CS 5320/4320

Project #1

Points 100, Due: 11:59 pm, Feb 14

Solve the following problem using a genetic algorithm (GA):

Minimize $f(X_1, X_2, X_3) = X_1^2 + X_2^2 + X_3^2$ for $-1.0 <= X_i <= 5.0$, $i = 1, 2, 3$.

The goal is to find values of X_1, X_2, and X_3 (in the given interval) that will cause f(.,.,.) to have the minimum value. Use real (floating-point) values, not bit-string representations. Use proportional selection, single-point crossover, and gene-wise mutation (as discussed in class).

Use the following parameter values:

$N = 30$; $p_c = 0.8$; $p_m = 0.1$; max generations (for use as a stopping condition) = 50.

For a "typical" **single** run, report the following:

a. At intervals of 10 generations, the best and the worst fitnesses (the f values) in the current generation and the corresponding X-vectors; the average fitness of all the members of the current generation;
b. The best-of-run solution (the vector and its fitness).

Next, collect data on 30 **independent** runs, and then report:

a. At intervals of 10 generations, the following averages (and the corresponding standard deviations) over 30 values: the best-of-generation fitness, the average-of-generation fitness;
b. The average (and the corresponding standard deviation) (over 30 values) of the best-of-run fitness.

That is, the three quantities in (a) and (b) above are each an average of 30 values.

Use Python/Java/C/C++ as the implementation language. (If you do not know any of these languages, please contact me immediately. I teach Python CS 4200 every Fall.) Please do not use an off-the-shelf implementation of GA from a package (such as Octave or Matlab). The goal is for students to develop a sound understanding of the working of the GA heuristic. Please ensure that your program's output is exactly reproducible (by recording, maybe in a file, the initial seed(s) of your random number generator). Do not use the system time or something similar as the initial seed.

This is a group project, with three students per group. There will be only one submission from a group (it doesn't matter which member submits it; the submissions of the other members will remain blank on Canvas.) Please write the names of the group members at the top of the very

first page (or cell) of the submission. Form your own groups by interacting amongst yourselves but please do NOT use Canvas's features to store group compositions. A student may choose to be in different groups for different projects. Working in groups is highly recommended but not mandatory; a student may choose to work independently.

Please submit on Canvas a single doc/docx/pdf/ipynb file (no other file type, please, with the following exception: if you are submitting an ipynb file, please ALSO submit the corresponding html) containing the source code and all output. Submission of multiple files is discouraged but may be resorted to only if you absolutely cannot manage to produce a single file. At the beginning of your report, you may add notes for any other special issues/techniques that you think might be important in your implementation.

Please keep the following issues in mind:

- code correctness and completeness
- replicability
- decisions made in the choice of algorithm parameters (it is important for you to understand and appreciate the effect that parameter settings may have)
- scalability (how easy is it to modify your code to incorporate any number of variables)
- program readability, maintainability, and software engineering issues (e.g., how easy is it to alter something – for example, the mutation policy -- in your implementation, are you hard-coding, etc.)
- [after you have a solid grasp of the basic stuff] in preparation for the next project, please consider how you can keep your code modular so that the same program can handle (a) using one out of multiple crossover schemes, (b) one of multiple selection strategies, (c) one of multiple mutation policies, (f) one of several fitness functions, (g) a variable chromosome (vector) size, etc.

I will provide further explanation on some issues in the lectures. You will find numerous implementations of GA on the Internet; feel free to check them, but write your own code.