Evolving artificial "brains": a biomimetic Evolutionary Neuro-Genetic Algorithm (ENGA) and its testing in simple games

PETER ITTZÉS, ZOLTAN SZATMÁRY, SZABOLCS SZÁMADÓ AND EÖRS SZATHMÁRY

Collegium Budapest (Institute for Advanced Study)

Running head: ENGA

2 Szentharomsag utca, H-1014 Budapest, Hungary
Phone: +36 1 2248300, Fax: +36 1 2248310
Email: szathmary@colbud.hu

Abstract

The evolution of autonomous communicating agents is a hot area in recent research. We have developed a software framework, called ENGA, to evolve flexible neuronal networks capable of solving first simple but then more and more complicated communication tasks. We took a biomimetic approach, that is, the design of the system was inspired by important biological phenomena such as brain ontogenesis, neuron morphologies, and indirect genetic encoding. We exposed our neuronal networks to simple coordination and task-allocation games to test the potential of the system. Neuronal networks were selected and were allowed to reproduce as a function of their performance in the given task. The selected neuronal networks in all scenarios were able to solve the communication problem they had to face. Although this result tells little about the full potential of the biomimetic approach, it is a promising start, on which we intend to build by introducing our neuronal networks to more and more complicated communication scenarios. The most striking feature of the model is that it works with highly indirect genetic encoding—just as brains do.

Key words

Biomimetic approach, neural networks, indirect encoding, origin of communication, artificial brain

Introduction

Emergence of communication systems in populations of agents is a field of active investigation [1-3]. Numerous approaches have been used and many of them include neural networks, evolutionary computing or both. Though these got their inspiration from biology, the actual level of biological realism varies, in some cases leaning toward extreme reductionism. The fact, that artificial systems up to the present day have proven to be inferior to natural systems in many aspects of communication, may suggest that biological phenomena should be imitated more closely. Of course, biology is a vast field ranging from biochemistry to evolutionary biology, and one would not want to incorporate it all in a huge simulation. However it would be of great help to experiment with such a system and be able to identify which parts of it can safely be reduced to simpler mechanisms or representations while at the same time identify vital components without which the simulation would fail.

We have developed a software framework called Evolutionary Neurogenetic Algorithm (ENGA in short) which offers researchers a fine control over biological detail in their simulations. Our original intent was to create software with much potential for variability. That is, we wanted a piece of software which is general enough to allow for a wide range of experimentation but appears as a coherent system and does not fall apart into a loose set of unrelated pieces of code. This required careful specification and design; especially in partitioning it into modules and the specification of interfaces in a programme that has grown to about 10,000 lines of C++ code. A product release is expected in the future for the benefit of the community.

In such a short communication it is impossible to acknowledge all researchers of all important input fields to this paper. We have been especially influenced by evolutionary robotics, such as the work by Baldassare et al. [1], and by the evolutionary approach to neuronal networks with indirect encoding by Rolls and Stringer [4]. Our model is a recombinant of these approaches, with some key new elements, such as topographical network architecture.

Below we provide a brief enumeration of the supposedly most important properties of neuronal networks, genetic codes and development processes. This enumeration is far from complete but we have taken it into account when designing our system.

Genome

Scalability. If our goal is to develop an evolutionary scenario, which leads to a complexification process, then we have to be very careful with the coding system behind the neural system. The encoded phenotype information has to be indirect to avoid the negative effect of genetic linkage [5] and exponential growth of the genome size. Nature has shown that the number of genes is orders of magnitudes less than the number of structural units of the phenotype. Even with 100 trillion neural connections in the human brain, there are only about 30 thousand active genes in the human genome [6].

Neuronal networks (phenotype)

Complex Network Properties. Recent research revealed that large-scale brain networks have interesting complex network properties. Scale-free and small-world properties, functional segregation (modularity) and functional integration might be necessary to achieve complex cognitive functions [7].

Degeneracy. A degenerate system, unlike a fully redundant one, is extremely adaptable to unpredictable changes [8].

Lamination. It was shown that lamination (as in the cortical layers) might be advantageous to support more complex information processing in the cortex [9].

Genotype-phenotype mapping and development

Synapse Development. If the scaling of a network causes problems, week or indirect mapping has to be applied [10]; it can be a graph-generating algorithm or a partly stochastic connection building process [11]. There is even a new subdiscipline of evolutionary computation that engages in the field of genotype-phenotype mapping in artificial systems called artificial embryogeny (AE) [6].

Learning. A learning phase in the development process can change the fitness landscape, and it can be as effective as a Lamarckian strategy at improving search [12]. Because synapse development is a non-deterministic process, learning plays a key role in synapse selection, too.

Synapse selection, pruning. One hypothesis for the importance of pruning is that if the synapses at first are overgrown and later pruned, the memory performance of the network is maximized in certain situations [13]. Another possible explanation for the importance of pruning arises from the developmental process. If the synapses join another neuron in a stochastic way, optimal wiring should emerge after learning based pruning. Redundancy is a precondition to this process [14].

Specification and design

When specifying a piece of software we delineate what features we expect from the programme and how these features should behave. Specification involves decision making, and decisions are better not taken on an arbitrary basis. We took biomimesis as our guiding principle and finally a set of core assumptions and a complementing set of potential variabilities have emerged. The core assumptions summarize ideas we consider fixed throughout all simulations. Some of the most important such assumptions are listed in the following sections.

Population and agents

There is a population of agents. Agents have a genotype, a brain and a body. Bodies contain sensors and effectors and as such they provide the necessary input and output facilities for brains. Bodies reside in a simulated or real physical environment.

The genetic system

In general, information stored in a genotype will be used during phenotype creation, i.e. during ontogenesis. Genotype-phenotype mapping is usually a nontrivial task and decisions involving this process can severely influence the performance and capabilities of a simulation. If we design a coding system, which is not open-ended, evolution will stop at the boundary of the genetic space. One way to create an open-ended coding is to use a marker-based encoding scheme that was inspired by the structure of the real genetic code [15]. The other promising way is to encode information of the network in a tree structure [10]. In our system we chose a hierarchical tree-based representation. Genetic operators (point mutation, deletion, duplication, and recombination) work on this tree structure and alter the values of the nodes or change the structure of the tree. Duplication ensures the open-ended behaviour of the system, and is able to scale up the search space, if it is required in a new situation. To reduce the negative effect of genetic linkage [5], the use of recombination is necessary. The optimal combination of these operators is an open question and it depends on the coding system and on the fitness landscape of the given task.

Neuronal networks

Neuron classes. Neurons are classified. Neurons belonging to the same class share the same functional and morphological properties, whereas neurons belonging to different classes may differ in these properties. Similarly, synapses belong to classes, too. We call them 'interclasses' because they represent functional properties that are shared by all synapses connecting a neuron belonging to a particular class A to a neuron belonging to class B (A and B can be the same class).

Topography. Unlike the majority of approaches to evolve neural networks we maintain the full topographic information of our networks, that is, neurons are situated in a layered neural space. Topographical information in a neural network can have a number of advantages. First, the interpretation of the structure can be easier. Second, developmental processes can model biological processes of neuron and synapse growth more accurately. Models which acknowledge spatial information in biological systems yield various scale-free and small-world network attributes like the ones that are common in brain structure [7].

Neuron Morphology. It was suggested that more structured neuron morphology is required for more complex cortical functions [16]. A more complex surface can facilitate more complex connection structure and if different neuron morphologies are encoded in the genome, the search process can change the network structure by changing morphology. Representation of neural networks in a topographic space allowed us to equip neurons with spatial dimensions. In our simulations neuron morphologies were only coarsely detailed and reduced to circles on each layer.

Tasks

In each simulation there is a task defined. Agents gain a so-called score (fitness) value according to their proficiency in the task. This fitness value can be used by death and reproduction processes (see Death-and-reproduction). Tasks can be solitary or social, reflecting the number of agents required to solve them. In case of social tasks the fitness of an agent may depend on the performance of its fellows as well.

Death and reproduction processes

There is a death process operating in the population. It takes out members from the population forever. Parallel to the death process a reproduction process is going on, replenishing the population by allowing some members of the population to reproduce. It is the responsibility of the reproduction process to stabilize the size of the population (this is necessary because of finite computing resources). The reproduction process may include a mating process (if the reproduction is sexual), selecting pairs from the population.

Lifecycle of agents

Following the initialization of the population agents start their lifecycle. It begins with the birth phase in which a body and a brain is generated. Only brain generation is affected by the agent's genotype. Brain ontogenesis consists of two steps: neurogenesis and synaptogenesis. The actual life of an agent can be separated into two parts: childhood and adulthood. Learning takes place during both ages, but pruning is present only during childhood and only adult members of the population can reproduce. The separation line between childhood and adulthood is dependent on the actual simulation and may completely disappear.

Ontogenesis of the brain. Development of the brain is controlled by the genotype in a highly indirect manner. By indirect we mean that no part of the genotype corresponds to individual neurons and synapses, and that only gross statistical properties of the brain are encoded in the genes. Individual brains are sampled from this statistical description. The neurogenesis phase goes on as follows. Neurons are situated in a layered topographic neural space mimicking real cortical layers but low-level biological mechanisms shaping the cortex such as concentration gradient dependence in neurogenesis are not present in the simulation. Instead, neuron classes define a probability density function over the neural space from which individual neuron soma positions are sampled. Neurons possess morphologies, i.e. there is some function over neural space describing their dendritic arborization. This morphology is applied to each neuron relative to its sampled soma position. Synaptogenesis exploits two mechanisms, just as in biology: a long-range one (called projections) and a subsequent short-range one (lock-and-key mechanism). Each neuron class has an associated list of projections. Projections are probability density functions over neural space. They can be defined either in absolute coordinates or in coordinates relative to a neuron's soma. When a neuron's efferents are to be determined, putative synapse locations are sampled from its projections (determined by its neuron class). Neurons having dendritic arborization near these putative locations become candidate efferents. Then the short-range mechanism selects from competing candidates at each synapse location. The short range lock-and-key mechanism mimic the receptor-ligand based binding mechanisms present in real synaptogenesis. Locks and keys are 30 long strings of bits. Every candidate postsynaptic neuron's lock is matched to the presynaptic neuron's key. Binding probability is then a decreasing function of the Hamming distance between the key and the lock. The complete ontogenetic algorithm is depicted in Fig. 1.

Potential for variation

Potential for variation arises within the boundaries of the core assumptions. For example, many kinds of activity models and dynamics can be imagined: rate code or spikes as activity model and summation of synaptic contributions fed through various transfer functions as activity dynamic. Of course, variations of core concepts are not necessarily compatible with each other. If, for example, a synapse model expects the presynaptic neuron activity to be rate coded then somehow it must be ensured that the presynaptic neuron class really defines its activity model as rate code. ENGA provides some mechanisms to enforce such compatibilities but eventually it is always the person designing the simulation who is responsible for the overall coherence.

Programme architecture

The architecture is of layered nature so that lower modules do not know about the existence of higher modules (Fig. 2). This allows easy modifiability of higher levels without the need to modify lower levels. Moreover, each layer exposes an interface that can be used by any client, even those deviating from the original purpose of simulating evolution of embodied communicating agents. The genetic module for example can be used in any evolutionary computation, not only those evolving artificial neural networks. We may as well talk about a multilevel software framework consisting of several modules that can be used individually or in combination with others to produce various kinds of evolutionary and neural computation related simulations.

Test tasks: simple coordination and task-allocation games

We have used simple 2x2 matrix games with one-way pre-play communication as test tasks: a coordination and a task-allocation game. There are two reasons we have chosen these games. First, because of their simplicity these scenarios are easy to implement. Second, these games are well studied in the economics literature, and it is shown both theoretically and empirically that (one-way) cost-free, pre-play communication (so called 'cheap talk') can resolve coordination problems in both coordination and task-allocation games [17-18]. Thus it is reasonable to expect that our agents will evolve the use of signals under these scenarios. To test further the flexibility of the neuronal architecture of our agents we have introduced two environments, which have an influence on the pay-off matrices.

Accordingly we have four different scenarios:

I. Control. The fitness of a given agent is independent from the choice of behaviour of the other agent in the game.
II. Coordination game. Agents have to pick the same response, corresponding to the given environment to get high fitness.
III. Task-allocation game. Agents have to pick complementary responses to get high fitness.
IV. Coordination or task-allocation depending on the environment.

There are: two kinds of environment, $E \in \{-1, 1\}$, a continuous signal S which can take values in [-1,1], and a continuous response function R which can take values in [-1,1]. For signal and response functions only the -1 and 1 values are taken as correct signal or response, all other values are taken as nonsense signal (NS) or nonsense response (NR), respectively. Nonsense signal and response are not rewarded in any of the following games.

I. Control

The fitness of a given agent is independent of the response of the other agent in the game; it only depends on the environment. In case of $E(1)$ the good response is $R(1)$, in case of $E(-1)$ the good response is $R(-1)$.

II. Coordination game

A pair of agents plays a coordination game in which they have to choose a given response conditional on the environment. (i) If both agents pick the correct response they both get a high reward; (ii) if only one of them picks the correct response then one gets a higher reward than the other; (iii) if both of them picks the wrong response then they both get low rewards.

III. Task-allocation game

A pair of agents plays a task-allocation game in which they have to choose complementary responses to get high fitness. If they pick the same response they get low reward regardless of the environment.

IV. Coordination/Task allocation game

A pair of agents plays a coordination or a task-allocation game depending on the state of the environment. In case of $E(1)$ they play a task-allocation game, in case of $E(-1)$ they play a coordination game.

Steps of the game

1. Mother Nature picks an environment: one of the environments is being selected randomly.
2. Observation: one randomly chosen agent out of each pair can observe the environment, that is, he has correct information about the type of the environment in which the game is situated.
3. Communication: the observer may or may not give any of the signals (S) available to it.
4. Picking the response: after the observer gave his signal both agents pick a response (R) simultaneously. The fitnesses of the agents are calculated from the pay-off matrices according to the game and to the given environment.

Steps of the simulation

1. The simulation is initialised with M naive agents.
2. Each turn of the simulation consists of two stages: playing the game (each agent plays with every other agent) and reproduction.
3. Reproduction: The agent with the least fitness dies. Reproduction is sexual, thus we pick a pair of individuals to produce an offspring that will fill the vacant

position. The chance of an agent being picked to reproduce is proportional to its fitness. After reproduction we return to stage one.

4. The simulation is iterated for *t* turns.

Results

Our agents have achieved a better-than-random performance in all scenarios which suggests that they have managed to communicate sufficient amount of information to solve the task under question. In case of the control probably only as a side-effect, given that signals are cost-free (i.e. not selected against even if they confer no benefit to its users).

The average fitness as a function of time in each population is depicted in Fig 3. One can see that the slowest increase in fitness can be observed in the population which played either a coordination or a task-allocation game, depending on the environment. Finally, it is worth mentioning the topology of the evolved networks. Figs 4 and 5 depict an 'early', and an 'advanced' neuronal network taken at the 10th and at the 750th generation out of the population playing the 4th scenario (coordination or task allocation game depending on the environment). As we can see selection had a profound effect on the topology. While the first network failed to make connections to decision output and is doomed to low fitness values, the later 'advanced' network utilizes both visual and audio input to determine the values of decision and signal outputs.

Discussion

The results of the games analysed in this paper suggest that the basic idea—the biomimetic approach—behind our neuro-genetic system is correct, and the highlighted design principles of the system are sufficient at least in this simple task. The other goal of the test was to prove that the developed software is working properly is useful for scientific experimentation. The core of our neuro-genetic system is the stochastic ontogenetic process and the indirect tree-based coding behind it. Evolution of agents in these scenarios found the right solution in a relatively short period, and the length of the search time was proportional to the complexity of particular game. The topology and size of the networks both changed in the simulations. It is important to emphasize that we have not taken advantage of any grossly unnatural (but sometimes very effective) algorithms such as back propagation.

Our results show that indirect encoding and a genetically controlled but stochastic ontogenetic process together can provide an appropriate framework for evolving communicating agents in simple scenarios. Our next goal is to introduce our neuronal networks to more and more complicated selective scenarios, by which we hope to obtain better understanding of the evolution of, among other things, symbolic communication.

Acknowledgement

This paper was supported by the IP ECAgents (http://ecagents.istc.cnr.it) in the 6th Framework Programme. Comments by Chrisantha Fernando are gratefully acknowledged.

1. Baldassarre G, Nolfi S, Parisi D: Evolving mobile robots able to display collective behavior. Artif Life 2003;9: 255-267.
2. Steels L: The Talking Heads Experiment.Words and Meanings. Antwerpen, 1999.
3. Kvasnicka V, Pospichal J: An emergence of coordinated communication in populations of agents. Artif Life 1999;5: 319-342.
4. Rolls ET., Stringer SM. On the design of neural networks in the brain by genetic evolution. Prog Neurobiol 2000;61: 557-579.
5. Calabretta R, Di Ferdinando A, Wagner GP, Parisi D: What does it take to evolve behaviorally complex organisms? BioSystems 2003;69: 245-262
6. Stanley KO, Miikkulainen R: A taxonomy for artificial embryogeny. Artif Life. 2003;9: 93-130.

7. Sporns O, Chialvo DR, Kaiser M, Hilgetag, CC: Organization, development and function of complex brain networks. Trends Cogn Sci 2004;8: 418-425.
8. Tononi G, Sporns O, Edelman, GM: Measures of degeneracy and redundancy in biological networks. Proc Natl Acad Sci 1998; 96: 3257-3262.
9. Montagnini A, Treves, A. (2003) The evolution of mammalian cortex, from lamination to arealization. Brain Res Bull 2003;60: 387-393.
10. Zhang B, Mühlenbein H: Genetic programming of minimal neural nets using Occam's razor. Proc. of the Fifth Int. Conf. on Genetic Algorithms, 1993: 342-349.
11. Balakrishnan K, Honavar V: Evolutionary design of neural architectures: A preliminary taxonomy and guide to literature. Technical report 1995; CS-TR-95-01, AI. Lab, Dept. of Computer Science, Iowa StateUniversity.
12. Gruau F, Whitley, D: Adding learning to the cellular development of neural networks: evolution and the Baldwin effect. Evol Comput 1993;1: 213-233.
13. Chechik G, Meilijson I, Ruppin E: Synaptic pruning in development: a computational account. Neural Comput 1998;10: 1759-1777.
14. Kerszberg M, Dehaene S, Changeux JP. Stabilization of complex input-output functions in neural clusters formed by synapse selection. Neural Networks 1992;5: 403-413.
15. Fullmer B, Miikkulainen: Using marker-based genetic encoding of neural networks to evolve finite-state behaviour. Proc. of the First European Conference on Artificial Life. Paris. 1991.
16. Elston GN, Benavides-Piccione R, DeFelipe J: The Pyramidal Cell in Cognition: A Comparative Study in Human and Monkey. J. Neurosci. 2001;21: RC163 1-5.
17. Cooper R, DeJong DV, Forshyte R, Ross, TW: Communication in Coordination Games. Quart J Econ 1992;107: 739-71.
18. Hurkens S, Schlag, KH. Communication, Coordination, and Efficiency in Evolutionary One-Population Models. J Econ. Literature 1999;

LEGENDS

Fig 1. Neurogenesis and long range mechanism of synaptogenesis
(a) Probability distribution of positions. (b) Sampled soma positions. (c) Morphology added to soma. (d) Projection (on layer 2) from a presynaptic cell marked with blue. (e) Putative synapse location (X mark) is sampled from projection. (f) Candidate postsynaptic cells (marked with magenta) are determined.

Fig. 2. The architecture of the programme
Modularity of the different component is apparent.

Fig. 3. Average fitness as a function of time in all 4 scenarios
Orange: control, green: coordination game, red: task allocation game, blue: coordination or task allocation game depending on the environment.

Fig. 4. Topology of an early, incompetent neuronal network
Colour code highlights the flow of information of the visual and audio input from the environment. When these channels join on a particular neuron, mixed colour indicates joint information processing.

Fig. 5. Topology of a late, advanced neuronal network
Colour code as in previous figure.

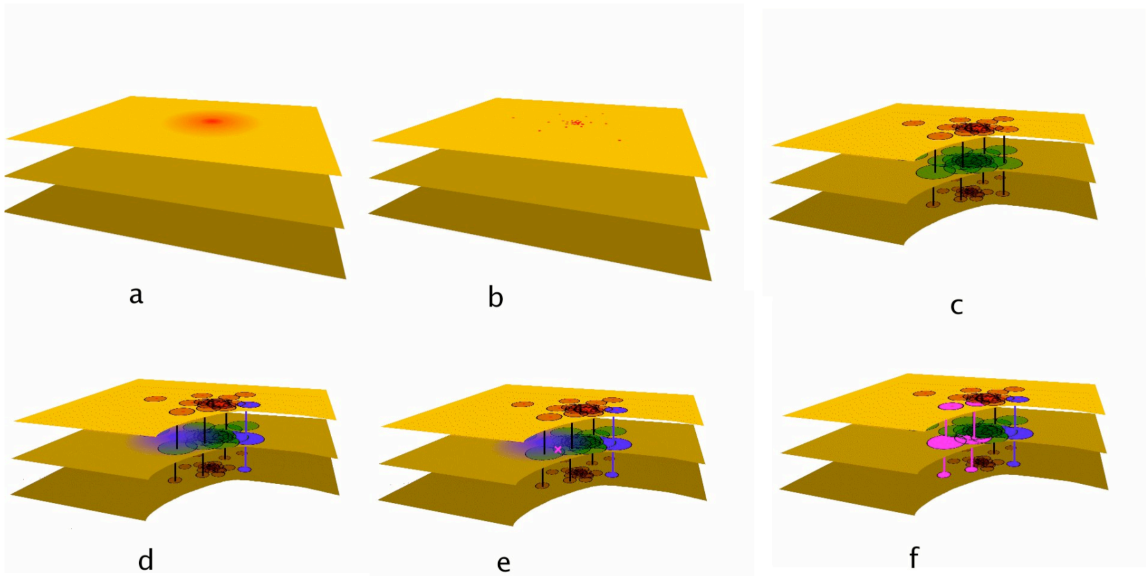
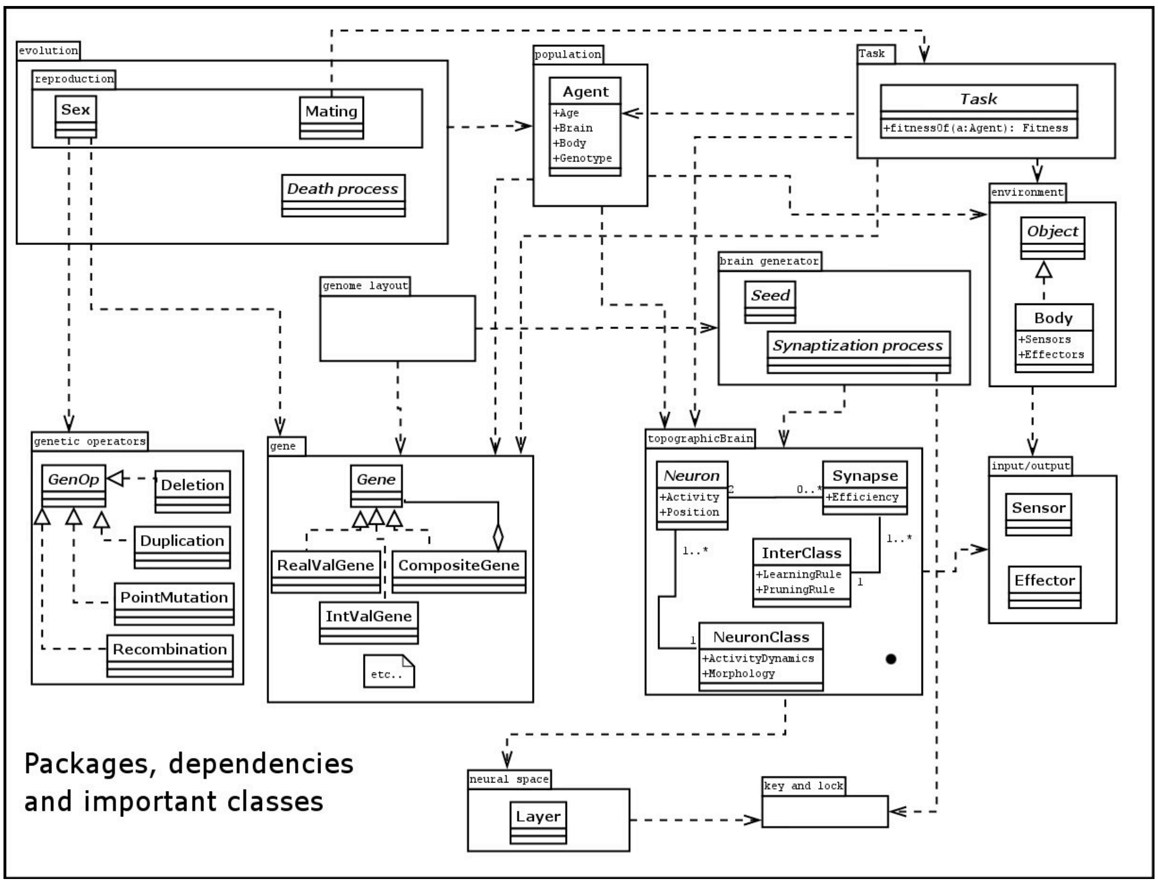
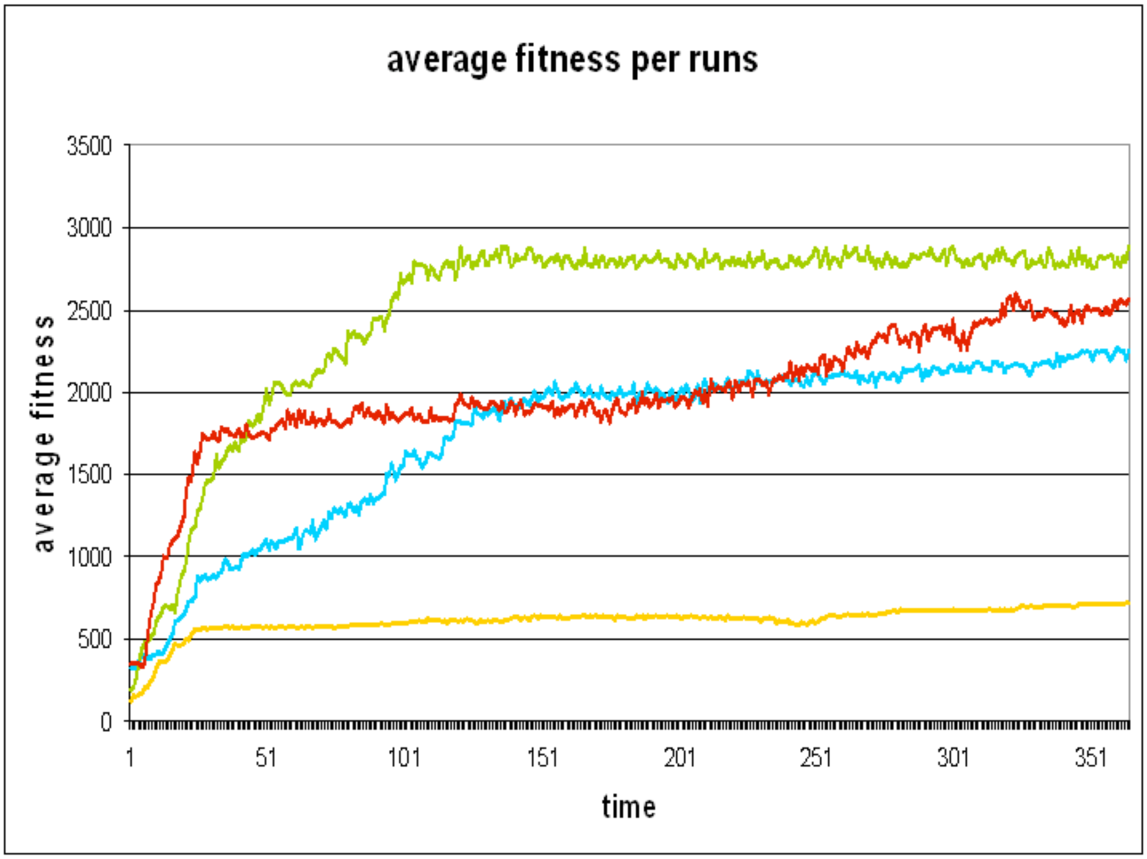


Figure 1.



evolution
reproduction
Sex
Mating
Death process
population
Agent
+Age
+Brain
+Body
+Genotype
Task
Task
+fitnessOf(a:Agent): Fitness
environment
Object
Body
+Sensors
+Effectors
brain generator
Seed
Synaptization process
genome layout
genetic operators
GenOp
Deletion
Duplication
PointMutation
Recombination
gene
Gene
RealValGene
CompositeGene
IntValGene
etc..
topographicBrain
Neuron
+Activity
+Position
Synapse
+Efficiency
InterClass
+LearningRule
+PruningRule
NeuronClass
+ActivityDynamics
+Morphology
0..*
1..*
1..*
1
1
input/output
Sensor
Effector
Packages, dependencies
and important classes
neural space
Layer
key and lock

Figure 2.



average fitness per runs
average fitness
time
3500
3000
2500
2000
1500
1000
500
0
1
51
101
151
201
251
301
351

Figure 3.

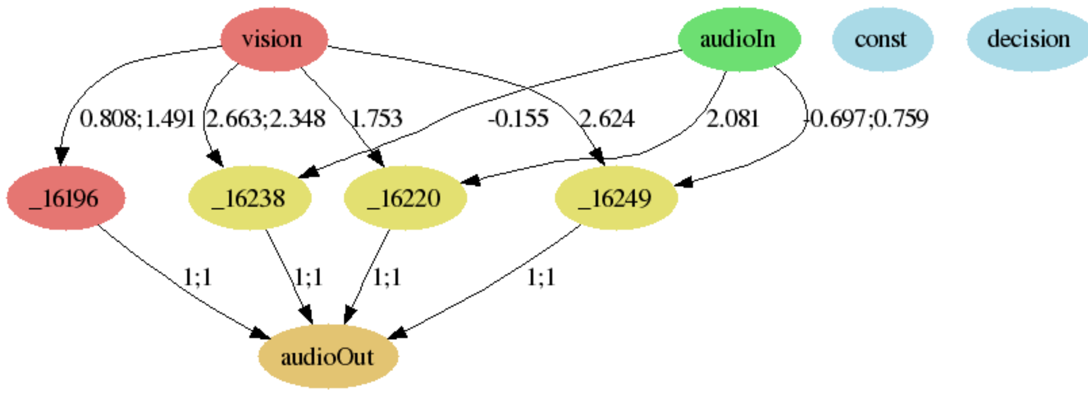

vision
audioIn
const
decision
0.808;1.491
2.663;2.348
1.753
-0.155
2.624
2.081
-0.697;0.759
_16196
_16238
_16220
_16249
1;1
1;1
1;1
1;1
audioOut

Figure 4.

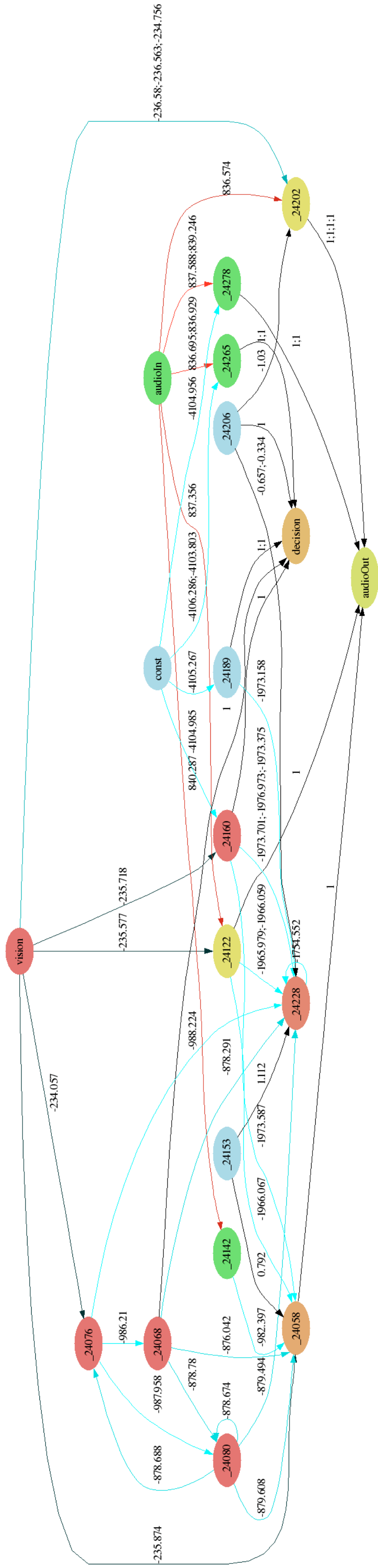

vision
audioIn
const
decision
audioOut
_24202
_24278
_24265
_24206
_24189
_24160
_24122
_24228
_24153
_24142
_24058
_24076
_24068
_24080

Figure 5.