by applying the system on four needles-in-haystack problems. Moreover, the relationship between neutrality and mutation rates is investigated.

The paper is organized as follows. Section 2 provides the background of this work. It first explains neutral theory in evolutionary biology and then gives a summary of our previous work on using neutrality for evolutionary search. Section 3 reviews related work. In Section 4, the GP system is described. Section 5 presents the four investigated needle-in-haystack problems. In Section 6, the experimental setup is given. The results are presented in Section 7. Section 8 discusses the experimental results and Section 9 gives our conclusions.

2 Background

The theory of natural evolution established by Darwin has had profound impact on biology. Most biologists are convinced that selection acting on advantageous mutations is the driving force of evolution. It was not until the late 1970s when molecular data became available, that the theory was challenged. In particular, Motoo Kimura found that the number of mutant substitutions in amino acid sequences of hemoglobin was too large to be explained by the theory of natural selection. Based on this discrepancy, he proposed the neutral theory, which states that most mutants at the molecular level in evolution are caused by random genetic drift rather than by natural selection [3]. In other words, the mutants involved are neither advantageous nor disadvantageous to the survival or reproduction of the individual. Around the same time, a similar theory was published by King and Jukes [4]. The two papers have provoked much controversy and are still subject to strong debate [6].

Darwinism has been the dominating principle behind the implementation of evolutionary algorithms: the evolved entities contain no *explicit* neutral mutants. In this way, a mutant is either advantageous or disadvantageous. Selection acts upon them to propagate those that are advantageous.

But can neutral mutations (those are neither advantageous nor disadvantageous) benefit evolutionary search? To investigate these questions, we have devised a methodology for systematic study of this subject [12]. In particular, we measure the number of neutral mutations that occur in the evolved entities during evolutionary search. In this way, the impact of neutrality on search performance can be analyzed quantitatively. Using this approach, we have studied a Boolean function problem. The results show that there is a positive relationship between neutral mutations and success rate: the larger the allowed neutral mutations quantity the greater is the possibility for the evolutionary search to find a solution.

The amount of neutral mutations is measured in the selection step, which evaluates both the fitness and the number of neutral mutations in the evolved entities. More

precisely, an offspring solution is selected to replace the current winner only when it has a better fitness or it has the same fitness but its neutral mutants are within a specified range (the Hamming bound). On can envisage all solutions with the same fitness and satisfy the Hamming bound are connected in a network (neutral network). The search process selects solutions in the network one after another in the manner of a neutral walk. We found that such a walk can lead to a solution with a better fitness if it satisfies the fitness improvement criterion. The criterion is concerned with the ratio of adaptive and neutral mutations. The analysis indicated that when this ratio for the neutral walks was close to the ratio for the fitness improvement, a high probability of success occurred.

Adaptive and neutral mutations play different roles in the evolutionary search: adaptive mutations exploit the accumulated beneficial mutations while neutral mutations provide an exploratory power by maintaining genetic diversity. Under the dynamics of the evolutionary process, neutral mutants may contribute to the fitness later. For an evolutionary search to be successful, it requires a balance between exploitation and exploration. The ratio between adaptive and neutral mutations is therefore an appropriate fitness improvement criterion in evolutionary search.

3 Related Work

Tomoko Ohta studied the ratio of adaptive and neutral mutations in DNA sequence data to test her nearly neutral theory [10]. Unlike Kimura's neutral theory, which believes that a mutation is either selective or neutral, the nearly neutral theory believes that there is a class of mutations (very weakly selected mutations) whose behavior is influenced by both genetic drift and selection. Moreover, in small population genetic drift dominates while in large populations selection is more influential.

She tested this theory by comparing synonymous and nonsynonymous mutations of 49 mammalian genes. The results agree with the theory. Later, McDonald and Kreitman also studied the relative number of synonymous and nonsynonymous mutations within a species. The results, however, were contrary to the theory [8].

Claus Wilke and colleagues studied the evolution of digital organisms (as computer programs) using the Avida system [11]. They reported that under high mutation rates, an organism that has its neighbors (those accessible by one mutation step) with a similar fitness (not necessary the same fitness) had a higher reproduction rate. The reason is that such flat fitness landscape is more robust against mutations than a fitness landscape that has high and narrow peak. Although they didn't mention neutral networks, where the neighbors have the same fitness, one would expect the same findings.

This result is not surprising, as there have been various research reports about the buffer effect of neutral mutations against disruptive mutations [1]. The existence of neutral networks hence is beneficial to the organism's reproduction rate. We have reported a similar result on a Boolean function landscape in our previous work [12]. However, with needle-in-haystack landscapes, an opposite result is found: high mutation rates also give high reproduction rates. The details are given in Section 8.2.

Marc Ebner and colleagues also studied the relationship between neutral networks and evolvability [2]. Particularly, they investigated a search space with 2^{16} possible fitness values. Moreover, these fitness values were divided into 64 groups. Their selection criterion was similar to ours in that they allowed an individual with better or

equal fitness to replace the current winner. They experimented with 3 different sizes of neutral network (1, 2^{112}, 2^{320}) using a single point mutation. They reported that the larger the network (more neutrality), the higher the average population fitness. This result is consistent with that in our previous paper where wider ranges of neutrality levels (6) and mutation rates (11) were investigated [12].

4 Cartesian Genetic Programming

Cartesian Genetic Programming (CGP) was first formally proposed in [9]. In CGP, a genotype is an integer string that encodes an indexed, feedforward, acyclic graph. Unlike the parse tree representation in the standard GP [5], a genotype-phenotype mapping is used to create the graph phenotype from the integer string genotype. Each node in the genotype contains a multiple number of genes; some of them are link values. The nodes that are not involved in the linked path between the inputs and outputs of the program are inactive in the phenotype. Such nodes have no effect on the behavior of the phenotype.

For example, Figure 1(a) is a genotype for even-5-parity function (described in Section 5) that is mapped into a phenotype in 1(b). The alleles in bold take value from the set `{0,1}`, which represent the Boolean function `xor` and `eq` respectively. The function inputs are denoted by labels 0 to 4 (and in the phenotype as x_0 to x_4). The node outputs are labeled from 5 to 10. There are six nodes in the genotype; each has 3 genes (two input link values and one Boolean function value). The genotype is read from the last gene on the right (10) towards the left. Since 10 refers to the rightmost node (with genes 8 4 1) it is in the phenotype: it is an EQ node with one input connected to the even-parity input 4, and the other is connected to the node output 8. So the node with output 9 is not involved in the phenotype. This inactive node is the only node that refers to the node with output 7, so this too is inactive (in gray in Figure 1(a)). All other nodes outputs are referenced so are involved in the phenotype.

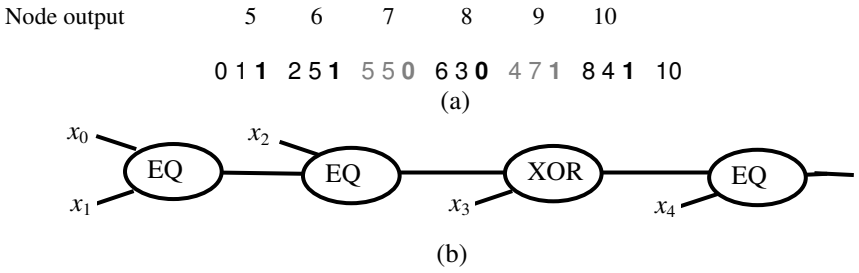

Figure 1: A genotype (a) and its phenotype (b) for even-5-parity.

Mutations applied to inactive genes are neutral while on active genes can be adaptive or neutral (see Section 4.1). This genotype representation, in which some genes are active and others inactive, allows neutral mutations to be measured as Hamming distance. This is described in Section 4.1.

4.1 Neutrality Measured as Hamming Distance

When a mutation operation generates a different program with the same fitness, it involves a number of neutral mutations. Neutral mutations on active genes are the results of functional redundancy or introns, hence represent implicit neutrality. In

contrast, neutral mutations on inactive genes represent explicit neutrality. Regardless the source of neutrality (implicit or explicit), the overall amount of neutral mutations between the parent-offspring genotypes pair can be measured according to their Hamming distance. For example, the following two genotypes have Hamming distance of 13. The number of active genes changes between the two genotypes is 12 (the node with output link number 6 contributes three active genes changes because it was inactive in Genotype 1 but becomes active in Genotype 2). The number of inactive genes changes is two (corresponding to the node with output link number 4). Note that the link path in the graph is traversed starting from the last node with the final output link. For example, in Genotype 1, the last node (with output link 7) has input links of 5 and 1. This makes node with output link 5 active. This node has input links of 3 and 1. It makes the node with output link 3 active. The nodes with output link 4 and 6 are not used as input links for any active nodes, hence are inactive.

Output link	2	3	4	5	6	7
Genotype 1	0 1 **1**	0 2 **1**	0 1 **0**	3 1 **0**	4 3 **1**	5 1 **1**
Genotype 2	0 0 **1**	2 1 **0**	1 3 **0**	3 0 1	2 5 **1**	6 5 **0**

Using Hamming distance to measure neutral mutations has two advantages:

- It provides a quantitative measurement of neutrality in evolutionary search.
- It provides a flexible way to implement neutrality in evolutionary algorithms.

5 Even-Parity Problems

Even-parity is a widely used benchmark problem in GP and Evolvable Hardware communities. The problem can be defined with a different number of Boolean inputs, e.g. even-3-parity, even-4-parity and so on. An even-parity function returns the value `True` if and only if an even number of inputs are `True`, otherwise it returns `False`.

Various research results show that the difficulty of the problem is highly dependent on the functions selected to construct the solution [14,7]. In particular, when only `xor` and/or `eq` functions are used, this problem has a needle-in-haystack property: there are only two possible fitness values. The search algorithm either finds a perfect solution or gets a solution that receives half of the mark. For example, with even-5-parity, the number of inputs is five (each can be either `True` or `False`). There are 2^5 different possible input combinations, hence 32 test cases. The fitness of a program is the number of the correct outputs for the 32 test cases. An even-5-parity constructed using `xor` and `eq` would either have fitness 16 or 32.

Table 1 shows the four such problems we investigate in this work. Note that the random search success rates were calculated by randomly generating 1,000,000 programs, except even-12 parity, where 4,000,000 trials were made (since it's a more difficult problem). No solution with fitness 4096 was found in this set of trials.

Problem	*even-5-parity*	*even-8-parity*	*even-10-parity*	*even-12-parity*
Function Set	xor, eq	eq	eq	eq
Terminal Set	x_1 to x_5	x_1 to x_8	x_1 to x_{10}	x_1 to x_{12}
Possible fitness	16 and 32	128 and 256	512 and 1024	2048 and 4096
# of Test Cases	32	256	1024	4096
Random Search (success rate)	0.81%	0.063%	0.0054%	0%

Table 1: Four needle-in-a-haystack even-parity problems.

6 Experimental Setup

6.1 Evolutionary Algorithm

The algorithm used for the experiments is a form of 1+λ evolutionary strategy, where λ=4, i.e. one parent with 4 offspring (population size 5). The algorithm is as follows:

1. Randomly generate an initial population of 5 genotypes with the lowest possible fitness and select one (randomly) as the winner.
2. Carry out point-wise mutation on the winning parent to generate 4 offspring;
3. Construct a new generation with the winner and its offspring;
4. Select a winner from the current population using the following rules:
   - If any offspring has a better fitness, it becomes the winner.
   - Otherwise, an offspring with the same fitness is randomly selected. If the parent-offspring pair has a Hamming distance within the permitted range (see Section 6.2), the offspring becomes the winner.
   - Otherwise, the parent remains as the winner.
5. Go to step 2 unless the maximum number of generations reached or a solution with needle fitness is found.

The mutation sites were selected randomly. A simple type checking is performed to make sure the new offspring is valid. If the gene site contains a function, it is changed to an alternative function. If the gene site is a connection link, it is replaced with another equally valid connection link.

6.2 Control Parameters

The genotype has 100 nodes; each has 3 genes (see Section 4). The total number of genes is therefore 300. Eleven different mutation rates and 7 neutrality levels were used in the experiments. Table 2 summarizes the parameters used for the experiments.

Parameters	Values
Genotype Length	300
Mutation Rate (%) on genotype	1,2,4,6,8,10,12,14,16,18,20
Max Generation	10,000
Neutrality Level (Hamming distance range)	0,50,100,150,200,250,300
Population Size	5
Number of Runs	100

Table 2: Summary of control parameters.

7 Results

We do not present the results of average population fitness because they do not give performance information in the needle-in-haystack type of problems (where only two fitness values are possible). For example, with even-5-parity, the average fitness in the initial population is 16. This value remains as the average fitness until a solution with fitness 32 is found. This average fitness pattern is the same for all neutrality level and mutation rate implementation, hence is not useful to identify the impact of neutrality on the search performance. Instead, we give the results of success rate.

For even-5-parity, all implementations (mutation rates and Hamming distances) have a 100% successful rate, i.e. all 100 runs find a solution (see Figure 2A). In contrast, even-8-parity (a harder problem) has lower success rates in some cases. In particular, the combination of low Hamming distance and low mutation rate has produced some unsuccessful runs (see Figure 2B).

When mutation rate is 1% or 2%, a small amount of neutrality (50) is able to improve success rates. However, when mutation rate is 4%, a neutrality level equal to Hamming distance 100 is required to improve the performance. For example, increasing mutation rate from 2% to 4% gives Hamming distance 0 a success rate jump from 82% to 94%. In comparison, increasing neutrality level from 0 to 50 on mutation rate 4% does not improve success rate. This suggests that raising mutation rate has a stronger impact than raising neutrality level on the evolutionary search. The relationship between neutrality and mutation rates will be discussed in more details in Section 8.2.

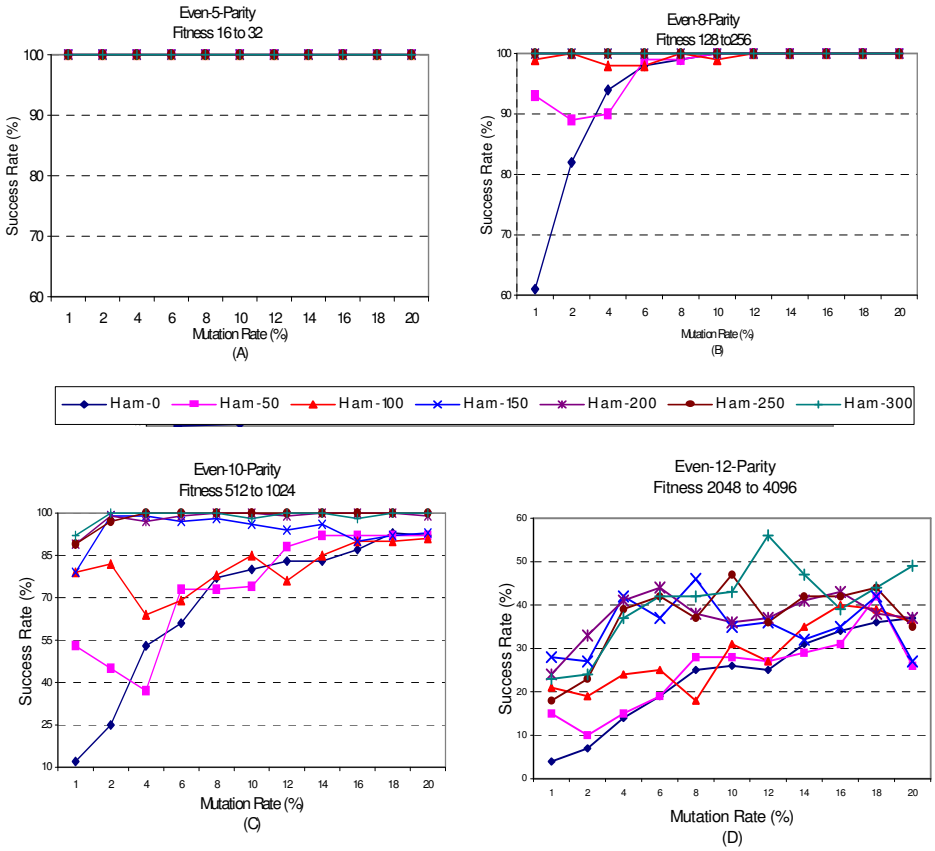

Figure 2: Success rate results.

Regardless of mutation rates, increasing neutrality level beyond 150 does not improve or impair the performance; all of them have 100% success rate. This suggests that equilibrium between the benefits of exploitation and exploration is reached at this point. Indeed, the adaptive/neutral mutation analysis (see Section 8.1) shows that the exploitation/exploration ratios are very similar for all Hamming distance beyond 150 (see Figure 3B). Increasing neutrality or mutation rates do not affect this equilibrium.

Even-10-parity is harder than even-8-parity: there are more unsuccessful runs, especially when low Hamming distance values were used (see Figure 2C). Similar to even-8-parity, the success rate remains approximately the same after a certain Hamming distance value is reached (200 is the equilibrium point, see Figure 3C). Moreover, mutation rates are more influential in this problem: neutrality level 150 is required to give consistent improvement of success rates (in contrast to neutrality level 100 in even-8-parity).

Even-12-parity is the hardest among all; none of the implementations has 100% success rate (see Figure 2D). Although not as precise as other problems, even-12-parity reaches the equilibrium point around neutrality level 200. We also made 100 experimental runs using Hamming distance 0 and 100% mutation rate. Among them, 48 runs find a solution (48% success rate). This suggests that high neutrality and mutation rate are not sufficient for the search algorithm to find a solution to this

problem. Modification of other parameters, such as gene length and the maximum number of generation, is required to improve performance.

The performance of the 7 different Hamming distances implementations can be roughly divided into two groups: the first group consists of neutrality level 0, 50, 100 while the second group consists of neutrality level 150, 200, 250 and 300. In the first group, increasing mutation rates increases success rates while in the second group little performance improvement is gained after mutation rate exceeds 4%. Nevertheless, the group with higher neutrality level also gives higher success rates.

8 Analysis and Discussion

The experimental results for applying neutrality on four needle-in-haystack problems give the following patterns:

- Neutrality does not have impact on success rates when the number of solutions in the search space is large (even-5-parity).
- However, when the difficulty level is slightly increased (even-8-parity), any amount of neutrality improves the success rates for 1% or 2% mutation rates.
- Increasing mutation rates also improves success rates. Yet, with a sufficient amount of neutrality, the success rates can be further increased. The required neutrality level is problem dependent; the harder the problem, the higher the required neutrality levels (100 for even-8-parity, 150 for even-10 & 12-parity).
- There is an equilibrium point, which gives the exploration/exploitation balance corresponding to the maximum possible performance. Increasing neutrality or mutation rates beyond this point has little effect on the performance.
- For more difficult problems (even-12-parity), high neutrality level and mutation rate is not sufficient for the evolutionary algorithm to find a solution.

These results indicate that for this type of needle-in-haystack fitness landscapes, the harder the problem is (in terms of the number of solution in the search space) the higher the neutrality level is required to give performance improvement. Moreover, it is important to note that neutrality does not impair search performance, in cases where it does not give performance improvement.

8.1 Adaptive/Neutral Mutations Analysis

We have also studied the adaptive/neutral mutation ratios for different Hamming distance and mutation rate implementations. Previously, we have proposed a hypothesis that when the adaptive/neutral mutation ratio during neutral walk is close to that of the fitness improvement step, the evolutionary search process is more likely to be successful (see Section 2). The four sets of experimental results are analyzed in the following subsections:

Even-5-Parity:

For even-5-parity, Hamming distance 150, 200, 250 and 300 implementations give the neutral walk adaptive/neutral mutation ratios (in dashed-lines) that are very close to those of fitness improvement step (in solid-lines). The difference between the two ratios is between 0.04 and 0.01 (see Figure 3A). Figure 2A indicates that these implementations have 100% success rates, which agrees with the hypothesis.

Hamming distance 50 and 100 implementations give the ratio differences that are larger than those given by other Hamming distance implementations do. However,

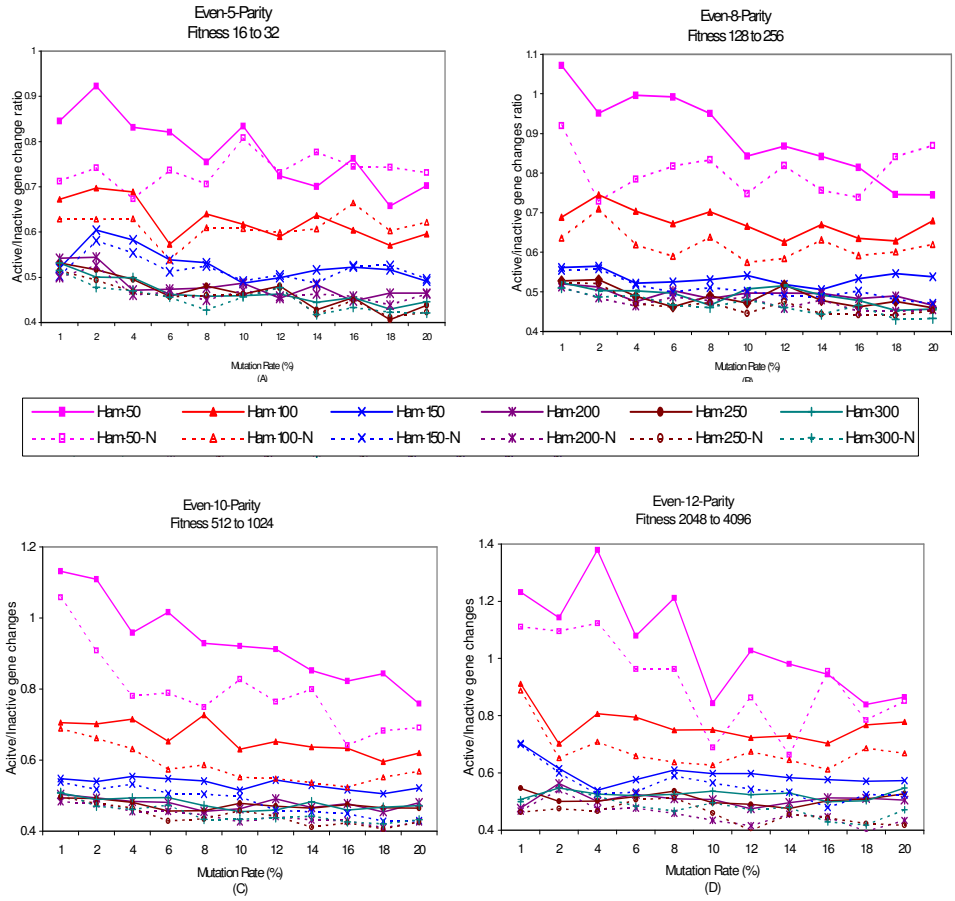

Figure 3: Adaptive/neutral Mutations ratio results.

they also have 100% success rates. As suggested in [12], we believe the magnitude of these ratios is also important to the success of evolutionary search.

The active/inactive gene change ratios for fitness improvements are all below 1 in this problem (regardless of neutrality level and mutation rates). This means that when a genotype of fitness 16 is transformed to a genotype of fitness 32, more inactive genes than active genes were changes (if the ratio is 1, an equal amount of active and inactive gene were changed). Such ratio pattern is also associated with 100% success

rate. This indicates that for this problem, as long as more inactive genes (than active genes) were changed during the evolutionary process, the search will be successful.

Even-8-Parity:

With even-8-parity, Hamming distances 150, 200, 250 and 300 give similar active/inactive gene changes patterns as those in even-5-parity. The ratios for neutral walk and fitness improvement are close to each other; the range of the ratios is also between 0.6 and 0.4 (see Figure 3B). Similarly, they also have 100% success rates.

Hamming distance-100 also gives the two ratios that are not too far from each other (0.02 to 0.09). The three with the largest gap (mutation rates 4%, 6%, 10%) are associated with a slightly lower success rates. This is consistent with our hypothesis.

Hamming distance 50 implementation has a noticeable lower success rate when mutation rate is 1%, 2% or 4%. The active/inactive gene changes give the two ratios that are farther apart from each other. Moreover, the ratio for fitness improvement is higher than that in even-5-parity, i.e. more active gene changes were made. For example, mutation rate 1% has an active/inactive gene changes ratio of 1.07 (43/40). This ratio range (> 1) indicates that more adaptation (than genetic drift) was made for fitness improvement. Since this ratio is associated with a lower success rate, we believe that inactive gene change (exploration) is more important for this problem. Neutrality 50 with 1% mutation rate does not provide sufficient exploration, hence leads to a lower success rate.

Mutation rate 2% and 4% give larger ratio gaps (0.22) than that of 1% mutation rate. Their success rates are also lower, which agrees with our hypothesis. Although the ratio difference for 6% mutation rate is similar to that of 1% mutation rate, it receives a higher success rate. We believe that this is because 6% mutation rate provides more exploration than 1% mutation rate (a lower ratio). In other words, high mutation rates complement neutrality level to provide exploration and achieve successful runs. We will discuss this more in Section 8.2.

Even-10-Parity:

The ratio pattern of even-10-parity is similar to even-8-parity in the following Hamming distance implementations: 200, 250 and 300. Figure 2C shows that they also have close to 100% success rates, just like those in even-8-parity. However, Hamming distance-150 implementation gives a different ratio pattern: the higher the mutation rates, the larger the ratio differences. For example, when mutation rate 1% is used, the active/inactive gene change ratio during neutral walk is 0.539 while the ratio for fitness improvement is 0.547 (a difference of 0.008). This distance increases as mutation rate increases. When mutation rate reaches 20%, the distance is 0.092. According to our hypothesis, high success rate is associated with low ratios distance. Indeed, Figure 2C shows that 20% mutation rate gives the lowest success rate; these success rates increase as mutation rates decrease.

The active/inactive gene change associated with 100% success rate is lower than the previous two problems (0.4 to 0.47). This suggests that more exploration is required for successful search in this problem. Neutrality level 50 and 100 implementations do not give enough inactive gene changes, hence are associated with lower success rates. Nevertheless, increasing mutation rate increases inactive gene changes (a lower ratio) and improves success rates.

Even-12-Parity:

The ratio pattern of even-12-parity has higher active gene changes than those in the other problems (see Figure 3D). They are also associated with lower success rates (see Figure 2D). This suggests that the algorithm does not provide sufficient inactive gene change (exploration) for the search to find a solution. Moreover, each time problem difficulty is increased (from even-5 to even-8 to even-10 to even-12), more inactive gene changes were required for the evolutionary search to be successful. This suggests exploration is more important than exploitation for search in needle-in-haystack type of space.

8.2 Relationship between Neutrality and Mutation Rates

The four needle-in-haystack problems show that both neutrality and mutation rates have a negative impact on the active/inactive gene change ratio: the higher they are, the lower the ratios. In other words, increasing neutrality or mutation rates increases *inactive* gene changes (more neutral mutations). Such increase of exploration also improves success rates.

However, this relationship does not apply to search space that has more than two possible fitness values. In [12], the problem we investigated has 8 possible fitness values. The study indicates that higher mutation rates give higher *active* gene changes (more adaptive mutations). Such increase of exploitation is beneficial when population average fitness is low but detrimental when the population has become fit. Moreover, increasing neutrality level does not always increase exploration. Nevertheless, higher neutrality level always gives higher success rates.

Yet, what is the relationship between neutrality and mutation rates? Can they replace each other? In the four problems we investigated, they both increase inactive gene changes. Thus, they complement each other in this type of needle-in-haystack problems to improve success rates. However, the following have to be noted:

- There is no quantitative measurement on which neutrality level gives the same performance as a certain mutation rate does. For example, 50 neutral mutations are about 16% of the total number of 300 genes. Yet, neutrality level 50 with 1% mutation rate does not give the same performance as neutrality level 0 with 16% mutation rate (see Figure 1).
- Once equilibrium point is reached, success rate is approximately the same regardless of neutrality level and mutation rate, i. e. the relationship between neutrality and mutation rate is irrelevant to the search performance.

9 Conclusions

Needle-in-haystack problems are hardly studied in the evolutionary computing community, possibly due to the difficulty of devising an evolutionary algorithm that outperforms random search. We investigated a set of four such problems and demonstrated that neutrality helps evolutionary algorithms to discover solutions in most cases. In situations where neutrality does not improve performance, it does not impair the search performance. This is an important message as it opens a new way to tackle the needle-in-haystack type of problems using an evolutionary approach.

The analysis of adaptive/neutral mutations ratios supports the hypothesis proposed in our previous work: when the ratio given by neutral walk is close to that of fitness improvement, the evolutionary search has a high success rate. Moreover, we have

extended our study on ratio magnitude and deduced that a ratio of <1 (more exploration than exploitation) is required for the evolutionary search to be successful in needle-in-haystack type of search space.

Different levels of neutrality attain equilibrium of exploration and exploitation for the 4 different problems; increasing neutrality beyond such level gives insignificant advantage/disadvantage to the search performance. Moreover, the harder the problem, the higher neutrality level for its equilibrium.

Both neutrality and mutation rates have a positive impact on exploration in the needle-in-haystack type of problems. Consequently, they can supplement each other to improve success rates. However, this is not true for other type of problems. For example, our previous work on a Boolean function landscape gives a different result. We are currently investigating other types of problems to understand better the relationship between neutrality and mutation rates.

References

[1] Altenberg, L.: The evolution of evolvability in genetic programming. In*: Advances in Genetic Programming*, K. E. Kinner Jr., ed. MIT Press, (1994) 47-74.

[2] Ebner, M., Langguth, P., Albert, J., Shackleton, M. and Shipman, R.: On neutral networks and evolvability. In: *Proceedings of the 2001 Congress on Evolutionary Computation*, IEEE Press (2001) 1-8.

[3] Kimura, M.: *The Neutral Theory of Molecular Evolution*. Cambridge Univ. Press (1983).

[4] King, J. L. and Jukes, T. H.: Non-Darwinian evolution. *Science* Vol. 164 (1969) 788-798.

[5] Koza, J. R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press (1992).

[6] Kreitman, M.: The neutral theory is dead. Long live the neutral theory. *BioEssays*, Vol. 18 (1996) 678-682.

[7] Langdon, W. B. and Poli, R.: Why "building blocks" don't work on parity problems. Technical report number CSRP-98-17. The University of Birmingham, July 13, 1998.

[8] McDonald, J. H. and Kreitman, M.: Adaptive protein evolution at the Adh locus in Drosophila. *Nature* Vol. 351 (1991) 652-654.

[9] Miller, J. F. and Thomson, P. Cartesian genetic programming. In: *Proceedings of the Third European Conference on Genetic Programming. LNCS*, Vol. 1802 (2000) 121-132.

[10] Ohta, T.: The nearly neutral theory of molecular evolution. In: *Annual Reviews Ecology & Systematic*, Vol. 23 (1992) 263-286.

[11] Wilke, C. O., Wang, J. L., Ofria, C., Lenski, R. E. and Adami, C.: Evolution of digital organisms at high mutation rate leads to survival of the flattest. *Nature,* Vol. 412 (2001) 331-333.

[12] Yu, T. and Miller, J.: Neutrality and the evolvability of Boolean function landscape. In: *Proceedings of the Fourth European Conference on Genetic Programming*. Springer-Verlag (2001) 204-217.

[13] Yu, T.: Structure abstraction and genetic programming. In*: Proceedings of the 1999 Congress on Evolutionary Computation*. IEEE Press (1999) 652-659.