**N-Queen Problem  
CS7IS2 Project (2021-2022)**

Garvit Vijai(20303251), Rohan Girotra(20304712), Saubhagya Sharma, Manu()

[vijaig@tcd.ie](mailto:vijaig@tcd.ie), [girotrar@tcd.ie](mailto:girotrar@tcd.ie),

**Abstract.** We focussed on Constraint Satisfaction problems and various algorithms to solve them. We used three different artificial intelligence algorithms namely: backtracking algorithm, hill-climbing algorithm, and genetic algorithm to solve the N-Queens problem, which is a type of constraint satisfaction problem. Each algorithm goes through examining various arrangements of the queens until the solution is found. We study the performance, speed, and nature of problem-solving of each of these algorithms.

Keywords: N-Queen, hill-climbing, backtracking, genetic algorithm

**1 Introduction**

We addressed Constraint Satisfaction Problems(CSPs) in our project. The constraint satisfaction is the process of finding a solution through a set of constraints that impose conditions that the variables must satisfy. The technique used in solving CSP depends on the kind of constraints being considered.

We have set of variables X, set of domain D and set of constraints C.

Values assigned to variables in X must be from Domain in D.

A constraint restricts the possible values of a subset of variables from X.

We have chosen to implement N-Queens problem using three different algorithms.

The goal of N - Queens is to arrange N - queens on a N x N chessboard so that no queen attacks another by being in the same column, row, or diagonal.

We have implemented following three algorithms in our implementation:

1. Backtracking - Backtracking algorithm is a problem solving algorithm that uses a brute force approach for finding the desired output. It incrementally builds candidates to the solutions and abandons a candidate as soon as it determines that the candidate will not lead to a valid solution.
2. Hill Climbing - Hill Climbing is a heuristic local search algorithm.
3. Genetic Algorithm

In modern times, it is often used as a demonstration problem for various computer programming methods. Historically, the problem has its roots in the case n=8, which was discussed in the 19th century. It was posed by Max Bezzel in 1848, and Carl Friedrich Gauss dealt with it as well. It is known that there are solutions for all natural numbers n, with the exception of n = 2 and n = 3. Although the exact number of solutions is only known for n \* 27, this number grows at an asymptotic rate (0.143 n)^n. As an illustration of structured programming, Edsger Dijkstra used this problem to describe a depth-first backtracking algorithm in 1972.

in this section, you should introduce your work: what are the motivations behind this work? What is the relevant problem that you are investigating? Why is it relevant?

Briefly, introduce the background information required to understand the problem and the concepts that you will develop.

In This section should also contain the link to the recording of your presentation (college OneDrive link – please make sure sharing permissions are such that everyone with tcd email can access it)

**2 Related Work**

In this section you will discuss all possible categories of approaches to solve the problem you are addressing, justifying your choice of the 3 you have selected to evaluate.

Also, briefly introduce the approaches you are evaluating with a specific emphasis on differences and similarities to the proposed approach(es).

**3 Problem Definition and Algorithm**

This section formalises the problem you are addressing and the models used to solve it. This section should provide a technical discussion of the chosen/implemented algorithms. A pseudocode description of the algorithm(s) can also be beneficial to a clear explanation. It is also possible to provide one example that clarifies the way an algorithm works. It is important to highlight in this section the possible parameters involved in the model and their impact, as well as all the implementation choices that can impact the algorithm.

**4 Experimental Results**

This section should provide the details of the evaluation. Specifically:

* Methodology: describe the evaluation criteria, the data used during the evaluation, and the methodology followed to perform the evaluation.
* Results: present the results of the experimental evaluation. Graphical data and tables are two common ways to present the results. Also, a comparison with a baseline should be provided.
* Discussion: discuss the implication of the results of the proposed algorithms/models. What are the weakness/strengths of the method(s) compared with the other methods/baseline?

**5 Conclusions**

Provide a final discussion of the main results and conclusions of the report. Comment on the lesson learnt and possible improvements.

A standard and well formatted bibliography of papers cited in the report. For example:

**References**

1. Baldonado, M., Chang, C.-C.K., Gravano, L., Paepcke, A.: The Stanford Digital Library Metadata Architecture. Int. J. Digit. Libr. 1 (1997) 108–121

2. Bruce, K.B., Cardelli, L., Pierce, B.C.: Comparing Object Encodings. In: Abadi, M., Ito, T. (eds.): Theoretical Aspects of Computer Software. Lecture Notes in Computer Science, Vol. 1281. Springer-Verlag, Berlin Heidelberg New York (1997) 415–438

3. van Leeuwen, J. (ed.): Computer Science Today. Recent Trends and Developments. Lecture Notes in Computer Science, Vol. 1000. Springer-Verlag, Berlin Heidelberg New York (1995)

4. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. 3rd edn. Springer-Verlag, Berlin Heidelberg New York (1996)