Project – Part A
Introduction to Artificial Intelligence

By:

Rayhaan Shakeel (100425726)
Khalil Fazal (100425046)
Baldip Bhogal (100252234)

Date:

March 22, 2013

INTRODUCTION

Artificial Intelligence (AI) is a science to make machines do things that would require intelligence if done by humans. Genetic Algorithm and Differential Evolution are methods of evolutionary computation, which is used to simulate natural evolution by creating a population of individuals, evaluating their fitness, generating a new population through genetic operations, and repeatings this process a number of times. GA is an iterative process, and each iteration is called a *generation*. The entire set of generations is called a *run*. GA is terminated after a specified number of generations, and the best chromosome in the population is examined. If no satisfactory solution is found, GA is restarted. Differential Evolution algorithm creates solutions as vectors and uses operations such as vector addition, scalar multiplication and crossover to construct new solutions from existing ones. When new solution is created, it is compared to its parent. If new solution is better than its parent, it replaces the parent in the population, otherwise it is discarded [1]. DE uses actual real numbers, which achieves better results and outperforms GA.

OBJECTIVES

- Understand Genetic Algorithm (GA) and the Differential Evolution (DE)
- Implementing GA, DE, and the five benchmark functions
- Reporting the comparative results properly and analyzing the results
- Modeling and Solving a real-world problem using GA

RESULTS

a)

The eight queens problem is a problem of placing eight queens on an 8x8 chessboard without any of them attacking each other. To achieve this, each queen has to be assigned its own row, column, and diagonal. There are many ways of solving the 8 queens problem but only 12 distinct solutions. The problem can be solved by searching for a solution by placing the queen in different locations.

A possible method to solve the problem is Genetic Algorithm. Genetic Algorithm is a searching technique that copies the natural process of evolution. GA is widely used to solve problems that revolve around optimization or searching. Inspired by natural evolution, GA uses chromosomes and genes, often deployed as arrays and bits respectively, to replace orthodox variables. Techniques like crossover, mutation, and cloning are used on the chromosome to generate solutions.
A typical genetic algorithm requires a genetic representation of the solution as well as a fitness function to calculate its effectiveness. A usual representation is an array of bits, with each bit called a gene. Initially, GA requires a population of a fixed size to be randomly created. Each generation a fixed number of new chromosomes are created using the aforementioned crossover, mutation and cloning techniques. These characteristics of GA make it a popular improvement to traditional brute-force techniques.

Below are 6 possible solutions to the 8-queens problem.

```
Solution found!8090 Fitness is 0
Chromosome: 2
Chromosome: 5
Chromosome: 1
Chromosome: 6
Chromosome: 0
Chromosome: 3
Chromosome: 7
Chromosome: 4
Run Time: 0 seconds
| | |Q| | | | | |
| | | | | |Q| | |
| |Q| | | | | | |
| | | | | | |Q| |
|Q| | | | | | | |
| | | |Q| | | | |
| | | | | | | |Q|
| | | | |Q| | | |
```

```
Solution found!50868 Fitness is 0
Chromosome: 1
Chromosome: 4
Chromosome: 6
Chromosome: 3
Chromosome: 0
Chromosome: 7
Chromosome: 5
Chromosome: 2
Run Time: 4 seconds
| |Q| | | | | | |
| | | | |Q| | | |
| | | | | | |Q| |
| | | |Q| | | | |
|Q| | | | | | | |
| | | | | | | |Q|
| | | | | |Q| | |
| | |Q| | | | | |
```

```
Generation:520252 Fitness is 0
Chromosome: 6
Chromosome: 2
Chromosome: 0
Chromosome: 5
Chromosome: 7
Chromosome: 4
Chromosome: 1
Chromosome: 3
Run Time: 35 seconds
| | | | | | |Q| |
| | |Q| | | | | |
|Q| | | | | | | |
| | | | | |Q| | |
| | | | | | | |Q|
| | | | |Q| | | |
| |Q| | | | | | |
| | | |Q| | | | |
```

```
Generation:3359 Fitness is 0
Chromosome: 5
Chromosome: 0
Chromosome: 4
Chromosome: 1
Chromosome: 7
Chromosome: 2
Chromosome: 6
Chromosome: 3
Run Time: 0 seconds
| | | | | |Q| | |
|Q| | | | | | | |
| | | | |Q| | | |
| |Q| | | | | | |
| | | | | | | |Q|
| | |Q| | | | | |
| | | | | | |Q| |
| | | |Q| | | | |
```

```
Global fitness is 68
Average fitness is 8
Solution found!1 Fitness is 0
Chromosome: 5
Chromosome: 2
Chromosome: 4
Chromosome: 7
Chromosome: 0
Chromosome: 3
Chromosome: 1
Chromosome: 6
Run Time: 0 seconds
| | | | | |Q| | |
| | |Q| | | | | |
| | | | |Q| | | |
| | | | | | | |Q|
|Q| | | | | | | |
| | | |Q| | | | |
| |Q| | | | | | |
| | | | | | |Q| |
```

```
Solution found!71996 Fitness is 0
Chromosome: 3
Chromosome: 1
Chromosome: 6
Chromosome: 2
Chromosome: 5
Chromosome: 7
Chromosome: 4
Chromosome: 0
Run Time: 4 seconds
| | | |Q| | | | |
| |Q| | | | | | |
| | | | | | |Q| |
| | |Q| | | | | |
| | | | | |Q| | |
| | | | | | | |Q|
| | | | |Q| | | |
|Q| | | | | | | |
```

Figure 1: 6 Solutions to queen problem

Our implementation of the GA for the queens problem assigns each solution to a chromosome. Each queen's position is a gene in the chromosome, with all 8 queens or genes formed a full solution. Each queen is assigned a column, so only its rows are to be manipulated.

b)

Table 1: Solved Benchmark Functions

100.0%		
Benchmark Function	Mean Fitness Value	Standard Deviation of Fitness Values
F1	0.00	0.00
F2	0.00	0.00
F3	0.00	0.00
F4	0.16	0.79
F5	21.35	3.17

Linear Plots:

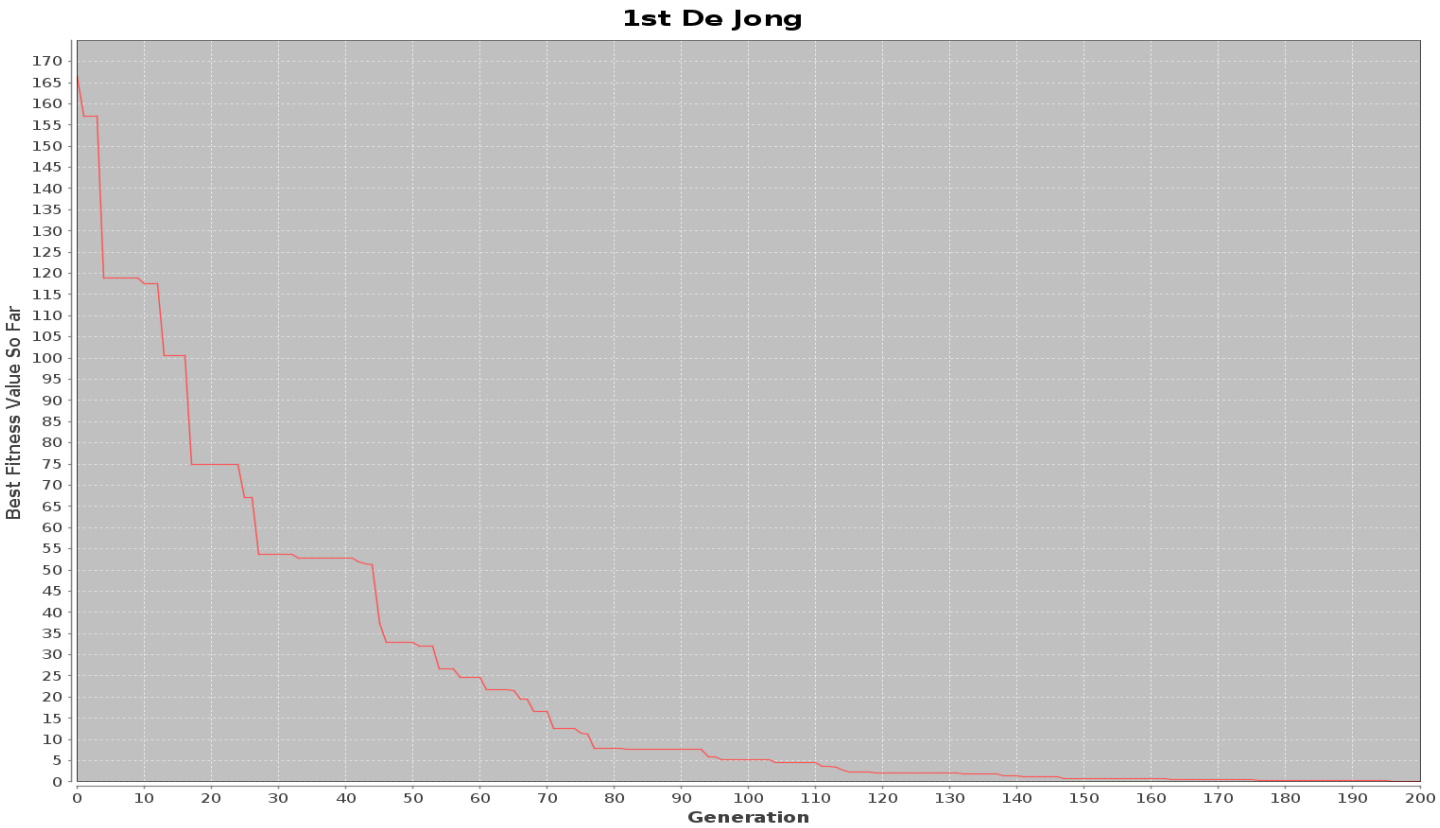

Figure 2: Linear plot for Function 1

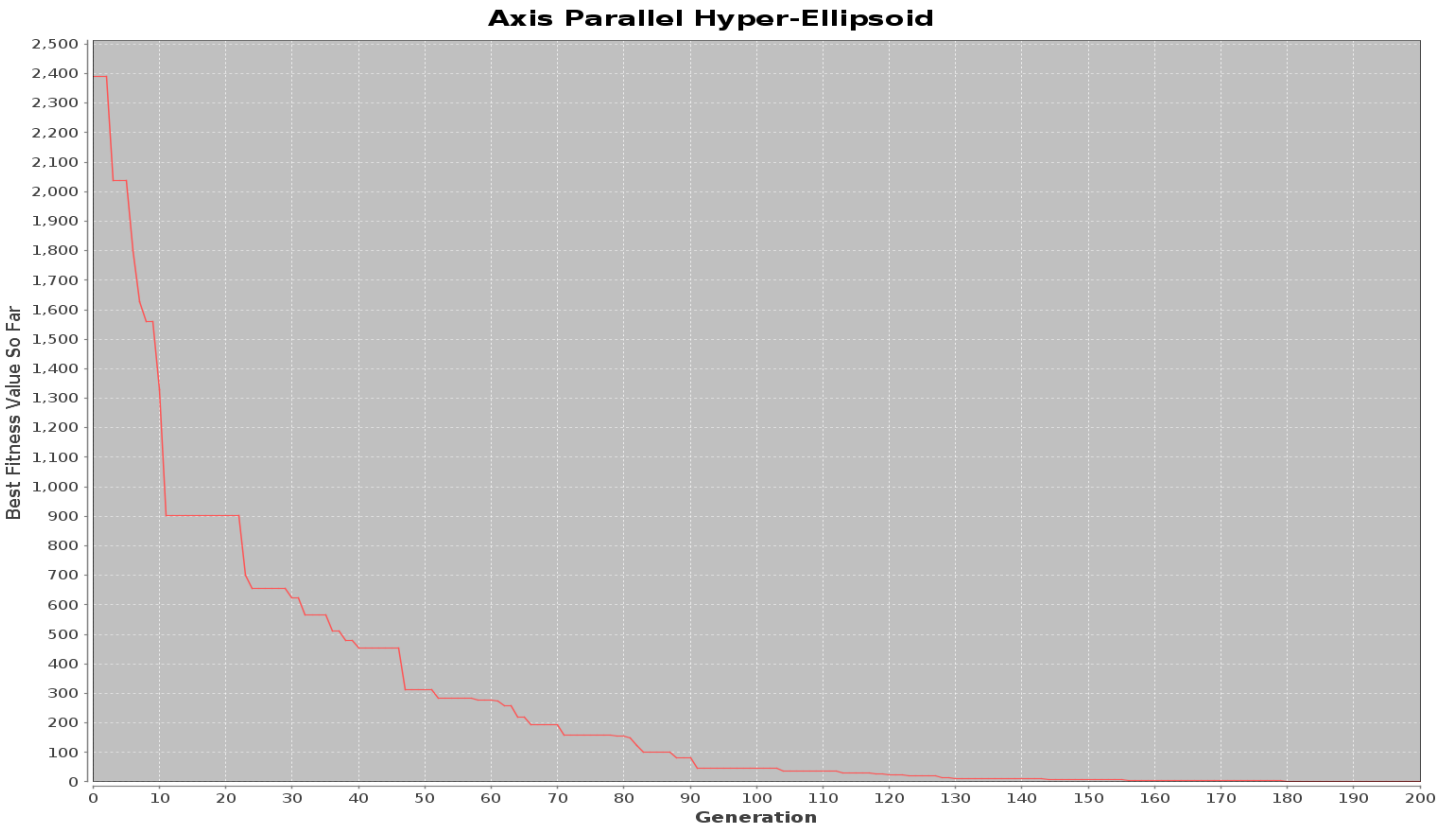

Figure 3: Linear plot for Function 2

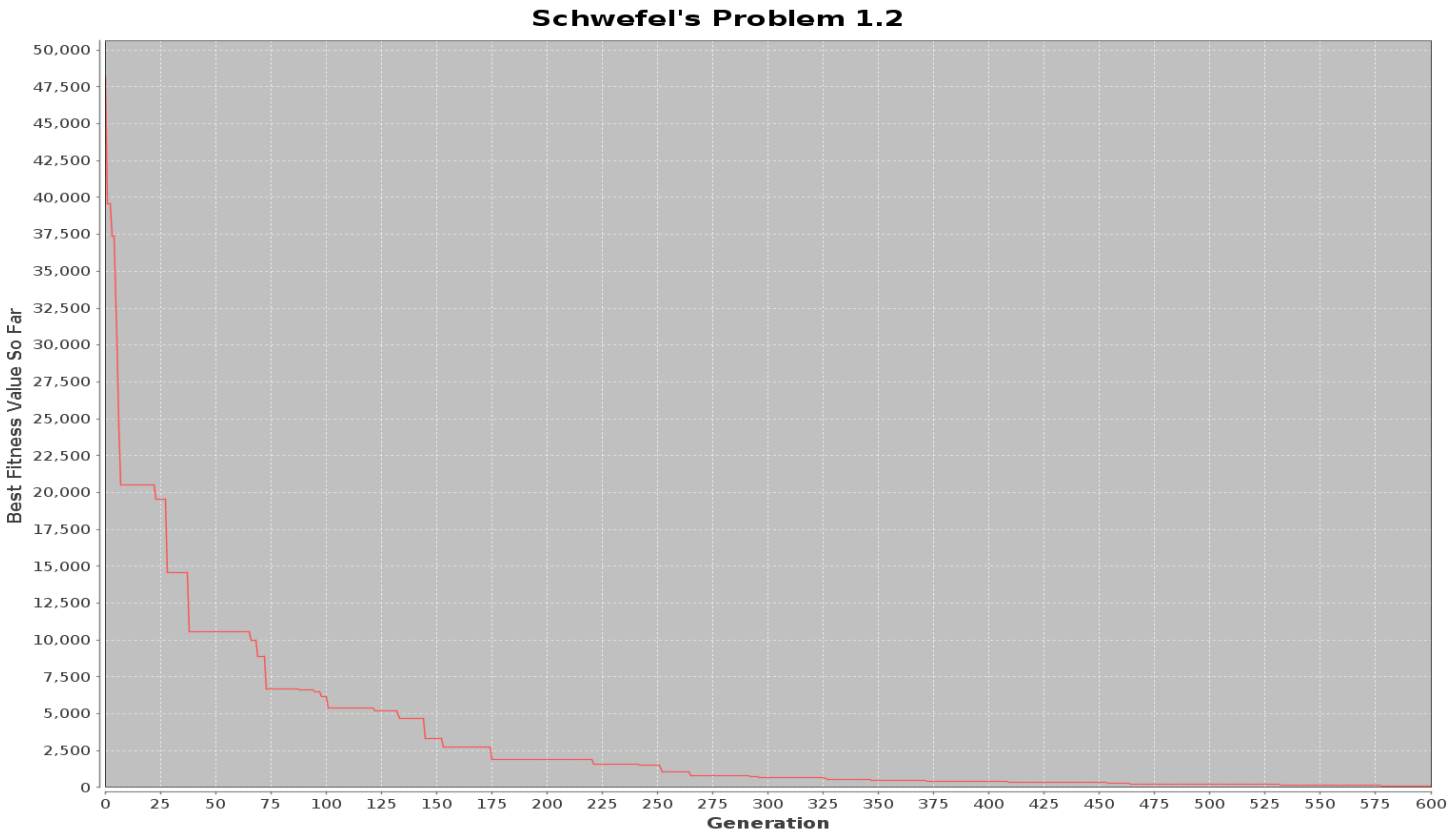

Figure 4: Linear plot for Function 3

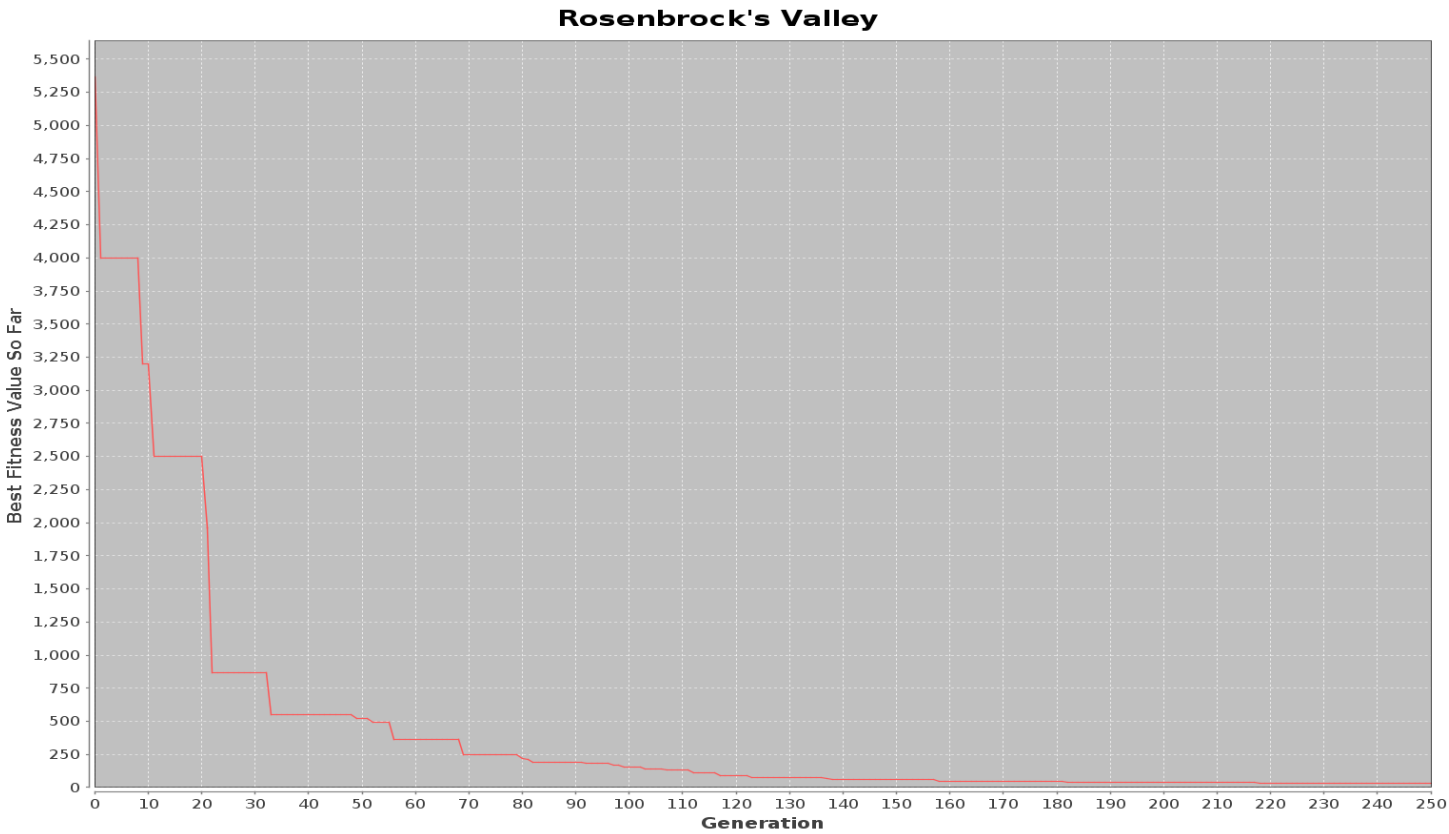

Figure 5: Linear plot for Function 4

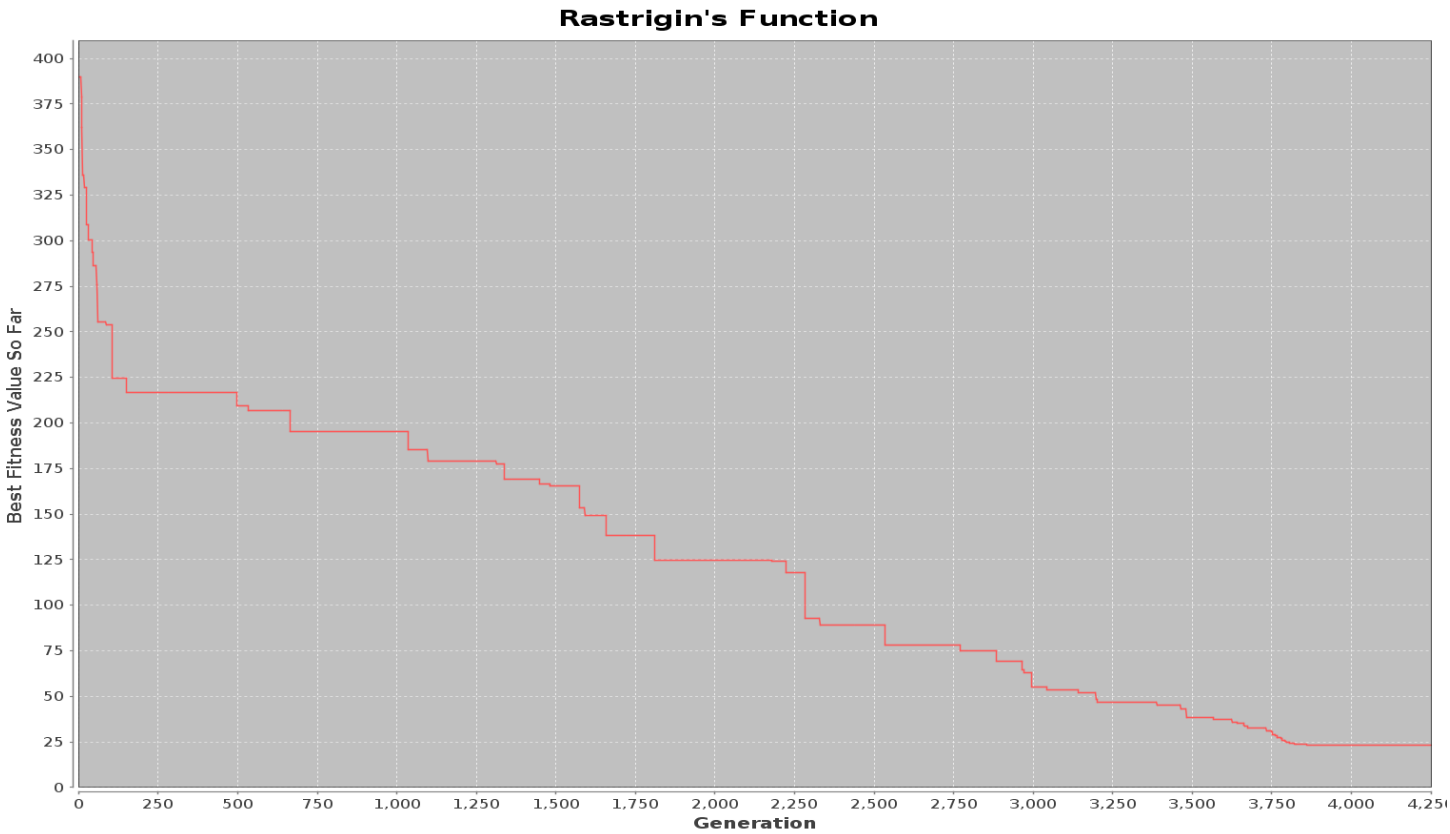

Figure 6: Linear plot for Function 5

Logarithmic Plots:

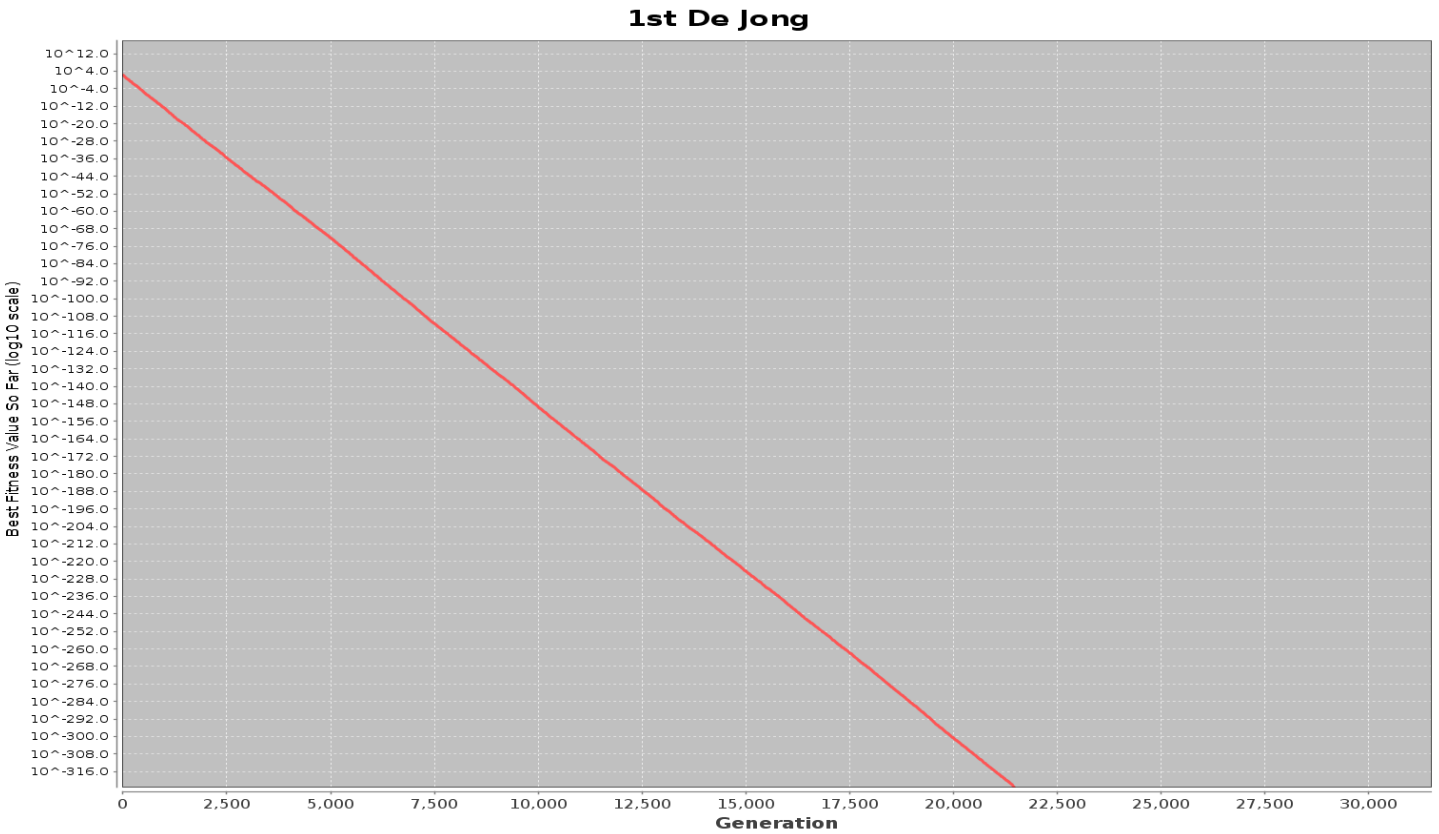

Figure 7: Logarithmic plot for Function 1

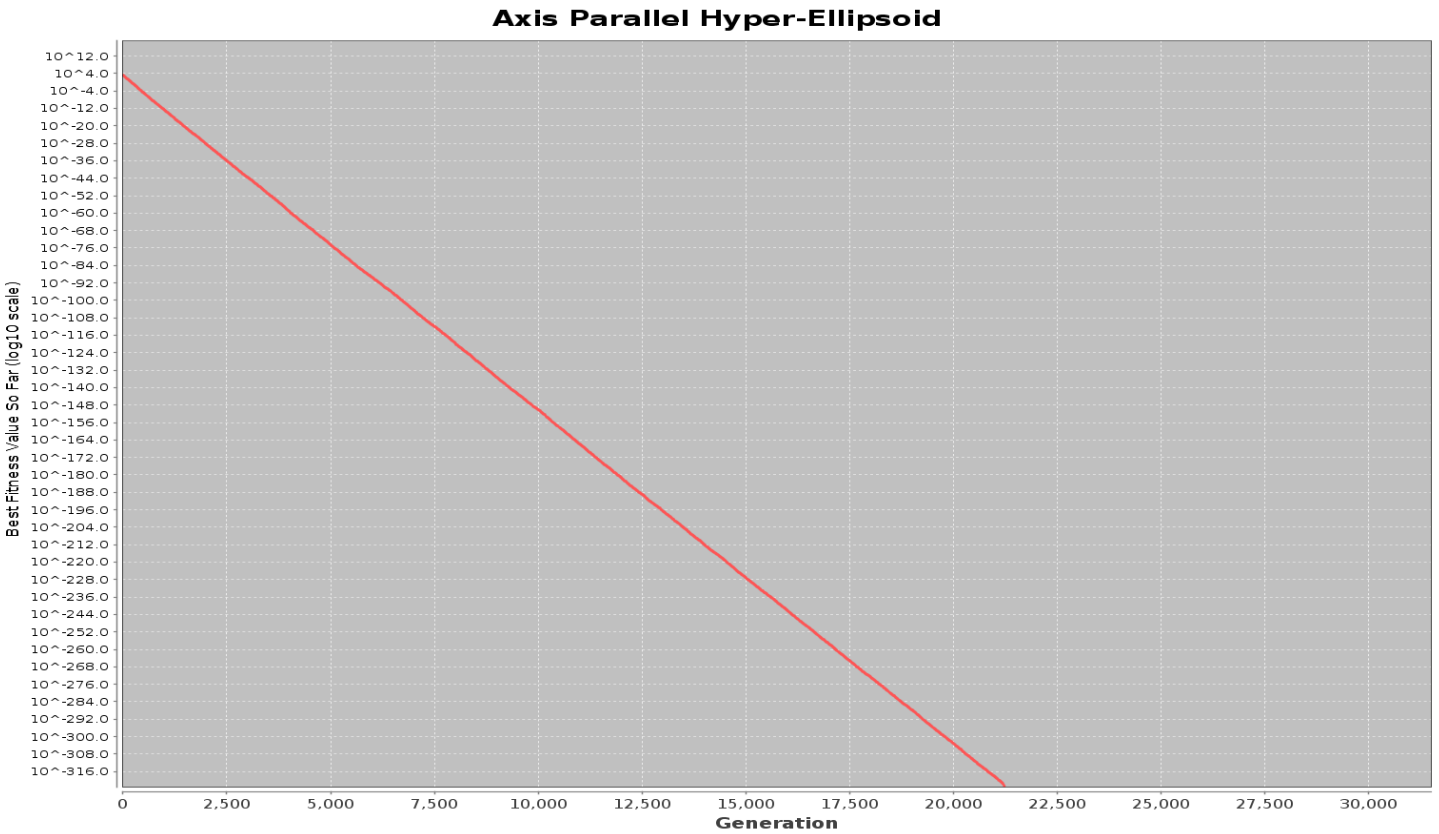

Figure 8: Logarithmic plot for Function 2

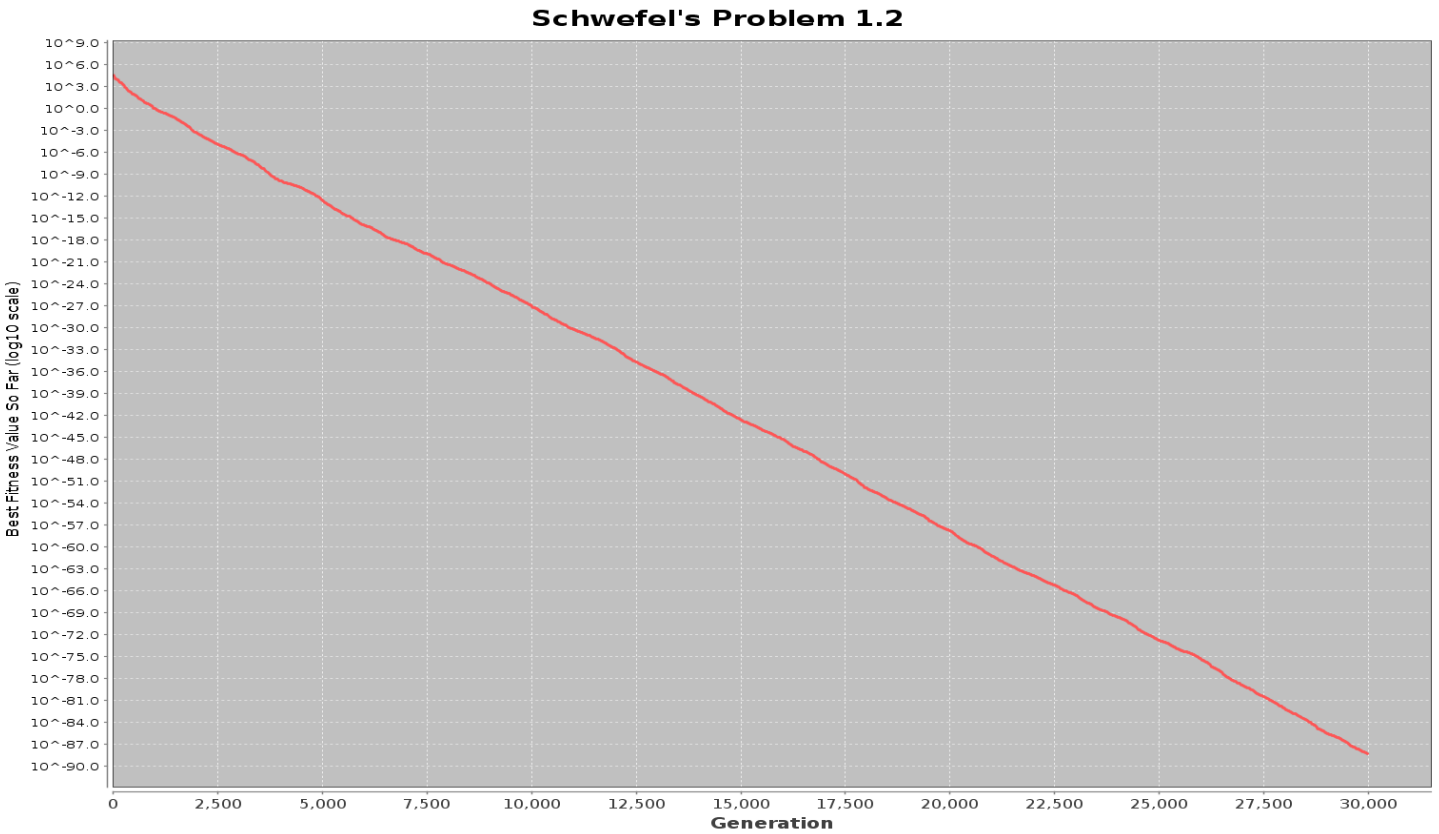

Figure 9: Logarithmic plot for Function 3

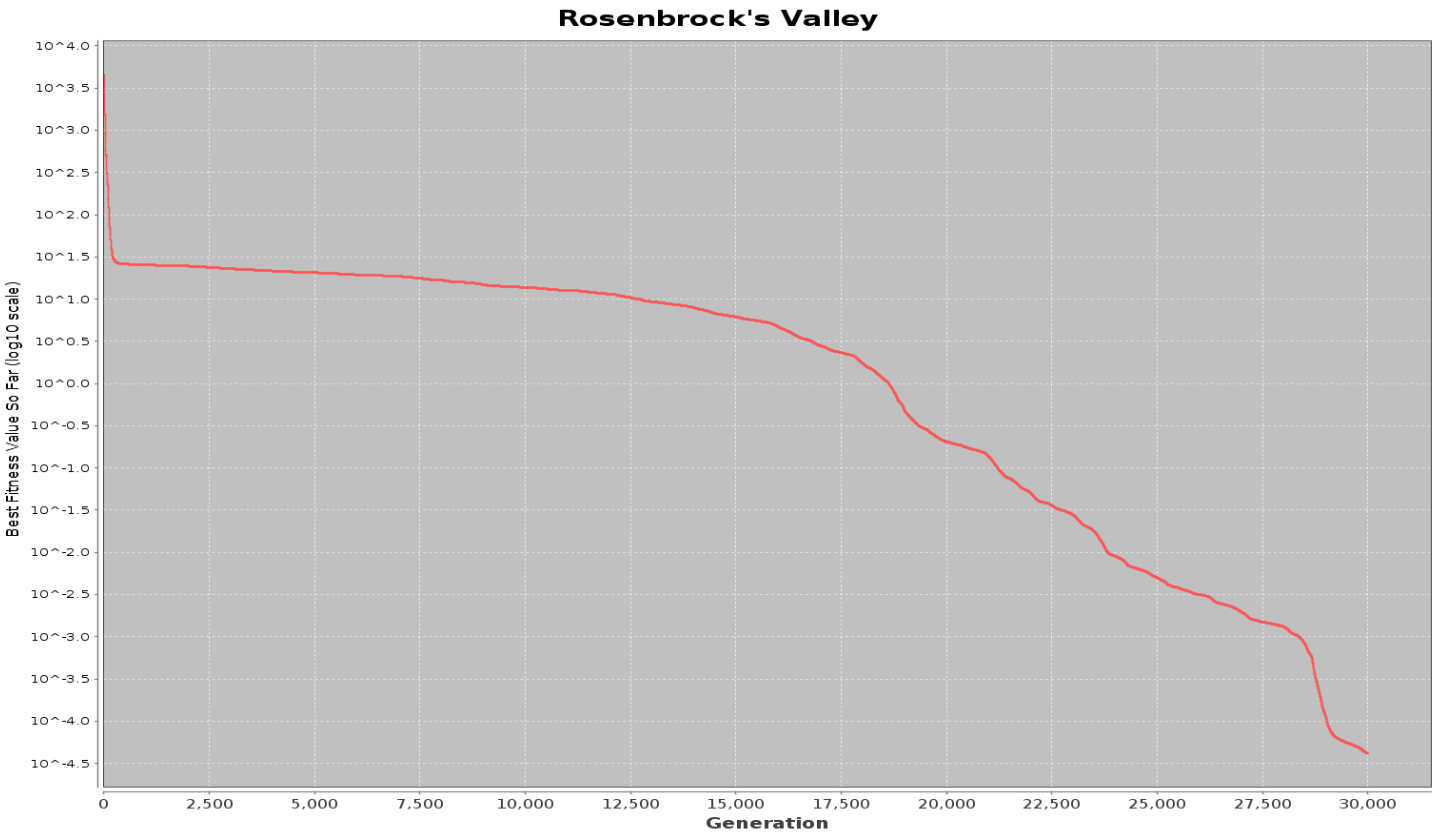

Figure 10: Logarithmic plot for Function 4

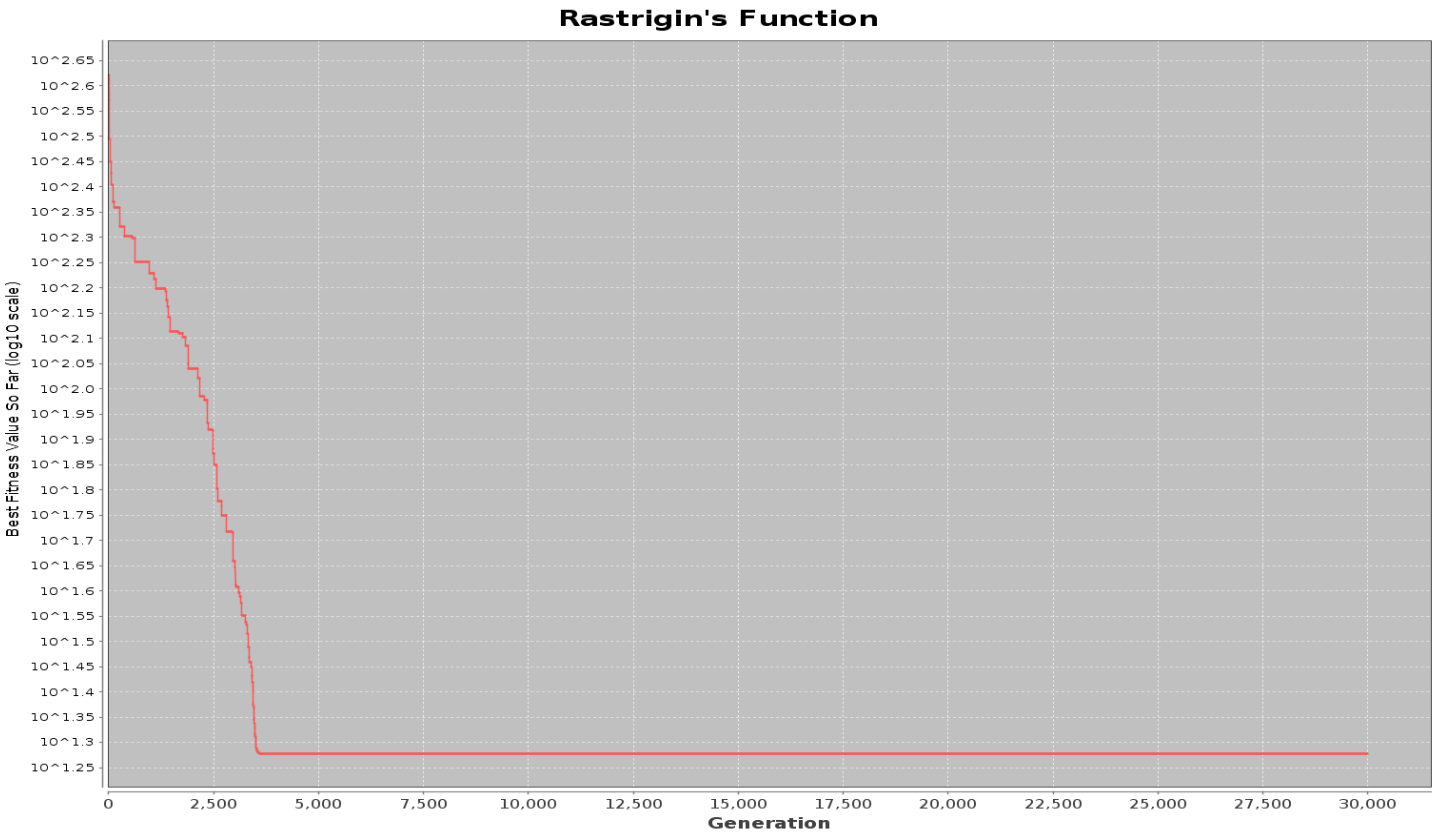

Figure 11: Logarithmic plot for Function 5

DISCUSSION

Function 1 is De Jong, the simplest test function, also known as the sphere model. It is described as continuous, convex and unimodal. Function 2 is axis parallel Hyper-Ellipsoid, similar to function 1, also known as weighted sphere model. It is also described as continuous, convex and unimodal [2].The logarithmic graphs can be seen to be linear for functions 1 (figure 7) and 2 (figure 8). The fitness values goes down to zero quicker for these two functions, which can be seen in the linear graphs. Function 3 is Schwefel, known to be deceptive in that global minimum is geometrically distant, over the parameter space, from the next best local minima. Thus, the algorithm is potentially prone to convergence in the wrong direction [2]. The logarithmic graph for this function looks almost linear and in the linear graph it takes more generations to get the fitness value to zero, compared to the first two functions. Function 4 is Rosenbrock's valley, also known as the Banana function, where the global optimum is inside a long, narrow, parabolic shaped flat valley. To find the convergence to the global optimum is difficult, therefore this is used to assess performance of optimization algorithms. Function 5 is Rastrigins, also based on function 1, has the addition of cosine modulation to produce multiple local minima. The test function is highly multimodal, but location of minima are regularly distributed. [2] The logarithmic graphs for functions 4 and 5 are not linear. It also takes function 5 more generations to get the fitness value to zero in the linear graph.

CONCLUSION

The Genetic Algorithm, Differential Evolution and the five benchmark functions have successfully been implemented.

It can be see that for DE, that functions 1,2 and 3 have the same fitness mean and standard deviation value. Function 5 has the highest mean fitness value. This is due to it being more difficult to minimize. It's mainly at that point, crossing over candidate solutions becomes useless since they are all similar. the optimal solution is [0 0 0 ... 0] but the candidates are stuck around that solution, like at [-1 0 1 0 -1 ... 0 1 -1]. One way the solution could improve during those dead zones is mutation.

We can perform a complexity analysis for the Differential Evolution Algorithm. The performance of the algorithm depends on the evaluation of the fitness function on each vector in the population. See the Javadoc for the repopulate() method in the Differential Evolution class. In conclusion, we determine that our implementation has a worse-case time complexity is O(n * T(n)) where n is the constant given in the project's instructions and T(n) is the worse-case time complexity for a fitness function. Given the five functions considered, T(n) is O(n^2). Therefore, our implementation in theory has a worse-case time complexity of O(n^3).

REFERENCES

[1] Tusar & Filipic. "Differential Evolution Versus Genetic Algorithms in Multiobjective Optimization". Depart of Intelligent System. Ljubljana, Slovenia. 2007.

[2] Hartmut Pohlheim. (1994). *GEATbx: Example Functions (single and multi-objective functions) 2 Parametric Optimization.* Available: http://www.geatbx.com/docu/fcnindex-01.html