The Tenth International Symposium on Artificial Life and Robotics 2005(AROB 10th '05),
B-con Plaza, Beppu, Oita, Japan, February 4-6, 2005

Evolution of Development and Heterochrony in Artificial Neural Networks

Artur Matos Reiji Suzuki Takaya Arita
Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, Nagoya 461-8601, Japan
amatos@create.human.nagoya-u.ac.jp, {reiji,arita}@nagoya-u.jp

Abstract

Recently, evolutionary algorithms coupled with simulated developmental processes have been used successfully for generating designs ranging from neural networks to artificial creatures. Although several of these models do exist, their evolutionary dynamics, and more specifically how the ontogenic models themselves interact with artificial evolution are still poorly understood. One of these specific interactions, and of particular importance in biological systems is heterochrony — the change in timing and rate of developmental events by evolution. In this paper, we analyze heterochronic change in one artificial developmental model - the cellular encoding model first described by Gruau [1]. For this purpose, we apply the framework and methods defined by Alberch *et al* [2] for biological systems to neural networks evolved for the odd-3-parity problem. Preliminary results show that: 1) All heterochronic changes occur with significant frequency; 2) The combined effects of predisplacement, hypermorphosis, and neoteny was the most common heterochronic change; 3) Pure recapitulation (isomorphosis) is prevalent.

Keywords: Heterochrony, evo-devo, Cellular encoding, Artificial embriogeny, Neural Networks.

1 Introduction

Evolutionary algorithms have been used successfully for generating designs ranging from neural networks, to full artificial creatures with both generated morphology and behavior. One of the reasons for this recent success is due to the integration of simulated developmental processes with evolutionary algorithms. Although several of these models do exist, their evolutionary dynamics, and more specifically how the ontogenic models themselves interact with artificial evolution are still poorly understood.

In biological systems, one of the key concepts in this interaction is known as heterochrony. Heterochrony, as it is usually defined in evolutionary biology, is the change in the rate and timing of developmental events caused by evolution. Heterochrony is prevalent in the evolution of species: some examples include the conservation of juvenile characters in salamanders and the loss of the tadpole stage in toads. Due to this, there is a wide array of studies and data available on heterochrony in biological systems.

For artificial neural networks, previous studies have shown that heterochrony does indeed occur in artificial systems: For instance, Cangelosi [3] evolved neural networks for foraging food in a simulated environment. By comparing the development processes between ancestor and descendant networks, he could observe changes in the timing of developmental events. However, this analysis was only done for a small number of individuals, and it was not extended to a whole phylogenetic tree. Thus, studies focusing on heterochrony in a large scale in artificial systems are still lacking. Specifically, these questions are still largely unanswered: 1) What kind of heterochrony processes are more common in artificial systems? 2) How does the developmental model and the evolutionary parameters affect heterochrony? 3) How does heterochrony in artificial and biological systems relate to each other? These points are the main motivations behind this paper.

In attempting to answer these questions, we applied the framework defined by Alberch *et al* [2] for biological systems to the evolution and development of neural networks. This framework defines a precise terminology for classifying heterochrony. For simulating neural network development, we used the cellular encoding model first described by Gruau [1]. This is one of the earliest models described in the literature, and representative for grammar based models of development. We used this encoding coupled with genetic programming for solving the odd-3-parity problem and then analyzed the resulting growing dynamics on important traits like the number of neurons, their average degree and also how the fitness value itself changes within ontogeny.

2 Alberch *et al*'s framework

The framework by Alberch *et al* for classifying heterochrony is widely used for biological systems. This framework is based on the measurement and comparison of quantitative traits, for instance, body length, width or height. The traits are measured as development unfolds, yielding growth curves. These growth curves can then be compared between related species for understanding the heterochronic change involved.

The basis for comparison lies on three metrics that can be extracted from the growth curves: α — the time when growth starts, β — the time when growth ends, and K — the growth rate. Comparing these values between species yields the outcomes summarized in figure 1. For instance, considering only changes in the K parameter, two outcomes are possible: if the descendant would grow faster than the ancestor (K would be larger), the corresponding

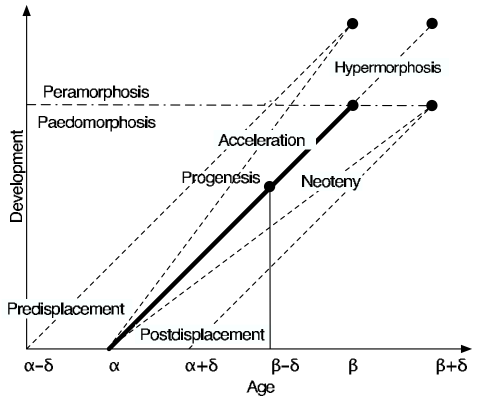

Figure 1: The formalism of Alberch *et al.* A trait measure is plotted against developmental time in the X axis. The solid line plotted from α to β represents the growth curve for the ancestor, while the remaining ones possible heterochronic outcomes for the descendant.

outcome is acceleration. The reverse process — the descendant growing slower than the ancestor — is labeled neoteny. Furthermore, heterochronic changes can be combined on the three parameters, as for instance postdisplacement and hypermorphosis would refer to an increase in both α and K respectively.

3 The model

The phenotypes in this model are simple boolean neural networks: Nodes are simple threshold neurons, with a threshold of either 0 or 1. The connections between neurons can either be -1 or 1. Neurons are activated if the sum of the values on their incoming connections is above their threshold.

For the developmental model, we used the cellular encoding model by Gruau [1]. In this model, the neural network starts as a single neuron and undergoes several developmental events as specified in the genotype. The genotype is represented as a Genetic Programming (GP) tree with nodes as developmental commands such as neuron division, setting the weight and threshold and similar. Each neuron has a pointer to the GP tree representing its current developmental stage. Development occurs in a parallel fashion: in each time step, each neuron executes the command pointed by its register in the tree and moves to the following leaf. The arity of each GP node depends on the command used: for instance, commands for neuron division have two children, representing separate programs for each of the daughter neurons. Development for the network finishes when all the neurons have reached their final leaf node in the tree.

Using this model, we evolved networks for the 3-odd-parity problem, a standard problem for GP. The solution is defined as a neural network with at least 3 inputs, that outputs *true* whenever the number of *true* inputs is odd. A fitness function based on the number of wrong outputs didn't produce a good performance so we used the fitness function used by Gruau in [1]. It is defined by:

$$f(out_{eval}) = 1 - \frac{I(out_{right}, out_{eval})}{H(out_{right})}, \qquad (1)$$

where out_{eval} is the output vector of the evaluated network and out_{right} the expected correct output vector for the problem. $I(X,Y)$ is the mutual information between X and Y:

$$I(X,Y) = \sum_{x=0}^{1}\sum_{y=0}^{1} P_{XY}(x,y) \cdot log_2 \frac{P_{XY}(x,y)}{P_X(x) \cdot P_Y(y)}, \qquad (2)$$

and $H(X)$ is the information entropy of X:

$$H(X) = \sum_{i=0}^{1} P_X(i) \cdot log_2(P_X(i)), \qquad (3)$$

with $P_X(x)$ as the probability of $X = x$, and $P_{XY}(x,y)$ as the joint probability of $X = x$ and $Y = y$. This fitness function is defined in the range $[0,1]$, with 0 as the best fitness. Please note that due to using mutual information, either correct networks or networks that output the inversed expected output will have the same best fitness.

We adopted a GP based system without crossover, and tournament selection was used. Each individual is mutated (addition, deletion, replacement of nodes) with a certain probability, while the others are simply reproduced. The individuals are therefore connected among generations by either reproduction or mutation, forming lineages.

4 Basic results

Using the model and fitness function described in the above section, we evolved networks for analyzing heterochronic change. Population size was 200, and the mutation rate was set at 80%. The fitness graph for a typical run can be seen in figure 2 and the best network in figure 3. All of the results reported on this article are from this run. We can see that the average fitness gradually decreased while the best fitness decreased discontinuously, and this model was able to solve the problem in 314 generations.

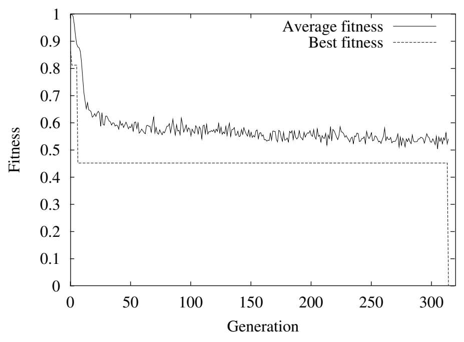

Figure 2: Fitness graph for a typical run.

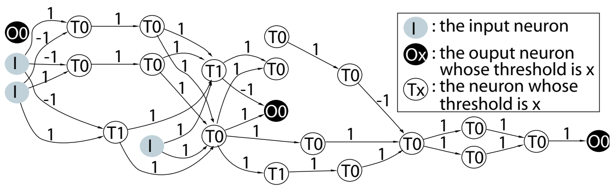
I : the input neuron
Ox : the ouput neuron whose threshold is x
Tx : the neuron whose threshold is x

Figure 3: The best neural network for the run. Please note that the cellular encoding model may produce more inputs and outputs then necessary, as it can be seen in this case, although only one output is actually used. The output node in the center is the used one.

Table 1: Summary for the determination coefficient for different growth models, on the number of nodes trait. R^2, the determination coefficient, indicates how well the model fits the data, with 1 being a perfect fit. The value shown is the average R^2 value among all the individuals in the winning lineage (the lineage of the best individual in the last generation).

Model	Average R^2
Linear regression	0.933
Von Bertalanffy	0.968
Sigmoid	0.974

5 Measuring heterochrony

Next, we analyzed the networks for the following traits: number of nodes, average connectivity (taking into account both incoming and outgoing connections) and how the fitness value changes within ontogeny. Sample growth curves for the best individual on each trait can be seen in figure 4.

One significant problem in using the Alberch *et al*'s framework described before is how to extract significant α, β and K parameters from the growth data. This is also an issue in biological systems although they tend to follow more regular patterns. One common approach is to use non-linear regression to fit the data to a growth model and extract the parameters from the fitted model. This, for instance, was applied by Creighton and Strauss [4] to rodent growth data.

This approach works well for biological systems because their growth dynamics tend to follow regular patterns and there are several sensible mathematical models. These models fit well the data and are grounded on experimental evidence. In contrast, our artificial developmental model can generate rather irregular growth curves — with sudden reverting ontogenic polarity as it can be seen in figure 4 (b) or even with sharp discontinuities as in figure 4 (c).

Another possible approach is simply to extract the values directly from the experimental data, for instance K could be defined as the average growth increment during the development period, or estimated by linear regression. An example of this approach can be seen in Pigliucci [5].

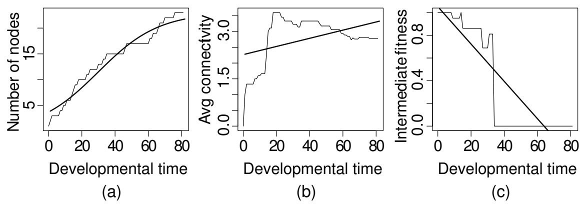

Figure 4: Growth curves for the analyzed traits on the best individual. Both growth data and fitted curves are shown. For the number of nodes, a sigmoidal curve was fitted, while simple linear regression was used for the other traits.

In this paper we decided to use both approaches depending on the trait used. The number of nodes trait seems to follow dynamics similar to biological systems, so we attempted to fit the data to growth models commonly described in the literature. We attempted the Von Bertalanffy's growth function defined as:

$$y = Sa(1 - e^{-k(t-t_0)}), \tag{4}$$

and also the standard sigmoidal function defined as:

$$y = \frac{Sa}{1 + e^{-k(t-t_0)}}. \tag{5}$$

In both functions, Sa stands for the maximum value achieved during growth and k for the speed of growth. We used a Gauss-Newton algorithm for fitting the data, with Sa fixed to the last value reached during development and the other remaining parameters (k, t_0) were initialized to adequate starting values. A summary of the results for the fit can be seen in table 1. The determination coefficient R^2 was the highest on average for the sigmoidal function and therefore we adopted this model . We defined K as the same parameter k in the sigmoidal function, α and β as the period when the model reaches 10% and 90% of the total growth Sa respectively. This approach is coherent with the above mentioned study by Creighton and Strauss [4]. An example of this fitting can be seen in figure 4 (a).

For the other remaining traits, we used simple linear regression and defined K as the slope of the fitted line. This approach was used because the other models assume monotonous increase in the trait while the developmental curves in these traits can decrease. α and β were defined as the period where growth can be observed to effectively start and stop in the data. Examples of these fittings can be seen in figures 4 (b) and 4 (c). This approach cannot be considered completely adequate as the fitting is not sufficient (R^2 is low due to the complexity of the developmental curves in these traits), but nevertheless it allows to have consistent values for the three parameters.

By comparing the parameters between related individuals in the lineage it is therefore possible to classify the heterochrony process involved as depicted in figure 1. Figure 5 shows the transitions of α, β and K for each trait in the winning lineage. In the transitions for the number of nodes trait, we can see the gradual decrease in α and the increase in β with large fluctuations, and that K converged to a small value. We also see transitions similar to the number of nodes in the average degree trait, except for the steady evolution of α, and a quite different evolution of all parameters can be seen for the fitness trait. Figure 6 shows the occurrences of the combined heterochrony processes for the number of nodes trait in the winning lineage. Isomorphosis refers to no change at all in any of the parameters between ancestor and descendant.

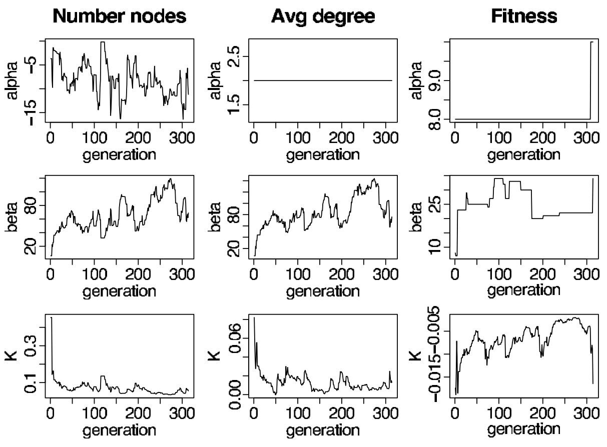

Figure 5: Evolution of the α, β and K parameters for the winning lineage on all traits.

As it can be seen in figure 6, all possible heterochronic changes occur with significant frequencies. One interesting point is that pure recapitulation (isomorphism) seems to be a common heterochronic change in these runs. Even accounting for errors in estimating the parameters, this should be significant considering the low reproduction rate (20%). The fitness function shows sharp discontinuities (as can be seen from figure 2) and therefore it may be exerting selection pressure for either neutral or invalid mutations.

The most frequent heterochronic change is the combination of predisplacement, hypermorphosis and neoteny. The net effect of this combination is that developmental time increases on the descendant. This can also be observed on the gradual increase in the β parameter in figure 5. In the cellular encoding model, developmental time is roughly proportional to the GP tree size; Therefore developmental time is expected to increase, as in GP systems the average tree size gradually increase with evolutionary time (tree bloat).

6 Conclusion

As the use of simulated developmental processes increases, it becomes more important to understand their dynamics. In this paper, we have shown that the framework by Alberch *et al* is a valid method for studying dynamics in artificial systems, by measuring heterochrony in the evolution of neural networks. We successfully observed heterochrony in the number of neurons trait such as the frequent occurrences of the combination of predisplacement, hypermorphosis and neoteny, and also pure recapitulation.

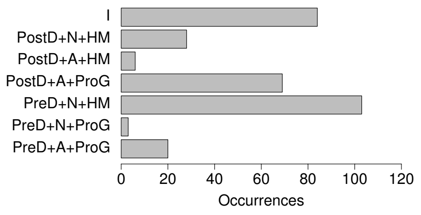

Figure 6: Occurrences of heterochrony for the number of nodes trait on the winning lineage. I - isomorphism; PostD - postdisplacement; PreD - predisplacement; N - neoteny; A - acceleration; HM - hypermorphosis; ProgG - Progenesis.

This framework is particularly well suited for artificial systems because it is solely based on observable quantitative traits and it can be applied regardless of the epigenetic process involved. This should be particularly important when comparing heterochronic change between different developmental models, as for instance, cell chemistry and grammar based models. Future work will therefore continue on extending this analysis for different models and fitness functions.

Acknowledgments

The simulated evolutionary experiments were conducted with the ECJ (Evolutionary Computation in Java) package. I would like to thank both Sean Luke and Gabriel Balan for their helpful assistance with some technical ECJ programming details. This research is funded with a Japanese MEXT scholarship (400000357010989).

References

[1] Frederic Gruau. *Neural Network Synthesis using Cellular Encoding and the Genetic Algorithm.* PhD thesis, Laboratoire de la Informatique du Parallelisme, Ecole Normale Superieure de Lyon, 1994.

[2] Pere Alberch, Stephen Jay Gould, George F. Oster and David B. Wake. Size and shape in ontogeny and phylogeny. *Paleobiology*, 5(3): 296–317, Summer 1979.

[3] Angelo Cangelosi. Heterochrony and adaptation in developing neural networks. In *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, pages 1241–1248, Morgan Kaufmann, 1999.

[4] G. K. Creighton and Richard E. Strauss. Comparative patterns of growth and development in cricetine rodents and the evolution of ontogeny. *Evolution*, 40(1): 94–106, January 1986.

[5] Massimo Pigliucci. Ontogenic phenotypic plasticity during the reproductive phase in arabidopsis thaliana (brassicaceae). *American Journal of Botany*, 84(7): 887–895, July 1984.