Project 3: Diversity and Difficulty

Due: 4/19/2022

Total: 100 points.

Invocation (same as previous project):
> sga [-h] [-g] [-G] [filename]
with filename being the name of your settings file. If this filename is left off, by default it should use a settings file called “gasettings.dat”. All of the arguments are optional

-h : Should output a help message describing what the options do and then terminate..

-g : Limited debugging information should be displayed while running.

-G : Full debugging should be displayed while running.

The settings file is same as before, with additional options described in this project

Task 1: Add diversity preservation: Restricted Tournament Replacement

I want you to add a new diversity preservation mechanism to your options. In this case, Restricted Tournament Replacement. We discussed Restricted Tournament Replacement in our multimodal notes, but the general idea is to first produce all your offspring population. Then, first select an individual in our offspring population, let’s call it X0. Then, select a random subset W of size w from our original population. Now find the individual in our subset W that is most similar to X_0 by hamming distance (difference in bits). If X_0 is better in fitness than that individual, replace it. Otherwise, discard X_0. Repeat this for all individuals in our offspring population. The resulting population is our new population for the next generation.

Remember that hamming distance is the difference in bits between two strings, not the difference in the number of bits.

You should add two new parameters to your parameter file. One should be a way to select our replacement method, with 0 being the default with elitism and 1 being RTR. You will also need to add a parameter for the size w of our subsets used for RTR. If it is not specified, have it set to the string size divided by 10 by default.

Task 2: Add a new fitness function, interleaved trap

You will be adding a new fitness function to your GA, namely interleaved trap. For this trap, the individual trap partitions are spread amongst the string. So for example, for trap 4 and a string size of 20, that would mean you would have 5 total partitions in your string. So the first 5 bits of the string would be the first bit of each partition. Then the second 5 bits would be the second bit of each partition and so on. This is to enforce that the bits of each partition are maximally spread across the string.

TASK 3: Experiments

Experiments:
I want you to experiment with the GA in order to see what operators and replacement methods work well and what do not for onemax, trap-4 and interleaved trap4. To do this, use bisection to get an idea of what the minimum population size is required to solve under the given settings.

You can use the same sizes as you did before, so string size of 24, then 48, then 100. If you are having trouble solving at these sizes, try a string size of 12 and 16. If 100 seems trivial, you can scale up from there.

What parameters should you vary? You should try all our different crossover operators (uniform, two-point, one-point). You will then want to try for the different replacement methods of elitism and RTR.

Do not be shocked if RTR performs worse in some cases. Remember, RTR preserves diversity, which could slow down convergence.

Submission:
I expect the following:

1) A readme.txt file describing how to compile and run your project. This file should also outline briefly the results of your experiments above.

2) A word document or pdf (called results.pdf or appropriate suffix) describing your experiments and results of bisection on all the different settings variants.

3) A summary at the end of this word document detailing any conclusions you can draw from this.

4) All your source files

To submit, put all these files in a folder and compress them using tar or zip. Then attach the archive to canvas.