Evolutionary Computation

Pages 1–3
2025

Solving the MAXSAT Problem with an Improved Binary Genetic Algorithm and Hyperparameter Optimization

Lorenzo Cardenas Cardenas

Lorenzo Cardenas Cardenas / ID: 2839898

MSc. Artificial Intelligence and Machine Learning

27/03/2025

Keywords: Hyperparameter Optimization , Stochastic Ranking , Binary Genetic Algorithm (BGA), MAXSAT Problem.

1 INTRODUCTION

The Maximum Satisfiability (MAXSAT) problem is a fundamental combinatorial optimization problem where the objective is to assign Boolean values to variables to maximize the number of satisfied clauses. Each clause is a disjunction of literals, and solving MAXSAT has practical applications in fields such as artificial intelligence, verification, and operations research.

We solve the MAXSAT problem using an Improved Binary Genetic Algorithm (IBGA), an evolutionary computation method inspired by natural selection. IBGA maintains a population of candidate solutions, evolving them over generations through selection, crossover, mutation, and stochastic ranking. This iterative process balances exploration and exploitation while incorporating Lagrangian relaxation for better constraint handling, making it suitable for tackling the NP-hard MAXSAT problem.

2 THEORY

2.1 Improved Binary Genetic Algorithm (IBGA) for MAXSAT

In the IBGA, solutions are encoded as binary strings, where each bit represents the assignment of a variable (0 = False, 1 = True). The fitness of an individual is defined as the number of clauses satisfied by the corresponding assignment.

Evolutionary Cycle:

1. **Initialization**: Heuristically generate a random population of binary assignments.
2. **Fitness Evaluation**: Calculate satisfied clauses for each individual.
3. **Stochastic Ranking / Lagrangian Ranking**: Sort individuals based on constraint satisfaction and fitness.
4. **Selection**: Select pairs of parents from the ranked population.
5. **Crossover**: Apply one-point crossover to produce offspring.
6. **Mutation**: Perform bit-flip mutation to maintain diversity.
7. **Survivor Selection with Elitism**: Preserve the best individuals (elite size configurable) and fill the rest with offspring.
8. **Repeat** until the stopping condition (generations or time budget) is met.

This process allows the algorithm to navigate the search space efficiently, balancing local exploitation with global exploration.

2.2 Pseudo code

```
BEGIN Improved BGA
INPUT: Dataset with clauses
CONFIGURE: Population size, max generations, crossover, mutation, elite size, time budget, ranking method

INITIALIZE:
   Generate initial population using Pseudo-Random Initialization
   Compute fitness and constraint violations separately
   Apply Stochastic Ranking or Lagrangian Ranking for constraint handling
   Sort population using ranking method

FOR each generation DO:
   SELECT parents using matching selection
   APPLY crossover and mutation to generate offspring
   PERFORM optional Lagrangian penalty adjustment if enabled
   COMPUTE fitness and constraints for offspring

   APPLY Stochastic Ranking or Lagrangian Ranking for selection
   REPLACE worst individuals with best solutions (elitism based on elite_size)

RETURN Best Individual and Best Fitness
END
```

Fig. 1. BGA Pseudo code

3 METHODS

The Lagrangian relaxation in our algorithm serves to balance the search between maximizing the number of satisfied clauses and minimizing constraint violations. Instead of relying solely on the fitness value, the algorithm penalizes individuals that fail to satisfy certain clauses. This is achieved by computing a Lagrangian fitness for each solution using the following equation:

$$\text{Lagrangian\_Fitness} = \text{Satisfied\_Clauses} - \lambda_t \cdot \text{Penalty} \qquad (1)$$

Here, the penalty represents the number of unsatisfied clauses, and λ_t is a multiplier that controls the impact of these violations. During the evolutionary process, λ_t is updated dynamically depending on the proportion of constraint violations in the population. If too many individuals violate constraints, the penalty increases; if most are feasible, the penalty decreases.

This mechanism allows the algorithm to explore the solution space while gradually guiding the population toward feasible and high-quality solutions. By balancing fitness and constraint handling, Lagrangian relaxation prevents the search from getting stuck in infeasible regions and ensures steady progress toward the optimal solution.

4 RESULTS

The experimental evaluation was conducted using the la04-567-0696.wcnf instance from the Job-Shop benchmark set, which consists of **35,772 variables** and **359,683 clauses**. A total of 100 independent runs of the Improved Binary Genetic Algorithm (IBGA) were executed, with Optuna hyperparameter optimization guiding the search for the best parameter configurations. Throughout the optimization process, the fitness progression exhibited natural fluctuations as diverse parameter combinations were explored. While some early trials produced varying objective values, the optimization quickly stabilized, consistently maintaining a high number of satisfied clauses. This behavior demonstrated the robustness of the algorithm and highlighted Optuna's effectiveness in balancing exploration and exploitation during the search.

As the optimization progressed, the impact of specific parameters on fitness became evident. The scatter plot examining mutation probability confirmed that moderate mutation values, particularly between 0.1 and 0.15, tend to produce better outcomes. Extremely low or excessively high mutation rates led to suboptimal solutions, highlighting the importance of carefully tuning this parameter to maintain population diversity without destabilizing convergence.

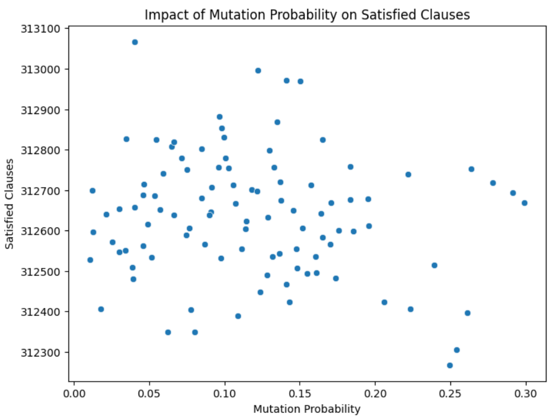

Fig. 2. Mutation Probability

To further understand the algorithm's convergence behavior, the empirical cumulative distribution function (CDF) of satisfied clauses was analyzed. The CDF showed a smooth and steep rise, indicating that the majority of runs clustered around high fitness values. More than 80% of the trials achieved fitness levels exceeding 312600 satisfied clauses, confirming the method's reliability and consistent convergence towards quality solutions.

The relationship between population size and mutation probability was also visualized through a density plot. The analysis revealed that successful trials frequently occurred when population sizes ranged between 50 and 60, paired with mutation probabilities around 0.08 to 0.15. This dense region of parameter combinations proved most effective in balancing the algorithm's search capabilities and convergence speed. Comparative analysis between Lagrangian Ranking and Classic Stochastic Ranking using boxplots provided additional insights. Both methods achieved similar median fitness levels; however, Lagrangian Ranking offered reduced variance, yielding more stable results. In contrast, Classic Stochastic Ranking occasionally produced better best-case fitness but at the cost of increased variability. This observation reflects a clear trade-off between robustness and peak performance potential.

The experimental process concluded with Optuna identifying the best-performing parameter configuration: a population size of 38, crossover

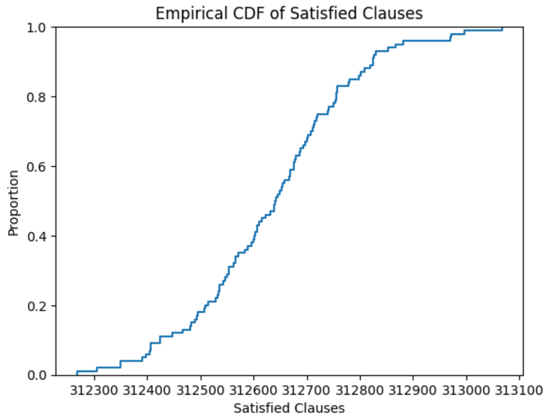

Fig. 3. CDF of Satisfied Clauses

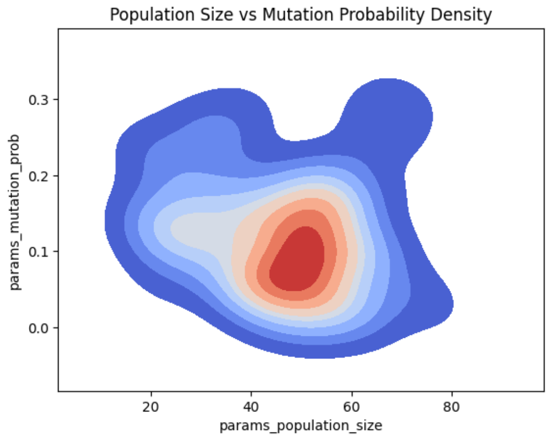

Fig. 4. Population Size vs. Mutation Probability Density

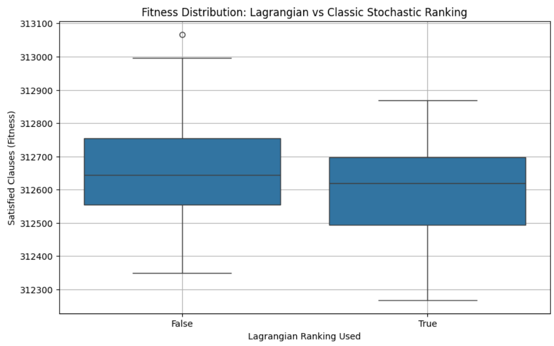

Fig. 5. Population Size vs. Mutation Probability Density

probability of 0.8725, mutation probability of 0.0403, and Lagrangian Ranking disabled. This configuration achieved the highest recorded fitness of 313067 satisfied clauses, representing the most optimal balance between exploration, exploitation, and solution quality observed during the campaign. This result corresponds to approximately **87.04%** of the total clauses satisfied, demonstrating the algorithm's ability to produce highly feasible solutions while maintaining strong optimization performance.

In summary, the combined analyses confirm that the IBGA, guided by Optuna, reliably identifies high-quality solutions for the MAXSAT

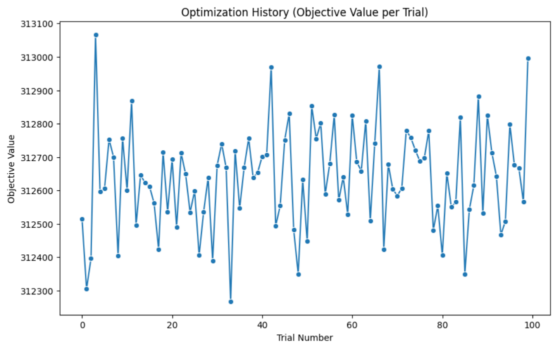

Fig. 6. Optimization History

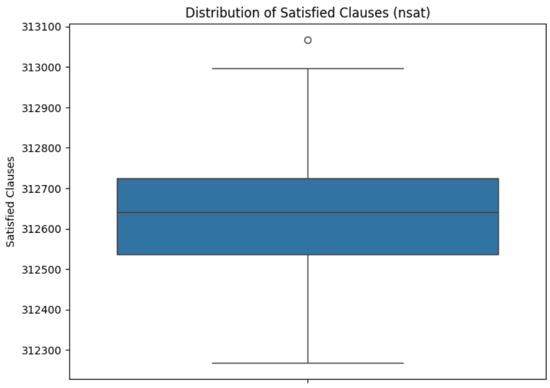

Fig. 7. Satisfied Clauses

problem. The algorithm demonstrates consistent convergence behavior, effectively navigates parameter trade-offs, and balances stability with the potential for discovering peak solutions.

5 CONCLUSION

The Improved Binary Genetic Algorithm provides an effective framework for solving the MAXSAT problem by evolving a population of candidate solutions. The integration of stochastic ranking, adaptive Lagrangian relaxation, and explicit elite size control ensures constraint handling and solution quality.

Optuna's automated hyperparameter optimization enhanced the experimental design, enabling dynamic exploration of parameter combinations rather than relying on fixed settings. This allowed us to maximize satisfied (correct) clauses efficiently. The combination of IBGA and Optuna offers a robust method for tackling MAXSAT instances, validated by experimental results and visualized through boxplots.

Our experimental results confirm that parameter tuning—especially population size, mutation, crossover, and ranking strategy—plays a crucial role in the algorithm's performance.

REFERENCES

[1]**Chu, P. C., & Beasley, J. E.** (1998). *Constraint Handling in Genetic Algorithms: The Set Partitioning Problem. Journal of Heuristics, 11*(4), 323–357

[2]**Runarsson, T. P., & Yao, X.** (2000). *Stochastic Ranking for Constrained Evolutionary Optimization. IEEE Transactions on Evolutionary Computation, 4*(3), 284–294