Unconstrain the Population: The Benefits of Horizontal Gene Transfer in Genetic Algorithms

Nicholas Tomko, Inman Harvey and Andrew Philippides

Abstract Two of the main challenges of using standard genetic algorithms (GAs) to solve complex tasks are: 1) evolution can take a long time and 2) there is the risk that evolution gets stuck on a sub optimal solution. For these reasons, standard GAs may not be appropriate for evolving sufficiently complex SmartData agents that can act as our on line privacy and security surrogates. In this paper we present the Unconstrained Genetic Algorithm (UGA) which is a novel GA that takes its inspiration from bacterial evolution and show how it outperforms a standard GA on a benchmark task with a large number of sub-optimal solutions. Based on these results we believe that the UGA warrants further testing on complex tasks such as the evolution of SmartData agents.

1 Introduction

Genetic algorithms (GAs) are a search/optimisation technique based on Darwinian evolution (Holland, 1975). In a standard GA, a population of individuals are genetically encoded and evaluated on a specific task. Each individual genotype is assigned a fitness based on how well it solves the task and the fitter individuals survive to pass on their genes to the next generation while the less fit die off. As in natural evolution, the new population of offspring is generated using recombination and random mutation. Over time, the population guided by fitness based selection, mutation and recombination, moves around the solution space looking for better solutions. A fitness landscape is one way to visualize the solution space, where neighbouring

---

Nicholas Tomko
University of Sussex, Brighton, UK, e-mail: nt79@sussex.ac.uk

Inman Harvey
University of Sussex, Brighton, UK e-mail: inmanh@gmaill.com

Andrew Philippides
University of Sussex, Brighton, UK e-mail: andrewop@sussex.ac.uk

points on the landscape represent neighbouring (similar) genotypes and the heights of these points represents their fitness. These fitness landscapes can be though of multi-dimensional mountain ranges where the peaks of the mountains correspond to high fitness individuals and the valleys to low fitness individuals. The highest peak on the landscape is known as the global optimum or "best solution" and the lower peaks are known as local optima or sub-optimal solutions. To apply GAs to the evolution of SmartData agents one needs to determine how these agents will be genetically encoded and how their fitness will be evaluated; in other worlds, what differentiates a good SmartData agent from a bad one? Once these choices are made, a population of agents can be evolved, with the hope that over time the agents will become fitter and fitter.

A common problem with using genetic algorithms (GAs) to solve complex tasks, is their tendency to get stuck on sub-optimal solutions. Increasing the mutation rate and/or population size increases evolutionary exploration, reducing the chances of getting stuck on a local optimum, but does so at the expense of the potential exploitation of good solutions. To overcome this issue, different diversity maintenance methods have been developed and applied to GAs (see Dick (2005) or Mahfoud (1995) for an overview). In this paper we present the Unconstrained GA (UGA) which maintains diversity in a novel way using horizontal gene transfer (HGT) and show how it outperforms a standard GA (SGA) on rugged fitness landscapes.

HGT occurs mainly in microbial communities such as bacterial colonies and is the process by which genes are transferred between individuals. There are many different ways bacteria transfer genes horizontally but some of the most common include the use of plasmids or bacteriophages. In nature, vertical gene transfer (VGT) is the process of sexual reproduction where the genes of two parents are recombined to produce one or more offspring. This is how most multi-cellular organisms such as humans and other mammals reproduce. In most standard GAs, gene transfer is confined to the vertical type, where parents are genetically recombined to produce one or more offspring. We show that the reason the UGA outperforms a standard GA (SGA) on rugged NK landscapes (Kauffman, 1993) is because the process of HGT shuffles the genes of the fittest (elite) individuals through the population.

The goal of SmartData is to evolve agents that can act as our on line privacy and security avatars. Because this is an extremely complex task the fitness landscape will likely be highly multi-dimensional and rugged containing a large number of sub-optimal solutions. As discussed above, standard GAs can struggle on these types of landscapes and therefore to successfully evolve SmartData agents alternative genetic algorithms, such as the UGA, may need to be explored.

2 Background

Related areas of research include: 1) GAs that implement some sort of multi-individual recombination, 2) GAs inspired by bacterial evolution and 3) diversity maintaining GAs.

2.1 Multi-Individual Recombination

Multi-Parent Recombination (Eiben and Schippers, 1996), Bit Based Simulated Crossover (BSC) (Syswerda, 1993), Probability Based Simulated Learning (PBIL) (Baluja and Davies, 1998) and Gene Pool Recombination (GPR) (Muhlenbein and Voigt, 1995) are examples of GAs that apply recombination to more than two individuals at a time.

In Eiben and Schippers (1996) a variety of multi-parent recombination schemes were tested using anywhere from 2 to 16 parents on different NK landscapes to try to understand how varying the amount of sexual recombination impacts evolutionary performance. In GPR (Muhlenbein and Voigt, 1995), new offspring are generated by recombining two randomly chosen parent alleles at each locus from a population that has already undergone evaluation and selecction. We discuss the similarities and differences between this algorithm and the UGA in the next section. BSC (Syswerda, 1993) is similar to GPR but is restricted to binary genotypes. In this algorithm, offspring are generated using population-level probability distributions. These distributions are constructed by counting the number of 1's and 0's at each locus position and then weighting this distribution by each individual's fitness. PBIL (Baluja and Davies, 1998) is almost identical to BSC but instead of having a population of individuals there is a probability vector that represents the population. This vector is used to generate new individuals which are evaluated and then used to update the vector.

2.2 GAs Inspired by Bacterial Evolution

One of the first GAs inspired by bacterial evolution was the Microbial GA (Harvey, 1996, 2001, 2011). This is a steady-state, tournament based GA that implements horizontal microbial gene flow rather than the more standard, vertical gene transfer from generation to generation.

The Pseudo-Bacterial GA (PBGA) (Nawa et al., 1997) and the Bacterial Evolutionary Algorithm (Nawa and Furuhashi, 1998) are two GAs that use a genetic operator which they call the 'Bacterial Operator'. This operator attempts to mimic gene transduction which is one process by which bacteria can horizontally transmit parts of their genome to other bacteria. The goal of implementing gene transduction in a GA is to try to speed up the spread of high fitness genes through the population.

Symbiotic GAs such SANE (Moriarty and Miikkulainen, 1995, 1996), the Binomics GA (Harvey and Tomko, 2010) and simulated ecosystem evolution (Williams and Lenton, 2007) evolve groups of entities working together to solve a problem instead of using the more standard method of evolving individuals. These algorithms take inspiration from the evolution of microbial ecosystems where individuals have to work together to survive.

2.3 Diversity Maintaining GAs

Here we briefly describe various diversity maintaining methods used in GAs. One goal of these algorithms is to maintain diversity in the population so that evolution doesn't get stuck on a local optimum. For a more in depth summary see Dick (2005) and Mahfoud (1995).

Fitness sharing (Goldberg and Richardson, 1987) is a niching method that relies on some distance metric or similarity measure between individuals. By using suitable methods to adjust the fitness of any individual according to how many other similar individuals are within some predetermined niche (similarity) radius, there is a tendency for the population to spread out over multiple solutions or niches in the fitness landscape; thus diversity is maintained. Clearing (Petrowski, 1996) is very similar to fitness sharing but, instead of degrading the fitness of individuals within the same similarity radius or subpopulation, it removes the least-fit individuals within the similarity radius from the population.

Crowding was first introduced in De Jong (1975) as a method of removing similar individuals from a population, with the goal of trying to maintain diversity during evolution. Deterministic Crowding (Mahfoud, 1995) is a specific type of crowding that mates two individuals in the population and then if the offspring is fitter, replaces the parent that is most similar to the offspring. It is similar to fitness sharing because it requires a similarity calculation done between individuals, but unlike fitness sharing there is no requirement to pre-specify a similarity radius.

Spatially structured GAs are another way to enforce diversity in a population, for a good review see Dick (2005) and Tomassini (2005). In these, the population is structured within some local geographical distribution (demes) that constrains which members of the population are allowed to be selected or recombined with one another. This deme structure allows more genetic diversity to be maintained across sub-populations.

3 The Unconstrained GA

The difference between the UGA and more standard GAs is that the UGA includes both horizontal (HGT) and vertical gene transfer (VGT) while most standard GAs only apply VGT. In GAs, HGT can be simulated in a variety of different ways. For example, one could implement HGT as a transfer of genes where an individual in the population transfers one or more genes to another individual and in the process overwrites the recipient's gene(s). In this case, there would be an increase in the frequency of the genes transferred and a decrease in frequency of the genes that were overwritten. An alternative way to implement HGT, would be as a gene swap, rather than a gene transfer. In this case, two individuals swap or trade genes so there is no change of gene frequency in the population. In the UGA we have chosen to implement HGT as gene swapping instead of gene transferring to ensure that there is no change in gene frequency in the population as a result of this process.

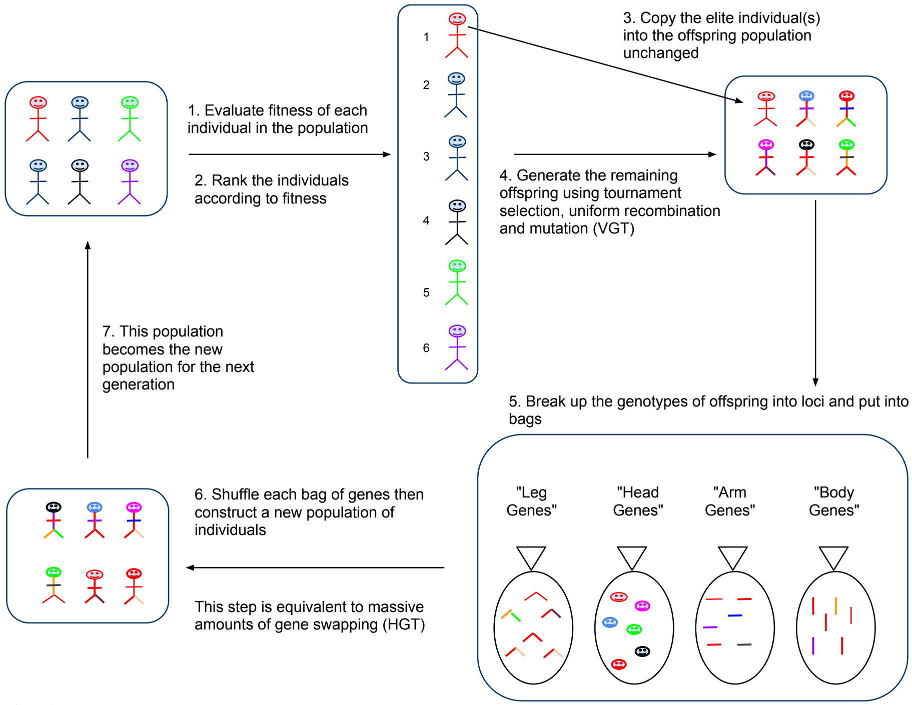

Fig. 1 A general flow diagram showing the main steps of the UGA.

To efficiently implement massive amounts of gene swapping, every generation the genes of at each locus are shuffled. Another way to think about this is process is that all the genes of a specific locus are put into bags and then randomly withdrawn to reconstruct the full genotypes in the population (see Figure 1).

A single generation of the UGA is carried out as follows. VGT occurs in step 4 where two parents are uniformly recombined to produce a single offspring and horizontal gene shuffling is done in step 6. In a SGA elitism conserves the fittest genotypes in the population by automatically copied the fittest individuals into the offspring population without mutating them. In the UGA elitism is carried out in step 3, but because of the gene shuffling of step 6, the fittest genotypes are not conserved but rather the genes of the fittest individuals are conserved. If steps 5 and 6 are skipped then the UGA becomes a standard GA (SGA) with only VGT.

1. Evaluate the fitness of each individual genotype in the population
2. Rank each genotype according to fitness
3. The fittest ELITENUM individuals are automatically copied into the new offspring population unchanged
4. The remaining POPSIZE-ELITENUM offpsring are generated as follows:

   a. Select two parents from the population using the following tournament selection method
      i. Randomly pick two individual genotypes from the population
      ii. Compare the fitness of the two individuals from the population with the fitter becoming a parent
      iii. If the fitness of the two individuals is the same then randomly pick a winner
      iv. repeat steps i-iii to select a second parent
   b. Produce a single offspring from these two parents by randomly choosing a single gene from either parent at each locus (this is known as uniform recombination)
   c. Mutate this offspring at each locus with a probability of MUT (the mutation rate)
   d. Add this offspring to the new offspring popualtion

5. Break up the genotypes of each offspring and put them into separate locus bags
6. Construct a new population be randomly choosing genes from each bag (this is equivalent to the population engaging in promiscuous HGT)
7. This new population becomes the population for the next generation

The reason for presenting the UGA as above is so that it can easily be seen that when steps 5 and 6 are eliminated, the UGA becomes a SGA with tournament selection and uniform recombination. Also, in this form, VGT (step 4) and the HGT (step 6) can be easily distinguished. Alternatively, steps 3-6 can be simplified as follows:

3. Put the genes of the elite genotypes directly into the locus bags
4. Choose two parents from the population using tournament selection
5. At each locus, randomly choose a gene from each parent, mutate with probability MUT and then put into the appropriate locus bag
6. Shuffle each bag of genes and reconstruct the population

This accomplishes the exact same thing but in fewer steps.

The UGA Compared to Gene Pool Recombination

Out of all the algorithms discussed in the Literature Review section, the UGA is most similar to the GPR algorithm (Muhlenbein and Voigt, 1995). In GPR the parents are constructed by randomly choosing genes from the population with replacement. This process can be thought of as a type of HGT, so is similar to the shuffling step in the UGA. There are a couple of important differences between the HGT in GPR and the UGA, the first is that in GPR, gene shuffling is done right after evaluation and selection have been applied to the population while in the UGA the shuffling is done on the offspring population before the new population is evaluated. A second difference is that the in the UGA shuffling does not change the genetic make-up of the population, but in GPR because genes are selected from the population with replacement, the genetic make-up of the population will likely change as a result. An interesting research question that is not part of the scope of this paper is to determine what impacts these differences have on evolution.

4 The NK and NKp Fitness Landscapes

We chose to compare the performance of the UGA to a SGA on NK and NKp landscapes. NK landscapes (Kauffman and Johnsen, 1991; Kauffman, 1993) are tuneable, rugged, binary landscapes where N defines the dimensionality or number of bits (genes) in the landscape and K defines how rugged the fitness landscape is, where more rugged landscapes have more local optima. In both NK and NKp landscapes the amount of rugedness (K) is controlled by changing the amount of genetic epistasis, where epistasis can be generally defined as the degree of interaction between genes. The higher K, the more genes there are that have an impact on any individual gene's fitness contribution (Kauffman and Johnsen, 1991). Along with increasing the number of fitness peaks in the landscape, increasing K also increases the steepness of these peaks and reduces their height Kauffman and Johnsen (1991). If K is set to zero then there is no epistatic interaction between genes and the fitness landscape can be classified as a 'Mt. Fuji' landscape which has a single global fitness optimum and no local optimum. If K is set to 2 then the fitness contribution of any given gene depends on the fitness of its two direct neighbors - this epistatic interaction increases the ruggedness of the landscape by adding local optima to it. The maximum value K can be set to is N-1, in this case the landscape is completely uncorrelated and the fitness of the genotypes are completely random Kauffman and Johnsen (1991).

NKp landscapes (Barnett, 1998) introduce neutrality into NK landscapes. One way to picture a neutral landscape is in terms of plateaus connecting what would be local optima in a non-neutral landscape; there are good reasons to expect many fitness landscapes in both natural (biological) and artificial scenarios to have such neutrality. In NKp landscapes, the amount of neutrality is controlled by parameter p which can be set between 0 and 1, where a higher p corresponds to more neutrality. If p is set to zero then the NKp landscape is equal to a normal NK landscape and if p is set to 1 then the landscape is completely flat with no optima Barnett (1998).

A general procedure for generating an NK landscape as summarized in Mayley (1996) is as follows:

1. Generate N look-up tables, one for each locus
2. Each look-up table has $2^{(K+1)}$ entires that are randomly generated between [0, 1]
3. The fitness of a give locus, $f(n)$ is found by taking the specific locus and K neighbors and finding the corresponding entry in the nth look up table.
4. The total fitness is equal to the average of all the loci fitnesses:

$$F(N,K) = \frac{1}{N}\sum f(n)$$

For example, if N=4 and K=2 then there will be four look-up tables each containing 8 entries. To find the fitness of the second locus in genotype [1 0 1 1] then one looks for the [1 0 1] entry of the second look-up table.

To generate an NKp landscape from an NK landscape a proportion of the entries in the look-up table, defined by p, are set to zero. So if p = 0.90 then a randomly chosen 90% of the entries in the look-up tables are set to zero.

We chose the NK and NKp landscapes as a test-bed for the UGA because the dimensionality (N), ruggedness (K), and neutrality (p) can be easily varied which allows us to test the UGA on a wide variety of landscapes in a reasonable amount of time. NK landscapes are also a well-known benchmark problem in the GA community and have been shown to be a good task to test the behaviour and performance of GAs (Aguirre and Tanaka, 2003).

5 Results

Here we compare the performance of the UGA to an SGA on a variety of NK and NKp landscapes. The SGA we use for comparison is exactly like the UGA described earlier except that steps 5 and 6 are eliminated. For each family of landscapes, where a family refers to landscapes with the same N, K, and p parameters, performance was measured as the median fitness reached on 50 different randomly generated landscapes of a given family over a fixed number of evaluations. The results presented in this section are the best performing parameter combinations of each each GA on the different landscapes. The population size was varied between 10 and 1000 and the mutation rate was varied between $0.1/N$ and $2/N$ where this mutation rate corresponds to the probability of flipping a bit at a given locus. To determine the effects of elitism we tested both 5% elitism, where the fittest 5% (rounded up to the nearest whole number) of the population were copied into the offspring population and 0% elitism where none of the fit individuals were copied through unchanged. For example, if the population size was 5 or 10 and elitism was set to 5% then the single fittest individual would be copied into the offspring population unchanged. Evolution was run for 20,000 generations and the maximum fitness reached during evolution was recorded. Statistical comparisons were made using the Wilcox rank sum test of equal medians at a 5% significance level.

The GAs were tested on (N,K) (30,0), (30,4), (30,6), (30,8) landscapes and (N,K,p) (30,4,0.99), (30,6,0.99), (30,8,0.99), (100,8,0.99), (100,8,0.90) landscapes.

Table 1 compares the performance of the UGA and the SGA, highlighting which GA performed better over 50 landscapes. As this table shows the UGA performed significantly better than the SGA on very rugged landscapes ($K >= 6$) but on landscapes with lower K's there was no significant difference in performance.

Table 2 shows that on the NKp landscapes, the UGA only outperformed the SGA on the (100,8,0.90) landscape and was no significant difference between the SGA and UGA on the other NKp landscapes.

Table 1 A comparison of the UGA and SGA on a variety of different NK landscapes, showing which GA performed significantly better based on the Wilcox rank sum test for equal medians at a 5% significance level.

Landscape	UGABetter	SGA Better	No Difference
(30,0)			X
(30,4)			X
(30,6)	X		
(30,8)	X		

Table 2 A comparison of the UGA and SGA on a variety of different NKp landscapes, showing which GA performed significantly better based on the Wilcox rank sum test for equal medians at a 5% significance level

Landscape	UGABetter	SGA Better	No Difference
(30,4,0.99)			X
(30,6,0.99)			X
(30,8,0.99)			X
(100,8,0.99)			X
(100,8,0.90)	X		

6 Why does the UGA perform well on rugged landscapes?

Our hypothesis is that the UGA outperforms the SGA on rugged, non-neutral landscapes because the gene shuffling step reduces the chance of the population getting stuck on a sub-optimal fitness peak. To test this hypothesis we varied the number of individuals whose genes were shuffled each generation. If our hypothesis is correct then the performance of the UGA should decrease as the amount of gene shuffling is reduced. We tested a modified version of the UGA on the (30,8) landscapes where we shuffled 50% and 20% of the indivduals' genes and compared the results to the original UGA where 100% of the population's genes are shuffled. We found that performance was fairly constant as the amount of gene shuffling was reduced and there was no significant difference between the original UGA and the modified UGA where only 20% of the population's genes were shuffled. Interestingly, our results are similar to those in Eiben and Schippers (1996), who found that even though multi-parent uniform recombination improved evolutionary performance on NK landscapes, increasing the number of parents did not further increase performance. The conclusion of Eiben and Schippers (1996) was that the benefit of multi-parent recombination is a result of more gene mixing and a bigger sample size to base decisions on, but they could not explain why increasing the number of parents didn't increase performance.

Upon further investigation we found that the key to the performance of the UGA was not how much gene shuffling occurred but instead was dependent on whether or not the genes of the elite individual(s) in the population were shuffled. In the UGA and SGA, every generation the fittest 5% of the population avoided being mutated and were automatically put in the new population. Adding elitism to GAs can be a doubled edged sword. On one hand, elitism has the potential to improve performance because it preserves the fittest individuals in the population, but on the other hand it has the potential to reduce performance because it can increase the chance that the population gets stuck on a sub optimal fitness peak. Our results imply that by shuffling the genes of the elite individuals through the population, the benefits of elitism are retained without increasing the chances of getting stuck on a local optimum. The details of the GA used in Eiben and Schippers (1996) were not included in their paper so we do not know whether or not elitism was used, but if they did use an elitist GA then our analysis could potentially explain their results as well.

7 Discussion

The goal of the SmartData project is to evolve agents to act as our on line privacy and security avatars. Even though this project is in its infancy, one of the challenges identified at the IPSI SmartData Symposium were the potential limitations of evolutionary methods including fact that evolution can take a long time and that evolutionary algorithms can get stuck on sub optimal solutions. Due to the complexity of the task it is safe to assume that the fitness landscapes of the SmartData agents will be highly multi-dimensional and extremely rugged. For these reasons, standard GAs may not be able evolve sufficiently fit SmartData agents in a reasonable amount of time and so it makes sense explore different GAs, such as the UGA.

We have shown that the UGA outperforms a standard GA on rugged ($K >= 6$) NK landscapes but when K is reduced or neutrality is added there is no significant difference in performance. The original hypothesis was that the performance boost was a result of gene shuffling reducing the possibility of the population getting stuck on a local optimum but as explained in the previous section we discovered that the performance increase was due to the genes of the elite individuals getting shuffled through the population. More work needs to be done to determine how general this result is and whether the performance of the UGA is solely due to elite gene shuffling on all tasks. Results in Baluja and Davies (1998); Muhlenbein and Voigt (1995); Eiben and Schippers (1996) show that GAs based on multiple individual recombination perform well on a wide variety of tasks and therefore an interesting area of research would be to further investigate these types of algorithms and try to understand when they work, when they don't work and why.

References

Aguirre, H. and Tanaka, K. (2003). A study on the behavior of genetic algorithms on NK-landscapes: Effects of selection, drift, mutation, and recombination. *IEICE*

Trans. Fundamentals, (9):2270–2279.

Baluja, S. and Davies, S. (1998). Pool-wise crossover in genetic algorithms: An information-theoretic view. In *In Foundations of Genetic Algorithms V*.

Barnett, L. (1998). Ruggedness and neutrality: The NKp family of fitness landscapes. In Adami, C., Belew, R., Kitano, K., and Taylor, C., editors, *Artificial Life VI: Proceedings of the Sixth International Conference on Artificial Life*, number Kimura 1983, pages 18–27. The MIT Press.

De Jong, K. (1975). *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, Ann Arbor.

Dick, G. (2005). A comparison of localised and global niching methods. In *17th Annual Colloquium of the Spatial Information Research Centre (SIRC 2005: A Spatio-temporal Workshop)*, pages 91–101, Dunedin, New Zealand.

Eiben, A. and Schippers, C. (1996). Multi-parent's niche: n-ary crossovers on NK-landscapes. In *Proceedings of the 4th Conference on Parallel Problem Solving from nature*, pages 319–328. Springer.

Goldberg, D. and Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimisation. In Grefenstette, J., editor, *Proc. of the Second International Conference on Genetic Algorithms*, pages 41–49, Hillsdale, NJ. Lawrence Erlbaum Associates.

Harvey, I. (1996). The Microbial Genetic Algorithm.

Harvey, I. (2001). Artificial Evolution : A Continuing SAGA. In Gomi, T., editor, *Evolutionary Robotics: From Intelligent Robots to Artificial Life, Proc of 8th International Symposium on Evolutionary Robotics*, pages 1–19. Springer-Verlag.

Harvey, I. (2011). The Microbial Genetic Algorithm. In Kampis, G., Karsai, E., and Szathmary, E., editors, *ECAL 2009, Part II. LNCS 5778*, pages 126–133, Heidelberg. Springer.

Harvey, I. and Tomko, N. (2010). Binomics : Where Metagenomics meets the Binary World. In Fellermann, H., Dorr, M., Hanczyc, M., Laursen, L., Maurere, S., Merkle, D., Monnard, P., Stoy, K., and Rasmussen, S., editors, *Proceedings of Artificial Life XII, 12th Intl. Conf. on the Synthesis and Simulation of Living Systems*, pages 370–377, Odense, Denmark. MIT Press.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*, volume Ann Arbor. University of Michigan Press.

Kauffman, S. (1993). *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press.

Kauffman, S. and Johnsen, S. (1991). Coevolution to the edge of chaos: coupled fitness landscapes, poised states, and coevolutionary avalanches. *Journal of theoretical biology*, 149(4):467–505.

Mahfoud, S. W. (1995). *Niching Methods for Genetic Algorithms*. PhD thesis, University of Illinois at Urbana-Champaign.

Mayley, G. (1996). Landscapes, learning costs, and genetic assimilation. *Evolutionary Computation*, pages 1–21.

Moriarty, D. and Miikkulainen, R. (1995). Learning Sequential Decision Tasks. In Honavar, V., Patel, M., and Balakrishnan, K., editors, *Advances in the Evolutionary Synthesis of Neural Systems*, Cambridge, MA. MIT Press.

Moriarty, D. E. and Miikkulainen, R. (1996). Efficient reinforcement learning through symbiotic evolution. *Machine Learning*, 22:11–32.

Muhlenbein, H. and Voigt, H.-M. (1995). Gene pool recombination in genetic algorithms. *Meta-heuristics: Theory and applications*.

Nawa, N. and Furuhashi, T. (1998). Bacterial evolutionary algorithm for fuzzy system design. In *IEEE Conference on Systems, Man, and Cybernetics*, pages 2424–2429.

Nawa, N., Hashiyama, T., Furuhashi, T., and Uchikawa, Y. (1997). A study on fuzzy rules discovery using Pseudo-Bacterial Genetic Algorithm with adaptive operator. *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC '97)*, pages 589–593.

Petrowski, A. (1996). A clearing procedure as a niching method for genetic algorithms. *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 798–803.

Syswerda, G. (1993). Simulated Crossover in Genetic Algorithms. In *Foundations of Genetic Algorithms*, pages 239–255. Morgan Kaufmann, San Mateo, 2 edition.

Tomassini, M. (2005). *Spatially Structured Evolutionary Algorithms: Artificial Evolution in Space and Time*. Springer Verlag, Berlin.

Williams, H. and Lenton, T. (2007). Artificial ecosystem selection for evolutionary optimisation. In Almeida E Costa, F. E. A., editor, *Advances in Artificial Life: Proceedings of the 9th European Conference on Artificial Life*, pages 93–102, Berlin. Springer Verlag.