Intelligent Systems (Fall 2012)
Assignment 4: Evolutionary Computation

Question 1. (50 points)

A. Implement a binary Genetic Algorithm that uses fitness proportional selection, 1-point crossover, and bit-flip mutation to solve the problem in which the fitness is the number of 1s in the chromosome, i.e. the optimal solution is the chromosome where all genes are set to 1.

// code: qa01.php

B. Run the algorithm 10 times for each of the four following versions of the problem: l = 5; 10; 20; 50, where l is the length of the chromosomes. Vary the population size and mutation rate to obtain good results (fast solution).

// code: qa02.php
added algorithm to calculate the best average value for each generation for 10 runs

L = 5

Population	Mutation rate	Optimal solution found at (number of generation)
10	0.2	5
20	0.2	5
20	0.02	3
100	**0.02**	**1**

L = 10

Population	Mutation rate	Optimal solution found at (number of generation)
10	0.2	70
20	0.2	72
20	**0.02**	**19**
50	0.002	31

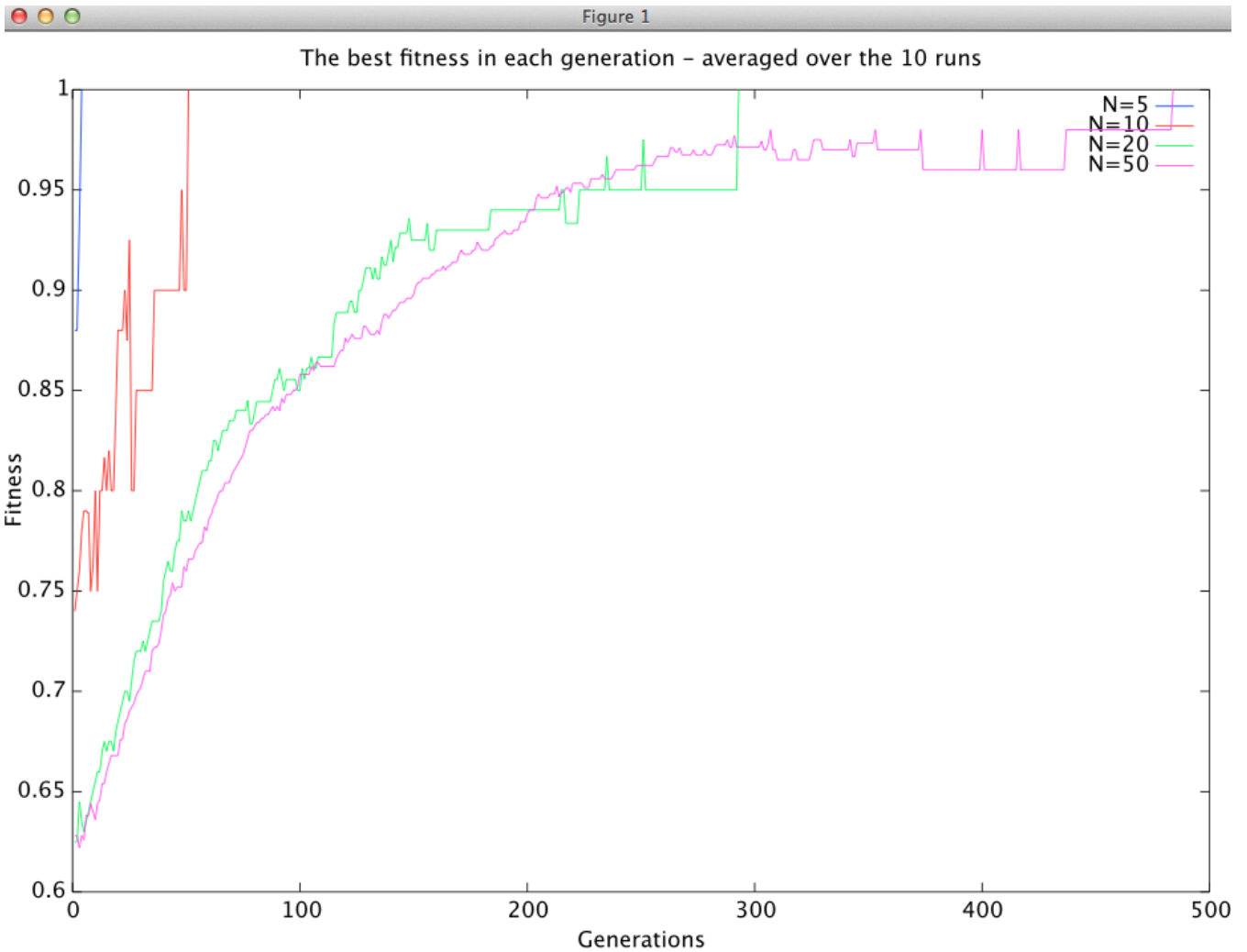
L = 20

Population	Mutation rate	Optimal solution found at (number of generation)
10	**0.002**	**309**
20	0.002	329
20	0.02	364
50	0.02	1732

L = 50

Population	Mutation rate	Optimal solution found at (number of generation)
10	**0.002**	**394**
20	0.002	640

C. Plot the best fitness in each generation (averaged over the 10 runs), for each of the four problems. There should be one graph with four curves, the x-axis being the generations, and the y-axis the average (best) fitness.



Question 2. (50 points)

A. Implement (1+)-ES, with one strategy parameter for each problem parameter, to optimize the known as the Rosenbrock function (http://en.wikipedia.org/wiki/Rosenbrock_function), where N is the number of dimensions, and 5 xi 10, i = 1; 2; : : : ; N.

// code es.php

Note that, please run this file in terminal to avoid timeout running via a web browser. You can simply run the code with the following command:

php es.php

B. Run the algorithm 10 times for each of the following four settings, N = 5; 10; 20; 50. Choose your own number of offspring

Each running time I have the following setting:

- Fix the number of iteration to 3,000.
- The number of offspring is different for each setting (number of dimensions). Instead of increasing the number of iteration for the bigger number of dimensions, I increase the number of offspring, which tends to increase the probability of having the minimum fitness values for each generation.

N-dimensions	Number of offspring	Minimum fitness of the parent
5	50	0.0026
10	100	1.2494
20	100	4.5579
50	100	6093.3128

C. Plot the average fitness (over the 10 runs) of the parent in each generation for each N. As with question 1B, there should be four curves here.

Note that y-axis, the fitness values have been represented as logarithm values. for example in Octave => plot(x,log(y)). The average fitness values were calculated by Octave.