6.836 Embodied Intelligence–Research Assignment 4

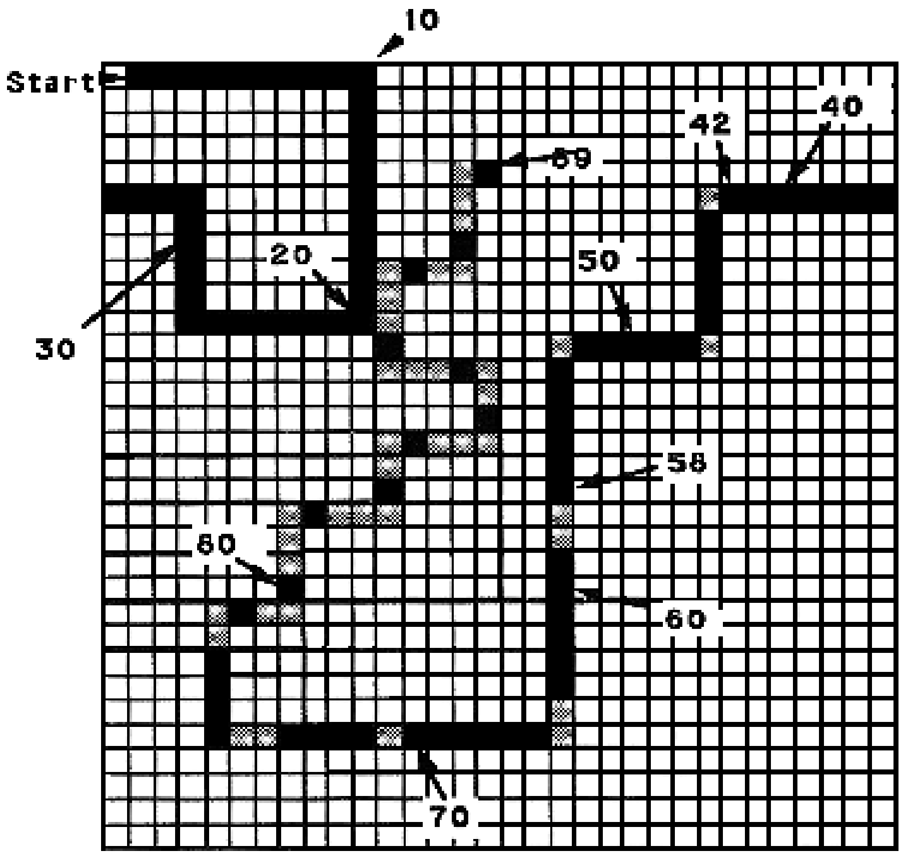
Massachusetts Institute of Technology

Due: Friday, April 12th, 2002

Please turn in EACH problem on a SEPARATE page. Please hand in a copy of the code you write for the problem set (it doesn't have to be pretty).

Please read the whole assignment before you start. **Note that this assignment is due in three weeks.**

During lecture we looked at evolutionary systems that evolved an 'ant' to prowl over a 2D grid of cells with toroidal boundary conditions (wrap-around), following trails of food. (We also handed out a paper on this on March 15.) One trail that has become something of a *de facto* standard is the "John Muir" trail illustrated in the following figure:



Cells are either empty or full (the light gray cells in the figure illustrate the optimal path, but are as empty as blank ones). The ant starts in the cell marked *Start*, facing right ("east"). Thereafter it is always in a cell and facing in one of four possible directions. At the beginning of each time-step it gets one bit of sensory information: whether there is food in the cell in front of the cell it currently occupies (i.e., the cell it would move to if it moved forward). At each time-step it has one of four possible actions. It can (**F**) move forward one cell; (**R**) turn right ninety degrees without changing cells; (**L**) turn left ninety degrees without changing cells; or (**N**) do nothing. If an ant moves onto a food-cell, it consumes the food and the food disappears; when the ant leaves that cell, the cell is empty. The fitness of the ant is rated by counting how many food elements it consumes in 200 time-steps. There are 89 food cells in the John Muir trail.

For the following exercises, files containing a representation of the John Muir trail, and the alternate Santa Fe trail, can be found on the course web pages (`http://www.ai.mit.edu/courses/6.836/handouts/handouts.html`) under the info for Research Assignment 4. There are an additional 1,024 trails generated randomly there also in files named `aa000.txt`, `aa001.txt`, and `bb000.txt`, etc. Each of these trails has 89 food cells and none of them ever cross themselves. All the trails are random and there should be no systematic differences between members of the two sets of trails. Feel free to reshuffle them into two sets any way you choose, if you want.

1. The sensory-motor coordination for the ant can be implemented using a finite state automaton (FSA). Design a 'genetic encoding' representation, specified by a fixed-length bit string, that encodes ant sensory-motor FSA transition tables. Then, by hand, try to design the best possible ant for the John Muir trail. Show the encoding and the corresponding FSA state-transition table and diagram. What fitness does your ant controller score?

Note: A blind ant with an 89 state FSA could traverse that particular trail perfectly — but would probably get very lost on another trail. Limit yourself to, at most, a 16-state FSA.

2. Does your ant use the **N** operation? Under what circumstances might the **N** operation be useful?

3. How many different individuals are possible in your representation?

4. How does your ant fare on the "Santa Fe" trail?

5. Generalize your representation to allow 2^n states. Express the number of bits in your representation, and the number of possible individuals, as functions of n.

6. Write an evolution program that can evolve individuals (for the John Muir trail) using your representation. Include multi-point crossover and mutation. Decide on a fixed number of states with which to run the program. (You do not have to limit yourself to 16 this time, but think twice before making it too large.)

6a. Write an outline in English of how your code works and what representations you use. Tell us about the algorithm for selection of individuals for the next generation. Run your system and plot how fitness increases by generation.

6b. Make your system work on some number of the `aannn.txt` files and evolve your ant. Take one of the highest scoring ants evolved and try it on the `bbnnn.txt` files. Report the results of this to us. If there is a significant difference what could be the explanation?

6c. Generate some tables on how the overall fitness varies on each set of files for different population sizes, different parameters for what proportion of the fittest individuals are retained, what proportion are used for reproduction, and what levels of mutation are used.

7. Take one of the best individuals your system produced for each of problem 6a and 6b (i.e., their FSA state-transition tables and diagrams), and analyze its behavior. Does the problem 6a ant demonstrate any particular specializations? What about the 6b ant? And if it does better on the "a" files than the "b" files, what might be going on? How do both compare to your hand-coded solution from Problem 1?

(Bonus, just for fun). Suppose you had evolved your hand-designed ant from Problem 1. Is there a series of one-bit mutations from an all-zero string to your best hand-designed machine such that the fitness never decreases at any step? Is there a series of one-bit mutations from an all-zero string to your best hand-designed machine such that the fitness always increases at every step? How many possible one-bit mutation evolutionary paths are there, from the all-zero species to your hand-designed machine?