DOI: http://dx.doi.org/10.7551/978-0-262-31709-2-ch080

Open-Ended Evolution of a Circadian Rhythm

Tiago Baptista and Ernesto Costa

CISUC, University of Coimbra, Coimbra, Portugal
baptista@dei.uc.pt

Abstract

Most biological systems have some sort of adaptation to our planet's cycle of day and night. This adaptation is a current subject of scientific research, and serves as inspiration to develop a multi-agent simulation to investigate the evolution of complexity in an open-ended evolutionary framework. In a previous work, we created a simulated world where artificial organisms evolve to synchronize with a daily cycle of light and darkness. A multi-agent, artificial life framework was used to implement these simulations. In this paper, we further develop that world, by adding caves to the environment. When in these caves, the agents will perceive a low level of light, as if it were night. This adds an extra layer of complexity to the desired behavior of the agents, as now they need to distinguish "night" from "cave". Using the same agent structure, and the same open-ended evolution framework, we show that the agents evolve to adapt to this new environment. We also show how the agents adapt to the environment with caves, by analyzing their brains.

Introduction

The study of circadian clocks and similar synchronization phenomena in biological systems is a current subject of scientific research (Rand et al., 2006; Strogatz, 2004). Despite having been extensively studied, these phenomena still have much to be investigated. Our goal, however, is not to learn more about this biological process, but to use it as an inspiration to study the emergence of complex behaviors in an open-ended evolution scenario. To that end, we implement a simulation where the environment has a day and night cycle, and analyze the evolution of the agents' behavior, and their adaptation to this cycle.

In a previous paper, the authors presented some experiments done with such a scenario (Baptista and Costa, 2008), and showed that the agents do develop behaviors adapted to the daily cycle. In an effort to create a more challenging environment, we now further developed that world by adding caves. When an agent enters a cave, the light level will be the same as if it were night time. This will force the agents to evolve behaviors capable of distinguishing the two different low light conditions, adding extra complexity to the requirements for survival.

Some previous work has been done with similar simulation scenarios, either by evolving neural networks (Mirolli and Parisi, 2003), or virtual CPU organisms in AVida (Beckmann et al., 2007). Our scenario can be mostly compared to that of (Mirolli and Parisi, 2003), as they also have an environment with varying light level and caves. However, they use a standard genetic algorithm to evolve the agents, whereas we use an open-ended evolution framework.

Although an established definition of open-ended evolution hasn't yet surfaced, most authors consider that one of the major requirements is the absence of an explicit fitness function. In other words, to have open-ended evolution, a system should be based on Natural Selection rather than Artificial Selection (Channon, 2000).

The simulations described in this paper were implemented using the BitBang framework. One of the purposes of these simulations is to serve as a proof of concept for the model developed for the framework. Implementing a modern autonomous agent model (Russell and Norvig, 2002), this framework has roots in Artificial Life systems and Complexity Science. The simulated world is composed of entities. These can either be inanimate objects which we designate as *things*, or entities that have reasoning capabilities and power to perceive and affect the world—the *agents*. Both have traits that characterize them, such as color, size, or energy—the *features*. The agents communicate with, and change the environment using *perceptions* and *actions*, taking decisions using the *brain*. In this model, there is no definition of a simulation step, as we won't have any type of centralized control. As such, the simulation is asynchronous. The agents will independently perceive, decide, and act. Moreover, there is no evolutionary mechanism included in the definition of the model, since evolution is implemented as an action. That is accomplished by giving the agents the capability of reproduction. Again, there is no central control bound to the process of reproduction. The agents choose when to reproduce and with what other agent to reproduce with. In addition, there is no explicit fitness function. The agents die due to lack of resources, predators, age, or any other mechanism implemented in the world. Thus, in this

model we have open ended evolution. To have a more in-depth view of the conceptual model and architecture of BitBang, refer to (Baptista et al., 2006).

In the next section we will describe the simulation world developed, detailing the agents, things, brain architecture, evolutionary process, and environmental settings. We will then present the experimental results and end with some conclusions.

The DayNight World

In this section we will set out all the implementation details and architecture of the simulations. As mentioned above, these simulations were implemented using the BitBang framework, and therefore we will present the architecture according to the framework's specifications. We begin by describing the simulation environment, then detail the agents' architecture (features, perceptions, actions, and brain). Next, we will present the two types of things defined in this world, and finally we present the architecture of brain used in these experiments.

The Environment

Our world is a 3D world where agents and resources are placed (see figure 1). The terrain is a square. This area restricts the placement of agents, caves, and resources, but does not restrict the movement of the agents. The world is infinite, i.e., an agent can move past the boundaries of the populated terrain. At startup, the field is populated with a configured amount of randomly placed food items. These are periodically replenished so that the total food count is maintained. The number of resources available is configurable to be able to fine tune the system so as to allow agents to survive but also provide enough evolutionary pressure.

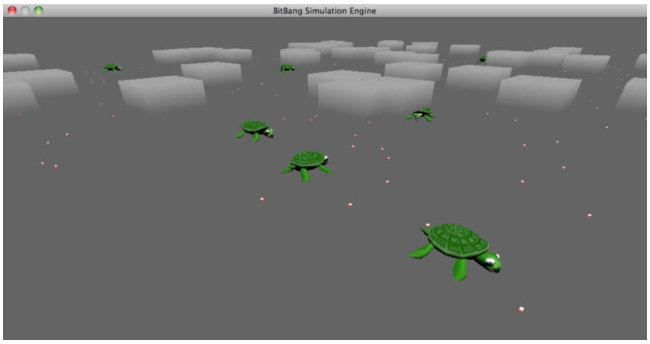

Figure 1: Screenshot of a running simulation. We can see the agents (turtles), the edible resources (small red cubes), and the caves (large grey cubes).

On initialization, the world is populated with randomly placed, and randomly generated agents. At this time, it is highly probable that the agents will not execute the reproduction action, either by not choosing it, or because they don't have enough energy to reproduce. To keep the population alive, whenever the number of agents in the world falls bellow a given threshold, new agents are created. If there are live agents in the environment, one will be picked for reproduction, otherwise a new random agent is created. Note that, as stated before, there is no explicit fitness function, so the agent chosen for reproduction will be randomly selected from the population.

To differentiate the day from the night, the environment has a light level that oscillates between a configurable maximum and minimum. For each day the maximum light level is randomly calculated as the overall maximum minus a random value between zero and the delta. The same applies for the day's minimum. For example, if the maximum light level is 100, the minimum light level is 0, and the delta is 10, each day's maximum light level will be a random value between 90 and 100, and each day's minimum light level will be a random value between 0 and 10. Additionally, the light level does not rise or fall abruptly, but rather changes linearly during a specified time interval, simulating dusk and dawn. To better illustrate, in figure 2 this variation of the light level can be observed. The duration of one day, can be configured, and remains constant for the duration of the experiment.

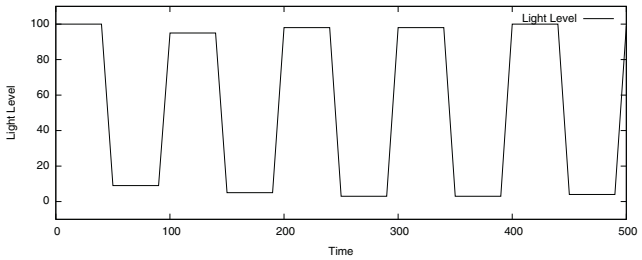

Figure 2: Example of the variation of the light level over the course of five days. In this example, the maximum light level is 100, the minimum is 0, and the delta is 10.

The total simulation time for these experiments was divided into two equal parts. For the first part of the simulation, the agents live in an environment that only has food items. Then, we randomly place in the environment a number of caves, and continue the simulation in this new environment. When an agent enters a cave, the light level it perceives will be the same as if it were night, creating a new challenge for our agents. By the time the caves are created, the agents will have evolved a behavior adapted to the light level, sleeping when the light is low, and being active when its high. However, when an agent enters a cave, it will not be able to distinguish if the low light level means "night" or "cave". If it simply goes to sleep whenever the light level is low, it will never wake up again, as the light level in the cave will never rise. Our agents will now have to adapt their behavior to this new environment.

The Agents

In this simulation only one type (species) of agent exists, and has the following architecture:

- **Features:** energy, metabolic rate, and birth date.
- **Perceptions:** energy, resource location, reach resource, light level.
- **Actions:** move front, turn left, turn right, sleep, eat, reproduce.
- **Brain:** rule list (see section).

We will now describe each one of these components.

Features

Energy This feature represents the current energy level of the agent. When this feature reaches zero, the agent dies. The feature is initialized with a predetermined value at agent birth. For these simulations, the agents are initialized with 10 energy units.

Metabolic Rate The metabolic rate is the amount of energy the agent consumes per time unit. This rate is initialized to its configured base value, and changes as the agent moves or sleeps. The increase or decrease amounts for move and sleep are configurable.

Birth Date This feature is set to the current time at birth and remains constant. It is used to calculate the agent's age. When the agent reaches a given age, it dies. This procedure allows the evolution to continue past the moment when the agents have developed good navigation and eating capabilities, whilst maintaining an asynchronous and open-ended simulation. The maximum age of the agents is configurable.

Perceptions

Energy This is a self-referencing perception on the agent's current energy level. This perception is tied to the corresponding feature. This is a numerical perception, and the range of values can be configured.

Resource Location This is the agent's main perception of vision, representing the position of the nearest resource, relative to the agent's position and orientation. The agent's vision is implemented as a 3D cone in front of the agent. The vision cone is configured with a given range and angle, representing its height and aperture. This is a numerical perception with possible values 0, 1, 2, and 3. The value 0 means no resource is visible. The value 1 means there is a resource to the left. The value 2 means there is a resource directly in front of the agent. The value 3 means there is a resource to the right. This perception is influenced by the light level of the environment. As the light level drops, so does the range of vision for the agent, using the following equation:

$$V(t) = V_0 \frac{L(t)}{L_0}, \tag{1}$$

where $V(t)$ is the vision range at time t, V_0 is the configured vision range of the agents, $L(t)$ is the light level at time t, and L_0 is the configured maximum light level.

Reach Resource This is a boolean perception that evaluates to true whenever the agent has a resource within its reach. The distance the agent can reach is configurable.

Light Level This perception gives the agent the power of sensing the brightness of the environment. This can also be considered a perception of vision. The value of the perception is numeric and, at each time, is evaluated to the environment's current light level if the agent is outside. When in a cave, the perception evaluates to the minimum light level whether it is night or day.

Actions

Movement We define three actions for movement. One to walk forward, one to turn left, and one to turn right. These actions have a tie to the metabolic rate feature in such a way that whenever the agent is moving, the metabolic rate increases.

Eat This action enables the agent to eat a resource within its range. If no resource is in range when the action is executed, nothing happens. This action will add a configured amount of energy to the agent's energy feature.

Sleep The agent can use this action to sleep. In this simulation, when an agent is sleeping, it will stand still and its metabolic rate will decrease, falling below the base metabolic rate and thus allowing the agent to conserve energy. As for the rest of the actions, it gets executed whenever the agent chooses to do so.

Reproduce This action allows the agent to reproduce itself. The reproduction implemented is asexual. When the action is executed, a new agent is created and placed in the world. The new agent will be given a brain that is a mutated version of its parent's brain. Note that, as each mutation operator has a given probability of being applied, the child's brain can be a perfect clone of its parent's brain. The action will also transfer energy from the parent to the offspring. The amount of energy consumed in the action is the sum of the initial energy for the new agent and a configurable fixed cost. It's important to have a cost of reproduction higher than

the initial energy of an agent, so as to provide evolutionary pressure.

Brain The agents' brain used in these experiments is a rule list. The architecture of this system is explained in section . On initial creation of an agent, the brain is randomly initialized. This initialization conforms to some configurable parameters: the maximum number of rules, the minimum number of rules, and the maximum number of conditions per rule. Other configured values are the mutation probabilities used in the reproduction action.

The Things

Two types of things have been defined for this world: the resources that the agents eat to acquire energy, and the caves. No features are associated with them. A configurable parameter defines the amount of energy each resource provides. Although the caves are also represented as things, they are not visible to the agents, or else, it would be easier for them to distinguish "night" and "cave".

The RuleList Brain

The Rule List brain is composed of an ordered list of rules. The reasoning process is straightforward. The rules are evaluated in order, and the first one whose conditions are all true, is selected. Each rule is composed of a conjunction of conditions and an action. The structure of a rule is shown in listing 1. Next, to illustrate, in listing 2 we provide an example of a rule.

Listing 1 Syntax of a rule in the RuleList brain.

```
<rule> ::= IF <cond-list> THEN <action>
<cond-list> ::= <condition>
<cond-list> ::= <condition> AND
<cond-list>
<condition> ::= <percept> <operator>
<percept>
<condition> ::= <percept> <operator>
```

Listing 2 Example of a rule.

```
IF energy < 10 AND reach_resource TRUE
THEN eat
```

The use of this brain architecture has the added benefit of readability. It is easy to understand the reasoning process by looking at the agent's rule list. This feature will permit a better analysis of the results.

To be able to evolve this brain architecture we need to define its equivalent to the genome, and the operators that modify it on reproduction. The brain's genome is the rule list itself, no translation is applied. To alter it we defined only mutation operators. These operators are show in table 1.

Table 1: Mutation operators of the RuleList brain.

Operator	Description
Mutate List	This operator iterates through the rule list, and replaces a rule with a new random one.
Mutate Rules	This is the lowest level operator. It drills down to the perceptions on the conditions and mutates both the perceptions and their operators. It also mutates the action of the rules.
Mutate Order	This operator iterates through the rule list and moves a rule to the top of the list.
Mutate Order 2	This operator iterates through the rule list and moves a rule one position towards the top.

Not all of the mutation operators must be used. The programmer decides which of them to use for a particular experiment. In the case of the experiment described in this paper, the operators used were the *Mutate Order, Mutate Rules,* and *Mutate List.*

Results

In this section we will expose and analyze the results of the experiments. But first we give an overview of the main configuration values used for the simulations. Most of the configuration values presented are the result of previous experimentation done on (Baptista and Costa, 2008).

The terrain is a square with sides of 1000 units. This field is populated initially with 20 agents, and that is also the minimum number of agents. These initial agents are generated with a brain composed of a random set of rules. The agents' brains are initialized with between 15 and 20 rules, having each up to 2 conditions. The initial and minimum number of food items is 200. One day lasts for 100 time units and the transitions from day to night, and vice versa, last 10 time units. The light level has a maximum of 100 and a minimum of 0, with a delta of 10. That means that for a given day the actual maximum light level will be between 100 and 90, and the minimum level between 0 and 10. Each agent is created with a vision range of 200 units and a vision angle of 60°. The agents reach is 20 units. Agents are initialized with energy 10 and consume a base metabolic rate of 0.1 energy units for each time unit. The metabolic rate increases by 0.01 when the agent is moving and drops by 0.03 when sleeping. The maximum age of the agents is 500 time units. The cost of reproduction is 2 energy units, plus the energy initialization for the child agent. Each food item gives an agent 3 energy units. The mutation probabilities are 0.01 for every operator. These experiments were run with a time limit of 200,000 time units. The caves were placed in the

environment at time 100,000.

Regarding the configuration parameters for the caves, as they represent the main addition to these experiments, we ran simulations with different configuration values. All caves have a side of 50. The field is populated with either 50, 20, or 10 caves.

For the experiments presented here, we ran 30 independent simulations for each configuration of the parameters. As is the case in (Baptista and Costa, 2008), from those 30 simulations, there are some where the agents will not be successful in evolving good foraging behavior, and thus will not be able to further evolve reproduction or synchronization to the day cycle. These experiments were not taken into account for some of the results shown in this section. Whenever that is the case, it will be stated.

Next, we will present and analyze the data of typical runs from the simulations. From all the simulations and runs analyzed, we found mainly two different types of plot (see figure 3 and figure 4). These represent the majority of results from the runs (except for those that are unsuccessful). Both the figures are taken from runs with the same parameter configuration (50 caves).

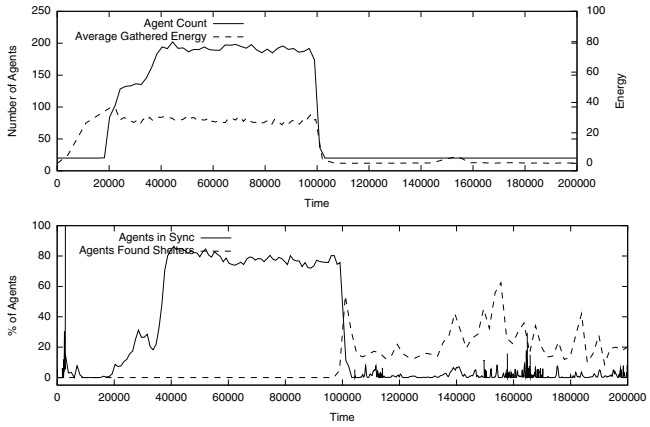

Figure 3: Plot of the evolution of the total number of agents in the population, average gathered energy, percentage of agents in sync with the day cycle, and percentage of agents that found caves, over the course of one simulation run (number of caves is 50).

The first type of plot, shown in figure 3, is what we expected to find in this experiment. Here, the population collapses and loses synchronization when the caves are inserted into the environment. Examining this plot, we can clearly see the agents are successful in evolving food gathering, reproduction, and synchronization in the first environment (up to time 100,000). Then, when the environment changes, most of the population dies and never recovers food gathering capabilities.

In fact, this data is consistent with what is presented in (Mirolli and Parisi, 2003). In that paper, the authors show that, when the agents only have an input of the light level, they are not able to differentiate the "caves" from the "night". They provide a possible solution to the problem, by incorporating in the structure of the brain, a clock source. However, in our simulations, we found that on a significant number of runs (see table 2) the agents do recover.

In figure 4 we show an example of a typical run where the agents recover in the second environment. In the first environment, we find a plot similar to that of figure 3. The agents develop good food gathering capabilities, reproduce, and synchronize with the daily cycle. When the environment changes at 100,000 time units, both the size of the population and the average gathered energy drop, but then quickly recover. More importantly, by analyzing the percentage of synchronized agents, we can see that, although it takes longer to recover, the agents also resynchronize to the daily cycle. One might wonder if it would be the case that the agents are simply not entering caves. But the plot also shows that about 80% of the agents find at least one cave during their lifetime. Note that, as these percentages are taken from the whole population at a given time interval, and there are constantly new agents being born, the percentage could never rise to 100%. New agents will normally need some time to move before they find a cave.

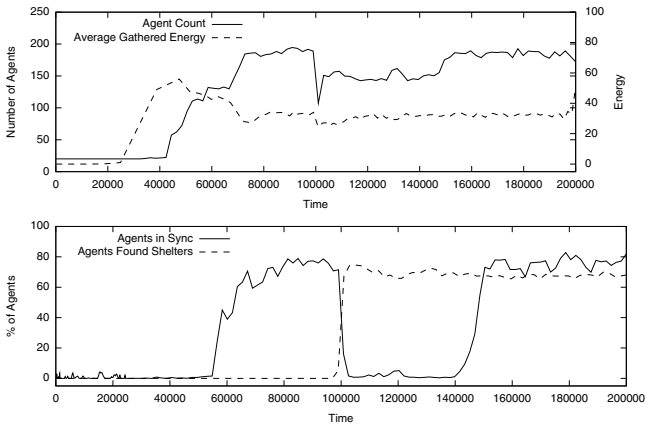

Figure 4: Plot of the evolution of the total number of agents in the population, average gathered energy, percentage of agents in sync with the day cycle, and percentage of agents that found caves, over the course of one simulation run (number of caves is 50).

These results may seem counter-intuitive, as there doesn't seem to be any way for the agents to detect the caves. To clarify, we analyzed some agents' brains. In listing 3 and listing 4 we show two examples of agents' brains from the simulation run shown in figure 4. The first one is taken from an agent living in the environment without caves (time 84536), and the second is taken from the environment with caves and at a time where the agents have resynchronized (time 183513).

Listing 3 Example of the structure of the brain of an agent born at time 84536, for the simulation run shown in figure 4. Used Rules are set in bold.

1. **IF Resource Location = 3 THEN turn right**
2. IF Light Level $<$ Light Level THEN eat
3. IF Resource Location $>$ 3 THEN eat
4. **IF Light Level $<$ 26.5859 THEN sleep**
5. IF Light Level $<$ 36.3748 THEN sleep
6. IF Light Level = 47.8427 THEN eat
7. **IF Feature energy $>$ 16.7159 THEN reproduce**
8. IF Feature energy = 26.0336 THEN reproduce
9. IF Resource Location = 3 THEN eat
10. IF Light Level $<$ Feature energy THEN sleep
11. **IF istrue(Reaching Resource) THEN eat**
12. **IF Resource Location $<$ 2 THEN turn left**
13. IF istrue(Reaching Resource) THEN turn left
14. **IF not(Reaching Resource) THEN go front**
15. IF Resource Location $>$ 1 THEN sleep
16. IF istrue(Reaching Resource) THEN turn right
17. IF Light Level $<$ 97.9668 THEN turn right

The analysis of the brain in listing 3 allows us to see that the agent has good food gathering behavior (rules 1, 11, 12, and 14), reproduces whenever it has more than 16.7159 energy (rule 7), and sleeps when the light level falls below 26.5859 (rule 4). If put in the environment with caves, this agent would clearly not survive, as it would fall asleep on a cave (from rule 4) and never wake up again.

Examining the brain presented in listing 4, we can finally see how the agents adapt to the environment with caves. The important rule in this case is the one at position 2. To explain the behavior induced by this rule, we need to take a closer look. Lets first consider that the agent is in a cave. If the agent doesn't have any resource within its vision range (very likely as in the cave the vision range is small), the value of *Resource Location* will be 0. As we know, in a cave the light level is equal to the minimum light level, which is 0. Therefore, this rule will make the agent move forward whenever it is inside a cave. In fact, we find this rule (or some small variation) in all the agents analyzed in runs where there is recovery of synchronization.

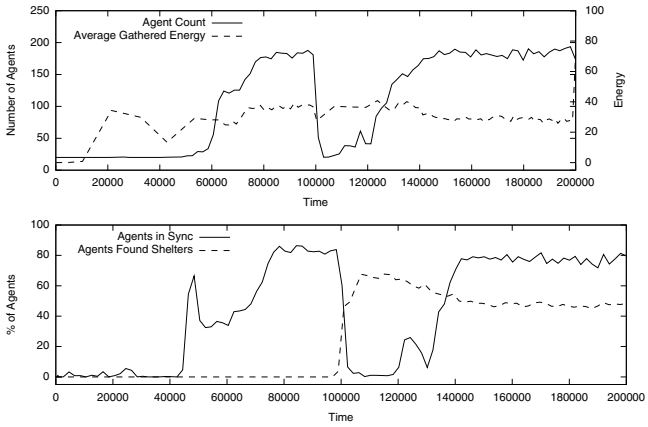

Figure 5: Plot of the evolution of the total number of agents in the population, average gathered energy, percentage of agents in sync with the day cycle, and percentage of agents that found caves, over the course of one simulation run (number of caves is 20).

The rest of the capabilities of the agent are also relatively easy to find in this rule list. From rules 1, 4, 13, and 14, we can see that the agent has a good foraging behavior. Rule 8 provides reproduction capabilities. And rule 16, when combined with rule 14, makes the agent sleep if the light level of the environment is less than 11.8449 and greater than 3 (maximum value for the *Resource Location* perception).

In table 2 we show an overview of the successful runs from all the tested simulation configurations. As stated earlier, we ran simulations with different values for the number of caves present in the environment. These results show that the configuration change doesn't seem to affect the number of successful runs out of the total of 30 runs. This was expected, as we were only changing the number of caves, which didn't affect the first environment. However, if we look at the number of runs where agents recover in the environment with caves, we find a different results for the three configurations. This is also rather straightforward to explain. With a smaller number of caves in the environment, the probability an agent has of finding a cave within its lifetime diminishes, making it easier maintain the synchronization from the first environment.

In figure 5 we show a run of the simulation with 20 caves. The plot is similar to that of figure 4, with the main difference being in the percentage of agents that find caves. In this case we can see that the percentage stabilizes at about 50%, whereas with the 50 cave configuration it stabilizes at about 80%. In the configuration with 10 caves, the percentage falls to about 30%. As expected, the less caves in the environment, the lower the probability of an agent finding a cave in its lifetime.

Listing 4 Example of the structure of the brain of an agent born at time 183513, for the simulation run shown in figure 4. Used rules are set in bold

1. **IF istrue(Reaching Resource) THEN eat**
2. **IF Resource Location = Light Level THEN go front**
3. IF Light Level = 29.2361 THEN turn right
4. **IF Resource Location = 1 THEN turn left**
5. IF Resource Location $<$ Resource Location THEN sleep
6. IF Feature energy $<$ Feature energy THEN sleep
7. IF istrue(Reaching Resource) THEN eat
8. **IF Feature energy $>$ 16.7159 THEN reproduce**
9. IF istrue(Reaching Resource) THEN reproduce
10. IF Light Level = Resource Location THEN eat
11. IF Light Level = 59.7254 THEN eat
12. IF istrue(Reaching Resource) THEN go front
13. **IF Resource Location = 2 THEN go front**
14. **IF Light Level $>$ 11.8449 THEN turn right**
15. IF Resource Location $<$ 0 THEN sleep
16. **IF Resource Location $<$ Light Level THEN sleep**
17. IF Resource Location $<$ 0 THEN reproduce

Table 2: Overview of the success of runs.

N. Caves	Runs	Successful	Don't Rec.	Recover
50	30	23	9	14
20	30	22	3	19
10	30	24	1	23

Conclusion

Even though it seemed unlikely for the agents to distinguish "night" from "cave" with only the light level as an input perception, evolution found a way to use the "tools at hand" to solve the problem. In this case, the agents take advantage of a specific feature of the environment created. As the light level, when inside a cave, is always zero, and outside a cave, at night, it is between zero and ten, the agents adapted to that fact. It is important to note that the scenario was not designed with this in mind, and in that regard that behavior was unexpected. This result may have a parallel in the real world, where it is common to find species that take advantage of specific properties of the environment, creating niches.

We believe these results are mainly due to the open-ended nature of the model used. The inexistent explicit fitness function allows the modeler not to over-specify and guide the solution, giving more freedom to the evolutionary system to produce viable solutions. Also, when compared with a fixed structure neural network, the brain architecture used may provide some added flexibility, and allow for these unexpected behaviors to evolve.

Regarding future work, even though the simulations showed that the agents adapt to the new environment without an internal clock, it would still be interesting to incorporate a clock source in the architecture of the agents, and compare the results of the simulations.

With these experiments we continued our investigation into the evolution of complex behaviors through open-ended evolution simulations. Following the results from the previous simulations (Baptista and Costa, 2008), where we showed that using the model of the BitBang framework we are able to evolve complex behaviors, from random initial conditions, in an open-ended evolution environment, we now also show that by simply adding extra complexity to the simulated world, the agents continue to evolve new behaviors adapted to their new environmental conditions.

References

Baptista, T. and Costa, E. (2008). Evolution of a multi-agent system in a cyclical environment. *Theory in Biosciences*, 127(2):141–148.

Baptista, T., Menezes, T., and Costa, E. (2006). Bitbang: A model and framework for complexity research. In *Proceedings of the European Conference on Complex Systems 2006*, Oxford, UK.

Beckmann, B., McKinley, P., and Ofria, C. (2007). Evolution of an adaptive sleep response in digital organisms. *Proceedings of the 9th European conference on Advances in artificial life*, pages 233–242.

Channon, A. (2000). Three evolvability requirements for open-ended evolution. In Maley, C. C. and Boudreau, E., editors, *Artificial Life VII Workshop Proceedings*, pages 39–40. Portland, OR.

Mirolli, M. and Parisi, D. (2003). Artificial Organisms That Sleep. In Banzaf, W., Ziegler, J., Christaller, T., Dittrich, P., and Kim, J. T., editors, *Advances in Artificial Life: Proceedings of the 7th European Conference on Artificial Life, ECAL 2003*, pages 377–386. Springer Berlin / Heidelberg.

Rand, D. A., Shulgin, B. V., Salazar, J. D., and Millar, A. J. (2006). Uncovering the design principles of circadian clocks: mathematical analysis of flexibility and evolutionary goals. *Journal of Theoretical Biology*, 238(3):616–635.

Russell, S. and Norvig, P. (2002). *Artificial Intelligence: A Modern Approach (2nd Edition)*. Prentice Hall, 2 edition.

Strogatz, S. (2004). *Sync: The Emerging Science of Spontaneous Order*. Penguin Books, London.