

**CSC 180-01 Artificial Intelligence Mini-Project 4: Solving N- Queens Problem using Genetic Algorithms**

**Due at 2 pm, Friday , November 20, 2020**

BY

Perry Gill

Rajvee Modi

Mary Ballesteros

**Problem Statement**

In this project, we practiced genetic algorithms by using Distributed Evolutionary Algorithms in Python(DEAP). This project aims to use DEAP in order to solve the 8 queens puzzle. The 8 queens puzzle is a problem which places eight queens on an 8 x 8 chessboard in order for no two queens to attack each other. The eight queens puzzle is another example of the more general N queens problem of placing n non-attacking queens on an n x n chessboard. With the exception of n = 2 and n = 3, solutions exist for all natural numbers.

We determined the feasible outcome of eight queens on a standard 8 x 8 chessboard by implementing evaFitness function in order to calculate the number of distinct attacks. In order to solve the 8 queens puzzle, we used position-index-based board representation and row-index-based board representation.

**Methodology**

The queen in chess, is the only piece that can attack in any given direction. We created this puzzle in order to remove the queens that are attacking each other. The queen can travel vertically, diagonally, or horizontally. Therefore, an attack can happen when a queen is positioned in the same horizontal, vertical, or diagonal line of that queen.

We implemented the genetic operators mate(), mutate(), evaluate() and select() in the toolbox. The given operators transform or select individuals. Such as, providing two individuals to the crossover will transform the individuals in place. Toolbox can be registered in the algorithm by registering needed operators under distinct names.

When calculating attacks, we needed to configure the distinct attacks. Allowing our evaFitness function to only calculate distinct attacks. For example, if a queen had multiple attacks from horizontal, vertical, and diagonal direction, our function would count only one attack for that one queen. EvaFitness function returns the total number of duplicate queens in the same position and the total numbers of distinct pairs of queens that are attacked in any position.

In order to find the best individuals in evolution and it’s fitness value, we implemented the Hall of Fame Function.The Hall of Fame finds the best individuals that appeared during an evolution. Every generation Hall of Fame saves the best individual in a separate location. If situations where the best population disappears, it’ll be saved in a separate archive through the Hall of Fame.

**Experimental Results and Analysis**

DEAP gave us practice on building our own algorithm. DEAP provides various tools in order to solve problems such as the 8 queen puzzle problem. We determined the different outcome of eight queens on a standard 8 x 8 chessboard by implementing evaFitness function in order to calculate the number of distinct attacks. We used two techniques for representing the chessboard position based and row index based. The evaFunction was easier to write for the row index based board because we did not have to take into account the horizontal cases.

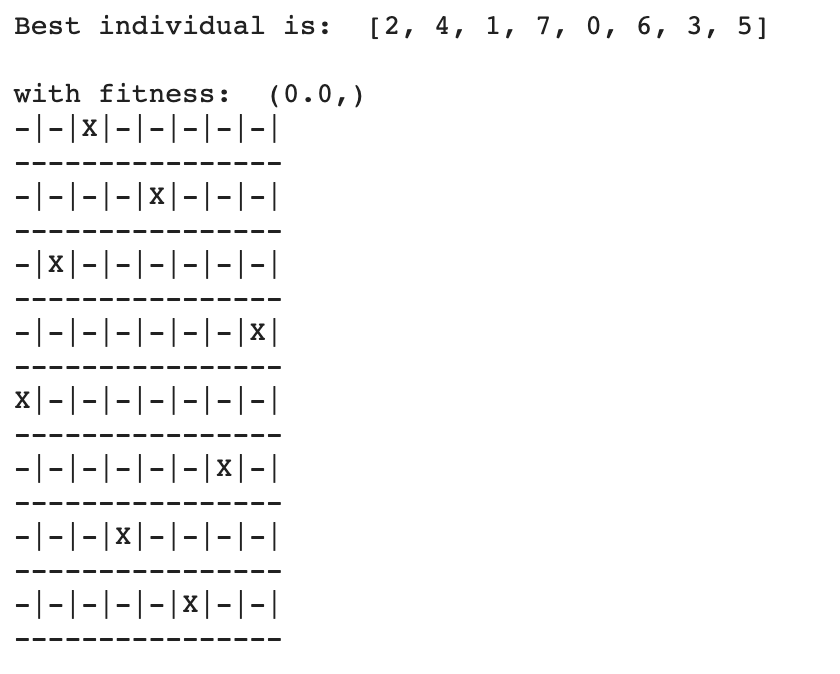
EvaFitness function returns the total number of duplicate queens in the same position and the total numbers of distinct pairs of queens that are attacked in any position. “evaFitness” function returns the fitness of any given board. By using checkDuplicate, we calculated the number of queen pairs in the same position.

The Hall of Fame finds the best individuals that appeared during an evolution. Every generation Hall of Fame saves the best individual in a separate location. In situations where the best population disappears, it’ll be saved in a separate archive through the Hall of Fame. Our best individual and fitness for our two different board representation is:

## Part I: Position-index-based board representation:

## 

## Part II: Row-index-based board representation:

****

**Task Division and Project Reflection**

**Task Division:**

Perry Gill:

1. Strategize and Implementation
2. Position Based Board
3. evaFunction
4. Write Report

Rajvee Modi:

1. Strategize and Implementation
2. Position Based Board
3. Row Index Based Board
4. Write Report

Mary Ballesteros:

1. Strategize and Implementation
2. Position Based Board
3. Row Index Based Board
4. Write Report

**Project Reflection :**

This project dove into the implementation of DEAP. It introduced us to genetic algorithms by using Distributed Evolutionary Algorithms in Python. This project used DEAP in order to solve the 8 queens puzzle. The 8 queens puzzle is a problem which places eight queens on an 8 x 8 chessboard in order for no two queens to attack each other.

Issues we ran into during this project was implementing the evaFitness function. Mutation would generate invalid boards that had a queen attacked in multiple positions. We ran issues with calculating the distinct attacks where the queen was only counted for one attack and then removed. In order to calculate the distinct attack, we had the queen removed right away in order to prevent it from being calculated for another attack.