Evolution and learning in artificial ecosystems

Evolution and Learning in Artificial Ecosystems

Claes Strannegård[1], **Wen Xu**[1], **Niklas Engsner**[1], **John A. Endler**[2]
[1] Computer Science and Engineering, Chalmers University of Technology, Gothenburg, Sweden
[2] Centre for Integrative Ecology, Deakin University, Waurn Ponds, Australia
claes.strannegard@chalmers.se, wenx@student.chalmers.se,
niklas.engsner@gmail.com, john.endler@deakin.edu.au

Abstract

A generic model is presented for ecosystems inhabited by artificial animals, or *animats*, that develop over time. The individual animats develop continuously by means of generic mechanisms for learning, forgetting, and decision-making. At the same time, the animat populations develop in an evolutionary process based on fixed mechanisms for sexual and asexual reproduction, mutation, and death. The animats of the ecosystems move, eat, learn, make decisions, interact with other animats, reproduce, and die. Each animat has its individual sets of homeostatic variables, sensors, and motors. It also has its own memory graph that forms the basis of its decision-making. This memory graph has an architecture (i.e. graph topology) that changes over time via mechanisms for adding and removing nodes. Our approach combines genetic algorithms, reinforcement learning, homeostatic decision-making, and dynamic concept formation. To illustrate the generality of the model, five examples of ecosystems are given, ranging from a simple world inhabited by a single frog to a more complex world in which grass, sheep, and wolves interact.

1 Introduction

Stuart Wilson defined *animats* as a type of artificial animals, whose sole goal, from the individual's perspective, is homeostasis [Wilson, 1986]. He also suggested the *animat path to AI* as a way of creating artificial intelligence by modeling animal behavior, which is a wider notion than natural intelligence [Wilson, 1991]. In this paper we propose to follow the animat path to AI by simultaneously modeling two fundamental processes underlying animal intelligence: evolution, which operates at the population level between generations, and learning, which operates at the individual level. Both these processes are fundamental to the ability of animal populations to adapt and survive in new environments. Ecological models that omit either evolution or learning are obviously severely limited in their predictive power. Nonetheless, ecosystem models that combine evolution and learning are relatively rare. Ecology and evolution were modeled jointly in [Hubbell, 2001], but learning was not included in the model.

Animals can learn via mechanisms for altering their nervous systems in response to changes in the environment (neuroplasticity). One fundamental type of learning is structural learning, which changes the topology of the graph by adding and removing neurons [Draganski and May, 2008]. The goal of this paper is to suggest a computational model for artificial animals that combines evolution and structural learning.

There are multiple reasons for building and studying artificial ecosystems. One is to increase understanding of how natural ecosystems operate. Another is to predict how ecosystems might be affected by human activity and e.g. adapt harvesting of natural resources to sustainable levels. A third reason -and this is our prime interest here- is to create artificial intelligence by modeling a small number of cognitive mechanisms of animals. Since intelligence has developed in real ecosystems, this idea is not far-fetched. Widening the scope from models of animals to models of ecosystems gives an adequate setting for developing artificial intelligence by following the animat path to AI.

1.1 Artificial ecosystems

In analytical approaches to ecosystem modeling, animal populations are frequently modeled with numbers representing biomass and the interaction dynamics between different population groups is modeled with systems of differential equations. A well-known example are the Lotka-Volterra equations for predator-prey dynamics [Lotka, 1925].

Biomass plays a central role also in simulation-based ecosystem models. For instance, models created in the Ecopath/Ecosim/Ecospace simulation environment for marine ecosystems [Christensen and Walters, 2004] might include variables for catch, predation, and net migration, all measured in tonnes of certain fish species.

In ecosystem models, learning is typically left out of the picture. Yet it is known that the ability to learn is closely correlated to fitness [Ashton *et al.*, 2018]. Thus, excluding learning from the ecosystem models might lead to substantial inaccuracy. On the other hand, by including learning, the models become more accurate and topics such as communication, cooperation, competition, and behavior in general can be studied in a more realistic setting and at a greater level of detail.

1.2 Evolution

Evolution is a natural process that has generated the phylogenetic tree of life as well as all animal behavior, including human intelligence [Futuyma, 2009]. In computer science, evolutionary computation is a family of algorithms for global optimization that attempt to imitate natural evolution. Of particular interest are the evolutionary algorithms that use mechanisms such as reproduction, mutation, and selection. Candidate solutions to the optimization problem are, e.g. represented as binary strings, and these strings are then subject to an evolutionary process with cross-over, mutation, etc. Evolutionary algorithms often use fitness functions for selecting solution candidates for reproduction, but in more realistic scenarios, where reproduction depends on who meets who and the like, handcrafted fitness functions are not needed. A successful technique of evolutionary computation is genetic programming [Koza, 1994], in which the genome encodes computer programs. Evolutionary computation has been studied in its own right, but also in the context of artificial life [Langton, 1997; Tuci *et al.*, 2016].

1.3 Learning

Nervous systems are ubiquitous in the animal kingdom and play a fundamental role in natural intelligence. Successful artificial neural network techniques include deep learning [LeCun *et al.*, 2015], Long Short-Term Memory [Hochreiter and Schmidhuber, 1997] that can store arbitrary values for arbitrary long time-periods, and Neural Turing machines, with their great generality but also slow convergence and need for external memory resources [Zaremba and Sutskever, 2015].

Artificial neural network models are frequently based on static architectures that are only plastic in the sense that their connectivity patterns develop over time. Several neural network models also allow nodes to be added and removed, however. For instance, the cascade-correlation architecture adds one hidden neuron at the time [Fahlman and Lebiere, 1990] and the progressive neural networks grow new columns while retaining previously acquired knowledge [Rusu *et al.*, 2016]. In the opposite direction there are regularization techniques [Goodfellow *et al.*, 2016] and pruning methods [Wolfe *et al.*, 2017] for reducing the size of neural networks, while improving generalization.

Reinforcement learning occurs across the animal kingdom and its biological basis is well understood [Niv, 2009]. Reinforcement learning algorithms, on the other hand, are powerful tools for learning and decision-making in a general setting [Sutton and Barto, 1998]. Q-learning is a basic algorithm for learning an optimal policy from experience for any Markov Decision Process [Watkins, 1989]. Other reinforcement learning algorithms included tree-based methods running in batch-mode [Ernst *et al.*, 2005] and local Q-learning, where Q-values collected from multiple agents are merged into a global Q-value [Russell and Zimdars, 2003]. A way of combining conjunctions of basic features in reinforcement learning is described in [Buro, 1998].

Reinforcement learning algorithms have also been used for homeostatic agents, whose single objective is to regulate their homeostatic variables and thus survive as long as possible [Keramati and Gutkin, 2011; Yoshida, 2017]. Homeostatic decision-making aims at keeping the "body fluids" in balance and combines naturally with models for hormonal control [Avila-García and Cañamero, 2005], cognitive modulation [Bach, 2015], and personality traits [Bouneffouf *et al.*, 2017]. In the case of multiple homeostatic variables, representing, e.g. water and energy, it may be natural to use multi-objective reinforcement learning [Roijers *et al.*, 2013].

Evolution and learning have been combined in computer models, e.g. when genetic algorithms were used for developing architectures of deep neural networks [Such *et al.*, 2017] and for evolving cognitive architectures within the MicroPsi framework [Dietzsch, 2008].

1.4 Structure of the paper

This paper presents a novel computational model of ecosystems that uses several elements and techniques from the research fields mentioned above. We suggest a fixed set of generic mechanisms for perception, action, learning, decision-making, reproduction, and death.

The paper extends our previous work [Strannegård *et al.*, 2017] by adding more sophisticated rules for architecture development, by widening the scope from agents to ecosystem of agents, and by including genetic operations. Section 2 defines our notion of graph. Section 3 defines the animats and Section 4 introduces the artificial ecosystems. Sections 5 and 6 describe how the animats learn and make decisions, respectively, while Section 7 describes how the animats reproduce. In Section 8, some examples of ecosystem simulations are given. Section 9, finally, draws some conclusions.

2 Graphs

In this section we define our own variety of graphs, which can can alternatively be described as sets of sentences of temporal logic [Gabbay *et al.*, 1994]. We also describe how activity propagates through these graphs.

2.1 Definition of graph

Let us start with a model of the sensory apparatus.

Definition 1 (Sensor, sensor set, concept)**.** *The sensory notions are defined as follows:*

A sensor *is a propositional variable.*

A sensor set *is a finite set of sensors.*

A concept *is a finite sequence of sensor sets.*

Thus, sensors, sensor sets, and concepts are all defined mathematical objects here. As we shall see later, these objects can be activated by stimuli from the environment. For instance, a concept will be activated when a certain sequence of stimuli is received. Moreover, a sensor p will be active if and only if the concept $[\{p\}]$ is active. We use the notation $\{\ldots\}$ for sets and $[\ldots]$ for sequences.

Example 1. *Here are some examples of these notions:*

- *Suppose red, green, and blue are sensors for colors.*
- *Then* $\{\}$, $\{green\}$, $\{blue, green\}$, *and* $\{red, blue, green\}$ *are sensor sets representing, respectively, black, green, turquoise, and white.*
- *Moreover,* $[\{\}]$, $[\{green\}]$, $[\{blue, green\}]$, *and* $[\{red\}, \{red\}, \{green\}]$ *are concepts that represent different color sequences.*

Concepts based on other modalities such as tastes, smells, sounds, and touch can be defined analogously. It is also straight-forward to define multi-modal concepts. Now let us turn to the motor apparatus, which is symmetric to the sensory apparatus defined above.

Definition 2 (Motor, motor set, action)**.** *The motor notions are defined as follows:*

- *A* motor *is a propositional variable.*
- *A* motor set *is a finite set of motors.*
- *An* action *is a finite sequence of motor sets.*

Example 2. *Here are some examples of these notions:*

- *left and right are motors for moving the left and right pectoral fins of a fish robot.*
- $\{\}$, $\{left\}$, *and* $\{left, right\}$ *are motor sets representing, respectively, idleness, moving the left fin only, and moving both fins.*
- $[\{right\}, \{left\}]$ *and* $[\{right, left\}]$ *are examples of actions. In the former, one fin is moved at a time and in the latter both fins are moved simultaneously.*

If x is a concept or an action we will write $y \in x$, if $x = [x_1, \ldots, x_n]$ and $y = x_i$ for some i.

Definition 3 (Reflex)**.** *A* reflex *is a pair (c, a), where c is a concept and a is an action.*

Again we take an existing term and give it a mathematical meaning without connotations. Intuitively, these reflexes represent neural connections that link sensory notions to motor notions. The human knee reflex is an example: when certain sensory receptors at the knee are activated, a nerve signal is transmitted to motor neurons that cause the leg muscles to contract and produce a kick. Other examples are the sucking and rooting reflexes in babies, both of which facilitate breastfeeding. Now we are ready to define our graphs.

Definition 4 (Graph)**.** *Let C be a finite set of concepts, A a finite set of actions such that $C \cap A = \emptyset$, and $R \subseteq C \times A$ a set of reflexes. Let $C_{set} = \{s : x \in C$ and $s \in x$ for some $x\}$, $C_{sensor} = \{s : x \in C_{set}$ and $s \in x$ for some $x\}$, $A_{set} = \{s : x \in A$ and $s \in x$ for some $x\}$, $A_{motor} = \{s : x \in A_{set}$ and $s \in x$ for some $x\}$. Let the* graph *Gbe defined as the graph $G = (C_{sensor} \cup C_{set} \cup C \cup A \cup A_{set} \cup A_{motor}, E)$, where $E(x, y)$ holds if one of the following conditions is fulfilled:*

- *$x \in y$ and $y \in C_{set}$,*
- *$x \in y$ and $x \in C$,*
- *$(x, y) \in R$,*
- *$x \in A$ and $y \in x$,*
- *$x \in A_{set}$ and $y \in x$.*

Thus each graph forms a feed-forward graph with six layers of nodes. An example of a graph is given in Figure 1.

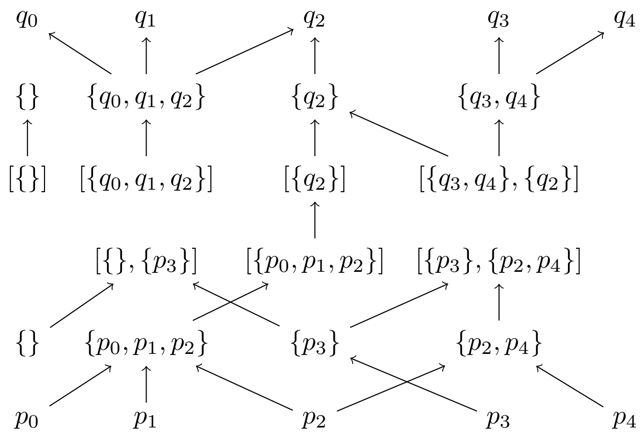

Figure 1: Example of a graph. The layers are from bottom to top: sensors, sensor sets, concepts, actions, motor sets, and motors. This graph has one reflex.

2.2 Graph activity

In this subsection we define how activity propagates through the graphs along the edges. Although real animals live in continuous time, we will use a simplified model where time proceeds in discrete steps called ticks.

Definition 5 (Input stream)**.** *An* input stream *for a graph G is a function that assigns a boolean value $p(t)$ to each sensor p and point in time t.*

Using boolean values for inputs is in line with the McCulloch–Pitts model of neurons that either fire all out (when the activity exceeds a threshold) or not at all. For graded responses one may use several neurons, e.g. one that fires when the pH is 6.2, another that fires when it is 6.3, etc. Now we can define how input streams give rise to activity that propagates through the graphs:

Definition 6 (Graph activity)**.** *This is how activity propagates through a given graph:*

- *A sensor p is* active *at t if $p(t)$ is true.*
- *A sensor set S is* active *at t if every sensor $p \in S$ is active at t.*

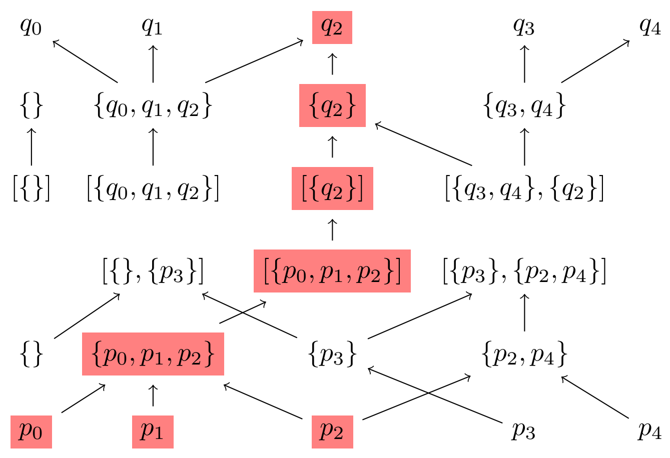

Figure 2: Example of graph activity. Activity is shown in red. The activity propagates from three sensors to a motor via a reflex within the same tick.

- *A concept $[S_0, \dots, S_n]$ is* active *at t if for every i such that $0 \le i \le n$, the sensor set S_i was active at $t - (n - i)$.*
- *An action a is* active *at t if there is a reflex (c, a), where c is active at t or if $\pi(t) = a$. Here $\pi(t)$ is the policy, which will be defined in Subsection 6.*
- *A motor set M_i is* active *at t if there is an action $[M_0, \dots, M_n]$ that was active at $t - i$.*
- *A motor q is* active *at t if $q \in M$, for some motor set M that is active at t.*

An example of graph activity is given in Figure 2.

Example 3. *For instance, the concept $[S_0, \dots, S_{10}]$ is active now if S_0 was active 10 steps ago and S_1 was active 9 steps ago and ... and S_{10} is active now. Also, the motor set M_4 is active now if the action $[M_0, \dots, M_{10}]$ was active 4 steps ago.*

2.3 Top activity

In this subsection we will introduce the notion of top activity, which plays a key role later for both decision-making and learning. First we introduce a partial order relation $\succeq$ on concepts:

Definition 7 (Extension)**.** *The concept $[S'_0, \dots, S'_n]$* extends *the concept $[S_{n-m}, \dots, S_n]$ if $n \ge m$ and $S'_i \supseteq S_i$, for all i such that $n - m \le i \le n$. If c' extends c, we write $c' \succeq c$. If $c' \succeq c$, but not $c \succeq c'$, then we write $c' \succ c$.*

Intuitively, $c' \succeq c$ means that c' is a more detailed concept than c.

Example 4. *Here are two examples of extensions concerning, respectively, tastes and melodies:*

- $[\{sweet, sour\}] \succ [\{sweet\}]$
- $[\{C\}, \{C\}, \{C\}, \{D\}, \{E\}] \succ [\{D\}, \{E\}]$

It is easy to see that the relation $\succeq$ forms a partial order on the set of concepts of a given graph.

Definition 8 (Top activity)**.** *Let C be a set of concepts. A concept $c \in C$ is* top active *at t if c is active at t and c is $\succ$-maximal with respect to the active concepts in C.*

Intuitively, the top active concepts constitute a description of the present situation at a maximum level of detail with respect to a given set of concepts.

Example 5. *Here are some examples of top activity:*

- *Suppose the concept $[\{sweet, sour\}]$ is top active. Then $[\{sweet\}]$ is active, but not top-active.*
- *Suppose the concept $[\{C\}, \{C\}, \{C\}, \{D\}, \{E\}]$ is top active. Then $[\{D\}, \{E\}]$ is active, but not top active.*
- *Suppose a graph contains the concepts budgerigar $\succ$ parrot $\succ$ bird and no others. When budgerigar is active, it will also be top-active, while the other concepts will also be active, but not top active.*

Several nodes can be top active at the same time. In Subsection 6 we will select actions based on experience associated with those nodes that are currently top active.

Before moving on, let us consider an example that illustrates the expressive power of the graphs.

Example 6. *A physical robot that plays a simple digital piano with 88 keys can be constructed by using a graph as follows: Let $p_1, p_2, \dots, p_{88}$ be sound sensors for the 88 keys and let $q_1, q_2, \dots, q_{88}$ be motors that press the corresponding piano keys. For simplicity we assume that the robot can play any number of keys at the same time. Now we can construct a graph with one concept for every piece of piano music that we want the robot to recognize and one action for every piece that we want the robot to be able to play. When a piece of music that the robot has stored in its graph is played by someone else, the corresponding node of the robot's graph will be top active. When an action that represents a given piece of music is activated, the robot will play that particular piece.*

3 Animats

In this section animats will be defined.

Definition 9 (Homeostatic variable)**.** *A homeostatic variable is a variable among $h_0, h_1, \dots$.*

We will use the symbol h for arbitrary homeostatic variables. Later we will see that each homeostatic variable h receives a real value $h(t) \in [0, 1]$ from the environment at each point in time t. Animals typically have physiological needs such as water, energy, and warmth. These needs often have associated interoceptors that indicate their status. Here are some examples of needs and their associated interoceptors: Water (osmoceptors); Energy (insulin receptors); Protein (amino acid receptors); Oxygen (CO_2 receptors); Integrity, i.e. freedom from pain (nociceptors); Sleep (melatonin receptors);

Heat (thermoreceptors); Proximity (pheromone receptors); Reproduction (sexual hormone receptors); Affiliation (oxytocin receptors). Next we will define the experience variables that will be used for storing various experiences.

Definition 10 (Experience variable)**.** *The experience variables are the following:*

- *age. This is an integer that represents the number of ticks since birth. Negative values will be used for the incubation time before birth, which happens at time 0.*
- *alive. This is a boolean value that tracks whether the animat is alive.*
- $Q_h(c, a)$. *These local Q-values are real values that estimate how good it is from the perspective of h to do action a when c is top-active. These values are initialized to 0.*
- $R_h(c, a)$. *These reliability values are real values that estimate the reliability of* $Q_h(c, a)$ *based on its historical standard deviation. These values are initialized to 1.*

Definition 11 (Parameter)**.** *A parameter is a real number encoded as a bitstring of type float.*

Parameters are used for regulating learning, decision-making, reproduction, and death. Examples of parameters are $MaxAge$, $MutationRate$, $DiscountRate$. There are also parameters such as Sex and $SexualReproduction$ that encode boolean values via the convention that positive values represents $True$, whereas non-positive values represent $False$. We will introduce the specific parameters that belong to the animat model gradually as the need to do so arises in the text.

Definition 12 (Genotype)**.** *A* genotype *consists of*

- *a graph*
- *a set of homeostatic variables*
- *a set of parameters.*

The genotype remains constant throughout the lifetime of the animat. It is used for specifying what the animat is like at birth. It also specifies the part of the animat that is involved in reproduction.

Definition 13 (Phenotype)**.** *A* phenotype *consists of*

- *a graph*
- *a graph activity*
- *values of the homeostatic variables*
- *values of the experience variables.*

The phenotype changes constantly throughout the life of the animat. It is used for specifying what the animat is like at present, save for its genetic information and its conformation.

Definition 14 (Conformation)**.** *A* conformation *is a subset of* $\mathbb{R}^3$*.*

For instance, a conformation can describe the location and body position of an animal. It can equally well describe the location and geometry of a dead object, like a rock.

Definition 15 (Animat)**.** *An* animat *consists of:*

- *a genotype*
- *a phenotype*
- *a conformation.*

The genotype remains constant over the animat's life, whereas the phenotype and conformation typically vary at each tick.

4 Ecosystems

Definition 16 (Object)**.** *An* object *consists of:*

- *a type: a natural number. The types could, e.g. represent rock, earth, sand, air, water, or carcass.*
- *a conformation*

Definition 17 (Ecosystem)**.** *An* ecosystem *is a pair* $(\mathcal{A}, \mathcal{O})$*, where* $\mathcal{A}$ *is a set of animats and* $\mathcal{O}$ *is a set of objects, such that all conformations are pairwise disjoint.*

We will now show how these ecosystems develop over time. Algorithm 1 shows the main loop of the ecosystem update algorithm. Details will be given in the following. In computer simulated environments, this algorithm is needed for specifying how the ecosystems develop. In a

Algorithm 1: Main loop of the Ecosystem update algorithm.

```
Input: An ecosystem $(\mathcal{A}, \mathcal{O})$
$t = 0$
while true do
    for $object \in \mathcal{O}$ do
        Update its conformation
        Update its type
    end
    for $animat \in \mathcal{A}$ do
        if animat is alive then
            Leave the genotype of animat unchanged
            For the phenotype of animat:
            Update the sensor values
            Update the homeostatic variables
            Call the Phenotype update algorithm
            Update the conformation of animat
        else
            Add an object of type carcass to $\mathcal{O}$ with
              the same conformation as animat
            Remove animat from $\mathcal{A}$
        end
    end
    Call the Reproduction algorithm
    Add the animats created at $t$ to $\mathcal{A}$
    $t = t + 1$
end
```

physical environment, however, e.g. the environment of a physical robot, no such specification is needed, since all the sensor values etc. are generated automatically.

Note that at each tick, the phenotypes of all animats of the ecosystem receive boolean values to their sensors and real values to their homeostatic variables.

Also note that at each tick, the conformations of animats and objects are updated according to real or simulated laws of physics. For instance, animals and objects may move due to gravity, wind, or currents. Animals may also move due to their own actions. The conformation of animals may also change, e.g. as a result of their own or other animals' actions, including predation.

5 Learning

Algorithm 2 shows the main loop of the phenotype update algorithm, which is common to all animats.

Algorithm 2: Main loop of the Phenotype update algorithm.

```
Input: An old phenotype $A$
New values for its homeostatic variables
New values for its sensors
if alive and age > 0 then
    Update the concept activity
    Update the concept top activity
    Update the experience variables
    Update the graph
    Update the action activity
    Output: The updated phenotype $A$
end
```

In the next few subsections, we will explain the steps of Algorithm 2 that require explanations in detail.

5.1 Experience variable update

age is updated by adding 1 at each tick.

alive is set to $True$ if $age = 0$. *alive* is set to $False$ if $h = 0$ for some homeostatic variable h (death from lack of resources) or if age $> \phi_{MaxAge}$ (death from senescence).

Updating the local Q-value is more complicated. First we need a definition of rewards in terms of homeostatic variables.

Definition 18 (Reward)**.** *For each homeostatic variable h, let*

$$r_h(t) = h(t) - h(t-1).$$

As usual, negative rewards can be understood as punishments. Now we are ready to state how the local Q-values are updated.

Learning rule 1 (Local Q-value updates)**.** *Suppose the concept c is top active at t and the action a_t was performed at t. Then at $t+1$, $Q_h(c, a_t)$ is updated by letting*

$$Q_h(c, a_t) = Q_h(c, a_t) + \Delta,$$

where

$$\Delta = \alpha \cdot \left[r_h(t+1) + \theta \cdot \max_{a'} Q_h^{global}(t+1, a') - Q_h(c, a_t) \right],$$

and

$$Q_h^{global}(t+1, a') = \frac{\sum_{c' \in TA(t+1)} Q_h(c', a') \cdot R_h(c', a')}{\sum_{c' \in TA(t+1)} R_h(c', a')}.$$

Here α and θ are parameters for learning rate and discount rate, respectively, while $TA(t+1)$ are the nodes that are top active at $t+1$.

Intuitively Q_h^{global} is a sum of the local Q-values Q_h that is weighted by R_h. Note that in the special case when there is only one homeostatic variable and all states are unique so that exactly one concept is top active at each t and the reliability values are all set to 1, this update rule coincides with that of standard Q-learning.

Below we sometimes refer to mean and standard deviations of data sets. In all cases, these are computed using the online algorithms for mean and standard deviation described in [Knuth, 1997].

Learning rule 2 (Local R-value updates)**.** *Suppose the concept c is top active and the action a_t was performed at t. Also let SD be the standard deviation of the set of values assigned to $Q_h(c, a_t)$ over time. Then put $R_h(c, a_t) = 1/(SD + 1)$.*

5.2 Graph update

Now we will proceed to learning rules that alter the graph itself. Let us begin with rules that are triggered by prediction errors, or surprises.

Definition 19 (Surprise)**.** *The animat is* surprised *at t if there is a homeostatic variable h such that $Q_h(c, a_t)$ is updated at $t+1$ by at least the quantity SurpriseThreshold, which may be positive or negative, for each concept c that is top active at t. Here SurpriseThreshold is a parameter.*

Now we are ready to define our first structural, i.e. architecture-modifying rule. Intuitively it adds a new sensor set when the outcome of an action was much better or much worse than predicted.

Learning rule 3 (Emotional merge)**.** *Suppose the animat gets surprised at t. Also suppose there are at least two distinct concepts $[S]$ and $[S']$ that are top active at t. Then select two such concepts $[S]$ and $[S']$ and add $S \cup S'$ to the phenotype graph. Also put $Q_h([S \cup S'], a_t) = r_h(t)$, for all homeostatic variables h.*

Note that Emotional merge never applies if only one concept is top active.

Now let us move on to the second structural rule. This rule is also triggered by prediction errors and it adds a new sequence when this happens.

Learning rule 4 (Emotional concatenation)**.** *Suppose the animat gets surprised at t and only one concept c is top active at t. Then randomly select a concept c' that was top active at $t - length(c)$. Add $concat(c, c')$*

to the phenotype graph. Also put $Q_h(concat(c, c'), a_t) = r_h(t)$, for all homeostatic variables h. Here concat is the concatenation operation.

Next we shall introduce a notion of similarity, whose exact meaning is regulated by a parameter. This notion will be used for identifying concepts that are similar enough for one of them to be deleted without causing too much damage to the global performance of the animat.

Definition 20 (Approximation). *Let $v \approx v'$ mean that*

$$\frac{|v - v'|}{|v| + |v'|} \leq \phi_{ApproximationThreshold}$$

if the denominator is nonzero and true otherwise.

For instance, if $\phi_{ApproximationThreshold} = 0.05$, then $0.52 \approx 0.5$, but $0.6 \not\approx 0.5$.

Now we are ready to state the two structural rules that remove concepts: the forgetting rules. These rules are simplified in the sense that forgetting is instantaneous rather than a gradual decay process.

Learning rule 5 (Combination forgetting). *Suppose $Q_h([S \cup S'], a) \approx Q_h([S], a)$, for all actions a and homeostatic variables h. Then delete the concept $[S \cup S']$ from the graph.*

Learning rule 6 (Sequence forgetting). *Suppose $Q_h(concat(c, c'), a) \approx Q_h(c', a)$, for all actions a and homeostatic variables h. Then delete the concept $concat(c, c')$ from the graph.*

6 Decision-making

In this section we will describe the decision-making algorithm (or action activation algorithm) of the animats. The decision-making is performed by a policy that selects exactly one action at every tick.

First, let us introduce a measure of how well the animat is doing at time t:

Definition 21 (Well-being). *The* well-being $w(t)$ *of an animat is defined as follows:*

$$w(t) = \min_h h(t).$$

Thus well-being is a value in $[0, 1]$, where 0 means death and 1 means full satisfaction. This definition will be most adequate if some care is taken when specifying the homeostatic variables. For instance, to model pain, which is something that animals seek to minimize, one may use a homeostatic variable for integrity, where 0 corresponds to minimum integrity (maximum pain) and 1 to maximum integrity (minimum pain). Also, to model body temperature, one may use two homeostatic variables, one for cool and one for warmth. When the body temperature is optimal, both these variables assume the value 1 and when it is warmer or cooler, one of them is strictly smaller than 1. To model critical needs, e.g. needs such as energy, whose dissatisfaction implies death, one may use a homeostatic variable that takes values in the full range $[0, 1]$. To model non-critical needs such as closeness or sexuality, one may use a homeostatic variable that takes values in $[r, 1]$, for some $r > 0$. The fact that well-being is bounded by 1 reflects the idea that needs can be fully satisfied. An animal that is full will not benefit from eating, and an animal that is not thirsty will not benefit from drinking. This model of well-being deviates from that of [Keramati and Gutkin, 2014] by having a criterion for death due to homeostatic unbalance and by taking that criterion into account in the decision-making.

The discounted accumulated well-being that a given action might lead to is estimated as follows:

Definition 22 (Utility). *Let*

$$\text{utility}(t, a) = \min_h \left[h(t) \cdot \delta_h + Q_h^{global}(t, a) \right]$$

Here δ_h is a parameter.

Note that the utility of an action depends on the present homeostatic variables. For instance, the utility of eating or drinking depends on the present levels of hunger and thirst. Next we will define the policy in its most basic form, which is a version of epsilon-greedy.

Definition 23 (Policy). *Let the* policy *be defined as:*

$$\pi(t) = \begin{cases} \text{argmax}_a \text{ utility}(t, a) & \textit{if random}() > \epsilon \\ \text{a random action} & \textit{otherwise} \end{cases}$$

Here ϵ is a parameter and random() is a random generator for the real unit interval.

This is the definition that was promised earlier in Definition 6 of action activity.

7 Reproduction

We want our model for reproduction to cover several types of organisms and both sexual and asexual reproduction, e.g. wind and insect pollination, grass that spreads organically to adjacent locations, sea turtle eggs that are left in the sand, fish eggs that drift with the streams, mammals that give live birth, etc. Here we will present a basic algorithm for sexual reproduction. The case of asexual reproduction is similar but simpler.

7.1 Mechanisms of reproduction

The reproduction mechanisms that will be used here are the standard ones: crossover, as defined in Algorithm 3 and mutation, as defined in Algorithm 4.

To be able to use crossover and mutation algorithms, we need to encode the genotypes as bitstrings. Since sensors and motors are propositional variables, sets of sensors and motors can be naturally represented as bitstrings. For simplicity we assume that all concepts and actions have a maximum length. Then they can easily be encoded as bitstrings as well. Moreover, any set of reflexes can also be encoded as a bitstring in a straightforward fashion. Consequently any graph can be encoded as a bitstring. Furthermore, parameters and experience variables are both computer representations of real numbers. Thus they too can be encoded as bitstrings.

Algorithm 3: Cross-over algorithm

```
Input: Two bitstrings $w$ and $w'$ of the same length
begin
    Let $w_0$ be a random element from the set $\{w, w'\}$
    Let $w_1$ be the other element of $\{w, w'\}$
    Let $n$ be the length of $w_0$ and $w_1$
    Generate a random (crossover) point $k \in \{1, \ldots, n-1\}$
    Let $w''$ be the first $k$ elements of $w_0$ concatenated with the last $n-k$ elements of $w_1$
end
Output: The crossover string $w''$
```

Algorithm 4: Mutation algorithm

```
Input: A bitstring $w$ and a number $MutationRate$
begin
    $w' = [\,]$
    for $bit \in w$ do
        if $random() < MutationRate$ then
            bit' = flip(bit)
        end
        $w' = concat(w', bit')$
    end
end
Output: The mutated bitstring $w'$
```

7.2 Preconditions of reproduction

Let us start by giving a measure of string similarity, which will be used for determining sufficient similarity for sexual reproduction. This measure was selected for simplicity and may well be replaced by some other similarity measure.

Definition 24 (String similarity)**.** *Two bitstrings w and w' of the same length $n > 0$ are similar if*

$$\frac{H(w,w')}{n} < SimilarityThreshold$$

Here $H(w,w')$ is the Hamming distance (i.e. number of differing positions) between w and w', and $SimilarityThreshold$ is a parameter.

Definition 25 (Genetic similarity)**.** *Two animats A and A are genetically similar if the following holds:*

- *their genotype graphs are similar (when encoded as bitstrings in the way outlined above)*
- *their sets of homeostatic variables are identical (although their values might differ).*

The similarity measure varies between 0 (no similarity) and 1 (identity). Like many other similarity measures, it is not transitive.

Now we are ready to state the preconditions for sexual reproduction: Suppose A and A' are animats such that

- The bodies of A and A' have physical contact. This can be made precise in terms of topology: They have physical contact if there is a point p belonging to the conformation of A such that all neighborhoods of p intersect the conformation of A';
- A and A' are genetically similar as defined above;
- One of A and A' is female and the other is male in the sense that the parameter Sex is positive for one of A and A' and non-positive for the other;
- A and A' are both fertile, in the sense that their *age* is inside the interval defined by their phenotype parameters $ReproductiveAgeStart$ and $ReproductiveAgeEnd$;
- Reproduction is successful. By definition this happens if $random() < ReproductionProbability_A$ for A and $random() < ReproductionProbability_{A'}$.

If these criteria are met, then we say that A and A' *reproduce* and refer to them as *parents*.

7.3 Consequences of reproduction

When the criteria for reproduction are met, a number of new animats (offspring) are formed and introduced into the ecosystem. Let us describe how this happens. The number of offspring N is determined by doing crossover followed by mutation on the parameters $NumberOfOffspring$ of the parents. Then a new animat A_k is defined for each $1 \leq k \leq N$. Each A_k is generated in a separate process that involves crossover followed by mutation. To begin with, the genotype of A_k is defined as follows:

1. The genotype graph of A_k is defined by encoding the graphs of the parents as bitstrings, as defined above, doing crossover, then mutation, and finally decoding the resulting string into a graph.
2. The parameters of A_k are obtained from the corresponding parameters of the parents by using crossover followed by mutation.
3. The experience variable *alive* is set to $True$ and *age* is set to $-round(abs(IncubationTime))$, where $IncubationTime$ is a parameter of A_k. Note the minus sign, which signals that A_k is yet to be born. The remaining experience variables are produced using crossover followed by mutation.

Turning to the (initial) phenotype of A_k now, it is defined as follows:

- The phenotype graph of A_k is identical to the genotype graph.
- The activity of phenotype graph of A_k is $False$ for all nodes.
- The phenotype experience variables of A_k are identical to those of the genotype.
- The homeostatic variables of A_k are initialized to the average values of the parents at the moment of reproduction.

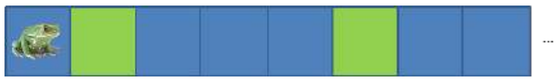

Figure 3: The frog in the forest. This ecosystem consists of a frog on an infinite path of cells that are green or blue (shallow water). The frog jumps forward at each tick automatically. It has two homeostatic variables (energy and water) that go up when it eats on green and drinks on blue cells, respectively. (Real frogs do not drink, but this animat frog happens to do so.) At each tick, small quantities of energy and water are consumed due to metabolism. This means that the animat eventually dies if it makes suboptimal actions too often.

$$\begin{tikzcd}
eat & & drink \\
\{eat\} \arrow[u] & \{\} & \{drink\} \arrow[u] \\
{[\{eat\}]} \arrow[u] & {[\{\}]} \arrow[u] & {[\{drink\}]} \arrow[u] \\
{[\{green\}]} & & {[\{blue\}]} \\
\{green\} \arrow[u] & & \{blue\} \arrow[u] \\
green \arrow[u] & & blue \arrow[u]
\end{tikzcd}$$

Figure 4: The phenotype graph of the frog. The action [{}] is the idle action (passivity), which activates no motors.

The conformation of A_k is located at the place of conception and its geometry must be specified by the ecosystem dynamics. This concludes the definition of the new animat A_k. As soon as A_k has been conceived, it is inserted into the ecosystem and Algorithm 2 starts running with A_k as input.

Before moving on, let us note that the reproduction model just described enables new "species" to be formed. In fact, reproduction involves mutation, which has an element of randomness. Mutation may lead from a population of animats that are genetically similar to a population where this is no longer the case. A similar genetic drift can happen also as a result of crossover alone.

8 Examples of ecosystems

In this section we will look at five concrete examples of ecosystems. The purpose is to show how the animat model works and at the same time illustrate its generality. Code of the animat model and several concrete ecosystems can be found at `www.github.com/animatai`.

8.1 The frog in the forest

Consider the frog world shown in Figure 3. This world is populated by a frog animat with the phenotype graph shown in Figure 4. This frog will quickly learn to eat on green and drink on blue cells. In fact, after a few ticks,

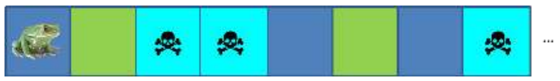

Figure 5: The frog in the swamp. This ecosystem consists of a frog on an infinite path of cells that are green (grass), turquoise (swamp), or blue (water). The frog jumps forward at each tick automatically. It has two homeostatic variables (energy and water) that go up when it eats on green and drinks on blue cells, respectively. If it eats or drinks on a turquoise cell (which is both blue and green), it will throw up and thus lose energy and water Again, a small quantity of energy and water is always consumed due to metabolism.

$$Q_{energy}([\{green\}], [\{eat\}]) > Q_{energy}([\{green\}], [\{drink\}])$$

and

$$Q_{water}([\{blue\}], [\{drink\}]) > Q_{water}([\{blue\}], [\{eat\}]).$$

This leads to the desired behavior (almost exactly as in Q-learning). In this case the phenotype graph will remain the same throughout the lifetime of the animat. In fact, it will never get surprised and no rules for adding or removing nodes will ever be triggered.

8.2 The frog in the swamp

Now consider a slightly different frog world, shown in Figure 5. This world is also populated by a frog animat, whose phenotype graph looks exactly as in the previous example (Figure 4). This time the graph will change, however. In fact, when eating or drinking on a turquoise square for the first time the frog will vomit and hence get punished. This punishment means that the animat gets surprised (e.g. it had previously learned that it got rewarded for drinking on blue cells, but now it got punished for doing this instead).

This surprise triggers the rule *Emotional merge*, which in this case leads to the formation of the new concept $[\{green, blue\}]$, representing turquoise. The resulting graph is shown in Figure 6. The animat will remember the bad experience now, since:

$$Q_{water}([\{green, blue\}], [\{drink\}]) < 0.$$

8.3 The frog in the field

Now let us consider a third frog world, this time it looks like in shown in Figure 7. This world is again populated by a frog animat, whose initial phenotype graph looks as in Figure 4. The animat will get surprised when it drinks poisonous water for the first time.

This surprise in turn triggers the rule *Emotional concatenation*. In fact, *Emotional merge* does not apply. This leads to the formation of the new concept $[\{green\}, \{blue\}]$, representing green followed by blue. The resulting graph is shown in Figure 8. Again the animat will remember the bad experience, since:

$$Q_{water}([\{green\}, \{blue\}], [\{drink\}]) < 0.$$

$$\begin{tikzcd}
\mathit{eat} & & \mathit{drink} \\
\{\mathit{eat}\} \arrow[u] & \{\} & \{\mathit{drink}\} \arrow[u] \\
{[\{\mathit{eat}\}]} \arrow[u] & {[\{\}]} \arrow[u] & {[\{\mathit{drink}\}]} \arrow[u] \\
{[\{\mathit{green}\}]} & {[\{\mathit{green}, \mathit{blue}\}]} & {[\{\mathit{blue}\}]} \\
\{\mathit{green}\} \arrow[u] & \{\mathit{green}, \mathit{blue}\} \arrow[u] & \{\mathit{blue}\} \arrow[u] \\
\mathit{green} \arrow[u] \arrow[ur] & & \mathit{blue} \arrow[u] \arrow[ul]
\end{tikzcd}$$

Figure 6: The phenotype graph of the frog after the automatic node addition.

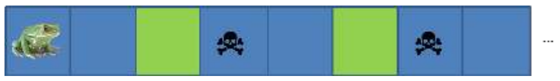

Figure 7: The frog in the field. This example is similar to the other ones, but the twist this time is that the water to the right of green cells is poisonous. For instance, the wind might have been blowing from left to right and mixed soil into the water pools, thus making some of them poisonous.

$$\begin{tikzcd}
\mathit{eat} & & \mathit{drink} \\
\{\mathit{eat}\} \arrow[u] & \{\} & \{\mathit{drink}\} \arrow[u] \\
{[\{\mathit{eat}\}]} \arrow[u] & {[\{\}]} \arrow[u] & {[\{\mathit{drink}\}]} \arrow[u] \\
{[\{\mathit{green}\}]} & {[\{\mathit{green}\}, \{\mathit{blue}\}]} & {[\{\mathit{blue}\}]} \\
\{\mathit{green}\} \arrow[u] \arrow[ur] & & \{\mathit{blue}\} \arrow[u] \arrow[ul] \\
\mathit{green} \arrow[u] & & \mathit{blue} \arrow[u]
\end{tikzcd}$$

Figure 8: The phenotype graph of the moor frog after the automatic node addition.

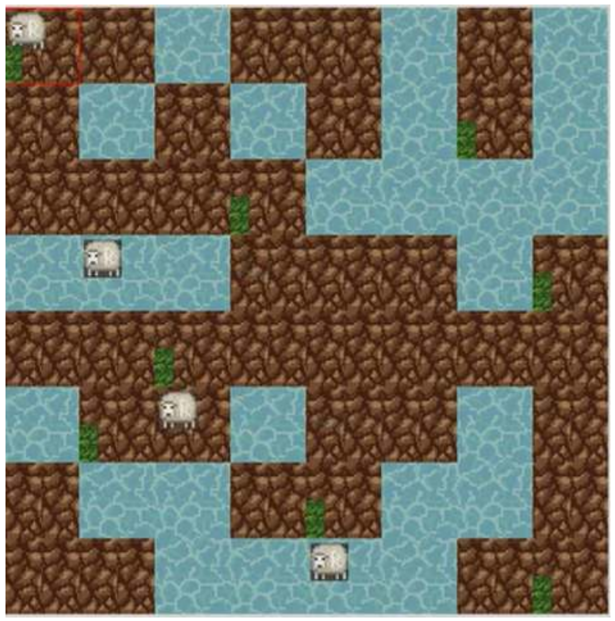

Figure 9: The sheep world at time 0. This world consists of green cells (grass), blue cells (water), and brown cells (sand). The world is populated with sheep, two males and two females. The sheep can move up, down, left, and right. They can also idle, eat, and drink. Moreover, they have sensors for green, blue and brown. Again their homeostatic variables are energy and water. Eating and drinking on green and blue cells give, respectively, energy and water reward. Eating or drinking on other cells yields a slight punishment in terms of both energy and water. The grass grows by propagation to adjacent cells and gets consumed when the sheep graze there.

8.4 The sheep world

Now let us consider a two-dimensional ecosystem with sheep. Figures 9, 10, and 11 show this ecosystem at time 0, 97, and 123, respectively.

8.5 The sheep and wolves world

The sheep and wolves world at time 0 is shown in Figure 12. Figure 13 shows scaled biomass (total weight) curves for this ecosystem, indicating how the populations of sheep and wolves develop over time. Figures 14 and 15 show the learning curves for one of the sheep and one of the wolves, respectively.

9 Conclusion

A computational model was presented for fully automatic genotype and phenotype development in artificial ecosystems. The model combines generic mechanisms for decision-making, learning, reproduction, and death. The model uses graphs, whose architectures develop over time as a result of learning as well as evolution. This feature sets it apart from other approaches that combine evolution and learning, e.g. [Ackley and Littman, 1992] and [Yaeger, 1994].

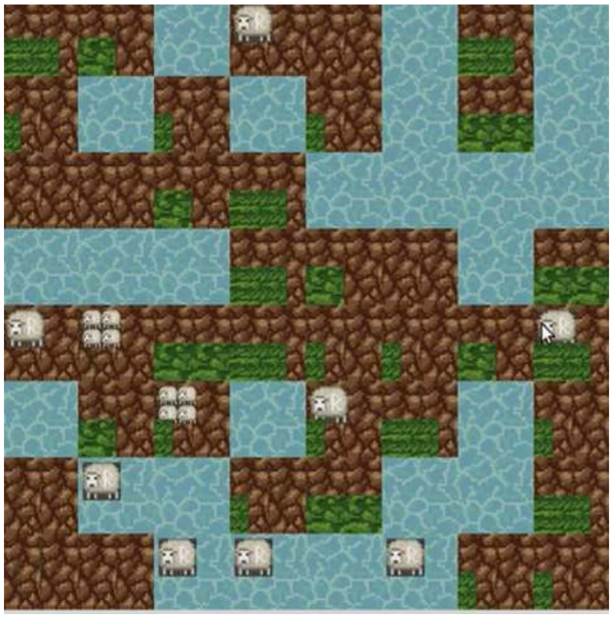

Figure 10: The sheep world at time 97. Here we see that the grass has grown and the sheep have reproduced. The new symbol here represents a pregnant female with offspring inside her body.

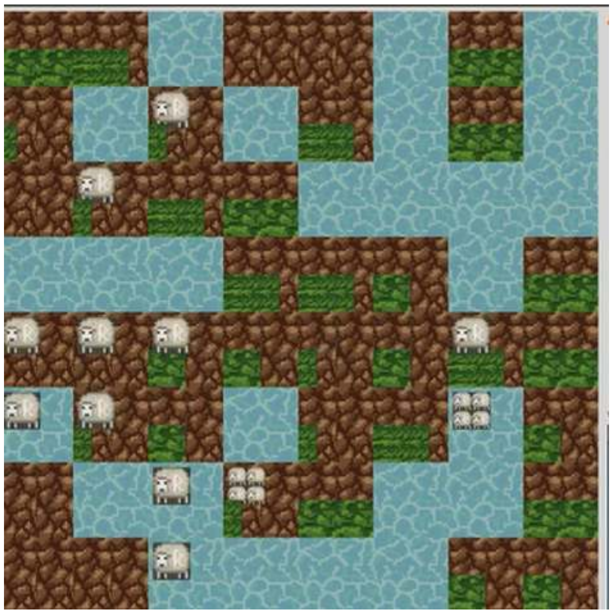

Figure 11: The sheep world at time 123. As the sheep multiply, the impact of competition gets more pronounced. For instance, the two sheep to the left might have difficulty finding grass. At this point several sheep have already died from lack of resources and others have died out of senescence. Several generations have also been born.

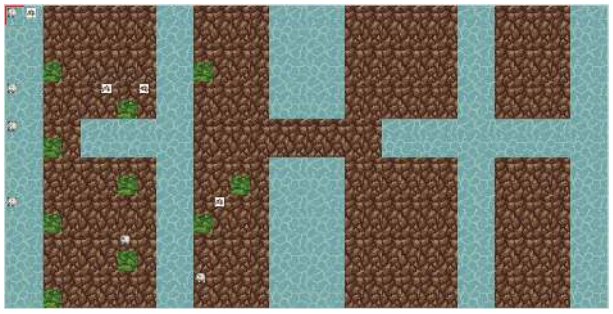

Figure 12: The sheep and wolves world at time 0. This world is initialized with 6 sheep, 4 wolves and 10 patches of grass. Both sheep and wolves have energy and water as their homeostatic variables. Both get their water from drinking at the blue cells. The sheep get their energy from grazing, whereas the wolves get theirs from predation, while eating on a cell that is shared with a sheep.

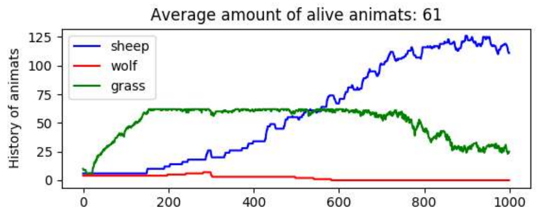

Figure 13: Biomass curve. These three curves show the development over time of the biomass for sheep, wolves, and grass, respectively.

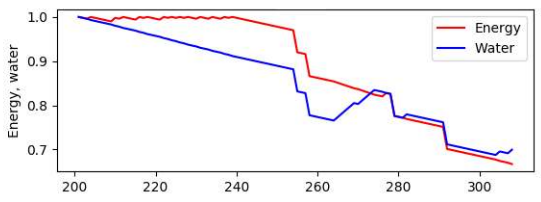

Figure 14: Learning curve for one of the sheep. This sheep was born at tick 200 with maximum levels of energy and water. After several ticks the sheep tried to eat grass and learned to eat. After tick 240 the sheep started to explore the world. We observe sharply declining curves between tick 252 and tick 258. The decline was caused by either inappropriate actions, e.g. eating or drinking in a sand cell, or being bitten by a wolf. Both would lead to significant decreases. After tick 260 the sheep drank water several times and quickly learned to drink to increase its level of water. Still, it has difficulty finding resources after about 282 steps.

Our computational model is still in early phase and much work remains to be done when it comes to fine-tuning, scalability testing, and optimization.

Several examples of developing ecosystem simulations were given in order to illustrate how the model works and give an indication of its generality.

From the perspective of ecosystem modeling, the simulations suggest that the model can handle a wide range of simple ecosystems. Our model has not been tried on

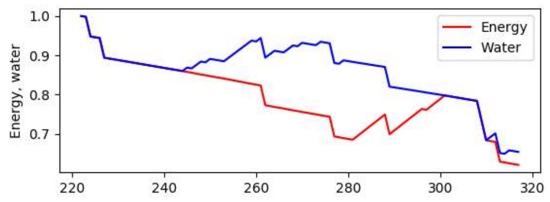

Figure 15: Learning curve for one of the wolves. This wolf was born at tick 220. After 20 steps it tried to drink water and learned it in a short time. At tick 260 and 276 the wolf took inappropriate actions in wrong cells which lead to significant loss of energy and water. At tick 280 the wolf tried to eat for the first time and learned to eat in few ticks. The wolf started to explore again after tick 300 when the curve began to decline.

more complex ecosystems, however. The combination of learning and evolution seems to be indispensable for realistic ecosystem simulations. In the future we expect prominent ecosystem models to include both these components. From the perspective of AI, our simulations suggest that dynamic architectures have a clear advantage over the static architectures that are, e.g. used in deep learning. As far as we understand, dynamic architectures could well be the next big step forward in AI.

Acknowledgement

This research was supported by the Torsten Söderberg Foundation Ö110/17.

References

[Ackley and Littman, 1992] David Ackley and Michae Littman. Interactions between learning and evolution. In C. G. Langton, C. Taylor, C. D. Farmer, and Rasmussen S., editors, *Artificial Life II, SFI Studies in the Sciences of Complexity*, pages 487–509. Addison-Wesley, Reading, MA, USA, 1992.

[Ashton *et al.*, 2018] Benjamin J. Ashton, Amanda R. Ridley, Emily K. Edwards, and Alex Thornton. Cognitive performance is linked to group size and affects fitness in australian magpies. *Nature online*, (2018/02/07), 2018.

[Avila-García and Cañamero, 2005] Orlando Avila-García and Lola Cañamero. Hormonal modulation of perception in motivation-based action selection architectures. In *Procs of the Symposium on Agents that Want and Like*. SSAISB, 2005.

[Bach, 2015] Joscha Bach. Modeling motivation in MicroPsi 2. In *AGI 2015 Conference Proceedings*, pages 3–13. Springer, 2015.

[Bouneffouf *et al.*, 2017] Djallel Bouneffouf, Irina Rish, and Guillermo A Cecchi. Bandit models of human behavior: Reward processing in mental disorders. In *AGI 2017 Conference Proceedings*, pages 237–248. Springer, 2017.

[Buro, 1998] Michael Buro. From simple features to sophisticated evaluation functions. In *International Conference on Computers and Games*, pages 126–145. Springer, 1998.

[Christensen and Walters, 2004] Villy Christensen and Carl J Walters. Ecopath with ecosim: methods, capabilities and limitations. *Ecological modelling*, 172(2-4):109–139, 2004.

[Dietzsch, 2008] Markus Dietzsch. Agentenentwicklung mit dem micropsi-framework. Master's thesis, Humboldt-Universität zu Berlin, 2008.

[Draganski and May, 2008] Bogdam Draganski and Arne May. Training-induced structural changes in the adult human brain. *Behavioural brain research*, 192(1):137–142, 2008.

[Ernst *et al.*, 2005] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(Apr):503–556, 2005.

[Fahlman and Lebiere, 1990] Scott E Fahlman and Christian Lebiere. The cascade-correlation learning architecture. In *Advances in neural information processing systems*, pages 524–532, 1990.

[Futuyma, 2009] D.J. Futuyma. *Evolution*. Sinauer Associates, 2009.

[Gabbay *et al.*, 1994] Dov M Gabbay, Ian Hodkinson, and Mark Reynolds. *Temporal logic (vol. 1): mathematical foundations and computational aspects*. Oxford University Press, Inc., 1994.

[Goodfellow *et al.*, 2016] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Hubbell, 2001] Stephen P Hubbell. The unified neutral theory of biodiversity and biogeography. 2001.

[Keramati and Gutkin, 2011] Mehdi Keramati and Boris S Gutkin. A reinforcement learning theory for homeostatic regulation. In *Advances in neural information processing systems*, pages 82–90, 2011.

[Keramati and Gutkin, 2014] Mehdi Keramati and Boris Gutkin. Homeostatic reinforcement learning for integrating reward collection and physiological stability. *Elife*, 3, 2014.

[Knuth, 1997] Donald Ervin Knuth. *The art of computer programming*, volume 2. Pearson Education, 1997.

[Koza, 1994] John R Koza. Genetic programming as a means for programming computers by natural selection. *Statistics and computing*, 4(2):87–112, 1994.

[Langton, 1997] Christopher G Langton. *Artificial life: An overview*. MIT Press, 1997.

[LeCun *et al.*, 2015] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[Lotka, 1925] Alfred James Lotka. *Elements of Physical Biology, by Alfred J. Lotka.* 1925.

[Niv, 2009] Yael Niv. Reinforcement learning in the brain. *Journal of Mathematical Psychology*, 53(3):139–154, 2009.

[Roijers *et al.*, 2013] Diederik M Roijers, Peter Vamplew, Shimon Whiteson, Richard Dazeley, et al. A survey of multi-objective sequential decision-making. *J. Artif. Intell. Res.(JAIR)*, 48:67–113, 2013.

[Russell and Zimdars, 2003] Stuart J Russell and Andrew Zimdars. Q-decomposition for reinforcement learning agents. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 656–663, 2003.

[Rusu *et al.*, 2016] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[Strannegård *et al.*, 2017] Claes Strannegård, Nils Svangård, David Lindström, Joscha Bach, and Bas Steunebrink. The animat path to artificial general intelligence. In *Workshop on Architectures for Generality and Autonomy, IJCAI-17*, 2017.

[Such *et al.*, 2017] Felipe Petroski Such, Vashisht Madhavan, Edoardo Conti, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. *arXiv preprint arXiv:1712.06567*, 2017.

[Sutton and Barto, 1998] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 1998.

[Tuci *et al.*, 2016] Elio Tuci, Alexandros Giagkos, Myra Wilson, and John Hallam, editors. *From Animals to Animats. 1st International Conference on the Simulation of Adaptive Behavior*. Springer, 2016.

[Watkins, 1989] Christopher John Cornish Hellaby Watkins. *Learning from delayed rewards.* PhD thesis, King's College, Cambridge, 1989.

[Wilson, 1986] Stewart W Wilson. Knowledge growth in an artificial animal. In *Adaptive and Learning Systems*, pages 255–264. Springer, 1986.

[Wilson, 1991] Stewart W Wilson. The animat path to AI. In J. A. Meyer and S. W. Wilson, editors, *From animals to animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, 1991.

[Wolfe *et al.*, 2017] Nikolas Wolfe, Aditya Sharma, Lukas Drude, and Bhiksha Raj. The incredible shrinking neural network: New perspectives on learning representations through the lens of pruning. *arXiv preprint arXiv:1701.04465*, 2017.

[Yaeger, 1994] L.S. Yaeger. Computational genetics, physiology, metabolism, neural systems, learning, vision, and behavior or polyworld: Life in a new context. In *Proceedings of the Artificial Life III conference*, pages 263–298. Addison-Wesley, 1994.

[Yoshida, 2017] Naoto Yoshida. Homeostatic agent for general environment. *Journal of Artificial General Intelligence*, 8(1), 2017.

[Zaremba and Sutskever, 2015] Wojciech Zaremba and Ilya Sutskever. Reinforcement learning neural turing machines-revised. *arXiv preprint arXiv:1505.00521*, 2015.