"ALEXANDRU IOAN CUZA" UNIVERSITY OF IAȘI
FACULTY OF COMPUTER SCIENCE

Genetic Algorithms
Boolean satisfiability problem

Diana-Isabela **Crainic** & Ioan **Sava**

January 2020

1 INTRODUCTION

Satisfiability problem (SAT), which is the cornerstone of computational complexity theory and a fundamental problem in many areas, was the first NP-Complete problem.

In logic and computer science, the Boolean satisfiability problem is the problem of determining if there exists an interpretation that satisfies a given Boolean formula. In other words, it asks whether the variables of a given Boolean formula can be consistently replaced by the values TRUE or FALSE in such a way that the formula evaluates to TRUE. If this is the case, the formula is called satisfiable.[1]

In the theoretical research, SAT can be used for combinatorial optimization, real-time scheduling, statistical physics and other basic issues. In industrial applications, SAT can also solve many problems, such as circuit verification, chip design, task scheduling, and other problems. SAT is of great use in electronic design automation, equivalence checking, bounded model checking, and so on.[2]

The purpose of this paper is to draw a comparison between Heuristic Method and Genetic Algorithm for solving SAT problems and to analyze which one is the best option.

This article is organized as follows. Section 2 introduces the basic knowledge of Satisfiability Problem, Heuristic Method and Genetic Algorithm for solving SAT problems. Section 3 describes the methods used and a comparison between them. In Section 4, the proposed approach is evaluated using several benchmark databases. In section 5 are presented the interpretations. Finally, section 6 discusses our main findings and suggests directions for future work.

2 MOTIVATION

Any non-Deterministic Polynomial problem can be solved in the polynomial time to the SAT, so an algorithm that can solve the SAT problem efficiently is sure to solve all NP problems. In practice, there are a lot of NP problems, so it is of great significance to find an efficient and fast SAT algorithm and apply it to engineering practice, which can improve productivity and promote social development.

Due to their theoretical and practical relevance, SAT and many exact and heuristic algorithms have been introduced. Exact algorithms give a definite answer to any problem instance (be it satisfiable or unsatisfiable) but have an exponential worst-case complexity, unless P = NP. Heuristic algorithms can find solutions to satisfiable instances quickly, but they are not guaranteed to give a definite answer to all problem instances.

Genetic Algorithm, first developed by Holland, is a global optimization probability search algorithm, which simulates the process of genetic evolution in the biological world. Compared with other optimization algorithms, it has many advantages, such as multi point search, parallel computing, scalability, robustness, and so on, so it has been extensively developed in theory and application.[2]

3 METHODS

3.1 SOLUTION REPRESENTATION

In this paper, the true and false assignments of each variable in formula F is mapped into a coding of candidate solutions, where length of a candidate is the total number of variables.

3.2 FITNESS FUNCTION

The fitness function is used to evaluate the individual, which is the only basis for the optimization process. The fitness of individuals reflects the adaptability of individuals to the environment in order to survive.

In this paper, for a F formula and for a given set of true and false assignments v_i, the fitness function is given by the number of clauses in F which are satisfied. That is, the fitness function expression is described as:

$$fitness(y_i) = \sum_{j=1}^{m} v_i(c_j)$$

where y_i represents the i individual, and regards the chromosome coding corresponding to the individual y_i as the assignment v_i of the formula. When all the clauses are satisfied, the value of fitness is equal to the number of clauses(m), then the optimal solution is found, and it means the maximum fitness is equal to the number of clauses(m).

3.3 HEURISTIC METHOD

The simulated annealing algorithm is an optimization method which mimics the slow cooling of metals, which is characterized by a progressive reduction in the atomic movements that reduce the density of lattice defects until a lowest-energy state is reached. In a similar way, at each virtual annealing temperature, the simulated annealing algorithm generates a new potential solution (or neighbour of the current state) to the problem considered by altering the current state, according to a predefined criterion.[3]

This algorithm is different from the hill climbing algorithm in that at some point there is the temperature-dependent probability of visiting solutions that are weaker than the current one, in order to escape from local optima. This probability is given by:

$$P = \begin{cases} 1 & \text{if } fitness(cC) \leq fitness(cN) \\ e^{\frac{-|fitness(cN) - fitness(cC)|}{T}} & \text{if } fitness(cC) > fitness(cN) \end{cases} \tag{3.1}$$

where cC is the current candidate, cN is the current neighbor and T is the current temperature. The annealing temperature followed a predifined cooling rate c = 0.9.

At the beginning of the algorithm, which started from a randomly generated candidate, the higher temperature and higher probability of acceptance of new solutions allowed to explore a wide region of the search space, thus escaping from a minimal location; however, as the temperature was reduced, the probability of acceptance of unfavourable solutions was reduced.

3.4 Genetic algorithm

The proposed genetic algorithm uses classical genetic operators (uniform mutation, one-cut-point crossover, roulette wheel selection) and also a series of optimizations to improve the results provided by the algorithm:

Hypermutation
Hypermutation consists of increasing the probability of mutation when the solution quality drops (e.g. 0.3 - 0.5). This method is a simple but still an effective way to reset the population and also guarantee the diversity of the population.[4]

Elitism
A practical variant of the general process of constructing a new population is to allow the best organism(s) from the current generation to carry over to the next, unaltered. This strategy is known as elitist selection and guar-

antees that the solution quality obtained by the genetic algorithm will not decrease from one generation to the next.

Hill climbing
After the genetic algorithm is finished and the best candidate of the last generation is returned, a hill climbing will be performed on it in order to find an even better solution.

4 Results

The proposed optimization algorithms are tested on a set of 5 inputs of forced satisfiable SAT formulas, with the problem size growing from 450 Boolean variables to 1150 Boolean variables. For each input, 20 runs were performed.

450 Boolean variables / 19084 clauses					
Algorithm	Minimum fitness	Maximum fitness	Mean	Median	Standard Deviation
SA	15069	17946	16202.9	16182	1019.1
GA	18488	18931	18708.4	18703.5	97.13821

Table 4.1: SAT benchmark 450 Boolean variables / 19084 clauses

595 Boolean variables / 29707 clauses					
Algorithm	Minimum fitness	Maximum fitness	Mean	Median	Standard Deviation
SA	23267	29140	27478.93	28920	2269.727
GA	28256	29098	28799.95	28819	193.8218

Table 4.2: SAT benchmark 595 Boolean variables / 29707 clauses

760 Boolean variables / 43780 clauses					
Algorithm	Minimum fitness	Maximum fitness	Mean	Median	Standard Deviation
SA	34402	40244	38165.5	37996.5	1390.068
GA	41482	42336	41917.25	41856.5	219.2933

Table 4.3: SAT benchmark 760 Boolean variables / 43780 clauses

945 Boolean variables / 61855 clauses					
Algorithm	Minimum fitness	Maximum fitness	Mean	Median	Standard Deviation
SA	47978	51883	49515.2	49156	1324.81
GA	57762	58842	58323.75	58357	274.188

Table 4.4: SAT benchmark 945 Boolean variables / 61855 clauses

1150 Boolean variables / 84508 clauses					
Algorithm	Minimum fitness	Maximum fitness	Mean	Median	Standard Deviation
SA	64704	67140	65989.8	66067	604.7186
GA	77731	79667	78733.1	78642	545.1194

Table 4.5: SAT benchmark 1150 Boolean variables / 84508 clauses

5 INTERPRETATION

The experiments indicate that the highest success rate in satisfying the clauses was for Genetic Algorithm in comparison with Simulated Annealing.

In the case of heuristic method, more precisely simulated annealing, the number of satisfied clauses is smaller in each scenario than using genetic algorithm. The results were pleasant for instances with lesser number of variables, however with large test cases, the success ratio made it to about 75%.

In the table in which 595 Boolean variables/ 29707 clauses (Table 2) were tested using simulated annealing, the minimum value was 23267 and the maximum 29140, being a considerable difference, therefore the results were very varied.

The genetic algorithm has managed to satisfy a fairly good number of clauses when it has come to both a small number and for a large number of Boolean variables and clauses, the success rate being over 90% in all cases. These outcomes are due to a series of optimization to improve the results by the algorithm like hypermutation, elitism and performing the hill climbing algorithm will help finding an even better solution.

6 CONCLUSIONS

Aiming at the SAT problem, an improved genetic algorithm is proposed compared to the simulated annealing algorithm.

Genetic Algorithms are a heuristic solution search technique inspired by natural evolution. They are particularly suited to problems where traditional optimisation techniques break down or because the search becomes computationally intractable. An improved genetic algorithm is proposed to avoid premature convergence and guarantee the diversity of the population.

In simulated annealing the algorithm runs by starting with a given temperature and gradually cool it as the iterations are running, then the probability of accepting a bad solution is proportional to the current temperature.

REFERENCES

[1] Boolean satisfiability problem
Wikipedia
https://en.wikipedia.org/wiki/Boolean_satisfiability_problem

[2] An Improved Adaptive Genetic Algorithm for Solving 3-SAT
Huimin Fu, Yang Xu, Guanfeng Wu, Hairui Jia, Wuyang Zhang, Rong Hu
https://www.atlantis-press.com/journals/ijcis/25888772/view

[3] Science direct
Composite materials – modelling, prediction and optimization
https://www.sciencedirect.com/topics/engineering/simulated-annealing-algorithm

[4] An Investigation into the Use of Hypermutation as an Adaptive Operator in Genetic Algorithms Having Continuous, Time-Dependent Nonstationary Environments
Helen G. Cobb
December 11, 1990
https://apps.dtic.mil/dtic/tr/fulltext/u2/a229159.pdf