2MP3 Assignment 2

Developing a Basic Genetic Optimization Algorithm in C

Jacob Foster

1 Programming Genetic Algorithm

Table 1: Results with Crossover Rate = 0.5 and Mutation Rate = 0.05

Pop Size	Max Gen	Best Solution			CPU time (Sec)
		x_1	x_2	Fitness	
10	100	0.150609	1.064333	3.417500	0.001000
100	100	0.014191	-0.054170	0.014191	0.003000
1000	100	-0.032807	0.010529	0.128780	0.026000
10000	100	0.009308	0.004730	0.032433	0.267000
1000	1000	0.009919	0.012055	0.050633	0.249000
1000	10000	0.000458	0.000458	0.001842	2.464000
1000	100000	-0.000153	-0.000153	0.000612	24.919000
1000	1000000	-0.000153	-0.000153	0.000612	249.718000

Table 2: Results with Crossover Rate = 0.5 and Mutation Rate = 0.2

Pop Size	Max Gen	Best Solution			CPU time (Sec)
		x_1	x_2	Fitness	
10	100	-0.093844	0.007782	0.486153	0.001000
100	100	-0.028840	0.019379	0.130161	0.002000
1000	100	0.007782	0.016938	0.061951	0.030000
10000	100	0.002289	0.001068	0.007314	0.233000
1000	1000	0.000458	0.007782	0.023667	0.261000
1000	10000	0.001068	0.001068	0.004333	2.621000
1000	100000	0.000153	0.000153	0.000612	25.163000
1000	1000000	0.000153	0.000153	0.000612	259.556000

1.1 Report and `Makefile` (3 points)

The makefile attached in this assignment’s submission is essential to running the genetic sequencing algorithm in an autonomous manner. To run the makefile, firstly type ”make clean” in the terminal to run and provide a clean reset, followed by ”make” to run the file.

The technical function of the file can be broken into sections as such. The variables CC, CFLAGS (-Wall, -Wextra, and -std=c99), SOURCES, OBJECTS, EXECUTABLE, and LIBS represent the GCC compiler, the compiling flags, the source code files, the object files, the executed file, and the imported libraries respectively.

The executable file target declares the the default target as all, and is influenced by the object files target. To ensure the makefile successfully executes the executable function, it must be built with the same compiler as outlined in the file and link to all libraries and object files. Furthermore, it must follow the ".o: .c" format since it was programmed in the C language. This ensures that the compiler is able to successfully interpret the executable file.

The last section of the code, "Clean", removes any excess object files that were previous compiled and ensure an uninterrupted execution.