The present study shows the important interaction between schemata and internal representation. As a genetic system grows more complex, a methodical testing of the effects of genetic operators is necessary. If novel genetic operations are to prove their usefulness, work of the type performed by De Jong (1975), is required. Similarly, other network specification representations should be studied, to observe effects such as the one described here.

The present study also shows how partially constraining a system may be enough to orient the search in the proper direction. It can't be generalized to all problem areas, but it can be used to simplify a genetic system when it becomes too complex.

References

De Jong, K. A. (1975). An Analysis of the Behavior of a Class of Genetic Adaptive Systems. (Doctoral dissertation, University of Michigan). *Dissertation Abstracts International* **36** (10), 5140B. (University Microfilms No. 76-9381)

de Garis, H. (1990) Genetic Programming. *Machine Learning: Proceedings of the Seventh International Conference*, 132-139.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading.

Grossberg, S. (1982). *Studies of Mind and Brain: Neural Principles of Learning, Perception, Development, Cognition, and Motor Control*, Reidel Press, Boston.

Harp, S. A, Samad, T. & Guha, A., (1989) Towards the Genetic Synthesis of Neural Networks. *Proceedings of the Third Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo.

Ho, M. & Fox, Sidney W. (1988). Processes and Metaphors in Evolution. In Ho, Mae-Wan & Fox, Sidney W. Eds. *Evolutionary Processes and Metaphors* John Wiley & Sons Ltd. Chichester.

Miller, G. F., Todd, P. M. & Hegde, S. U. (1989) Designing Neural Networks using Genetic Algorithms. *Proceedings of the Third Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo.

system. The improvement in fitness across generations is shown in Figure 3. The fitness function used was:

$$O(x) = \frac{100}{1 + \sum(K_t - y_t)}$$

where K_t is the optimal output value at time t, and y_t the actual output from the network at time t.

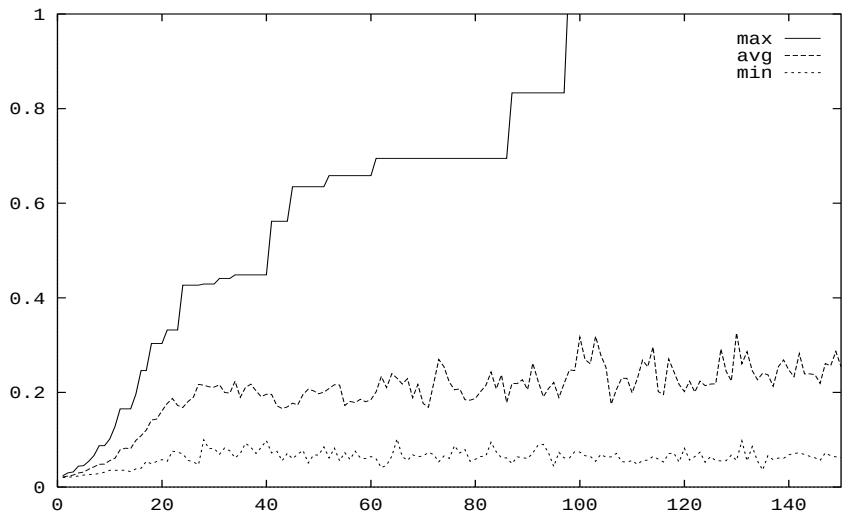

Figure 3: Best, worst, and average population members of the search for a recurrent competitive network over 150 generations.

IV. Constraint Modification

At this point, the fitness function was simplified in order to provide feedback only when the node's activation had settled after each input had changed. The output activations were then compared with two threshold levels, giving three possible states: inactive, inhibited and excited. The discrete result was then compared with a table of the desired network. The disparity from the table was then used as the metric to be optimized. This simpler method of network specification was similarly robust in guiding the GA towards the desired network specification. Since the calculation of the metric is now simpler, the system executed a similar number of generations in less time (about half).

V. Conclusions

The design and use of neural sub-systems is a complex area that merits further research. In the present study, only small, fix-sized networks were treated. How these networks can increase in size, how they are maintained, modified, and coupled to form more complex systems is an important area that must be investigated to better understand the evolutionary processes.

likely to move the schemata to any possible state. This can be achieved in a number of ways. One possible solution would be to use a tri-valued allele, where mutation would be equally likely to switch to any state. This option would avoid the problem of crossover effects on schemata, by not allowing crossover to modify the type of connection used.

The solution chosen still maintains bi-valued alleles, but the meaning of the alleles has been altered. Table 2 shows the table for a connection under the new configuration. Here, it is quite likely that an excitatory connection will eventually move to a disconnected state, and from a disconnected state it is possible to move to either an excitatory or inhibitory state.

Allele pair	Connection
00	Inhibitory
01	Disconnected
10	Disconnected
11	Excitatory

Table 2: Table for new allele representation of connection

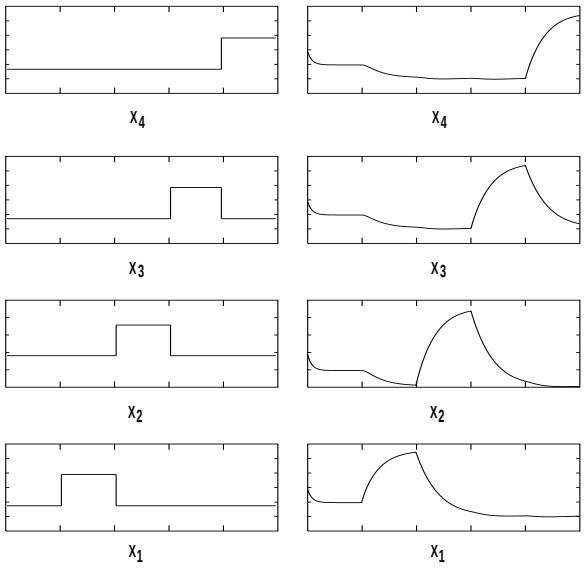

Figure 2: Left: Input sequence. Right: Output of recurrent competitive recurrent circuit

A characteristic response curve similar to the one shown in Figure 2 was requested, given the shown inputs. Since this curve does not contain all possible combinations of inputs, the optimal circuit may respond unexpectedly to inputs not tested.

The resulting network matched exactly that of Figure 1. As desired, the network contains both positive feedback within all the nodes and inhibitory connections to all other nodes. This shows that all possible inputs need not be tested in order to provide sufficient constraints for a unique

Allele pair	Connection
00	Disconnected
01	Disconnected
10	Inhibitory
11	Excitatory

Table 1: Table for allele representation of connection

In the current experiment, the exact resulting circuit and its response curve were known a priori, so a measure of the difference from this curve was used as the function to be maximized.

The problem studied with this setup was a network of feedback nodes. The target configuration was a recurrent competitive feedback circuit (Grossberg, 1982), as shown in Figure 1.

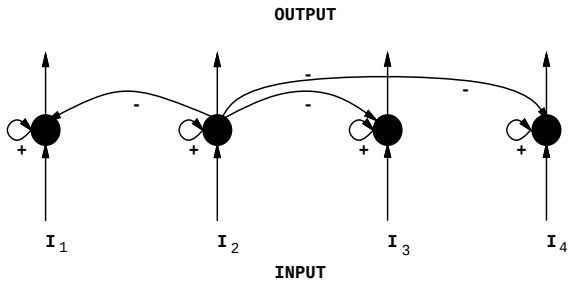

Figure 1: Competitive recurrent circuit. For clarity, only node 2 is shown with all its efferent connections.

III. Representation Modifications

The setup just described did not converge to an optimal result within a reasonable time (400 generations). An analysis of the schemata (similarity templates) involved in the representation of connections, reveals that it is more likely for the system to change genetic material from one state to a state with lower fitness, rather than to a state with higher fitness. This is due to the allele representation, in conjunction with the metric used, and the manner in which crossover and mutation affects genes.

For example, let's assume that a given connection was initially set as not connected (00) when the optimal setting is excitatory (11). Since the probabilities of mutation are quite low (0.03) compared with the probabilities of crossover (1.0), it is quite unlikely that both alleles will be mutated during the same generation in the same individual, therefore, making crossover the more likely candidate for improving performance. This means that a population member with values of 10 must be combined with another member with values of 01. But a setting of 10 is of lower fitness than the original setting of 00. So the member with the lower fitness is quite unlikely to survive and reproduce, in effect slowing the improvement of genetic material.

In order to avoid this problem, the representation must allow for stepwise improvements in performance through the combination of short length schemata. Crossover should be equally

generating a probability of reproduction. Once all individuals have been tested (have "lived") the individuals with higher performance will be more likely to procreate and pass on their genes.

This paper uses GAs to search for the parameters that describe a neural network. These parameters will be used to generate such a network and to analyze its behavior when required to perform a specific task. The aim here is not only to use GAs as a parameter search tool, but as a code building tool. Garis (1990), Miller, Todd, & Hegde (1989), and Harp, Samad, & Guha, (1989) have done some work in this direction. The Miller, Todd, & Hegde's system (1989) can't be considered strictly a network building model, but as a network design or configuration system. It basically determines the weight values in a fully interconnected network where the number of nodes is predetermined. Harp, Samad, & Guha (1989) on the other hand, determine the number of nodes and their connectivity in a fairly complete way. Here, the type of network searched for will be of a more biologically based type.

The internal genetic representation is critical to the speed and optimization level of a genetic system. This was shown by testing different approaches to the genome representation. In order to further accelerate the optimization process, the effects of varying the constraints imposed upon the desired network were also studied.

In this paper, the formation of subsequent generations is based on two genetic operators: mutation and crossover. Mutation is performed by switching each allele to its complementary value with certain probability. Crossovers are performed by selecting two individuals from the population for reproduction. A crossover point is randomly selected somewhere along the extent of the gene, and two progenitors are generated by switching the genetic material of the two progenitors after the point chosen. In the simulations carried out here, the probability of mutation was 0.03. The probability of crossover was 1.00.

In addition, the genome of the best individual of each generation was copied unchanged for the next generation. For further reference on genetic operations and implementation details see Goldberg (1989).

II. System Description

An approach similar to that of Miller, Todd, & Hegde (1989) was used. A network of fixed node size was implemented. The connections between nodes were represented by a 4 by 4 matrix. The GA was used to find which connections should exist and whether these should be inhibitory or excitatory. The activation equation was of the form:

$$\frac{dx_i}{dt} = -Ax_i + (B - x_i)(I_i + \sum f(x_g)) - (x_i + D)\sum f(x_h)$$

where x_i is node i from 1 to 4, A, B and D are constants set at 6.0, 5.0, 5.0 respectively, $f(x)$ is the neuron's feedback equation ($f(x) = x$; if $x > 0$ otherwise $f(x) = 0$), g is the set of excitatory nodes, and h the set of inhibitory nodes. The sets of excitatory and inhibitory nodes are determined by the contents of the genome. For the target circuit, g was the node itself ($f(x_i)$), and h consisted of every other node ($\sum_{h \neq i} f(x_h)$).

The representation of the matrix in the genome was implemented by allocating two alleles to each connection. So, locations 1 and 2 specify the type of connectivity between node 1 and itself. Locations 3 and 4 specify the type of connectivity between node 1 and node 2, and so on. The meaning of each pair of alleles is shown in Table 1.

Genetically Generated Neural Networks I: Representational Effects

CAS/CNS-TR-92-014
Leonardo Martí
Boston University
Center for Adaptive Systems
111 Cummington Street
Boston, MA 02215
lmarti@cns.bu.edu

Abstract

This paper studies several applications of genetic algorithms (GAs) within the neural networks field. After generating a robust GA engine, the system was used to generate neural network circuit architectures. This was accomplished by using the GA to determine the weights in a fully interconnected network. The importance of the internal genetic representation was shown by testing different approaches. The effects in speed of optimization of varying the constraints imposed upon the desired network were also studied. It was observed that relatively loose constraints provided results comparable to a fully constrained system. The type of neural network circuits generated were recurrent competitive fields as described by Grossberg (1982).

I. Introduction

Genetic Algorithms (GAs) have a lot in common with neural networks. While used in engineering applications, neural networks are noted for their neurobiological foundations. GAs are also based on biological foundations. However, not all known natural genetic functions have been incorporated into GAs. GAs have been used mainly as search and optimization procedures. Natural genetics perform some of these tasks, but more importantly, genetic material contains the "program" for life-building. Although recent discoveries (Ho & Fox, 1988) have changed our view on this "program", it is still undisputed that the genetic material (DNA, RNA) contains enough information to generate an organism.

A genetic system consists of a population of organisms or individuals where each member is composed of a gene. A gene is composed of a string of alleles. In this particular case, alleles are represented as single bit binary values. An initial population of an arbitrary number of members is created by assigning random values to each allele. Once the population is established, each individual's gene is tested against some metric. This can be seen as their "life" performance,