Lab 10: Genetic Algorithms

Jorge García Ferreiro Pedro García Castillo

20/May/2016

Introduction

Objetive: Implement a Java program with 3 different algorithms: Brute Force, Random and Genetic algorithm. After the development process, we analyze the 3 algorithms with 3 different test cases.

The results should reflect the quality for each of the algorithms: Which one of the algorithms obtain the best result? What is the worse one?

Algorithms Description

Brute Force: This algorithm will always get the best possible solution as it tries with all the possible combinations for all 3 parameters. But at the same time it is also the most costly algorithm of all three.

Random: The random algorithm generates random solutions and keeps the first valid one. This makes it the fastest of all three algorithms because in many of the cases it will only generate a few solutions until it gets the first valid one.

Genetic algorithm: The genetic algorithm generates a random initial population and from it tries to improve the population as much as possible in terms of radius size but also taking into account that the circle is a valid solution for the problem (doesn't intersect other circles neither it falls out of the solution space).

This kind of algorithms needs many generations to obtain good results (very close to the brute force) because it evolves and improves during time. So, we use the following parameters in our algorithm:

Number of generations = 200 Genetic Population Size = 100

Brute Force Vs. Random Vs. Genetic algorithm

Test 1: No initial circles.

In this test case where We don't have any circle in the initial board. We want to know if all the algorithms can find the best possible solution. In this case, the best solution is 512 (the maximum possible circle with a board of 1024x1024).

The random Algorithm in this outcome is good, but does not reflect the real behaviour. In other executions we also obtain a low result. The brute force algorithm finds the best solution (512) and the Genetic is very close.

Why genetic obtain such a great result? Because we use a very huge Population Size (100) and the number of iterations is quite big. So the algorithm evolves and improves the result over time.

Algorithm name	X	Y	Radius
Random Algorithm	660	801	170
Brute force Algorithm	511	511	512
Genetic Algorithm	511	511	511

Test 2: One Huge Circle.

So what could happen if we create an initial big circle in the middle of the screen?

Once again the random Algorithm obtains the worse solution because only generates solutions until a valid one is found. So in this test case where the place in which a random circle can be placed is very limited (sometimes can find a better solution that in this outcome, but in general is quite bad).

The rest of the algorithms are better than the random. The brute force obtains the best solution because it tests all the possible circles. And the Genetic Algorithm gets a very good score because of our parameters (large individuals per generation and many generations).

Algorithm name	X	Y	Radius
Random Algorithm	956	125	19
Brute force Algorithm	889	889	135
Genetic Algorithm	130	138	127

Test 3: 9 Circles that almost fill completely the board.

We want to test our algorithm in the most extreme scenarios. So what happen if the put 9 circles with the same radius uniformly distributed in all the board? So in this example, the random is really bad. Just obtain a circle with radius 2. As usually, the brute force found the best solution (89), because it tests all the circles. And also the Genetic Algorithm is so closed to the Brute force one, but does not found the best solution.

Algorithm name	X	Y	Radius
Random Algorithm	32	119	2
Brute force Algorithm	88	88	89
Genetic Algorithm	90	89	87