Multi-Level Neutrality in Optimization

Colin G. Johnson

Abstract— **This paper explores the idea of *neutrality* in heuristic optimization algorithms. In particular, the effect of having multiple levels of neutrality in representations is explored. Two experiments using a *fitness-adaptive walk* algorithm are carried out: the first is concerned with function optimization with Random Boolean Networks, the second with a tunable neutral mapping applied to the hierarchical if-and-only-if function. In both of these cases it is shown that a two-level neutral mapping can be found that performs better than both non-neutral mappings and mappings with a single level of neutrality.**

I. Introduction

IN this paper we introduce a new representation scheme for representing the potential solutions in an optimization algorithm. This extends existing work by Shipman et al. [1] that shows how *neutrality* helps evolution to escape local optima. Our new representation is based on the idea of neutral representations acting at multiple levels. We show in two sets of experiments that this leads to improved search compared to non-neutral and single-level neutral evolution.

The remainder of this paper is structured as follows. Section II gives a number of definitions that are used throughout the paper. Section III gives an overview of related work, and section IV reviews the methods used in the experiments, which are fully described in sections V and VI. Finally, the experiments are discussed is section VII and some conclusions and suggestions for future work are given.

II. Definitions

Neutrality in a biological system [2] is a form of redundancy between the DNA encoding and the fitness of organisms. We can find sets of different DNA sequences that give rise to organisms which, placed in an identical environment, will have more-or-less the same fitness. A *neutral network* [3], [4] is defined as a set of connections on the space of DNA sequences, where these connections represent mutations and each labelled either as having a neutral of non-neutral effect on fitness (again, within a given environment).

In this paper we will expand this notion of neutrality to encompass redundancies between any two levels of a biological system. By levels we mean the various stages between the DNA sequence and the organism's large-scale behaviour. This is illustrated in figure 1. In each of these levels we can define an equivalence relation, grouping objects from that level together into an equivalence class if any pair of items from that class encode for the same object in the level above. That is, items within that equivalence class are substitutable for each other.

Colin G. Johnson is with the Computing Laboratory, University of Kent, Canterbury, Kent, CT2 7NF, England (email: C.G.Johnson@kent.ac.uk).

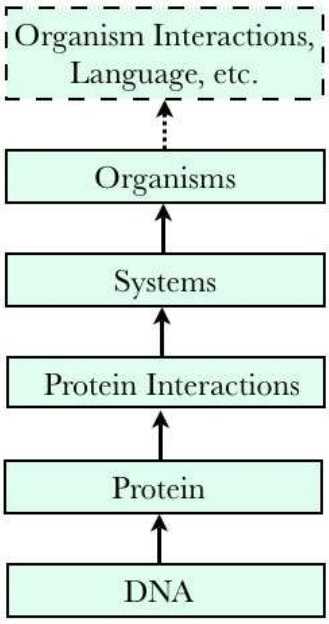

Fig. 1. Levels in a biological system between which neutral mappings can be found.

Given such an equivalence relation on the set of objects at each level, we can now define neutrality between any two levels: two objects in the lower level are regarded as related by a neutral mutation if that mutation is to an object in the same equivalence class; i.e. they encode for the same object in the higher level. In reality, these relations may be regarded as somewhat fuzzy; this is similar to the way in which, in the original definition of neutrality, we defined neutral changes as those which "more or less" leave the fitness unchanged.

We can clearly see that the traditional definition of neutrality fits into this defintion. Another example is the neutral encoding of proteins by DNA sequences, where many DNA sequences can encode for the same amino acid sequence. A further example is the relationship between protein sequence and protein interaction: a number of different amino acid sequences might fold into proteins which react in (more or less) the same way with another protein.

III. Related Work

In this section we review related work in two areas, *viz.* neutrality in biological systems and in bio-inspired computation.

978-1-4244-1823-7/08/$25.00

A. Neutrality in Biology

Neutrality was first significantly studied in biological systems by Kimura [5]. An overview of the work by Kimura and related work can be found in Kimura's 1983 book [2], and an overview of more recent work in this tradition can be found in a paper by Nei [6]. This work was focused on questions relating to *population genetics*, for example studying the proportion of diversity amongst the population that can be attributed to neutral drift rather than adaptation.

Work by Schuster [7] and Huynen and colleagues [8], [9] explored the *adaptive* role of neutrality within evolution. They argue that neutrality has an influence on the ability of the evolutionary process to escape from locally optimal niches. The argument is concerned with movement along a "neutral ridge" [10] over a number of generations. Such a ridge consists of a chain of successive neutral mutations. This enables escape from local optima in the following fashion. An organism with fitness f might be in an evolutionary niche such that any mutated offspring will have fitness less than or equal to f. However, a neutral mutation could represent a different genotypic representation for an organism that is phenotypically identical, and therefore also has fitness f. However, after one or more generations of these neutral moves, the new organism may have a genotype where a mutation exists that codes for a phenotype which has a fitness greater than f. Therefore, it would be able to move from that point to a fitness that is greater than f.

B. Neutrality in Bio-Inspired Computation

An important part of the process of solving a problem by a bio-inspired search algorithm (such as a genetic algorithm) is devising a representation for the problem. For some problems, there is an obvious representation (though alternative representations can be used). For other problems, devising a representation is a complicated part of the process.

In light of the discussion of neutrality in biology in the previous section, we can ask whether it makes sense to use a representation that has neutrality. We can also view this from the point of view of problem difficulty, and ask: are problems that necessarily have some neutrality in them (by the nature of its fitness distribution) easier for such heuristic search algorithms to solve than other problems?

This idea has been explored experimentally in a pair of papers by Ebner, Shackleton, Shipman and Harvey [1], [11] (this work is referred to as ESSH below). They generate neutrality by creating complex genotype-phenotype mappings, based on cellular automata and random boolean networks, and contrast these with a direct genotype=phenotype representation. These are then applied to function optimization problems using a mutation-based algorithm. They map out the extent of the neutral networks for these representations, and show that the direct mapping tends to converge early on low-fitness individuals, whilst the neutral representations facilitate the ongoing exploration of the search space, and, ultimately, higher fitness values being discovered.

In the remainder of this paper we will explore the impact of having multiple levels of neutrality in algorithms of the type studied by ESSH. Our *prima facie* case for exploring these is that the neutrality in real biological systems is distributed between the levels; we can hypothesise that this is a high-level adaptation rather than just an "implementation detail".

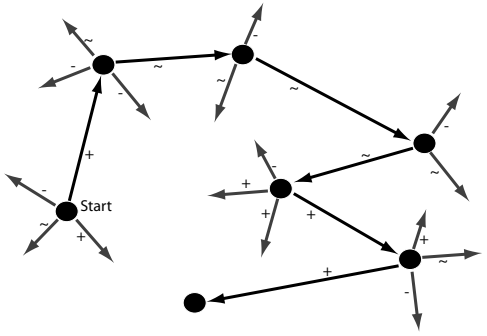

Fig. 2. The Fitness-Adaptive Walk optimization algorithm

IV. Methods

The optimization method that we use in the two experiments below is the *Fitness-Adaptive Walk* method of ESSH [1], [12]. This consists of a directed walk by a single point through the genotype space of a problem. Starting from a random genotype, the point moves around the space by the following hill-climbing rule:

1) Generate a set of mutations from the current point.
   a) If one or more of those mutation has fitness that represents an improvement to the current fitness, move to the one that increases the fitness the most.
   b) If one or more of those mutations has fitness that is equal to the current fitness, move to one of those at random.
   c) If all of the mutations represent a decrease in fitness, stop.
2) Return to 1

Basically, this is steepest-ascent hill-climbing combined with a neutral move if no fitness-improving move can be found. This algorithm is summarized in figure 2.

V. Experiment 1: Random Boolean Networks

In the first experiment our search space consists of *random boolean networks* (RBNs). These were introduced by Kauffman (see [13] for an overview) as a case study of a complex system with a tunable amount of complexity. One view of these is that they represent a caricature of the interactions that occur during gene expression, with promoter and repressor genes that control the expression of genes (including, in

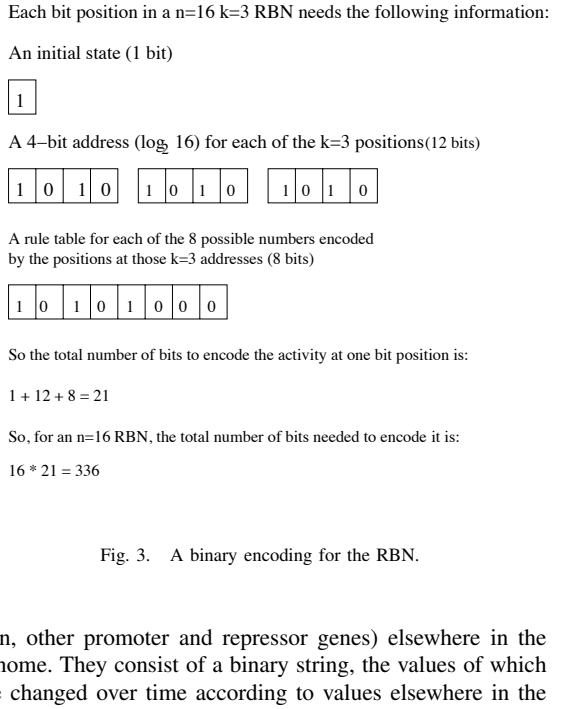

Fig. 3. A binary encoding for the RBN.

turn, other promoter and repressor genes) elsewhere in the genome. They consist of a binary string, the values of which are changed over time according to values elsewhere in the string. This process of updating the values is iterated until a stable point is found or a parameter-fixed iteration limit is reached.

Formally, an RBN is defined by the following 5-tuple of information:

1) The length of the binary string (n)
2) A parameter k which will specify the number of bits from the string that will act as inputs to a rule table (informally, we refer to these as the bits that *influence* the value).
3) A starting state for each position in the string
4) A list of n lists of length k, one for each position $1 \ldots n$ on the string, which specifies which k other positions in the string act as inputs to the rule table for that position.
5) A rule-table for each position in the string: the input for this is a k-bit binary number, the output a single bit.

This RBN specification itself can be represented by a binary string, which consists of a list of the initial states, followed by a list (in a binary encoding) of the k positions that influence each position, followed by a binary encoding of the rule-table for each position, i.e. a list of length 2^k of the output column of the rule table. It is assumed in this experiment that the actual values n and k are defined as constants elsewhere in the program, and are the same for all individuals. This description is summarised in figure 3.

Therefore, for the $n = 16$, $k = 3$ RBN, the search space consists of all 336-bit binary strings.

To calculate the fitness, the first part is to carry out the

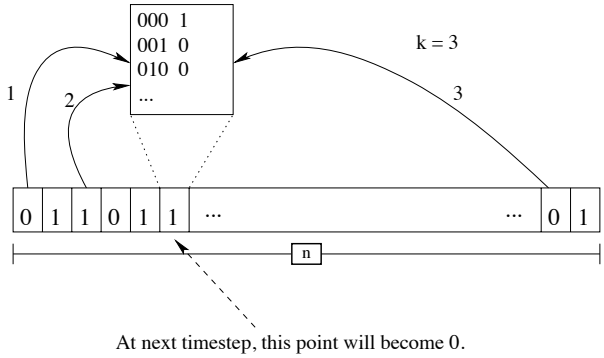

Fig. 4. Update step for the RBN.

iteration of the functions.

Initialize a length n string s_1 using the starting state
LOOP (until stability or 20 iterations):
  Create an empty string of length n called s_2
  FOREACH position p in the string s_1:
    Read the values of the k positions that influence that position
      Calculate the result of applying the rule-table at position p to those values
    Set the value $s_2[p]$ to that result
  END FOREACH
  Set s_1 equal to s_2
END LOOP

This process is illustrated in figure 4.

Secondly, the results of this process are converted into a fitness value, by taking the final value of the string, treating this as a binary number and using that as input into a fitness function. Mirroring the experiments by ESSH, a fitness table is generated into which each of the values from $0 \ldots 2^{16}$ are assigned a fitness value from the distribution $e^{100(r-1)}$ is used, where r is a random number from $[0, 1]$. This is so that there is a sparse distribution of high-fitness values. This is the fitness function that we use in this experiment. In the experiments below $n = 16$ and $k = 3$.

The application of the fitness-adaptive walk to the RBN model is illustrated in figure 5. ESSH experiment with this model [1] and show that it reaches a much higher fitness than a direct encoding (i.e. direct application of the fitness-adaptive walk to binary strings).

Our extension of this to multi-level neutrality exploits the fact that both the output from an RBN, and the representation of an RBN, are both binary strings. We apply the fitness-adaptive walk to a search space of RBNs, where the fitness is calculated as follows. The RBN is iterated, producing a binary string. This binary number is then interpreted as a (shorter-length) RBN (using the mapping from binary strings to RBNs illustrated in figure 3), and this RBN is iterated, and the fitness function applied to the final state of this second RBN. This process is summarised in figure 6.

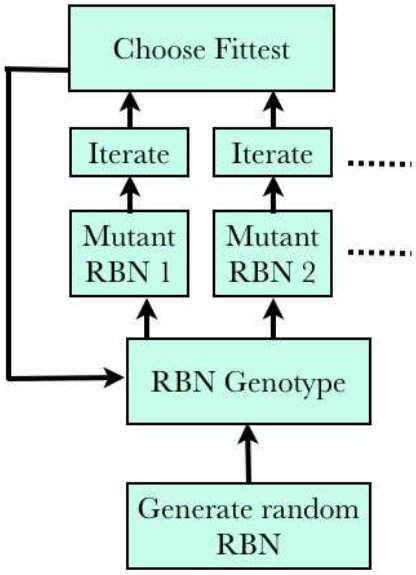

Fig. 5. Applying the fitness-adaptive walk to the RBNs.

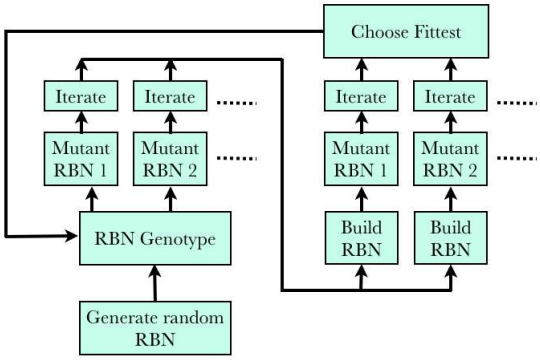

Fig. 6. Two-level neutrality for the RBN model.

This can be seen as loosely analogous to the process of gene expression in the cell: the first level of encoded RBNs, on which the search algorithm acts directly, is analogous to the gene; the second level, analogous to the translated protein; the fitness, the action of that protein.

We performed 40 runs of this experiment for the following three conditions:

- Direct encoding (no RBN)
- Single-level neutrality (as ESSH; figure 5)
- Two-level neutrality (figure 6)

For fairness of comparison, for the one-level neutrality experiment all 336 mutations were tried compared with 336 random mutations for the two-level neutrality. For the direct encoding, all 20 mutations were tried. The results are shown in figure 7.

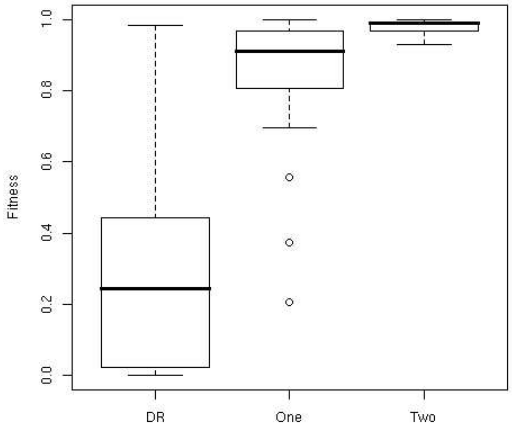

Fig. 7. Results for Experiment 1. The outer bars represent the range of values, the box represents one standard deviation from the mean, the bold line represents the mean, and small circles represent outliers.

VI. Experiment 2: Tunable Representations and H-IFF

The second experiment is designed to allow the amount of neutrality in each of the two levels of an encoding. We apply this to solving Watson's *Hierarchical If-and-only-If* (H-IFF) problem using the fitness-adaptive walk algorithm described earlier.

Members of the population are binary strings of length $n_1 \times n_2 * \times n$ (n is a parameter, n_1, n_2 are described below. These are transformed into (shorter) binary strings (of length n) by two successive mappings, each of which breaks the string down into a list of substrings of a given length (call this n_1 for the first mapping, and n_2 for the second), then maps these substrings to bits using a (randomly generated) mapping (as illustrated in figure 8). The composite mapping therefore maps strings of length $n_1 \times n_2 \times n$ into strings of length n, and therefore there is a many-to-one mapping from the search space to the space of strings that are used as input to the fitness function.

The fitness function used in this experiment is the Hierarchical If-and-only-If (H-IFF) function by Watson and Pollack [14]. This is designed to be a function that is easy for a recombination-based search algorithm to explore, whilst also being difficult for mutation-based algorithms to explore (this is a refinement of the earlier notion of a *royal-road function* [15]). The main reason that this is used as an example in these experiments is because it provides a challenging problem for a mutation-based algorithm such as the fitness-adaptive walk.

For a binary string B of length 2^k for some k, the fitness

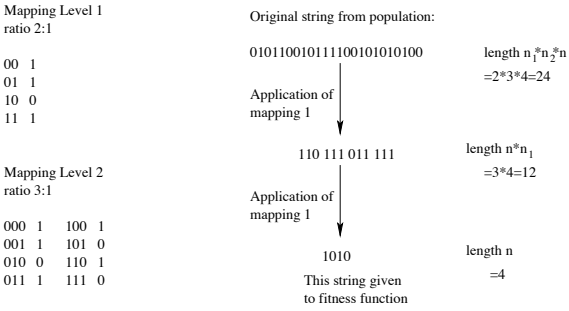

Fig. 8. An example of the kind of mapping used in Experiment 2. In this example, $n_1 = 2$, $n_2 = 3$ and $n = 4$.

of that string is defined as [14]:

$$f(B) = \begin{cases} 1 & \text{if } |B| = 1; \\ |B| + f(B_1) + f(B_2) & \text{if } (|B| > 1 \text{ and} \\ & \quad (\forall i, b_i = 0 \\ & \quad \text{or } \forall i, b_i = 1)); \\ f(B_1) + f(B_2) & \text{otherwise} \end{cases}$$

Where b_i represent the bits in B, and B_1 and B_2 represent the right and left hand halves of the string B.

This function has a maximum when all entries in the string are 0 or all are 1. Essentially, it is trying to reward "blocks" of zeroes or ones.

This experiment is summarized in figure 9. The output string from the first mapping is used as the input string to the second (i.e. the functions are composed). In biological terms we are using the phenotype of the first mapping as the genotype for the second; alternatively, we can regard this as being similar to the two levels of DNA→protein translation, and the protein→function activity.

More formally, here is the experiment in pseudocode. The search space S is all binary strings of length $n_1 \times n_2 \times n$, and h represents the H-IFF function.

Input: parameters m, n, n_1 and n_2
Randomly generate mappings f_1 and f_2
Generate initial random point p in S
LOOP (for fixed number of rounds):
  Make m mutations of p,
    call these $p_{1,1}, \ldots, p_{m,1}$.
  Translate all the $p_{1,i}$s with mapping 1:
    $f_1(p_{1,i}) \rightarrow p_{2,i}$, for $i = 0, \ldots, m$
  Translate all the $p_{2,i}$s with mapping 2:
    $f_2(p_{2,i}) \rightarrow p_{3,i}$ for $i = 0, \ldots, m$
  calculate $h(p_{3,1})$ for $i = 0, \ldots, m$
  t = fitness of fittest $p_{3,i}$ for $i = 0, \ldots, m$
  if t represents a fitness decrease, exit loop
  if t represents a fitness increase,
    p becomes equal to the corresponding $p_{3,i}$
  if t represents no fitness change, then
    p becomes a randomly chosen neutral $p_{3,i}$
END LOOP
Output: fitness of p

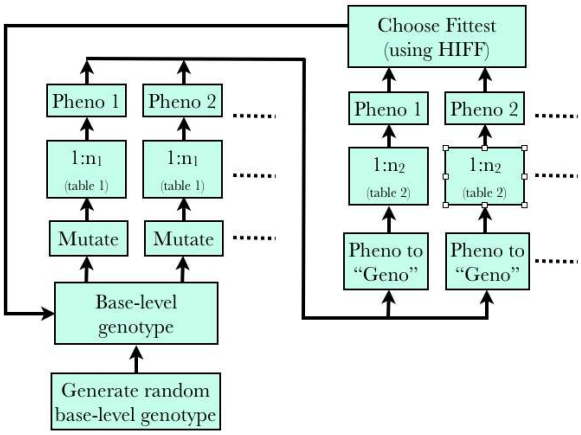

Fig. 9. Experiment 2.

Second Level

First Level	1	2	3	4	5	6
1	48.96	58.84	56.32	50.32	48.8	45.12
2	60.64	63.68	58.24	57.12	51.76	52.32
3	53.68	60.24	54.96	50.72	51.6	46.08
4	50.32	59.28	51.68	49.52	45.12	43.76
5	47.2	58.32	51.76	47.44	44.16	45.12
6	47.44	59.2	51.44	48	47.36	45.92

Fig. 10. Results for Experiment 2.

The results are shown in figure 10. Each entry in the matrix represents the mean fitness over 100 runs of the experiment. Settings of the parameters n_1 and n_2 are varied from 1–6. The closeness of the fitness to the maximum (64) is shown by the shading (lighter representing higher fitness).

VII. Discussion

In both of these experiments, multiple levels of neutrality are shown to perform better than having a single level of

neutrality. In the first experiment, the amount of neutrality (i.e. the ratio of the many-to-one mapping) in the function is fixed, therefore it could be argued that the multiple levels of neutrality are not important, and that the results obtained are simply a result of increasing the overall level of neutrality. This argument is shown not to hold in the second experiment, for example the best results are obtained by two levels of (1:2, 1:2) neutrality ratios, whilst the (1:4, 1:1) and (1:1, 1:4) neutral mappings perform considerably worse.

A tentative explanation for the superior performance of the two-level is that it expands the number of "directions" in which neutral ridges can be explored. Even if the search gets stuck in a local optimum with respect to one of the neutral levels, there is a second level of neutrality which can be exploited to escape from that region.

In experiment 2 it is interesting that for high levels of redundancy, the performance degrades. This suggests that at these levels, there is too much randomness in the mapping, and, instead of exploring useful neutral ridges, the algorithm is often moving to an effectively random part of the search space.

VIII. Conclusions and Future Work

We have demonstrated in two experiments that two levels of neutrality in a representation can lead to an enhanced ability to explore a search space. One area for future work will be to explore how widely this idea can be applied, investigating the conditions under which this holds. A related piece of work would be to examine how mutations explore the search space when differing amounts of neutrality are used.

Another direction will be to explore multiple levels of neutrality using other search techniques, for example the effect of multi-level neutrality on crossover and its use for representations in particle swarm optimization [16].

References

[1] R. Shipman, M. Shackleton, and I. Harvey, "The use of neutral genotype-phenotype mappings for improved evolutionary search," *BT Technology Journal*, vol. 18, no. 4, pp. 103–111, October 2000.

[2] M. Kimura, *The Neutral Theory of Molecular Evolution*. Cambridge University Press, 1983.

[3] P. Schuster, W. Fontana, P. Stadler, and I. Hofacker, "From sequences to shapes and back: A case study in rna secondary structures," *Proceedings of the Royal Society B*, vol. 255, pp. 279–284, 1994.

[4] C. Reidys, C. Forst, and P. Schuster, "Replication and mutation on neutral networks," *Bulletin of Mathematical Biology*, vol. 63, no. 1, pp. 57–94, January 2001.

[5] M. Kimura, "Evolutionary rate at the molecular level," *Nature*, vol. 217, pp. 624–626, 1968.

[6] M. Nei, "Selection and neutralism in molecular evolution," *Molecular Biology and Evolution*, vol. 22, no. 12, pp. 2318–2342, 2005.

[7] P. Schuster, "Extended molecular evolutionary biology," *Artificial Life*, vol. 1, pp. 39–60, 1994.

[8] M. Huynen, "Exploring phenotype space through neutral evolution," *Journal of Molecular Evolution*, vol. 43, pp. 165–169, 1996.

[9] M. Huynen, P. Stadler, and W. Fontana, "Smoothness within ruggedness: The role of neutrality in evolution," *Proceedings of the National Academy of Sciences*, vol. 93, pp. 397–401, 1996.

[10] I. Harvey and A. Thompson, "Through the labyrinth evolution finds a way: A silicon ridge," in *Evolvable Systems: From Biology to Hardware*, T. Higuchi, M. Iwata, and L. Weixin, Eds. Springer-Verlag, 1996.

[11] M. Ebner, M. Shackleton, and R. Shipman, "How neutral networks influence evolvability," *Complexity*, vol. 7, no. 2, pp. 19–33, Nov-Dec 2001.

[12] J. Esparza, C. Schröter, and S. Schwoon, "The model-checking kit (computer software)," http://wwwbrauer.informatik.tu-muenchen.de/gruppen/theorie/KIT/.

[13] S. Kauffman, *The Origins of Order*. Oxford University Press, 1993.

[14] R. Watson and J. Pollack, "Hierarchically-consistent test problems for genetic algorithms," in *Proceedings of the 1999 Congress on Evolutionary Computation*. IEEE Press, 1999, pp. 1406–1413.

[15] M. Mitchell, S. Forrest, and J. Holland, "The royal road for genetic algorithms : Fitness landscapes and GA performance," in *Towards a Practice of Autonomous Systems : Proceedings of the First European Conference on Artificial Life*, F. Varela and P. Bourgine, Eds. MIT Press, 1992.

[16] J. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm Intelligence*. Morgan Kaufmann, 2001.