**Solving Nonograms using AI**

Project Plan

Hozyfa Mohammed Khair

CS3821 – BSc Final Year Project

# **Abstract:**

Nonograms, a popular puzzle game, challenge the player to draw or reveal an image by satisfying the numerical constraints within rows and columns. These constraints guide players to determine which blocks should be filled and which should remain blank. Human solvers typically start by looking at the row or column with the highest number, which simplifies the puzzle solving process by revealing key information for other blocks.

The primary objective of the project is to develop an artificial intelligence (AI) system capable of solving nonograms efficiently and accurately. Additionally, there is a secondary objective of creating a software tool that will allow users to solve nonograms by providing them with step-by-step hints or instructions.

Nonograms are a great example of constraint-satisfaction problems (CSP), each block has a domain of potential values - filled, empty, and unknown [1]. The constraints would be the constraints on the rows and columns that the block belongs to. Making constraint satisfaction key part of developing an AI to accurately solve the nonograms.

I aim to research and employ AI algorithms such as constraint satisfaction and back tracking. Efficiency can be improved using other techniques such as: forward checking – pruning invalid branch solutions, and dynamic variable ordering – ordering variables to improve efficiency, i.e., by branches with the lowest number of possible solutions.

Furthermore, I aim to create a software tool to allow users to easily use the AI. The user will have the ability to input their own nonogram puzzles while the AI checks if there is a solution for them. The software will be able to offer hints and step-by-step solutions to the user to assist them in solving the nonograms.

Finally, I intend to integrate machine vision into my project, allowing the AI to recognise and interpret images or scanned documents. The user will also be able to use this feature most likely through uploading a scanned document. This will allow me to learn more about image, character, and pattern recognition.

# **Timeline:**

**Term 1:**

**Week 1-2 (October 1 - October 15, 2023)**

* Research nonogram-solving algorithms and AI techniques.
  + Backtracking and constraint satisfaction problems
* Set up development environment. \/
* Implement a solution for the 8-queens problem using backtracking. \/

**Week 3-4 (October 16 - October 31, 2023)**

* Develop a simple GUI to display nonogram puzzles.
* Adapt the backtracking algorithm for solving a simple nonogram. V
* Research constraint satisfaction techniques. V
* Implement constraint satisfaction for nonograms. V

**Week 5-6 (November 1 - November 15, 2023)**

* Research strategies to improve efficiency and effectiveness.
  + i.e., forward checking, dynamic variable ordering
* Continue improving the constraint satisfaction solver.
* Implement puzzle generation.

**Week 7-8 (November 16 - November 30, 2023)**

* Enhance the GUI to handle user input and puzzle display.
* Integrate solving capabilities into the user interface.
* Test the system on various nonogram puzzles.
* Prepare for interim report.

**Week 9-10 (December 1 - December 15, 2023)**

* Optimize algorithms for solving and generating nonograms.
* Research more CSP techniques and other algorithms.

**Term 2:**

**Week 1-2 (January 1 - January 15, 2024)**

* Begin machine vision research.
* Select libraries or frameworks to use.
* Start implementing machine vision.

**Week 3-4 (January 16 - January 31, 2024)**

* Finish implementing using machine vision to enter puzzles.
* Integrate with solver.

**Week 5-6 (February 1 - February 15, 2024)**

* Research and implement using recursion to solve nonograms.
* Optimise recursion algorithm.

**Week 7-8 (February 16 - February 29, 2024)**

* Add user-friendly features like hints and step-by-step solutions.
* Implement printing, support for different screen sizes, and timers.

**Week 9-10 (March 1 - March 14, 2024)**

* Conduct testing and performance checks.
* Address any usability or functionality issues.
* Perform benchmarking and performance tests.

**Week 11 (March 15 - March 22, 2024)**

* Perform a final review of the project, documentation, and reports.
* Make any last-minute adjustments or improvements.
* Submit the project on or before the due date (March 22, 2024).

# **Risk Assessment:**

The risks associated with this project are skewed toward an incorrect estimation of difficulty, as many ambitious projects are. However, in this section I will address the possible risks and mitigations for the project.

Hardware and Technical Failures:

One risk for all projects is an accidental loss of project data, but this is easily mitigated by frequent commits to GitLab. I will also upload my reports to GitLab, but I am considering using version control for reports such as OverLeaf.

A more specific hardware risk for my project is with regards to the hardware used to capture images for machine vision. Therefore, it would be best if my program simply parses a commonly used image format such as png. This would allow machine vision to still be usable while not having to worry about needing a dedicated camera.

Time Management and Task Estimation:

Arguably the biggest risk for my project, especially considering machine vision. I have attempted to thoroughly plan out my timeline so I will easily be able to see when I am off schedule. There is also a decent amount of buffer time to allow me to ensure everything stays up to speed.

Balancing Code and Documentation:

The report is a crucial part of the project, and I must ensure that the report does not lag behind the code. I plan to work on the report throughout all the weeks especially when completing a milestone, this along with the final week for finalizing the report I will be able to ensure that my report doesn’t fall behind.

Poor Efficiency of Solving:

If the speed at which nonograms are solved is too slow, it will make the program tedious to use and potentially lock the user into long wait times. To avoid this, I have added time to research constraint-satisfaction-problem strategies to make the solver more efficient. I may also have to add a grid limit, but what this limit will be exactly is yet to be decided.

Machine Vision Risks:

Machine vision is quite a complex feature to add to my project and comes with many risks.

Image recognition and preprocessing may encounter difficulties with changes in image quality, lighting, and contrast. There will have to be extensive testing for different inputs. I can mitigate these risks by using established libraries and frameworks for image analysis.

Puzzle generation:

Puzzles generated should have one unique solution. With nonograms its usually more fun for the user if the nonogram reveals an image of some kind, but this can be difficult to implement with the puzzle generator. If I have time remaining, I can attempt to improve the generator to integrate patterns rather than randomly spacing the constraints around.

# **Bibliography:**

1. Yu, CH., Lee, HL. & Chen, LH. An efficient algorithm for solving nonograms. Appl Intell 35, 18–31 (2011). https://doi.org/10.1007/s10489-009-0200-0
2. Dandurand, F., Cousineau, D., & Shultz, T. R. (Year). Solving nonogram puzzles by reinforcement learning.
3. . -C. Wu et al., "An Efficient Approach to Solving Nonograms," in IEEE Transactions on Computational Intelligence and AI in Games, vol. 5, no. 3, pp. 251-264, Sept. 2013, doi: 10.1109/TCIAIG.2013.2251884.
4. <http://ktiml.mff.cuni.cz/~bartak/constraints/intro.html>
5. <https://chihyulai.com/nonogram-ai/>
6. Brailsford, S. C., Potts, C. N., & Smith, B. M. (1999). Constraint satisfaction problems: Algorithms and applications. European Journal of Operational Research, 119(3), 557-581. <https://doi.org/10.1016/S0377-2217(98)00364-6>
7. Dechter, R., & Frost, D. (1999). *Backtracking algorithms for constraint satisfaction problems* (Vol. 56). Technical Report.
8. Batenburg, K. J., & Kosters, W. A. (2012). On the difficulty of Nonograms. *ICGA journal*, *35*(4), 195-205.