**Project Synopsis**

**Title** : Study of a Constraint Satisfaction Problem

**Abstract/ Project definition :**

Constraint satisfaction problems (CSPs) are mathematical questions defined as a set of objects whose state must satisfy a number of constraints or limitations. CSPs represent the entities in a problem as a homogeneous collection of finite constraints over variables, which is solved by constraint satisfaction methods. CSPs are the subject of research in both artificial intelligence and operations research, since the regularity in their formulation provides a common basis to analyze and solve problems of many seemingly unrelated families.

In this project we study and solve the N-Queen constraint satisfaction problem. The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. Queens can attack horizontally ,diagonally(both diagonals) and vertically.

**Literature review :**

The n-queens problem is to place n non attacking queens on an n×n board. This is a generalization of the problem of putting eight non attacking queens on a chessboard, which was first posed in 1848 by M. Bezel, a German chess player, in the *Berliner Schachzeitung*. The earliest paper on the general n-queens problem is F.J.E. Lionnet’s 1869 work . There were no solutions to the original 8-queens problem by Bezzel himself, but two non attacking board configurations were published as partial answers to his problem in the next edition of that serial. Starting in 1850, the 8-queens problem was then studied by C.F. Gauss, for example in his letters to Schumacher. According to Ahrens, the first to solve the problem by finding all 92 solutions was Nauck, in 1850; Gauss later claimed this was the total number of solutions, saying that it is possible in principle to use brute force computation to show this, and that 92 is indeed the total number of solutions was shown by Pauls in.

Gauss is often cited as the originator of this problem or the first to solve it, but this is almost certainly a case of “broken telephone”, which P. J. Campbell notes in a paper about this historical error. Indeed Bezzel seems to have come up with the problem before Gauss, and Gauss was not the first to solve the problem, having found only 72 of the 92 total solutions given by Nauck, and proved complete by Pauls. (Sprague and É. Lucas in the section “Quatrième récréation” both give excellent summaries of the work that has been done by many mathematicians on the original 8-queens problem; Sprague also gives the total number of solutions for n=4,…,11, respectively 2, 10, 4, 40, 92, 352, 724, 2680, and the number of fundamental solutions, those solutions invariant under the symmetry group of the square, respectively 1, 2, 1, 6, 12, 46, 92, 341, for these board sizes.

The n-queens problem is often studied as a “mathematical recreation”, but there are several applications: parallel memory storage schemes, VLSI testing, traffic control and deadlock prevention. introduce a memory storage scheme for conflict free access for parallel memory systems using n-queens solutions.We can use modular n-queens solutions to study reconfigurable meshes with buses (RMB). The modular n-queens problem is equivalent to finding “valid” periodic skewing schemes for parallel memories .

**Expected outcome** : Given a n\*n chessboard finding a possible combination of positions of queens

**Timeline :**

**PO/PSO mapping :**

**PROGRAM OUTCOMES (POs)**

**PO1: Engineering Knowledge:** Apply the knowledge of Mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

**PO2: Problem Analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

**PO3: Design/Development of Solutions:** Design solutionsfor complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

**PO4:** **Engineer and Society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

**PO5:** **Environment and Sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

**PO6:** **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

**PO7:** **Individual and Team Work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

**PO8:** **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

**PO9:** **Project Management and Finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

**PO10:** **Life-long Learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

**PROGRAM SPECIFIC OUTCOMES (PSO)**

Students will have proficiency in fundamental engineering and computing techniques and knowledge on contemporary topics like artificial intelligence, data science and distributed computing towards development of optimized algorithmic solutions.

Students will have capabilities to participate in the development of software and embedded systems through synergized teams to cater to the dynamic needs of the industry and society.

**PO & PSO MAPPING**

Example given for your Reference:

| **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **PO10** | **PSO1** | **PSO2** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **3** | **3** | **3** | **2** | **3** | **3** | **2** | **2** | **3** | **2** | **3** | **2** |

**Methodology :**

**BackTracking -**

In this project we use the popular method of backtracking to solve N Queens problem.

The idea is to place queens one by one in different rows, starting from the topmost row. When we place a queen in a row, we check for clashes with already placed queens. In the current row, if we find a column for which there is no clash, we mark this row and column as part of the solution. If we do not find such a column due to clashes then we backtrack.

1) Start in the topmost row

2) If the row is =-1 stop searching. Row will be -1 when the queen in the first row has been placed in the last column and now no new solutions are there.

3) Try to find out the NEXT APPROPRIATE column for the queen in the row.

NEXT APPROPRIATE column can be

1. First non-attacking column(including 0) if it's the first time a queen is put on this row
2. First non-attacking column after the column that was previously containing the queen.

4) If no NEXT APPROPRIATE column was found backtrack to previous row and undo to step 3

5) If NEXT APPROPRIATE column was found then place queen there

6) If this is the last Row then this is a successful case. So display it and backtrack to the previous row and undo to step 3.

7) If this isn't the last row, move forward to the next row with step 2.

**Hill Climbing -** Now backtracking will always work ,bit for very less n. Using hill climbing we will be able to find close answers most of the times for very high n. Here we try to minimize the heuristic function which will be equal to the number of unique attacking pairs.

The algorithm is as follows:

* select an initial (random or preselected) assignment of queens
* while there are threatened queens (or until we get tired of trying... it's worthwhile to put this in a for loop to limit the number of tries):
  + select a random threatened queen
  + move the selected queen to the square that minimizes conflicts

In that last step, we are assuming that each queen is constrained to her row, so she can only change columns within the row. If there are several columns that minimize conflicts for the current queen, we can choose randomly among them.

**Unique contribution / novelty :**

None

**References :**

i)<http://www.durangobill.com/N_Queens.html>

ii) <https://www.youtube.com/watch?v=xFv_Hl4B83A&t=517s>

iii) https://www.codesdope.com/blog/article/backtracking-explanation-and-n-queens-problem/