

## **'Genotypes' for neural networks**

Stefano Nolfi    Domenico Parisi  
Institute of Psychology  
National Research Council  
15, Viale Marx, 00137 - Rome  
e-mail: stefano@kant.irmkant.rm.cnr.it    domenico@gracco.irmkant.rm.cnr.it

### **INTRODUCTION**

Neural networks are computational models of nervous systems. However, organisms do not possess only nervous systems and other phenotypic traits but also genetic information stored in the nucleus of their cells (genotype). The nervous system is part of the phenotype which is derived from this genotype through a process called development. The information specified in the genotype determines aspects of the nervous system which are expressed as innate behavioral tendencies and predispositions to learn. When neural networks are viewed in the broader biological context of artificial life, they tend to be accompanied by genotypes and to become members of evolving populations of networks in which genotypes are inherited from parents to offspring.

To study evolution computationally genetic algorithms are used (Holland, 1975). An initial population of artificial organisms is tested to determine the fitness of each individual and individuals reproduce as a function of their fitness. To maintain population variability an offspring's genotype can be a new combination of parts of its two parents' genotypes (sexual reproduction) and/or random modifications can be introduced in the reproduction process (mutations). If organisms are modeled by neural networks what an offspring inherits from its parent(s) is a blueprint for the offspring's neural network.

The inherited genotype can completely specify the phenotypic network, i.e. both the network's architecture and connection weights are genetically determined, and in this case the behavior of the network is entirely innate and there is no learning (Parisi, Cecconi, and Nolfi, 1990). Or the genotype specifies the network's architecture but the weights are learned. In this case the genetic algorithm is used to design appropriate network architectures for learning specific tasks (Miller, Todd, and Hedge, 1989; Kitano, 1989). In still other cases, what is selected during evolution are good initial weights for learning or good learning rates and momentums (Belew, McInerney, and Schraudolph, 1991). In all these cases what is genetically inherited are not innate behaviors but innate predispositions to learn certain behaviors.

### **DEVELOPMENT AS MAPPING FROM GENOTYPES TO PHENOTYPES**

To apply genetic algorithms to neural networks it is necessary to codify the neural network (phenotype) into a string (genotype). However, the way in which this coding should be realized is not straightforward. In most current

models the representations of the genotypic and phenotypic forms coincide. That is, the inherited genotype directly and literally describes the phenotypical neural network. This approach encounters problems of scalability (Kitano, 1990). The number of bits of information necessary to encode a network increases exponentially with the number of neurons of the network and, as a consequence, the space to be searched by the evolutionary process also increases exponentially.

Aside from scalability problems, a direct genotype-to-phenotype mapping is biologically implausible. As Langton (1992) has stressed, in real life we cannot predict which phenotype will emerge from a given genotype because of the large non-linearities present in the mapping process. If the genotype is viewed as a set of instructions, it is not the case that each of these instructions will result in a single network property. Rather, the properties of the network emerge as the result of many interactions among the various genetic instructions and their products.

Another problem concerns the nature of the network which is encoded in the genotype. In conformity to standard practice, almost all proposed schemes for the genotypic encoding of network architecture (Miller, Todd, and Hegde, 1989; Kitano, 1990; Belew, 1993) view this architecture as a topological structure or connectivity matrix. On the contrary, real nervous systems are physical objects in three-dimensional space. For example, neurons that are responsible for a single function, e.g. vision, tend to be located in the same part of the brain. Developmentally, the probability that two neurons will end up as being connected depends on the physical location of the two neurons. (For an account of development that emphasizes the role of spatial information, cf. Edelman, 1988.)

Finally, a crucial property of the genotype/phenotype mapping is its temporal character. Biological development (or maturation) does not simply yield an "individual" but the phenotypic individual is a sequence of different phenotypic forms which are sequentially generated by the genotype in interaction with the environment.

In this chapter we describe a method for codifying a neural network architecture into a genetic string which is inspired by the characteristics of neural development in real animals. Inherited genetic material specifies developmental instructions that control the axonal growing and branching process of a set of neurons. The phenotypic network architecture is not directly specified in the genotype and cannot be "read" in the genotype by simply inspecting it. It gradually emerges from the execution of the various developmental instructions that constitute the genotype and from their interactions.

During the growth process (see Figure 1, left side), connections between neurons with their synaptic weights are established. Both individual neurons and their axonal growth processes are physically located in (bidimensional) space. When a growing axonal branch of a particular neuron reaches another neuron a connection between the two neurons is established. Hence, the spatial location of neurons is critical in determining the resulting connectivity. However, only some neural branchings succeed in establishing connections and only some of the interconnected neurons take part in the travelling of activation from input to output and therefore are functional in determining

the network's behavior. Functional neurons and connections are represented as larger circles and thicker lines on the right side of Figure 1. (Nonconnecting branchings have been removed.)

We first describe simulations in which the growth process is supposed to take place instantaneously at birth and then we describe other simulations in which the growth process continues for a certain time after birth.

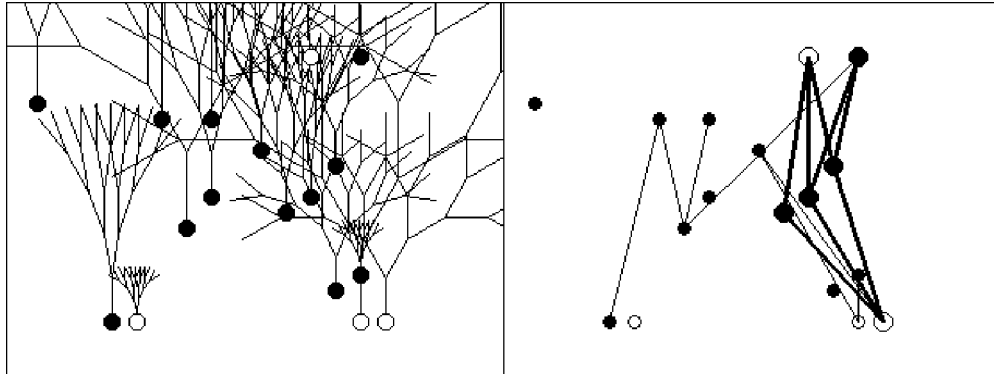


Figure 1: Development of a neural network from a randomly generated genetic string. The growing and branching process of axons is shown on the left side of the Figure. The resulting neural network after removal of nonconnecting branches is shown on the right side. Isolated or nonfunctional neurons and connections are represented by small circles and thin lines. Functional neurons and connections are represented as large circles and thick lines. The bottom layer contains sensory neurons, the upper layer motor neurons, and the remaining layers internal or hidden neurons.

### EVOLUTION OF ARTIFICIAL ORGANISMS USING GROWING NEURAL NETWORKS

We are interested in the evolution of organisms that can efficiently capture food elements randomly distributed in the environment. The environment is a two-dimensional square divided up into cells. At any particular moment an organism (O) occupies one of the cells. Food elements are randomly distributed in the environment with each food element occupying a single cell. O has a facing direction and a rudimentary sensory system that allows it to receive as input a coding of the angle (measured clockwise relative to where O is currently facing) and distance of the nearest food element. O is also equipped with a simple motor system that provides it with the possibility, in a single action, to turn through any angle from  $90^\circ$  left to  $90^\circ$  right and then move from 0 to 5 steps forward. Finally, when an O happens to step on a food cell, it eats the food element which disappears.

To evolve Os which are able to reach food elements, a genetic algorithm is applied to the Os' "genotypes". We begin with 100 randomly generated genotypes each yielding a network with a different architecture and a different assignment of connection weights. This is Generation 0 (G0). G0 networks are allowed to "live" for 8 epochs, with an epoch consisting of 250 actions in 5 different environments (50 actions in each) for a total of 2000 actions. The environment is a grid of  $40 \times 40$  cells with 10 pieces of food randomly distributed in it. Os are placed in individual

copies of these environments, i.e., they live in isolation.

At the end of their life, Os have the possibility to reproduce. However, only the 20 individuals which have accumulated the most food in the course of their life are allowed to reproduce by generating 5 copies of their genotype. These  $20 \times 5 = 100$  new organisms constitute the next generation (G1). Random mutations are introduced in the copying process (by replacing 20 bits with a randomly selected value) resulting in possible changes of the architecture and of the values of the weights (sexual reproduction or crossover is not applied). After the Os of G1 are created, the process is repeated for 1000 generations.

Os' genotypes (see Figure 2) are bit strings divided up into 40 blocks, each block corresponding to the growth instructions for a single neuron. There are three types of neurons, sensory neurons, internal neurons, and motor neurons. The first 5 blocks in a genotype correspond to sensory neurons, the last 5 blocks to motor neurons, and the 30 intermediate blocks to internal neurons. Internal neurons can be arranged in a maximum of 7 layers. The standard logistic function is used to compute the activation value of the neurons. Each block of the genotype contains instructions that determine the following properties of the corresponding neuron:

- (a) The "neuron expression gene" determines if the other instructions contained in the block will be executed or not, i.e. if the corresponding neuron will be present or not in O's nervous system.
- (b) The two "physical position genes" specify the Cartesian coordinates of the neuron in the bidimensional nervous system.
- (c) The "branching angle gene" and the "segment length gene" determine the angle of branching of the neuron's axon and the length of the branching segments. While angle of branching and segment length can vary from one neuron to another in the same network, all the branchings of the axon of a given neuron have the same branching angle and the same length. The growing axon of all neurons branches a fixed number of times (five) with exclusively binary branchings. In order to obtain feedforward networks without recurrent connections, the branching angles are so restricted that all units send their growing axons in the direction from input units (bottom layer) to output units (top layer).
- (d) The "synaptic weight gene" determines the synaptic coefficients of all the connections that will be established by the neuron. All connections departing from the same neuron have the same weight.
- (e) The "bias gene" represents the activation bias or threshold of the neuron.
- (f) The "neuron type gene" specifies, in the case of a sensory neuron, if the sensory neuron codifies angle or distance of food, and in the case of a motor neuron, if the motor neuron codifies the angle of turn or the amplitude of the motor step. If more than a single output neuron codifying the same motor information (e.g. length of step forward) exist, the actual motor output is taken to be the average of the activation levels of the various neurons.

If we look at Os' fitness (i.e. number of food elements eaten) throughout the 1000 generations we see the evolutionary emergence of Os that are increasingly able to approach food elements. Figure 3 shows the fitness values of the Os of the winning lineage for each of the 1000 generations. The winning lineage is the lineage of the

best individual of the last generation. The lineage is constituted by this individual, its (single) parent in the preceding generation, its grandparent, its grandgrandparent, etc., until the originator of the lineage in the first generation is reached. The evolutionary increase in fitness implies that generation after generation, Os are able to adapt their architecture and synaptic weights to the evolutionary task. The total number of food elements available in the entire lifetime of an O is 400. After 1000 generations an average of 72% of this food is eaten in 10 replications of the simulation. In the best of these 10 simulations a value of 89% is reached.

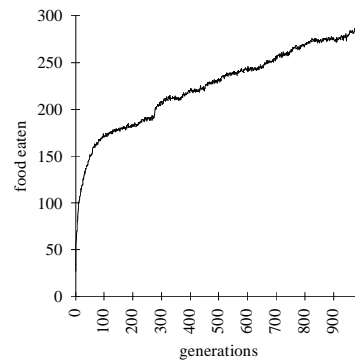
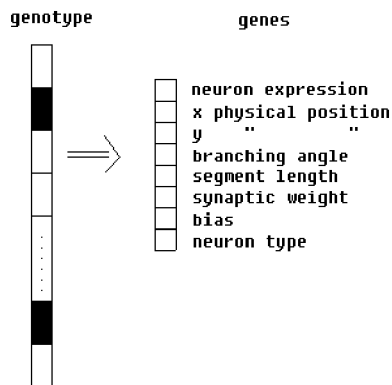


Figure 2: Developmental instructions specified in an O's genotype. Inactive blocks which correspond to unexpressed neurons are represented as filled cells while active blocks are represented as empty cells.

Figure 3: Fitness (=total number of food elements eaten during life) of Os of the winning lineage. The graph represents the average result of 10 simulations starting from initial populations with different randomly generated genotypes.

If we analyze how network architecture changes in these individuals during the course of evolution we notice that evolved architectures are extremely simple. There is some increase in complexity but even the architectures of the last generations include very few units (much less than the maximum number allowed, i.e. 40) and very few connections. Similar simulations conducted with networks with a fixed externally decided architecture required many more connections and resulted in poorer performance.

Other interesting results concern the observed regularities in the evolutionary development of the Os' sensory and motor systems. In the very first generations (cf. G0 of Figure 4) Os have neural architectures which take into account only food direction (angle), on the sensory side, and control only direction of movement, on the motor side. Some generations later (from G5 to G225) we see the emergence of architectures which control the amplitude of the motor step in addition to direction of movement but still rely on sensory information which is restricted to food direction. Only in the last generations (G999) do Os become sensitive to food distance as well. This type of progression in the selection of sensory and motor neurons throughout generations is observed in most of the simulations we performed.

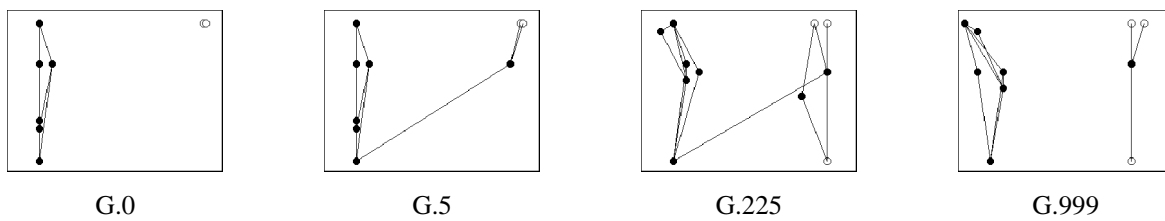


Figure 4. Functional architectures of individuals of the winning lineage in the most successful simulation. Only 4 of the 1000 individuals are shown. Within the sensory layer filled circles represent neurons sensitive to the angle of the nearest food element and empty circles neurons sensitive to its distance. In the motor layer, filled circles represent motor neurons that codify the angle of turn and empty circles neurons that codify the amplitude of the motor step.

It is also interesting that, despite the simplicity of the task, neural architectures progressively structure themselves into functional sub-networks or neural modules. (For the importance of structure in the architecture of neural networks cf. Rueckl, Cave and Kosslyn, 1989). At G999, the division of O's neural system into two functional sub-networks is evident. There is a larger sub-network which computes O's direction of movement as a function of the state of the neuron sensitive to food angle, and a smaller sub-network that controls O's movement speed as a function of food distance.

Finally, we observe that the complexity of the sub-networks correspond to the complexity of the mapping performed by these sub-networks. In other words, neural resources are allocated where they are needed. The mapping between the angle of the nearest food element and the angle of turn should be relatively complex in order to ensure high performance. An efficient network must produce similar motor responses for stimuli that are located few degrees on the right or on the left with respect to the facing direction of O (i.e. for input values close to 0.0 or to 1.0) and very different output responses for stimuli that are located a few degrees to the left or to the right of the back of O (i.e. input values around .5). On the contrary, the mapping between the distance of the nearest food element and the amplitude of the motor step can be much simpler. What is observed in the simulations is that the evolved sub-networks that are responsible for computing the angle of turn are always more complex than the sub-networks that are responsible for computing the amplitude of the motor step.

### **DEVELOPMENT AS A TEMPORAL PROCESS**

The developmental model used in the simulations of the preceding section and, as far as one can tell, all those published so far, are completely atemporal. Development is conceived as a process already completed when the network begins to "live". On the contrary, development is, above all, a temporal phenomenon. Development means that the individual which is developing is undergoing a sequence of changes. The information contained in the genotype is not only reflected in the initial state of the organism at birth but is reflected in the sequence of changes that occur in the individual throughout the individual's developmental age.

The successive forms of an individual must be adapted to the developmentally changing environments in which each form happens to live. Therefore, evolution shapes not an abstract 'individual' but each and all of its different developmental forms. This implies that not only evolution shapes development but also that development constrains evolution. Each developmental form is adapted in that it is part of a particular succession of forms. Hence, evolution cannot just change one of these forms without also changing the other forms.

In order to realize a developmental process that takes place in time we can simply replace the 'neuron expression' parameter in the genotype of our growing networks with a 'neuron expression time' parameter. While the neuron expression parameter specifies in a binary fashion whether the corresponding neuron will appear or not in the phenotypic network, the new expression time parameter specifies at what point in time, during the life of an individual, the neuron will make its appearance and will grow its axonal branchings in the individual's neural network. The life of an individual is divided up into 8 epochs. To allow for the possibility that a particular neuron will never be part of the phenotypic network, the parameter specifies a value that may go from epoch 0 (the neuron is expressed at birth) to epoch 15. If the value of the parameter is higher than 8, the corresponding neuron won't be part of the network controlling O's behavior during its life.

Another set of 10 simulations has been run with this new coding of the genotype. All the other parameters remained the same as those of the simulation described in the preceding section. Os reached on average 61% of the optimal performance after 1000 generations, with one of the 10 simulations reaching 95%.

It is interesting to see how the neural growth process which takes place during the life of an individual affects the individual's fitness, i.e. to count the number of food items eaten in successive epochs of life by the same individual. The results are shown in Figure 5. From this figure it is clear that, even if individuals are evaluated in the same way during all their life, evolution selects genotypes that result in useful developmental changes. Genetically controlled neural development causes changes in behavior that allow an individual to capture an increasing number of food elements as the individual develops.

It is reasonable to expect that the role of neural development would be enhanced if the fitness criterion would change during an individual's lifetime, i.e. if young individuals were rewarded differently from adults. As shown in Lewis, Fagg, and Solidum (1992), complex behaviors can emerge more easily if the sub-components of such behavior are separately rewarded. Temporal development allows evolution to use different fitness criteria in different parts of an individual's lifetime and therefore to reward both simpler and more complex behavior.

A second aspect which it is interesting to note in these simulations is when functional and nonfunctional neurons appear during development and whether their time of appearance is anticipated or postponed in Os of successive generations. Figure 6 shows that functional neurons appear earlier than nonfunctional neurons and that functional neurons tend to be evolutionarily anticipated during ontogeny while this is much less true for nonfunctional

neurons.

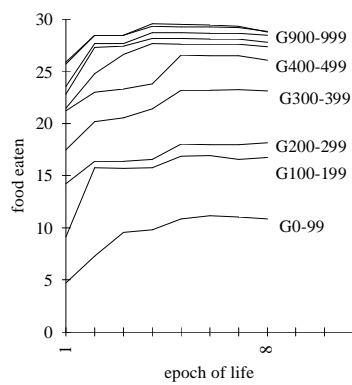


Figure 5. Number of food elements eaten in successive epochs of life by Os of the successful lineage. The graph shows the average result of 10 different simulations. Each curve represents the average performance of 100 successive generations.

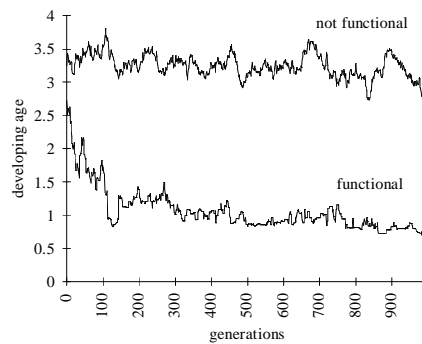


Figure 6. Average epoch of appearance during development of functional and nonfunctional neurons across evolutionary time. Functional neurons appear earlier than nonfunctional neurons. While the time of appearance of functional neurons decreases significantly during evolution, especially in the first generations, this is less true for nonfunctional neurons. Each curve represents the average result of 10 different simulations.

These results have interesting consequences from an evolutionary perspective. After the initial stages of evolution in which a minimal functional architecture and a sufficient level of performance have been established, changes due to mutations are likely to have negative rather than positive influences on fitness. On the other hand, changes are necessary to allow evolution to discover more effective solutions. The fact that Os develop in time allows changes to have a less dramatic impact on fitness by occurring in later stages of development. Useful changes that are preserved for a sufficient number of generations can then be anticipated in development in order to be more effective while useless ones are discarded. Thus, development may represent an exploratory "growth cone" for evolution by favouring the possibility to incorporate novelties with reduced risk.

### LEARNING AND EVOLUTION

The networks described so far do not learn anything during their life. Their behavior (searching for food) is entirely genetically transmitted although it is congenital (already present at birth) in the networks with instantaneous development and it develops during life in the networks with temporally distributed development.



However, as already noted in the introduction, genotypes for neural networks may incorporate not innate behaviors but innate predispositions to learn some behaviors. Evolution creates the genetically inherited conditions for learning to occur. However, as in the case of development, the interaction between learning and evolution is bidirectional, i.e. evolution shapes learning but at the same time learning constrains evolution.

To investigate the interaction between learning and evolution we have run a new set of simulations using genotypes that encode the weights of a fixed network architecture. The architecture includes two parts, a standard network and a teaching network (cf. Figure 7). The two networks share the same input units but have separate sets of internal and output units. The output units of the standard network are interpreted as generating motor actions while the output units of the teaching network (i.e. the teaching units) are interpreted as generating teaching input for the motor output units of the standard network.

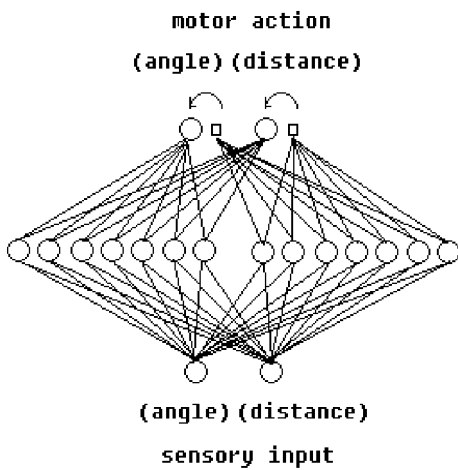


Figure 7. Os' architecture. The 3-layer network includes two sub-parts, a standard network and a teaching network. The two sub-networks share the input units but each has a separate set of seven hidden units and two output units.

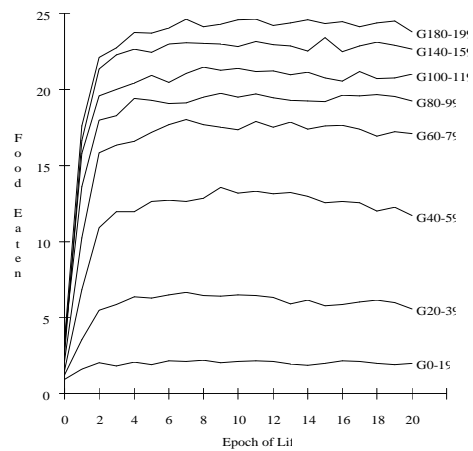


Figure 8. Eating performance of the best individuals of successive generations during lifetime (performance at epoch 0 of life, i.e. at birth, is calculated by letting Os live for one epoch with no learning). The average results of the best Os of a group of 20 successive generations are shown. Each curve represents the average of 10 different simulations.

When an O is placed in the environment that has been described above, a sequence of events will occur. Sensory input is received on the input units. Activation flows from the input units to both the hidden units of the standard network and to those of the teaching network and then from these hidden units separately to the output units of the

standard network (encoding motor actions) and to the output units of the teaching network (encoding teaching input for the standard network). The activation values of the motor output units are used to move *O* in the manner specified by these values, thereby changing the sensory input for the next cycle. The backpropagation algorithm uses the discrepancy between the two sets of output units to change the connection weights of the standard network.

As in the preceding simulations, *Os* should be able to capture food elements randomly distributed in the environment. *Os* are allowed to "live" for 20 epochs for a total of 5000 actions. A learning rate of 0.15 was used for backpropagation learning. Mutations are introduced in the copying process by selecting at random 4 weights of the standard sub-network and 4 weights of the teaching sub-network and adding a random value between +1.0 and -1.0 to their current value. All other parameters are those of the preceding simulations.

If we look at *Os*' fitness throughout the 200 generations we see that *Os* that are increasingly able to approach food elements tend to evolve. However, the ability to approach food is not present at birth but is learned during life. Figure 8 shows the number of food items eaten in successive epochs of life by the best individuals of each generation. While there is no improvement in eating performance during life for the first 20 generations, later in the evolutionary process eating increases after the very first epochs of life and this increase is greater with successive generations of *Os*. This implies that evolution progressively selects networks that internally generate teaching input which can be used by the networks themselves to learn during their life how to search for food efficiently.

It is interesting to note that evolution does not select *Os* which perform well at birth but selects *Os* which have the property of learning to perform well, i.e. *Os* that have a predisposition to learn. *Os*' performance at birth does not increase across generations. What increases across generation is the ability to learn the desired performance. This genetically transmitted ability to learn is incorporated in the weights of both the teaching network and the standard network. The teaching weights generate increasingly good teaching input for the standard network. But the weights of the standard networks are also critical. In fact, evolved *Os* lose their ability to learn to approach food elements if their standard weights (i.e. the weights that are modified by learning) are randomized at birth. The standard weights, which determine *Os*' behavior at birth, are not selected for directly incorporating good eating behaviors but they are selected for their ability to let such a behavior emerge as a result of learning.

As in the case of development, we find that both evolution shapes learning and learning constrains evolution. Evolution shapes the result of learning by selecting favourable initial conditions (i.e. the weights of the standard network). Learning constrains evolution by ruling out the possibility of evolving an innate and congenital ability to find food and favouring the evolution of a predisposition to learn to find food.

## **CONCLUSIONS**

If neural networks must be credible as models of the nervous system of real organisms (aside from the inevitable simplifications of simulations) they must allow us to study neural evolution and neural development and not only the functioning of the adult nervous system in controlling the organism's behavior. To this purpose what is needed are distinct representations for inherited genotypes and for phenotypic networks and some model of the complex mapping from genotypes to phenotypes (development).

This chapter has presented some models of genotypes for neural networks and of the genotype-to-phenotype mapping. Genotypes can directly incorporate innate behaviors that require no learning or they can incorporate innate predispositions to learn some behaviors. Innate behaviors can be present at birth (congenital) or they can develop during life under the control of genetically specified information. With regard to the mapping of the information contained in the genotype into the phenotypic network, the mapping can be direct and one-to-one or, with more biological plausibility, it may be more indirect and many-to-many. We have presented a model of a complex mapping in which genetic instructions interact among themselves to determine the final shape of a network viewed as a physical object and not simply as a topological structure.

In multicellular organisms, other important processes involved in the mapping between genotype and phenotype are cell division (one cell becomes two "daughter" cells) and the migration of neurons to reach their terminal position in the brain. These processes are ignored in the models described in this article but have been examined in other models (Wilson, 1987; Belew, 1993). In Cangelosi, Parisi and Nolfi (submitted) both cell division and migration and the growth of neurites are included in the same model.

The result of simulations with neural networks with genotypes show that for both genetically controlled development (maturation) and experientially controlled learning, their relation with evolution is that of a reciprocal influence. Evolution shapes the course of development and the nature of what is learned and development and learning modify the way in which evolution operates. Future works in this area will concentrate on the interaction between genetically controlled changes (maturation) and changes due to experience and learning.

## **ACKNOWLEDGMENTS**

This research was supported by P.F. "ROBOTICA", C.N.R., Italy

## **REFERENCES**

Belew, R.K., 1993, Interposing an ontogenetic model between Genetic Algorithms and Neural Networks, in

Advances in Neural Information Processing (NIPS5), (J. Cowan, ed.), San Mateo, CA, Morgan Kaufmann.

Belew, R.K., McInerney, J, and Schraudolph, N.N., 1992, Evolving networks: Using the Genetic Algorithm with connectionist learning, in Artificial Life II, (C.G. Langton, ed.), Reading, Mass., Addison-Wesley.

Cangelosi, A., Parisi, D., and Nolfi, S. Submitted. Cell division and migration in 'genopytes' for neural networks. Network

Edelman, G.M., 1988, Topobiology. An Introduction to Molecular Embriology. New York, Basic Books.

\*Hinton, G.E., and Nowlan, S.J., 1987, How learning guides evolution. Complex Systems, 1:495-502.

\*Holland, J.J., 1975, Adaptation in natural and artificial systems. Ann Arbor, Michigan, University of Michigan Press.

Kitano, H., 1990, Designing neural networks using genetic algorithms with graph generation system. Complex Systems, 4:461-476.

\*Langton, C.G., 1992, Artificial life, in Lectures in Complex Systems, (L. Nadel and D. Stein, eds.), 1991 Reading, Mass., Addison-Wesley.

Lewis, M.A., Fagg, A.H., and Solidum, A. 1992, Genetic programming approach to the construction of a neural network for control walking robot, Proc. IEEE Int. Conference Robotics & Automation, Nice, France, pp.2681-2623.

\*Miglino, O., Nolfi, S., Parisi, D., in press, Discontinuity in evolution: how different levels of organization imply pre-adaptation, in Plastic Individuals in Evolving Populations, (R.K. Belew, M. Mitchell, eds.), SFI Series, Reading, Mass., Addison-Wesley.

Miller, G.F., Todd, P.M., and Hedge, S.U., 1989, Designing neural networks using genetic algorithms, in Proc. Third Intl. Conf. on Genetic Algorithms, San Mateo, CA, Morgan Kaufmann.

Parisi, D., Nolfi, S., in press, How learning can influence evolution within a non-Lamarckian framework., in Plastic Individuals in Evolving Populations, (R.K. Belew, M. Mitchell, eds.), SFI Series, Reading, Mass., Addison-Wesley.

\*Parisi, D., Cecconi, F., Nolfi, S., 1990, Econets: neural networks that learn in an environment. Network, 1:149-168.

Rueckl J.G., Cave, K.R., and Kosslyn, S.M. 1989, Why are "what" and "where" processed by separate cortical visual systems? A computational investigation. Journal of Cognitive Neuroscience, 1:171-186

Wilson, S.W., 1987, The genetic algorithm and biological development, in Proceedings of the Second International Conference on Genetic Algorithms. Hillsdale, NJ, Erlbaum.

