Nonogram Puzzle Solution Using SA

Mohid Arshad, Mohammad Hasnain, Mohammad Umar
School of Electrical Engineering and Computer Science, NUST, Islamabad, Pakistan
{marshad.bsds23seecs, mhasnain.bsds23seecs, mraza.bsds23seecs}@seecs.edu.pk

***Abstract*—Nonogram puzzles, also known as Picross or Griddlers, require filling a grid based on row and column constraints. This paper presents a solution using the Simulated Annealing (SA) algorithm, a probabilistic optimization technique inspired by metallurgy. SA iteratively improves a randomly generated initial solution by flipping grid cells and accepting changes based on a temperature-dependent probability. The paper discusses the methodology, implementation, and results, highlighting the algorithm's effectiveness in solving puzzles of varying difficulty. Results show that SA efficiently solves moderate to large puzzles, making it a practical solution for Nonogram puzzles.**

I. Introduction

A. Background and Introduction

Nonogram puzzles involve a grid with numbers indicating consecutive filled cells in rows and columns. Solving these puzzles requires determining the correct configuration of filled and empty cells based on these constraints. While solving Nonogram puzzles can be challenging for large grids, computational techniques like Simulated Annealing (SA) offer an efficient approach. SA stands out due to its ability to escape local optima and find near-optimal solutions within reasonable time limits.

B. Problem Statement

This paper focuses on solving Nonogram puzzles using the Simulated Annealing algorithm, aiming to find an optimal grid configuration that satisfies both row and column constraints, especially for larger puzzles where traditional methods may be ineffective.

C. Research Objectives

The goals of this research are:

- To explore Simulated Annealing for solving Nonogram puzzles.
- To evaluate SA's performance on puzzles of varying sizes.
- To provide visual representations of solutions.

D. Contributions

This paper:

- Introduces SA as a solution for Nonogram puzzles.
- Proposes an efficient implementation for different puzzle sizes.
- Analyzes SA's performance through experiments.

E. Paper Organization

Section II reviews related literature. Section III describes the system model and mathematical formulation. Section IV explains the SA algorithm and its implementation. Section V presents results and analysis. Section VI concludes the paper.

II. Literature Review

Nonogram puzzle-solving has shifted from exact algorithms like backtracking to heuristic methods such as Genetic Algorithms (GA) and Simulated Annealing (SA). Backtracking guarantees optimal solutions but is inefficient for large puzzles due to its exhaustive search. In contrast, Simulated Annealing (SA) is a probabilistic technique that iteratively refines solutions, making it more efficient for solving complex problems like Nonogram puzzles.

Recent studies, including those by Lee et al. [**?**] and Zhang et al. [**?**], have shown SA's effectiveness in solving small-scale Nonogram puzzles, outperforming traditional methods like backtracking. However, these studies typically focus on static or small puzzles. Few have explored SA's performance on larger, dynamic configurations, which require more adaptive algorithms. Furthermore, SA's behavior can vary significantly based on the choice of parameters, such as temperature schedules and cooling rates. This paper extends previous work by evaluating SA on a broader range of puzzle sizes and complexities, providing a more comprehensive assessment of its scalability and performance.

III. System Design and Model

The system design is based on the Simulated Annealing algorithm, which operates as a probabilistic optimization method. The goal is to iteratively refine an initial random solution by flipping grid cells and evaluating the fitness of the resulting configuration. The system is initialized with a random grid, and the fitness is calculated by comparing the current state of the grid with the given row and column constraints.

The fitness function is defined as:

$$\text{Fitness}(S) = \sum_{r=1}^{R} \text{fitness}(row_r) + \sum_{c=1}^{C} \text{fitness}(col_c)$$

where S is the current grid state, and fitness refers to how well the grid satisfies the row and column constraints. The algorithm accepts new solutions based on a probability that decreases over time, as the temperature decreases in the cooling schedule.

IV. Validation of Constraints

Before applying the Simulated Annealing (SA) algorithm, it is important to ensure that the provided row and column constraints are valid and consistent. The following validation checks are implemented:

- **Constraint Length Check**: Ensures that the total length of filled blocks (including required spaces between blocks) does not exceed the dimensions of the grid for both rows and columns.

- **Matching Constraints Check**: Verifies that the total number of filled cells across the rows matches the total number across the columns, ensuring consistency between the row and column constraints.

V. Proposed Methodology

A. Techniques and Algorithms

Simulated Annealing is an iterative optimization technique inspired by the process of physical annealing. It starts with a high temperature, which allows the algorithm to explore various configurations, including worse solutions, to escape local optima. As the temperature cools, the algorithm becomes more selective and converges toward an optimal solution.

For Nonogram puzzle-solving, the algorithm iteratively flips cells in the grid, evaluates the fitness of the new configuration, and accepts or rejects the new state based on a probability function. The temperature gradually decreases, reducing the acceptance probability for worse solutions, until a satisfactory solution is found.

B. Architecture Diagram

The system architecture follows a simple structure:

- **Initialization**: Generate a random initial grid.
- **Fitness Evaluation**: Calculate fitness based on constraints.
- **Annealing Process**: Refine the solution by iteratively flipping cells and adjusting based on the cooling schedule.
- **Termination**: The algorithm terminates when a solution meets the constraints or a maximum iteration limit is reached.

C. Process Flow

The process flow of the system can be summarized as follows:

- Step 1: Generate a random initial solution (grid).
- Step 2: Evaluate the fitness of the solution.
- Step 3: Randomly flip a cell and assess the new configuration.
- Step 4: Accept or reject the new configuration based on the fitness and temperature.
- Step 5: Repeat the process until the solution meets the constraints or the maximum number of iterations is reached.

VI. Implementation and Results

A. Implementation Details

The solution was implemented using Python and its standard libraries, including ‘random‘ for random grid initialization and ‘math‘ for probability calculations. The system was executed on a standard laptop with Python 3.x installed.

B. Experimental Setup

The Nonogram solver was tested on a variety of puzzles with different sizes and constraints. Each experiment was run multiple times to gather data on the average time to reach a solution and the success rate of finding a valid configuration.

C. Results

The results indicate that Simulated Annealing is highly effective for solving Nonogram puzzles, especially for medium-sized puzzles. The algorithm consistently converged to a valid solution within a reasonable time frame.

D. Analysis

While the Simulated Annealing algorithm performed well for most puzzle configurations, its performance is sensitive to the cooling schedule and the temperature parameter. For more complex puzzles, fine-tuning these parameters could improve solution times and accuracy.

VII. Conclusion and Future Work

A. Conclusion

This paper presents a solution to Nonogram puzzles using the Simulated Annealing algorithm. The method effectively solves puzzles by iteratively refining a random solution based on fitness evaluations. Experimental results demonstrate that SA is a promising approach for solving Nonogram puzzles.

B. Future Work

Future work could explore hybrid algorithms that combine Simulated Annealing with other optimization techniques, such as Genetic Algorithms, to improve performance. Additionally, further studies could focus on dynamically adjusting the cooling schedule based on the puzzle's difficulty.

References

[1] M. Hatem, E. Burns, and W. Ruml, “Solving large problems with heuristic search: General-purpose parallel external-memory search,” *Journal of Artificial Intelligence Research*, vol. 62, pp. 233–268, 2018. [Online]. Available: https://findscholars.unh.edu/display/publication447189

[2] W.-L. Wang and M.-H. Tang, “Simulated annealing approach to solve nonogram puzzles with multiple solutions,” *Procedia Computer Science*, vol. 36, no. 1, pp. 541–548, 2014. [Online]. Available: https://www.researchgate.net/publication/275541594_Simulated_Annealing_Approach_to_Solve_Nonogram_Puzzles_with_Multiple_Solutions

[3] A. E. Iordan, “A comparative study of the A* heuristic search algorithm used to solve efficiently a puzzle game,” *IOP Conference Series: Materials Science and Engineering*, vol. 294, no. 1, p. 012049, 2018. [Online]. Available: https://www.researchgate.net/publication/322453869_A_comparative_study_of_the_A_heuristic_search_algorithm_used_to_solve_efficiently_a_puzzle_game

[4] D. O. Hasan *et al.*, “The Fifteen Puzzle—A new approach through hybridizing three heuristics methods,” *Computers*, vol. 12, no. 1, p. 11, 2023. [Online]. Available: https://www.mdpi.com/2073-431X/12/1/11

[5] S. Markaki and C. Papaodysseus, “Jigsaw puzzle solving techniques and applications: A survey,” *The Visual Computer*, vol. 39, pp. 4405–4421, 2023. [Online]. Available: https://colab.ws/articles/10.1007/s00371-022-02598-9

[6] B. Suman and P. Kumar, “A survey of simulated annealing as a tool for single and multiobjective optimization,” *Journal of the Operational Research Society*, vol. 57, no. 10, pp. 1143–1160, 2006. [Online]. Available: https://sci2s.ugr.es/sites/default/files/files/Teaching/GraduatesCourses/Metaheuristicas/Bibliography/SA-review_paper-2006.pdf