Natural Computing Project Progress

NEAT and (a)sexual evolution

Jules Kruijswijk
s4140023

Paul Bertens
s4512448

May 18, 2015

1 Introduction

There are many theories on the evolution of sex and on the advantages of sexual reproduction over asexual reproduction [1]. It remains an open question as to why sexual reproduction is so common in the natural world despite the relatively higher cost over asexual reproduction. One hypothesis, and the hypothesis that will be tested in this project, is that sexual reproduction does better in a faster changing environment [2]. One explanation for this is that sexual reproduction allows for faster adaptation in a fast changing environment due to recombination. With recombination groups of genes can be swapped faster which would result in higher variations and a higher chance of adapting to the new environment. This hypothesis has been explored before using Avida [2]. The study however is not very ecologically valid, since it performs on logical operations and its environment does not match the real world. We therefore will use NEAT [3], an evolutionary algorithm which operates on neural networks, in a simulated environment. Although the simulation is a gross simplification it is still closer to the real world than Avida and should give better insight into the advantages of sexual over asexual reproduction.

2 Methods

To see whether sexual reproduction is indeed preferred over asexual reproduction in a fast changing environment, an Artificial Life simulation was designed in Java. In this simulation, NEAT was used as an algorithm to evolve neural networks, which will be the brains of the agents.

2.1 Simulation

The simulated world is a simple 2D world with both food and poison represented by different colors. Agents, whose brains will be evolved using NEAT, will inhabit this environment and have an extra evolvable parameter that determines whether they will reproduce sexually or asexually. The input to the neural network of each agent will be two small 'eyes' that can read the pixel value in-front of them and the output will be their movement, one output for moving forward and one for turning. Agents can then reproduce if they surpass a certain energy threshold, energy can be gained by eating food, and lost when eating poison. If they reproduce asexually they can only create offspring with mutation, while if they reproduce sexually they can create offspring with cross-over and mutation. There will be two experiments, one where the environment changes slowly and one where it will change more rapidly. The environmental changes will be the food and poison, which will have some chance of switching and changing in value resulting in food becoming poison and vice versa. Thus in a rapid changing environment poison and food will switch more frequently and with higher magnitude than in a slow changing environment.

2.2 NEAT

To implement the NEAT algorithm, we will refer to the original paper [3]. Modeling and evolving Artificial Neural Networks (ANNs) in a genetic algorithm is not trivial, because the structure of different neural networks are not necessarily related. To perform crossover operations, the network structures have to be analyzed to find appropriate crossover points, and naturally the genomic representations for ANNs do not clearly show where these points are. With NEAT the authors have tried to overcome this problem by encoding history in each element of every single genome. This guarantees that structures are identifiable and thus analyzed more easily.

Each genome in the algorithm consists of a list of nodes and a list of arcs. Each arc has values for its input node, output node, the weight on the arc, whether the link is enabled or not and a so called innovation number. The innovation number is the historical value, which allows the crossover operations to detect if the gene is similar to another gene in a different genome. Each node contains a unique identification number, activation response threshold, whether it is enabled or not and also an innovation number.

Mutation can occur randomly in NEAT and it can change arc weights, activation thresholds and the network topology. The weights and thresholds can be changed when a random arc or node is chosen and the values are constrained. The network topology changes by randomly adding nodes or arcs. When an arc mutation occurs, the algorithm first randomly selects two nodes. If there did not exist an arc yet, it will then insert an arc with initial weight one. If an arc already existed, the only thing it will do is enable the arc again if it was disabled. And a last special case, if there is no arc between the two nodes, but an equivalent arc is already present in another genome of the population, this arc will be copied with the same innovation number - the reason for this is that it is not a new innovation. Mutation of nodes arrives in another fashion, as instead of randomly choosing two nodes, the algorithm now takes an arc, disables it and also adds a new node. This node, with a random activation threshold, will be connected to the nodes that were connected to the disabled arc - effectively creating two new arcs as well. The weight of the disabled arc will be copied to the output arc of the new node and the weight of the input arc will be set to one, such that it does not interfere with any learning that has taken place, yet.

Before crossover can be introduced, the innovation numbers have to be explained. Since the innovation numbers are not unique to genes specifically but only to the innovation (i.e. new gene), you can compare two genomes in the population and check if they share any genes. If any of the genes share the same number, they also share the same manifestation. NEAT stores the innovations in a database that have occurred since the initial population. Before a new innovation is introduced, the database will be checked if it does not already exist. If it is a new innovation, the database will store the innovation and increment a global innovation number which will pass on to that gene. This ensures that all related genes are eventually identical.

When a crossover operation is applied to two genomes, the new genome will receive the same innovation numbers in the genes. Because NEAT uses these innovation numbers, it can easily align two genomes to see which parts are similar and which are different. The genes that have different innovation numbers are called disjoint and added to the child during crossover. Genes that have newer innovation numbers are called excess and also added to the child during crossover. The genes that have shared innovation numbers are inherited by the parent genome that has the highest fitness value. The crossover operations in this algorithm limit the support for big diversity, as a new innovation, which can have a high impact, can be erased from the population before it reaches its potential.

To overcome this problem, NEAT uses speciation. In nature, speciation is defined as when two organisms once shared a genome sometimes diverge such that they can no longer mate with each other. When a genome diverges far enough (according to a certain formula) from that of the other genomes in the population, NEAT will find this genome and put it in its own species. It calculates the distance based on the excess and disjoint genes and the average weight differences of two genomes - each of these parameters are weighted. The initial population consists of one species. Following new generations, NEAT can identify a genome that should be in a new species and makes one accordingly. If other genomes in one species are diverging from their own species towards another, the algorithm will transfer this genome between species. Through certain fitness testing, NEAT can identify whether or not new genomes in a species have improved fitness compared to the original genome in that species. If a species has not improved over a certain amount of generations, it can be deleted entirely from the population - because this means that a certain innovation did not have enough impact. Based on the fitness value of the species, each species will have a percentage of spots for the new generation.

2.3 Analysis

3 Results

4 Discussion

References

[1] D. Misevic, C. Ofria, and R. E. Lenski, “Sexual reproduction reshapes the genetic architecture of digital organisms,” *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 273, no. 1585, pp. 457–464, 2006.

[2] D. Misevic, C. Ofria, and R. E. Lenski, “Experiments with digital organisms on the origin and maintenance of sex in changing environments,” *Journal of Heredity*, vol. 101, no. suppl 1, pp. S46–S54, 2010.

[3] K. O. Stanley and R. Miikkulainen, “Evolving neural networks through augmenting topologies,” *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002.