Evolve Cellular Automata Rules Using Genetic Algorithms
Complex Adaptive Systems, CS 591
Assignment 2
Due April 1, 2009

Your assignment is to evolve CA rules using a GA. You get to choose the CA, the measure of fitness toward which you are evolving the CA, and the parameters for the GA. To get full credit you must

1) Describe the rules for your CA, the fitness measure you’re using, and explain why you’ve chosen that fitness measure.

2) Explain how you encoded CA rules in your GA. For example, you can represent every rule explicitly, as in the 1 dimensional density classification CA in Mitchell et al that uses 2 states and a neighborhood size of 7. You could feasibly represent all rules for a 2D CA with 2 states and a neighborhood size of 9. Alternatively, with larger k and N, you can represent a ‘sum rule’ in your CA.)

3) Describe the parameters for the GA (e.g. mutation rates, whether you’re using tournament selection or some other method, number of generations, etc.), and why you’ve chosen those parameters.

4) Plot the average and best fitness of individuals in your GA over the generations that you run the GA.

5) Plot some meaningful characteristic of the best CA rules that have been evolved at each generation (e.g. lambda, or the length of the longest transient period, or the number of infected cells), and explain why you’ve chosen to plot that characteristic.

6) After you complete the above steps for a simple GA, incorporate an additional evolutionary component to your GA. For example, you can evolve GA parameters or implement a messy GA or allow the length of your strings to evolve. Repeat Steps 4 and 5, and explain whether you were able to improve the GA performance in some way.

You may chose any CA you like—Examples include a 1D CA that solves the density problem, a 2D SIR CA where the goal is to evolve rules that maximize (or minimize) the number of infected cells, or you could implement a 2D 2state CA and see if you can specify fitness criteria that cause the Game of Life rules to evolve. You can also chose a novel CA that we have not discussed in class.
Guidelines:

You are strongly encouraged to work in pairs. You must report any code you use that you did not write yourself. Sample Matlab GA code will be posted, but you can use any language to implement the CA and GA.

Reports Guidelines:

Your report should be between 5 and 10 pages of text, excluding code and figures. Figures should be embedded into your report. All figures should include a descriptive caption, figure axes should be clearly labeled with legible text. You should include your CA and GA code in an appendix. The body of the report should include the following:

1. (~1/2 page) Introduction: State the problem you are trying to solve in your own words and explain why your approach is useful.

2. (1-2 pages) Methods: Describe
   a) The neighborhood size and number of states for your CA
   b) The parameters of your GA (what kind of selection, crossover, mutation, etc.)
   b) how your bit string encodes rules for the CA and
   c) the basic parameters for each of your GA runs (population size, generation time, crossover rate, mutation rate, etc.) and
   e) Your fitness function.

3. (2-6 pages) Results: Report
   a) a description of how your population of strings improved over time, with figures showing fitness vs. generation of your GA
   b) a description of how your CA changed over time, using a plot of some feature of your best CA (e.g. lambda, length of a transient) vs GA generation
   c) the best, or best few, strings you evolved for each experiment, the fitness of those strings, and a description of the CA behavior with those strings
   d) Repeat a, b and c for your more complicated GA

4. (1-2 pages) Discussion and analysis of your results: Explain
   1) Explain the behavior of the most fit CA’s that you can evolve. What are common features among the best rule sets, and why do they maximize fitness?
   2) Explain why a GA is or is not a good way to find an interesting CA rule set.
   3) Discuss how easy or difficult it is to find a set of rules that maximize the fitness you defined.
   4) Summarize the most interesting results of your experiments.

5. Provide your well-commented code and instructions for running it in an appendix.