



## **The evolution of artificial neurology in ALEPH\_world**

by

Antonia J. Jones and Philip J. Pratt

**Abstract.** Recent advances in the genotype encoding and artificial evolution of artificial neurologies are briefly outlined and described by illustrations from ALEPH\_world.

**Keywords:** Artificial life, Evolutionary algorithms, Cellular encoding.

Correspondence to:

DEPARTMENT OF COMPUTING  
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY  
AND MEDICINE  
UNIVERSITY OF LONDON  
180 Queen's Gate, London, SW7 2BZ  
Telephone: 0-171-594-8350/8370  
Telefax: 0-171-581-8024

Date/version: 14 October 2000

Copyright © 1995. Antonia J. Jones and Philip J. Pratt

# The evolution of artificial neurology in ALEPH\_world

## CONTENTS

Preface .....	1
An evolutionary approach to artificial intelligence .....	1
Genetic encodings of networks .....	2
Cellular encoding systems .....	2
The ALEPH_world simulation .....	4
References .....	6

## List of figures

Figure 1 Animats in a complex virtual world [ALEPH simulator, Pratt 1995]. .....	1
Figure 2 A direct encoding of a 2-bit parity (XOR) network. ....	2
Figure 3 The <b>seq</b> and <b>par</b> divisions. ....	3
Figure 4 An example warmth field. ....	4
Figure 5 The genotype for 'LEFTY', a directionally challenged individual. ....	5
Figure 6 LEFTY's neurology. ....	5

## The evolution of artificial neurology in ALEPH\_world

*Antonia J. Jones and Philip J. Pratt*

*Department of Computing, Imperial College*

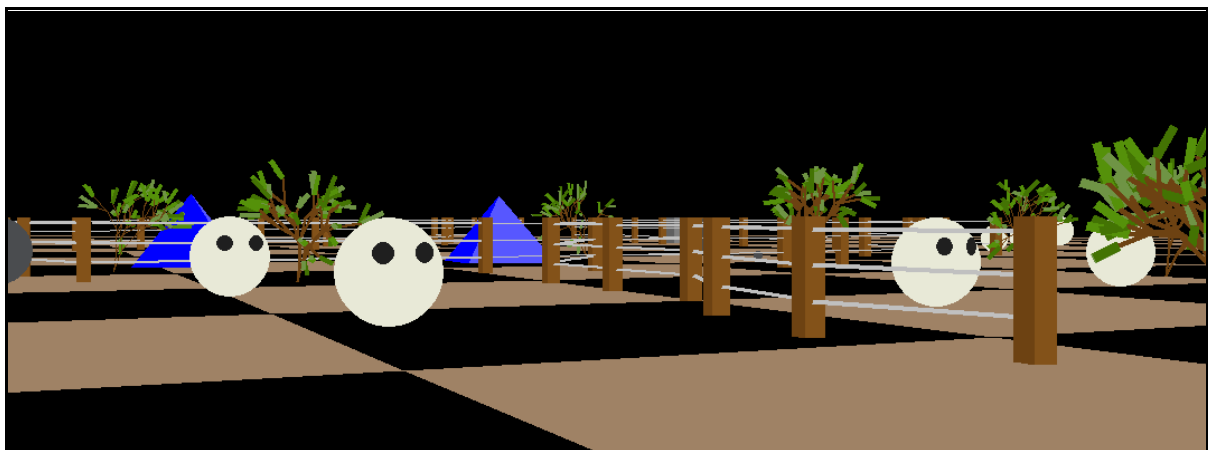
### **Preface.**

One of the key questions facing geneticists and (oddly enough) computer scientists working on artificial life, is precisely how genes encode complex neural structures. Both sets of scientists face different constraints, geneticists seek to unravel a code which describes a growth process of unbelievable complexity and are limited by practical and ethical considerations in their experiments, whereas computer studies in the evolution of complex neurologies require vast computing power not readily available, but have free reign with regard to the genotype design. These two groups are increasingly working together in a fascinating new research area.

### **An evolutionary approach to artificial intelligence.**

In his now famous 1950 paper *Computing Machinery and Intelligence*, Turing suggested an approach to artificial intelligence that was largely ignored for the next 30 years: "Instead of trying to produce a program to simulate the adult mind, why not rather try to produce one which simulates the child's? If this were then subjected to an appropriate course of education one would obtain the adult brain". Turing's so called 'child machine' would have sensorimotor capabilities and exist in a dynamic environment with which it could interact and learn by experience. This approach differed radically from what we now call 'traditional' artificial intelligence, in which specific, and usually quite advanced behaviours are modelled. Such traditional, often explicitly rule based, systems tend to exhibit a non-adaptability or 'brittleness' [Holland 1986], in the sense that they often perform very poorly when exposed to problems differing only slightly from those for which they were designed. How could Turing's 'child machine' be realised? As Turing observed "We cannot expect to find a good child machine at the first attempt". He went on to suggest how one might use an artificial evolutionary process to progressively improve such machines.

More recently considerable interest has developed among the artificial life community in simulations of artificial animals or 'animats'. The animat approach to artificial intelligence echoes Turing's original suggestion, indeed a child machine can be thought of as an advanced form of animat. What is important is the underlying hypothesis that by simulating animat systems holistically at an initially simple level, it might then be possible to work up to more intelligent behaviour as a result of the incremental process of evolution. A suitable definition of 'intelligent behaviour' in this context might be: those behaviours which satisfy an animat's physiological needs to the extent of it being able to reproduce and propagate its genetic material, in a dynamic, unstructured and possibly hostile environment.

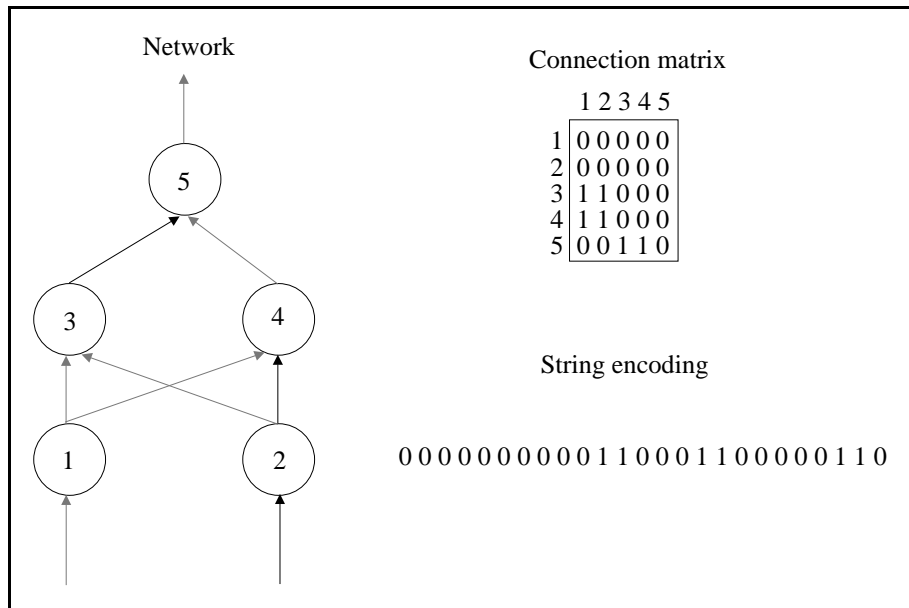


**Figure 1** Animats in a complex virtual world [ALEPH simulator, Pratt 1995].

Within such simulation studies the details of the environment may be very simple, as in Ray's *Tierra* [see, TREE 9(11)], or much more realistic, as in the ALEPH simulation (see Figure 1), but the important aspect is the *processing* done internally by the animats: the means by which an animat integrates environmental and proprioceptory sensory information and bridges the gap between the incoming data and its own effector actions. In real biological animals this processing is effected by adaptive neural circuitry and, although we have a moderately detailed picture of neural activity at the Hodgkins-Huxley level of ion gates and pumping mechanisms, which goes some way to explaining the input-output behaviour of individual neurons, we have almost no understanding of how to hook together large numbers of component neurons to build complex adaptive circuits which will endow the animat with an ability to learn the multi-faceted and coordinated behaviours required to survive. It is at this point that the power of evolution can be brought into play.

**Genetic encodings of networks.**

Early work, see [Jones 1993], on evolving artificial neural networks typically concentrated on evolving feedforward networks which once generated often employed some variation of the backpropagation algorithm for learning the interconnecting weights. The representation of the genome frequently employed was a simple connection matrix, see Figure 2. However, problems rapidly emerged with this naive approach: many different connection matrices could describe



**Figure 2** A direct encoding of a 2-bit parity (XOR) network.

functionally identical networks (this problem was dubbed permutational redundancy), worse it was possible to cross two functionally identical networks and obtain a totally different child network. We know from the work of [Price 1970], [Price 1972] that evolution can only be an effective search procedure if child fitness has a reasonable correlation with parental fitness. Unsurprisingly as the size of the networks, or the problem to be solved, increased the effectiveness of the genetic search based on simple connection matrices progressively decreased.

Whilst simple feedforward neural networks based on backpropagation learning have proved a useful tool for many practical applications in pattern recognition and non-linear parameterisation problems, such models do not begin to describe the complex dynamic behaviour of real neurologies. High dimensional phase portraits [Freeman 1991] made from EEGs generated by computer models reflect the overall activity of the olfactory system of a rabbit at rest and in response to a familiar scent (e.g. banana). The resemblance of these portraits to irregularly shaped, but still structured, coils of wire reveals that brain activity in both conditions is chaotic but that the response to the known stimulus is more ordered, more nearly periodic during perception, than at rest.

**Cellular encoding systems.**

An alternative approach to evolving artificial neural networks which allows more complex circuitry with feedback loops was recently developed by Gruau. *Cellular encoding* [Gruau 1992a]-[Gruau 1994b] is a graph rewriting grammar expressed with a tree-based representation. It is somewhat similar to the *Genetic Programming Paradigm* of Koza's school, see for example [Koza 1992], in that genotypes are recombined by swapping randomly chosen

subtrees between parents, and consequently the size and structure of each tree is not fixed. Cellular encoding has a number of desirable coding properties, including hierarchical organisation, efficiency, closure and modularity.

Gruau uses *pointer cells* to determine which cells in the network become inputs and outputs. In effect, these are dummy cells connected to the initial cell prior to development. Once complete, the inputs and outputs are those cells connected to the input and output pointer cells, respectively.

Trees have one of a number of possible developmental instructions at each node. Network development begins from a single graph node or *cell*, which has a pointer to the root of the tree. This initial cell is placed in a first-in-first-out queue, and the following sequence of operations is performed until the queue becomes empty:

A cell is taken from the head of the queue, and the developmental instruction to which it points is executed;

If the cell pointed to an **end** instruction, it is removed from the queue.

If the instruction caused the cell to divide, then the corresponding tree node has two subtrees and the resulting two child cells are made to point to them, then the child cells are added to the back of the queue.

Thus, the network will grow as a result of a sequence of cellular divisions. The leaves of a tree, called *terminal nodes*, are always **end** instructions. In the basic version of cellular encoding a small number of developmental instructions orchestrate the incremental growth. For example

**seq** Sequential division. The first child cell inherits the inputs from its parent, and the second child cell inherits the outputs from its parent. A new connection is made from the first child to the second with weight 1.0.

**par** Parallel division. Both child cells inherit all inputs and outputs from their parent. If the parent cell has a connection to itself, then both children are made to have connections to themselves and each other.

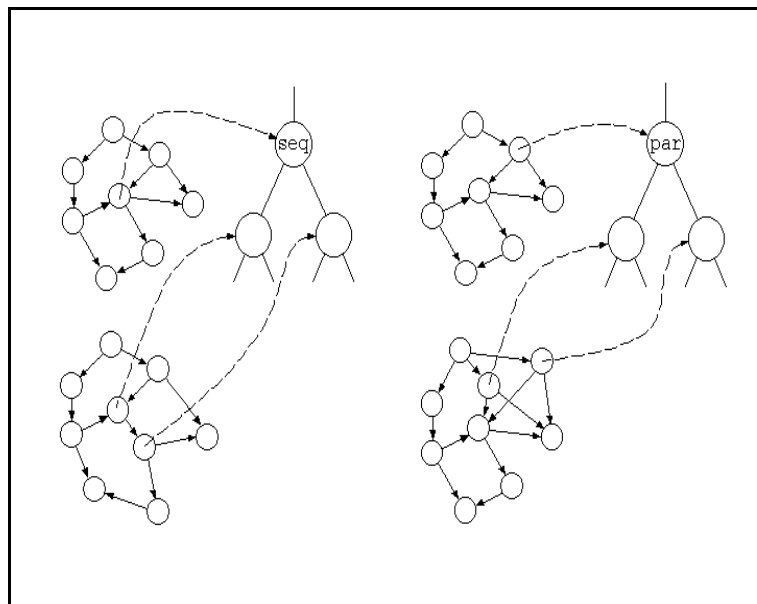


Figure 3 shows partially developed networks and the function of the sequential and parallel divisions. The dashed lines represent pointers from cells to tree nodes.

Figure 3 The seq and par divisions.

Some other basic instructions include: **clone** which is similar to **par** except that both child cells are made to point to the node's only subtree; **end** the cell is removed from the queue; **wait** has one subtree and forces the cell to the back of the queue, delaying its development. Each cell may also have a number of internal registers, whose values are passed on whenever a cellular division occurs and one such register is the link register which points at one of the cells's inputs and is used to reference that input during the execution of another instruction: **inc** and **dec** increment or decrement the link register; the **inclr** and **declr** instructions increment and decrement its position; the **val+** and **val-** instructions set the weight referenced by the link register to +1.0 and -1.0, respectively; **cut** removes the connection altogether; the threshold of a cell is stored in another internal register, and can be modified with the **incbias** and **decbias** instructions. Recursive structure can be introduced with the **rec** instruction. When executed,

the cell is made to point to the root of tree and development continues as normal. After a pre-specified number of recursions, the process terminates.

By introducing a number of new developmental instructions, Gruau [Gruau 1992b] describes a technique for expressing programs, written in a subset of the *PASCAL* language, as neural networks. In short, the parse tree of the program is generated, and each node rewritten as a combination of cellular encoding instructions. Development of the tree will then yield a neural network which performs the same computation as the original program.

Gruau and Whitley [Gruau 1993] investigate the interaction between learning and evolution in the context of evolving Boolean networks to compute parity and symmetry. A Hebbian learning algorithm is used to adjust network weights after development, in an attempt to exploit the *Baldwin effect*. Two other variants are considered, namely *Developmental learning*, in which learnt information can effect subsequent development, and *Lamarkian learning*, in which learnt information is written back to the genotype. Their results suggest that merely using developmental learning to improve evolutionary search may be as effective as a Lamarkian strategy.

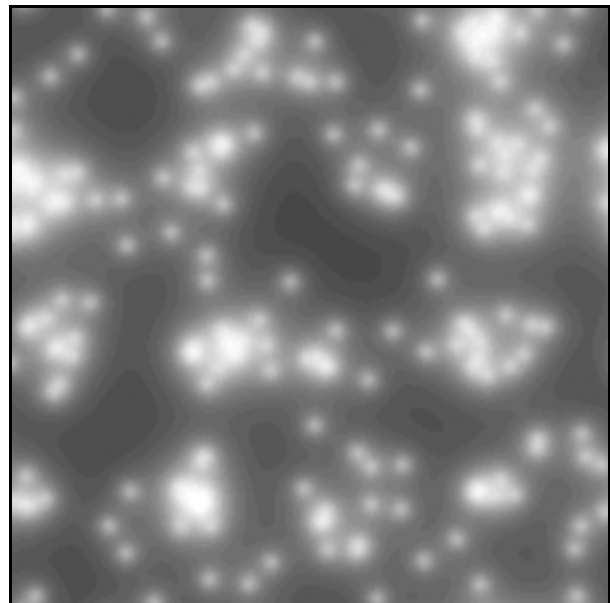
More recently Gruau has added an 'Automatic definition of sub-neural networks (ADSN)' scheme to cellular encoding, identical in operation to Koza's 'Automatic function definition' extension to genetic programming. In comparison with the original cellular encoding, the new scheme is shown to significantly increase the speed of evolution of controllers for the model of six-legged animat locomotion proposed by Beer and Gallagher.

In [Pratt 1994] further developmental instructions are added which enhance the evolution of weights and sensorimotor systems. These facilitated the evolution of systems able to balance broom handles, a favourite test for AI researchers.

#### **The ALEPH\_world simulation.**

The ALEPH simulation was developed in the Neural Systems Engineering Laboratory at Imperial College, see Figure 1, and takes place on the surface of a two dimensional toroidal world. Objects exist in this world and have real valued positions, orientations, masses and energies, and obey some rudimentary 'laws of physics'. Object states are updated at regular discrete time intervals. The simulation strives to be as realistic as possible within the bounds of available computational resources; in addition to conventional workstations, ALEPH has been developed to run on the Fujitsu AP1000 massively parallel supercomputer.

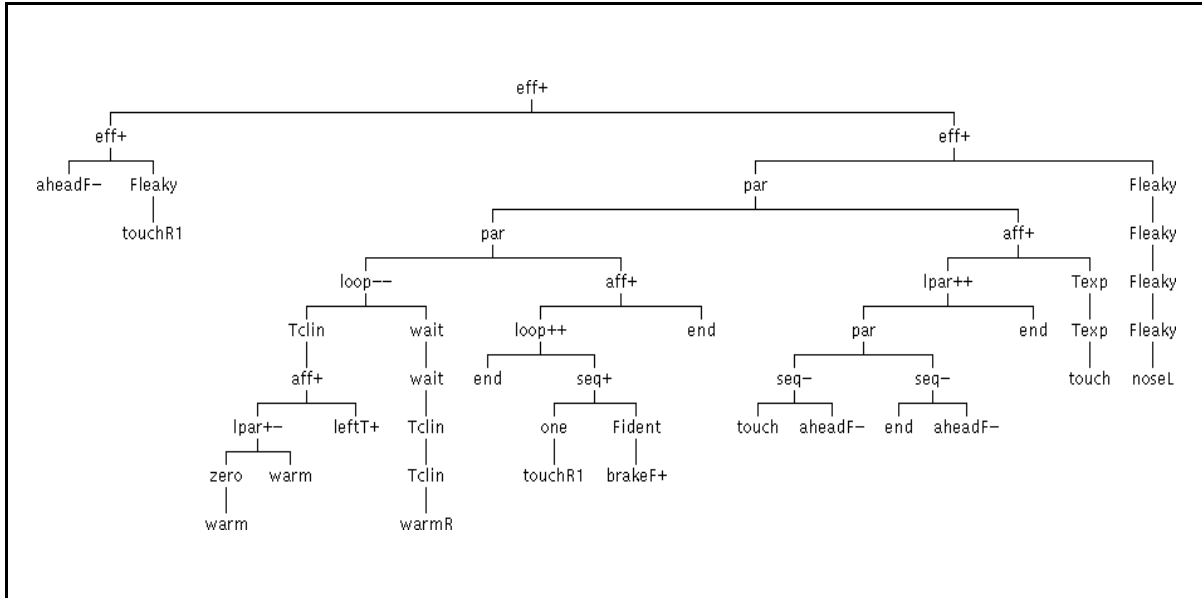
Each object has actual physical dimension, but animat shapes are restricted to spheres of varying radii, in order to speed up collision detection. Full, arbitrary shape, collision detection would be prohibitively slow. Objects can interact with each other and with a number of scalar fields defined over the toroidal surface. In the present version of ALEPH there are three fields, namely *warmth*, illustrated in Figure 4, *sound* and *smell*.



**Figure 4** An example warmth field.

At each time step, the fields are updated by taking scalar contributions from each object. It is then possible to compute the value of a given field at any position, and thus provide objects with appropriate stimuli. *Beacons*, which may be stationary or mobile, are also used to provide strong field sources.

When an animat runs out of energy, or has reached the maximum allowed age, it immediately dies and then gradually decays and eventually vanishes. Other objects in the ALEPH simulation include, *plants*, some of which are edible and provide the animats with an energy source, and *barriers* such as walls and fences.

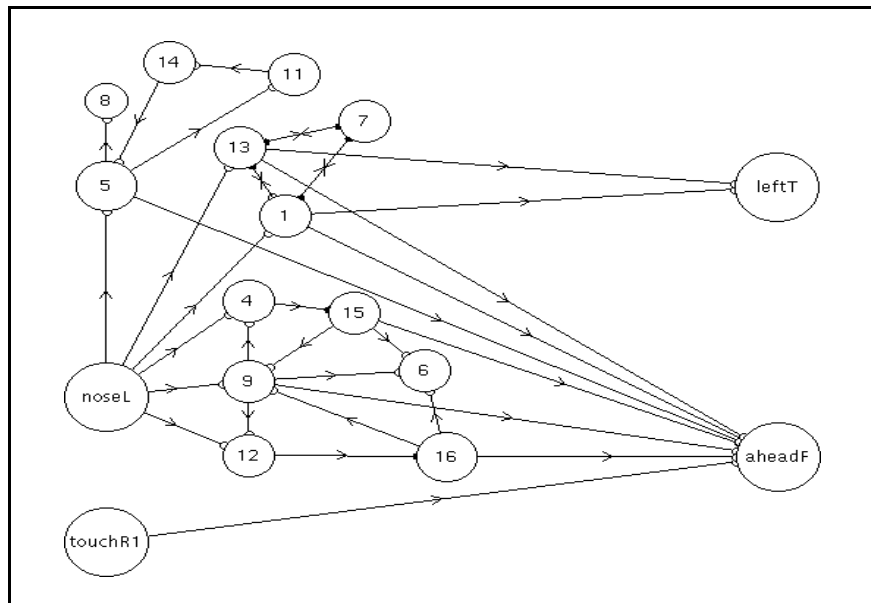


**Figure 5** The genotype for 'LEFTY', a directionally challenged individual.

Animats move and interact with other objects in their vicinity by means of a sensorimotor system, an integral part of their neural networks. Each animat neural system is individual and is defined by a tree structured genotype based on an extended system of Gruau's cellular encoding.

Figure 5, illustrates a typical genotype. This particular individual is not destined for great things, it can only make left turns or move forward! The neural circuit which develops from LEFTY's genotype is shown in Figure 6.

The ALEPH\_world simulator provides various tools for the real world observer, for example to display the genotype, neural circuit or real time 'EEG' of any selected individual.



**Figure 6** LEFTY's neurology.

By progressively making the environment more testing it is possible to evolve individuals with complex neurologies who move, eat, breed via sexual reproduction, and eventually die. Are these animats evolving? If we replace a later population, which can successfully survive a more hostile environment, by a population of distant ancestors, the ancestor population rapidly dies off. Beyond doubt evolutionary adaptation is taking place in the animats of ALEPH\_world. Although these studies are in their infancy, tools such as the ALEPH\_world simulator provide powerful insights into the evolutionary processes which lead to the development of sophisticated neurologies.

It is interesting to speculate on the possibility that animats, endowed with the capability of generating a repertoire of 'sound' in environmental circumstances where cooperation benefits survival, might spontaneously develop the beginnings of language?

**References**

- [Baldwin 1896] J. M. Baldwin. *A new factor in evolution*. American Naturalist **30**:441-451, 1896.
- [Freeman 1991] W. J. Freeman. *The physiology of perception*. Scientific American, pp 34-41, February 1991.
- [Gruau 1992a] F. C. Gruau. *Cellular encoding of genetic neural networks I: The theoretical properties*. Technical report, LIP-IMAG l'Ecole Normal Supérieure de Lyon, 1992.
- [Gruau 1992b] F. C. Gruau. *Cellular encoding of genetic neural networks II: Compilation of computer programs*. Technical report, LIP-IMAG l'Ecole Normal Supérieure de Lyon, 1992.
- [Gruau 1992c] F. C. Gruau. *Genetic synthesis of boolean neural networks with a cell rewriting developmental process*. In Whitley and Schaffer, editors. Proceedings of the international Workshop on Combinations of Genetic Algorithms and Neural Networks, pp 55-74. IEEE Computer Society Press, 1992.
- [Gruau 1993] F. C. Gruau and D. Whitley. *Adding learning to the cellular developmental process. A comparative study*. Evolutionary Computation, 1(3):213-233, 1993.
- [Gruau 1994a] F. C. Gruau. *Automatic definition of sub-neural networks*. Technical report 94-28, Department of Computer Science, Colorado State University, 1994.
- [Gruau 1994b] F. C. Gruau. *Genetic micro-programming of neural networks*. In Kinnear, editor, Advances in genetic programming, pp 495-518. The MIT Press, 1994.
- [Gruau 1994c] F. C. Gruau. *Neural Network synthesis using cellular encoding and the genetic algorithm*. Ph. D. thesis, l'Ecole Normal Supérieure de Lyon, 1994.
- [Holland 1986] J. H. Holland. *Escaping brittleness: the possibilities of general-purpose learning algorithms applied to rule based systems*. In Michalski, Carbonell and Mitchell, editors, Machine Learning, an Artificial Intelligence Approach, **2**:593-623, Morgan Kaufmann, 1986.
- [Jones 1993] Antonia J. Jones. *Genetic Algorithms and their Applications to the Design of Neural Networks*, Neural Computing & Applications, **1**(1):32-45, 1993.
- [Koza 1992] J. L. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Bradford Books, MIT Press, 1992.
- [Pratt 1994] P. J. Pratt. *Evolving neural networks to control unstable dynamical systems*. Proceedings Third Annual Conference on Evolutionary Programming, pp 191-204. Ed. A. V. Sebold and L. J. Fogel, World Scientific, February 1994.
- [Price 1970] G. R. Price. *Selection and covariance*. Nature, **227**:520-521.
- [Price 1972] G. R. Price. *Extension of covariance mathematics*. Annals of Human Genetics **35**:485-489.