Developing Adaptive Systems: Epigenetic Neuroevolution

eingereichtes
HAUPTSEMINAR
von

cand. ing. Thomas Wolf

geb. am 13.02.1990
wohnhaft in:
Gebelestrasse 26
81679 München
Tel.: 0176 92615190

Lehrstuhl für
STEUERUNGS- und REGELUNGSTECHNIK
Technische Universität München

Univ.-Prof. Dr.-Ing./Univ. Tokio Martin Buss

Betreuer: M.Sc. Cristian Axenie
Beginn: 07.10.2014
Abgabe: 14.01.2014

In your final hardback copy, replace this page with the signed exercise sheet.

Abstract

This research paper presents different approaches in the field of Epigenetic Neuroevolution and investigates their capabilities. The goal is to understand the underlying principles and how they can be combined to create brain-like learning machines that can adapt to and learn from environmental changes, just like humans, by taking sensory feedback into account. Therefore, a short overview of basic neuroscience and machine learning principles is given, followed by a derivation of the Epigenetic Neuroevolution approach. Based on these concepts, different approaches are presented to build adaptive and robust systems for machine learning tasks. The work concludes with an imaginary scenario combining the discussed approaches and further investigating the possibilities of the gained knowledge followed by a short discussion.

Zusammenfassung

In diesem Forschungsbericht werden verschiedene Ansätze im Bereich Epigenetic Neuroevolution vorgestellt und auf deren Leistungsfähigkeit untersucht. Das Ziel ist, die zugrundeliegenden Methoden zu verstehen, um sie zu einem lernfähigen intelligenten System kombinieren zu können. Mit diesen Methoden soll es einem System möglich sein, ähnlich einem Menschen, von seiner Umwelt und dessen Feedback zu lernen um sich an neue, bisher nicht erfahrene Situationen anpassen zu können. Zu Beginn werden grundlegende Methoden aus dem Bereich Maschinelles Lernen und der Neurowissenschaften erläutert, gefolgt von einer Herleitung des Epigenetic Neuroevolution Ansatzes. Anschließend werden verschiedene Arbeiten aus diesem Feld untersucht, welche dann in einem fiktionalen Szenario angewendet werden. Der Bericht schließt mit einer Diskussion über die erworbenen Kenntnisse.

Contents

Chapter 1

Introduction

Starting from performing simple tasks robots and artificial agents are nowadays capable of dealing with more and more complex situations. A limiting factor for these systems is that they are especially programmed or trained for a specific task. If the environment, or the conditions they work in change, these systems cannot adapt and will probably fail. The concept of Growing Adaptive Systems ([KBD14]) uses bio-inspired principles trying to overcome these limitations by developing systems that are able to learn from their experience and adapt to environmental changes rather than being retrained for the new environment.

This seminar investigates the approach of Epigenetic Neuroevolution, which takes the evolutionary process in the human brain as a motivation for adaptive learning systems. Learning in the human brain, a huge net of interconnected neurons, is a highly complex process. Each connection has its purpose and has evolved through a long learning period. Depending on the performed task, different neurons fire different patterns. Busy connections with lots of neuronal activity are strengthened, and others with a low amount of activity are weakened. If a new task is learned, the structure of the brain will adapt and new connections will be established. Likewise if a connection is not needed anymore it will possibly be pruned. This process is triggered given exposure to new external sensory inputs and influenced by experience.

Artificial Neural Networks (ANNs) are already an approach to use the concept of the human brain in artificial intelligence. One drawback is that standard ANNs are constructed and trained for a specific task beforehand can therefore not adapt to new scenarios. Specially handcrafted topologies are used and the synaptic connections weights are trained afterwards. Recently, approaches were made to further optimize ANNs by using Genetic Algorithms and/or Constructive and Pruning Algorithms to figure out optimal topologies as well. It has to be noted that this optimization happens during the training phase of an ANN. Once the training is done and the network is specified for a specific task it cannot and will not adapt to a changing environment.

Epigenetic Neuroevolution is a new approach trying to overcome these limitations

by letting the network learn from the environment and adapt to changes. This is achieved by a combination of different bio-inspired techniques from Artificial Neural Networks, Evolutionary Algorithms and the concept of Reinforcement Learning.
Section 1.1 explains the fundamental components the approach is built upon, Section 2 gives an overview of the previous acquisitions in the field of Growing Adaptive Systems and Section 2.1 shows the general structure of an Epigenetic Neuroevolution approach as well as possible design choices. In Chapter 3 already implemented approaches are investigated and in Chapter 4 a fictional scenario were a Epigenetic Neuroevolution approach could be applied is presented.

1.1 Basic Concepts: neural models of computation and development

This section is gives a deeper insight into the basic concepts and ideas that are combined in Epigenetic Neuroevolution.

1.1.1 Artificial Neural Networks and Supervised Learning

Artificial Neural Networks (ANNs) are very popular models used in machine learning for a variety of applications such as classification, data clustering, function approximation and regression up to robot control ([WSM43], [MM69]). The inspiration for ANNs comes from the structure of the human nervous system, consisting of a huge network of interconnected neurons. In ANNs each neuron is defined by its number of inputs, its synaptic weights and an activation function, that maps the inputs to an output. The main purpose of an ANN is the mapping of input sets to desired output sets. Classical feed forward networks are optimized using supervised learning methods. In supervised learning, the expected desirable output for each set of inputs is already known and can be used in the training process to get an error signal that can be used to improve the network. The most popular algorithm to optimize ANNs is called Error-Backpropagation in which this error signal is propagated back through the network to correct the synaptic connection weights.
A physiology based method for training ANNs is Hebb's rule ([Heb49]), that describes how neuronal activities influence the connection between neurons. The basic idea is just as in the human brain that neurons that fire together will most likely strengthen or build a connection. A general model of Hebb's Rule looks like the following

$$\Delta\omega_{ij} = A * a_i * a_j + B * a_i + C * a_j + D \tag{1.1}$$

where $\Delta\omega_{ij}$ is the change in the synaptic connection between neuron i and j, a_i and a_j are the activation levels of the neurons and A, B, C and D are real numbers.
ANNs are a fundamental component in Epigenetic Neuroevolution, serving as the adaptive brain in the approaches presented in the following Chapters.

A special type of ANNs are the Compositional Pattern-Producing Networks (CPPNs), which are used in certain approaches of Epigenetic Neuroevolution. The main difference is that CPPNs use various different activation functions in their networks where standard ANNs normally mostly use sigmoidal functions. The usage of different activation functions is discussed in the HyperNEAT approach (in Section 2.1) where CPPNs are used to encode ANNs with symmetric and/or repeating topologies utilizing the different activation functions.

1.1.2 Development of the human brain and Unsupervised Learning

During the lifetime of a human it undergoes a constant evolutionary process, where each neuron or group of neurons have a specific task or react to a specific stimuli ([All99] and [Ede87]). The more incoming stimuli there are, the more strengthened the interconnections between these neurons will become. Otherwise connections will be weakened for less incoming stimuli up to the process of pruning connections. Additionally, the learning and adapting process happens online while the human is performing a task. As explained in Section 1.1.1 this is a major difference to ANNs that are trained separately during a training phase before being applied. Another difference is unsupervised learning in the human brain compared to supervised learning in ANNs. In the real world there is no expected output signal, instead the human brain uses the environmental feedback it receives to tune the synaptic connections. As shown in [WMJ+07] this feedback loop of performing an action and getting environmental feedback is an important process that drives the learning experience. While a supervised ANN can only learn what it is provided with in the training set, the human brain uses new environmental stimuli to let the brain adapt to situations that have not occurred before. All these concepts drive the motivation for Epigenetic Neuroevolution to get another step closer to an artificial brain.

1.1.3 Reinforcement Learning

Reinforcement Learning is a Machine Learning technique inspired by psychology [RSS98], where the objective is to maximize the reward an artificial agent is receiving while it tries to achieve a specified goal. This is done by exploring the environment and learning to take the correct actions in each step. A popular example is a robot trying to escape a maze where the robot gets a small negative reward for each taken step and a big reward for escaping the maze. This let's the robot learn to escape the maze as fast as possible in order to maximize the reward. In the past studies have come up with the idea that the mammalian brain may use learning methods resulting in behaviors similar to those in Reinforcement Learning ([MDS96], [WS97]). Therefore, the concept of using a reward based environment where the agent gets a positive or negative stimuli, depending on actions is picked up in Epigenetic Neuroevolution to get an environmental feedback that can be utilized to learn optimal behavior.

1.1.4 Genetic Algorithms

Genetic Algorithms (GAs) are optimization methods inspired by Darwinian Evolution (see [MTK96]). An algorithm is initialized with a population of solutions, where each solution is represented as a genome, that contains the defining characteristics as parameters. Each individual of this population is then applied to the problem and evaluated on its performance that is called its fitness. In the next step the best solutions are used to create offspring either through cross mutation of different candidates or self mutilation. This process repeats until a solution converges or a terminal condition is reached.
This is a fundamental concept in Neuroevolution, an approach to evolve weights and topology of ANNs with Evolutionary Algorithms. In Epigenetic Neuroevolution GAs are used in a similar way that will be explained in Section 2.1.

Chapter 2

Deriving Epigenetic Neuroevolution

The human brain has always been a great inspiration and motivation in machine learning. A first step towards an artificial brain was formulated with Artificial Neural Networks (ANNs). As explained in Section 1.1 ANNs consist of an aggregation of interconnected artificial neurons. A crucial difference between this model and the human brain is that the topology of an ANN is handcrafted and it is trained in advance for a specific problem. This means it is not guaranteed that the chosen topology is optimal for a specific task and that the network will produce false results for inputs not covered within training data. This is a big difference compared to learning in the human brain which undergoes an ongoing update process that takes new experiences in consideration, where constantly new synaptic connections are formed and others get pruned. With this motivation constructing and pruning algorithms have been developed to imitate this behavior. A next big step towards an artificial brain was the introduction of Neuroevolution, the concept of evolving the structure and the synaptic weights of ANNs with the use of Evolutionary Algorithms. Even though these methods are capable of solving complex tasks they are not able to handle new situations or can learn from them. After their training phase these networks are optimized for a specific task and will most likely fail for unexpected input data. The concept of Epigenetic Neuroevolution tries to go one step further by evolving more general systems that take environmental stimuli into account achieving an optimal robust behavior and adapting to- and learn from new situations. It uses therefore ANNs with special types of network encoding, Genetic Algorithms and a Reinforcement Learning like framework where the environment allocates a reward to the agent depending on it's behavior. Additionally, as it can be observed in biological brains, the change of synaptic connections is influenced by certain neuromodulatory neurons that react to particular stimuli . The following section will explain this components in more detail.

2.1 Introducing Epigenetic Neuroevolution: algorithms, formalisms, applications

A big difference between artificial brains and the biological brain is the construction and learning process. Artificial systems are usually designed using engineering experience, and or with optimization algorithms before being trained. Once the training is done the network will not change anymore, and is unable to learn anything it has not encountered during the training phase.
In the biological brain, there is no strict separation into constructing, learning, or application phase. Where in the artificial brain the process chain is sequential, the human brain performs reconstructing, learning and adaption concurrently, while performing a task. Therefore, it uses the external stimuli provided by the environment. In Epigenetic Neuroevolution this key fact is taken into account to develop a general system that is not restricted to only scenarios it has already learned, but is able to use new experienced data to adapt it's behavior. The basic concept can be describes as the following and is visualized in Figure 2.1.

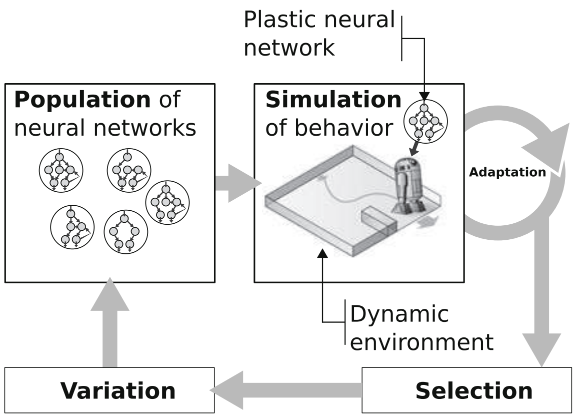

Figure 2.1: Concept of an adaptive system in Epigenetic Neuroevolution ([KBD14])

1. Initialize a population of genotypes that encodes the ANN (the phenotype)
2. Construct ANN from phenotype.
3. Simulate agent using ANN as brain. While simulating, let the network adapt to the environment.
4. Evaluate adapted ANN and select best individuals depending on a metric.
5. Mutate genotype.
6. Go-to 1.

The design of the reward metric plays a key role in this approach. A straight forward solution is to use a Reinforcement Learning like scenario (see Section 1.1) where rewards are earned depending on the performance of the agent. It will be seen in Chapter 3 that this kind of metric can be deceptive for a learning task and the idea of novelty search is introduced as an alternative strategy. The termination criteria for the algorithm explained above can be set as a reward threshold, a number of trials or if the behavior converged.

2.1.1 Network representation

An important factor in Neuroevolution is the representation of a network. There are different approaches that influence computational speed and complexity as well as they help to build modular structures and provides structures to reuse building pattern as it is known in the human brain. In the following the most important types are described.

- **Direct representation:** In this representation there is a one to one mapping from the genes to the parameters of the neural network. The genes specify e.g. how many inputs a neuron has, the synaptic weights and/or the activation function of the network. This type of encoding is easy to implement, especially for genetic algorithms, but it is costly to encode each neuron separately for big structures. The NEAT algorithm that is explained later in this section is an example that uses a direct encoding.
- **Developmental representation:** The developmental representation is inspired by the human brain and uses an encoded process in the genes that provides rules on how the neural network will be constructed. This method is suited to encode big structures with certain patterns by following the rules provided in the genes. An example of this type of encoding can be seen in the HyperNEAT algorithm explained below, where the genotype is already an ANN that specifies the rules for the connections of the phenotype.
- **Map encoding**: In map encoding the genes of the genome do not only specify single neurons but whole maps of identical neurons, that are then connected following certain patterns (mostly one-to-one or one-to-all). This is inspired by computational neuroscience to let the networks scale up to larger maps and to be able to higher dimensional inputs. A map encoding approach is evaluated in Section 3.5.

2.1.2 Plasticity

The term plasticity describes the ability of a brain to change it's structure and functionality in response to environmental stimuli during a learning process. It can be distinguished between structural plasticity, where new connections grow to change the topology of a network, and synaptic (or functional) plasticity which describes the mechanism of strengthening or weakening the connections in a nervous system. The concept of neuromodulation explained below supports adjusting the networks plasticity regarding to certain stimuli.

2.1.3 Neuromodulation

Neuromodulation describes a process where specialized neuromodulatory neurons can control the plasticity of a synaptic connection. In biology this effect is maintained by special neurotransmitters as dopamine or serotonin. In Neuroevolution this concept is used to modify the Hebbian Rule (see Equation 1.1) so that certain connection weights can only adjust when a certain stimuli is active. Using this concept the network can switch on and off the learning ability for needed situations. The standard Hebbian Rule:

$$\Delta\omega_{ij} = A * a_i * a_j + B * a_i + C * a_j + D \tag{2.1}$$

can therefore be changed to

$$\Delta\omega_{ij} = m * (A * a_i * a_j + B * a_i + C * a_j + D) \tag{2.2}$$

where m is the modulation parameter and corresponds to the sum of inputs of a modulatory neuron. This concept can be used to activate learning at a crucial point. Approaches in Chapter 3 will show the usage of neuromodulation and in Figure 2.2 the concept of a modulatory neuron can be seen.

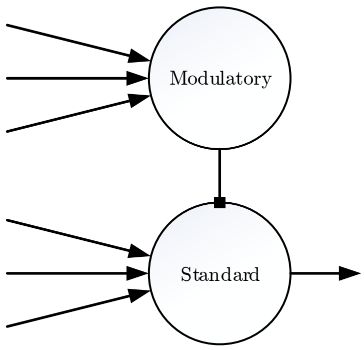

Figure 2.2: A standard neuron that is influenced by a neuromodulatory neuron

2.1.4 Novelty Search

Traditional Evolutionary Algorithms optimize systems according to an objective based fitness function. In the task of acquiring an adaptive neuronal system this can lead to certain deceptiveness. After adding a new neuron or a connection it may take some time for the system to tune this new connection during which the fitness most likely decreases. Therefore, networks may be discarded prematurely that would otherwise evolve to a reasonable solution, whereas non-adaptive systems with a higher fitness stay in the population. Novelty search is a concept which takes intermediate steps towards an optimized solution into account. Rather than searching for a final objective, in novelty search a reward is given for finding functionally different behaviors from ones that have been discovered before. In the approach presented in Chapter 3.3 Risi et Al. [RVHS09] could show that by using a novelty metric to search the space of behaviors they could outperform systems using traditional objective-based fitness functions.

2.1.5 Topology augmentation

NeuroEvolution of Augmenting Topologies (NEAT, [SM02]) is one of the most popular approaches in NeuroEvolution to evolve the topology and synaptic weights of an ANN. The goal is to find the minimal topology and the corresponding weights for a given problem. Therefore it starts from the smallest topology and increases it's complexity incrementally. The most important concepts that differentiates NEAT from earlier Neuroevolution approaches are: Tracking the gene history and protecting innovation with speciation. NEAT uses a direct encoding where each gene specifies the connection strength between two neurons. Additionally, each gene states if a connection is enabled or not and has an innovation number that shows in which mutation step the gene was added to the genome (See Figure 2.3).

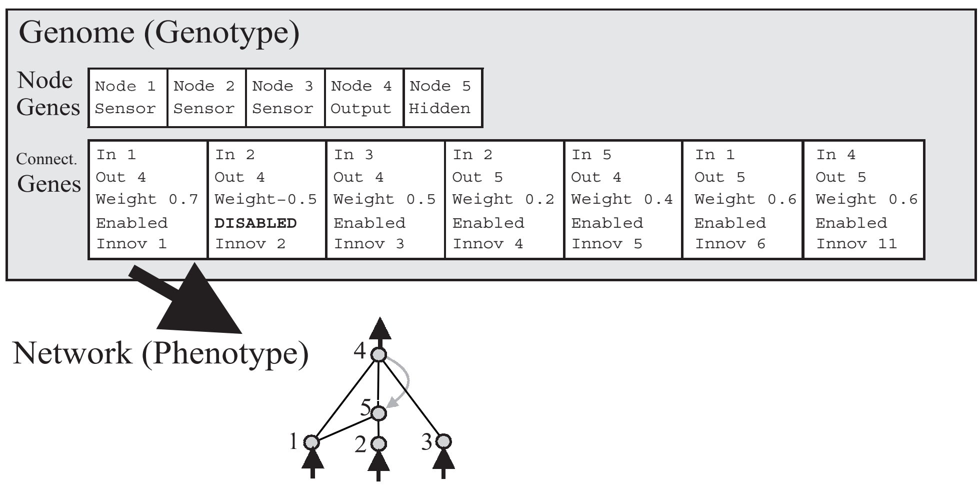

Figure 2.3: Mapping between genotype and phenotype for NEAT ([SM02])

If a genome mutates by adding a connection or a node (two connections), the genome grows by one (or two) genes with incremental innovation numbers. These innovation numbers can be used to separate all individuals into species depending on the differences of their innovation numbers and connection weights. This is done to protect new innovations that are still being tuned and would maybe discarded by bad fitness. In experiments NEAT could outperform other Neuroevolution techniques with fixed topologies for the XOR-problem and in a pole balancing test.

Hypercube-Based NEAT (HyperNEAT, [SDG09]) is an enrichment of the NEAT algorithm that uses Compositional Pattern Producing Networks (CPPNs) as an indirect encoding. The idea is that the genome should not encode each connection separately, but use CPPNs as a weight pattern generator for the network. The CPPNs can produce connection patterns with symmetries and or repeating motifs. This is achieved by transforming spatial patterns from a hyper-cube into connectivity patterns in a lower dimensional space. The main idea is to exploit the geometry of a task and map it onto the topology of a neural network. The motivation comes from the existence of symmetries and repeating patterns in the human brain, as well as the fact that nearby events in physical space are also represented by neurons in the brain that are close to each other. Additionally this structure is well suited to produce large-scale Neural Networks. The encoding of the ANN works as follows. Initially a so called substrate has to be defined as a domain for the neurons (e.g. a 2D grid as shown in Figure 2.4).

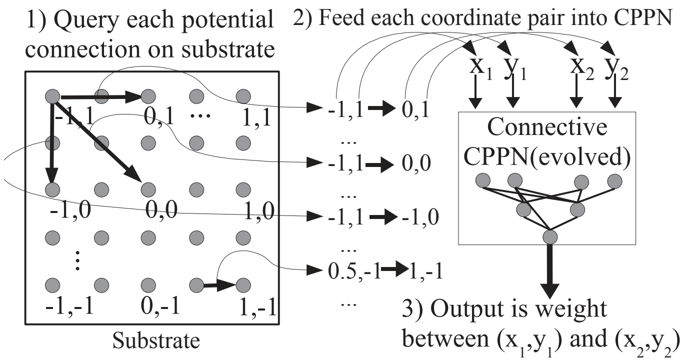

Figure 2.4: Indirect encoding of HyperNEAT; Mapping from the substrate to connection weights (from [SDG09])

The substrate should be related to the geometry of the agent or the environment in order to exploit regularities. The coordinates of two neurons can then be used as inputs for the CPPN that will return a connection weight. Therefore, the activation functions used in the CPPN can produce different connection patterns in the ANN, where Gaussians result in symmetries and periodic functions return repeating patterns. As an example if a robot has input sensors from the left to the right, the neurons could be placed in the same order on the substrate that HyperNEAT can exploit regularities (e.g. symmetry). Some examples of ANNs constructed by CPPNs are shown in Figure 2.5. The HyperNEAT algorithm follows the concept of Figure 2.1 where NEAT is used to evolve CPPNs.

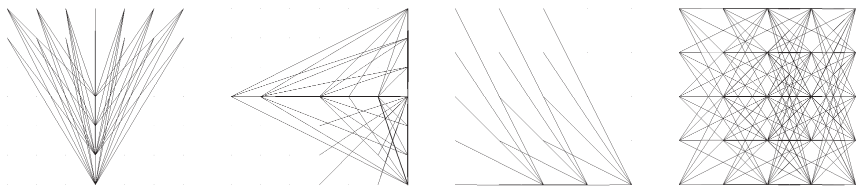

Figure 2.5: ANN topologies evolved by CPPNs in HyperNEAT from left to right: Symmetry, Imperfect Symmetry, Repetition, Repetition with variation

Chapter 3

Simulations and implementations in autonomous robotics

In this chapter different implementations of Epigenetic Neuroevolution or underlying concepts are presented to show their potential and applications that can possibly benefit from these approaches.

3.1 Robot navigation to target area with structural plasticity

In this paper Nolfi et al. [NMP94] present an early approach of a developing system by combining genetic algorithms with neural networks and phenotypic plasticity. It uses a form of developmental encoding where the genotype-to-phenotype mapping happens during the lifetime of the agent depending on environmental feedback and is guided by the rules specified in the genotype. The goal is to reach a circular target area using a little robot that is steered by two wheels with DC motors and has infrared/light proximity sensors that can detect obstacles and/or light sources. To evaluate the adaptiveness of the system two scenarios were chosen. In one scenario the target area was illuminated, where in the other it was not.

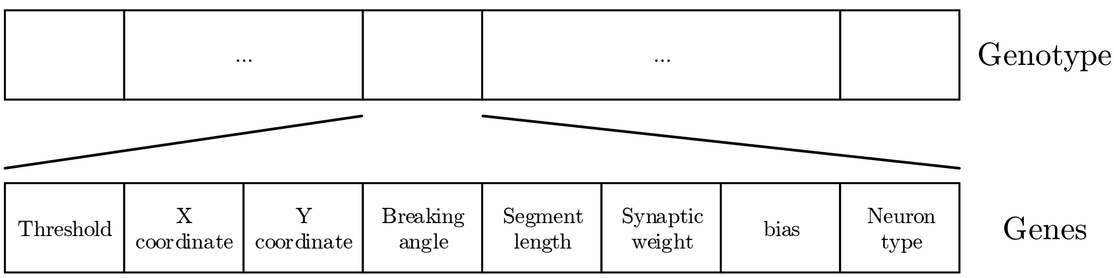

Figure 3.1: Structure of the genes and the genotype

The neural network that controls the robot uses the sensor data as inputs and has its outputs connected to the DC motors. The genotype has a fixed length where each gene defines the properties of one neuron. It is specified after how many activations an axon starts growing, the physical location of a neuron, the branching angle and segment length of an axon branch, the synaptic weight of the connection, a bias that serves as a firing threshold and the neuron type (input, hidden or motor). The development process looks like the following. Once an input neuron exceeds its expression threshold depending on incoming light stimuli, the axon starts growing. Once an axon hits another neuron a connection is established and the connected neuron will start growing an axon as well as soon as the expression threshold is exceeded. In the end all non connected axons and neurons are removed from the network. The development process can be seen in Figure 3.2. The complete algorithm follows the

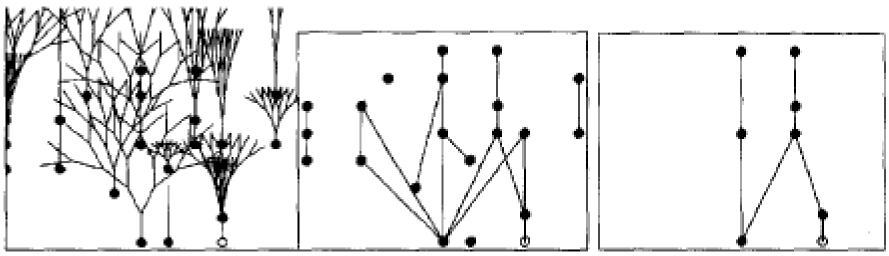

Figure 3.2: Development process of the neural network

structure presented in Figure 2.1 where the faster an agent navigates to the target area, the higher the reward is and a standard genetic algorithm was used to find the best systems.
This paper presented an early approach to mimic the developmental process of the brain, where the mapping from genotype to phenotype is not predefined, but happens during the agent's lifetime. Additionally environmental stimuli were used to let the agent adapt to the environment.

3.2 Using neuromodulation in the dangerous foraging task

In this approach Soltoggio et Al. [SDMF07] solve the so called dangerous foraging task of bees and bumblebees with a combination of Neuroevolution and neuromodulation (see Section 2.1). The goal is to maximize a bee's nectar intake by only visiting flowers with big quantities of nectar.
The simulated environment consists of a field of two different kinds of flowers with the colors blue and yellow that correspond to high and low amounts of nectar and a grey border. The simulated bees fly downwards towards the field in a random direction and get a visual input for the neural controller representing the percentages of grey, blue and yellow of the current field of view. In each time step a bee can decide if it continues on it's path or changes it into a random direction. After a

certain amount of flights the rewards of the colors are changed and with a certain probability the colors get inverted as well. The challenge is that the bee should adapt to this change to maximize the nectar income.
The neural controller consists of three input neurons that correspond to the three colors seen at each step, one reward neuron that has an input set to zero while flying and the amount of collected honey when landing and a landing indication neuron that is zero during the flight and one when landing. The landing neuron signals when a reward is incoming and can be used to calculate a prediction error. One output neuron controls the actions of the bee. The rest of the topology and the connection weights are evolved through Neuroevolution with an implicit encoding called AGE ([MF07]) including a parameter that can specify modulatory or standard neurons.

The results show that the networks with neuromodulation clearly outperform the ones without. After a switch of the rewards the bees need a certain number of flights to change their strategy but can always achieve the correct association between color and high rewards in the end. An illustration of a well-performing network is given in Figure 3.3.

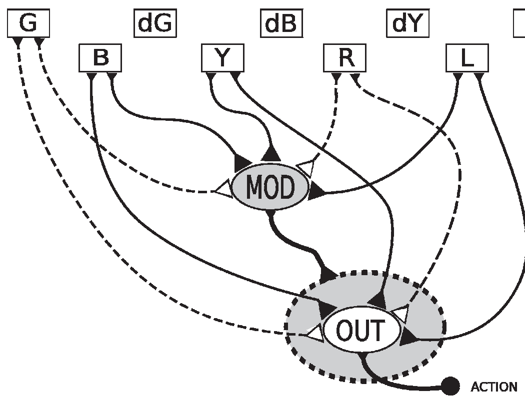

Figure 3.3: A well performing ANN with neuromodulation. The differential inputs were not used in this approach and the 1 stands for a constant bias. Additionally the dashed lines stand for inhibitory neurons.

3.3 Using novelty search for faster converging in T-Maze environment

In this work Risi et Al. [RVHS09] compare the concept of novelty search with fitness based evolution. As mentioned in Section 2.1 the problem to obtain learning systems with fitness based objective functions is that adding plastic synapses to an ANNs a priori will often decrease the fitness of a system. The objective of novelty search is to find adaptive behavior as fast as possible by exploring the

behavioral space. Therefore a modified version of the NEAT algorithm including neuromodulation is evaluated by its behavioral novelty and is used to address T-Maze domain. In the standard fitness based approach the fitness of an agent is evaluated by summing all collected rewards over all trials. This means if two agents collect a different amount of reward per trial, but collect the same amount in total their behavior cannot be distinguished, even though they most likely used completely different strategies. Novelty Search introduces a metric called novelty distance that sums up the different outcomes of each trial of two agent to obtain the behavioral difference of two agents. This information can be fed into the NEAT algorithm to further distinguish the different species of its ANNs. This method proofed to be very performant to effectively search for new behaviors.
The results show that the novelty approach outperformed the fitness based approach by far (1.4 - 2.0 times faster).

3.4 Indirectly encoding neural plasticity as a pattern of local rules

Risi et Al. [RS10] show an enhancement for the previously discussed HyperNEAT algorithm called adaptive HyperNEAT in this paper. Instead of encoding patterns of synaptic connection weights as seen in HyperNEAT, adaptive HyperNEAT encodes patterns of learning rules. The idea is again inspired by biological brains where the synaptic plasticity is not encoded in the DNA for each synapse separately.
Throughout this paper mainly two concepts for learning rules are investigated:
A general iterated model that enhances the four dimensional CPPN (known from [SDG09]) by including the post- and pre-synaptic weights a_i and a_j and the current connection weight ω_{ij}. This means the CPPN is not only queried in the evolutionary phase (as in HyperNEAT), but also in each simulation step to update the connection weights. The weight update for the synaptic plasticity can be represented as

$$\omega_{ij} = CPPN(x_1, y_1, x_2, y_2, a_i, a_j, \omega_{ij}) \tag{3.1}$$

A less general rule is specified by the Hebbian ABC model where the learning rule looks as the following:

$$\omega_{ij} = \eta * (A * a_i * a_j + B * a_i + C * a_j) \tag{3.2}$$

In this case the inputs of the CPPN stay the same as in HyperNEAT, but A, B and C are added to the outputs of the CPPNs. Figure 3.4 shows the two models side by side. Even though this model is less general than the iterated model, it involves less computational complexity since the CPPN has not to be queried during the simulation.
The evaluation is done with two T-Maze scenarios, where the wings of the maze have one high and one low reward and a fitness based objective function is used.

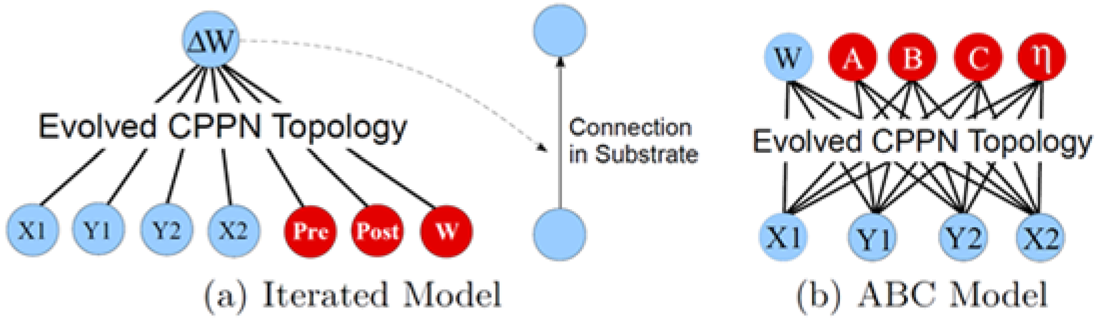

Figure 3.4: The two models of the adaptive HyperNEAT algorithms to evolve ANNs.

Scenario 1 uses the rewards 0.1 (Blue) and 1.0 (Red) and scenario 2 adds two additional values: 0.3 (Green) and 0.8 (Yellow). The first deployment is done with the settings of scenario 1 and in the second deployment the two intermediate rewards are introduced. The new reward values give the reward a signature that is not linearly separable. It should be noted that the ANNs in this experiment have no hidden nodes, which means the learning rule needs to be nonlinear to cope with this problem.
Scenario 1 can be solved by both learning rules with about the same number of generations whereas, in Scenario 2 the iterated model can solve the task and where the ABC struggles to achieve the fitness threshold for a successful run. The assumption is that with this learning rule a hidden layer of neuron would be needed to succeed in this task.
This enhancement of the HyperNEAT model to include patterns learning rules proofs to be another step towards a biologically inspired artificial brain. The iterated model shows learning capabilities with a high adaptation to environmental changes. On the other hand the ABC model in connection with an appropriate ANN structure can probably cope with most domains and shows a lower computational complexity than the iterated approach. It has to be noted that this approach could as well be combined with the concepts of neuromodulation and novelty search (presented in Section 2.1) to further enhance the system.

3.5 Synaptic General Learning Abilities and Synaptic Transitive Learning Abilities

In this paper Tonelli et Al. [TM13] formulate a framework to investigate the learning capabilities of ANNs for scenarios they have not experienced before. Additionally three different approaches are evaluated.
The framework is formulated as the following:

$N(I, \lambda)$:	ANN where I is an Input set and λ is the set of synaptic weights providing the maximal reward
K:	Best rewarded output vector
(I, K):	An association: A pair of in- and outputs that maximizes the reward
$R_{I,K}$:	Reward function.
$A = \{(I_1, K_1), ..., (I_n, K_n)\}$:	Association set - list of associations including all input patterns.
$\mathbb{A}$:	Set of all possible Association sets.
$\mathbb{L}_N$:	Learnable set: A set A is learnable by N, if and only if $\forall \lambda_R \in \mathbb{R}^Z$ and $\forall (I, K) \in A, \exists \lambda = g(\lambda_R, I, R_{I,K})$ such that $N(I, K) = K$ where λ_R is a random weight vector.

synaptic General Learning Ability (sGLA): To possess sGLA, an ANN must be able to learn each possible association of stimulus/action with the same topology, the same learning function but a different reward scheme.
The network $N(I, \lambda)$ must adjust its weights λ_R such that each input pattern I is associated to K. The weights are adjusted according to $\lambda = g(\lambda_R, I, R_{I,K})$
A plastic ANN is said to to possess synaptic General Learning Abilities, if and only if $\forall A \in \mathbb{A}, A \in \mathbb{L}_N$

synaptic Transitive Learning Ability (sTLA): An ANN has sTLA if and only if it has sGLA and the list of association sets used in the evolutionary phase of $N(I, \lambda)$ is a subset of $\mathbb{A}$.
The number of association sets used in the evolutionary phase is then called the sTLA-level. Additionally a network posses optimal sTLA for an sTLA-Level of 1.

In the evaluation section, an approach with map based encoding (see [MDG10]) combined with a NEAT-inspired algorithm is investigated for learning abilities and compared with a simplified HyperNEAT algorithm and an algorithm using direct encoding. The task to solve was the so called 'Skinner Box' where an agent sits in a box and is given a certain stimuli (in this case a light switching on). When the stimuli is present the agent has to perform a certain action (in this case pushing one of four leverages). If the action was correct, the agent gets a positive reward and if it is incorrect, it gets punished with a negative reward.
For the evaluation each approach (direct and map encoding) is provided with a certain amount of association sets and is then tested if it is able to learn new (never before experienced) sets. It could be shown that the map based encoding approach has optimal sTLA for this approach, where the sTLA level for the direct encoding was about 7 to reach the same results as for the map encoding one. The HyperNEAT approach lies in between.

Chapter 4

Applying Epigenetic Neuroevolution: Maintaining forward motion for damaged robot

In this chapter an imaginary scenario to apply the previously discussed methods is investigated. The Epigenetic Neuroevolution approach is utilized to maintain the desired motion of a damaged exploration robot.

4.1 Set-up

Interpreting the steering commands of a robot and translating them into the right motor commands is normally a straightforward task. In the imagined scenario however, we assume an environment that can damage the robot. For example falling rocks, debris, unexpected heat or radiation that influences the behavior or shape of some parts. We are investigating cases where the wheels, the suspension or the servo motors are damaged or completely broken. This damage can lead from non turning wheels to pieces that scrape along the ground and influence the motion of the robot up to unstable movement (image the front right and back left wheel fallen of a 4 wheeled robot). The idea is that the neural network produced by Epigenetic Neuroevolution approaches should adapt and change the mapping between steering commands and motor commands to maintain a correct translation from the actual commands to the executed movement of the robot. The robot is a 6 or 8 wheeled rover that navigates either with autonomous path planning or a human steering the robot. However, in both cases the robot receives some sort of steering commands that should result in a desired movement. The current position and orientation are measured via GPS or are assumed to be known.

4.2 Network structure and algorithms

The inputs for the network are the steering commands and a drift signal. The drift signal computes a distance between the actual position of the robot and the desired position for the non damaged robot. For each motor there is one output neuron. As a basis for the algorithm two possibilities can be chosen. Either the map based encoding that is tested in Section 3.5 or adaptive HyperNEAT (see Section 3.4). Both of those algorithms have proven to have sTLA and should therefore be suitable for the task. In both approaches neuromodulation is enabled and novelty search is used as those concepts showed to further improve the performance. For the HyperNEAT approach a substrate has to be defined. An example structure to exploit the geometry of the robot can be seen in Figure 4.1, where I are the input neurons and O are output neurons.

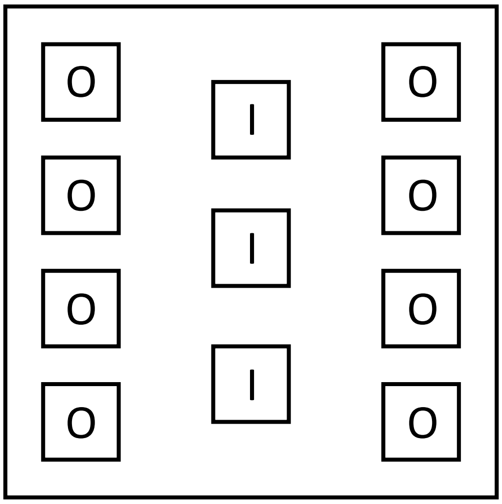

Figure 4.1: Example substrate for HyperNEAT with inputs (I) and outputs (O)

4.3 Training and evaluation

The training environment features a map where start and goal regions can be set and it can cover different cases of damage. Each motor can be switched off, the suspension of a wheel can be set to stiff, the wheels can be unmounted or replaced by a telescoping rod. These cases simulate broken motors or suspension, fallen of wheels or twisted parts scraping along the ground.
The training and evaluation works similar as in Figure 2.1 and as described in [TM13]. A genotype is tested for a certain amount of scenarios consecutively, where each scenario is a mix of different types of damage. Once the network solves the task (reaches a high enough fitness), the next scenario is presented. The reward is generated by summing up the drift signal in each step and receives a (negative)

bonus for reaching the target area. Once genotypes are found that can adapt to and solve the test scenarios, they are applied to new scenarios, not covered in the test scenarios and investigated if they can solve the unknown task. In the end the approach that needs the least training scenarios to succeed (and therefore has the best sTLA) is selected as solution.

4.4 Possible extensions

The approach has several possibilities to be extended. It could be investigated if the agent is able to adapt to varying surface textures or what improvement would make this possible. Another improvement could be to include one input neuron for each output neuron and establish a feedback connection to let the network learn from the last taken actions as well. An obvious enhancement is also to further analyze the geometry of the task and develop different surface structures for the HyperNEAT approach.

Chapter 5

Discussion

The discussed methods and approaches in this report give a rough overview of the current progress in Epigenetic Neuroevolution. The bio-inspired concepts that have been transferred to artificial systems have shown to clearly increase performance of previous developed systems. The usage of environmental feedback has proven to be a key concept to let an agent learn online and adapt to situations that have not been encountered during the evolutionary phase. Another important component is the encoding (as discussed in Section 2, where the developmental representation as it has been seen in [SDG09]) and [MDG10]) is currently the most promising approach. Additionally, by the concept of GLAs a measurement has been presented (in Section 3.5) to evaluate the learning performance of a system and enable comparing different approaches.

Altogether there exists a promising framework to further enhance the concepts to build an artificial brain with learning abilities. However, it has to be noted that the current systems are often designed for rather small problems and are often still limited by computational complexity. Should the structures be small and adapt fast, or big and general? Part of future work could therefore be investigating approaches to reduce complexity without losing generality. Another challenge to address is the exact connection between the plasticity, the encoding and learning abilities.

List of Figures

Bibliography

[All99] John Allman. *Evolving Brains.* Scientific American Library New York, 1999.

[Ede87] Gerald M Edelman. *Neural Darwinism: The theory of neuronal group selection.* Basic Books, 1987.

[Heb49] Donald Olding Hebb. *The organization of behavior: A neuropsychological theory.* Wiley, 1949.

[KBD14] Taras Kowaliw, Nicolas Bredeche, and Ren Doursat. *Growing Adaptive Machines: Combining Development and Learning in Artificial Neural Networks.* Springer Publishing Company, Incorporated, 2014.

[MDG10] Jean-Baptiste Mouret, Stéphane Doncieux, and Benoît Girard. Importing the computational neuroscience toolbox into neuro-evolution-application to basal ganglia. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 587–594. ACM, 2010.

[MDS96] P Read Montague, Peter Dayan, and Terrence J Sejnowski. A framework for mesencephalic dopamine systems based on predictive hebbian learning. *The Journal of neuroscience*, 16(5):1936–1947, 1996.

[MF07] Claudio Mattiussi and Dario Floreano. Analog genetic encoding for the evolution of circuits and networks. *Evolutionary Computation, IEEE Transactions on*, 11(5):596–607, 2007.

[MM69] Seymour Papert Marvin Minsky. *Perceptrons: An Introduction to Computational Geometry.* M.I.T. Press, 1969.

[MTK96] Kim-Fung Man, Kit-Sang Tang, and Sam Kwong. Genetic algorithms: concepts and applications. *IEEE Transactions on Industrial Electronics*, 43(5):519–534, 1996.

[NMP94] Stefano Nolfi, Orazio Miglino, and Domenico Parisi. Phenotypic plasticity in evolving neural networks. In *From Perception to Action Conference, 1994., Proceedings*, pages 146–157. IEEE, 1994.

[RS10] Sebastian Risi and Kenneth O Stanley. Indirectly encoding neural plasticity as a pattern of local rules. In *From Animals to Animats 11*, pages 533–543. Springer, 2010.

[RSS98] Andrew G. Barto Richard S. Sutton. *Reinforcement Learning: An Introduction.* M.I.T. Press, Cambridge, 1998.

[RVHS09] Sebastian Risi, Sandy D Vanderbleek, Charles E Hughes, and Kenneth O Stanley. How novelty search escapes the deceptive trap of learning to learn. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 153–160. ACM, 2009.

[SDG09] Kenneth O Stanley, David B D'Ambrosio, and Jason Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artificial life*, 15(2):185–212, 2009.

[SDMF07] Andrea Soltoggio, P Durr, Claudio Mattiussi, and Dario Floreano. Evolving neuromodulatory topologies for reinforcement learning-like problems. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 2471–2478. IEEE, 2007.

[SM02] Kenneth Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127, 2002.

[TM13] Paul Tonelli and Jean-Baptiste Mouret. On the relationships between generative encodings, regularity, and learning abilities when evolving plastic artificial neural networks. *PloS one*, 8(11):e79138, 2013.

[WMJ+07] Gert Westermann, Denis Mareschal, Mark H Johnson, Sylvain Sirois, Michael W Spratling, and Michael SC Thomas. Neuroconstructivism. *Developmental science*, 10(1):75–83, 2007.

[WS97] P. Read Montague Wolfram Schultz, Peter Dayan. A neural substrate of prediction and reward. *Science*, 275(5306):1593–1599, 1997.

[WSM43] Walter Pitts Warren S. McCulloch. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

License