**Sudoku Solvers' Analysis**

Prepared for

Dr. Athanasios Aris Panagopoulos

CSCI 191T - Bio-Inspired Machine Learning

California State University, Fresno

Prepared by

**Super Unsupervised Learners**

Nav Sanya Anand

Enrique Verduzco

Clay Freitas

Kamarin San Nicolas

Ryan Perez

[Github Link](https://github.com/EnriqueVerduzco/ML-Sudoku-Analysis)

4 May 2021

The ML-Sudoku-Analysis project consists of solving for a 9x9 sudoku game through 3 strategies: Satisfiability, Simulated Annealing, and an Evolutionary Algorithm.

The satisfiability algorithm was something that the students have done in a previous course (Computer Science 60 - Foundation of Computer Science). To refresh the team’s memory Dr. Thanos provided notes for reference (Panagopoulos). There were multiple clauses that had to be considered. After implementing all the clauses the team struggled to combine them for each cell. Initially the python library NumPy (Python, R) was considered but the team couldn’t figure out the best way to solve the said problem. The team ended up using the pycosat library which made it easier to combine all the clauses for each cell in the sudoku code (pycosat). The next problem faced was that pycosat was not able to compile using VSCode and Python 3.9. To solve this problem the team utilized Anaconda3 and Jupyter Notebook which is why you will see the two versions of the Satisfiability code, a Python version and a Jupyter Notebook version.

Within the Simulated Annealing Algorithm temperature plays a crucial role in controlling the evolution and progress of the system, this temperature is decreased according to a cooling schedule (Panagopoulos). We implemented the initial temperature to be very high at first but found this to be very time consuming when using the default cooling schedule from the Simulated Annealing python library (Matthew Perry). When using a smaller initial temperature we found that the algorithm was able to process a puzzle faster yet was unable to solve the puzzle due to the limitation set by the iterations or steps we would allow the algorithm to be able to use. We ended up creating time analysis mainly based on the different number of steps used for each puzzle because of the large factor of randomness of this algorithm. This allowed us to view how much the varying number of steps affected the temperature values and cooling schedules values efficiency on solving each puzzle.

The evolutionary algorithmwas known to be successful in solving combinatorial problems (Behmanesh R et al. 2021). In the case of the sudoku solver we were expecting rapid accurate results. However, due to the random nature of the algorithm we encountered occasional infinite loops within a population set. We experimented by adjusting different variables in order to find an optimal solution. In the end this algorithm's randomness was reflected in the result of each puzzle, producing a pattern. The greatest variable affecting the time to a solution is the accuracy of the initial randomly produced population relative to the end solution. In order to improve our results we implemented a row checking function and increased the fitness value to 243. This addition decreases the overall time it takes to complete a solution yet increases the time between each generational mutation. Combining this with a shortening of the generational resets tends to return a faster result in the overall time.

Since it is observed that satisfiability was the best possible solution when it comes to time complexity, some of the future goals can be to expedite the evolutionary algorithm and simulated annealing to make them better. One of the possible ways is through Tabu Search. Simulated annealing could be made better by not only testing a larger variety of initial and final temperatures but also a variety of cooling schedules or even a simulated annealing algorithm without cooling schedules (Walid Mahdi et al. 2017). We could also solve the sudoku problem through other metaheuristics like Ant Colony Optimization, Particle swarm optimization etc.

Work Cited

Behmanesh, R., Rahimi, I. & Gandomi, A.H. (2021). Evolutionary Many-Objective Algorithms

for Combinatorial Optimization Problems: A Comparative Study. *Arch Computat*

*Methods Eng* 28, 673–688 Retrieved 3 May 2021, from

<https://doi.org/10.1007/s11831-020-09415-3>

Python, R. (2021). NumPy Tutorial: Your First Steps Into Data Science in Python – Real Python. Retrieved 3 May 2021, from <https://realpython.com/numpy-tutorial/#:~:text=NumPy%20is%20a%20>

pycosat. (2017). Retrieved 3 May 2021, from <https://pypi.org/project/pycosat/>

Matthew Perry. (2021). Retrieved 3 May 2021, from <https://github.com/perrygeo/simanneal>

Walid Mahdi, Seyyid Ahmed Medjahed, & Mohammed Ouali. (2021). Performance Analysis of Simulated Annealing Cooling Schedules in the Context of Dense Image Matching. Retrieved 3 May 2021, from <http://www.scielo.org.mx/pdf/cys/v21n3/1405-5546-cys-21-03-00493.pdf>

Panagopoulos, Athanasios Aris Computer Science 191T - Bio-Inspired Machine Learning, California State University-Fresno from [apanagopoulos@mail.fresnostate.edu](mailto:apanagopoulos@mail.fresnostate.edu)