TJHSST Computer Systems Lab
Senior Research Project
Research Paper Quarter 2
Applications of Genetic Algorithms
2008-2009

Mary Linnell

January 23, 2009

Abstract

The purpose of this project is to explore the applications of Genetic Algorithms, an evolutionary computation search technique, to find approximate solutions to optimization problems. This project will focus on computing the minimum point on a three dimensional graph. The goal is to find the minimum point without testing every single point on the graph, a very computational intensive process.

Keywords: genetic algorithms, machine learning, OpenGL, C

1 Introduction

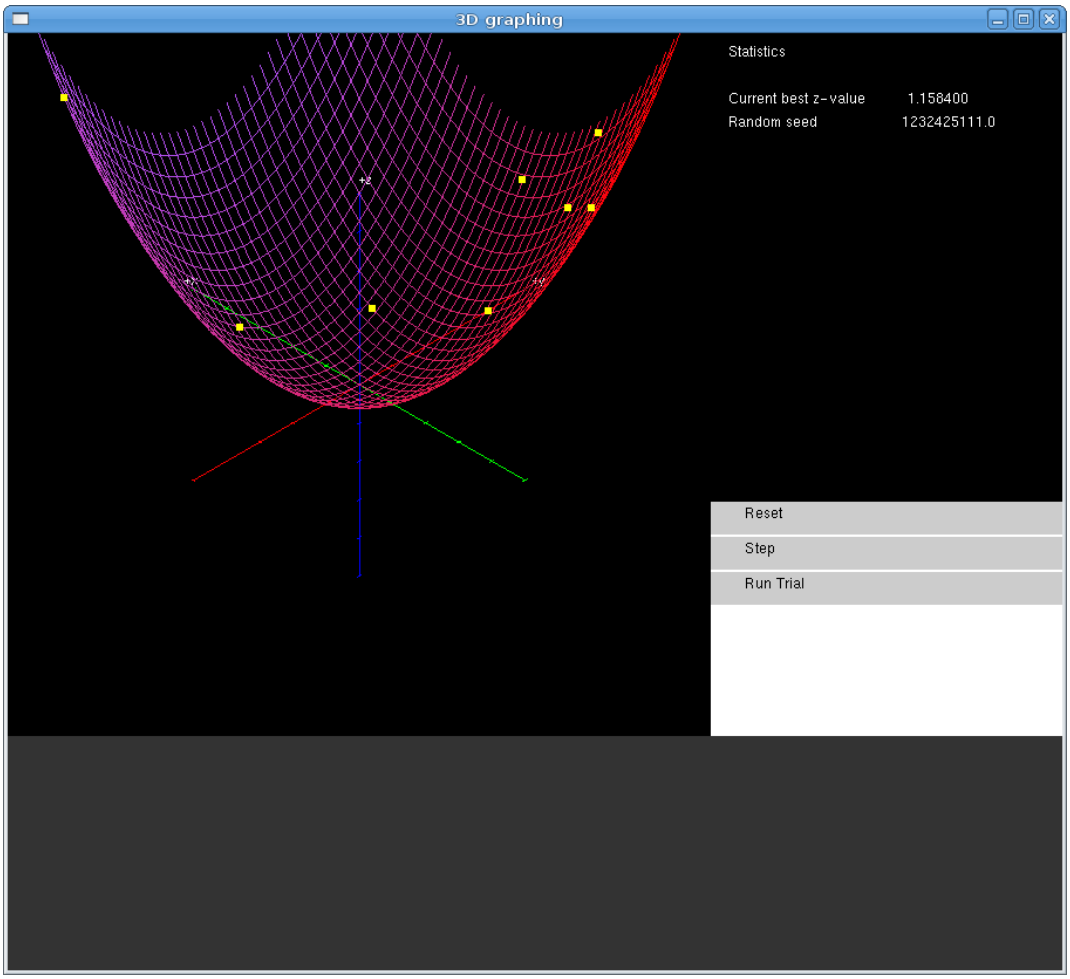
1.1 Scope of Study

The program will require a good knowledge of OpenGL 3D graphing, as well as programming in C for genetic algorithms. The research that will be required is how to optimize the genetic algorithm to achieve the best results when running multiple trials.

When run, the program displays wire-mesh graph, eight randomly chosen points on the graph, a results panel, a blank chart at the bottom, and several buttons to click on to control the simulation. The buttons include

a “Reset” to initialize all variables, a “Step” to cycle through the steps of the genetic algorithm, and a “Run Trial” to run multiple iterations of the genetic algorithm on the current set of points.

1.2 Screenshot



1.3 Expected results

I expect the results to approximate the exact result obtained using calculus. The results will be analyzed by seeding the random number generator differently and running different tests.

Currently, the results of a trial can be determined by analyzing the chart displayed at the bottom of the screen. This chart shows the progression of the best z-value (the fitness function) for each iteration of the genetic algorithm.

I hope to learn more about genetic algorithms and how to optimize the parameters to obtain the most accurate and precise results. I will also learn a lot relating to OpenGL and C.

2 Theory

2.1 Definition

A genetic algorithm is a theory used to compute approximate solutions in fewer iterations than other search techniques. It is based upon a structured evolutionary biology, with a "population" containing "individuals."

2.2 Set up

The population in this genetic algorithm is composed of points ("individuals").

2.3 Selection

The fitness function determines which points are the "best." In this case the fitness function will determine the lowest points by evaluating the known function and assigning a value (z value) to each point in the population.

A set number of the population will be killed off in each iteration. The fitness function determines which individuals of the population will be killed off, which will always be the members with the highest fitness function value.

2.4 Crossover

A crossover point was chosen within the population of points. The new individuals in the population were chosen based on an algorithm of duplicating, cutting, and slicing the existing population.

2.5 Mutation

The random mutation helps maintain genetic diversity in the population from one generation to the next. This factor can be important to avoid becoming trapped in a local minimum instead of the absolute minimum.

3 Procedures and Methodology

I am using C with OpenGL to write my program. I currently have the OpenGL component (3D graphing) completed and I am writing my genetic algorithm in C.

The graph of $z = x^2 + y^2$ appears on the screen in a wire-mesh of points. Eight randomly-generated yellow points appear on the screen. They consist of the population.

1. The step button is pressed. The four worst points (as determined by the fitness function) are highlighted in white and enlarged.

2. Step is pressed again. Those four points have been deleted, because this is the selection process.

3. When the step button is pressed the third time, new points appear (in this version of the program, they happen to be the points that were deleted, but in the future they will be new points chosen based upon a breeding algorithm).

4. The fourth time the step button is pressed, the points stay the same but become the permanent new population.

When the step button is continued to be pressed, the cycle repeats according to the steps above. Note that the "Run Trial" button automates pressing the step button by calling the step function multiple times and displaying the results.

4 Expected Results

I expect to see results that approximate the exact answer to the minimum point of the graph. I will show the genetic algorithm trial using a 3D graph - and I will analyze the results of many trials. A chart displays the current trial results at the bottom of the screen in the program.

References

[1] Ahrens, B. 2005. Genetic algorithm optimization of superresolution parameters. In *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation* (Washington DC, USA, June 25 - 29, 2005). H. Beyer, Ed. GECCO '05. ACM, New York, NY, 2083-2088. DOI= http://doi.acm.org/10.1145/1068009.1068354

[2] Coello, C. A. 2000. An updated survey of GA-based multiobjective optimization techniques. *ACM Comput. Surv.* 32, 2 (Jun. 2000), 109-143. DOI= http://doi.acm.org/10.1145/358923.358929

[3] Wainwright, R. L. 1994. A family of genetic algorithm packages on a workstation for solving combinatorial optimization problems. *SIGICE Bull.* 19, 3 (Feb. 1994), 30-36. DOI= http://doi.acm.org/10.1145/182063.182071

[4] Aranha, C. C. and Iba, H. 2008. A tree-based GA representation for the portfolio optimization problem. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation* (Atlanta, GA, USA, July 12 - 16, 2008). M. Keijzer, Ed. GECCO '08. ACM, New York, NY, 873-880. DOI= http://doi.acm.org/10.1145/1389095.1389267

[5] Yu, T. and Lin, W. 2008. Optimal sampling of genetic algorithms on polynomial regression. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation* (Atlanta, GA, USA, July 12 - 16, 2008). M. Keijzer, Ed. GECCO '08. ACM, New York, NY, 1089-1096. DOI= http://doi.acm.org/10.1145/1389095.1389294