Salzburg Interest Group on Integrated Systems

siGis-00x-2000 April 12, 2000

Project Ideas

Proposal

Roland Schwaiger
rschwaig@cosy.sbg.ac.at

Department of Computer Science
University of Salzburg

siGis
Salzburg Interest Group on Integrated Systems

Correspondence to:

Universität Salzburg
Institut für Computerwissenschaften und Systemanalyse
Jakob–Haringer–Straße 2
A–5020 Salzburg
Austria

Project Ideas

Roland Schwaiger
Department of Computer Science
University of Salzburg
A–5020 Salzburg, Austria
rschwaig@cosy.sbg.ac.at

1 Evaluation of netJEN (BioInf)

netJEN is a software tool used for evolution of artificial neural networks (ANN) by means of genetic algorithms (GA). Furthermore coevolution of training data sets (TDS) is implemented. Your task comprises the evaluation of

1. performance of the TDS evolution
2. verifying the distribution of the evolved data in the TDS, i.e. given two examples – unit square and spirale – what are the data points selected for learning?
3. end user test of the netJEN system.

Optional tasks are

1. encoding of the training data
2. investigation of n-dimensional encoding.

So the basic setup consists of

1. netJEN: where is it installed, how to use, any user docu
2. BugReport tool: how to use, where installed
3. test problems: reference, where to find, how to encode

2 GO with netJEN (BioInf)

Some prior project groups have implemented the game of GO with the former netGEN system. Based on their results, which are by the way available (CD-ROM, paper, ...), your task is to improve the encoding of the board and checker positions. A starting point for encoding the whole scene is a one to one mapping of the positions of the checkers and as output again use a one to one mapping of the future position of the checker. As can be seen easily this is a highly inefficient approach, so what's better? Some ideas are,

1. extraction and encoding of features (grouping, ...)
2. multi-resolutional encoding
3. transformation in a different domain
4. ...

We will use a 9×9 practice board for evaluation of our approaches.

3 Evolution of transfer functions (BioInf)

A quite new research direction, in the area of evolution of ANN by means of GA, focuses on the evolution of socalled transfer functions (TF). TF are responsible for the output of the single units in an ANN, the neurons, and as a whole for the overall output of the ANN. Till now primarily sigmoidal TFs were used partially because the most prominent learning algorithm – error back-propagation – requires differentiable TFs. In our setup ANNs are learned by evolution, i.e. weights are manipulated based on predefinded fitness functions. Using the evolutionary approach no differntiability of TFs is required. Your taks will be to investigate various TF types, especially the sin function, compared to standard TFs.

A paper will be provided by Schwaiger Roland.

4 Visualization of ANNs (PPKT)

As a result of the evolutionary cycle, netJEN saves the best ANNs to hard disk in a predefined format – StuttgarterNeuronaleNetzeSimulator (SNNS) file format. The number of ANNs to be saved for later processing may be set by the experimenter. Thus your taks will be to

1. show thumbnail images of the ANNs
2. display single networks
3. arrange ANN elements (neurons, connections, weights, ...) based on user requirements, such that optimal layout is guaranteed.

Especially the last subtask requires literature search in the area of graph theory and chip layout.

The SNNS file format is given below showing an example for an XOR network

```
SNNS network definition file V1.4-3D
generated at Mon Apr 25 15:58:28 1994

network name : xor
source files :
no. of units : 4
no. of connections : 5
no. of unit types : 2
no. of site types : 2


learning function : Quickprop
update function   : Topological_Order


site definition section :

 site name | site function
-----------|------------------
 excite    | Site_WeightedSum
 inhibit   | Site_Pi
-----------|------------------


type definition section :

name           | act func     | out func     |sites
---------------|--------------|--------------|---------
LongeroutType  | Act_Logistic | Out_Identity |
outType        | Act_Logistic | Out_Identity |
---------------|--------------|--------------|---------


unit default section :

act      | bias     | st | subnet | layer | act func     | out func
---------|----------|----|--------|-------|--------------|-------------
 0.00000 |  0.00000 | h  |      0 |     1 | Act_Logistic | Out_Identity
---------|----------|----|--------|-------|--------------|-------------


unit definition section :
```

```
no. | typeName       | unitName | act      | bias     | st | position | act func | out func | sites
----|----------------|----------|----------|----------|----|----------|----------|----------|---------
  1 |                | in_1     |  1.00000 |  0.00000 | i  | 3,5,0    |||
  2 |                | in_2     |  1.00000 |  0.00000 | i  | 9,5,0    |||
  3 |                | hidden   |  0.04728 | -3.08885 | h  | 6,3,0    |||
  4 |                | result   |  0.10377 | -2.54932 | o  | 6,0,0    |||
----|----------------|----------|----------|----------|----|----------|----------|----------|---------


connection definition section :

target | site    | source:weight
-------|---------|------------------------------------------------------------------------------------------------------------
     3 |         | 2:-4.83963, 1: 4.92521
     4 |         | 3:11.11523, 2: 4.53903, 1:-4.67122
-------|---------|------------------------------------------------------------------------------------------------------------
```

5 Visualization of netJEN problem domains and results (PPKT)

netJEN is applied to a variety of problem domains, ranging from stock exchange forecasts to classification of satellite images to extracted rules of the ANNs. Your task is to visualize

1. the problem domain, i.e. a satellite image
2. the result of the netJEN system

You will have to find and classify the possible application areas of ANNs. Based on your findings you will plan your software system ensuring to cover a broad spectrum of visualization the found domains. Some examples of problem domains, encoding in SNNS format and result files are available under

http://www.cosy.sbg.ac.at/ rschwaig/teaching/SNNSexamples/

6 Optimizing object groupings (PPKT)

You will use a GA to optimize groupings of objects $o_1, \ldots, o_n$ in a two dimensional area with respect to a given fitness function. The various objects o_i will be asigned values in a predefined interval $[l_i, u_i]$ measured on a discrete time interval, e.g. on a daily basis. The two dimensional area is discretized, where the single patches are

1. forbidden
2. belong to a predefined group
3. occupied or not by an object o_i

Your task will be to investigate the problem theoretically – is it possible to categorize this kind of problem – and implement a GUI and the GA. As a result you will give suggestions for the groupings in the area based on an overall fitness function, i.e. maximize assigned values of the objects with respect to an underlying model of the value assignment.

Basically the system consists of two components.

- The first component is used to simulate the agents within the world $\mathcal{W}$, with dimensions $w_x \in 1, \ldots, n$ and $w_y \in 1, \ldots, m$, where agents and objects coexist. The world is bounded by walls, where some spots are left open for exits and entrances. Agents $a_i \in \mathcal{A}$ may enter and leave the world according to some underlying distribution and the number of agents populating the world has an upper bound. Each agent a_i occupies a certain spot in the world which

  is not occupied by another agent,

  is not forbidden

  is not occupied by an object.

Each agent has assigned a certain value v_i^a where the value is reduced if the agent resides in front of an object. The amount deducted from the agents value depends on the object the agent is visiting and is denoted by c_i^o. The agents $\mathcal{A}$ may wander in the world randomly or according to a predefined or evolved scheme. Possibly trying to visit all objects with a $v_i^a > 0$.

The agents may wander and visit objects for $\mathcal{T}$ time units. We may call it a dynamic generational based population. Dynamic because agents are allowed to enter and leave the world and generational because after $\mathcal{T}$ time units another generation of agents will populate the world. The update of the simulation should be synchronous, i.e. let the generation be given at time t then the update for time $t+1$ depends solely on time step t. All collisions with other agents, etc. are calculated based on time step t[1]

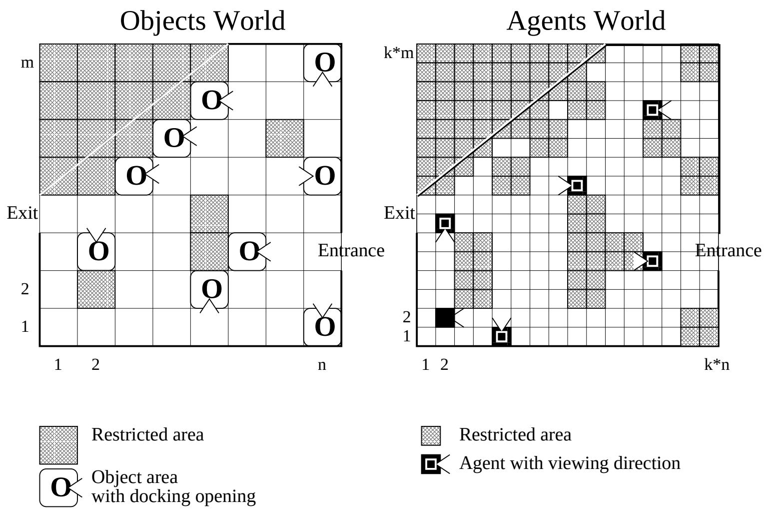

Figure 1: Object and agent world.

The setup of the objects is changed after one generation in order to collect enought data for the optimization cycle.

- The second component takes care of the object world. Basically the same size of the world is used as in the agent component case but the discretization of the world will be coarser. That is because agents do not, in general, occupy as much space as our objects do. The objects occupy one unit in the world. Objects have a certain type and therefore a certain cost to use c_i^o assigned which is deducted from an agents value if the object is visited by the agent. The objects collect the deducted values from the agents and after one generation the object has the value v_i^o. After each generation the setup of the objects is changed, basically to collect enough data for the optimization cycle. Some objects my be replaced by other objects from the object store.

Once the initial run has taken place, where e.g. data is collected over a period of $p \times \mathcal{T}$ time steps, the optimization of the object setup may take place. Through an evolutionary cycle, realized by a genetic algorithm, the setup of the objects is optimized according to a given fitness function. Usually we will try to maximize the sum of all object values.

In the following we will give relevant details for various topics.

Evolutionary Cycle We build our system on the generational based approach, i.e. populations of individuals are build whereas at the end of one generation fitness values are determined according to a predefined fitness function. In our case each individual (= encoding of the object position) gets assigned the sum of all values assigned to the individuals by the agents.

[1] As you might imagine the calculation of time step $t+1$ from time step t needs a certain update rule. If the same order of the agents is used for each time step the first agent will be preferred all the time. Thus a shuffling of the agent update positions will be advantageous.

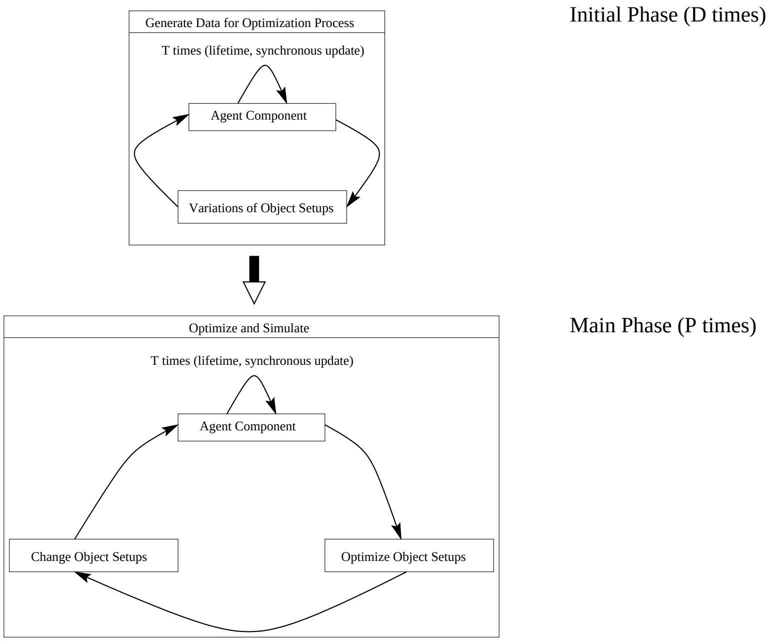

Figure 2: Overall optimization process.

Next the crucial question arises: How to encode the object position in a bit string? In our simulation we want to encode the object position in two chromosomes. Chromosome 1 (C_1) stands for the various possible positions of objects in the object world. Basically these are all non restricted areas. Since the object and agent world are organized in matrix form we may read the object world row by row and reserving one bit per non restricted area. In this case 1 represents the presence and 0 the absence of an object in that particular spot. The second chromosome (C_2) encodes the index for the objects. C_2 may be larger, equal or smaller than C_1. Since we encode the index of the objects we have to used more than one Bit for the index, namely $\lceil \log_2(\#objects) \rceil$. Further we have to take care in case of mutation that an object may appear exactly once in C_2. Thus we hava to modify the mutation operator in the following way:

1. According to mutation probability p_m determine a mutation location on C_2.
2. Choose a second spot on C_2 randomly.
3. Swap those two elements.

Using the algorithm above guarantees that an object occurs exactly once in C_2. In case of C_1 we can use the standard bit mutation operator.

In figure 3 we show the crossing over for two distinct individuals. For chromosome C_2 we could apply the same mechanism but we have to guarantee the appearance of one object exactly once. This requirement clearly implies the modification of the crossover operator in a non trivial manner. Thus we propose to copy the original C_2 of individual i_i and i_j and keep one original set. Thus we get again four chromosomes for exchange.

How do we have to determine the relevant objects for locations with value = 1 in C_1? C_1 is read out from left to right. Each time a 1 appears in the bit string we read the appropriate object index C_2 and increase the counter for C_2.

7 ALGAE (PPKT)

evolution, learning and communication (elc) is an implementation of a multi agent system (MAS) conceptualized and implemented by Roland Schwaiger and Konrad Lang. They investigate the evolution of

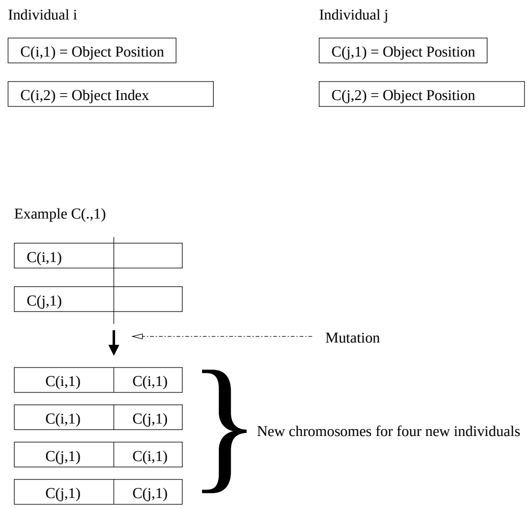

Figure 3: The details for crossing over between chromosome C_1 of individual i_i and i_j.

wandering behavior of simple (reactive) agents in a highly dynamic environment. Those simple agents wander based on the outcomes of feed forward ANNs (multi layer perceptrons). The inputs to the network are calculated from positions of other agents within a sensor range of each single agent. As an outcome the ANN determines the steering direction (Forward, Back, Left, Right). The ANNs of the agents are evolved according to some given criteria, e.g. maximize the number of places visited or maximize the number of distinct partners within sensor range. Surprising results evolve over time s.a. simple steering rules and complex group behaviors.

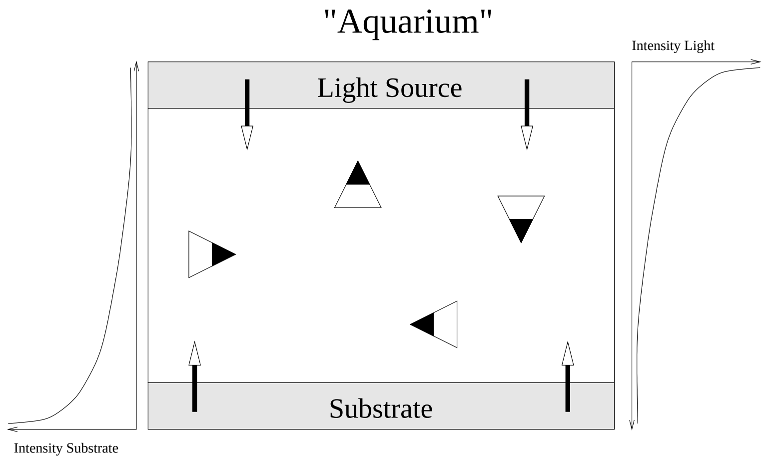

Figure 4: The world where the various agents live.

Based on the first findings in elc a followup system will be implemented based on the algae LOGOPODIUM. Imagine an aquarium where on top a uniform light source resides. At the bottom you may find some substrate. Algae (agents) live in the world between – for now we may view the aquarim as a two dimensional object. From time to time the light is switched on and off. As you might imagine, light has a higher intensity at the top of the aquarium than at the bottom. We might model the intensity by the function $1/x$ but any other function with similar shape will do, for now. The same holds for the substrate which diffuses into the water. The agents may consume the substrate, which after some time will be renewed.

All actions of one agent result from influences of its neighborhood. The agents (a_i) have sensors (receptors) for light ($L_i^j, j = 1,2,3,4$), substrate ($S_i^j, j = 1,2,3,4$), pheromone ($P_i^j, j = 1,2,3,4$) which is

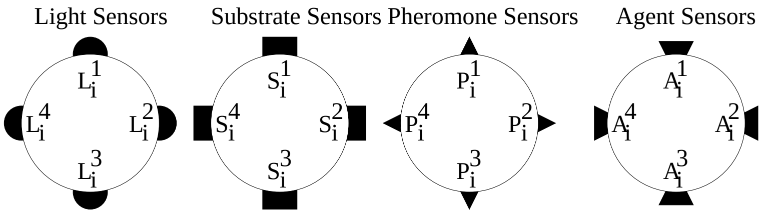

Figure 5: The various sensors of agents a_i.

released by other agents and agents ($A_i^j, j = 1, \ldots, 4$). According to the density of the various sources (light, substrate, pheromone) the sensors will show different values[2]. Through different values in the sensors an agent may "learn" to steer towards a target.

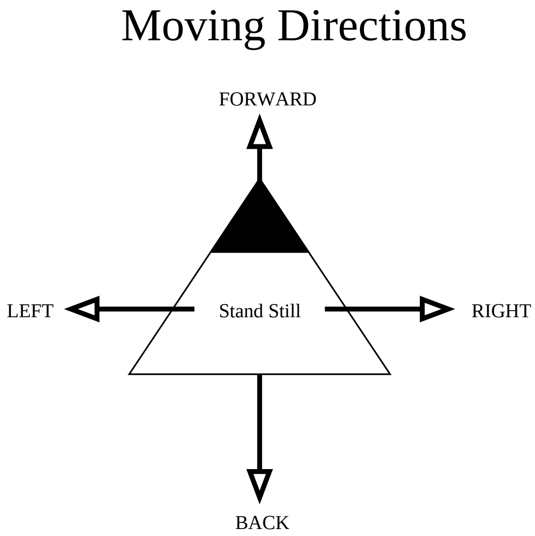

Figure 6: The possible moving directions of agent a_i.

The agents perform some actions according to their internal model. They may move in the world (aquarium). According to their local coordinate system they may move FORWARD, LEFT, RIGHT and BACK. The agents may stand still and do nothing. Obviously other agents or the borders of the aquarium hinter the agent to move according to the ANN output. Each action as described before requires a certain amount of energy. Agents consume substrate and light and collect and increase their energy level E. Agents may reproduce by mitosis (asexual) or meiosis (sexual).

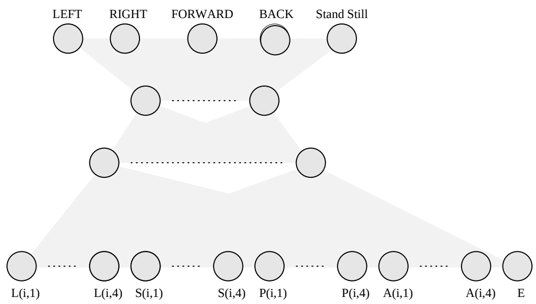

Figure 7: The ANN of agent a_i (first proposal).

[2]In order to model this situation a world object will set the sensor values

The various actions of an agent are partially determined by its mapping from input of the environment to output, i.e. reactions. This type of agent may be calles reactive agent or stimulus/response agent. In our multi agent system we model this stimulus/response relationship by means of an artificial neural network – multi layer perceptron – (ANN). The input to the ANN consists of the sensor inputs and the energy level of the agent. Various hidden layers are used to perform transformations of the signals. Finally at the output level the calculated moving directions are shown. At the moment of writing the shown ANN is our first proposal, it will change during course of discussion.

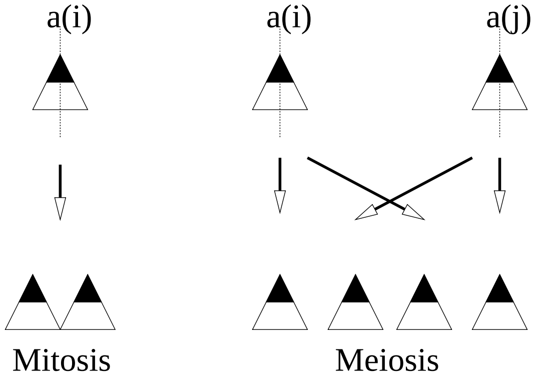

Figure 8: Reproduction of the agents a_i and a_j .

In the following we want to deepen the ideas given above, especially we want to list the relevant parameters for the simulation.

As has been pointed out, the algae live their lifes in a steady state population. That means that agents may life and prosper as long as their energy level does not reach 0. Once it has reached 0 the algae is damned to die. On the other hand if the energy level reaches a certain amount E_r, the algae may reproduce either sexually (meiosis) or asexually (mitosis). In order to find a partner the algae may produce pheromones. These pheromones may be detected by other agents throught their pheromone sensors. Possibly some algae learn to navigate to reproduction partners.

In the following we want to deepen the understanding of the discussion above and concentrate on relevant parameters and notations of the prblem under discussion.

Light and Substrate Let us denote the light distribution by $d_L(x)$ and the substrate distribution by $d_S(x)$. In the discussion above we have given an example for the distributions, namely $d(x) = \frac{1}{x}$. This is just one example and may be set by the user, possibly out of a given set of functions or a user definded distribtuion. Further we would like to mention that the two distributions do not have to be the same!

Contrary to the discussion above, in case of substrate distribution, we would like to model the time which a particular spot in the aquarium needs to rebuild the substrate concentration.

In figure 9 the distribution of the substrate is given. In each location (x, y) the substrate has concentration 1. Once a substrate is consumed by an agent the substrate will be rebuild according to the given $d_S(x)$. In the given figure we have chosen $d_S(x) = x^2$, for $y = 3$ we get $d_S(3) = 9$ which are the time steps the system needs to rebuild the substrate concentration[3].

Next let's consider the light distribution $d_L(x)$. Basically we may follow the discussion given above. In addition consider that an agent a_i in spot (x, y) blocks the light for all the agents a_j in locations $(x, y'), y' < y$. Naturally the licht intensity will degrade according to a certain transparency factor τ_i of agent a_i.

In order to model day and night in the simulation $d_L(x)$ may depend on the time. As an example consider the sin function, as seen in figure 10. Further the time dependend modeling of the substrate concentration may be considered.

[3]In a future release of the simulation the substrate concentration in (x, y) only may be rebuild if there is a substrate concentration in locations $\{(x, y')|\forall y' : y' < y\}$

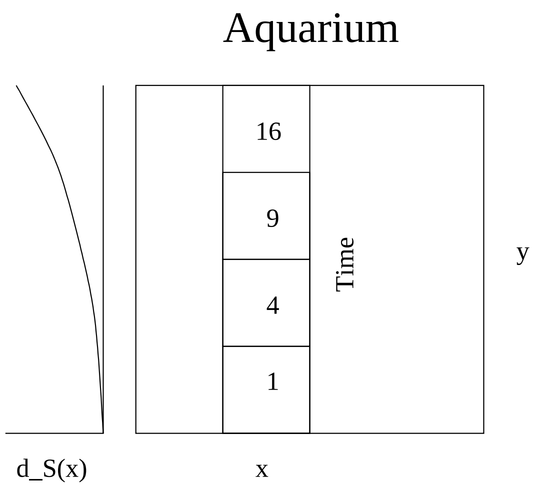

Figure 9: The concentration and distribtution of the substrate, according to $d_S(x) = x^2$.

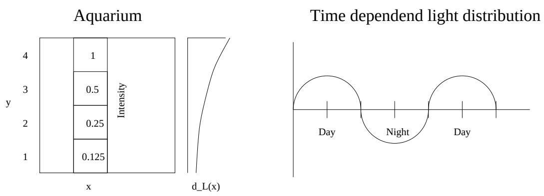

Figure 10: The concentration and distribtution of the light, according to $d_L(x) = 1/x$.

Aquarium The aquarium has, for the moment, a two dimensional extension, specified by $(worldx, worldy)$. The borders on the left and right have a transparecy factor $= 0$, i.e. no light may shine through the borders. Naturally the light sorce has the highest light concentration and the substrate, as the borders the lowest light intensity.

Pheromone, Mitosis, Meiosis Once an agent a_i has collected enough energy points it may reach a critical limit for sexual reproduction E_s. From that moment on the agent may produce pheromone in order to signal other agents its ability for sexual reproduction. Of course, producing pheromomes will reduce the energy level of the agent, let's denote the reduction ΔE_s. The agent will release the pheromone in its actual location[4]. The pheromone will be apparent in the spot for given amount of time steps, let's denote the amount of time by π_t.

Once an agent has an energy level greater then E_s and it meets another agent we have various ways to consider meiosis.

1. To distinguish between the pheromones of different agents we may introduce two genders, male and female. The two genders release different kinds of pheromone. As can be seen easily, we reduce the probability of reproduction significantly.

2. Another way to consider pheromone of different agents we may intro two agent types. The first type just releases pheromone and the second agent type just senses pheromone. Apply the argument above.

3. If one agent releases pheromone and meets another agent it will force the agent to reproduce, no matter if the other agent has an energy level greater than E_s.

4. Finally, we introduce a flag for pheromone for each agent. With this method we get the possibility to differentiate between the pheromone of the various agetns and just agents with energy level greater than E_s may reproduce. Unfortunately not a biological plausible proposal.

[4] In future releases of the simulation we may consider some distribution function for pheromone release in the neighborhood of the agent

In our simulation we will focus on the third possibility.

On the other hand, if no other agent is in reach and the agent collects enough energy points then it may reach an upper limit, E_a. In this case the agent will perform mitosis and decrease its energy level by ΔE_a.

World Basically the world is responsible for controlling the experiment, some of its tasks are:

1. Set the output of the experiment in the GUI, i.e. show the agent, light distribution, substrate distribution, show details of the agents, s.a. energy level, genotype, phenotype, etc.
2. Perform all calculations from one time step t_i to time step t_{i+1}, new agent positions[5], set sensor values of the agents[6]
3. Insert new agents in the world once reproduction has taken place.
4. ...

Artificial Neural Network The ANN is responsible for mapping the sensor values and the energy level to steering behavior of the agent. Thus, at the moment of writing it has four input units for each sensor and one input unit for the energy level. The signal is transfered via hidden units to the output (steering units). At the moment we consider four directional output units (Left, Right, Forward, Back) and one unit for stand still. The various units may be seen in figure 11.

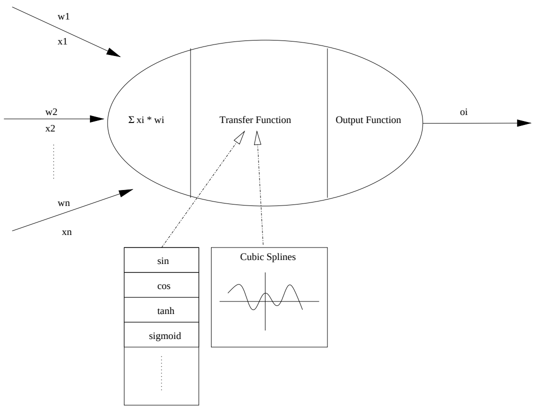

Figure 11: The detail view of one neuron in the ANN.

Evolutionary Cycle and Reproduction As pointed out above we want to build our system based on the steady state approach, i.e. no gereations are built and definite end is proposed but reproduction occurs on the fly. Basically all relevant information has to be encoded into various chromosomes (bitstrings). In our simulation we want to use three chromosomes per agent. The first chromosome C_1 encodes the topology (neurons and connections) of the ANN in a Modified Miller Matrix (MMM) manner. Basically the main diagonal of the connectivity matrix where an entry $m_{ii} = 1$ states that the neuron is present. Whereas entry $m_{ii} = 0$ states that the neuron is not present. If the entry is not in the main diagonal m_{ij}, where $i <> j$, then the entry states the existance of the connection from neuron i to neuron j.

The second chromosome C_2 encodes the weights of the connections and the biases of the neurons, two Bytes are used for encoding. The third chromosome C_3 encodes the cubic splines, where each point of the spline is encoded with its x and y coordinate, two Bytes respectively.

In order to reproduce we have to consider two different cases, meiosis and mitosis. In figure 12 we give an example for meiosis. In the mitosis case we duplicate the chromosomes and perform the same steps as in the meiosis case.

[5]The agents perform their steps and request from the world their new positions, since only the world knows about the positions of the other agents. Further just the world knows about the concentrations (light, substrate, pheromone).

[6]This has to happen in the first phase of intervall $[t_i, t_{i+1}]$ since all further actions of the agents rely on the sensors values.

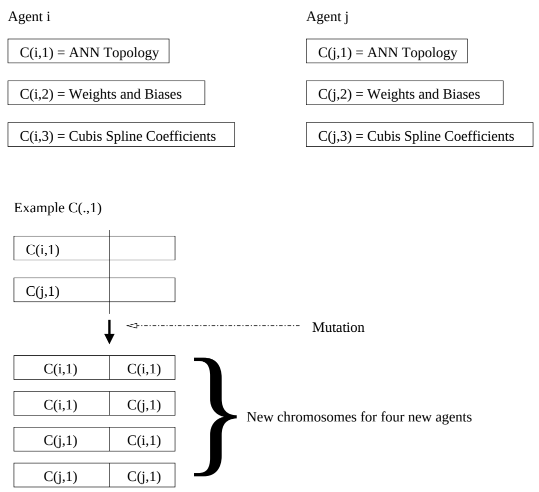

Figure 12: The details for crossing over between chromosome C_1 of agent a_i and a_j.