Adaptive Approaches Towards Better GA Performance in Dynamic Fitness Landscapes.

Henrik Hautop Lund

Institute of Psychology
National Research Council, Viale Marx 15, 00137 Rome, Italy
tel.: (+39) 6 88 94 596
fax : (+39) 6 82 47 37

DAIMI
University of Aarhus, Ny Munkegade, 8000 Aarhus C., Denmark
tel.: (+45) 89 42 32 21
fax : (+45) 89 42 32 55

e-mail: hhl@daimi.aau.dk

Abstract

We review different techniques for improving GA performance. By analysing the fitness landscape a correlation measure between parents and offspring can be provided, and we can estimate effectively which genetic operator to use in the GA for a given fitness landscape. The response to selection equation further tells us how well the GA will do, and combined the two approaches gives us a powerful tool to automatic ensure the selection of the right parameter settings for a given problem. In dynamic environments the fitness landscape change over time, and the evolved systems should be able to adapt to such changes. By introducing evolvable mutation rates and evolvable fitness formulae we obtain such systems. The systems are shown to be able to adapt to both internal and external constraints and changes.

1 Introduction.

In the 1970s Holland [7] introduced genetic algorithms (GAs) as a means to design and implement robust adaptive systems. Holland emphasized that these systems should be able to handle uncertain and changing environments, and that the systems via feedback from the environments in which they operate should be able to self-adapt over time. Yet, most work with GAs has been done in time-invariant environments and not in dynamic environments, as was Hollands original idea. This originates partly from the early insight that when implementing GAs the parameter choices, such as population size, internal genetic representation, and genetic operators are very significant for the performance of the GAs, even in static environments. Therefore, the main interest and achievement of the first decades of research in GAs was emperical exploration of parameter settings (e.g. [1], [3]), and later the study of GAs use in (function) optimization and other applications in static environments has been highly appreciated (e.g. [20]). In this paper it is outlined how the reintroduction of Hollands original idea of studying GA evolved systems in uncertain

[0] *Technical Report DAIMI PB-487, Aarhus University, Aarhus.*

and changing environments gives us more insight into when and why GAs work.

In one environment a specific combination of one type of crossover and one type of mutation may be suitable, while in another environment another combination of crossover and mutation type may be preferable. In section 2 we introduce statistical features of the fitness landscape to tune GAs in different environments. These statistical tools gives us a means to choose for a specific environment (i.e. problem) which type of crossover and mutation to use. Having chosen the right genetic operators, we then turn to predict the performance of the GA in section 3, by outlining how the response to selection equation can be used on GAs. In uncertain and changing environments a system must be able to adapt to new situations, why all parameter settings, at least in theory, should be adaptive. In section 4 we discuss adaptive mutation rates. By further introducing the fitness formula in section 5, the evaluation function is let free to evolve to adapt to the environment, and through simulations we can show speciation and pre-adaptation in complex systems without strong assumptions (e.g. on number of species and/or the distribution of niches in the environment), as in crowding [3], sharing [5], and tagging [1] methods. Finally, in section 6 we conclude.

2 Fitness Landscape Analysis.

When we want to analyze when and why a GA works on a specific problem, we will have to know if we use the right parameter settings, e.g. population size, the chosen crossover and mutation operator and their rates. The first idea that comes to mind is a systematic testing of numerous parameter settings, or to let the GA itself find the optimal parameter values. Both approaches has been investigated extensively, e.g. [16],[19], but have the major drawback to be computational very expensive. An alternative and computational much less expensive approach is the fitness landscape approach, as suggested in [12][13].

A genotype can be represented as a point in a N-dimensional space, where N is the number of alleles in the genotype, e.g. connections in a neural network or bits in a bitrepresentation. The value of an allele determines the position of the genotype on the particular dimension representing the allele. In that way, such a space can represent all possible genotypes. The performance, i.e. the fitness, of a genotype on a particular task can be represented by an additional N+1th dimension. There will be a performance surface with N+1 dimensions that goes through all the genotype space and has higher and lower points. Each point in the genotype space (i.e. each possible genotype) will be located on this surface. If it is placed on a high point on the surface this means that its performance on the task is good; if it is on a low point, its performance is bad. We call this the fitness landscape.

It has been shown how simple statistical features derived from the fitness landscape like the autocorrelation function, the correlation length, and the fitness correlation coefficients of the different genetic operators can be used to tune different components of the GA [12]. Let us here concentrate on the correlation coefficient, ρ_{OP}, of a genetic operator, OP. The genetic operator could for instance be one of the RAR, inversion, and swap mutation operators or one of the order-based, position-based, partially mapped, and adjacency-based crossover operators that are often used in ordering GAs on problems such as the travelling salesman problem (TSP). We apply the operator OP to the parent(s) to generate offspring. For each application of the operator OP, F_p and F_c represent the fitness of the parent (or mean fitness of two parents) and the fitness of its child (or mean fitness of its two children). The operator OP is applied to a number of parents to get their offspring, and we calculate the correlation coefficient between the fitnesses F_p and F_c as follows:

$$\rho_{OP}(F_p, F_c) = \frac{Cov(F_p, F_c)}{\sigma(F_p)\sigma(F_c)}$$

where Cov(F_p,F_c) is the covariance between the values of F_p and F_c, and $\sigma(F_p)$ and $\sigma(F_c)$ are the standard deviations of the values F_p and F_c.

There is a relationship between an operators correlation coefficient and the correlation length of the underlying fitness landscape and it expresses how correlated the landscape appears to the operator. We can for example look at crossover. Here the distance between parents and children depends both on the distance between the two parents, and on the chosen crossover point. If the fitness landscape is rugged, i.e. neighboring points are weakly correlated and we do not have strong causality [18], then the distance between parents and children may be larger than the correlation length of that landscape and the fitnesses of the parents does not tell anything of the expected fitnesses of the children. Crossover can therefore not do anything more helpful than random search in to rugged fitness landscapes. Through extensive experimentations Oliver et al. [16], and Starkweather et al.[19] found that GA performance on TSP depends on the chosen genetic operators. With the fitness landscape approach, such a result and the definition of which operators to use can be obtained with much less computational efforts and even before doing the actual problem solving, as shown by Manderick and Spiessens [13]. On the TSP they found the correlation coefficients shown in table 1 and table 2 by taking the average of 30 different samples of 2*1024 parent genotypes.

Crossover operator	*Correlation coefficient ρ*
Adjacency-based	0.78
PMX	0.61
Position-based	0.37
Order-based	0.68

Table 1. Correlation coefficients of the different crossover operators on TSP.

Mutation operator	*Correlation coefficient ρ*
RAR	0.89
Inversion	0.93
Swap	0.84

Table 2. Correlation coefficients of the different mutation operators on TSP.

The performance of the GA with the operators specified in table 1 and table 2 on the TSP showed, that the GAs with operators with higher correlations coefficients led to better performances. For example, with the position-based crossover a tour of length 750 was found after 50 generations, while with the adjacency-based crossover with higher correlation coefficient a tour of length 450 was found after 50 generations. Likewise the GA with inversion mutation did better than the two other GAs. We conjecture that the fitness landscape approach is an important tool to do prelimenary determination of which type of GA to use for a given problem.

3 Response to Selection.

Supposed to have chosen the right GA for a given problem with the fitness landscape approach, we would know like to know if this actually will lead to finding an acceptable solution to the problem. What we want is a measure of the GA performance. To this end we will look at a specialized variant of the fitness landscape approach that can be used to construct what has been called a Breeder Genetic Algorithm [15]. The Breeder Genetic Algorithm models artificial selection as performed by human breeders. The response to selection equation and concept of heritability known by human breeders are of special interest, since they give us a means to predict the performance of the Breeder Genetic Algorithm. The response to selection, R, is defined as the difference between the population mean fitness of generation t+1 and the population mean fitness of generation t :

$$R(t) = M(t+1) - M(t)$$

The human breeder measures the selection with the selection differential, S, which is defined as the difference between the mean fitness of the selected parents and the mean fitness of the population:

$$S(t) = M(t, P_s) - M(t)$$

where P_s is the selected parents. The breeder tries to predict R(t+1) from S(t). The prediction of the response to selection is given by

$$R(t) = b_t * S(t)$$

where b_t is the realized heritability which the breeder e.g. can have estimated or measured in previous generations. The realized heritability is normally assumed constant for a certain number of generations, so we have

$$R(t) = b * S(t)$$

and the prediction of the cumulative response for s generations is therefore

$$R_s = \sum_{t=1}^{s} R(t) = \sum_{t=1}^{s} b * S(t)$$

To compute R_s we need to compute S(t) which is given by

$$S(t) = I * \sigma_p(t)$$

when the fitness values are normal distributed and truncation selection is applied. σ_p is the standard deviation and I is called the selection intensity. The relation between the truncation threshold T and the selection intensity I is shown in table 3, from which we see that large differences in truncation lead only to minor changes in selection intensity.

T	80%	50%	40%	20%	10%	1%
I	0.34	0.8	0.97	1.2	1.76	2.66

Table 3. Selection intensity as function of truncation threshold.

We now obtain the response to selection equation:

$$R(t) = b * I * \sigma_p(t)$$

The science of artificial selection consist of estimating b and $\sigma_p(t)$, which depend on the fitness function. Application of this technique can be found in [14][15]. Interesting one can find the number of generations to convergence based on genetic drift and the genetic operator used by the genetic algorithm.

There are different techniques for estimating heritability and regression. By calculating differences in mean fitness of successive parents and children we can estimate the ruggedness of the fitness landscape, as described in the previous section. This information can then be used to construct a genetic algorithm that chooses the appropriate genetic operators and perform as described in the response to selection equation.

4 Evolvable Mutation Rates.

In so far we have been looking at tools to predict and select the right operators for the GA for a specific problem in a static environment. Yet, in uncertain and changing environments the evolved system must be able to adapt to new situations. The parameter settings can therefore not be expected to be fixed once and for all for the whole evolution process, but should rather be let free to evolve by itself. The selection of genetic operators presented in section 2 could become evolvable in order to adapt to new environments and changed fitness landscapes. In this section we will present how another parameter setting, namely the mutation rate, could (and should) change itself through evolution.

We find wide evidence for the evolution of mutation rates in nature. In replication of DNA, differences between the original and its copy are detected and the errors are corrected. There can be various levels of proof- reading, leading to a control of mutation rates [4]. Thus mutation rates are not given, fixed numbers, but are themselves variable by mutation. A mutation rate in an immune system is enhanced to increase the diversity of antibodies, which face an unknown antigenetic site [21]. The high mutation rate may look natural as for the level of the whole immune system. However, each antibody itself is a replicating unit. Many antibodies compete with each other for replication, and Darwinian selection acts for each antibody.

In GAs the mutation rate is decided before the actual simulation and remains fixed for the whole evolution process. In the simulated annealing method, on the other hand, the temperature for the Monte Carlo simulation is slowly cooled down according to a schedule fixed in advance. The temperature here plays the same role as the mutation rate, since both give an error rate of replication of a bit string. The proposed evolution of mutation rates gives a spontaneous change of mutation rates, and thereby a spontaneous simulated annealing method.

Kaneko and Ikegami [8] have reported results with evolution of mutation rates. As could be expected the mutation rate ultimatively went down with time. Once a species with higher fitness was selected, species with lower mutation rates had more offspring, independent of the population distribution, since the population dynamics of one species did not directly depend on the population of the other species. This argument is true after the species finds a string with a high fitness in the landscape. Yet, in the transient time, the mutation rate can increase, if the initial ensemble does not include species of high fitness. Later, the mutation rate decreases towards zero. The transient increase of mutation rates occur since species with higher mutation rates are faster in the search for larger fitness. Once the species with larger fitness appears, it is then better to have a strategy to reproduce the species with a smaller error, and the mutation rate starts decreasing.

The above example shows that the inclusion of mutation of mutation rates gives an innovation in the GAs by an automatic simulated annealing. In the conventional simulated annealing [9], one has to change the temperature (”mutation rate”) externally. Initially the system should be put in a high temperature to search effectively for many local minima, and the temperature is lowered externally as the system finds a configuration with lower energy. In the simulated annealing, one has to give a schedule of temperature decrease in advance, which depends on the problem to optimize.

The algorithm with mutation of mutation rates automatic enhances a mutation rate initially and then lowers it as the system finds a lower energy. During the time steps with enhanced mutation rates, lower energy is globally searched for, while the mutation is decreased once a species of very low energy is found. Thus one can reach optimal sequences very effectively with this inclusion of mutation of mutation rates. We note that the algorithm does not require any external change of mutation (temperature) rates. Everything goes spontaneously, once an initial condition is given.

5 Evolvable Fitness Formula.

Not only the mutation rates should be let free to evolve, but also the individual fitness formula, which indirectly leads to an evolution of the evaluation function in the GA. The notion of a fitness formula arises from the Artificial Life field. In simulations of the evolution of populations of artificial organisms (neural networks) individual organisms generate offspring as a function of the degree that they satisfy a criterion of fitness, or a *fitness formula* [10]. For example, organisms which live in an environment that contains food elements may reproduce in proportion to the number of food elements they are able to capture during life. The fitness formula in this case is number of food elements captured. The fitness formula of organisms living in another environment may be number of food elements eaten of type A minus number of food elements eaten of type B.

In simulations using *ecological* neural networks [17], the fitness formula is important, not only as a criterion measure, that networks should maximize evolutionarily, but because it determines the type of behavior networks tend to exhibit in the environment. Lund and Parisi [10] have shown, that given a particular environment and a particular fitness formula, organisms evolve behaviors which are appropriate both to the environment *and* the fitness formula.

In almost all simulations using GAs with populations of neural networks the fitness formula is fixed and decided by the researcher. Yet, the fitness formula should be let free to evolve as any other trait of the organisms without any necessary control from the researcher, since a moment of reflection suggests, that in biological reality the fitness formula isn't fixed, but as a trait of organisms, is evolvable. More precisely, the fitness formula summarizes a number of properties of a particular species of organisms related to their nutritional needs, to the mechanisms and processes in their bodies which extract energy from ingested materials, etc. (For the relation between food and the evolution of our species, cf. [6]) Like all other traits of organisms, the fitness formula can evolve. In fact, we can interpret the fitness formula of a particular species of organisms as an inherited property of that species of organisms. If the fitness formula varies (slightly) from one individual to another, is inherited, and is subject to random mutation, we can study its evolution in a population of organisms as that of any other trait.

Lund and Parisi [10] showed how fixed fitness formulae and sensory apparatus of populations of organisms shape the behavior which emerges in a particular environment. With evolvable fitness formula it is possible to study the role of co-evolution of the fitness formula and of behavior given a particular sensory apparatus, for example one can observe how organisms with either a generalist behavior (i.e. organisms which tend to eat of all kinds of food distributed in the environment) or a specialist behavior (i.e. organisms which tend to eat only some, and not all, kinds of food distributed in the environment) emerges, along with the evolution of a fitness formula that either assigns equal values to all food types or assigns high values only to the preferred type(s).

The concept of *limited evolvable fitness formulae* should also be considered. In a limited evolvable fitness formula there is a limit on the energy, that an organism can extract with its internal processing mechanism (fitness formula) from a given food source. Although an evolving population of organisms can change its internal mechanism for extracting energy from food, there are likely to be various constraints and limits on this evolutionary process of change. A limited evolvable fitness formula limits the maximum quantity of energy that can be extracted from a given food source. When the limit has been reached, any mutation can only decrease this quantity. Simulations with limited evolvable fitness formulae result in very abrupt behavioral changes of organisms, as shown in figure 1-3, where a limit of 4 energy units is put on the evolvable fitness formula.

Figure 2 shows that the organisms tend to evolve a specialist behavioral strategy for most initial conditions. They obtain almost all of their energy from eating A elements while B and C elements are virtually ignored. At the same time, the energy value of A elements quickly reaches its maximum value of 4 units at around generation 200 (cf. Figure 3), which is accompanied by a parallel rapid increase in fitness (cf. Figure 1).

After generation 200, the energy value of B elements begins to increase gradually and, more slowly, the energy value of C elements increases (cf. Figure 3). Further, it is remarkable that during this stage the number of B or C elements eaten continues to be very low while the number of A elements increases until a stable state at around generation 500 is reached (cf. Figure 2). This is paralleled by a slow increase in fitness (cf. Figure 1).

It is not until around generation 1750, when the energy value of B elements

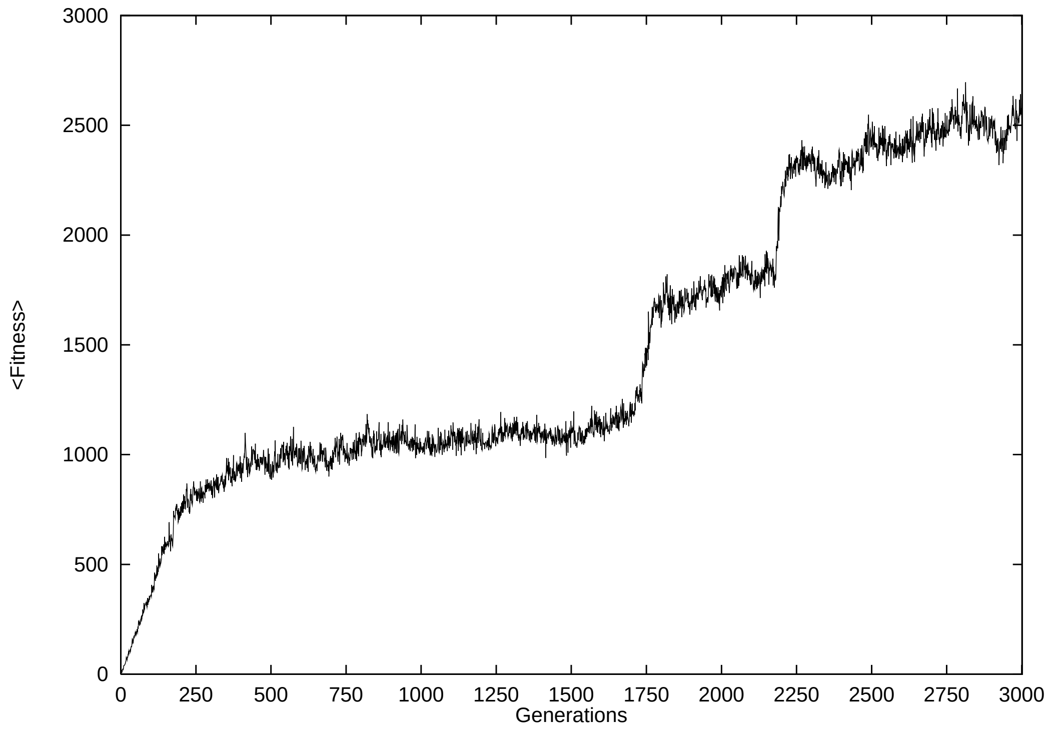

Figure 1: Total fitness of the best organism of each generation with a limited evolvable fitness formula.

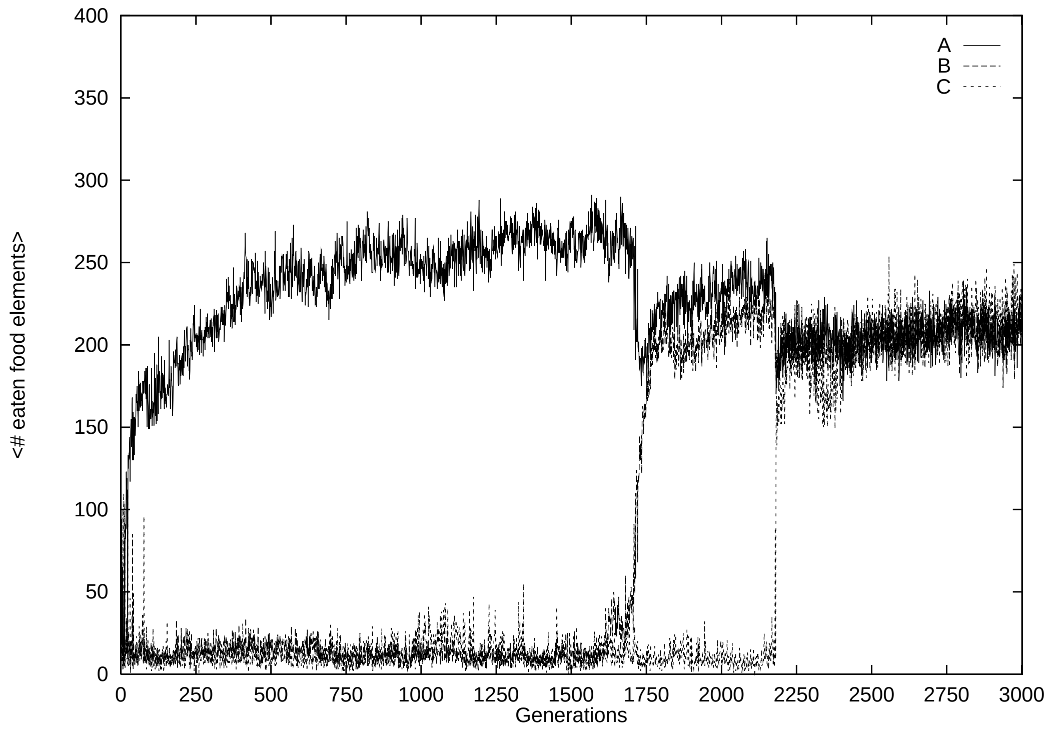

Figure 2: Number of food elements of each type eaten by the best organism of each generation with a limited evolvable fitness formula.

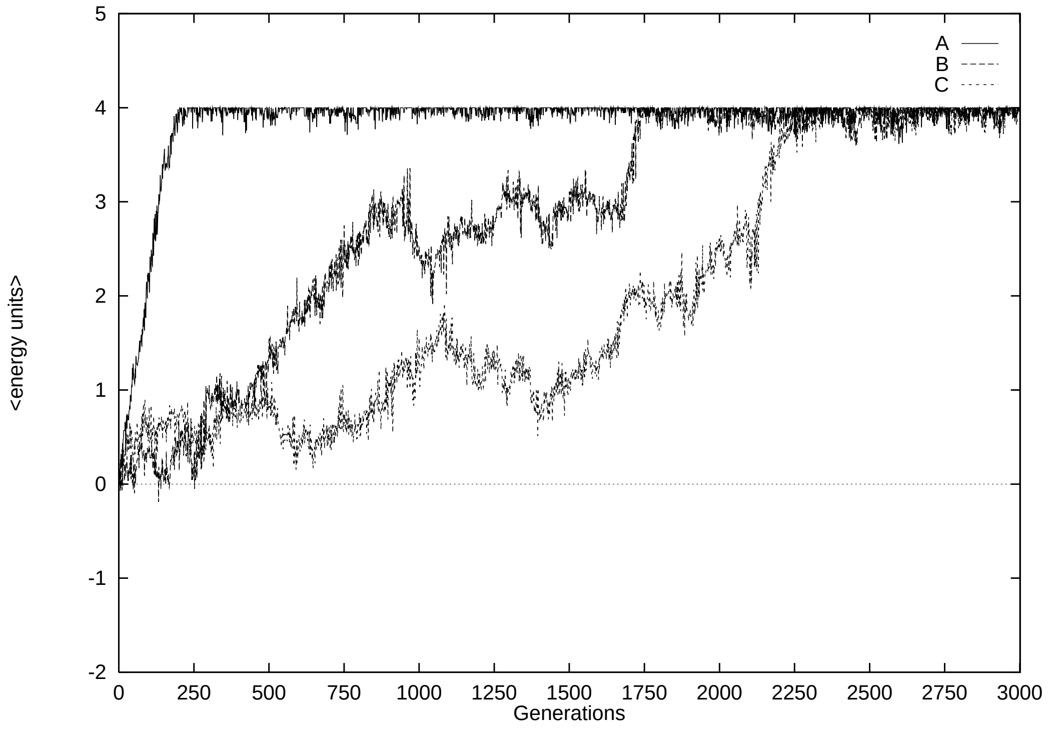

Figure 3: Evolution of the fitness formula for the best organism of each generation.

also has reached its maximum value of 4 energy units, that B elements are actively sought and eaten. In fact, at this stage it happens quite suddenly that A elements and B elements are eaten almost with the same frequency, at a level only slightly below the frequency of eating A elements in previous generations (cf. Figure 2). This result causes a rapid increase in global fitness at this stage (cf. Figure 1).

The same happens some generations later for C elements. They reach their maximum energy value at around generation 2250 and, at this point, organisms suddenly start to eat them with the same frequency as elements of type A and B. As expected, the curve for global fitness shows three sudden increases: before generation 200, at generation 1750, and at generation 2250, while it only increases slightly in the periods in between.

The general conclusion is that when the fitness formula evolves, but there are limits on the possibile energy value of single food types, the behavioral strategy which emerges initially is specialist but then this strategy is replaced by progressively more generalist strategies. The organisms first prefer the single food type that has the highest energy value but when the energy value that can be extracted from this food type has reached its maximum value, they include other food types that can provide increasing quantities of energy.

In a *dynamic environment* , e.g. an environment where food elements are depleted due to some environmental catastrophe such as polution, the evolved system will have to self-adapt to the changed situation. With an evolvable fitness formula the system is allowed to change the fitness landscape in order to cope with the changed environment.

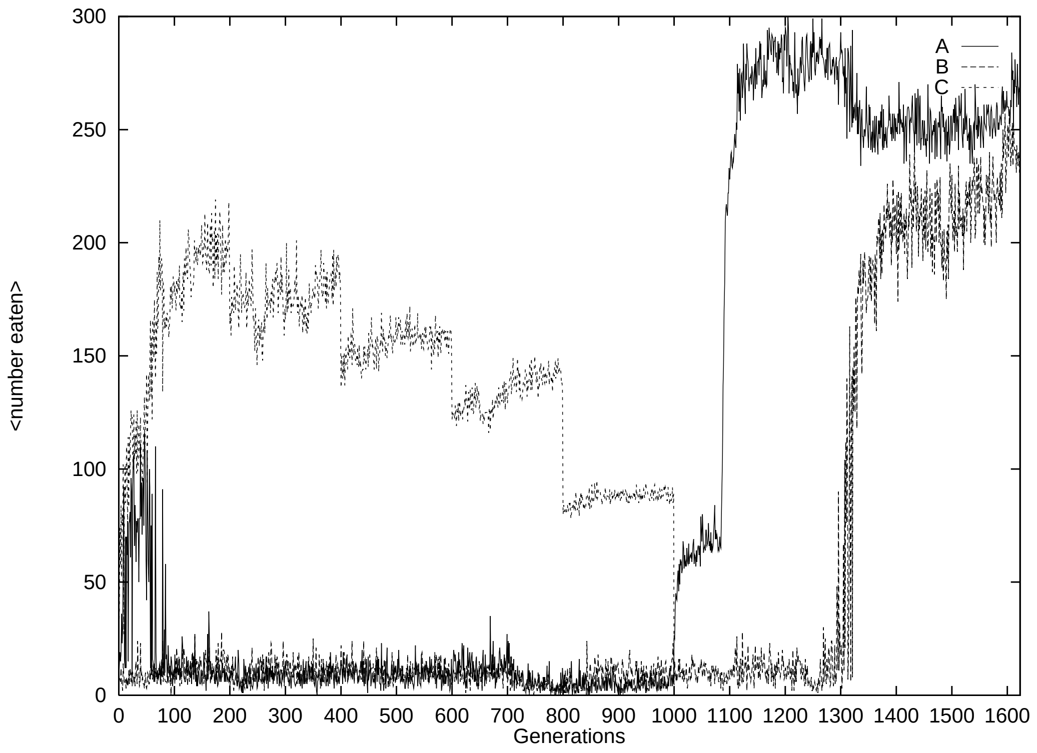

Figure 4: Number of food elements of each type eaten by the best organism of each generation.

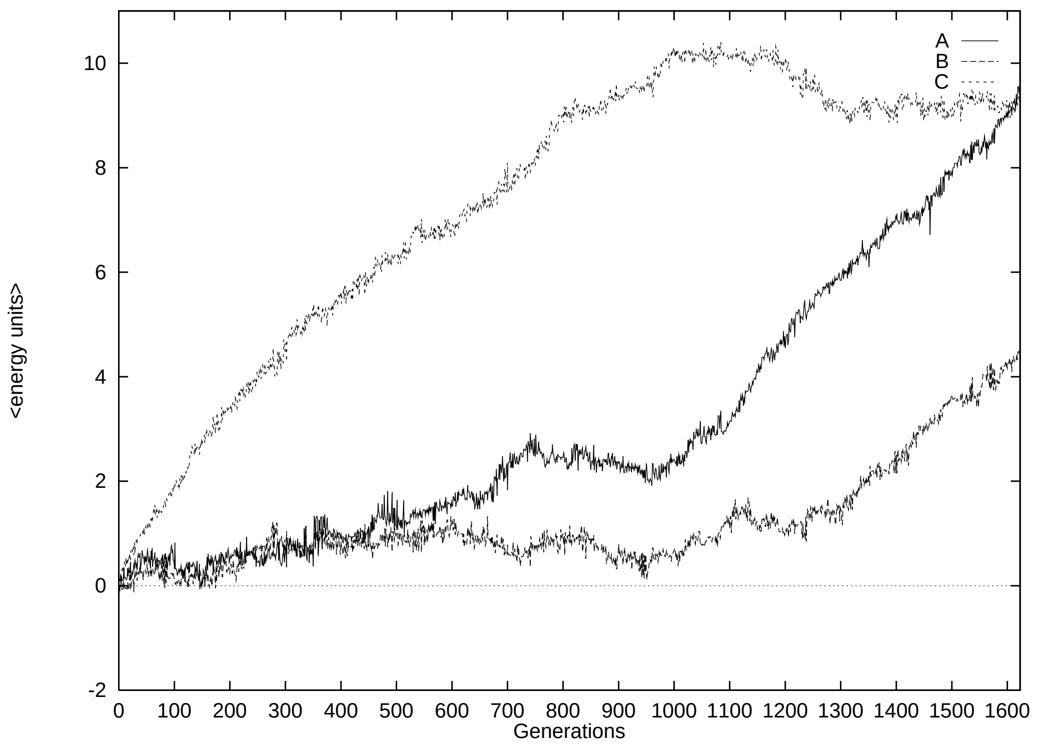

Figure 5: The fitness formula of the best organism of each generation.

A population of organisms with unlimited evolvable fitness formula live in an environment as the one in the simulation above with organisms with limited evolvable fitness formula. However, each 200 generation a number of food elements of the preferred food type (C) disappears from the environment. There is therefore no presure posed from the researcher in parameter settings to change behavior, but from the environment that changes over time. As we will see, the organisms with evolvable fitness formulae are able to adapt to these changes in the environment.

Figure 4 shows the change in behavioral strategy over time. Initially the organisms prefer and eat food elements of type C, and the fitness formula is co-evolved to let the organisms extract high energy from the preferred food type (cf. figure 5). As food elements of type C disappear from the environment each 200 generations, at generation 1000 there will be no food elements of the preferred food type left in the environment. Yet, the organisms quickly turn to eat food elements of type A through a very sudden change in behavioral strategy (cf. figure 4), which is accompanied by an increase in the fitness value of type A in the evolvable fitness formula. The same happens with type B later in the evolution. We can conclude that populations of artificial neural networks with individual evolvable fitness formulae have shown pre-adaptation to environments different from those in which they evolved. A more thourough analysis of this phenonemon is provided by Lund and Parisi [11], who did analysis of the neural network units activations to explain the abrupt changes in behavior.

6 Conclusion.

In many years parameter settings in GAs have been done based on emperical studies, with no underlying theory. The fitness landscape analysis provide such a theory for the selection of genetic operators, and has proven advantageous both quantitatively and qualitatively. It ensures the selection of the right genetic operators in for the problem specific fitness landscape through a very fast estimation of the fitness landscape. But as the fitness landscape may change over time due to both internal and external causes, we have to study the evolved systems in dynamic environments. By using evolvable mutation rates and evolvable fitness formulae we get tools to evolve systems able to adapt to dynamic environments. We have shown how the systems are able to adapt when internal constraints occur (i.e. limited evolvable fitness formula) and when external constraints occur (i.e. dynamic environments).

References.

[1] Bethke, A. D. (1981). Genetic Algorithms as Function Optimizers, Doctoral Thesis, University of Michigan, Ann Arbor.

[2] Booker, L. B. (1982). Intelligent behavior as an adaptation to the task environment. Doctoral Thesis, University of Michigan, Ann Arbor.

[3] De John, K. A. (1975) An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Doctoral Thesis, University of Michigan, Ann Arbor.

[4] Friedberg, E. C. (1985) DNA Repair. Freeman, San Francisco.

[5] Goldberg, D. E. and Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. In Greffenstette (ed.) Proc. of the 2nd Intl. Conf. on Genetic Algorithms. Lawrence Erlbaum, Cambridge, MA.

[6] Harris, M. and Ross, E.B. (1987). Food and Evolution. Toward a theory of human food habits. Philadelphia, Temple University Press.

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

[8] Kaneko, K., and Ikegami, T. (1992) Homeochaos: dynamic stability of a symbiotic network with population dynamics and evolving mutation rates. Physica D, 56, pp. 406-429.

[9] Kirkpatrick, S., Gellat, C. D., and Vecchi, M. P. (1983) Science, 220, pp. 671

[10] Lund, H. H. and Parisi, D. (1994). Simulations with an Evolvable Fitness Formula. Technical Report PCIA-1-94, C.N.R., Rome.

[11] Lund, H. H. and Parisi, D. (1994). Pre-adaptation in Populations of Neural Networks Evolving in a Changing Environment. In *Artificial Life*, 2:2, 179-197.

[12] Manderick, B., de Weger, M., and Spiessens, P. (1991) The genetic algorithm and the structure of the fitness landscape. In Belew and Booker (eds.) Proc. of the 4th Intl. Conf. on Genetic Algorithms. Morgan Kaufmann, San Mateo.

[13] Manderick, B., and Spiessens, P. (1994) How to Select Genetic Operators for Combinatorial Optimization Problems by Analyzing their Fitness Landscape. In Zurada, Marks II, and Robinson (eds.) Computational Intelligence: Imitating Life. IEEE Press, New York.

[14] Mühlenbein, H., and Schlierkamp-Voosen, D. (1993) Analysis of Selection, Mutation, and Recombination in Genetic Algorithms. Neural Network World, 3, pp. 907-933.

[15] Mühlenbein, H., and Schlierkamp-Voosen, D. (1993) Predictive Models for the Breeder Genetic Algorithm: Continuous Parameter Optimization. Evolutionary Computation, 1, pp. 25-49.

[16] Oliver, I. M., Smith, D. J., and Holland, J. R. C. (1987). A study of permutation crossover operators on the travelling salesman problem. In Greffenstette (ed.) Genetic Algorithms and their Applications: Proc. of the 3rd Intl. Conf. on Genetic Algorithms. Lawrence Erlbaum, Cambridge, MA.

[17] Parisi, D., Cecconi, F., and Nolfi, S. (1990). ECONETS: neural networks that learn in an environment. Network, 1, pp. 149-168.

[18] Rechenberg, I. (1994). Evolution Strategy. In Zurada, Marks II, and Robinson (eds.) Computational Intelligence: Imitating Life. IEEE Press, New York.

[19] Starkweather, T., McDaniel, S., Mathias, K., Whitley, D., and Whitley, C. (1991) A comparison of genetic sequencing operators. In Belew and Booker (eds.) Proc. of the 4th Intl. Conf. on Genetic Algorithms. Morgan Kaufmann, San Mateo.

[20] Whitley, D. (1989). The Genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. Proc. 3rd Intl. Conf. on Genetic Algorithms, Morgan Kaufmann, Fairfax.

[21] Wysocki, L., Manser, T. and Gefter, M. L. (1986) Proc. Nat. Acad. Sci. USA, 83, pp. 1847.