**Constraint Solving Algorithm**

**Description:** A CSP (Constraint Satisfaction Problem) Algorithm, that will take variables, domains (labels) and constraints as input (in a binary form using logical operators) and will generate a solution that satisfies all constraints. This algorithm can have multiple applications, which will be showcased as follows: Graph Coloring (given a graph and a number of colors, find a way to color it so that no connected nodes have the same color), Sudoku solving, n-Queen Problem (having n queens on a chess board such that there are no 2 that attack each other), Radio Frequency Assignation (assigning frequencies to different radio stations in the same area such that they don't interfere) and Timetable Scheduling (considering the time courses run during the week and their prerequisites and allowing the user to select which courses they want to take, the algorithm will come up with an efficient solution such that they finish their courses as fast as possible and all constraints are maintained).

**Competitive Analysis:** There are multiple implementations of several algorithms CSP that focus on one application, yet very few offer a visualization and a diversity in terms of the applications of a CSP algorithm. The main goal of this project is showcasing the multiple benefits and applications one AI algorithm has, and how it can be used in multiple real-life scenarios. Furthermore, this project will showcase the difference between traditional backtracking method (the brute force approach) compared to a more efficient way.

**Structural Plan:** I will be building a library for the general CSP program (which will be called AI.py or something similar) and a .py file for each application (hence, nQueens.py, Sudoku.py, TimeTabling,py etc). Another application that will be further developed through this project is implementing an algorithm to read a university or high-school timetable (in an Excel format) and the constraints (prerequisites, corequisites, classes, professors, timing conflicts) and offer solutions for a student who wants to take a list of courses ordered by preference.

**Algorithmic Plan:** I will be implementing pruning algorithm through AC3 and a heuristic approach to backtracking. I will be showcasing the advantages of turning problems into ARC consistent problems.

**Timeline Plan:** By the end of October, all standard features will be implemented (enabling the functionality of the project). Further I will develop the graphical user interface and focus on the user experience.

**Version Control Plan: Graphical user interface, application, Teams

Description automatically generated**

**Module List: No module list as of right now**