Coevolution of Noisy Environments and Governing Bitsets in a Cellular Automatonic Model

By Jonathan Pezzino
Junior, Nicolet High School

Question

The question that this study will aim to answer is *Can a one-dimensional cellular automata rule that successfully completes the density classification test for a large percentage of initial configurations be evolved in a noisy, rewired, dynamic, evolving environment through a genetic algorithm starting with an unbiased initial population of rules?*

Reasons for Research

The question posed by this research, if conclusively answered, could provide theoretical insight into the processes of evolution and other applications of network theory. If a null hypothesis were found to be true, it would certainly be a blow against the theory of evolution because the model used in the study will be theoretically analogous to the process of natural selection; if evolution does not form out of randomness, either something is wrong with the model or the theory of evolution will be undermined, if only in a small way. On the other hand, results showing the null hypothesis to be false would strengthen the theoretical foundations of evolution and possibly provide insight into the logic and processes behind natural selection, recombination, and random mutation.

Hypotheses

Although the hypothesis will be dependent on what specific algorithms are chosen to perform certain functions in the experiment, it is hypothesized that an successful rule will be evolved. This is supported by the results found by Moreira, *et. al* in their study "Efficient system-wide coordination in noisy environments." The results of this paper found that there are definite values of p and *n* with which it is possible to evolve rules

that complete the density classification task. Although this experiment will be adding in several important new factors, it will be fundamentally similar to the study done by Moreira, *et. al*. It is predicted that the additional factor of an evolving environment as well as evolving rules will benefit rather than hinder the completion of the density classification task, as evolution (genetic algorithms) tends to weed out solutions that do not help as much. On the other hand, this study will also be using unbiased initial populations, unlike the Moreira study, which will most likely hinder evolution. It is predicted, however, that this will be outweighed by the evolutionary push provided by an evolving environment.

Background

Cellular Automata; Coordination & the Density Classification Task; Genetic Algorithms

One of the difficulties in physics today is that many systems are difficult or impossible to describe in terms of traditional formulas due to the fact that the interactions between their constituent parts are unpredictable in a traditional sense.[1] One of the solutions to this problem is to instead model the interactions of the parts of the system with a computational tool called cellular automata. In cellular automata, the interactions between neighboring nodes contained in a 1+ dimensional lattice according specific rules generate behavior that can then be applied to a real life model or described by a formula because of the reduction of complexity[2]; only vital components of the system are simulated.

The simplest type of cellular automata is a group of doubly linked nodes in a one-dimensional lattice whose values can be either 1 or 0 (the number of states a cell can

1 Wolfram, Stephen. Theory and Applications of Cellular Automata.
2 Wolfram, Stephen. Cellular Automata and Complexity.

assume is represented by the variable k, thus k=2)[3]. Each cell has access only to itself and its two neighboring cells (the number of cells to each side that a cell has access to is represented by the variable r, hence r=1). Based on this information, rule sets, known as bit sets when k = 1[4], describing the cells' behavior can be formulated. These rules can be applied to a lattice consisting of N cells, and the result is called the next generation. Bit sets are strings of 1s and 0s that correspond to a cell's output for the next generation for the n^{th} possible combination of neighbors a cell can have. For example, if a cell were operating under bit set 10010010 and its combination of neighbors were 001, then its output bit would be 1 because the number represented by the neighbor combination is 1 in binary; the output bit is therefore the first digit (counting right to left and starting at 0) of the bit set.. The result of the application of a rule to a given lattice over a number generations can create a model that exhibits extremely complex behavior despite the simple rules behind the operation. One such behavior is coordination, where all cells in the lattice assume the same state (either all 1's or all 0's)[5]. The most efficient strategy for coordination across the lattice is the so-called "majority rule" where a cell will assume the state assumed by the majority of its neighbors.

As a general rule, N is usually an odd number.[6] This is because of a feature called the periodic boundary condition[7], wherein the neighbors of the first and last cells in the lattice are the last and first cells, respectively, effectively linking the lattice such that it becomes a circle of nodes.

One modification that is sometimes made to one-dimensional cellular automata

3 Wolfram, Stephen. Cellular Automata and Complexity.
4 Various Authors. "Genetic Algorithms."
5 Franks, Alexander. "Understanding Evolved Strategies...in Noisy Environments."
6 Mitchell *et. al*. "Evolving Cellular Automata...Recent Work."
7 Various Authors. "Cellular Automata."

environments is the introduction of noise. Noise is represented by a probability *n*. While accessing information about its neighbors, a cell will have *n* probability of misreading its neighbor. This convention is seen in natural and social systems; organisms sometimes perform incorrectly with neither rhyme nor reason.

Another modification that can be added to a one-dimensional environment is a rewiring probability. After being initialized, each node in the lattice is rewired with probability *p*. If a node is rewired, one of its links is randomly distributed to another node. If the immediate links between node represent social interaction in a social network, rewiring is analogous to individuals who socialize across several cliques of other individuals because they have connection to other nodes farther away in the lattice.

One application of the above model is in genetic algorithms. The concept behind genetic algorithms is the 'natural selection' (*fitness evaluation*) of 'fit' algorithms that can be described in terms of a number or string (*chromosomes*), the 'breeding' of these algorithms to create 'offspring' (with the addition of random 'mutations') *(genetic operators*), and the repetition of the above process over a number generations[8].

The question asked by Alex Franks in his study "Understanding Evolved Strategies for System-Wide Coordination in Noisy Environments" could be summarized "Starting with an initial, non-majority-rule set, can a genetic algorithm generate a rule that is x% similar to the majority rule?" In cellular automata, rule sets are analogous to algorithms, and in Alex's study fitness was tested by determining the percentage of an arbitrary number of initial lattices that a rule set could successfully coordinate.

Alex used an environment with k=2, r=3, N=99, and variable p and *n* values. The first generation of rules was chosen in an arbitrary manner to insure that the population

8 Various Authors. "Genetic Algorithms."

had a fair chance of evolving to the majority rule. Populations' efficiencies were evaluated by testing their success at classifying 100 arbitrary initial configurations; a chromosome's efficiency was the percentage of initial configurations for which it successfully performed the density classification task.

Experimental Design

The experiment will follow a genetic algorithm procedure using a computer program written in Java. First an unbiased initial population of rules will be randomly generated. This is a crucial point because previous similar experiments have all used a biased initial population (using a flat distribution of 1s and 0s), providing the system with an evolutionary 'push' in the right direction. This experiment will seek to discover if the need for such a push can be eliminated if compensation is provided by the environment's ability to evolve. An initial population initial configurations will also be chosen with variable noise and rewiring probabilities. The information vital to these chromosomes will most likely be rewiring configuration, but this aspect of the experiment still needs to be determined. This population of initial configurations will coevolve with the rule population.

Both rule and configuration populations will be tested for fitness. It is possible that all rules will be tested using the population of ICs (initial configurations) as the test environments, but this also has yet to be determined. The most fit rules and ICs will then be selected and bred using a modified crossover algorithm and then mutated to create individuals to populate the next generation. This process will be repeated over a number of generations under varying noise and rewiring probabilities. The efficiency (percentage of ICs successfully classified) of the final evolutionary products will be plotted using

colors against rewiring probability and noise amplitude, as in the study done by Moreira *et al.* Data collection will be a trivial matter because of the ease of automating it using the same program that performs the genetic algorithm procedure.

The size of the environment (N) will always be 99. It is possible that k (neighborhood size) will be varied, but it will most likely be kept constant at 7, as in Franks's study. This means that rule bitsets will all be of length 2^7=128 bits.

Timetable

Many of the specific aspects of this experiment still need to be worked out, but the expected time for this project could be up to six months, based on the timetable of Franks's study. Writing and debugging code can be a very lengthy process, taking up to several months. Additionally, running the program to perform the genetic algorithm is takes a long time because of the sheer number of repetitions and iterations it must perform on vast amounts of data. It is estimated that it may take a week for the program to complete one run. Another time-consuming aspect of this project will be data organization and analysis. There are going to be a significant number of variables that must be accounted for and a huge number of trials performed.

Unfortunately, this time requirement means that the project will not be ready for this year's Siemens-Westinghouse or Intel competitions, which are in late September and late November, respectively. Although it's possible that the project will be ready for the Symposium (mid-to-late January), it's an unlikely scenario. The upside of this is that there will be plenty of time to prepare the project for next year's competitions without feeling rushed, a definite advantage.

Resources

No special resources will be required for this project. All that will be needed is a computer running a Java compiler (which the researcher already hasmome) and a fair amount of time invested. The most valuable resource will be consultation with Professor Luís Amaral, who will likely be partly supervising the experiment since it is essentially a continuation of his study in an area of science in which he has made many monumental advancements and discoveries.

References

Wolfram, Stephen. Cellular Automata and Complexity.

Wolfram, Stephen. Theory and Applications of Cellular Automata.

http://www.pnas.org/cgi/doi/10/1073/pnas.0400672101. Moreira, André; Mathur, Abhishek; Diermeier, Daniel; Amaral, Luís. "Efficient system-wide coordination in noisy environments." PNAS.

http://chem-phys.com/intel/Alex.pdf. Franks, Alexander. "Understanding Evolved Strategies for System-Wide Coordination in Noisy Environments."

http://www.cs.pdx.edu/~mm/evca-review.pdf. Mitchell, Melanie; Crutchfield, James P.; Das, Rajarshi. "Evolving Cellular Automata with Genetic Algorithms: A Review of Recent Work."

http://en.wikipedia.org/wiki/Cellular_automata. Various Authors. "Cellular Automata." Wikipedia.

http://en.wikipedia.org/wiki/Genetic_algorithms. Various Authors. "Genetic Algorithms." Wikipedia.