Lab exploration 6: Evolutionary algorithms
Math 309 Fall 2019
Deadline: class 21 October

- Conduct experiments as indicated.
- **Journal entry.** Respond to each of the "journal queries." Using *concise and clear sentences*, incorporate data, symbols, and illustrations into your text. Have an audience in mind. Focus on *developing* an explanation or argument that stems from your simulations.

  Submit 300-400 words 2-3 pages double-spaced in **hard copy**.
- **Recommended.** Work in groups of 2 or 3. Submit one journal entry for the group.
- **Suggestion.** Before running the simulations, read the "What is it?" and "How it works" sections under the Info tab.

Model: Simple Genetic Algorithm. (Location: Models Library/Sample Models/Computer Science.) The basic task here is to find a string of all 1s among a population of strings of 1s and 0s (represented as vertical segments that are white and black). A string is considered more fit than another if it contains more 1s. The model evolves strings by means of cloning a single string (selected for its fitness) or mating two strings (selected for their fitness). Mutations can occur at a specified rate as can crossover (how much of a given parent string is included in the offspring string). The number of time-steps required to produce the all-1 string measures the speed of evolution.

6.1 Journal query.

With mutation-rate set at 0, look for settings for population-size and crossover-rate for which the algorithm finishes (that is, obtains all-1).

6.2 Journal query.

Set population-size=200 and crossover-rate=50. Estimate a value of mutation-rate that minimizes the number of time-steps to completion. Does the value that you found make evolutionary sense?

Model: Robby the Robot. (Location: Models Library/Sample Models/Computer Science.) As discussed in class, this is the can-collecting robot whose protocol-directed actions evolve by application of a genetic (evolutionary) algorithm (GA). Notice the parameters that you can set:

number-of-generations: how many times the population will be reconstituted by mating and mutation when the go-n-generations button is pressed

population-size: how many protocols (called strategies in the *NetLogo* model) there are in each generation

mutation-rate: probability that a random change is introduced into a new protocol.

You can watch the best fitness score as it develops both graphically and numerically. For what it's worth, symbolic descriptions of the protocols that are running are visible in the display on the right.

6.3 Journal query.

Set

number-of-generations = 100 population-size = 100.

With mutation-rate $= 0$, run the GA for 100 generations and note the **best fitness score** (BFS) achieved. Do the same with mutation-rate $= 1$. How do the two BFSs compare? Is the outcome reasonable?

For values of mutation-rate from .1 to .9 incrementing by .1, run the GA and record the BFS for each. (The data will be more robust if you run the GA several times for a given mutation-rate and then take the average.) On the interval $[0, 1]$, plot the BFS as a function of mutation-rate. In which interval $[.k, .(k+1)]$ is the BFS maximal?

6.4 Journal query.

On the interval $[.k, .(k+1)]$, find a BFS for the mutation-rates

$$.k + .01,\ .k + .02,\ \ldots,\ .k + .08,\ .k + .09.$$

Plot the BFS as a function of mutation-rate on $[.k, .(k+1)]$ and estimate a rate that maximizes the BFS.

6.5 Journal query.

How sensitive is the BFS to the rate of mutation? That is, can relatively small changes in mutation-rate lead to a large change in BFS?

6.6 Journal query.

What does the Best Fitness plot reveal about the relationship between BFS and number of generations? How do the plots change with mutation-rate?

6.7 Journal query.

Note that you can test the **most fit protocol** (MFP) by pressing

view Robby's environment and then step through best strategy.

How much variability is there in the scores achieved by the MFP over five runs where a run executes the strategy until a steady state appears or all cans are gone? Is there a "simple" description of how some MFPs behave?

6.8 Journal query.

Does the MFP ever *not* pick up a can when it could do so? How could such an action be better than always picking up a can when available?

Bonus.

By adjusting parameters, generate a protocol that achieves a fairly high best fitness score (say, above 250). When you run the protocol, look for and describe behavior that seems somewhat clever—maybe something that's not intuitively obvious.

Project idea. Can a protocol benefit from having the robot revisit a site? Can site re-visitation be excluded by a protocol? That is, can a protocol "remember" which sites it visits?