Bio-inspired Artificial Intelligence (BAI)
Prof. Dario Floreano

BAI laboratory exercises, 29.9.2009

Genetic algorithms

Author: Daniel Marbach

Responsible assistants (contact for questions):
Daniel Marbach (daniel.marbach@epfl.ch)
Thomas Schaffter (thomas.schaffter@epfl.ch)

Goal. The goal of this lab is to understand some advantages and pitfalls of genetic algorithms (GAs). In particular, you will observe the effects of the mutation rate, crossover rate, selection pressure, and population size in artificial evolution, and reflect to what extent these observations also apply to biological evolution.

Getting started. Download the file GA_exercises.zip from the website of the course and unzip it. This lab is done using the software Matlab, a basic understanding of Matlab is required.[1] Each exercise has a corresponding m-file. To solve the exercises, you will have to open, edit, and run these m-files.

In this lab, the genome of an individual is always a vector of real-valued parameters $\mathbf{x} = [x_1\ x_2\ \ldots\ x_N]^T$ (keep in mind that there are other types of encodings, e.g. bit strings, which will not be explored in this lab). The fitness of an individual is given by the fitness function $f(\mathbf{x})$. Here, the aim of the GA is to *minimize* the function $f(\mathbf{x})$, i.e., to find the vector $\mathbf{x}_{min}$ that has the lowest value $f(\mathbf{x})$. In other words, *lower* values $f(\mathbf{x})$ correspond to a *better* fitness.[2]

Tips. If you do not know what a Matlab function/command does, type "help mycommand" in the command window. If your workspace is getting too cluttered, you can delete all variables, clear the console, and close all figures using the commands "clear", "clc", and "close all", respectively.

Exercise 1

In this first exercise, we will not yet run a complete GA. Instead, we consider a single parent individual $\mathbf{x}_0$, from which a number of offspring individuals are

[1] If you are unfamiliar with Matlab, this is a good opportunity to change this. In the menu "Help", select "Product Help" and read the section "Getting started".

[2] The Matlab GA toolbox does function minimization because other Matlab function optimization tools also do minimization by convention. Note that if you want to maximize a function $f(\mathbf{x})$, you can always do it by minimizing $-f(\mathbf{x})$.

created using a Gaussian mutation operator (which adds a random number from a Gaussian distribution with mean zero and standard deviation σ to each parameter x_i of the parent). The fitness function is defined as

$$f(\mathbf{x}) = \frac{\sum_{i=1}^{N} x_i^2}{N}$$

This fitness function is unimodal, it has a single global minimum at the origin. We will analyze the effects of mutations on the fitness depending on the value of the parent $\mathbf{x}_0$, the mutation rate (the standard deviation σ), and the number of dimensions N of the search space.

Exercise 1.1

Run the m-file to generate offspring from a single parent $\mathbf{x}_0$ using a Gaussian mutation operator (which adds a random number from a Gaussian distribution with mean zero and standard deviation σ to the parent). Generate offspring from different parents (e.g. $\mathbf{x}_0$=0.1, 1, 10) using different mutation rates (standard deviations σ). First consider the one-dimensional case, then two dimensions, and finally many dimensions (e.g. N =100). For one or two dimensions, the fitness landscape with the parent and the offspring is shown. For more dimensions, a boxplot with the fitness of the offspring is shown where the green, dashed line is the fitness of the parent (type "help boxplot" if you are unfamiliar with boxplots).

- Do the mutations tend to improve or worsen the fitness of the parent?
- Are low or high mutation rates best for improving the fitness? How does this depend on the initial value of the parent and on the number of dimensions of the search space?

Exercise 1.2

Run the m-file to confirm the observations that you did qualitatively in the previous exercise by plotting boxplots with notches side-by-side to evaluate the statistical significance of observed differences. Compare different values for

- The number of dimensions of the search space
- The value of the parent (how close it is to the optimum)
- The mutation rate

If you vary one of these three parameters, *make sure that you set the other two at a constant value* (otherwise it may be difficult to interpret your results). Try to confirm the answers that you gave in the previous exercise.

Exercise 2

We will now use a GA to find the minimum of the unimodal fitness function defined in the previous exercise and analyze the effect of the mutation rate and the dimensionality of the search space on the results.

Exercise 2.1

Run the GA first on the one-dimensional function. How close is the best individual from the global optimum? Increase the dimensionality of the search space to two and more. How close are the best individuals now from the global optimum? Can you get as close as in the one-dimensional case by modifying the mutation rate and/or the number of generations?

Exercise 2.2

Run the m-file to do three batches of 10 runs of the GA at different mutation rates (it may take a minute). The boxplot compares the best fitness values obtained in the three conditions. Explain the result.

Exercise 3

In this exercise we will analyze the effect of crossover in the GA. An offspring individual is formed from two parent individuals $\mathbf{x}_1$ and $\mathbf{x}_2$ by randomly taking the value for each entry x_i either from $\mathbf{x}_1$ or $\mathbf{x}_2$. The GA has a parameter defining the fraction of offspring that is created using crossover at each generation. Note that the mutation operator is not applied to individuals that were created using crossover in the Matlab GA.

Exercise 3.1

Using the same setup as in the first two exercises, this m-file does 3 runs using mutation only (as in the previous exercises), and 3 runs using crossover only. The boxplots compare the best fitness values obtained in the two cases. Explain the result.

Exercise 3.2

Compare the best fitnesses obtained by varying the fraction of offspring created using crossover only. Is there an optimal crossover fraction for this fitness function? Interpret the results.

Exercise 4

We will now investigate the effect of the selection pressure. In the previous exercises, we were using tournament selection with a tournament size of 2. Run the m-file of this exercise to compare the best fitness values and the distances from the global optimum obtained using tournament sizes 2 and 10.

- Which tournament size gives better results for the fitness function fsphere and why?
- Which tournament size is better for the fitness function frastrigin and why?

Exercise 5

The m-file of this exercise contains many test functions commonly used to benchmark optimization algorithms. Run the GA on some of these test functions (especially the multimodal functions) and adapt the mutation rate, crossover rate, selection pressure, and population size so as to get the best results. Read the comments in the code, sometimes they indicate a suitable initial range for the initial population. You may first try the 1D or 2D case, which has the advantage that the fitness landscape can be visualized. However, keep in mind that sometimes the resolution of the plot is not sufficient to accurately represent a function.

Conclusions

- Are there optimal parameters for a GA?
- What are the advantages and disadvantages of
    - Low/high mutation rates?
    - Low/high selection pressure?
- Based on these observations, do you think there is an optimal mutation rate for a biological organism? Do mutations typically improve or worsen the fitness of a biological organism? In which situations do you think low/high mutation rates are advantageous for a population of bacteria?
- What is the genotype and what is the phenotype in the problems considered in this lab?