PS Einführung in die Bioinformatik (SS99)

Growing Artificial Neural Networks

Wolfgang Klier (wklier@cosy.sbg.ac.at)
Peter Palfrader (ppalfrad@cosy.sbg.ac.at)
Bernhard Stader (bstader@cosy.sbg.ac.at)

Version: June 30, 1999
Gruppe: 6
Zeit: 15:00-15:30
Veranstaltungsleiter: Roland Schwaiger

Abstract

In this document we review the current state of research in the area of artificial neural networks(ANNs) with respect to methods for growing them on a biologically plausible basis.

Contents

Chapter 1

Theoretical Background

According to [3] neural network theory is relatively old – the first publications date back to 1890 – but draws growing attention in recent years in connection with ANNs modeled in computers. Many people think of androides or artificial brains when hearing neural network, but taking into consideration that the human brain consists of $10^{10} - 10^{11}$ neurons, recent research is aimed at much simpler goals. Computers are exact machines what is appreciated in many cases, but for other problems this is not very well suited. ANNs provide a way to cope even with incomplete or inexact data as it is encountered for example in pattern recognition applications. One big problem when working with ANNs is to find the right architecture for a given task. But here nature can help too. Scientists begin to develop methods for growing ANNs on a naturally plausible basis, which will be the main content of this paper.

1.1 Natural and Artificial Neurons

Our nervous system consists of neuron cells which are interconnected via synapses. In [4] the authors name three types of synapses: In most cases synapses are located between a cells axon and another cells dendrites or between an axon and a cells body but there also exist synapses that connect two axons. A nerve cell receives excitatory and inhibitory stimuli which result in one combined stimulus. If the resulting stimulus exceeds a certain value – called the threshold value of this cell – the neuron fires an impulse down its axon, called a spike. If the threshold is not reached, nothing is propagated to the other cells. It is called the all or nothing principle.

The grade of excitement is always the same, only the frequency of spikes may change. This kind of encoding is called pulse-coding. It is clear that you can transmit less information in the same amount of time than you could with an analogous signal for the very obvious reason that you have to receive at least 2 spikes to determine their frequency and as a result the strength of the stimulus while with an analogous signal you had the strength after receiving only one signal. One advantage of pulse coding is security. In [4] the authors mention that investigations revealed that spikes are not always of the exact same strength. If the strength of a stimulus was encoded with an analogous signal small errors multiple times could easily distort the signal. A big advantage of pulse coding is the threshold system, which has already been described earlier in this document. It prevents that small random fluctuations of a cell's potential fire a spike so that you need an actual noticeable stimulus to fire it which means in short that noise is suppressed.

But even without analogous encoding one could think of other methods than pulse coding to propagate the strength of a stimulus, for example a digital encoding ([4]). The duration of a strength transmission would be divided into intervals of similar duration. A spike during an interval would mean a value of 1 while no spike would simply mean 0. To distinguish for example 8 levels of strength one would need 3 signals (000, 001, 010, etc). However this kind of encoding was not found in nature yet and it seems very unlikely that it is actually used. A clear disadvantage of digital encoding is that the more levels you want to distinguish, the more signals you need and so it takes longer to actually transmit the strength of a stimulus.

At an axon's end, at the synapse, the incoming spikes free chemical substances (the more spikes arrive at the end of the axon the more substances are released) So a strong stimulus which produced a high frequency of spikes produces a high concentration of the transmitting substance. When it reaches the postsynaptic membrane it causes the next cell's potential to change and so a forward of information.

1.2 The artificial neuron

Computer scientists try to simulate this behavior in different grades of detail. The basic artificial neuron as described in [3], gets several input values which are added up. These values are often multiplied by so called connection weights (negative weights for inhibitory, positive weights for excitatory connections), which represent the different grades of influence the preceding cells have. This sum can be propagated as is or it must exceed a threshold to propagate a value. A noteworthy fact is that most (but not all) ANNs propagate information as analogous values and not in a pulse coded form which can be found in nature. At any point in this process additional functions(linear, exponential, logarithmic, etc) can be inserted to transform the signal in some way.

According to [5] the original neural network was based on the work of Warren McCulloch and Walter Pitts in 1943. They built up an artificial neuron, whose structure was loosely based on the nerve cell of the brain, i.e. it was based on the theory that a nerve's action potential only fires in an all-or-nothing manner if the threshold for that nerve has been exceeded. Thus it sends an impulse to neurons connected via it's axon. If the neuron fires, it is "on" (i.e. the neuron has an out put of "1"), otherwise it is "off" (has an output of "0"). If the threshold operations are expanded temporally so that the logic signal is represented by a train of binary pulses (this is an analog value) one gets multivalued logic operations. The existence of an analog value allowed that these operations become adaptive using multiplication factors called weights. Because in which way this adaptability is implemented in the ANN, it acts and

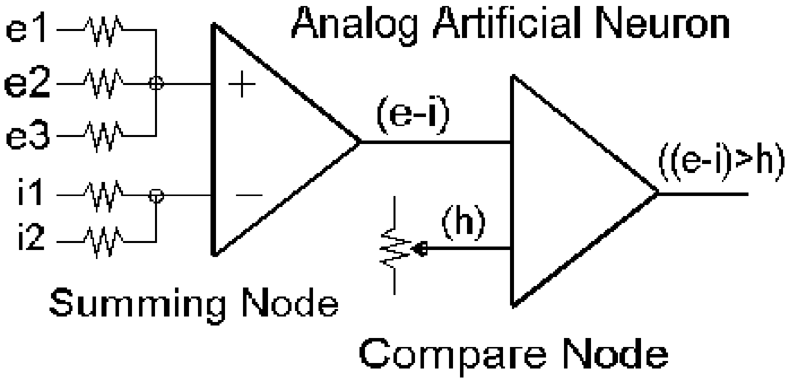

Figure 1.1: Analog Artifical Neuron [5]

reacts very different. This implementation is also important for the training phase. An artificial neuron – shown in figure 1.1 – fires not continously, but at any time t if and only if *(e-i)* exceeds h, where e is the number of excitatory synapses at that time, i is the number of the inhibitory ones and h is the firing threshold for that neuron.

McCulloch and Pitts used three different kinds of neurons: Receptor (input neurons), central or inner neurons, effector (output neurons). The receptor receives the impulse to fire from a sensor. The central neuron connects the receptor and other neurons with the effector and other neurons. The effector receives its impulses directly from the receptor or from the inner neurons.

1.3 Development of the Brain in Vertebrates

As [4] describes, the nervous system develops out of the neural plate in the embryo. The nerve cells are produced in the center of the brain from where they migrate to their final positions. They stick to the Glia-Cells which form long fibers reaching from the inner to the outer surface of the brain or the axons of neurons which are further developed already. The neurons know which way to go due to chemical labels on the surface of other cells. After reaching their final position the neurons begin to grow the dendrites and the axon and they develop the electrical properties needed for propagating stimuli.

Axon growth and especially axon guidance is a very complicated process where some mechanisms are thought to be at work. As mentioned above neurons recognize each other with chemical markers. This could help an axon find its way along axons from related cells. Pioneer cells which take the right place in early stages of development where distances are small could act like a path here too allowing the axon to grow along them.

Another possibility is that axons are guided by chemical gradients. There are substances which attract the axon while others repel it. The tissue or cell with which the neuron should form a connection produces an attractor substance, while other cells in the vicinity produce repulsory substances. To direct an axon over longer distances or along complicated paths guidepost cells are needed. They form a gradient landscape for the axon to grow through.

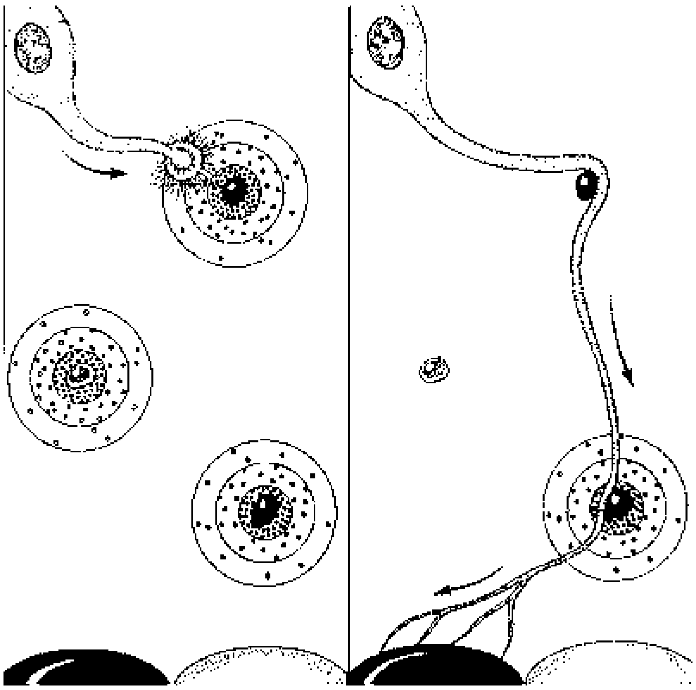

Figure 1.2: Axon Growth

It is believed that neurons can actively change the receptors of the axon for these substances. So they can respond to different molecules at different times, allowing the axon to change direction several times.

Figure 1.2 shows this process. The three small cells are guidepost cells, while the big cells at the bottom belong to the target tissue. The black cells produce attractor substances, the white ones repulsory substances. First the axon grows to the guidepost cell in its vicinity. After reaching it the cell can recognize it as a guidepost cell due to the chemical markers on the surface. So the neuron changes receptors and continues to the next cell without forming a synapse. The repulsory cell on the left ensures that the axon does not grow directly to the target cell but takes the way round the third guidepost cell.

The guidance substances and other substances that influence neuron growth are called neuro-trophic factors. They play a very important role in the following consolidation of the brain. Initially more neurons than needed are produced to allow better adaption to changing environmental influences. The neurons which have not formed enough connections up to this point die making room for more important ones to develop further. This is controlled by the neuro-trophic factors since neurons that have formed many connections receive greater amounts of these substances allowing them to survive.

1.4 Changes to Biological Neurons due to Learning

Learning follows the Hebb rule as mentioned above. In [4] the authors describe two experiments in connection with learning on cell level. In one experiment changes occurred in the synapses leading to a stronger influence of the preceeding cell to the examined one. The other experiment showed a different result. There the observed cell changed its cell membrane but not the synapses. This resulted in easier excitement of the cell by every incoming stimulus.

1.5 Basic Structure of ANNs

A basic neural network consists of input neurons, hidden neurons and output neurons according to [3]. The different types of neurons do not have to be separate units, but one neuron can fulfill several functions. The input neurons have no incoming connections and are turned on and off according to some input pattern. The hidden neurons do the real work. They have incoming and outgoing connections and so all the possibilities of information processing such as input weights, thresholds and other functions can be facilitated. The output neurons main purpose is to represent the networks resulting pattern. Therefore the neurons have no outgoing connections instead their state is read out directly. But the output neurons also do the final processing of the signals from the hidden neurons.

The network can sometimes be dynamic, i.e. it can grow or shrink during learning. Change can take place on several levels:

- at the cell level where the connections between cells develop by growing the dendrites and axons of the artificial neurons. In [1] some experiments simulating this behavior are described. They used chemical gradients in space to determine the direction of growth.

- at the network level where neurons and connections can be inserted as they are needed or the other way around they can be deleted if they are oblivious. [6] describes some methods to shrink ANNs as used in the SNNS[1] .

1.6 Important ANNs

1.6.1 Multi Layer Perceptron

There are many different solutions for ANNs, like the two layer perceptron from Rosenblatt or the the Multi Layer Perceptron (MLP) which is now considered the standard form of ANNs. The MLP consist of a minimum of three different layers, the input-, hidden- and output layer. The MLP has one or more hidden layers between the input and the output layer. If there are more hidden layers the first hidden layer is the input layer of the second and so on.

The two layer perceptron is much like the MLP, but with one big difference. As the name suggests it consists of only two layers: the input and the output layer.

[1]Stuttgarter Neural Net Simulator

1.6.2 Kohonen ANNs

Kohonen Artificial Neural Network (KANNs) is a so called un-supervised training network, which consists of two layers, [9] [11] an input and an output layer, which is often called *Kohonen layer.*. The neurons of a KANN use not a threshold based firing method, but the principle of **the winner takes all** is used. This means that for a certain input vector, there is only one neuron which becomes activated. This neuron is called the *Best Matching Unit* (BMU). To find out which of the input vectors is the BMU, they are compared with a vector. This vector is called *weight vector*, only the neuron which weight vector fits best to the current input vector fires. KANNs are self-organizing networks [12] or self-organizing feature maps (SOFM) [10]. This means that during normal operation the weight factors (or vectors) can be slowly adjusted to the present input patterns. Another important thing about KANNs is their training phase. At the start the weight vector are filled with random vectors. During the training the BMU is determined and the BMU and some neurons surrounding the BMU are adjusted in such a way that the weight vector resembles the input vector more closely. The surrounding neurons are adjusted less strongly, according to the distance they have to the BMU. The distance between these is simply the Euclidean distance[10]. The weight vectors are adjusted by applying the *Kohonen Learning Rule* [9].

1.7 Learning Rules

1.7.1 Hebb Rule

In [7] Hebb proposes a rule now called the Hebb rule. He says that

> When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.

The Hebb rule was one of the first neural network training rules to be used. It is (by todays standards) also one of the simplest to use. The rule is used to alter the weights of a unit in a neural network, so that the actual output of the network matches that of the expected output. This learning rule can be used for many topologies (types, layouts) of networks, including single-layer networks and especially networks which learn by un-supervised methods (where no desired output is given).

1.7.2 Kohonen Learning Rule

$\omega(t+1) = \omega_r(t) + \varepsilon(t) \cdot \Phi_{rs}(t) \cdot (\chi(t) - \omega_r(t))$ $\omega_r(t)$ is the weight vector of a neuron at time t, $\chi(t)$ is the input vector at time t and so $\chi(t) - \omega_r(t)$ is the difference between the input vector and the weight vector of neuron r. [9]

1.8 Evolutionary Algorithms (EAs)

Evolutionary Algorithm (EAs) is an umbrella term to describe computer-based problem solving systems which use computational models of some of the known mechanisms evolution as key elements in their design and implementation. Many different EAs have been proposed. They all share a common conceptual base of simulating the evolution of individual structures via processes of selection, mutation and reproduction , says the author of [14].

The major ones are

- GENETIC ALGORITHMS
- EVOLUTIONARY PROGRAMMING
- EVOLUTION STRATEGIES
- CLASSIFIER SYSTEMS
- GENETIC PROGRAMMING

1.8.1 Genetic Algorithms

As the authors describe in [8] Genetic Algorithms (GA) are based on the Darwinian principle of natural selection ("Survival of the Fittest"). The idea is to evolve good solutions to a real-world problem.

GAs require a suitable coding of the problem. The parameters ("genes") are joined together. This string of values (the "chromosome") represent one possible solution to the problem. This solution is then evaluated (by the "fitness function") and the chromosome gets associated with this fitness value. In the following reproduction phase each chromosome gets a certain chance – determined by its fitness value – to reproduce. The chromosome then either gets copied directly into the next generation ("asexual reproduction") or gets combined with some other chromosome to form new children ("sexual reproduction"). In the next step some mutation takes place, where some genes are altered at random.

GAs work with a population of solutions. The process described above is run many times. If the GA is correctly used the population should then evolve toward some uniform goal. When the whole population has converged to some point a good if not the best solution to the problem should be reached.

Chapter 2

The Implementation

2.1 Introduction

The goal is to write a program similar to Vaario's in [1]: Some simulated creatures populate the artificial world and fight for survival. Each creature has an artificial brain which processes the incoming information and controls its bahaviour. This brain is modelled as an ANN with a fixed number of neurons, input cells and output cells but the connections between them grow on a biological plausible basis.

2.2 External World

The external world is a bounded plane. Obstacles are spread out over the world. When a creature crashes into an obstacle or into the boundary it receives a penalty. On the other hand food is initially placed on the plane. If a creature "eats" some food it gets a bonus.

2.3 Connection between External and Internal World

Each creature has a fixed set of sensors for food detection("Eyes") and obstacle detection("Whiskers"). The strength of a stimulus from a food detecting sensor depends on the distance and on the angle to the target food. Obstacle detection works a bit different: Each sensor has an area of influence and the strength of a stimulus depends on how much of this area is occupied by obstacles similar to Vario's approach in [1].

2.3.1 Sensors

The two types of sensors are modelled similarly. Both check the squares in front of the creature by calculating concentric arches as shown in the pictures below. Only the action on finding something differs. The "eyes" (see figure 2.1) search for the closest food and return a stimilus depending on the distance to this food. The "whiskers" (figure 2.2) simply count how many of the squares in their range are blocked by obstacles. Their resulting stimulus is dependend on the ratio of blocked to non blocked squares.

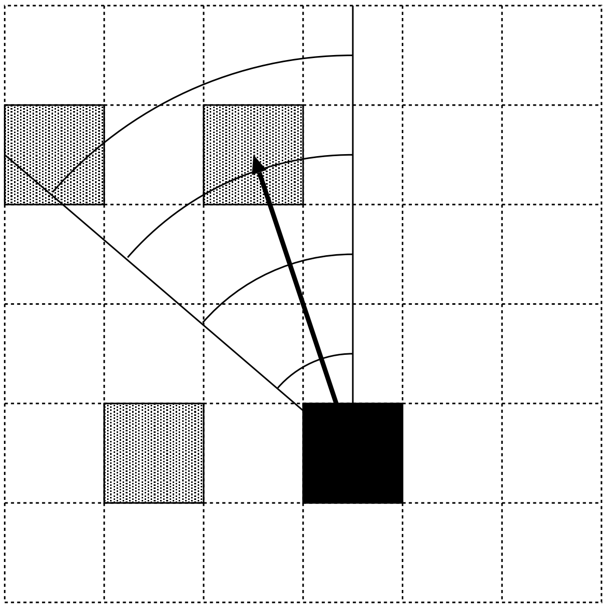

Figure 2.1: An "eye" searches the nearest food square.

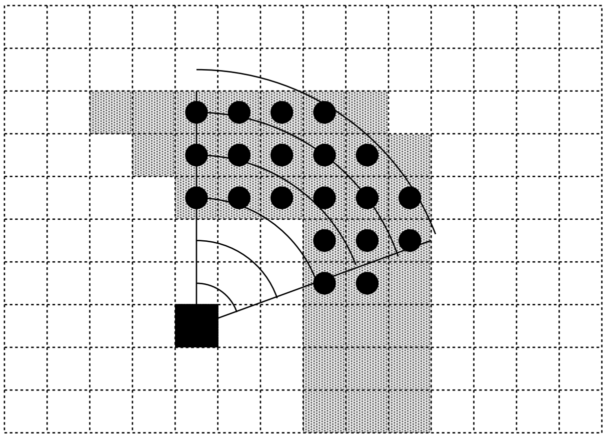

Figure 2.2: A "whisker" counts the number of blocked squares

2.3.2 Movement

To move in the virtual world each creature has two wheels. Velocity and change of direction of a creature are computed based on the strength of the signals both wheels receive. See figure 2.3.2 for further explanation of the following formulae.

If the way both wheels move is the same ($v_r = v_l$), then

$$\begin{aligned}\Delta_\alpha &= 0\\ \Delta_y &= v_r = v_l\\ \Delta_x &= 0\end{aligned}$$

otherwise

$$\begin{aligned}\Delta_\alpha &= \frac{v_l - v_r}{d}\\ r &= \frac{v_r}{\Delta_\alpha} + \frac{d}{2}\\ \Delta_y &= r \cdot \sin(\Delta_\alpha)\\ \Delta_x &= r \cdot (\cos(\Delta_\alpha) - 1)\end{aligned}$$

where d is the constant distance from one wheel to the other, v_r and v_l is the way each wheel moves, Δ_α is the change of direction of the creature. Δ_x and Δ_y represent the change of position of this creature in this turn.

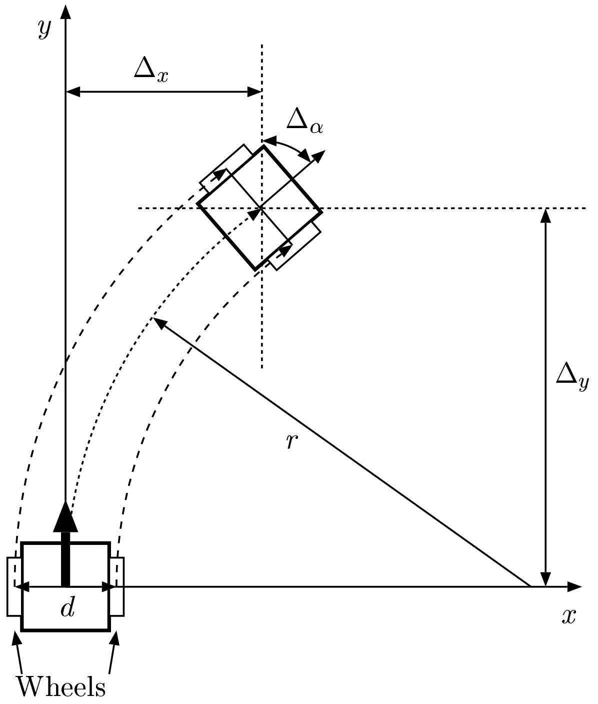

Figure 2.3: $\Delta_x, \Delta_y.\Delta_\alpha$ are calculated from the speed of the wheels as shown above

And finally

$$\begin{aligned}\Delta_X &= \cos(\alpha) \cdot \Delta_y + \sin(\alpha) \cdot \Delta_x\\ \Delta_Y &= \cos(\alpha) \cdot \Delta_x - \sin(\alpha) \cdot \Delta_y\end{aligned}$$

where Δ_X and Δ_Y are the distances the creature moved in the world with it's original heading of α.

2.4 Internal Word

The internal world is a cube with the sensors and wheels on fixed positions. Several neurons and guidance cells are scattered in space. From sensors and neurons axons grow guided by "chemical" gradients. Every neuron, guidance cell, and wheel produces one of several substances (we use 8 different substances) that may attract axons. Each axon has a list of substances (in our program 4) which it follows consecutively. When the concentration of the substance the axon is following exceeds a certain threshold the axon "switches" to the next substance in it's list. After reaching the treshold of the last substance in the list the axon stops growing.

After the connections have grown during development of the phenotype the "brain" is abstracted to an ANN. The length of a connection in the brain and the distance of an axon's end to a cell determine a connection weight for the ANN.

2.5 Growth

2.5.1 Building the concentration grid

Since the axon's growth is guided by chemical gradients we have to built a "concentration grid" first in which the gradients for the various chemicals are stored. A grid cell's concentration value is determined by the formula

$$C_{x,y,z,s} = \sum_{\varphi \in \Phi_s} d_{\varphi,x,y,z} \cdot \varphi_c \tag{2.1}$$

$$d_{\varphi,x,y,z} = \begin{cases} \dfrac{1}{\sqrt{(\varphi_x - x)^2 + (\varphi_y - y)^2 + (\varphi_z - z)^2}} & \text{if } \sqrt{\cdots} \neq 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.2}$$

where x, y, z are the grid cell's coordinates, Φ_s is the set of all axon attracting cells which send out the substance s and φ_x, φ_y, and φ_z are φ's coordinates. φ_c is the initial concentration of the substance the cell produces. If $(\varphi_x - x)^2 + (\varphi_y - y)^2 + (\varphi_z - z)^2$ equals 0 then the value of the fraction becomes 1.

2.5.2 Axongrowth

The axonheads get a chance to grow consecutively. For each directly adjacent grid cell that is not yet occupied by an axon the chance that the axon grows into this grid cell is calculated with the formula

$$P_{x,y,z} = \frac{(C_{x,y,z,\psi_s} - \min C_{\Theta,\psi_s})^{\psi_\eta}}{\sum\limits_{\theta \in \Theta} (C_{\theta_x,\theta_y,\theta_z,\psi_s} - \min C_{\Theta,\psi_s})^{\psi_\eta}} \cdot \left(p_b - \frac{1}{e^{\sum_{\theta \in \Theta} C_{\theta_x,\theta_y,\theta_z,\psi_s}}} \right) \tag{2.3}$$

where $x, y.z$ are the coordinates of the grid cell in question. C_{x,y,z,ψ_s} is the concentration of the substance the axon ψ is following at this time. Θ is the set of

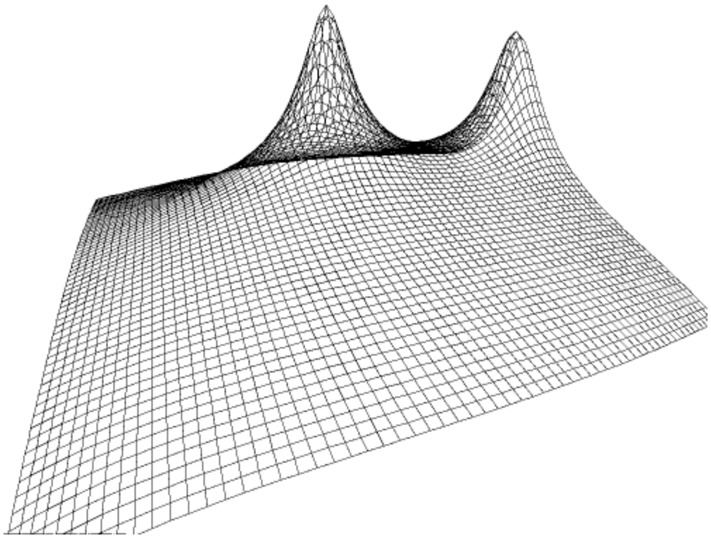

Figure 2.4: The concentration landscape of one plane in the cube. In this example the concentration is made up of two substance-emmitting cells.

all directly adjacent grid cells not occupied, and $\min C_{\Theta,\psi_s}$ is the smallest concentration in Θ. ψ_η is the axon's connectivity parameter, and p_b determines the minimum probability a cell has to grow which is 1.7 in our simulation. The right fraction is an exponential function since the average concentration in our grid is quite low (< 0.1) and the axons would grow very slowly using other functions.

The first part of formula 2.3 calculates the normalized, relative concentrations of the adjacent grid cells to determine the probabilities that the axon is growing in a certain direction. The second part modifies these values in respect to the absolute amount of the substance ψ_s. If this part becomes greater one it is assumed to be one.

If the concentration of the substance the axon follows exceeds a certain inherited threshold the axon changes to the next substance in an also inherited list.

The growth terminates when the axon did not grow for a certain number of turns. This number is the same for all the axons of one cell and is stored in the cell's genome.

2.5.3 Building up the ANN

After all axons finished growing the connections grown are abstracted into weights useable in an ANN. A cell's s influence $I_{s,d}$ on another one d is computed with

$$I_{s,d} = \sum_{\psi \in \Psi_{s,d}} (-1)^{\psi_\kappa} \cdot (\chi \cdot \rho) \tag{2.4}$$

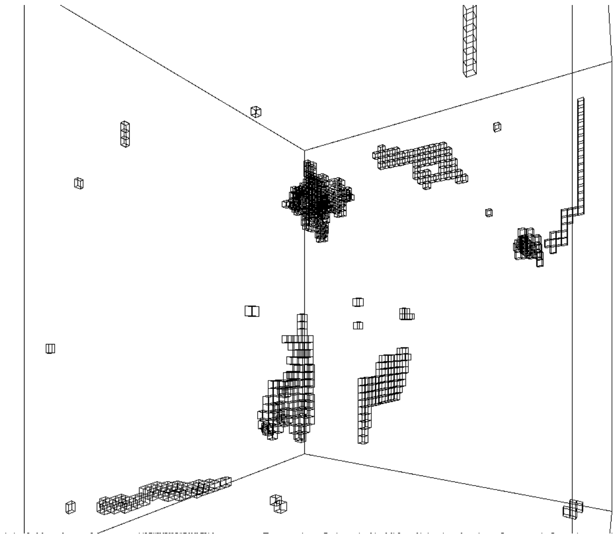

Figure 2.5: A "brain" after several turns of growth.

with

$$\chi = \begin{cases} 1 - \overline{s\psi} \cdot \alpha & \text{if } \overline{s\psi} \cdot \alpha \leq 1, \\ 0 & \text{otherwise.} \end{cases} \tag{2.5}$$

$$\rho = \frac{1}{\sqrt{(d_x - \psi_x)^2 + (d_y - \psi_y)^2 + (d_z - \psi_z)^2}^{\gamma} \cdot \beta} \tag{2.6}$$

where $\Psi_{s,d}$ is the set of all axonheads from cell s that are in d's vicinity, ψ_κ is either 0 or 1 depending on whether the synapse is inhibitory or excitatory, $\overline{s\psi}$ is the length of the axon, and α, β, and γ are constant distance penalty modifiers (we use 0.01, 1.0, and 3.0 respectively).

Those weights are then stored as a graph in an adjacency matrix.

Incoming stimuli are added up at neurons and if the sum does exceed a certain threshold it propagates a stimulus of a ceratin strength which is independent from the incoming stimuli. If the threshold is not exceeded the level of excitement is stored in the neuron and added to the incoming stimuli in the next cycle. (Pulse coded)

In one clock cycle information is only propagated from one cell to its direct successors. So it is no problem if a recurrent network develops.

2.6 Genetic Coding

The genome is stored in a directly encoded bitstring.
The following data is stored for every axon-growing cell (sensors, neurons):

- Substances which the axon follows and threshold values from which concentration on this substance is ignored.

- A propability whether this cell has an inhibitory or excitatory influence on other cells.
- Connectivity parameter: the degree of "hairiness" of axons.
- A number determining after which number of inactive turns this cell's axons stop to grow.

Data for axon attracting cells (guidance cells, neurons, wheels):

- A substance to produce and its initial concentration.

Data for free placed cells (guidance cells, neurons):

- X, Y, Z coordinates which define the position of the cell within the cube.

Additional data for neurons

- The threshold value which determines the strength a stimulus must exceed in order to be propagated

Our implementation of this together with the values' ranges is shown in table 2.1.

2.7 Evolution and Fitness

Every creature has a certain amount of energy which is reduced every turn. If the creature finds some food it gets extra energy; if it bumps into an obstacle it is injured and so loses some energy. As soon as a creature's energy gets less or eqal zero it dies and so is removed from the world. Since amount of food available decreases with time all the creatures finally die.

$$P_n = \sqrt[3]{\frac{10}{(n+10)}} \tag{2.7}$$

We are choosing the mating partners with a roulette selection mechanism. An agent with a high fitness (i.e. it survived for a long time) has a greater probability to get selected. Function 2.7, as can seen in figure 2.6, determines – after normalizing the sum of all P_i to 1 – the probabilities a certain creature is selected. In this formula n is 1 for the fittest agent, 2 for the second, etc. . After selecting two creatures we derive two offsprings using crossover and bitflip mutation operations each with a certain propability. We use 0.6 as a crossover probality and $1/l$ where l is the length of the genotype as a bitflip probability.

Celltype	No. Cells	Part		Feature	No. Bits	Range
Sensors	4	Axongrowing	4×	Following Substance	3	0 - 7
				Threshold for Substance	8	0 - 2.0
			1×	Inhibitory Probability	8	0 - 1.0
			1×	Connectivity	7	0 - 12.7
			1×	Terminate Growth	2	0 - 3
Wheels	2	Axonattracting	1×	Substance	3	0 - 7
			1×	Initial Concentration	7	0 - 1.0
Neurons	10	Axongrowing	4×	Following Substance	3	0 - 7
				Threshold for Substance	8	0 - 2.0
			1×	Inhibitory Probability	8	0 - 1.0
			1×	Connectivity	7	0 - 12.7
			1×	Terminate Growth	2	0 - 3
		Axonattracting	1×	Substance	3	0 - 7
			1×	Initial Concentration	7	0 - 1.0
		Position	1×	X Coordinate	6	0 - 63
			1×	Y Coordinate	6	0 - 63
			1×	Z Coordinate	6	0 - 63
		Threshold	1×	Threshold Value	8	0 - 1.0
Guidance Cells	10	Axonattracting	1×	Substance	3	0 - 7
			1×	Initial Concentration	7	0 - 1.0
		Position	1×	X Coordinate	6	0 - 63
			1×	Y Coordinate	6	0 - 63
			1×	Z Coordinate	6	0 - 63

Table 2.1: Our genotype.

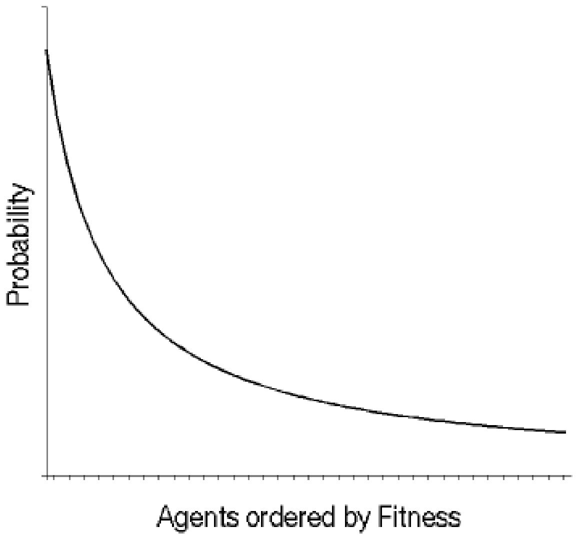

Figure 2.6: the probability distribution

Appendix A

Glossary

connection weights CWs control the influence of the neurons on each other.

Euclidean Distance $\sqrt{\sum_{i=0}^{n} (weight_i - input_i)^2}$

hidden layer Group of artificial neurons which are not accessible directly but propagate signals through the ANN.

input layer Group of artificial neurons which normally have no incoming connections but are activated externally.

neuron, natural special kind of cell able to propagate and process electric signals. They form the nervous system in animals.

neuron, artificial processing unit in an ANN, also called a node.

node see: *neuron, artificial*

output layer Group of artificial neurons which normally have no outgoing connections but represent the output of the ANN.

pattern A set of values which represent the input or output of an ANN.

synapses connection between two natural neurons.

threshold A value which has to be exceeded to generate some action.

Protocol

Compiled: June 30, 1999

1999-Mar-09

Location: Room Nr.: 2.30
Time: 15:00 - 15:30
Participants: Peter Palfrader, Bernhard Stader
Protocol: Peter Palfrader
Preliminary discussion
Tasks:

- General outline of this paper
- Familiarize with the topic

1999-Mar-16

Location: Room Nr.: 2.30
Time: 15:00 - 15:30
Participants: Wolfgang Klier, Peter Palfrader, Bernhard Stader
Protocol: Berhard Stader
Discussing current paper version.
Tasks:

- Connection of neurons in nature (graphic)
- Growing of connections in nature
- Construction of connection in ANNs
- Make a glossary

1999-Mar-23

Location: Room Nr.: 2.30
Time: 15:00 - 15:30
Participants: Wolfgang Klier, Peter Palfrader, Bernhard Stader
Protocol: Wolfgang Klier
Discussing current paper version
Tasks:

- Corrections
- Rethink paper structure
- Add: encoding of stimulus strength in natural neurons
- Add: Hebb-rule nets
- Indepth discussion of methods used to grow ANNs
- Search the Net for methods of ANN construction

1999-Apr-13

Location: Room Nr.: 2.30
Time: 15:00 - 15:30
Participants: Wolfgang Klier, Peter Palfrader, Bernhard Stader
Protocol: Bernhard Stader
Discussing current paper version
Tasks:

- Corrections
- Add: Genetic Algorithms

1999-May-04

Location: Room Nr.: 2.30
Time: 15:00 - 15:30
Participants: Wolfgang Klier, Peter Palfrader
Protocol: Wolfgang Klier
Discussing project proposal.
Meetings from now on every 2 weeks.

1999-May-18

Location: Room Nr.: 2.30
Time: 15:00 - 15:30
Participants: Wolfgang Klier, Peter Palfrader
Protocol: Peter Palfrader
Discussing internal world affairs.

1999-Jun-1

Location: Room Nr.: 2.30
Time: 15:00 - 15:30
Participants: Peter Palfrader, Bernhard Stader
Protocol: Bernhard Stader
Discussing internal world affairs.

Discussing visualization of internal and external world.

1999-Jun-15

Location: Room Nr.: 2.30
Time: 15:00 - 15:30
Participants: Wolfgang Klier, Peter Palfrader, Bernhard Stader
Protocol: Peter Palfrader
Discussing external world affairs.
Discussing sensor types.
Discussing nomenclature.
Tasks:

- Change nomenclature.

1999-Jun-29

Location: Room Nr.: 2.30
Time: 15:00 - 15:30
Participants: Wolfgang Klier, Peter Palfrader, Bernhard Stader
Protocol: Peter Palfrader
Discussing external world vizualization.
Discussing sensor types.
Agreeing on nomenclature.
Discussing further work on the project.

Bibliography

[1] Jari Vaario, Akira Onisuka and Katuso Shimohara. *Formation of Neural Structures.* Technical Paper, ATR Human Information Processing Laboratories.

[2] Dietmar Heinke and Fred H. Hamker. *Comparing Neural Networks: A Benchmark on Growing Neural Gas, Growing Cell Structures, and Fuzzy ARTMAP.* TU Ilmenau 1998.

[3] David D. Olmsted. *History and Principles of Neural Networks.* http://www.neurocomputing.org/history.htm.

[4] Hermann Haken, Maria Haken-Krell. *Entstehung von biologischer Infomation und Ordnung.* Wissenschaftliche Buchgesellschaft Darmstadt 1995.

[5] Human Brain Project. *Consciousness: The Development of Neural Networks.* http://sig.biostr.washington.edu/projects/brain/

[6] Tobias Schreiner. *Ausdünnungsverfahren für Neuronale Netze.* University of Stuttgart 1994.

[7] Donald Hebb. *The Organization of behaviour.* John Wiley 1949.

[8] David Beasley, David R. Bull, Ralph R. Martin. *An Overview of Genetic Algorithms: Part 1, Fundamentals.* University Computing 1993, 15(2) 58-69.

[9] Louis Nagtetaal *Concept-Formation with Kohonen Feature Maps* http://earth.let.uva.nl/ louis/research1/research1.html.

[10] Kohonen Artificial Neural Networks*KANNs* http://gepasi.dbs.aber.ac.uk/roy/koho/kohonen.htm

[11] KANNs *Kohonen Networks* http://ppl.mines.colorado.edu/bill/bak/subsection3_6_1.html

[12] The Kohonen self-organising network *Self-organisation* http://www.hooghen-windt.nl/english/pages/nn_3.htm

[13] Artificial Neural Networks *Hebb Rule* http://home.clara.net/mll/ai/nn/nn2.htm

[14] What are EAs? *A short description* http://209.182.13.227/related.html