Genetic algorithms

Vranceanu Andi-Costin

January 6, 2020

1 Introduction

Deterministic algorithm currently cannot solve hard problems like finding the global minimum of a function. Fortunately the is other tool we can use to get better results for this kind of problem. Genetic algorithms are a powerful tool in resolving this kind of problems, as well as heuristic methods like Hill Climbing and Simulated Annealing.

2 Algorithms

The algorithms used in this experiment are: Hill Climbing (best improvement), Simulated Annealing and a Genetic algorithm. All solutions are represented in bitstrings for all algorithm.
Hill Climbing (best improvement) will choose a random solution, then chose the best neighbor solution that minimises our function.
Simulated Annealing will also choose a random solution, but rather than always choosing a better solution than the current one, we have a temperature T that will determine the probabilty of choosing a better or worse solution. This will allow for the solution to get out of the local minima at the expense of time.
The Genetic aglorithm will initiate the each individual of it's population with a solution with a random value than repeat 3 fundamental part: mutation, crossover and selection. In the mutation part, each bit of the solution has a liitle chance to modify itself (from 1 to 0 or from 0 to 1). The crossover will combine genes to make children for a new population. The selection part will be a RouleteWhell where individuals will be select for the new generation.

3 Functions

3.1 Rastrigin Function

$$F(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10cos(2\pi x_i))\ , x_i \in [-5.12, 5.12]$$

3.2 Rosenbrock Function

$$F(x,y) = \sum_{i=1}^{n} [b(x_{i+1} - x_i^2)^2 + (a - x_i)^2],\ a = 1,\ b = 100,)\ x_i \in [-5, 10]$$

3.3 Griewangk Function

$$f(x,y) = 1 + \frac{1}{4000} \sum_{i=1}^{n} x_i - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}}))\ x_i \in [-600, 600]$$

3.4 Six-hump camel back function

$$F(x) = (4 - 2.1x_1^2 + (x_1^4)/3)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2)x_i \in [-3, 3]$$

4 Results

The genetic algorithm was runned 30 times for each dimension and functions and 3000 for each instance of the genetic algorithm. The probability chance of a bit to mutate is 1/100 and the probability of crossover is 30/100.

4.1 Rastrigin Function

	Genetic Alg			Hill Climbing BI		
Dimensions	2	5	20	2	5	20
Average	1.3388	11.4613	106.793	0.13283	2.04692	21.4074
Min	0.0002	4.5609	51.4402	0	0	9.9284
Max	3.1041	25.1163	182.291	1.00002	3.2258	29.144
Standard deviation	0.8643	5.6177	28.6999	0.3386	0.7048	3.8814

	Simulated Annealing		
Dimensions	2	5	20
Average	0.0780	3.8924	33.7032
Min	0	1.1178	22.8667
Max	0.5768	7.6524	42.8064
Standard deviation	0.1190	1.6734	6.1320

4.2 Griewangk's function

	Genetic Alg			Hill Climbing BI		
Dimensions	2	5	20	2	5	20
Average	0.1166	4.2155	77.3859	0	0.0060	0.2403
Min	0.0061	1.1009	19.2534	0	0	0
Max	0.4934	16.8065	188.286	0	0.02713	0.4273
Standard deviation	0.1067	3.3915	32.2593	0	0.00816	0.1146

	Simulated Annealing		
Dimensions	2	5	20
Average	0.06038	1.0196	7.6032
Min	0.00865	0.6241	4.1121
Max	0.1181	1.3938	10.5641
Standard deviation	0.03276	0.1887	2.0098

4.3 Rosenbrock's function

	Genetic Alg			Hill Climbing BI		
Dimensions	2	5	20	2	5	20
Average	0.03716	19.1907	514.765	0	0.0057	0.3136
Min	0	1.1656	199.05	0	0	0.1701
Max	0.2290	76.9218	1275.64	0	0.0452	0.4182
Standard deviation	0.0471	19.2837	226.622	0	0.00831	0.0692

	Simulated Annealing		
Dimensions	2	5	20
Average	0.00046	2.5718	37.5571
Min	0	0.04742	21.7116
Max	0.00781	4.90352	93.1833
Standard deviation	0.00148	1.63704	23.5849

4.4 Six camel hump

	Genetic Alg	Hill Climbing BI	Simulated Annealing
Dimensions	2	2	2
Average	-0.9791	-1.03163	-1.02039
Min	-1.00051	-1.03163	-1.03153
Max	-0.8699	-1.03162	-0.978038
Standard deviation	0.02751	0	0.01234

The results show us that Genetic Algorithms are reliable way to get good results with hard problems. In this, experiment it managed to even get to the function minima several time, even though not as many times as Hill Climbing or Simulated Annealing. This is mainly because of the mutations and crossover modifying the solutions so much. Thanks to this modification it's easier for the Genetic Algorithm to escape local minima, but it's a bit harder to find the global minima. Also, the result of algorithm improved greatly upon using elitism. Each generation the 10 best individuals will be kept in the population. This gave the mutation a lower change of destroying good genes, and it improved the solutions of the algorithm by a lot.

5 Conclusion

The experiment showed us the differences between Genetic Algorithm and other heuristic approaches like Hill Climbing and Simulated Annealing. The Hill Climbing had the best results for this functions, but it would struggle a lot more with functions with a lot more local minima. The Simulated Annealing and Genetic Algorithm have different ways in which the can escape this local minimas. The Simulated Annealing has a higher chance to choose a worse solution at the beginnig, but in the end it will almost always choose better results than the current one, making it easier to aproach the global minima. On the other hand, the Genetic Algorithm has a consistent and diverse population that can always escape local minima but it's a lot harder to get to the global one, because of all the mutations.

References

[1] Functions informations
`http://www.geatbx.com/docu/fcnindex-01.html`